GPU Acceleration of Nonlinear Modeling by the Discontinuous Galerkin Time-Domain Method

Huan-Ting Meng and Jian-Ming Jin

Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
meng2@illinois.edu, j-jin1@illinois.edu

Abstract ─ A discontinuous Galerkin time-domain (DGTD) algorithm is formulated and implemented to model the third-order instantaneous nonlinear effect on electromagnetic fields due the field-dependent medium permittivity. The nonlinear DGTD computation is accelerated using graphics processing units (GPUs). Two nonlinear examples are presented to show the different Kerr effects observed through the third-order nonlinearity. With the acceleration using MPI + GPU under a large cluster environment, the solution times for nonlinear simulations are significantly reduced.

Index Terms ─ Computational electromagnetics, DGTD, GPU acceleration, Kerr effect, nonlinear electromagnetics, third-order nonlinearity.

I. INTRODUCTION

Nonlinear phenomena in electromagnetics generally involve changes in the material properties due to the presence of electromagnetic fields. The changes in the material properties in turn modify the state of the original electromagnetic fields in the medium. Since the material properties and the contained fields interact with each other constantly, it is most natural to describe and model these interactions in the time domain, where at each time instant the changes in the fields induce nonlinear modifications on both the material properties and the fields themselves.

The nonlinear Kerr effect [1] is one of the most studied and exploited optical effects. It describes the third-order interaction between the electric field and the permittivity of the material, which produces a variety of nonlinear phenomena [1], [2], such as third-harmonic generation (THG), self-phase modulation (SPM), self-focusing, and frequency mixing. Much investigation has been carried out for the simulation of the nonlinear optical effects using the finite-difference time-domain (FDTD) algorithms [3], due to their straightforward implementation.

This work is focused on the modeling of the third-order Kerr instantaneous nonlinearity using the discontinuous Galerkin time-domain (DGTD) algorithm. The nonlinear DGTD algorithm possesses many advantages of the linear DGTD algorithms over nonlinear FDTD algorithms, including the flexibility in complex geometry modeling, reduced phase shifts, and the ease to achieve higher order accuracy and convergence. To speed up the computation, the MPI + GPU framework developed in [4] is adapted to accelerate the nonlinear DGTD algorithm.

II. FORMULATION

For a general third-order nonlinear medium, the relative permittivity can be written as:

$$\varepsilon_r = \varepsilon_r(E) = \varepsilon_{r,\mathrm{L}} + \varepsilon_{r,\mathrm{NL}} = \varepsilon_{r,\mathrm{L}} + \chi^{(3)}E^2, \tag{1}$$

where $\varepsilon_{r,\mathrm{L}}$ and $\varepsilon_{r,\mathrm{NL}}$ are the linear and nonlinear parts of the relative permittivity, respectively, $\chi^{(3)}$ is the third-order nonlinear polarization coefficient, and E is the magnitude of the time-varying electric field. Here we focus on the derivation of the DGTD algorithm to model a nonlinear, lossless, and non-dispersive medium to update the electric field since the updating equation for the magnetic field has no nonlinear components and thus is identical to that in a linear medium. Testing Ampere's law using the Galerkin method, substituting in the expansion of the fields, and applying the central flux, the equation after taking the time derivative on $\vec{D}$ for element e becomes:

$$\begin{aligned}[S_e]\{h\} + \iiint_{V_e}\left[\varepsilon_0 \frac{\partial \varepsilon_r}{\partial t}\vec{N}_i^e \cdot \vec{N}_j^e\right] dV\{e\} \\ + \iiint_{V_e}\left[\varepsilon_0 \varepsilon_r \vec{N}_i^e \cdot \vec{N}_j^e\right] dV \frac{\partial\{e\}}{\partial t} = [F_{eh}]\{h^+ - h\},\end{aligned} \tag{2}$$

where

$$S_e(i,j) = \iiint_{V_e} \frac{1}{\mu_\infty}(\nabla \times \vec{N}_i^e)\cdot(\nabla \times \vec{N}_j^e)\, dV, \tag{3}$$

and $\{e\}$ and $\{h\}$ are the electric and magnetic field solution vectors and $\vec{N}_i^e$ and $\vec{N}_j^e$ are vector basis functions. The terms associated with the boundary conditions are omitted for simplicity. Since the time-varying permittivity is embedded in the mass matrix of

Submitted On: August 29, 2015
Accepted On: March 30, 2016

the DGTD algorithm, the volume integration pertaining to the electric field is now split into two terms by the product rule, where for the nonlinear medium, both the relative permittivity and the electric field are functions of time. Discretizing Equation (2) in the time domain using central difference gives:

$$\iiint_{V_e}\left[\varepsilon_0\frac{\varepsilon_r^{n+1}-\varepsilon_r^n}{\Delta t}\vec{N}_i^e\cdot\vec{N}_j^e\right]dV\left(\frac{\{e\}^{n+1}+\{e\}^n}{2}\right)$$
$$+\iiint_{V_e}\left[\varepsilon_0\frac{\varepsilon_r^{n+1}+\varepsilon_r^n}{2}\vec{N}_i^e\cdot\vec{N}_j^e\right]dV\left(\frac{\{e\}^{n+1}-\{e\}^n}{\Delta t}\right) \quad (4)$$
$$=\{b\}^{n+1/2},$$

where ε_r^{n+1} is the field-dependent nonlinear permittivity at the future time step, ε_r^n is the converged permittivity at the current time step, and

$$\{b\}^{n+1/2}=[F_{eh}](\{h^+\}^{n+1/2}-\{h\}^{n+1/2})-[S_e]\{h\}^{n+1/2}. \quad (5)$$

After rearranging the terms, Equation (4) can be cast into a field-marching form as:

$$[M_e]^{n+1}\{e\}^{n+1}-[M_e]^n\{e\}^n=\{b\}^{n+1/2}, \quad (6)$$

where

$$[M_e]^{n+1}=\frac{\varepsilon_0}{\Delta t}\iiint_{V_e}\left[\varepsilon_r^{n+1}\vec{N}_i^e\cdot\vec{N}_j^e\right]dV, \quad (7)$$

and

$$[M_e]^n=\frac{\varepsilon_0}{\Delta t}\iiint_{V_e}\left[\varepsilon_r^n\vec{N}_i^e\cdot\vec{N}_j^e\right]dV. \quad (8)$$

Due to the variation of the field magnitude at each time step, $\varepsilon_r(E)$ of each element changes with time, and therefore the mass matrix $[M_e]^{n+1}$ has to be reassembled at every time step. Note that, we have recovered the original expression for $[M_e]$ as in the linear DGTD algorithm, albeit with a field- and time-dependent permittivity. The dependency of $\{e\}^{n+1}$ in $[M_e]^{n+1}$ renders Equation (6) a nonlinear equation.

At each time marching step *n*, the fixed-point method is employed to solve Equation (6), where $\{b\}^{n+1/2}$ is computed with the initial guess $\{e\}_0^{n+1}=\{e\}^n$ and $[M_e]_0^{n+1}=[M_e]^n$. At the *k*th iteration step, the mass matrix $[M_e]_{k-1}^{n+1}$ is inverted to update the field solution $\{e\}_k^{n+1}$. The updated solution is in turn used to update the mass matrix $[M_e]_k^{n+1}$ using Equation (7). If the norm of the residual $\{r\}_k^n$ of Equation (6) is smaller than a predefined threshold, then the nonlinear iteration is converged, and the equation can be marched to the next time step $n+1$. Otherwise it continues with the $(k+1)$th iteration step.

III. GPU IMPLEMENTATION

Because of the necessity to solve nonlinear equations in each time step, the nonlinear DGTD computation is very time-consuming. This computation can be effectively accelerated by exploiting the power of graphics processing units (GPUs). The GPU implementation for the nonlinear DGTD algorithm is similar to the approach described in [4], employing the same coalesced memory accessing pattern and thread/block allocation. Since the electric field update processes that are not related to $\{e\}^{n+1}$ are similar to the ones found in [6], here we focus on the parallelization of the computation related to $\{e\}^{n+1}$, which includes the assembly of the nonlinear mass matrix $[M_e]^{n+1}$ and the inversion of this mass matrix.

To assemble the nonlinear mass matrix, note that each mass matrix entry is numerically integrated through quadrature, where the contribution from each weighted quadrature point is summed. Due to the presence of nonlinearity, ε_r on each quadrature point changes during each iteration step, while the other constituting terms in equation (7) remain identical. To parallelize the assembly of the mass matrix, the constituting matrices at each quadrature point are pre-calculated and stored, and then summed together at each iteration step by first multiplying with the updated ε_r. The proposed parallelization strategy and the memory access pattern are shown in Fig. 1, with each of the total *numTets* elements parallelized over its *numTetDofs* unknowns using CUDA threads. Each threadblock is assigned with a calculated number of elements to utilize all warps [4]. At each iteration step, the mass matrices are assembled by looping through *numQuads* quadrature points and summing their contribution, which is completely parallelizable.

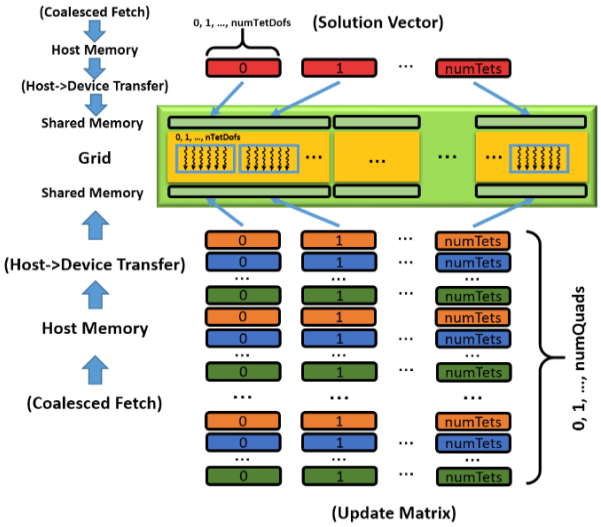


Fig. 1. Parallelization and memory access pattern for the assembly of the nonlinear mass matrices.

To invert the nonlinear mass matrix, we parallelize the standard non-pivoting element-level Gaussian elimination on the GPU. Each *numTetDofs threads* for an element loops over each elemental matrix rows and

reduce them into row echelon form. Although the elimination is only semi-parallelizable, the batch processing of the elimination process for the nonlinear elements somewhat provides a decent speedup. Note that, the mass matrix has a small condition number, and therefore can be easily inverted using the standard Gaussian elimination without partial pivoting. This is beneficial for the GPU acceleration since the partial pivoting process involves many conditional statements and branches, which are undesirable for the parallelization on GPUs.

IV. NUMERICAL EXAMPLES

Two examples are presented here to demonstrate the self-phase modulation, the third-harmonic generation, and the self-focusing effects captured by the extended DGTD algorithm and the GPU speedup. The simulation was carried out on the XSEDE Stampede cluster with NVIDIA Tesla K20 GPUs and Xeon E5-2680 CPU threads.

A. Demonstration of the self-phase modulation and the third-harmonic generation

The first example is a coaxial waveguide with an inner and outer radius of 1 and 2 mm, respectively, and a length of 40 mm. A small section of linear medium is placed near each end for excitation and absorption of the fields, and the rest of the coaxial waveguide is filled with either a linear or nonlinear medium, with a linear permittivity of $\varepsilon_{r,\mathrm{L}} = 1.0$ and a third-order nonlinearity coefficient of $\chi^{(3)} = 4\text{e-}8$. The input signal is a modulated Gaussian pulse with a center frequency of 20 GHz. The number of finite elements is 110,715, and the solution marches at a time step of $\Delta t = 0.075\text{ps}$ for a total of 10,000 time steps for both the linear and nonlinear cases. Mixed first-order basis functions are used for the computation. The time-domain response for the two cases is shown in

Fig. 2. It can be observed that with a linear medium, the shape of the output signal is identical to the input, whereas with a nonlinear medium the output signal steepens and forms shock waves, showing the self-steepening effect [1].

The frequency-domain response for the output signal is shown in

Fig. 3. For the linear case, we have retained the frequency profile of the original input Gaussian pulse centered at 20 GHz. For the nonlinear case, the third-harmonic effect generates harmonics at odd multiples of the original 20 GHz signal at 60 GHz, 100 GHz, 140 GHz, and so on. In addition, the self-phase modulation effect broadens the input bandwidth, where the leading and the trailing edges shift to lower and higher frequencies, respectively [1]. This result is validated using COMSOL. Table 1 gives the average per-step CPU and GPU timing for the simulation. The lower speedup as comparing to [4] is in large due to the uneven nonlinearity encountered by the different elements, which correlates to thread idling in a warp, and the semi-serial nature of the Gaussian elimination process. This thread idleness effectively lowers the number of FLOPS as well as the overall bandwidth.

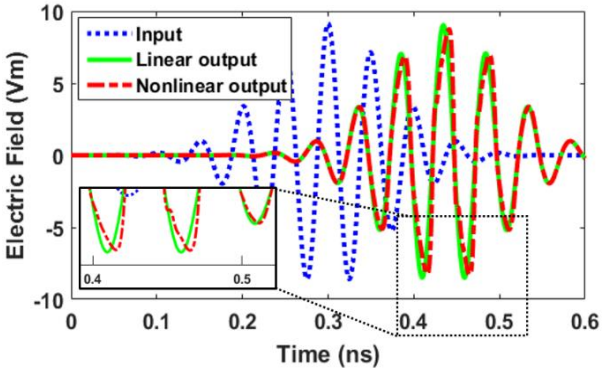


Fig. 2. Time-domain response of the electric field for a coaxial waveguide filled with a section of linear or nonlinear medium.

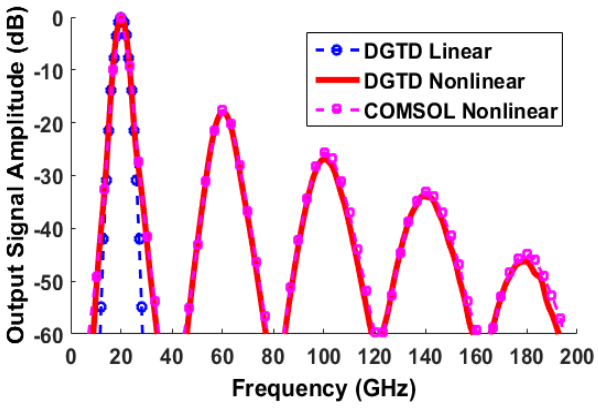


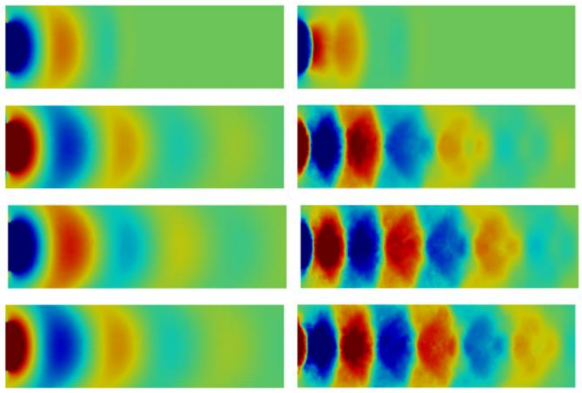
Fig. 3. Frequency-domain response of the output signal for a coaxial waveguide filled with a section of linear or nonlinear medium.

Table 1: Average per-step timing comparison for the simulation of a nonlinear coaxial waveguide

# MPI	1	2	4	8
	CPU Time per Step (ms)			
Marching	1,482.00	741.61	369.71	183.15
Comm.	0	35.51	33.74	38.79
Per-Step	1,482.00	777.12	403.45	221.94
	GPU Time per Step (ms)			
Marching	47.21	23.73	12.00	6.14
Comm.	0	2.94	1.53	4.57
Per-Step	47.21	26.67	13.52	10.71
Speedup	31.39	29.14	29.83	20.72

B. Demonstration of the self-focusing effect

The second example demonstrates the self-focusing effect through beam-shaped field propagation in a $1\text{mm}\times1\text{mm}\times3\text{mm}$ bulk medium. The linear relative permittivity is $\varepsilon_{r,\text{L}}=1.0$ and the third-order nonlinearity coefficient is $\chi^{(3)}=8$. The excitation is a tapered TEM sine wave at 300 GHz, launched through a square aperture with a dimension of a half of the excitation wavelength. The number of finite elements is 664,039, and the solution marches at a time step of $\Delta t=0.01\text{ps}$ for a total of 5,000 time steps, where mixed first-order basis functions are used for the simulation. The field profiles in the bulk medium at various times for both linear and nonlinear cases are shown in Fig. 4. In the nonlinear medium, the specific electric field generates a strong nonlinearity, which results in a maximum instantaneous relative permittivity of $\varepsilon_r=8.27$, or a 727% change to the linear relative permittivity. As can be seen, due to nonlinearity, the field experiences pulse compression which shortens the duration of each pulse. This effect is due to self-phase modulation. As the field propagates along the bulk medium, the wave is naturally diffracted in the linear medium, where the magnitude of the field decreases significantly after a couple of wavelengths. In the nonlinear medium, the intensity of the field modifies the surrounding medium into a self-induced waveguide, which counteracts natural diffraction and preserves the magnitude of the propagating wave for a longer distance in the medium.



(a) (b)

Fig. 4. Time-domain field profile for wave propagation in a: (a) linear and (b) nonlinear medium at 5, 20, 25, and 50ns, respectively.

Table 2 shows the GPU average per-step timing. Since different elements experience different levels of nonlinearity at different times due to the propagation of the field, the CUDA threads for a converged element will idle and wait for the rest of the elements in the same GPU to synchronize before completing the kernel (a single time step). This results in some MPI nodes having to idle and wait for the others to iteratively converge before moving onto the next time step together. This idling time is taking into account in the average communication time, which is significantly longer for the fixed-point method due to the large differences in the number of iterations between different regions at any particular moment. Due to the high nonlinearity of the example, it is impractical to analyze the CPU performance. However, it is expected that higher speedup can be achieved comparing to the previous example, due to the increasing number of elements [4].

Table 2: Average GPU per-step timing (in ms) for the wave propagation in a bulk medium

# MPI	1	2	4	8
Volume	569.93	287.23	142.95	72.41
Surface	10.01	5.04	2.55	1.31
Comm.	0	35.51	33.74	38.79
Per-Step	1,482.00	777.12	403.45	221.94

V. CONCLUSION

The DGTD algorithm was extended to model the instantaneous third-order Kerr-type nonlinearity. The resulting computationally intensive DGTD algorithm was accelerated with GPUs based on the parallelization framework from our prior work. Numerical examples demonstrated that the DGTD simulation was able to capture various nonlinear phenomena and the GPU acceleration was able to achieve a good speedup for this computationally intensive simulation.

REFERENCES

[1] R. W. Boyd, *Nonlinear Optics*. Burlington, MA: Academic Press, 2008.

[2] B. Saleh and M. Tech, *Fundamentals of Photonics*. New York, NY: Wiley, 2013.

[3] R. M. Joseph and A. Taflove, "FDTD Maxwell's equations models for nonlinear electrodynamics and optics," *IEEE Trans. Antennas Propag.*, vol. 45, pp. 364-374, Mar. 1997.

[4] H.-T. Meng and J.-M. Jin, "Acceleration of the dual-field domain decomposition algorithm using MPI-CUDA on large-scale computing systems," *IEEE Trans. Antennas Propag.*, vol. 62, no. 9, pp. 4706-4715, Sept. 2014.