Modeling 3D anisotropic elastodynamics using mimetic finite-differences and fully staggered grids

Harpreet Sethi*, Fatmir Hoxha†, Jeffrey Shragge* & Ilya Tsvankin*

*Center for Wave Phenomena and Dept. of Geophysics, Colorado School of Mines, Golden CO 80401
†NVIDIA Corporation, Santa Clara, California
Corresponding Author email: hsethi@mines.edu

ABSTRACT

Accurate modeling of elastic wavefields in 3D anisotropic media is important for many applications in seismic processing and inversion applications. However, efficient wavefield simulation for tilted transversely isotropic (TTI) media and, especially, for orthorhombic and lower symmetries remains challenging. Finite-difference (FD) implementations using centered Taylor-series coefficients on single staggered grids suffer from reduced accuracy due to problems in computing the partial wavefield derivatives in TTI or tilted orthorhombic (TOR) media and in enforcing the free-surface (zero-traction) boundary conditions. To address these issues, we develop a 3D mimetic FD (MFD) algorithm for arbitrarily anisotropic media that uses fully staggered grids. This CUDA-based algorithm is implemented on graphics processing units (GPUs) to leverage the massive parallelism of this computer architecture. For multi-GPU parallelization, we employ the CUDA-aware MPI library that exploits the remote direct memory access feature (RDMA) for buffer transfers. Weak- and strong-scaling tests on one DGX NVIDIA A100 node with eight GPUs demonstrate that our efficient multi-GPU implementation achieves a quasilinear computational speedup.

Key words: Elastic media, anisotropy, orthorhombic symmetry, wave propagation, finite-difference modeling, GPU, boundary conditions

1 INTRODUCTION

Generating accurate numerical wavefield solutions for large-scale anisotropic elastic models is a computationally challenging problem. There is a need for efficient modeling engines for arbitrarily anisotropic media that do not reduce the accuracy of the overall numerical scheme. An additional complexity is in implementing the correct boundary conditions at interfaces (e.g., at a free surface, fluid/solid interface) with strong property contrasts. Inaccurate handling of the boundary conditions with lower-order numerical approximations can produce significant wavefield distortions, especially for surface waves (De Basabe and Sen, 2014). Here we aim to improve the performance of 3D anisotropic elastic wavefield solvers by developing an accurate high-order implementation of the free-surface boundary conditions (FSBC) using multiple graphic processing unit (GPU) cards (Foltinek et al., 2009). Also, we employ fully staggered grids to avoid the time-consuming interpolation of the partial wavefield derivatives.

The advent of GPUs as accelerators in elastic wavefield modeling, imaging, and inversion is due to their single-instruction multiple-data (SIMD) nature. Numerous methodologies for modeling wave propagation have been proposed and ported to GPUs. Komatitsch et al. (2010) use a GPU-based spectral-element method to simulate elastic wave propagation at the continental scale. They employ nonblocking message passing based on MPI to overlap the data transfer with computations on GPUs across the network via peripheral component interconnect express (PCIe). Mu et al. (2013) study the portability of a higher-order discontinuous Galerkin method for modeling of elastic seismic wave propagation on a single GPU using unstructured tetrahedral meshes. Micikevicius (2009) and Abdelkhalek et al. (2009) discuss the GPU implementation of finite-difference (FD) algorithms designed to solve the acoustic wave equation. Nakata et al. (2011) present the solutions of the 3D isotropic elastic wave equation on multiple GPUs. Weiss and Shragge (2013) introduce a FD algorithm for modeling elastic wave propagation in anisotropic media and discuss both single- and multi-GPU implementations. Rubio et al. (2014) present a FD solution of the anisotropic elastic wave equation using standard and fully staggered grids (SSGs and FSGs, respectively).

FD methods are often employed for modeling seismic wavefields in exploration geophysics because of their advantages in terms of implementation, code parallelization, compact stencils, and moderate computational cost. However, they suffer from reduced numerical accuracy in tilted anisotropic media due to problems in the interpolation of the partial wavefield derivatives, especially near the free surface and fluid/solid interface (Sethi et al., 2021). Using FD methods based on mimetic FD (MFD) operators with fully staggered grids (FSGs), overcomes this issue by preserving the underlying physics of the employed partial differential equations (PDE) in the discretization process (Castillo and Miranda, 2013). Using FSGs helps avoid the interpolation of partial wavefield derivatives, thus making it easier to implement the algorithm for arbitrarily anisotropic media (de la Puente et al., 2014). Otero et al. (2017) compare the performance of the MFD method for acoustic wave propagation on CPUs and GPUs. They discuss 1D and 2D implementations using a second-order formulation of the acoustic wave equation and show that the computational performance of the GPU implementation is consistently better than its CPU counterparts.

In this work, we employ GPUs to solve the 3D anisotropic elastic wave equation using the MFD+FSG approach. Our implementation involves a velocity-stress formulation with FSGs to handle arbitrarily anisotropic media without losing global numerical discretization accuracy. The MFD+FSG algorithm and MFD operators are applied only to the near-surface part of the model to reduce the thread divergence in CUDA kernels. The absorbing boundary conditions are applied on the remaining five sides of the model. Multi-GPU communications are handled with a CUDA-aware MPI library for both single and multi-nodes.

First, we briefly review the theory of elastic wave propagation, the free-surface boundary conditions, and the velocity-stress formulation of the elastic wave equation for orthorhombic media (Tsvankin, 2012) with a horizontal symmetry plane. The major algorithmic steps of the GPU-based MFD+FSG implementation are described next. Finally, we present a numerical simulation example, as well as code profiling and scaling performance tests on up to eight GPUs in a single Nvidia A100 node.

2 THEORY

The 3D elastic wave equation in a heterogeneous anisotropic medium can be written as:

$$\rho\,\dot{v}_i = \partial_j \sigma_{ij} + f_i, \tag{1}$$

where ρ is the density, v_i is the ith component of the particle velocity, the dot marks the temporal derivative, σ_{ij} is the stress tensor, and f_i is the body force per unit volume. We use the following linear constitutive relationship (i.e., the generalized Hooke's law) between the temporal derivatives of the stress σ_{ij} and strain ϵ_{kl} tensors,

$$\dot{\sigma}_{ij} = C_{ijkl}\,\dot{\epsilon}_{kl}, \tag{2}$$

where C_{ijkl} is the (time-independent) stiffness tensor and

$$\dot{\epsilon}_{kl} = \frac{1}{2}\left(\partial_l v_k^s + \partial_k v_l^s\right). \tag{3}$$

The free-surface boundary condition (FSBC) requires a vanishing traction vector across the free surface,

$$\sigma_{ij}\,n_j = 0, \tag{4}$$

where n_j are the direction cosines for a unit vector orthogonal to the area (surface) element. Assuming a flat free surface located at x_3 = 0, equation 4 yields:

$$\sigma_{13} = \sigma_{23} = \sigma_{33} = 0. \tag{5}$$

3 3D MFD+FSG APPROACH

Generating *globally* high-order accurate [i.e., $O(\Delta x^4)$ or greater] implementations of the FSBC specified in equation 5 remains challenging for standard FD methods even for simple Cartesian geometries. Numerical solutions that satisfy the elastic (anisotropic) wave equations and the boundary conditions with the same accuracy tend to become unstable. In addition, many lower-order-accuracy boundary-condition implementations introduce unphysical ghost points to ensure the continuity of the wavefield derivatives obtained with two-sided FD operators (Sun et al., 2017). Wave-equation discretizations based on standard Taylor-series FD operators also do not honor such underlying physical concepts as the conservation laws and tensorial calculus properties that are naturally satisfied by their continuum counterparts (Castillo and Grone, 2003). These shortcomings lead to numerical instabilities and errors in the simulated wavefields.

An efficient way to address these issues is by employing mimetic operators (Castillo and Grone, 2003; Rojas, 2007; Corbino and Castillo, 2017). The MFD divergence and gradient operators, $\mathbf{D}$ and $\mathbf{G}$, honor the global conservation laws (Castillo and Grone, 2003) and can be constructed with $O(\Delta x^4)$ (or greater) accuracy throughout the entire computational domain including at and near boundaries and partitioned interfaces (Castillo and Miranda, 2013; Corbino and Castillo, 2017).

We use an FSG scheme to implement the MFD operators because such grid layout increases the numerical stability by making all partial wavefield derivatives available at each grid point (de la Puente et al., 2014; Shragge and Tapley, 2017). In contrast to SSGs, this approach eliminates the need for costly high-order interpolation of the partial derivatives (Lisitsa and Vishnevskiy, 2010; de la Puente et al., 2014). Employing an FSG system is particularly useful for lower-symmetry anisotropic models including the most general, triclinic symmetry (Lisitsa and Vishnevskiy, 2010).

A 3D FSG grid is comprised of four complementary SSGs staggered by half-grid spacing in all three directions. Figure 1a shows the distribution of the particle-velocity variables defined on four SSG grids that form the FSG system: $[\mathbf{v},\mathbf{v},\mathbf{v}]$, $[\mathbf{v},\mathbf{f},\mathbf{f}]$, $[\mathbf{f},\mathbf{v},\mathbf{f}]$ and $[\mathbf{f},\mathbf{f},\mathbf{v}]$, where $\mathbf{f} \in \mathbb{R}^{N+2}$, $\mathbf{v} \in \mathbb{R}^{N+1}$ and N represents the number of points in the model along the x_1, x_2, x_3 dimensions. Figure 1a shows the number of points corresponding to each SSG grid that contributes to the injection and extraction of the velocity variables in a single FSG cell. The $[\mathbf{v},\mathbf{v},\mathbf{v}]$ grid contributes eight points at the corners in an FSG system, whereas $[\mathbf{v},\mathbf{f},\mathbf{f}]$, $[\mathbf{f},\mathbf{v},\mathbf{f}]$ and $[\mathbf{f},\mathbf{f},\mathbf{v}]$ grids contribute two points each at the face center in Figure 1a. Similarly, the stress variables have to be defined on four SSG grids: $[\mathbf{f},\mathbf{f},\mathbf{f}]$, $[\mathbf{v},\mathbf{f},\mathbf{v}]$, $[\mathbf{v},\mathbf{v},\mathbf{f}]$ and $[\mathbf{f},\mathbf{v},\mathbf{v}]$. Figure 1b shows the number of points corresponding to each employed SSG grid. The $[\mathbf{f},\mathbf{f},\mathbf{f}]$ grid contributes to one point in the cell center. The $[\mathbf{v},\mathbf{f},\mathbf{v}]$, $[\mathbf{v},\mathbf{v},\mathbf{f}]$ and $[\mathbf{f},\mathbf{v},\mathbf{v}]$ grids contribute four points each in Figure 1b. For injection and extraction of the velocity and stress wavefields, different weighting coefficients need to be used for different grids as explained below.

The mimetic $\mathbf{D_i}$ and $\mathbf{G_i}$ operators act on the complementary SSG grids in 1D fashion but along the three coordinate directions. The $\mathbf{G_i}$ operator is applied only to the field variables defined on the $\mathbf{f}$-grid and maps them to a vector defined on the $\mathbf{v}$-grid. The $\mathbf{D_i}$ operator acts only on the field variables defined on the $\mathbf{v}$-grid and maps them to a vector defined on an $\mathbf{f}$-grid. The next section describes the discretization of the stress-velocity formulation with MFD operators for orthorhombic media with a horizontal symmetry plane. However, it can be extended to arbitrarily anisotropic media in a straightforward fashion.

3.1 Stress field updating

The stress-updating equation for 3D orthorhombic media with a horizontal symmetry plane can be written using equations 2 and 3 as:

$$\dot{\sigma}_{11} = c_{11}\frac{\partial v_1}{\partial x_1} + c_{12}\frac{\partial v_2}{\partial x_2} + c_{13}\frac{\partial v_3}{\partial x_3}, \tag{6}$$

$$\dot{\sigma}_{22} = c_{12}\frac{\partial v_1}{\partial x_1} + c_{22}\frac{\partial v_2}{\partial x_2} + c_{23}\frac{\partial v_3}{\partial x_3}, \tag{7}$$

$$\dot{\sigma}_{33} = c_{13}\frac{\partial v_1}{\partial x_1} + c_{23}\frac{\partial v_2}{\partial x_2} + c_{33}\frac{\partial v_3}{\partial x_3}, \tag{8}$$

$$\dot{\sigma}_{12} = c_{66}\left(\frac{\partial v_2}{\partial x_1} + \frac{\partial v_1}{\partial x_2}\right), \tag{9}$$

$$\dot{\sigma}_{13} = c_{55}\left(\frac{\partial v_3}{\partial x_1} + \frac{\partial v_1}{\partial x_3}\right), \tag{10}$$

$$\dot{\sigma}_{23} = c_{44}\left(\frac{\partial v_3}{\partial x_2} + \frac{\partial v_2}{\partial x_3}\right), \tag{11}$$

where c_{ij} are the stiffness coefficients in the two-index (Voigt) notation and dots over the stress components indicate the temporal derivative. All stress variables are defined at the four complementary staggered grids mentioned above. For example, the $\dot{\sigma}_{ij}$ updates, herein assumed to be computed on the half time step $n+\frac{1}{2}$, are obtained on the grid $[\mathbf{f},\mathbf{f},\mathbf{f}]$ using MFD operators as:

$$\dot{\sigma}_{11}^{[f,f,f]} = c_{11}\mathbf{D_1}v_1^{[v,f,f]} + c_{12}\mathbf{D_2}v_2^{[f,v,f]} + c_{13}\mathbf{D_3}v_3^{[f,f,v]}, \tag{12}$$

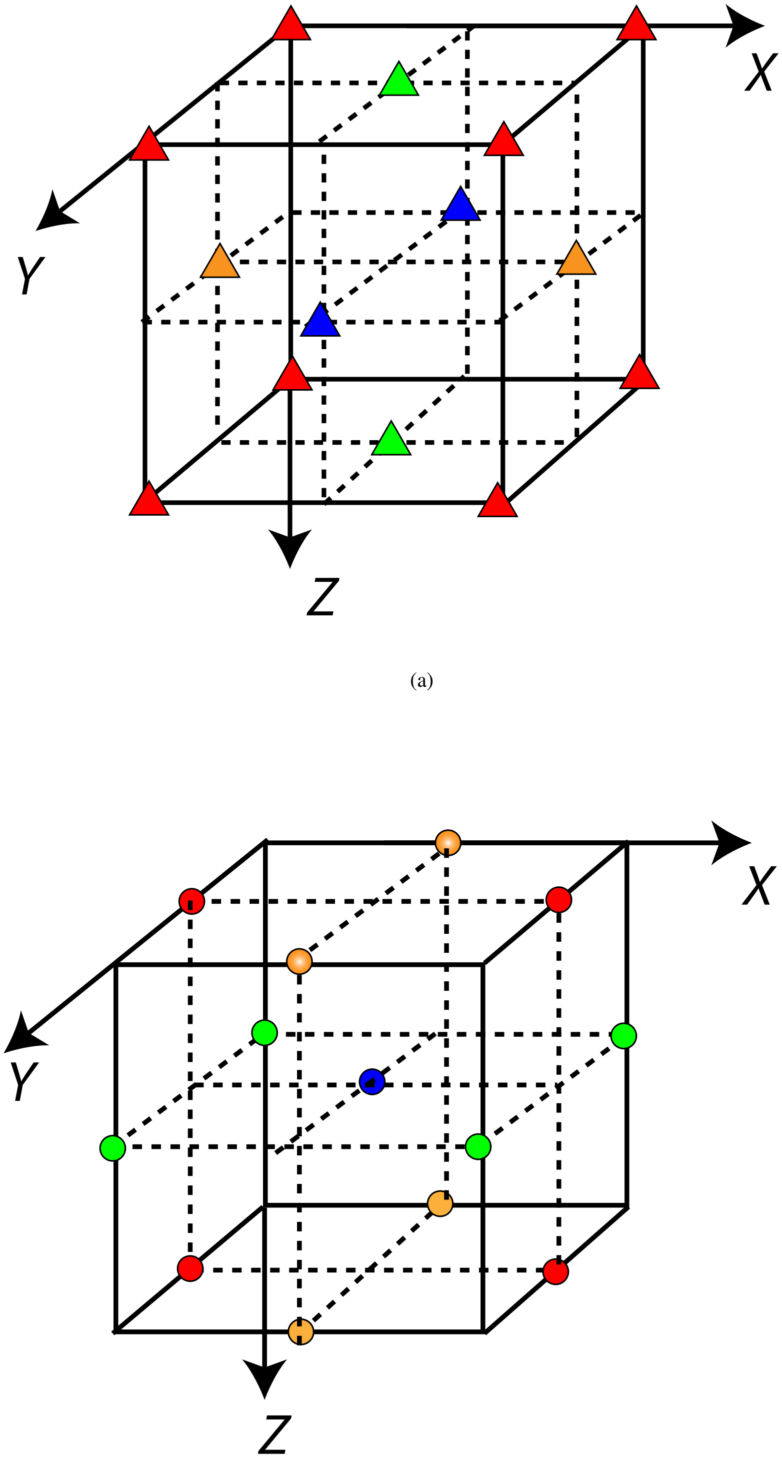
$$\dot{\sigma}_{22}^{[f,f,f]} = c_{12}\mathbf{D_1}v_1^{[v,f,f]} + c_{22}\mathbf{D_2}v_2^{[f,v,f]} + c_{23}\mathbf{D_3}v_3^{[f,f,v]}, \tag{13}$$

$$\dot{\sigma}_{33}^{[f,f,f]} = c_{13}\mathbf{D_1}v_1^{[v,f,f]} + c_{23}\mathbf{D_2}v_2^{[f,v,f]} + c_{33}\mathbf{D_3}v_3^{[f,f,v]}, \tag{14}$$

$$\dot{\sigma}_{12}^{[f,f,f]} = c_{66}\left(\mathbf{D_1}v_2^{[v,f,f]} + \mathbf{D_2}v_1^{[f,v,f]}\right), \tag{15}$$

$$\dot{\sigma}_{13}^{[f,f,f]} = c_{55}\left(\mathbf{D_1}v_3^{[v,f,f]} + \mathbf{D_3}v_1^{[f,f,v]}\right), \tag{16}$$

$$\dot{\sigma}_{23}^{[f,f,f]} = c_{44}\left(\mathbf{D_2}v_3^{[f,v,f]} + \mathbf{D_3}v_2^{[f,f,v]}\right), \tag{17}$$



(a)

(b)

Figure 1. Coupled system of SSGs inside a 3D MFD FSG cell. (a) The [**v**, **v**, **v**], [**v**, **f**, **f**], [**f**, **v**, **f**], and [**f**, **f**, **v**] grid points are in red, green, blue and yellow, respectively. (b) The [**f**, **f**, **f**], [**v**, **f**, **v**], [**v**, **v**, **f**] and [**f**, **v**, **v**] grid points are in blue, green, red and yellow, respectively.

where the variables v_i are assumed to be computed on the whole time step n. The superscripts denote the SSG grid where the variables are defined. Table 1 shows the stress updates for the other complementary SSG grid staggerings.

3.2 Velocity field updating

The velocity-updating equation for 3D orthorhombic media can be obtained using equation 1:

$$\begin{aligned} \rho\dot{v}_1 &= \frac{\partial\sigma_{11}}{\partial x_1} + \frac{\partial\sigma_{12}}{\partial x_2} + \frac{\partial\sigma_{13}}{\partial x_3}, && (18) \\ \rho\dot{v}_2 &= \frac{\partial\sigma_{21}}{\partial x_1} + \frac{\partial\sigma_{22}}{\partial x_2} + \frac{\partial\sigma_{23}}{\partial x_3}, && (19) \\ \rho\dot{v}_3 &= \frac{\partial\sigma_{31}}{\partial x_1} + \frac{\partial\sigma_{32}}{\partial x_2} + \frac{\partial\sigma_{33}}{\partial x_3}, && (20) \end{aligned}$$

where the $\dot{v}_i$ and σ_{ij} components are again assumed to be obtained on the whole (n) and half-time ($n + \frac{1}{2}$) steps, respectively.

Similarly, all velocity variables are defined at the four complementary staggered grids mentioned above. For example, the update of the velocity fields on the $[\mathbf{v}, \mathbf{v}, \mathbf{v}]$-grid can be computed using MFD operators as follows:

$$\begin{aligned} \rho\,\dot{v}_1^{[v,v,v]} &= \mathbf{G_1}\sigma_{11}^{[f,v,v]} + \mathbf{G_2}\sigma_{12}^{[v,f,v]} + \mathbf{G_3}\sigma_{13}^{[v,v,f]}, && (21) \\ \rho\,\dot{v}_2^{[v,v,v]} &= \mathbf{G_1}\sigma_{21}^{[f,v,v]} + \mathbf{G_2}\sigma_{22}^{[v,f,v]} + \mathbf{G_3}\sigma_{23}^{[v,v,f]}, && (22) \\ \rho\,\dot{v}_3^{[v,v,v]} &= \mathbf{G_1}\sigma_{31}^{[f,v,v]} + \mathbf{G_2}\sigma_{32}^{[v,f,v]} + \mathbf{G_3}\sigma_{33}^{[v,v,f]}. && (23) \end{aligned}$$

The superscripts denote the SSG grids where the variables are defined. Table 2 shows the velocity updates obtained using MFD operators for the other complementary SSG grid staggerings.

3.3 Free-surface implementation

The mimetic nodes at the free surface are updated using the strategy described in Sethi et al. (2021) by employing the zero-traction boundary conditions (equation 5). The velocities $v_1^{[f,f,v]}$,$v_2^{[f,f,v]}$ and $v_3^{[f,f,v]}$ at the free-surface mimetic points for orthorhombic media are obtained by setting $\dot{\sigma}_{33}^{[v,f,v]} = \dot{\sigma}_{23}^{[v,f,v]} = \dot{\sigma}_{13}^{[v,f,v]} = 0$ at $x_3 = 0$:

$$\begin{aligned} 0 &= c_{13}\,\mathbf{G_1}v_1^{[f,f,v]} + c_{23}\,\mathbf{D_2}v_2^{[v,v,v]} + c_{33}\,\mathbf{G_3}v_3^{[v,f,f]}, && (24) \\ 0 &= \mathbf{G_3}v_2^{[v,f,f]} + \mathbf{D_2}v_3^{[v,v,v]}, && (25) \\ 0 &= \mathbf{G_1}v_3^{[f,f,v]} + \mathbf{G_3}v_1^{[v,f,f]}. && (26) \end{aligned}$$

Rearranging equations 6-8 leads to:

$$\begin{aligned} v_1^M &= \frac{1}{c_{33}\mathbf{G_1}[\mathbf{0}]}\left(-c_{13}\mathbf{G_1}v_1^{[f,f,v]} - c_{23}\mathbf{D_2}v_2^{[v,v,v]} - c_{33}\mathbf{G_3}v_3^{[v,f,f]}\right), && (27) \\ v_2^M &= -\frac{1}{\mathbf{G_3}[\mathbf{0}]}\left(-\mathbf{G_3}v_2^{[v,f,f]} - \mathbf{D_2}v_3^{[v,v,v]}\right), && (28) \\ v_3^M &= -\frac{1}{\mathbf{G_1}[\mathbf{0}]}\left(-\mathbf{G_1}v_3^{[f,f,v]} - \mathbf{G_3}v_1^{[v,f,f]}\right), && (29) \end{aligned}$$

where the superscript M indicates mimetic points, and $\mathbf{G_1}[\mathbf{0}]$ and $\mathbf{G_3}[\mathbf{0}]$ denote the mimetic-gradient operator coefficients at $x_1 = 0$ and $x_3 = 0$, respectively. The mimetic velocities $v_1^{[f,v,f]}$, $v_2^{[f,v,f]}$, and $v_3^{[f,v,f]}$ are updated in a similar fashion.

3.4 Source injection and wavefield extraction

The coupling of multiple grids requires distributed source injection and wavefield extraction in an FSG system. Figure 1b shows the distribution of stress nodes on four SSGs which are involved in the cell-centered injection and extraction in the FSG system. The stress nodes at $[\mathbf{f}, \mathbf{f}, \mathbf{f}]$ are injected with a unit weight, whereas the nodes at the three other SSGs are injected with a weight of 0.25. For extraction, the stress nodes at $[\mathbf{f}, \mathbf{f}, \mathbf{f}]$ are extracted with a 0.25 weight and rest of the three SSGs are extracted with a weight of 0.0625. Figure 1a shows the distribution of the velocity nodes on the four SSGs involved in the cell-centered velocity injection and extraction. The velocity nodes at $[\mathbf{v}, \mathbf{v}, \mathbf{v}]$ are injected with a weight of 0.125 and the three other SSGs are injected with a weight of 0.5. For velocity wavefield extraction, the velocity nodes at $[\mathbf{v}, \mathbf{v}, \mathbf{v}]$ are extracted with a weight of 0.03125 and other three SSGs are extracted with a weight of 0.125.

Updated variable σ	SG1 [**v**, **v**, **v**]	SG2 [**v**, **f**, **f**]	SG3 [**f**, **v**, **f**]	SG4 [**f**, **f**, **v**]
[**f**, **f**, **f**]	—	$\mathbf{D_1}$	$\mathbf{D_2}$	$\mathbf{D_3}$
[**v**, **f**, **v**]	$\mathbf{D_2}$	$\mathbf{G_3}$	—	$\mathbf{G_1}$
[**v**, **v**, **f**]	$\mathbf{D_3}$	$\mathbf{G_2}$	$\mathbf{G_1}$	—
[**f**, **v**, **v**]	$\mathbf{D_1}$	—	$\mathbf{G_3}$	$\mathbf{G_2}$

Table 1. Updates of the stress variables obtained from the velocity variables defined on four different SSGs using cyclic MFD operators.

Updated variable $\mathbf{v}$	SG1 [**f**, **f**, **f**]	SG2 [**v**, **f**, **v**]	SG3 [**v**, **v**, **f**]	SG4 [**f**, **v**, **v**]
[**v**, **v**, **v**]	—	$\mathbf{G_2}$	$\mathbf{G_3}$	$\mathbf{G_1}$
[**v**, **f**, **f**]	$\mathbf{G_1}$	$\mathbf{D_3}$	$\mathbf{D_2}$	—
[**f**, **v**, **f**]	$\mathbf{G_2}$	—	$\mathbf{D_1}$	$\mathbf{D_3}$
[**f**, **f**, **v**]	$\mathbf{G_3}$	$\mathbf{D_1}$	—	$\mathbf{D_2}$

Table 2. Updates of the velocity variables from the stress variables defined on four different SSGs using cyclic MFD operators.

4 GPU IMPLEMENTATION

Concurrently running thousands of threads leads to significant GPU (compared to CPU) speedups. To simplify the GPU code, we make several algorithmic modifications. First, all four SSGs are adjusted to coincide with the $[\mathbf{f}, \mathbf{f}, \mathbf{f}]$ grid [i.e., $(N+2, N+2, N+2)$]. Second, we use the MFD operators only near the free surface to compute the partial derivatives and minimize the thread divergence; absorbing boundary conditions are implemented on the other five model sides.

The major algorithmic steps, such as updating velocity and stress wavefield variables, source injection and wavefield extraction, and FSBC application are implemented as separate GPU kernels. Table 3 shows the major steps in our MFD+FSG algorithm; steps 3-8 are implemented as GPU kernels. We use the strategy proposed by Micikevicius (2009) for updating the field variables to reduce global memory usage and read redundancy. The order of axes from slow to fast is x_3, x_2, and x_1. The data in the $[x_1, x_2]$-plane are loaded into the shared memory, and variables along x_3-axis are held in registers. Domain decomposition is performed along the x_3-axis (Figure 2) to facilitate the application of FSBC and transfer the data only in the x_3-direction.

The intertwining of the four SSGs, however, introduces greater algorithms complexity and further reduces read redundancy. For example, $v_1^{[v,f,f]}$ is only involved in the update of $\sigma_{11}^{[f,f,f]}$, $\sigma_{22}^{[f,f,f]}$, and $\sigma_{33}^{[f,f,f]}$, but not in the other stress components. First, we read $v_1^{[v,f,f]}$, load it to the shared memory, and then partially update the three stress components to reduce the number of global memory accesses. Updating the shear stress components requires the field v_1 to be defined on different SSGs; however, such redundancy is not feasible for the velocity update kernel that relies on different stress components.

We use CUDA streams to compute the halo regions, to update mimetic nodes, and for communicating halos across GPUs for both single- and multi-node implementation. Figure 3 shows the CUDA code profile for two GPUs. The communication of the halo region across the GPUs in Figure 2 is completely overlapped with velocity and stress computation in the internal region. For communication, we employ a CUDA Aware MPI library, which can directly use the GPUDirect technologies and the RDMA feature. That feature permits direct inter-GPU communication of information in the GPU buffers without first staging them to CPU through a connector. NVIDIA GPUDirect technologies provide high-bandwidth, low-latency communications with NVIDIA GPUs and cover all types of inter-rank communications (intra-node, inter-node, and RDMA inter-node).

5 EXPERIMENTS

For numerical tests, we use the $O(\Delta x^4)$ MFD divergence and gradient operators and test our approach on an orthorhombic model (Tsvankin, 1997) with the symmetry planes that coincide with the Cartesian coordinate planes. The model size with a 7.5 m spacing is $[N_x, N_y, N_z] = 256^3$. We inject a Ricker wavelet with a peak frequency of 20 Hz with Δt = 0.8 ms at $[x, y, z] = [0.96, 0.96, 0.48]$ km. Figure 4 displays the particle-velocity wavefield snapshots for all three components. After running the simulation for 500 time steps, we observe both P- and S-wavefields that include reflections and mode conversions at the free surface.

Step	Substep	Instruction	Equation(s)
0		Read stiffness, source wavelet, and density fields.	—
1		Initialize data structures on GPU and copy the data from CPU to GPU.	—
2		For all time steps:	—
3		Update mimetic stress nodes	6-11
4		Update stress field	6-11
	4a	Update stress halos	—
	4b	Asynchronously communicate stress halos and compute stress in the middle.	—
5		Inject source	—
6		Update mimetic velocity nodes	18-20
7		Update velocity field	18-20
	7a	Update velocity halos	—
	7b	Asynchronously communicate velocity halos and compute velocity in the middle.	—
8		Apply FSBC	27-29
9		Iterate steps 1-8	—

Table 3. Pseudocode that outlines the main steps of the 3D MFD+FSG numerical solution and lists the relevant equations.

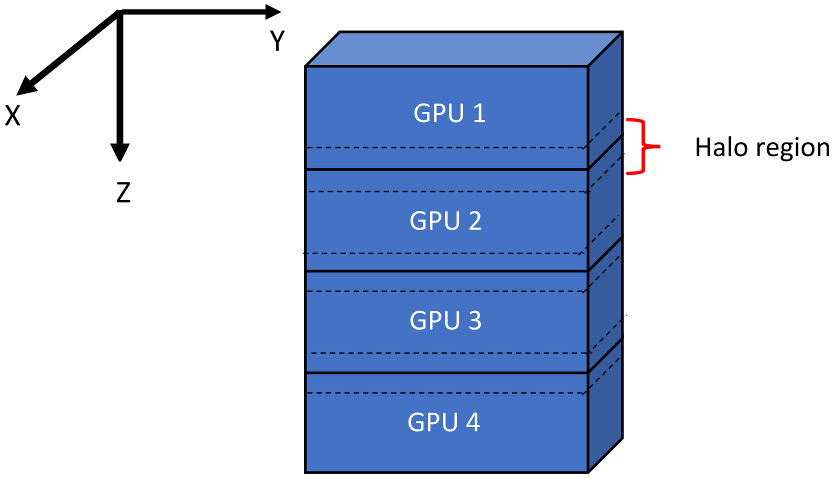
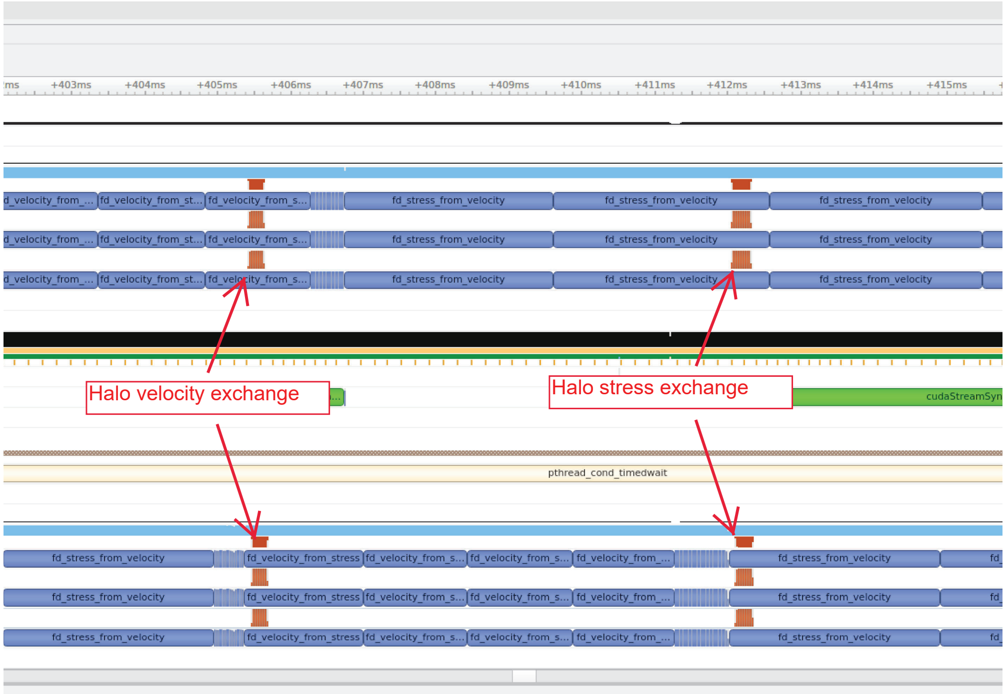


Figure 2. Domain decomposition along the z-direction. The dashed line marks the halo region that needs to be communicated across consecutive GPUs.

The code is tested on the DGX A100 system, which consists of eight A100 GPUs (80 GB per card) connected via the NVLINK. Figure 5a shows initial weak-scaling efficiency tests (i.e., the model size per GPU is the same) when using up to eight GPUs. The model size is $[N_x, N_y, N_z] = [1024, 1024, 256 \times N]$, where N is the number of GPUs. The blue line in Figure 5a suggests that the communication overhead using NVLINK on a single node is negligible. For a strong-scaling test we use the model size $[N_x, N_y, N_z] = 512^3$ and increase the number of GPUs, while keeping the model size fixed. Figure 5b shows the quasilinear speedup when increasing the number of A100 GPUs from one to eight.



Halo velocity exchange

Halo stress exchange

Figure 3. Overlap of computation and communication across two GPUs. The top and bottom parts of the plot represent the workflow of two MPI processes corresponding to two GPUs. The blue boxes mark the computational kernels and the brown ones mark the communication between the GPUs. The exchange of the velocity and stress halos completely overlaps with computation of the internal velocity and stress nodes, respectively.

6 CONCLUSIONS

We developed an efficient GPU-based MFD+FSG algorithm using the velocity-stress formulation of the anisotropic elastic wave equation. The algorithm can be used for large-scale models (i.e., $N_x \times N_y \times N_z > 512^3$) with orthorhombic and even lower symmetry. The initial weak- and strong-scaling tests demonstrate that the communication overhead is negligible for a single A100 node with eight GPUs using the CUDA Aware-MPI library. We plan to implement the convolutional perfectly matching layer (C-PML) absorbing boundary conditions on the model sides (except for the free surface) and make our parallelization more efficient by extending it to multinode GPU systems.

7 ACKNOWLEDGMENTS

This work was supported by the sponsors of the Consortium Project on Seismic Inverse Methods for Complex Structures at Center for Wave Phenomena at Colorado School of Mines. We would also like to thank NVIDIA for providing the computational resources for running the numerical experiments.

REFERENCES

Abdelkhalek, R., H. Calandra, O. Coulaud, J. Roman, and G. Latu, 2009, Fast seismic modeling and reverse time migration on a GPU cluster: 2009 International Conference on High Performance Computing & Simulation, IEEE, 36–43.

Castillo, J. E., and R. Grone, 2003, A matrix analysis approach to higher-order approximations for divergence and gradients satisfying a global conservation law: SIAM Journal on Matrix Analysis and Applications, **25**, 128–142.

Castillo, J. E., and G. F. Miranda, 2013, Mimetic discretization methods: Chapman and Hall/CRC.

Corbino, J., and J. Castillo, 2017, Computational Science & Engineering: San Diego State University.

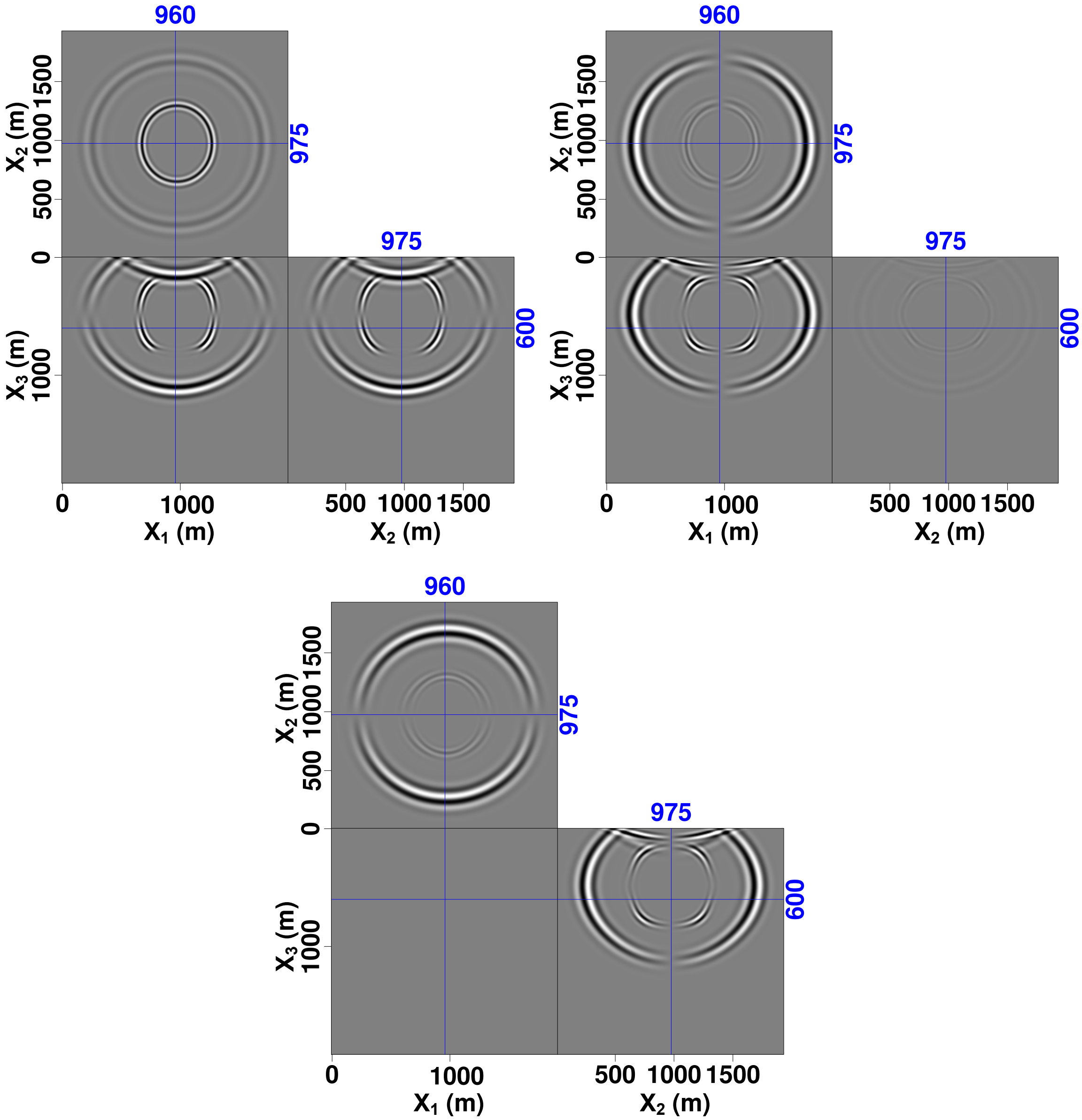


Figure 4. Snapshots of the velocity components (a) v_1, (b) v_2 and (c) v_3 extracted at t=0.4 s . The model is an orthorhombic solid with the following Tsvankin's (1997; 2012) parameters: V_{P0} = 2000 m/s, V_{S0} = 1200 m/s, ϵ_1 = 0.2, ϵ_2 = 0.15, δ_1 = 0.1, δ_2 = 0.07, δ_3 = 0.05, γ_1 = 0.1, and γ_2 = 0.2.

De Basabe, J. D., and M. K. Sen, 2014, A comparison of finite-difference and spectral-element methods for elastic wave propagation in media with a fluid-solid interface: Geophysical Journal International, **200**, 278–298.

de la Puente, J., M. Ferrer, M. Hanzich, J. E. Castillo, and J. M. Cela, 2014, Mimetic seismic wave modeling including topography on deformed staggered grids: Geophysics, **79**, no. 3, T125–T141.

Foltinek, D., D. Eaton, J. Mahovsky, P. Moghaddam, and R. McGarry, 2009, Industrial-scale reverse time migration on GPU hardware, *in* SEG Technical Program Expanded Abstracts 2009: Society of Exploration Geophysicists, 2789–2793.

Komatitsch, D., G. Erlebacher, D. Göddeke, and D. Michéa, 2010, High-order finite-element seismic wave propagation modeling with MPI on a large GPU cluster: Journal of Computational Physics, **229**, 7692–7714.

Lisitsa, V., and D. Vishnevskiy, 2010, Lebedev scheme for the numerical simulation of wave propagation in 3D anisotropic elasticity: Geophysical Prospecting, **58**, 619–635.

Micikevicius, P., 2009, 3D finite difference computation on GPUs using CUDA: Proceedings of 2nd workshop on general purpose processing on graphics processing units, 79–84.

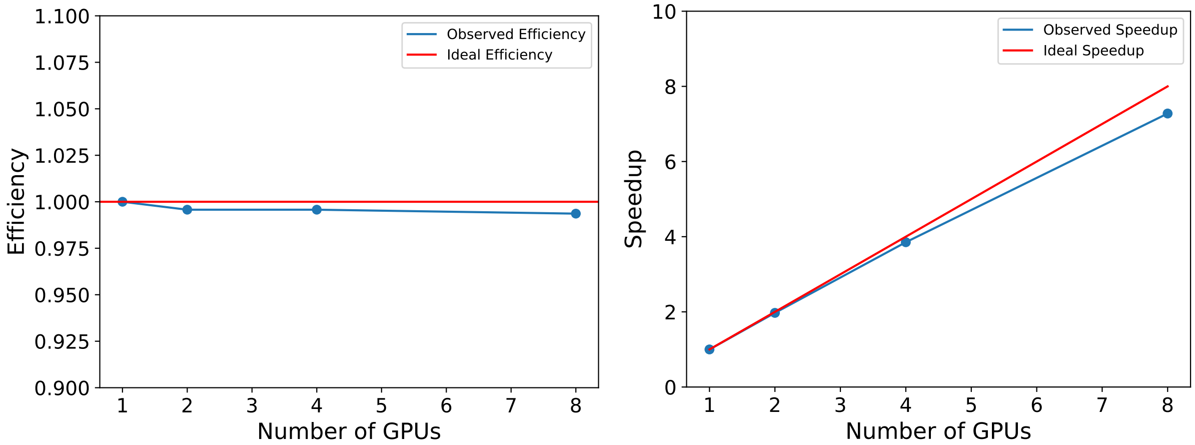


Figure 5. (a) Weak-scaling efficiency for $[N_x, N_y, N_z]$=$[1024, 1024, 256 \times N]$. (b) The strong-scaling efficiency for $[N_x, N_y, N_z]$=512^3.

Mu, D., P. Chen, and L. Wang, 2013, Accelerating the discontinuous Galerkin method for seismic wave propagation simulations using the graphic processing unit (GPU)—single-GPU implementation: Computers & geosciences, **51**, 282–292.

Nakata, N., T. Tsuji, and T. Matsuoka, 2011, Acceleration of computation speed for elastic wave simulation using a graphic processing unit: Exploration Geophysics, **42**, 98–104.

Otero, B., J. Francés, R. Rodriguez, O. Rojas, F. Solano, and J. Guevara-Jordan, 2017, A performance analysis of a mimetic finite difference scheme for acoustic wave propagation on GPU platforms: Concurrency and Computation: Practice and Experience, **29**, e3880.

Rojas, O., 2007, Mimetic finite difference modeling of 2D elastic P-SV wave propagation: Qualifying examination report, Computational Science Research Center, San Diego State University.

Rubio, F., M. Hanzich, A. Farrés, J. de la Puente, and J. M. Cela, 2014, Finite-difference staggered grids in GPUs for anisotropic elastic wave propagation simulation: Computers & Geosciences, **70**, 181–189.

Sethi, H., J. Shragge, and I. Tsvankin, 2021, Mimetic finite-difference coupled-domain solver for anisotropic media: Geophysics, **86**, no. 1, T45–T59.

Shragge, J., and B. Tapley, 2017, Solving the tensorial 3D acoustic wave equation: A mimetic finite-difference time-domain approach: Geophysics, **82**, no. 4, T183–T196.

Sun, Y.-C., W. Zhang, J.-K. Xu, and X. Chen, 2017, Numerical simulation of 2-D seismic wave propagation in the presence of a topographic fluid–solid interface at the sea bottom by the curvilinear grid finite-difference method: Geophysical Journal International, **210**, 1721–1738.

Tsvankin, I., 1997, Anisotropic parameters and P-wave velocity for orthorhombic media: Geophysics, **62**, 1292–1309.

———, 2012, Seismic signatures and analysis of reflection data in anisotropic media; 3rd edition: Society of Exploration Geophysicists.

Weiss, R. M., and J. Shragge, 2013, Solving 3D anisotropic elastic wave equations on parallel GPU devices: Geophysics, **78**, no. 2, F7–F15.