3D-FDM for elastic wave modeling in the presence of irregular topography by using unstructured index array representation on a GPU

Ivan J. Sánchez-Galvis *, William Agudelo†, Herling González†, Daniel O. Trad‡, and Daniel Sierra*,

ABSTRACT

Simulating seismic wave propagation in complex structures geological areas is a challenging task in exploration geophysics. Realistic simulations of seismic waves in foothill areas face challenges from complex near-surface models that include rough topography, irregular bedrock interfaces, low-velocity surface sediment, and significant heterogeneities. Although existing numerical methods can solve the seismic wave equation under these conditions, they typically require a discretization model with a small grid size, leading to a high computational cost. To address this, we introduce a novel 3D solver (PMFD3D-GPU), based on the finite-difference method for elastic wave propagation in the presence of irregular topography and designed for GPU acceleration. Our solver features an innovative approach that involves an unstructured index array representation (UIAR) to implement the parameter modified (PM) formulation that satisfy the free-surface condition for topographic variations. We validated the PMFD3D-GPU solver against the well-known SPECFEM3D solver, and in conditions of rough topography, we demonstrated misfit errors of less than 1% in most cases. Additionally, our solver accelerates the simulations, achieving a speed-up approximately 20 times faster compared to the CPU implementation. Therefore, PMFD3D-GPU solver enables cost-effective realistic and detailed simulations of near-surface seismic scattering using heterogeneous earth models with irregular topography.

INTRODUCTION

The generation of synthetic seismic data requires a computational application capable of numerically solving the elastic wave equation to simulate seismic waves. The methods commonly used to model seismic waves are the finite difference method (FDM) and the finite element method (FEM). Finite Difference Methods (FDMs), which are effective for larger-scale problems and relatively simple to use, run into difficulties when dealing with complex geometries like interface discontinuities, topographic variations, and the imposition of free boundary conditions (Virieux, 1986; Robertsson, 1996; Hayashi, 2001). This is due to the staircase artifacts produced by the rectangular grid. Furthermore, high-order accuracy generally requires high-order finite difference stencils, which increases both computational cost and complexity (LeVeque, 2007; Durran, 2013). Several modifications of FDM for curved or irregular domains have been proposed to reduce these staircase artifacts (Hestholm and Ruud, 1998; de la Puente et al., 2014; Solano et al., 2016). However, these modifications often complicate the interpretation of the solution and may introduce

*Universidad Industrial de Santander
†Centro de Innovación y Tecnología - ECOPETROL S.A.
‡CREWES - University of Calgary

potential distortion to the solution grid. In contrast, FEM can manage complex geometries and heterogeneous materials (Bathe, 2006; Hughes, 2012). Despite its versatility, it is more computationally demanding and difficult to implement than FDM (Trefethen, 1996; Moczo et al., 2014). Modern extensions of FEM, like the Spectral Element Method (SEM) and the Discontinuous Galerkin Method (DGM), have been developed to overcome some of FEM's limitations. SEM improves accuracy and rate of convergence by approximating the solution within each element with high-degree polynomial basis functions (Komatitsch et al., 1999; Komatitsch and Tromp, 2002; Capdeville et al., 2005). Additionally, SEM is highly parallelizable nature aids in reducing computation time (Martin et al., 2008; Komatitsch et al., 2010). Similar to SEM, DGM can manage discontinuities in the solution, making it particularly suitable for problems with shocks or other sharp features (Käser and Dumbser, 2006; Dumbser and Käser, 2006; Käser et al., 2007). Both SEM and DGM allow for high computational efficiency on parallel computers due to their high degree of parallelism Komatitsch et al. (2010); Breuer et al. (2016). Despite these advantages, they are generally more complex to implement and computationally intensive due to the higher degree of freedom in each element.

Since several methods have been studied, many solvers are available to model elastic waves, each with its unique strengths and limitations in addressing the intrinsic challenges of complex media and complex computational implementation. Some of the most used solvers are presented below:

- **SEISMIC_CPML**: This solver employs a Convolutional Perfectly Matched Layer (CPML) alongside a FDM for the numerical simulation of wave propagation (Komatitsch and Martin, 2007; Martin et al., 2010). However, it does not accommodate media with topographic variations.
- **EWEFD**: This comprises a suite of programs designed to solve the 2D and 3D anisotropic elastic wave equations (Weiss and Shragge, 2013). The programs runs on NVIDIA GPUs, providing a significant computational speed-up. Although the codes are part of the open-source Madagascar software package (`https://www.reproducibility.org`), they are not designed to handle irregular topography.
- **SPECFEM3D**: This is a widely-recognized software package that employs a spectral-element method to solve the wave equation in three dimensions on unstructured meshes (Komatitsch et al., 1999; Komatitsch and Tromp, 1999; Komatitsch et al., 2010). While it can handle complex geometries and irregular topography, SPECFEM3D mesher requires each layer to be homogeneous, which is a limitation in realistic scenarios where elastic parameters are often heterogeneous. External meshers can be incorporated to address this issue. However, open-source meshers like Gmsh (`https://www.gmsh.info/`) struggle with complex models, and more robust meshers, such as CUBIT (`https://www.cubit.sandia.gov/`), come at a charging fees.
- **SeisSol**: This is an open-source, high-performance solver developed by the Geophysics Section of the Department of Earth and Environmental Sciences at LMU Munich. It is particularly suitable for seismic wave propagation, including dynamic rupture processes. SeisSol uses the Arbitrary high-order DERivative (ADER) with

DGM, allowing it to handle complex geometries and scenarios with irregular topography (Käser and Dumbser, 2006; Dumbser and Käser, 2006; Käser et al., 2007) . However, it shares the meshing limitations common to SPECFEM3D.

- **SALVUS**: This is a robust software for simulating waves in intricate 3D geological models by effectively accommodating irregular topography via SEM (Afanasiev et al., 2019). Yet, it is not fully open-source and charges to use it are required.

Due to the lack of available open-source software capable of simulating 3D elastic wave propagation in random heterogeneous media with irregular topography, we developed a solver named PMFD3D-GPU. This solver, based on the Finite Difference Method (FDM), is specifically designed for GPU acceleration. Utilizing an unstructured index array strategy, the solver incorporates the parameter-modified (PM) formulation introduced by Cao and Chen (2018). This formulation satisfies the free-surface condition for arbitrary surface topographies on rectangular grids. The PM formulation has been shown to achieve accuracy results comparable to SPECFEM3D for spatial sampling of 15 grid points per minimum wavelength (Cao and Chen, 2018). In comparisons with SPECFEM3D, PMFD3D-GPU achieved misfit errors close to 1% in a homogeneous model with rough topography. Moreover, compared to the CPU implementation, this solver increases speed-up approximately 20 times faster by leveraging GPU acceleration. Finally, we present a numerical example using the SEAM Foothills Phase II model, demonstrating the capacity of our solver to generate realistic synthetic data.

ELASTIC WAVE EQUATION

In the context of seismic exploration, seismic waves are predominantly modeled as elastic waves. This approximation is due to the predominantly elastic behavior exhibited by Earth's subsurface at the scales and frequencies pertinent to seismic exploration. In such circumstances, rock formations undergo deformation when subjected to stress and consequently return to their original state once the stress is removed (Sheriff, 2002; Yilmaz, 2015).

In the derivation of the elastic wave equation, we follow the approach outlined by Shearer (2009), which entails combining the equation of motion with the constitutive relation for elastic materials. Adopting the index notation, the equation of motion is represented as:

$$\rho \partial_t^2 u_i = \partial_i \sigma_{ij}. \tag{1}$$

Here, u_i denotes the displacement in the $i = \{x, y, z\}$ direction, σ_{ij} represents the stress tensor, and ρ is the material density. The operators ∂_t and ∂_i represent the partial derivatives with respect to time and space, respectively.

On the other hand, the constitutive relation is expressed as:

$$\sigma_{ij} = \lambda \delta_{ij} e_{kk} + 2\mu e_{ij}. \tag{2}$$

In this equation, λ and μ are the Lamé parameters and e_{ij} denotes the strain tensor, which is defined as

$$e_{ij} = \left(\partial_i u_j + \partial_j u_i\right). \tag{3}$$

Substituting equation (3) into (2), the constitutive relation becomes:

$$\sigma_{ij} = \lambda \delta_{ij} \partial_k u_k + \mu \left(\partial_i u_j + \partial_j u_i\right). \tag{4}$$

Subsequently, integrating equation (4) into the equation of motion (1) yields:

$$\rho \partial_t^2 u_i = \partial_i \lambda \partial_k u_k + \partial_j \mu (\partial_i u_j + \partial_j u_i) + \lambda \partial_i \partial_k u_k + \mu \partial_i \partial_j u_j + \mu \partial_j \partial_j u_i \tag{5}$$

Equation (5) govern the elastic wave propagation in heterogeneous media. Alternatively to the index notation, this equation can be expressed in the operator notation as:

$$L_{ij}(\rho, \lambda, \mu, t) u_j(\mathbf{x}, t) = 0, \tag{6}$$

where

$$L_{ij}(\rho, \lambda, \mu, t) = \delta_{ij} \rho(\mathbf{x}) \partial_t^2 - \partial_i[\lambda(\mathbf{x}) \partial_j] - \partial_j[\mu(\mathbf{x}) \partial_i] - \delta_{ij} \partial_k[\mu(\mathbf{x}) \partial_k]. \tag{7}$$

Here, $\mathbf{x}$ denotes the position vector.

Another common approach to express the elastic wave equation for heterogeneous media is using the vector notation as follows:

$$\rho \ddot{\mathbf{u}} = \nabla \lambda (\nabla \cdot \mathbf{u}) + \nabla \mu \cdot \left[\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}}\right] + (\lambda + 2\mu) \nabla \nabla \cdot \mathbf{u} - \mu \nabla \times \nabla \times \mathbf{u}, \tag{8}$$

where $\mathbf{u} = (u_x, u_y, u_z)$ represents the displacement vector, and $\nabla = (\partial_x, \partial_y, \partial_z)$ denotes the nabla operator.

In this study, we will focus on how to solve this equation numerically using the FDM.

REVIEW OF 3D FDM

Several FDM schemes have been proposed for elastic wave modeling (Clayton and Engquist, 1977; Virieux, 1986; Moczo et al., 2014). In this study, we follow the staggered-grid scheme proposed by Virieux (1986) to the first-order velocity-stress form.

First order velocity-stress form

The derivation of the elastic wave equation leads to a second-order partial differential equation. However, this can be alternatively expressed as a system of first-order partial differential equations. This transformation can be accomplished by introducing the velocity vector $\mathbf{v} = \partial\mathbf{u}/\partial t$. By replacing each component of vector $\mathbf{v} = (v_x, v_y, v_z)$ into the equation of motion (1), we obtain:

$$\begin{aligned}
\frac{\partial v_x}{\partial t} =& \frac{1}{\rho}\left(\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z}\right), \\
\frac{\partial v_y}{\partial t} =& \frac{1}{\rho}\left(\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z}\right), \\
\frac{\partial v_z}{\partial t} =& \frac{1}{\rho}\left(\frac{\partial \sigma_{zx}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z}\right).
\end{aligned} \tag{9}$$

In the same way, replacing each component of the velocity vector in equation (4), and subsequently applying time differentiation to both sides of the equations, we get:

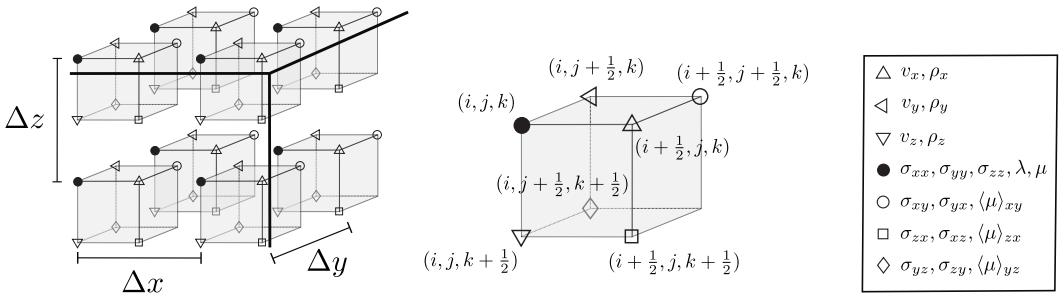
$$\begin{aligned}
\frac{\partial \sigma_{xx}}{\partial t} =& (\lambda + 2\mu)\frac{\partial v_x}{\partial x} + \lambda\frac{\partial v_y}{\partial y} + \lambda\frac{\partial v_z}{\partial z}, \\
\frac{\partial \sigma_{yy}}{\partial t} =& \lambda\frac{\partial v_x}{\partial x} + (\lambda + 2\mu)\frac{\partial v_y}{\partial y} + \lambda\frac{\partial v_z}{\partial z}, \\
\frac{\partial \sigma_{zz}}{\partial t} =& \lambda\frac{\partial v_x}{\partial x} + \lambda\frac{\partial v_y}{\partial y} + (\lambda + 2\mu)\frac{\partial v_z}{\partial z}, \\
\frac{\partial \sigma_{xy}}{\partial t} =& \mu\left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y}\right), \\
\frac{\partial \sigma_{yz}}{\partial t} =& \mu\left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z}\right), \\
\frac{\partial \sigma_{zx}}{\partial t} =& \mu\left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x}\right).
\end{aligned} \tag{10}$$

Equations (9) and (10) are collectively known as the first-order velocity-stress form of the elastic wave equation.

Staggered grid scheme

The staggered grid scheme is widely utilized in geophysical simulations, particularly for modeling seismic wave propagation, primarily due to its improved accuracy, stability, conservation properties, and computational efficiency Moczo et al. (2014). Initially, the 3D space domain is discretized into a numerical grid with spatial steps $\Delta x, \Delta y, \Delta z$, as depicted in Figure 1(a). This scheme then arranges the components of the velocity and stress tensors at varied locations in the numerical grid to achieve more precise and stable results, as displayed in Figure 1(b). Here is a brief overview of the component arrangement:

- The normal stress tensor components σ_{xx}, σ_{yy}, and σ_{zz} are situated at the integral-grid points (i, j, k). The medium parameters (μ, λ, ρ) are also assumed to be located at these points.
- The shear stress tensor components σ_{xy}, σ_{yz}, and σ_{zx} are located at half-grid points, shifted from the integral points (Figure 1(b)).
- Similarly, the velocity components v_x, v_y, v_z are positioned at corresponding half-grid points, also shifted from the integral points (Figure 1(b)).



(a) (b)

FIG. 1. Sattegred grid scheme. (a) 3D grid. (b) 3D cell.

Note that the medium parameters are not defined at half-grid points as they are assumed to be situated at integral grid points (i, j, k). To overcome this, a parameter averaging method is typically utilized to determine the values of the parameters at the half-grid points.

Parameter averaging method

The averaging method proposed by Moczo et al. (2002) is typically used to estimate parameter values at half-grid points. According to this method, the parameter ρ at a half-grid point is calculated as the arithmetic mean of parameter values at the nearest integral grid points, as illustrated below:

$$\rho_x = \frac{1}{2}(\rho_{i,j,k} + \rho_{i+1,j,k}), \quad \rho_y = \frac{1}{2}(\rho_{i,j,k} + \rho_{i,j+1,k}), \quad \rho_z = \frac{1}{2}(\rho_{i,j,k} + \rho_{i,j,k+1}). \tag{11}$$

Here, ρ_x, ρ_y, and ρ_z denote the density values at the locations of the velocity components v_x, v_y, and v_z, respectively, as shown in Figure 1(b). However, in the majority of FDM implementations, the reciprocal value of density is preferred, which is computed at the half-grid points as follows:

$$b_x = \frac{2}{\rho_{i,j,k} + \rho_{i+1,j,k}}, \quad b_y = \frac{2}{\rho_{i,j,k} + \rho_{i,j+1,k}}, \quad b_z = \frac{2}{\rho_{i,j,k} + \rho_{i,j,k+1}}. \tag{12}$$

On the other hand, the parameter μ at a half-grid point is determined as the harmonic mean of parameter values at the nearest integral grid points, defined as:

$$
\begin{aligned}
\langle\mu\rangle_{xy} &= \left[\frac{1}{4}\left(\frac{1}{\mu_{i,j,k}}+\frac{1}{\mu_{i+1,j,k}}+\frac{1}{\mu_{i,j+1,k}}+\frac{1}{\mu_{i+1,j+1,k}}\right)\right]^{-1},\\
\langle\mu\rangle_{yz} &= \left[\frac{1}{4}\left(\frac{1}{\mu_{i,j,k}}+\frac{1}{\mu_{i,j+1,k}}+\frac{1}{\mu_{i,j,k+1}}+\frac{1}{\mu_{i,j+1,k+1}}\right)\right]^{-1},\\
\langle\mu\rangle_{zx} &= \left[\frac{1}{4}\left(\frac{1}{\mu_{i,j,k}}+\frac{1}{\mu_{i+1,j,k}}+\frac{1}{\mu_{i,j,k+1}}+\frac{1}{\mu_{i+1,j,k+1}}\right)\right]^{-1}.
\end{aligned}
\tag{13}
$$

In the above equations, $\langle\mu\rangle_{xy}$, $\langle\mu\rangle_{yz}$, and $\langle\mu\rangle_{zx}$ represent the μ values at the positions of the shear stress tensor components σ_{xy}, σ_{yz}, and σ_{zx} respectively, as indicated in Figure 1(b).

FDM equations

By incorporating the parameter averaging method, the FDM resulting equations are:

$$
\begin{aligned}
D_t^- v_x &= b_x\left(D_x^+\sigma_{xx}+D_y^-\sigma_{xy}+D_z^-\sigma_{zx}\right),\\
D_t^- v_y &= b_y\left(D_x^-\sigma_{xy}+D_y^+\sigma_{yy}+D_z^-\sigma_{yz}\right),\\
D_t^- v_z &= b_z\left(D_x^-\sigma_{zx}+D_y^-\sigma_{yz}+D_z^+\sigma_{zz}\right),
\end{aligned}
\tag{14}
$$

and,

$$
\begin{aligned}
D_t^+\sigma_{xx} &= (\lambda+2\mu)D_x^- v_x+\lambda D_y^- v_y+\lambda D_z^- v_z,\\
D_t^+\sigma_{yy} &= \lambda D_x^- v_x+(\lambda+2\mu)D_y^- v_y+\lambda D_z^- v_z,\\
D_t^+\sigma_{zz} &= \lambda D_x^- v_x+\lambda D_y^- v_y+(\lambda+2\mu)D_z^- v_z,\\
D_t^+\sigma_{xy} &= \langle\mu\rangle_{xy}\left(D_x^+ v_y+D_y^+ v_x\right),\\
D_t^+\sigma_{yz} &= \langle\mu\rangle_{xy}\left(D_y^+ v_z+D_z^+ v_y\right),\\
D_t^+\sigma_{zx} &= \langle\mu\rangle_{xy}\left(D_z^+ v_x+D_x^+ v_z\right),
\end{aligned}
\tag{15}
$$

where D_i^+ and D_i^- are the forward and backward differencing operators with respect to the variable $i=\{x,y,z,t\}$. These operators depend on the order of approximation N of the stencil. For example, operator D_x^+ and D_x^- are defined as:

$$
\begin{aligned}
D_x^+ p_{i,j,k}^n &= \frac{1}{\Delta x}\sum_{\alpha=0}^{N/2-1} W_\alpha\left(p_{i+\alpha+1,j,k}^n-p_{i-\alpha,j,k}^n\right),\\
D_x^- p_{i,j,k}^n &= \frac{1}{\Delta x}\sum_{\alpha=0}^{N/2-1} W_\alpha\left(p_{i+\alpha,j,k}^n-p_{i-\alpha-1,j,k}^n\right).
\end{aligned}
\tag{16}
$$

Similar definitions are used for D_y^+, D_y^-, D_z^+ and D_z^-. The coefficients W_α depends on the order of approximation of the staggered-grid scheme. For the case of fourth-order approximation ($N = 4$), $\mathbf{W} = \left[\frac{9}{8}, -\frac{1}{24}\right]$

Conversely, time differencing operators with second order of approximation are defined as:

$$\begin{aligned} D_t^+ p_{i,j,k}^n =& \frac{1}{2\Delta t}\left(p_{i,j,k}^{n+1/2} - p_{i,j,k}^{n-1/2}\right), \\ D_t^- p_{i,j,k}^n =& \frac{1}{2\Delta t}\left(p_{i,j,k}^{n} - p_{i,j,k}^{n-1}\right), \end{aligned} \tag{17}$$

where Δt is the time step.

Numerical accuracy and stability

The accuracy and stability of numerical solutions obtained through the FDM significantly depend on the spatial step size $(\Delta x, \Delta y, \Delta z)$. Generally, a smaller step size is associated with a more accurate approximation of the differential equation. For a uniform grid, the step size is consistent in all directions and is denoted by Δh. This step size is commonly determined based on the number of points N_λ per minimum wavelength λ_{min} that propagates through the medium, and is given by:

$$\Delta h = \frac{\lambda_{\text{min}}}{N_\lambda} = \frac{V_{\text{min}}}{f_{\text{max}} N_\lambda}, \tag{18}$$

In this equation, f_{max} signifies the maximum frequency used in the simulation, which is typically set by the source, and V_{min} represents the minimum propagation velocity in the model, which is usually dictated by the smallest S-wave velocity. It is critical to choose an appropriate N_λ to maintain accuracy. In the context of the Parameter Modified (PM) method, selecting $N_\lambda = 15$ has been found to yield accuracy levels comparable to those obtained using the SPECFEM3D method (Cao and Chen, 2018). We will look into the specifics of the PM method in subsequent sections.

The choice of step size also influences the stability of the numerical solution of the FDM. If the step size is too large, the numerical solution may become unstable and not converge to the correct solution. This is often regulated by the CFL (Courant-Friedrichs-Lewy) condition, which imposes a certain relationship between the spatial step size and the time step size for stability LeVeque (2007). Following the criterion presented by Lines et al. (1999), the stability of the staggered-grid scheme is ensured when the time step Δt is less than a certain limit Δt_{lim} given by:

$$\Delta t_{\text{lim}} = \frac{1}{2V_{max}\sum_\alpha |W_\alpha| \sqrt{\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} + \frac{1}{(\Delta z)^2}}}. \tag{19}$$

During simulations, we choose our time step such that $\Delta t = \kappa \Delta t_{\text{lim}}$, with a safety factor of $\kappa = 0.99$ to ensure stability (Cao and Chen, 2018).

Model expansion and absorbing boundary conditions

The numerical simulations often involves finite model space. Therefore, Absorbing Boundary Conditions (ABC) are necessary to prevent artificial reflections from the boundaries of the computational domain. The most common ABC methods employed in wave modeling include the sponge ABC, Clayton-Enquist boundary condition, and the Perfectly Matched Layers (PMLs). The sponge ABC, introduced by Cerjan et al. (1985), is the simplest method and requires the least computational resources. Despite its simplicity, its performance is less effective compared to other methods. The Clayton-Enquist boundary condition, proposed by Clayton and Engquist (1977); Engquist and Majda (1977), offers a more effective performance than the sponge ABC. It is particularly beneficial for high-frequency waves and waves incident at steep angles. However, this method requires more computational resources than the sponge ABC, as it relies on a one-sided derivative approximation. Finally, PMLs provide superior absorption performance, even for waves with grazing incidence (Berenger, 1994; Collino and Tsogka, 2001; Komatitsch and Martin, 2007). However, they demand the most computational resources and memory among the three methods, especially in 3D simulations. In our solver, we opted for the sponge ABC due to its straightforward implementation. However, further enhancements could potentially be achieved by incorporating PMLs into our system.

The first step in integrating ABCs is to expand the model, creating a specific region where the ABCs can operate. This process increases the dimensions of the grid points in the model. Specifically, if the original model dimensions are (N_x^0, N_y^0, N_z^0), after expansion they become (N_x, N_y, N_z). As illustrated in Figure 2, the new dimensions are determined as:

$$N_x = N_x^0 + 2N_b, \quad N_y = N_y^0 + 2N_b, \quad N_z = N_z^0 + N_b, \tag{20}$$

where N_b is the number of points in each boundary layer, typically chosen between 20 and 30 points (Yang, 2014). In our implementation, note that the ABCs are not applied to the top of the model (Figure 2). This is because free-surface conditions are implemented at the top of the model.

The expanded model is achieved by resizing each parameter $m = \rho, \lambda, \mu$ as:

$$m_{i,j,k} = m^0_{i-N_b, j-N_b, k} \quad \forall (i,j,k) \notin \Omega_b, \tag{21}$$

where m^0 represents each parameter with the original dimensions (N_x^0, N_y^0, N_z^0), and Ω_b denotes the boundary region defined as:

$$\Omega_b : \left\{ (i,j,k) \in \mathbb{N}_0^3 | N_x^0 + N_b - 1 \leq i < N_b, N_y^0 + N_b - 1 \leq j < N_b, k < N_z^0 \right\}. \tag{22}$$

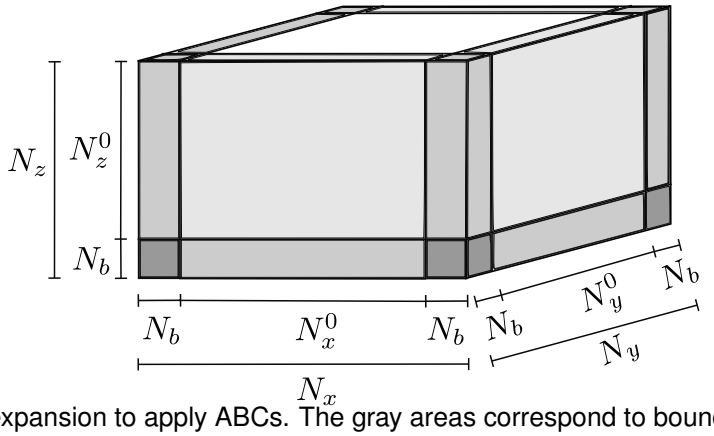


FIG. 2. Model expansion to apply ABCs. The gray areas correspond to boundary region Ω_b.

$$m_{i,j,k} = \begin{cases} m_{0,0,k} & \text{if } i < N_b, j < N_b, \text{ and } k < N_z - N_b, \\ m_{N_x-N_b-1,0,k} & \text{if } i > N_x - N_b - 1, j < N_b, \text{ and } k < N_z - N_b, \\ m_{0,N_y-N_b-1,k} & \text{if } i < N_b, j > N_y - N_b - 1, \text{ and } k < N_z - N_b, \\ m_{N_x-N_b-1,N_y-N_b-1,k} & \text{if } i > N_x - N_b - 1, j > N_y - N_b - 1, \text{ and } k < N_z - N_b, \\ m_{i,j,N_z-N_b-1} & \text{if } k > N_z - N_b - 1, \end{cases} \tag{23}$$

Finally, sponge ABCs are incorporated by gradually decreasing the amplitude of each variable $p = v_i, \sigma_{ij}$. This is done by multiplying each variable with a damping function within the boundary region as:

$$p^n_{i,j,k} = p^n_{i,j,k} D(i,j,k) \quad \forall (i,j,k) \in \Omega_b, \tag{24}$$

where

$$D(i,j,k) = \exp(-\alpha \left[(N_b - i)^2 + (N_b - j)^2 + (N_b - k)^2\right]), \tag{25}$$

is the damping function with a damping factor of α, typically set around 0.015 (Yang, 2014). This function ensures a gradual reduction of wave amplitudes as they approach the boundary, performing absorption and preventing reflections.

Source wavelet

The simulation of wave propagation necessitates the use of a source wavelet, which is a function representing the initial disturbance generating the wave motion through a medium. In seismic simulations, the most commonly used source is the Ricker function, defined as follows (Ricker, 1953):

$$w(t) = \left(1 - 2\pi^2 f_p^2 (t - t_0)^2\right) e^{-\pi^2 f_p^2 (t-t_0)^2}, \tag{26}$$

where f_p is the peak frequency and t_0 is a delay parameter usually chosen to be $t_0 = 1/f_p$. As depicted in Figure 3, this wavelet has a waveform that is symmetric about a peak, and its spectral content is concentrated around the frequency f_p.

Commonly, the wavelet is inserted at a single point (the source location) in the numerical grid for some of the variables v_i, σ_{ij} (Moczo et al., 2014). The wave equation is subsequently solved to simulate the wavefield propagation from the source. When simulating explosive sources, the normal stress components are updated at every time step according to:

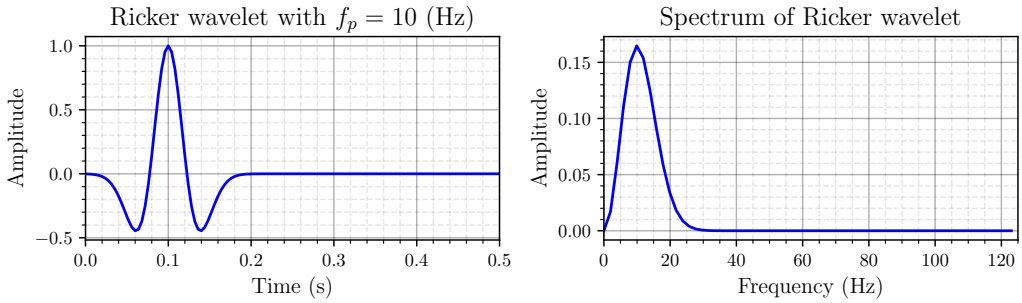


FIG. 3. Ricker wavelet with 10 Hz of peak frequency (left) and its spectrum (right).

$$\sigma_{ii}^n(s_x, s_y, s_z) = \sigma_{ii}^n(s_x, s_y, s_z) + w(n\Delta t) \quad \text{for } i = \{x, y, z\}, \tag{27}$$

where (s_x, s_y, s_z) denotes the source location coordinates. By using equation (27) to inject the source into a homogeneous medium, only compressional P-waves are generated.

It is important to note that the choice of source wavelet and its implementation method can significantly affect the accuracy and stability of FDM simulations. Specifically, the source wavelet sets the bandwidth of the frequencies to be simulated, thereby influencing the spatial step size according to equation (18). Although the Ricker wavelet theoretically contains frequencies up to infinity, the practical frequency range it covers leads to a modification of equation (18) for the case of the Ricker wavelet as follows (Sun et al., 2019):

$$\Delta h = \frac{V_{\text{min}}}{2.5 f_p N_\lambda}. \tag{28}$$

THE PARAMETER MODIFIED METHOD FOR FREE SURFACE CONDITION

The inclusion of the free.surface contition is the most challenging part when modeling elastic waves in media with irregular topography using FDM. This condition states that the stress component normal to the surface is set of zero Moczo et al. (2014). In mathematical terms, the free-surface boundary condition for stress is often written as:

$$\sigma_{ij} n_j = 0, \tag{29}$$

where σ_{ij} is the stress tensor, and n_j is the outward normal vector at the surface topography. For a 3D problem, if we consider the surface to be the plane $z = 0$, this condition simplifies to:

$$\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0. \tag{30}$$

However, Implementing the free-surface condition in FDM simulations involving scenarios with irregular topography brings with it several complexities. Specifically, staircase-shaped discretization is employed to approximate irregular surfaces. This, however, introduces artificial discontinuities that can lead to wave scattering and the generation of spurious reflections. These artifacts can significantly affect the accuracy of the wavefield simulation. To address this issue, various strategies have been explored, including the vacuum formulation (Zahradník et al., 1993; Bohlen and Saenger, 2006), the stress-image method (Levander, 1988; Robertsson, 1996; Hayashi et al., 2001), and medium averaging methods (Mittet, 2002; Xu et al., 2007; Zeng et al., 2012; Cao and Chen, 2018). Among these, the parameter modified (PM) method proposed by Cao and Chen (2018) stands out. It has demonstrated accuracy levels comparable to those achieved by SPECFEM3D For this reason, we implement this method in our solver to satisfy the free-surface condition in arbitrary irregular topography.

The Parameter Modified (PM) method involves adjusting the constitutive relation for the horizontal boundary. With the application of matrix rotations, different constitutive relations emerge for various cases. A more detailed exploration of the theory behind the method can be found in (Cao and Chen, 2018).

In practical terms, the method categorizes the grid cells on the surface based on their relative position to the air. There are five basic classes:

- H: grid cell with air above,
- VR: grid cell with air to the right,
- VL: grid cell with air to the left
- VF: grid cell with air to the front
- VB: grid cell with air to the back

In addition to these basic classes, some transition cells such as inner and outer corner points also exist. An example of the point classification for the $x - z$ plane is depicted in Figure 4.

To facilitate the implementation of PM method, Cao and Chen (2018) put forth generalized expressions for FDM in interior and surface grid cells as:

$$\begin{aligned} D_t^- v_x &= \tilde{b}_x \left(D_x^+ \sigma_{xx} + D_y^- \sigma_{xy} + D_z^- \sigma_{zx} \right), \\ D_t^- v_y &= \tilde{b}_y \left(D_x^- \sigma_{xy} + D_y^+ \sigma_{yy} + D_z^- \sigma_{yz} \right), \\ D_t^- v_z &= \tilde{b}_z \left(D_x^- \sigma_{zx} + D_y^- \sigma_{yz} + D_z^+ \sigma_{zz} \right), \end{aligned} \tag{31}$$

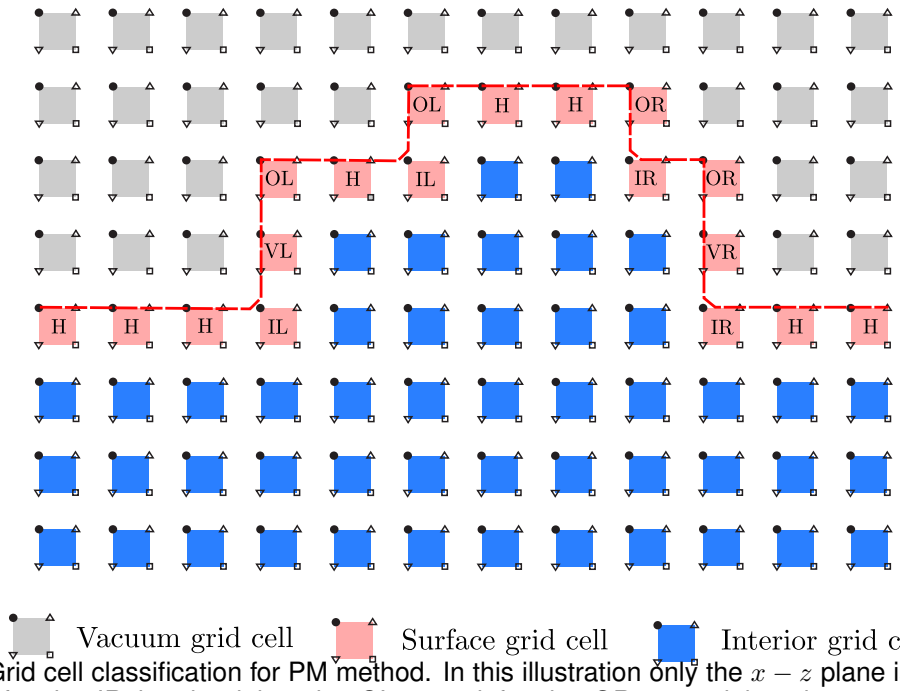


FIG. 4. Grid cell classification for PM method. In this illustration only the $x-z$ plane is shown. IL: interior left point, IR: interior right point, OL: outer left point, OR: outer right point.

and

$$
\begin{aligned}
D_t^+ \sigma_{xx} &= \eta_1^{xx} D_x^- v_x + \eta_2^{xx} D_y^- v_y + \eta_3^{xx} D_z^- v_z, \\
D_t^+ \sigma_{yy} &= \eta_1^{yy} D_x^- v_x + \eta_2^{yy} D_y^- v_y + \eta_3^{yy} D_z^- v_z, \\
D_t^+ \sigma_{zz} &= \eta_1^{zz} D_x^- v_x + \eta_2^{zz} D_y^- v_y + \eta_3^{zz} D_z^- v_z, \\
D_t^+ \sigma_{xy} &= \langle \tilde{\mu} \rangle_{xy} \left(D_x^+ v_y + D_y^+ v_x \right), \\
D_t^+ \sigma_{yz} &= \langle \tilde{\mu} \rangle_{xy} \left(D_y^+ v_z + D_z^+ v_y \right), \\
D_t^+ \sigma_{zx} &= \langle \tilde{\mu} \rangle_{xy} \left(D_z^+ v_x + D_x^+ v_z \right).
\end{aligned}
\tag{32}
$$

The newly defined parameters in these generalized equations are assigned based on the cell classification. In interior points, the parameters are set as follows:

$$
\begin{aligned}
&\tilde{b}_x = b_x, \quad \tilde{b}_y = b_y, \quad \tilde{b}_z = b_z, \\
&\eta_1^{xx} = \eta_2^{yy} = \eta_3^{zz} = \lambda + 2\mu, \\
&\eta_2^{xx} = \eta_3^{xx} = \eta_1^{yy} = \eta_3^{yy} = \eta_1^{zz} = \eta_2^{zz} = \lambda, \\
&\langle \tilde{\mu} \rangle_{xy} = \langle \mu \rangle_{xy}, \quad \langle \tilde{\mu} \rangle_{yz} = \langle \mu \rangle_{yz}, \quad \langle \tilde{\mu} \rangle_{zx} = \langle \mu \rangle_{zx}.
\end{aligned}
\tag{33}
$$

With these values, the generalized FDM equations (31) and (32) are consistent with the original FDM equations (14) and (15).

For surface points, the parameters are adjusted according to the constitutive relation.

A summary of these modifications for each specific case can be found in Table 1. In the following section, we will explain the implementation of this method on a GPU.

IMPLEMENTATION STRATEGY ON A GPU

The Graphics Processing Units (GPUs) are specialized electronic circuits with thousands of cores that can perform operations simultaneously, which makes them especially well-suited for tasks that can be performed in parallel. This characteristic allows for a significant speedup in FDM computations, which can be readily parallelized due to the stencil nature of finite differences. Numerous GPU implementations have been performed to solve the wave equation (Abdelkhalek et al., 2009; Micikevicius, 2009; Michéa and Komatitsch, 2010; Komatitsch et al., 2010; Nakata et al., 2011; Weiss and Shragge, 2013; Rubio et al., 2014; Fabien-Ouellet et al., 2017; Liu et al., 2018; Alkhimenkov et al., 2021). In this subsection, we will present the implmentation of PMFD3D-GPU solver by emplying a subdomain decomposition strategy with unstructured index array representation.

Subdomain decomposition

As discussed in the previous section, the PM method for FDM necessitates the recalculation of medium parameters at surface points. To facilitate the implementation of the PM method, Cao and Chen (2018) proposed generalized expressions for FDM in both interior and surface grid cells (equations (31) and (32)). However, implementing these generalized expressions on a GPU would not be efficient in terms of memory resource management. While the original FDM expressions (equations (14) and (15)) only require storing three field parameters (ρ, μ, λ), the generalized expressions necessitate the storage of fifteen field parameters $(\tilde{b}_x, \tilde{b}_y, \tilde{b}_z, \eta_1^{xx}, \eta_2^{xx}, \eta_3^{xx}, \eta_1^{yy}, \eta_2^{yy}, \eta_3^{yy}, \eta_1^{zz}, \eta_2^{zz}, \eta_3^{zz}, \langle\tilde{\mu}\rangle_{xy}, \langle\tilde{\mu}\rangle_{yz}, \langle\tilde{\mu}\rangle_{zx})$. An alternative approach to circumvent this challenge would involve recalculating the parameter values at each step rather than saving them. However, this alternative would compromise computational efficiency since it would require the classification of each grid cell at every step.

Therefore, we propose a subdomain decomposition strategy for computing FDM simulations that effectively minimizes memory usage while maintaining high computational efficiency. This strategy entails dividing the effective domain (all grid points where medium parameters differ from zero) into two sections: the surface subdomain and the interior subdomain (Figure 5). The surface subdomain, denoted by Ω_S, corresponds to the grid cells situated at the surface. In this domain, we implement the generalized expressions, storing the fifteen field parameters. The interior subdomain, represented by Ω_I, includes grid cells located internally, below the surface subdomain. In this interior subdomain, we implement the original FDM equations, performing parameter averaging at each time step. With this strategy, memory usage is minimized as field parameters are saved only at the surface, and computational efficiency is ensured since grid cell classification is conducted just once.

Note that these subdomains do not necessarily conform to a square shape (Figure 5). As a result, additional procedures must be undertaken to manage the indices of each grid cell within each respective subdomain.

Parameters	H	VR	VL	VF	VB	OP
$\widetilde{b}_x$	b_x	b_x	b_x	$2b_x$	$2b_x$	$2b_x$
$\widetilde{b}_y$	b_y	$2b_y$	$2b_y$	b_y	b_y	$2b_y$
$\widetilde{b}_z$	$2b_z$	$2b_z$	$2b_z$	$2b_z$	$2b_z$	$2b_x$
η_1^{xx}	$\lambda+2\mu$	0	0	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	0
η_2^{xx}	λ	0	0	0	0	0
η_3^{xx}	λ	0	0	$\frac{\mu\lambda}{\lambda+2\mu}$	$\frac{\mu\lambda}{\lambda+2\mu}$	0
η_1^{yy}	λ	0	0	0	0	0
η_2^{yy}	$\lambda+2\mu$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	0	0	0
η_3^{yy}	λ	$\frac{\mu\lambda}{\lambda+2\mu}$	$\frac{\mu\lambda}{\lambda+2\mu}$	0	0	0
η_1^{zz}	λ	0	0	$\frac{\mu\lambda}{\lambda+2\mu}$	$\frac{\mu\lambda}{\lambda+2\mu}$	0
η_2^{zz}	λ	$\frac{\mu\lambda}{\lambda+2\mu}$	$\frac{\mu\lambda}{\lambda+2\mu}$	0	0	0
η_3^{zz}	$\lambda+2\mu$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	$\frac{2\mu(\lambda+\mu)}{\lambda+2\mu}$	0
$\langle\tilde{\mu}\rangle_{xy}$	$\langle\mu\rangle_{xy}$	$\langle\mu\rangle_{xy}$	$\langle\mu\rangle_{xy}$	$\langle\mu\rangle_{xy}$	$\langle\mu\rangle_{xy}$	$\frac{\langle\mu\rangle_{xy}}{2}$
$\langle\tilde{\mu}\rangle_{yz}$	$\langle\mu\rangle_{yz}$	$\frac{\langle\mu\rangle_{yz}}{2}$	$\frac{\langle\mu\rangle_{yz}}{2}$	$\langle\mu\rangle_{yz}$	$\langle\mu\rangle_{yz}$	$\frac{\langle\mu\rangle_{yz}}{2}$
$\langle\tilde{\mu}\rangle_{zx}$	$\langle\mu\rangle_{zx}$	$\langle\mu\rangle_{zx}$	$\langle\mu\rangle_{zx}$	$\frac{\langle\mu\rangle_{zx}}{2}$	$\frac{\langle\mu\rangle_{zx}}{2}$	$\frac{\langle\mu\rangle_{zx}}{2}$

Table 1. Calculation of parameters at the surface points according to grid cell classification of the PM method. OP refers to outer transition points.

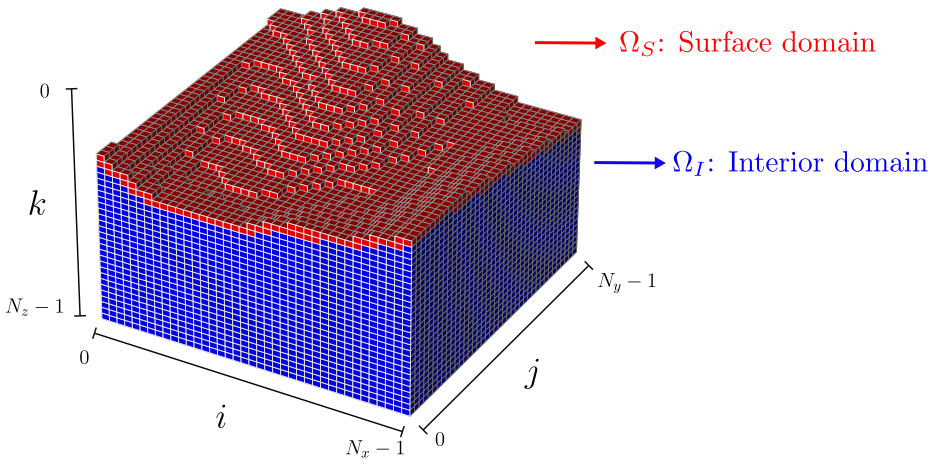


FIG. 5. Subdomain decomposition based on the grid cell classification. The surface subdomain (Ω_S) is represented by red cells, while blue cells illustrate the interior subdomain (Ω_I).

Unstructured index array representation

In structured grid data (such as a square or rectangle), each point in the grid can be accessed by using its row and column indices because all the rows and columns are equally spaced and organized in a regular manner. However, in unstructured or irregularly shaped grids (like the subdomains in Figure 5), this regular access pattern does not exist. This can make it more difficult to perform computations and navigate through the data, especially on parallel processors like GPUs. To overcome this problem, we introduce the Unstructured Index Array Representation (UIAR), a technique that facilitates the representation of indices for each grid cell within subdomains of arbitrary shapes. This means that it provides a way to organize and access data from irregular or complex subdomains in a structured and systematic manner. UIAR involves the creation of an index array that maps the irregular grid points in the subdomain to a structured format.

Specifically, the surface subdomain Ω_S, containing N_s grid cells, is represented by a one-dimensional array S. This array encapsulates the indices of the irregular subdomain in the following manner:

$$S(l) = iN_yN_z + jN_z + k, \quad \forall (i,j,k) \in \Omega_S, \tag{34}$$

Here, the index l spans from 0 to $N_S - 1$, covering all the cells within the subdomain Ω_S.

Similarly, the interior subdomain Ω_I, which includes N_I grid cells, is represented by a one-dimensional array I. This array incorporates the indices of the irregular subdomain as follows:

$$I(m) = iN_yN_z + jN_z + k, \quad \forall (i,j,k) \in \Omega_I, \tag{35}$$

In this case, the index m extends from 0 to $N_I - 1$, accounting for all cells within the subdomain Ω_I.

These one-dimensional arrays (S and I) facilitate efficient data access and manipulation within these irregularly shaped subdomains, providing a systematic representation of otherwise unstructured index data. Consequently, this method enables the effective execution of computations on GPUs. Algorithms 1 and 2 present the kernel pseudocodes for updating particle velocities and the stress tensor at the grid cells within the surface subdomain. Correspondingly, algorithms 3 and 4 exhibit the kernel pseudocodes for updating particle velocities and the stress tensor at the grid cells within the interior subdomain.

Algorithm 1 Kernel for update particle velocities in Ω_S

if $l < N_S$ **then**
  $i = S[l]/(N_y N_z)$
  $j = (S[l] - iN_y N_z)/Nz$
  $k = S[l] - iN_y N_z - jNz$
  $v_x, v_y, v_z \leftarrow$ **velocitySurface**$(\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx}, i, j, k)$ ▷ Equation (31)
end if

Algorithm 2 Kernel for update stress tensor in Ω_S

if $l < N_S$ **then**
  $i = S[l]/(N_y N_z)$
  $j = (S[l] - iN_y N_z)/Nz$
  $k = S[l] - iN_y N_z - jNz$
  $\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx} \leftarrow$ **stressSurface**(v_x, v_y, v_z, i, j, k) ▷ Equation (32)
end if

Algorithm 3 Kernel for update particle velocities in Ω_I

if $m < N_I$ **then**
  $i = I[m]/(N_y N_z)$
  $j = (I[m] - iN_y N_z)/Nz$
  $k = I[m] - iN_y N_z - jNz$
  $v_x, v_y, v_z \leftarrow$ **velocityInterior**$(\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx}, i, j, k)$ ▷ Equation (14)
end if

Workflow implementation

Figure 6 presents the workflow utilized to implement the PMFD3D-GPU solver. As shown, several steps of the workflow are executed on the CPU, including model expansion, domain decomposition, and the calculation of medium parameters in the surface subdomain. Subsequently, the GPU is set up, and model data is transferred from the CPU to the GPU. Once this transfer is complete, the GPU conducts the FDM computations for both the surface and interior subdomains. These computations includes updating the particle velocities and stress tensor components, injecting the source, and applying the Absorbing Boundary Conditions (ABCs). Following these computations, the resulting receiver data are transferred back to the CPU and subsequently written to disk. This procedure is repeated until all time steps are completed.

Algorithm 4 Kernel for update stress tensor in Ω_I

if $m < N_I$ **then**
  $i = I[m]/(N_y N_z)$
  $j = (I[m] - iN_y N_z)/Nz$
  $k = I[m] - iN_y N_z - jNz$
  $\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx} \leftarrow$ **stressInterior**(v_x, v_y, v_z, i, j, k) ▷ Equation (15)
end if

NUMERICAL TESTS

In this section, we aim to assess the efficiency and accuracy of PMFD3D-GPU by conducting two distinct numerical tests. The initial test involves a direct comparison of accuracy with SPECFEM 3D. The second test is designed to demonstrate the capacity of PMFD3D-GPU to generate realistic synthetic data within a portion of the SEAM Foothills Phase II model. We also evaluated the computational performance of PMFD3D-GPU during these tests.

Accuracy validation with SPECFEM3D

In this test, we validate the accuracy of PMFD3D-GPU by conducting elastic wave modeling in a homogeneous model with rough topography, using both PMFD3D-GPU and SPECFEM3D for comparison. The synthetic data obtained from each method is subsequently analyzed and compared.

The homogeneous model, with dimensions and medium parameters specified in Table 2, is discretized uniformly in all directions with a spacing of $\Delta h = 5$ m. We generated the irregular topography via the elevation map function $T(x, y)$, defined as:

$$T(x, y) = 350 + 250 \sin\left(\frac{x}{1000}\right) \cos\left(\frac{y}{1000}\right) \tag{36}$$

This function was then corrupted with normally distributed random noise and subsequently smoothed using a repeated triangle filter, applied over 40 points on each axis. The resulting elevation map can be observed in Figure 7.

V_P (m/s)	V_S (m/s)	ρ (Mg/m^3)	V_P/V_S	Dimension (m)	Grid cell size (m)
3550	2050	2.0	3.5	$3000 \times 2000 \times 2000$	$5 \times 5 \times 5$

Table 2. Medium parameters for homogeneous model.

For synthetic seismic data generation, we established an acquisition geometry with a line of 301 receivers, deployed at intervals of 10 m along the x-axis at $Y = 1000$ m, as illustrated in Figure 7. We also positioned an explosive Ricker source with a peak frequency of 10 Hz at the coordinates (1500,1000) m, at a depth of 20 m.

To carry out modeling with SPECFEM3D, the solver requires an initial setup that specifies the number of spectral elements for the simulation. In this test, we set the solver to

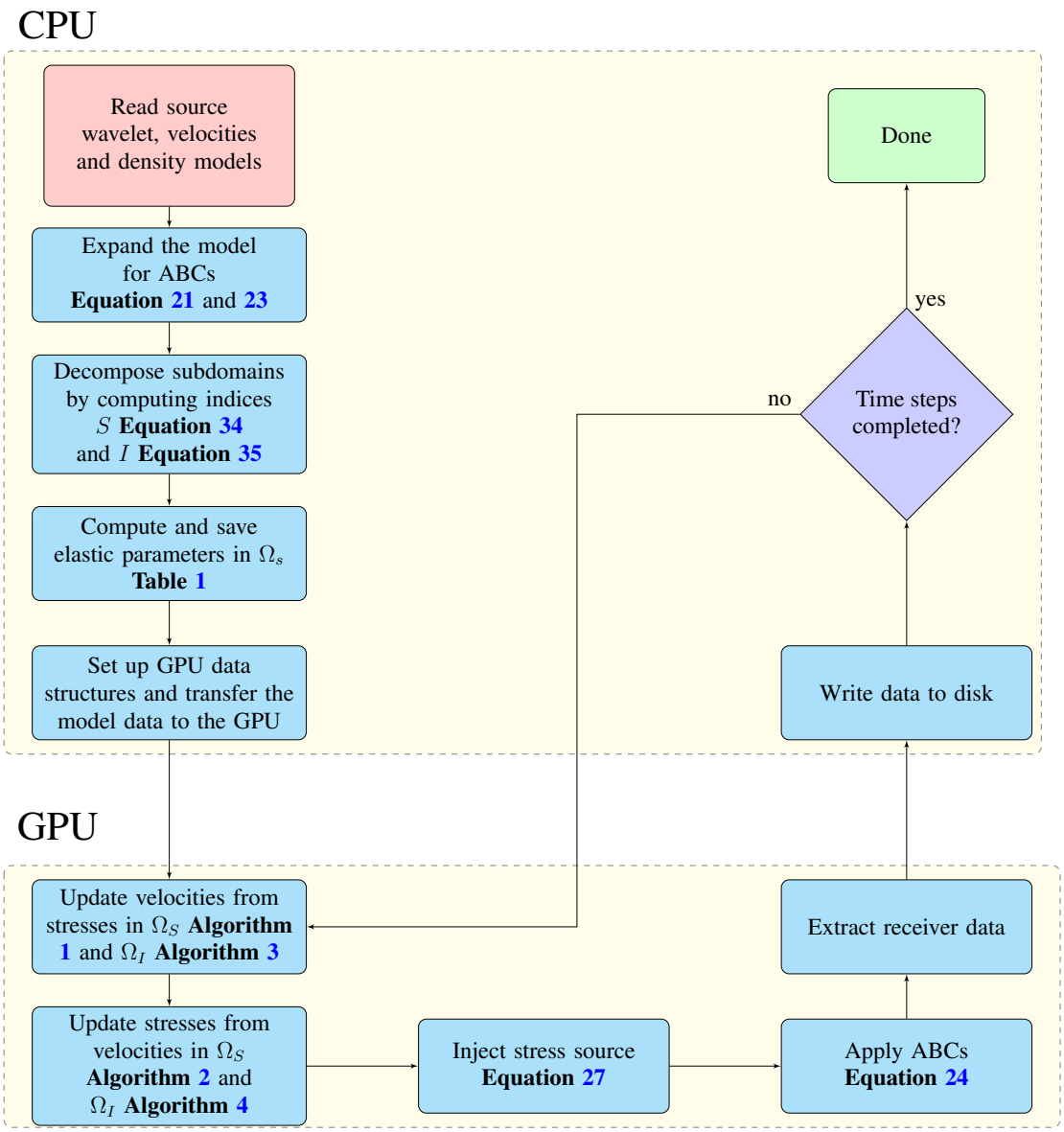


FIG. 6. Workflow used to implement the PMFD3D-GPU solver.

use $96 \times 72 \times 40$ elements to represent the model. SPECFEM3D comes with an internal mesher, XMESHFEM3D, which organizes the geometry of the elements to cover the irregular surface accurately. The resulting mesh for the current model is illustrated in Figure 8.

We conducted simulations for 1.5 seconds using both solvers. The resulting multi-component shot gathers, produced by SPECFEM3D and PMFD3D-GPU, are illustrated in Figures 9(a) and 9(b), respectively. Upon comparison, it's noted that differences between the shots are minimal around the source position and progressively increase as we move away from the source position, as depicted in Figure 9(c).

To further compare the methods, we extracted the traces from four receivers along the receiver line. These receivers, labelled R1, R2, R3, and R4, are positioned at distances

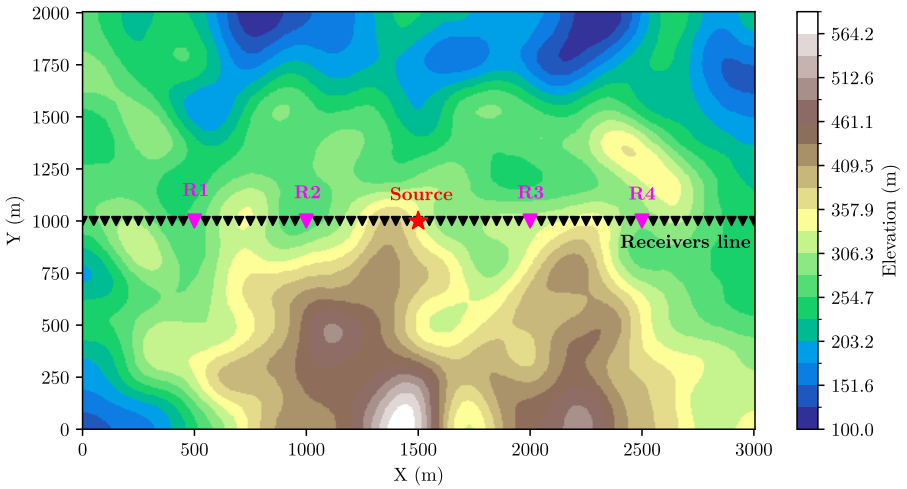


FIG. 7. Elevation map and acquisition geometry. The source is situated at coordinates (1500,1000) m, 20 m beneath the surface. 3C receivers are placed along the surface at Y=1000 m, in a line from 0 to 3000 m, spaced every 10 m. Key receivers R1, R2, R3, and R4 are located at X= 500 m, 1000 m, 2000 m, and 2500 m; respectively.

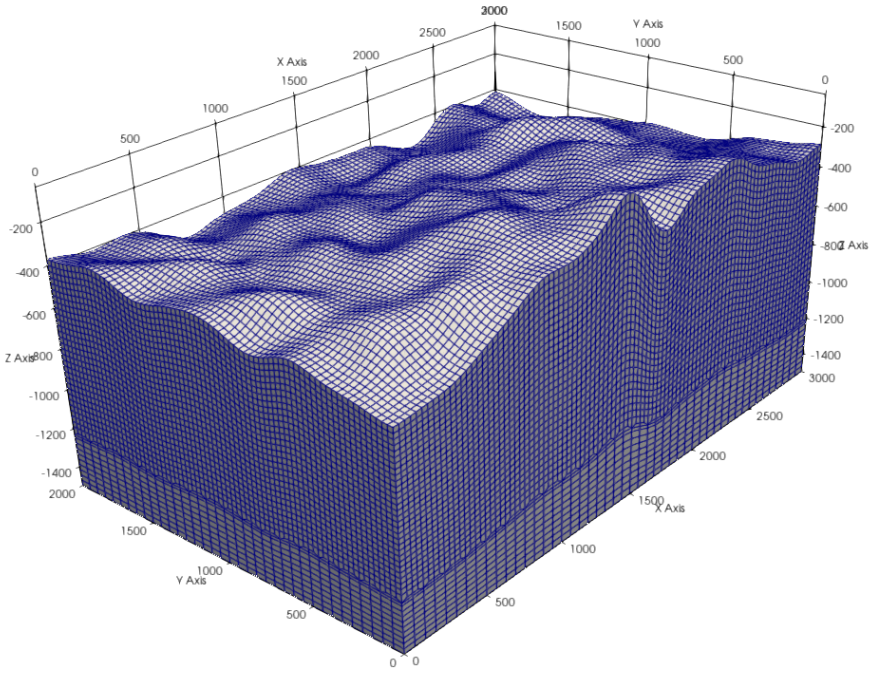
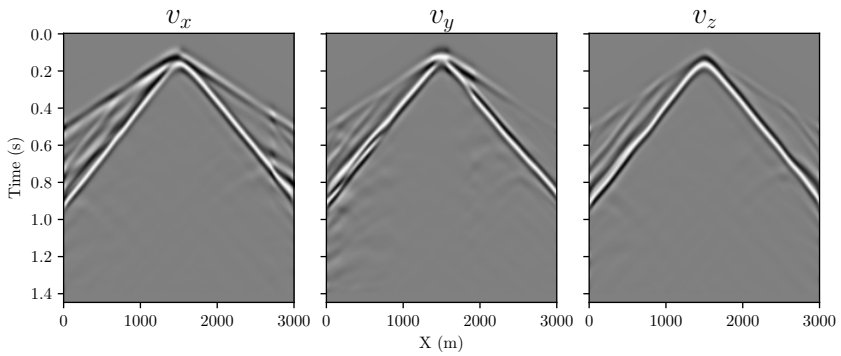


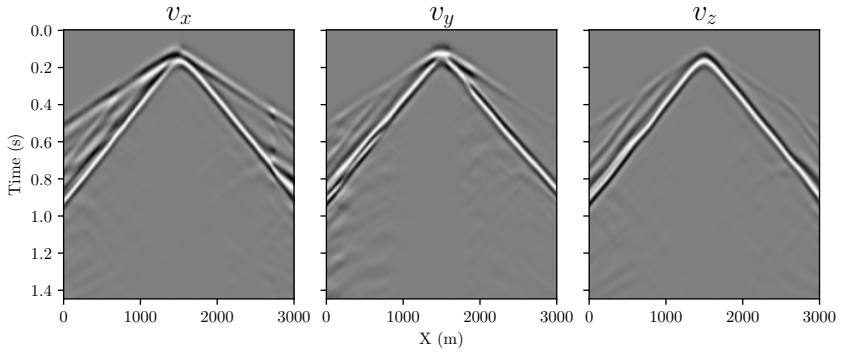
FIG. 8. Mesh produced by XMESHFEM3D, the internal mesher of SPECFEM3D, utilizing $96\times72\times40$ elements to represent the model corresponding to the elevation map shown in Figure 7.

of 500 m, 1000 m, 2000 m, and 2500 m along the x-axis, as shown in Figure 7. The normalized seismic traces for each receiver, obtained from both solvers, are displayed in Figure 10. Visually, the traces appear identical across all components and receivers.

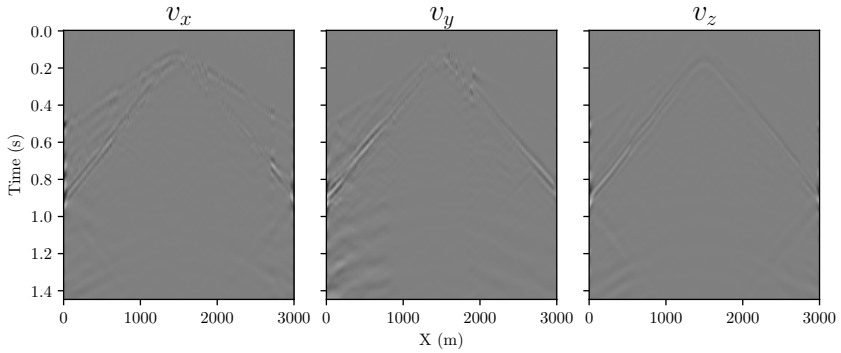
For a more quantitative comparison, we computed the L_2 misfit error between the shot gathers from PMFD3D-GPU and SPECFEM3D, using the following formula:



(a)



(b)



(c)

FIG. 9. Multicomponent shot gather. (a) SPECFEM3D. (b) PMFD3D-GPU. (c) Difference between (a) and (b).

$$L_2 = \frac{\|\mathbf{u} - \mathbf{u}^0\|^2}{\|\mathbf{u}^0\|^2}, \tag{37}$$

where $\mathbf{u}^0$ denotes the shot gather computed by SPECFEM3D, and $\mathbf{u}$ represents the shot gather computed by PMFD3D-GPU. The calculated error percentages for each component from each receiver are presented in Table 3. From these results, it's clear that most of the errors were under 1%. Notably, higher errors were detected in receivers R1 and R4, which are situated furthest from the source. Particularly these errors are more prominent in the horizontal components (along the $x-$ and $y-$axes).

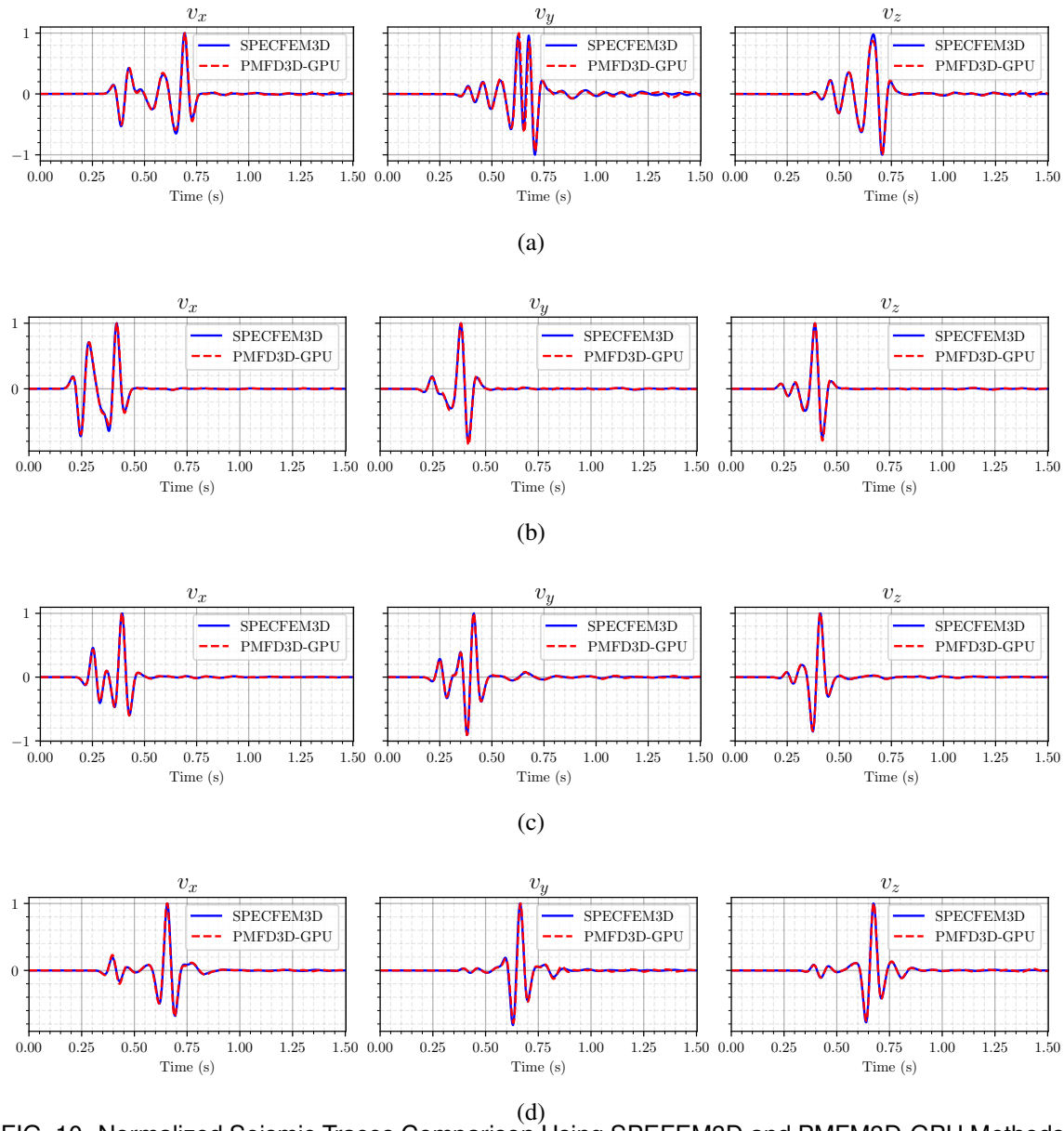


FIG. 10. Normalized Seismic Traces Comparison Using SPEFEM3D and PMFM3D-GPU Methods. (a) R1 at $x = 50$ m, (b) R2 at $x = 100$ m, (c) R3 at $x = 200$ m, and (d) R4 at $x = 250$ m. Solutions from SPEFEM3D are solid blue lines, and PMFM3D-GPU solutions are red dashed lines.

3D SEAM Foothills model example

In this example, we aim to generate realistic seismic synthetic data by performing elastic wave modeling in a portion of the SEAM Foothils Phase II model using PMFD3D-GPU. The chosen portion measures 6000 m $\times$ 2000 $\times$ 4400 m. The model comes discretized uniformly across all axes with $\Delta h = 10$ m. The topographic elevation map for this selected segment is illustrated in Figure 11.

For the acquisition geometry, we placed receivers on the surface along a line stretching from 500 to 5500 m at Y=1000 m, with an inter-receiver distance of 10 m (as seen in Figure 11). Additionally, we positioned an explosive source at coordinates (200,1000) m,

Receiver	v_x (%)	v_y (%)	v_z (%)
R1	1.15	7.77	0.80
R2	0.88	0.97	0.51
R3	0.27	0.42	0.11
R4	1.24	2.25	0.37

Table 3. L_2 misfit norm error of the PMFD3D method compared to the SPECFEM3D method for receivers R1, R2, R3, and R4.

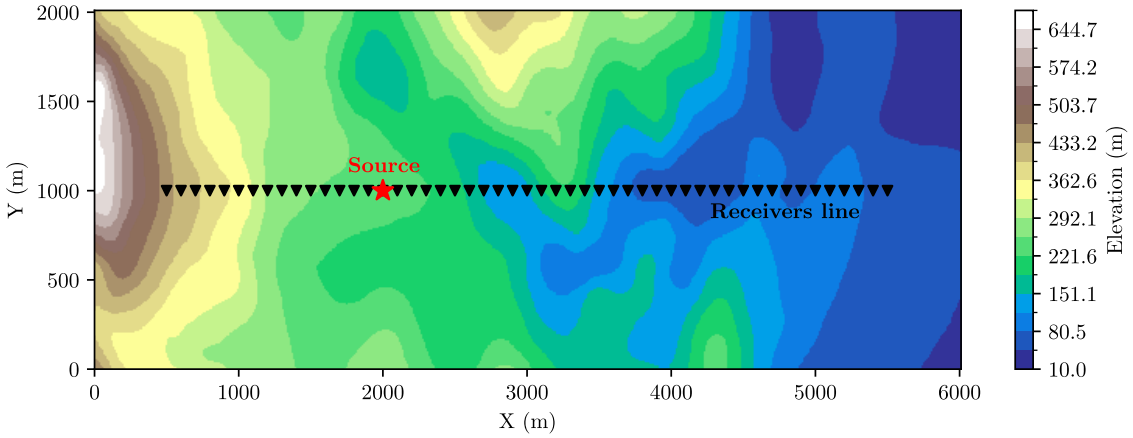


FIG. 11. Elevation map and acquisition geometry for the portion of the SEAM model. The source is positioned at coordinates (200,1000) m, with a depth of 12 m. 3C receivers are aligned on the surface line from 500 to 5500 m at Y=1000 m, with an inter-receiver spacing of 10 m.

at a depth of 12 m. The synthetic data was generated by applying PMFD3D-GPU with 4 seconds of simulation. The resultant multicomponent shot gather is displayed in Figure 12. Here, one can observe realistic seismic events such as Rayleigh waves, near-surface refractions, and near-surface scattering. This result demonstrates the capability PMFD3D-GPU in handling elastic wave modeling in complex geological setting.

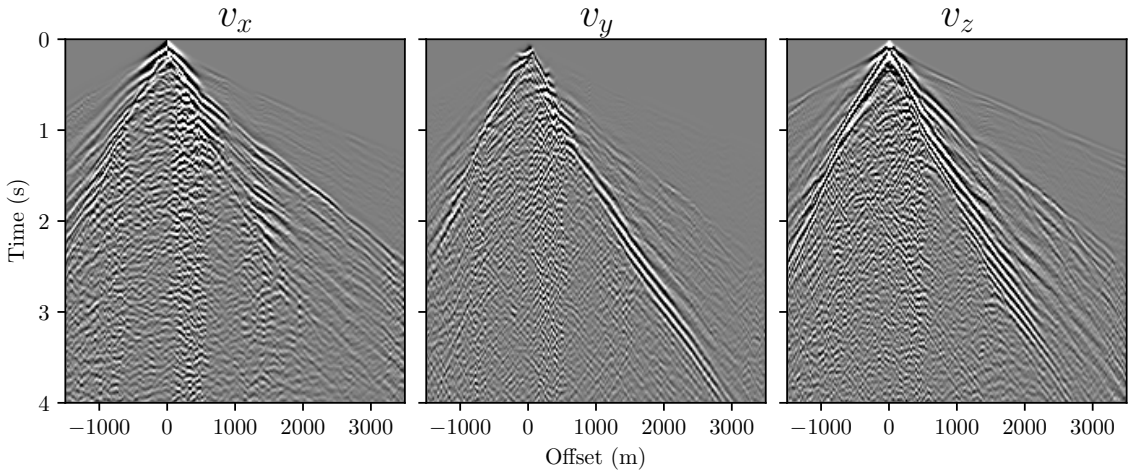


FIG. 12. Multicomponet shot gather SEAM with PMFD3D-GPU.

CPU: Intel Core i7-12700 processor, 12 Core, 2.1GHz to 4.9GHz, 16GB of memory using C and OpenMP **GPU:** NVIDIA GeForce RTX 3060 12 GB of memory using CUDA-C				
Model	Dimension ($N_x \times N_y \times N_z$)	Time samples	CPU (min)	GPU (min)
Homogeneous topography	$(681 \times 481 \times 340)$ $410,341$ surface points $86,335,041$ interior points	2256	115.65	5.47
SEAM portion	$(641 \times 241 \times 461)$ $177,235$ surface points $56,195,513$ interior points	5003	166.61	7.53

Table 4. Comparative analysis of computational performance between the PMFD3D solver implemented on a CPU and the proposed GPU-accelerated implementation (PMFD3D-GPU).

Computational performance

The numerical examples that we have presented show that PMFD3D-GPU is able to perform accurate elastic wave simulations in heterogeneous media. However, the strengths of PMFD3D-GPU extend beyond just accuracy, as it also offers significant computational efficiency. A comprehensive performance analysis, as detailed in Table 4, showcases the time consumption of PMFD3D-GPU in comparison to the language C CPU implementation using OpenMP across various 3D models. When comparing the time spent by PMFD3D-CPU on a uniform grid with that of PMFD3D-GPU on the SEAM model, the GPU implementation is found to be approximately 20 times faster.

DISCUSSION

Numerical modeling of elastic waves in 3D complex media is a computationally demanding task. While many numerical methods are available for undertaking this task, each invariably faces a trade-off between accuracy and computational efficiency, regardless of the chosen method. Within this context, PMFD3D-GPU has proven to be a suitable solver, providing high accuracy results in a relatively short execution time. The GPU-accelerated with CUDA-C implementation strategy significantly speeds up the simulation of the PM method, widely outpacing the CPU implementation. Furthermore, the solver reached accuracy levels comparable to those achieved by SPECFEM3D in the rough topography medium test.

Contrasted with existing solvers, PMFD3D-GPU fills the gap for open-source solvers that are capable of propagating elastic waves in heterogeneous media featuring irregular topography. While useful alternatives do exist, they typically demand additional efforts for application in complex scenarios. For instance, SPECFEM3D is capable of simulating elastic wave propagation with high accuracy, but its internal mesher is not designed to handle arbitrary heterogeneous models. This limitation necessitates the use of external meshers, which could potentially lead to license fees. In contrast, our solver can be effortlessly applied in any scenario without incurring extra costs or efforts.

Although PMFD3D-GPU brings significant benefits, there is scope for further enhancements in various aspects. Currently, we have implemented sponge absorbing boundary con-

ditions (ABCs) due to their ease of implementation and minimal computational resource requirements. However, perfectly matched layers (PMLs) outperform sponge ABCs in terms of absorption performance, and including them would heighten the accuracy of our solver. Implementing PMLs can be computationally expensive in terms of memory as they necessitate the storage of multiple auxiliary fields. However, PML memory allocation strategies as proposed by Páez et al. (2020) could be employed to reduce the memory requirements for simulations. In terms of computational execution time, we could potentially achieve further reduction by adopting a shared memory strategy in the GPU kernel implementation instead of the currently used global memory strategy. Shared memory strategy has been empirically proven to offer much lower latency than global memory owing to its proximity to the cores and its smaller size, as outlined by Cheng et al. (2014). These potential improvements highlight the ongoing possibilities for optimization and increased efficiency in GPU-accelerated numerical modeling.

CONCLUSIONS

We have presented PMFD3D-GPU, a finite-difference solver designed for 3D elastic wave modeling in heterogeneous models with irregular topography. The solver leverages a GPU implementation strategy to incorporate the parameter modified method that satisfy the free-surface condition for varying topographic elevation maps. This strategy includes a subdomain decomposition method featuring an unstructured index array representation to facilitates the managing of indices of grid cells within irregular subdomains shape, improving the computational efficiency. The accuracy of PMFD3D-GPU was validated through numerical testing using a variable topography model, with results similar to those from SPECFEM3D. Our solver demonstrated its capability to generate realistic seismic synthetic data, as exemplified in an exercise with the SEAM Foothills Phase II model. Furthermore, PMFD3D-GPU achieved a significant speedup of the PM method implementation, widely exceeding the performance of the original CPU implementation. In this study, we also discussed potential improvements, such as the application of PMLs to enhance absorption in the ABCs and the use of shared memory strategy to further reduce computational execution time.

ACKNOWLEDGMENTS

This work was carried out within the framework of the Agreement “Acta No. 27 del Convenio de Cooperación Tecnológica 5222395” between Universidad Industrial de Santander and Ecopetrol S.A.- Centro de Innovación y Tecnología. The authors also thank to Emerging Leaders in the Americas Program (ELAP) from Global Affairs Canada for funding the research internship in CREWES-University of Calgary, where this work was developed.

Code availability section

3DPMFD is a solver optimized for simulating seismic waves in terrains with complex topography. This solver leverages GPU acceleration to enhance performance. The source codes are available for downloading at the link: https://gitlab.com/ivanjav/pmfd3d

REFERENCES

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

Afanasiev, M., Boehm, C., van Driel, M., Krischer, L., Rietmann, M., May, D. A., Knepley, M. G., and Fichtner, A., 2019, Modular and flexible spectral-element waveform modelling in two and three dimensions: Geophysical Journal International, **216**, No. 3, 1675–1692.

Alkhimenkov, Y., Räss, L., Khakimova, L., Quintal, B., and Podladchikov, Y., 2021, Resolving wave propagation in anisotropic poroelastic media using graphical processing units (gpus): Journal of Geophysical Research: Solid Earth, e2020JB021,175.

Bathe, K.-J., 2006, Finite element procedures: Klaus-Jurgen Bathe.

Berenger, J.-P., 1994, A perfectly matched layer for the absorption of electromagnetic waves: Journal of computational physics, **114**, No. 2, 185–200.

Bohlen, T., and Saenger, E. H., 2006, Accuracy of heterogeneous staggered-grid finite-difference modeling of Rayleigh waves: Geophysics, **71**, No. 4, T109–T115.

Breuer, A., Heinecke, A., and Bader, M., 2016, Petascale local time stepping for the ader-dg finite element method, *in* 2016 IEEE international parallel and distributed processing symposium (IPDPS), IEEE, 854–863.

Cao, J., and Chen, J.-B., 2018, A parameter-modified method for implementing surface topography in elastic-wave finite-difference modeling: Geophysics, **83**, No. 6, T313–T332.

Capdeville, Y., Gung, Y., and Romanowicz, B., 2005, Towards global earth tomography using the spectral element method: a technique based on source stacking: Geophysical Journal International, **162**, No. 2, 541–554.

Cerjan, C., Kosloff, D., Kosloff, R., and Reshef, M., 1985, A nonreflecting boundary condition for discrete acoustic and elastic wave equations: Geophysics, **50**, No. 4, 705–708.

Cheng, J., Grossman, M., and McKercher, T., 2014, Professional CUDA c programming: John Wiley & Sons.

Clayton, R., and Engquist, B., 1977, Absorbing boundary conditions for acoustic and elastic wave equations: Bulletin of the seismological society of America, **67**, No. 6, 1529–1540.

Collino, F., and Tsogka, C., 2001, Application of the perfectly matched absorbing layer model to the linear elastodynamic problem in anisotropic heterogeneous media: Geophysics, **66**, No. 1, 294–307.

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

Dumbser, M., and Käser, M., 2006, An arbitrary high-order discontinuous galerkin method

for elastic waves on unstructured meshes-ii. the three-dimensional isotropic case: Geophysical Journal International, **167**, No. 1, 319–336.

Durran, D. R., 2013, Numerical methods for wave equations in geophysical fluid dynamics, vol. 32: Springer Science & Business Media.

Engquist, B., and Majda, A., 1977, Absorbing boundary conditions for numerical simulation of waves: Proceedings of the National Academy of Sciences, **74**, No. 5, 1765–1766.

Fabien-Ouellet, G., Gloaguen, E., and Giroux, B., 2017, Time-domain seismic modeling in viscoelastic media for full waveform inversion on heterogeneous computing platforms with opencl: Computers & Geosciences, **100**, 142–155.

Hayashi, K., Burns, D. R., and Toksoz, M. N., 2001, Discontinuous-grid finite-difference seismic modeling including surface topography: Bulletin of the Seismological Society of America, **91**, No. 6, 1750–1764.

Hestholm, S., and Ruud, B., 1998, 3-d finite-difference elastic wave modeling including surface topography: Geophysics, **63**, No. 2, 613–622.

Hughes, T. J., 2012, The finite element method: linear static and dynamic finite element analysis: Courier Corporation.

Käser, M., and Dumbser, M., 2006, An arbitrary high-order discontinuous galerkin method for elastic waves on unstructured meshes i. the two dimensional isotropic case with external source terms: Geophysical Journal International, **166**, No. 2, 855–877.

Käser, M., Dumbser, M., De La Puente, J., and Igel, H., 2007, An arbitrary high-order discontinuous galerkin method for elastic waves on unstructured meshes-iii. viscoelastic attenuation: Geophysical Journal International, **168**, No. 1, 224–242.

Komatitsch, D., Erlebacher, G., Göddeke, D., and Michéa, D., 2010, High-order finite-element seismic wave propagation modeling with mpi on a large gpu cluster: Journal of computational physics, **229**, No. 20, 7692–7714.

Komatitsch, D., and Martin, R., 2007, An unsplit convolutional perfectly matched layer improved at grazing incidence for the seismic wave equation: Geophysics, **72**, No. 5, SM155–SM167.

Komatitsch, D., and Tromp, J., 1999, Introduction to the spectral element method for three-dimensional seismic wave propagation: Geophysical journal international, **139**, No. 3, 806–822.

Komatitsch, D., and Tromp, J., 2002, Spectral-element simulations of global seismic wave propagation-i. validation: Geophysical Journal International, **149**, No. 2, 390–412.

Komatitsch, D., Vilotte, J.-P., Vai, R., Castillo-Covarrubias, J. M., and Sánchez-Sesma, F. J., 1999, The spectral element method for elastic wave equations-application to 2-d and 3-d seismic problems: International Journal for numerical methods in engineering, **45**, No. 9, 1139–1164.

Levander, A. R., 1988, Fourth-order finite-difference P-SV seismograms: Geophysics, **53**, No. 11, 1425–1436.

LeVeque, R. J., 2007, Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems: SIAM.

Lines, L. R., Slawinski, R., and Bording, R. P., 1999, A recipe for stability of finite-difference wave-equation computations: Geophysics, **64**, No. 3, 967–969.

Liu, X., Chen, J., Lan, H., and Zhao, Z., 2018, Seismic wavefield simulation with irregular surface topography and graphic processing unit (gpu) implementation: Journal of Seismic Exploration, **27**, 445–472.

Martin, R., Komatitsch, D., Blitz, C., and Le Goff, N., 2008, Simulation of seismic wave propagation in an asteroid based upon an unstructured mpi spectral-element method: blocking and non-blocking communication strategies, *in* High Performance Computing for Computational Science-VECPAR 2008: 8th International Conference, Toulouse, France, June 24-27, 2008. Revised Selected Papers 8, Springer, 350–363.

Martin, R., Komatitsch, D., Gedney, S. D., and Bruthiaux, E., 2010, A high-order time and space formulation of the unsplit perfectly matched layer for the seismic wave equation using Auxiliary Differential Equations (ADE-PML): Computer Modeling in Engineering and Sciences, **56**, No. 1, 17–42.

Michéa, D., and Komatitsch, D., 2010, Accelerating a three-dimensional finite-difference wave propagation code using gpu graphics cards: Geophysical Journal International, **182**, No. 1, 389–402.

Micikevicius, P., 2009, 3d finite difference computation on gpus using cuda, *in* Proceedings of 2nd workshop on general purpose processing on graphics processing units, 79–84.

Mittet, R., 2002, Free-surface boundary conditions for elastic staggered-grid modeling schemes: Geophysics, **67**, No. 5, 1616–1623.

Moczo, P., Kristek, J., and Gális, M., 2014, The finite-difference modelling of earthquake motions: Waves and ruptures: Cambridge University Press.

Moczo, P., Kristek, J., Vavryčuk, V., Archuleta, R. J., and Halada, L., 2002, 3d heterogeneous staggered-grid finite-difference modeling of seismic motion with volume harmonic and arithmetic averaging of elastic moduli and densities: Bulletin of the Seismological Society of America, **92**, No. 8, 3042–3066.

Nakata, N., Tsuji, T., and Matsuoka, T., 2011, Acceleration of computation speed for elastic wave simulation using a graphic processing unit: Exploration Geophysics, **42**, No. 1, 98–104.

Páez, A., Sánchez, I., and Ramírez, A., 2020, Computational strategies for implementation of 2d elastic wave modeling in gpu: Entre Ciencia e Ingeniería, **14**, No. 28, 52–58.

Ricker, N., 1953, The form and laws of propagation of seismic wavelets: Geophysics, **18**, No. 1, 10–40.

Robertsson, J. O., 1996, A numerical free-surface condition for elastic/viscoelastic finite-difference modeling in the presence of topography: Geophysics, **61**, No. 6, 1921–1934.

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

Shearer, P., 2009, Introduction to Seismology: Cambridge University Press, second edn.

Sheriff, R. E., 2002, Encyclopedic dictionary of applied geophysics.

Solano, C. P., Donno, D., and Chauris, H., 2016, Finite-difference strategy for elastic wave modelling on curved staggered grids: Computational Geosciences, **20**, No. 1, 245–264.

Sun, Y.-C., Ren, H., Zheng, X.-Z., Li, N., Zhang, W., Huang, Q., and Chen, X., 2019, 2-d poroelastic wave modelling with a topographic free surface by the curvilinear grid finite-difference method: Geophysical Journal International, **218**, No. 3, 1961–1982.

Trefethen, L. N., 1996, Finite difference and spectral methods for ordinary and partial differential equations.

Virieux, J., 1986, P-SV wave propagation in heterogeneous media: Velocity-stress finite-difference method: Geophysics, **51**, No. 4, 889–901.

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

Xu, Y., Xia, J., and Miller, R. D., 2007, Numerical investigation of implementation of air-earth boundary by acoustic-elastic boundary approach: Geophysics, **72**, No. 5, SM147–SM153.

Yang, P., 2014, A numerical tour of wave propagation: Madagascar, **2**, No. 4, 1–34.

Yilmaz, Ö., 2015, Engineering seismology with applications to geotechnical engineering: Society of Exploration Geophysicists.

Zahradník, J. í., Moczo, P., and Hron, F. e., 1993, Testing four elastic finite-difference schemes for behavior at discontinuities: Bulletin of the Seismological Society of America, **83**, No. 1, 107–129.

Zeng, C., Xia, J., Miller, R. D., and Tsoflias, G. P., 2012, An improved vacuum formulation for 2d finite-difference modeling of rayleigh waves including surface topography and internal discontinuities: Geophysics, **77**, No. 1, T1–T9.