GEOPHYSICS, VOL. 85, NO. 3 (MAY-JUNE 2020); P. F65–F76, 8 FIGS., 2 TABLES.
10.1190/GEO2018-0760.1

Seismic modeling and inversion using half-precision floating-point numbers

Gabriel Fabien-Ouellet[1]

ABSTRACT

New processors are increasingly supporting half-precision floating-point numbers, often with a significant throughput gain over single-precision operations. Seismic modeling, imaging, and inversion could benefit from such an acceleration, but it is not obvious how the accuracy of the solution can be preserved with a very narrow 16-bit representation. By scaling the finite-difference expression of the isotropic elastic wave equation, we have found that a stable solution can be obtained despite the very narrow dynamic range of the half-precision format.We develop an implementation with the CUDA platform, which, on most recent graphics processing units (GPU), is nearly twice as fast and uses half the memory of the equivalent single-precision version. The error on seismograms caused by the reduced precision is shown to correspond to a negligible fraction of the total seismic energy and is mostly incoherent with seismic phases. Finally, we find that this noise does not adversely impact full-waveform inversion nor reverse time migration, which both benefit from the higher throughput of half-precision computation.

INTRODUCTION

Seismic imaging and inversion are now mostly based on the full-wave equation, through techniques such as full-waveform inversion (FWI) (Virieux and Operto, 2009) or reverse time migration (RTM) (Baysal et al., 1983). The cost of numerically solving the wave equation can be quite high, especially with the recent trends of large 3D seismic surveys and complex physics including elasticity (Raknes and Arntsen, 2015), anisotropy (Warner et al., 2013), and attenuation (Yang et al., 2016; Fabien-Ouellet et al., 2017c).

To cope with the ever-increasing computational cost of such methods, numerical algorithms used for seismic modeling must be able to benefit from hardware evolution. This has been the case with the shift to general purpose GPU computing, which has accelerated significantly the finite-difference time-domain (FDTD) and the spectral-element methods (Micikevicius, 2009; Komatitsch et al., 2010; Fabien-Ouellet et al., 2017b).

The recent boom of machine-learning applications is now a major driving force behind processors and hardware evolution. Because training neural networks can be done with lower precision formats, new accelerators now support half-precision (FP16) storage and arithmetic. For example, Nvidia Tesla P100 and V100 can use half-precision to achieve twice the throughput of single precision. Even larger accelerations are possible with the use of Tensorcores, processing units specialized for matrix multiplication. This shift in favor of FP16 is industry wide and includes nearly all major manufacturers such as Intel, Nvidia, and AMD.

The number of bits used in floating-point representation directly affects the accuracy of numerical algorithms. The standards for scientific computation are the single-precision (FP32) and double-precision format (FP64) (Bailey, 2005). The FP16 format, containing only 16 bits, has long been deemed unfit for scientific computing. However, accuracy requirements are highly variable between algorithms and applications.

For instance, different migration algorithms and even different parts of one migration algorithm may have different precision requirements. Abma (2006) shows that double precision is required for triangle filter antiliasing for Kirchhoff migration, but that single precision can be used with appropriate preprocessing. Levin (2004) concludes that single precision is sufficient for 3D prestack Kirchhoff migration. Fu et al. (2008) investigate different bit reduction strategies for the double-square-root condition used in downward-continued-based migration and conclude that the bit width of a fixed-point format can be reduced from 32 to as low as 10 bits for different operations of the algorithm. Medeiros et al. (2013)

Peer-reviewed code related to this article can be found at http://software.seg.org/2020/0005.
Manuscript received by the Editor 12 November 2018; revised manuscript received 22 November 2019; published ahead of production 19 February 2020; published online 15 April 2020.
[1]Polytechnique Montréal, 2900 Boulevard Edouard-Montpetit, Montréal, Quebec H3T 1J4, Canada. E-mail: gabriel.fabien-ouellet@polymtl.ca (corresponding author).

investigate the use of fixed-point precision for RTM and conclude that 15 bits are required to keep an acceptable signal-to-noise ratio. Finally, Clapp et al. (2010) list bit-width reduction as a possible optimization strategy for RTM.

In this paper, we investigate the use of FP16 for elastic wave propagation in the context of FWI and RTM. We first review the basics of seismic modeling and inversion with the FDTD method. We then explain the main caveats of using the half-precision standard for solving the isotropic wave equation and its adjoint and propose a way to circumvent those issues. We present an implementation of this algorithm in the CUDA language and benchmark it in terms of performance and accuracy. Finally, we compare the results of FP32 and FP16 for FWI and RTM on the Marmousi model.

THEORY

The wave equation

We are interested in solving the wave equation describing the propagation of mechanical waves in a heterogeneous solid for the purpose of imaging and inversion. Without loss of generality, here we adopt the isotropic elastic wave equation in the velocity-stress formulation:

$$\begin{aligned} &\partial_t v_i - \frac{1}{\rho}\partial_j \sigma_{ij} = 0, \\ &\partial_t \sigma_{ij} - (M - 2\mu)\partial_k v_k \delta_{ij} - \mu(\partial_j v_i + \partial_i v_j) = s_{ij}, \end{aligned} \tag{1}$$

in which $v_i(\mathbf{x}, t)$ are the particle velocities in direction i, $\sigma_{ij}(\mathbf{x}, t)$ are the stress components, $s_{ij}(\mathbf{x}, t)$ are the source terms, and $\rho(\mathbf{x})$, $M(\mathbf{x})$, and $\mu(\mathbf{x})$ are the material parameters, respectively, the density, the P-wave modulus, and the S-wave modulus. The velocity-stress formulation can be solved efficiently by the FDTD method on a staggered grid (Virieux, 1986). For the sake of simplicity, we will discuss the 1D wave equation, which exhibits the main difficulties of using half-precision for seismic modeling. Later numerical results will be performed in two dimensions and three dimensions, for which the explicit FD solutions are given by Virieux (1986) and Graves (1996), respectively. In one dimension, we have

$$\begin{aligned} v_x^{-1/2} &= 0, \\ \sigma^0_{x+1/2} &= 0, \\ v_x^{t+1/2} &= v_x^{t-1/2} + \Delta t \rho_x^{-1} D^+ \sigma^t_{x+1/2}, \\ \sigma^{t+1}_{x+1/2} &= \sigma^t_{x+1/2} + \Delta t M_{x+1/2} D^- v_x^{t+1/2} + \Delta t s^t_{x+1/2}. \end{aligned} \tag{2}$$

The grids are staggered in space and time. The superscript t and subscript x specify the position of a grid point on the time and space axis, respectively. Staggered grid points are indicated by noninteger indices, for example, $x + 1/2$. Finally, we left the spatial difference operators $D^{\pm}$ arbitrary for brevity. The solution is obtained by updating the grid positions in the classic leapfrog approach, that is, velocities and stresses are updated sequentially.

Imaging and inversion

The goal of imaging is to recover the variations of the earth parameters or the structure of the subsurface, whereas the goal of inversion is to recover the true magnitude of those parameters. In other words, we want to recover ρ, M, and/or μ given measurements d_{obs}, which can be particle velocities v_i or pressure. To do so, we define an optimization problem, with the objective of minimizing the difference between the observed and modeled wavefields:

$$\begin{aligned} \underset{\boldsymbol{\rho},\mathbf{M}}{\text{minimize}} \quad & \mathcal{X} = \tfrac{1}{2}\sum_s \sum_t (d^{s,t}_{\text{mod}} - d^{s,t}_{\text{obs}})^2 \\ \text{subject to} \quad & \text{equation 1}, \end{aligned} \tag{3}$$

where $d^{s,t}_{\text{mod}}$ and $d^{s,t}_{\text{obs}}$ are, respectively, the modeled and observed data for source s at time t. FWI is a (discretized) partial differential equation (PDE) constrained optimization problem that is usually solved with local gradient-based optimization algorithms. The computation of the gradient is performed efficiently by the adjoint state method, which necessitates solving two discretized systems: the discretized forward wave equation (e.g., equation 2) and its adjoint. The adjoint equation has the same complexity as the forward equation. For example, the adjoint of the 1D wave equation for pressure measurements (p_{obs}) is

$$\begin{aligned} \tilde{\sigma}_x^{T+1} &= 0, \\ \tilde{v}_{x+1/2}^{T+1/2} &= 0, \\ \tilde{\sigma}^t_x &= \tilde{\sigma}^{t+1}_x - \Delta t D^+ (M_{x+1/2} \tilde{v}^{t+1/2}_{x+1/2}) + r^t_x, \\ \tilde{v}^{t-1/2}_{x+1/2} &= \tilde{v}^{t+1/2}_{x+1/2} - \Delta t D^- (\rho_x^{-1} \tilde{\sigma}^t_x). \end{aligned} \tag{4}$$

Note that time propagation is reversed in time and begins at the final time T. The adjoint source r corresponds to the residual $r = p_{\text{mod}} - p_{\text{obs}}$ in the case of FWI based on the l_2 norm and corresponds to the recorded data $r = p_{\text{obs}}$ in the case of RTM. Note that equations 2 and 4 have the same structure, have the same order of accuracy, and thus have the same sensitivity to numerical noise.

Once the solution of the forward and the adjoint PDEs is computed, the gradient (or the image for migration) can be computed with an imaging condition that relies on the crosscorrelation in time of both solutions. For example, the gradient of the 1D wave equation with respect to M is

$$\nabla_M \mathcal{X}_x = \Delta t \sum_s \sum_t \tilde{v}^{s,t+1/2}_{x+1/2} D^- v_x^{s,t+1/2}. \tag{5}$$

Specifics of this correlation depend on the particular imaging condition, the parameterization, and the cost function. In all cases, the gradient is the sum of thousands of samples and hundreds of sources. Any noncoherent noise contained in the solution of the forward or the adjoint will be attenuated by this stacking operation. It is thus conceivable to reduce the accuracy of the PDE solution without affecting too much the final image or search direction, as long as the noise it produces does not bias the solution; that is, it does not introduce numerical dispersion.

Floating-point precision

The numerical solution of the FD approximation of the wave equation is usually computed with a floating-point representation of real numbers. In modern computers, floating-point numbers usually follow the IEEE 754 standard (Zuras et al., 2008). A floating-point representation is comprised of a fixed number of bits, divided into three categories: the sign bit, the exponent bits e, and the significant (or mantissa) bits p. The bit layout for the IEEE half-precision format is presented in Figure 1. Any real number can be approximated by a floating-point representation $\pm p \times 2^e$, in which the significant p and exponent e are integers of infinite precision. However, with integers containing a finite number of bits, most real numbers can only be approximated. The error between the true value and its finite floating-point representation is caused by three phenomena:

1) Rounding errors: A real number can lie between two exactly representable numbers of a particular floating-point format; that is, it needs more digits than can be contained in the significant. The number must then be rounded to one or the other.
2) Overflow: The maximum integer representable by the exponent bits emax constrains the maximum value of a floating-point format, given by $(2 - 2^{1-p}) \times 2^{\mathrm{emax}}$. Overflow happens when a number exceeds this value.
3) Underflow: On the other end of the spectrum, the minimum value that can be represented by a normal number is given by $1 \times 2^{\mathrm{emin}}$, where emin is the smallest value of the exponent. Below this value, 0 or the nearest subnormal number is assigned.

The precision of the half, single, and double IEEE standards is presented in Table 1. As can be seen, halving the number of bits more or less halves the number of significant decimals — the precision is proportional to the number of bits of the significant. However, the dynamic range drastically decreases between single and half-precision, which is due to the exponential relationship between dynamic range and the exponent bits. When using half-precision, all of the computed values must stay within a very narrow range, which may be impossible for many algorithms.

METHODS

Scaling the wave equation

Solving the wave equation by the FDTD method in half-precision requires careful considerations. In fact, naively converting every variable from single to half in the modeling algorithms cannot work. To illustrate this point, we use the 1D wave equation (equation 2). Consider the material parameters: The velocity updates require stress derivatives to be multiplied by $\Delta t \rho^{-1}$, and the stress updates require velocity derivatives to be multiplied by $\Delta t M$. However, the scales of those terms are problematic. Take $\rho = 2000\ \mathrm{kg/m^3}$, $V_\mathrm{P} = 3500\ \mathrm{m/s}$, and $\Delta t = 10^{-4}\mathrm{s}$, very reasonable values. Then, $\Delta t M = 2{,}450{,}000$, which is much larger than 65,504, the maximum FP16 value. Similarly, $\Delta t \rho^{-1} = 5 \times 10^{-8}$, which is much smaller than the smallest normal number 6.1×10^{-5}. It is thus impossible to compute directly the FD solution of the elastic wave equation in FP16.

To solve this problem, we propose to rescale equations 2 and 4, so that each term is normalized around unity. We first define three scaling factors:

$$\begin{aligned} e_v &= -\log_2(\Delta t \max(M)), \\ e_s &= -\log_2(\Delta t \max(s)), \\ e_r &= -\log_2(\Delta t \max(r)). \end{aligned} \tag{6}$$

The first factor e_v takes the maximum value of the heterogeneous P-wave modulus, which is the highest numerical value taken by any material parameters for the elastic wave equation. The two other factors depend on the maximum value of the source signal and of the residuals, respectively. In the code, the scaling happens before computations. To do so, we pack the material parameters into scaled coefficients and we define the normalized sources and residuals:

$$\begin{aligned} a_x &= 2^{-e_v} \Delta t \rho_x^{-1}, \\ b_{x+1/2} &= 2^{e_v} \Delta t M_{x+1/2}, \\ s'^{t}_{x+1/2} &= 2^{e_s} \Delta t s^{t}_{x+1/2}, \\ r'^{t}_{x+1/2} &= 2^{e_r} \Delta t r^{t}_{x+1/2}. \end{aligned} \tag{7}$$

We can solve the transformed forward PDE of equation 2:

$$\begin{aligned} v_x'^{t+1/2} &= v_x'^{t-1/2} + a_x D^+ \sigma'^{t}_{x+1/2}, \\ \sigma'^{t+1}_{x+1/2} &= \sigma'^{t}_{x+1/2} + b_{x+1/2} D^- v_x'^{t+1/2} + s'^{t}_{x+1/2}, \end{aligned} \tag{8}$$

and the transformed adjoint PDE:

$$\begin{aligned} \tilde{\sigma}'^{t}_{x} &= \tilde{\sigma}'^{t+1}_{x} - D^+ (b_{x+1/2} \tilde{v}'^{t+1/2}_{x+1/2}) + r'^{t}_{x}, \\ \tilde{v}'^{t-1/2}_{x+1/2} &= \tilde{v}'^{t+1/2}_{x+1/2} - D^- (a_x \tilde{\sigma}'^{t}_{x}). \end{aligned} \tag{9}$$

The primes indicate that the variables are scaled. Using the linearity of the wave equation with respect to sources, the original solution to the unscaled PDEs can be recovered by rescaling the velocities and the stresses:

$$\begin{aligned} v &= 2^{e_v - e_s} v', \\ \sigma &= 2^{-e_s} \sigma', \\ \tilde{v} &= 2^{e_v - e_r} \tilde{v}', \\ \tilde{\sigma} &= 2^{-e_r} \sigma'. \end{aligned} \tag{10}$$

This must also be applied to the gradient expression. Equation 5 then becomes

$$\nabla_M \mathcal{X}_x = 2^{2e_v - e_s - e_r} \Delta t \sum_s \sum_t \tilde{v}'^{s,t+1/2}_{x+1/2} D^- v_x'^{s,t+1/2}. \tag{11}$$

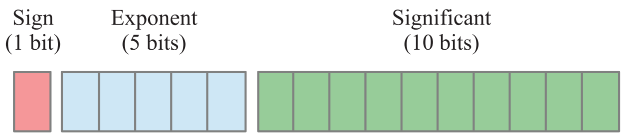

Figure 1. Bit description of the IEEE half-precision standard.

Table 1. IEEE floating-point formats and their precision.

Format	Bits	Min	Max	Decimals
Half	16	$\sim 6.1 \times 10^{-5}$	65,504	~3.3
Single	32	$\sim 2.2 \times 10^{-38}$	$\sim 3.4 \times 10^{38}$	~7.2
Double	64	$\sim 10^{-308}$	$\sim 10^{308}$	~15.8

Observe that the scaled parameters a and b now have magnitudes commensurate to the dynamic range of half-precision. Using the same values as before ($\rho = 2000 \mathrm{kg/m}^3$, $V_\mathrm{P} = 3500$ m/s, and $\Delta t = 10^{-4}$ s), $a = 0.1225$ and $b = 1$ for spatially constant parameters. Scaling the sources and the adjoint sources further guarantees that σ and v will stay within the range of half-precision. The scaling approach is directly applicable in two dimensions and three dimensions, for the elastic equation (equation 1). In that case, a scaled parameter for μ, $c = 2^{e_v} \Delta t \mu$ is also required.

Algorithm 1 summarizes the half-precision scaling strategy. Notice that the scalers are simple constants and that the scaled parameters are computed once for all sources (lines 2 and 3). The computation overhead of this operation is thus minimal. Base 2 scaling allows us to preserve the accuracy of the scaled values because this only modifies the exponent bits of the floating-point representation, leaving the significant intact. Using the precomputed scaled parameters, time stepping (line 4) and adjoint time stepping (line 12) involve the same number of operations as the unscaled PDEs. Finally, descaling is required to produce the right output in the form of the computed wavefield (line 7) or the gradient (line 14). In short, by scaling the parameters before computation then descaling the solution, the elastic wave equation can be solved by FD in FP16 precision.

Algorithm 1. Half-precision seismic modeling and imaging.

```
1: **Inputs**: $\rho$, $V_\mathrm{P}$, $V_\mathrm{S}$, $s$, $d_\mathrm{obs}$
2: Compute scalers $e_v$, $e_s$ with equation 6
3: Compute scaled parameters $a$, $b$, $c$, $s'$ with equation 7
4: Time stepping with equation 8, storing $d'_\mathrm{mod}$
5: Obtain $d_\mathrm{mod}$ by rescaling $d'_\mathrm{mod}$ with equation 11
6: **if** modeling only **then**
7:     **Output** $d_\mathrm{mod}$
8: **else**
9:     Compute the adjoint source, that is, $r = d_\mathrm{mod} - d_\mathrm{obs}$
10:    Compute the scaler $e_r$ with equation 6
11:    Compute the scaled residuals $r'$ with equation 7
12:    Adjoint time stepping with equation 9
13:    Compute the gradient or image (equation 11)
14:    **Output** $\nabla \mathcal{X}$ (or the image)
```

RESULTS

We have implemented the scaled modeling algorithm in CUDA, which allows computations on Nvidia GPUs. Our choice is motivated by the recent support on the latest GPU models of half-precision storage and arithmetic. A particularity of working in half-precision with Nvidia's GPUs is that reading/writing from global memory and arithmetic operations should be carried out on vector types (half2) to reach the full throughput attainable with FP16. To do so, we divided the number of threads in the fast dimension (z) by two, assigning to each thread the update of two consecutive grid points in z. This strategy is similar to Ho and Wong (2017). For further technical details, we refer the reader to our code, which is accessible through GitHub.

Performance

Conversion from FP16 to FP32 leads directly to a reduction of memory consumption by half. On its own, this is a significant improvement, considering the enormous memory requirements of 3D FWI. However, the reduced precision of FP16 over FP32 is only acceptable if it also comes with a significant reduction in runtime. We here quantify the gain in performance by comparing three different versions of our code: (1) the baseline FP32 version, in which all the operations are performed in FP32, (2) the FP16 IO version, which reads material parameters and seismic variables in half-precision from global to shared memory, converts to floats, computes the update, converts back to half-precision, and writes the updated values to global memory, and (3) the FP16 COMP version, which performs every operation in half-precision (storage and arithmetic). The three versions are identical and use preprocessor directives to change between half and float types.

To compute the acceleration of our different implementations, we measured the time required for time stepping (line 4 of algorithm 1). The acceleration is defined as the ratio between runtime in FP32 and runtime for the FP16 versions. We measured runtimes for 2D square and 3D cubic models, with edges between 2048 and 7744 points, every 64 points in two dimensions, and edges between 256 and 544 points every eight points in three dimensions. For each size, three shots with 1000 time steps were computed. This measure was performed for different Nvidia GPU models: the Tesla K40, the Tesla M40, only supporting half-precision storage, and Tesla P100 and the Tesla V100, supporting full half-precision reading/writing and arithmetic.

The average acceleration of all model sizes along their standard deviation is presented in Figure 2. The acceleration of FP16 IO is

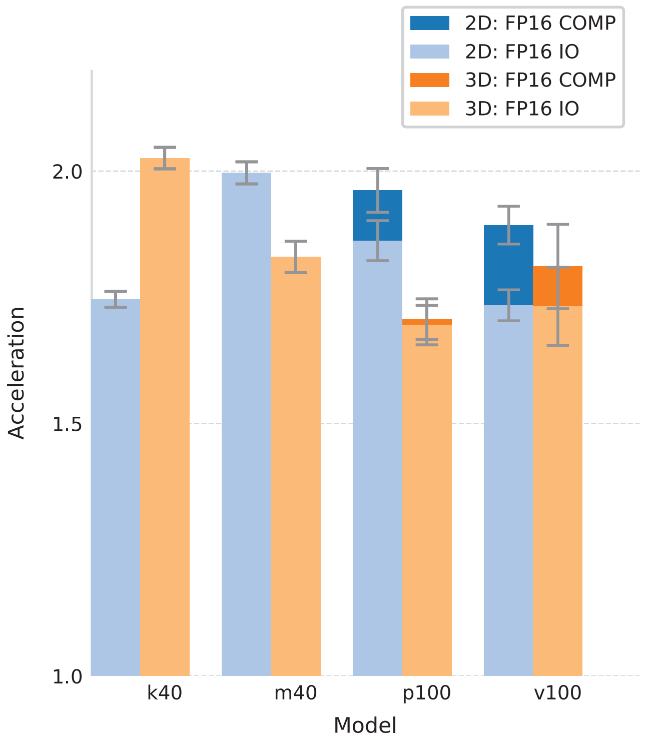

Figure 2. Acceleration obtained by using half-precision over single precision for different GPU architectures. Uncertainty bars indicate ± a standard deviation observed over all model sizes.

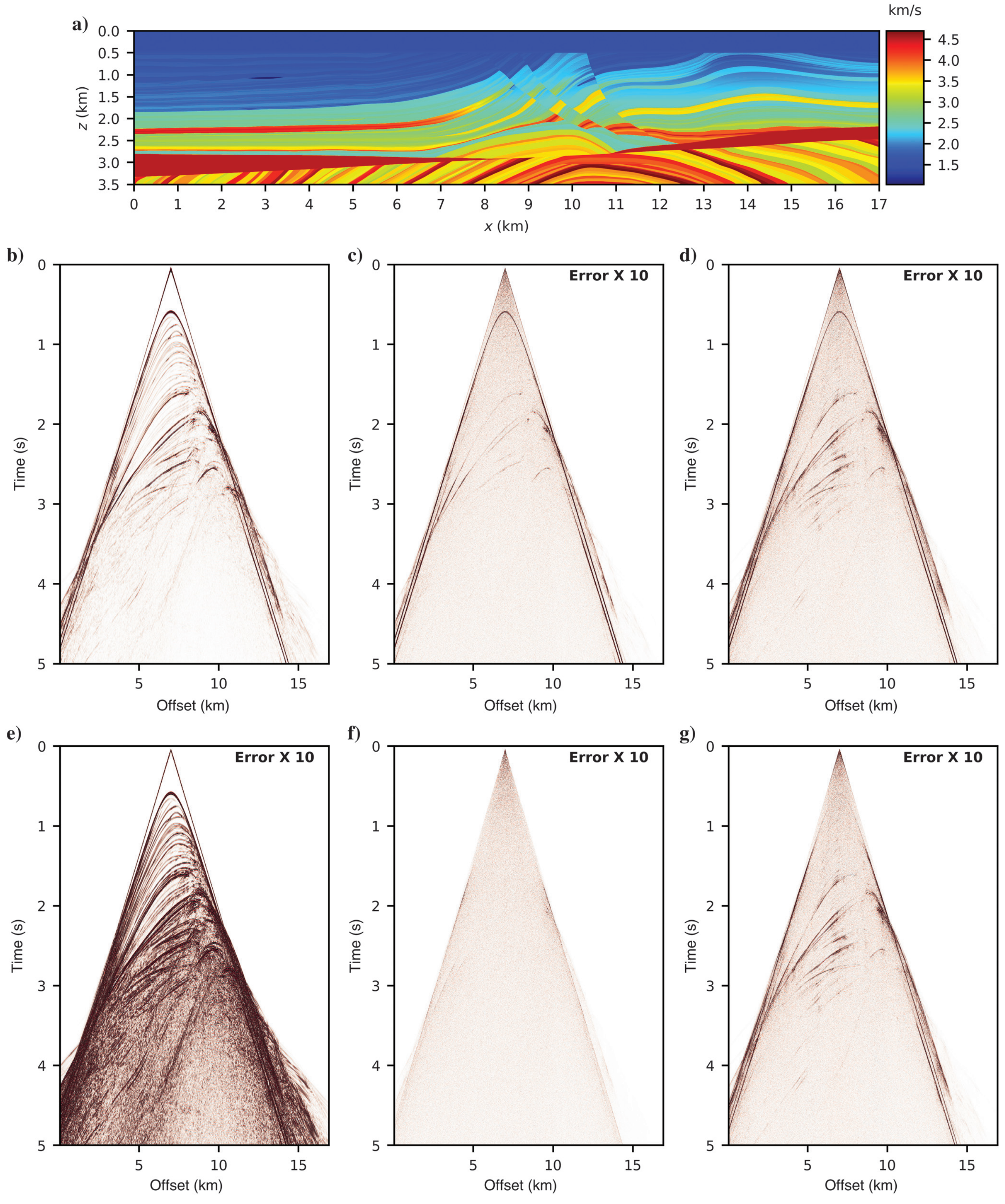

Figure 3. Comparison between modeling with FP16 and FP32 for the elastic Marmousi II model: (a) the V_P velocity, (b) the FP32 shot gather, (c) the error of FP16 IO, (d) of FP16 COMP, (e) the error between FP32 modeling with the Taylor and Holberg coefficients, and the error between FP32 with models truncated to (f) FP16 and FP16 IO and (g) FP1632.

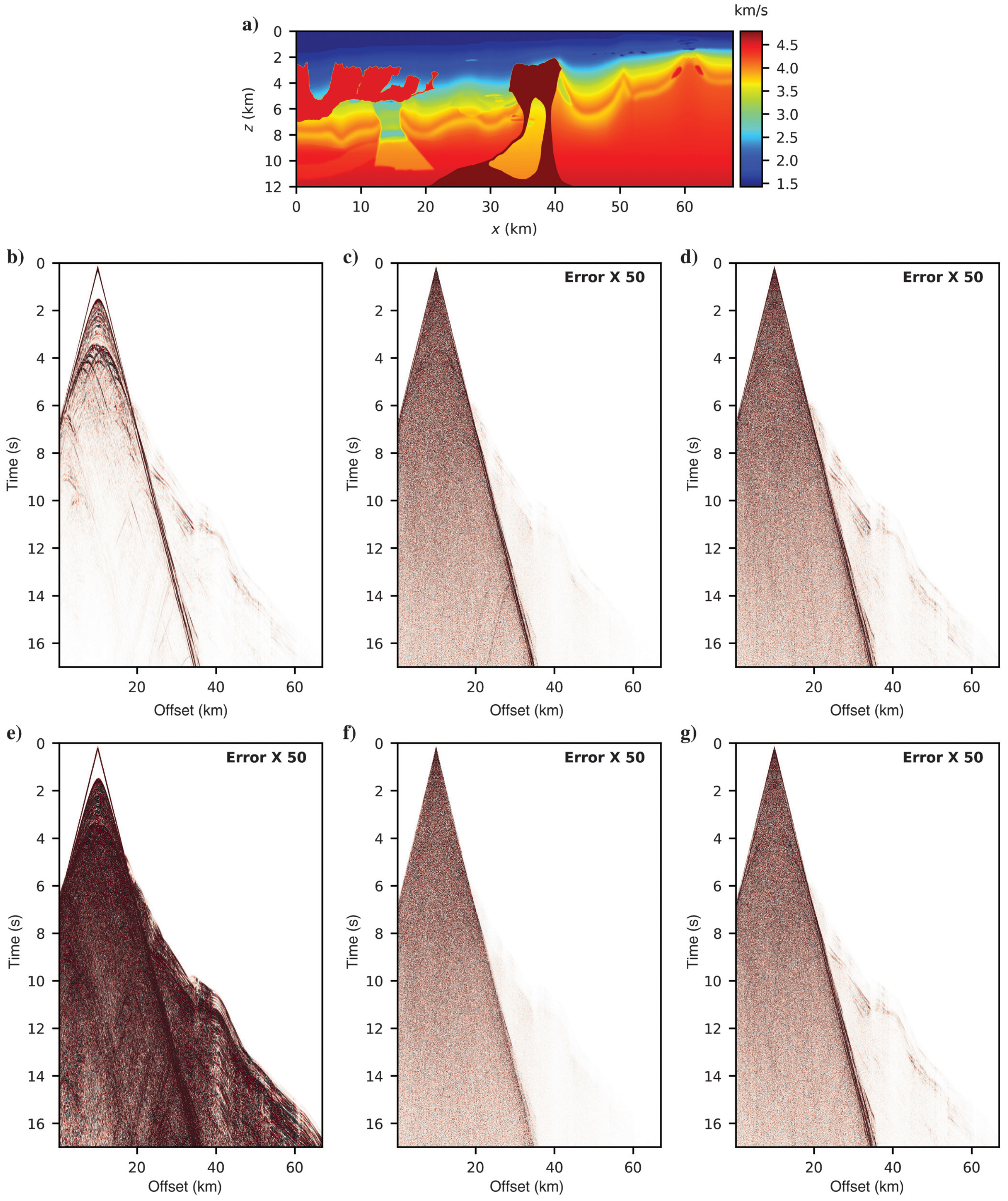

Figure 4. Comparison between modeling with FP16 and FP32 for the 2004 BP velocity benchmark model: (a) the V_P velocity, (b) the FP32 shot gather, (c) the error of FP16 IO, (d) of FP16 COMP, (e) the error between FP32 modeling with the Taylor and Holberg coefficients, and the error between FP32 with models truncated to (f) FP16 and FP16 IO and (g) FP16 COMP.

between 1.7× up to over 2×, in two dimensions and three dimensions. In all cases, this is very near the expected 2× speedup attainable when FP16 storage and arithmetic are enabled. With the P100 and V100 that support FP16 arithmetic, the added gain of FP16 arithmetic is between 5% and 15%. This highlights the fact that FD kernels are highly bandwidth limited, meaning that the time to read and write from/to global memory is much larger than the time required by arithmetic operations. FP16 computation is certainly useful; however, older models without this feature can greatly benefit from using half-precision for storage.

Solution accuracy

Using half-precision leads to a substantial gain in performance, but how accurate is the solution? To answer this question, we modeled seismic data for different benchmark models. As 2D and 3D wave propagation differs by how amplitude decreases with distance, we separated tests for the 2D and 3D implementations. In two dimensions, we tested benchmark models commonly used in exploration geophysics: the elastic Marmousi II model (Martin et al., 2006) and the 2004 BP velocity benchmark (Billette and Brandsberg-Dahl, 2005). The elastic Marmousi model allows us to study the accuracy of elastic effects in a model with moderately large velocity contrasts and the BP model is used to test the effect of very large velocity contrasts and very long offsets. In three dimensions, we present a comparison with the analytical solution and we test the 3D SEG/EAGE salt benchmark model (Aminzadeh et al., 1995).

2D propagation

The elastic Marmousi V_P model is shown in Figure 3a. The reader is referred to Martin et al. (2006) for V_S and ρ, which are structurally similar. Marmousi II is 17 km long and 3.5 km deep, allowing very long propagation distance to be studied. The model also contains quite large velocity variations — the maximum and minimum P-wave velocities are 4.7 and 1.03 km/s, respectively. The wide range of V_P/V_S ratios, between 1.58 and 5.47, allows us to test the accuracy of elastic effects. We simulated a high-frequency survey by using a 40 Hz Ricker wavelet as the source. To prevent dispersion with such a high-frequency content, the grid spacing was set at 1.25 m. To respect stability conditions, the time step was chosen as 0.137 ms. Note that we did not include a free surface and we put absorbing layers (Cerjan et al., 1985) on all sides of the model.

The BP model is shown in Figure 4a. As it is acoustic, we set μ to 0 GPa in our elastic code. The BP model contains many salt bodies exhibiting very large velocity contrasts. The minimum and maximum velocities are 1.429 and 4.790 km/s, respectively. Its large size (67.4 km long × 11.9 km thick) allows for the very long propagation distance. Here, we used a 10 Hz source, with a 6.25 m grid spacing and a 0.57 ms time step.

As a baseline, we use the FP32 solution computed with fourth-order Taylor FD coefficients. The seismograms are shown in Figures 3b (Marmousi II) and 4b (BP model). Traces were clipped at 0.5% of the maximum value of the gather to highlight faint reflections and refractions, which are of interest. The Marmousi shot gather exhibits numerous reflections between 0.8 and 3 s and clear refractions for distances over 12 km. The BP model shows numerous refractions and diffractions at short offsets and complex head wave behaviors at larger offsets.

Figures 3c–3g and 4c–4g show the error between the FP32 shot gather and different modeling precision. On all error figures, we used the same clipping as the FP32 gathers, multiplied by 10 (Marmousi II) and 50 (BP) to highlight the pattern of the numerical error.

The error obtained with the FP16 IO version is shown in Figures 3c and 4c for Marmousi and BP models, respectively. For both models, two different components can be observed: a background noncoherent noise with an amplitude that decreases in time and a coherent noise correlated with the main arrivals contained in the shot gathers. The coherent noise is concerning because it could sum positively during imaging and change the depth to reflectors or their amplitude. However, it has a very low amplitude, which represents only 0.49% (Marmousi II) and 0.035% (BP model) of the total energy contained in the shot gathers. Thus, its impact on the gradient or the migrated section should be minimal. The coherent noise of the FP16 COMP version is more energetic than for the FP16 IO version and is visible for most events found in the shot gathers (Figures 3d and 4d). Furthermore, the error is larger for longer propagation distances; that is, the error correlated with refraction events is more pronounced. The error remains very low, though, with 0.37% (Marmousi II) and 0.030% (BP model) of the total energy.

To put into perspective the noise produced by the FP16 kernels, we show in Figures 3e and 4e the difference between seismograms computed with two different sets of fourth-order FD coefficients: the baseline Taylor coefficients (9/8 and −1/24) and the Holberg coefficients with a maximum group velocity error of 0.1% (1.1382 and −0.046414). Computations are performed with FP32 in both cases. The coherent noise has a much stronger amplitude than either version of our FP16 kernels, but it remains still quite low for imaging purposes (note that the errors are magnified by 10× and 50× in Figures 3 and 4). The impact of using FP16 can thus be

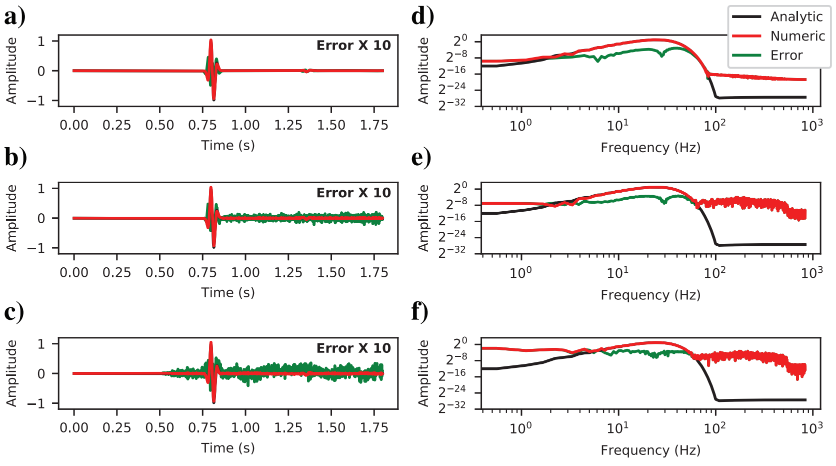

Figure 5. Comparison to the analytical solution for a 3D homogeneous elastic space. Results for (a) FP32, (b) for FP16 IO, and (c) for FP16 COMP are shown in red, the analytical solution is in black, and the error multiplied by 10 is in green. Panels (d-f) show the amplitude spectra of the numerical solution and its error for FP32, FP16 IO, and FP16 COMP, respectively.

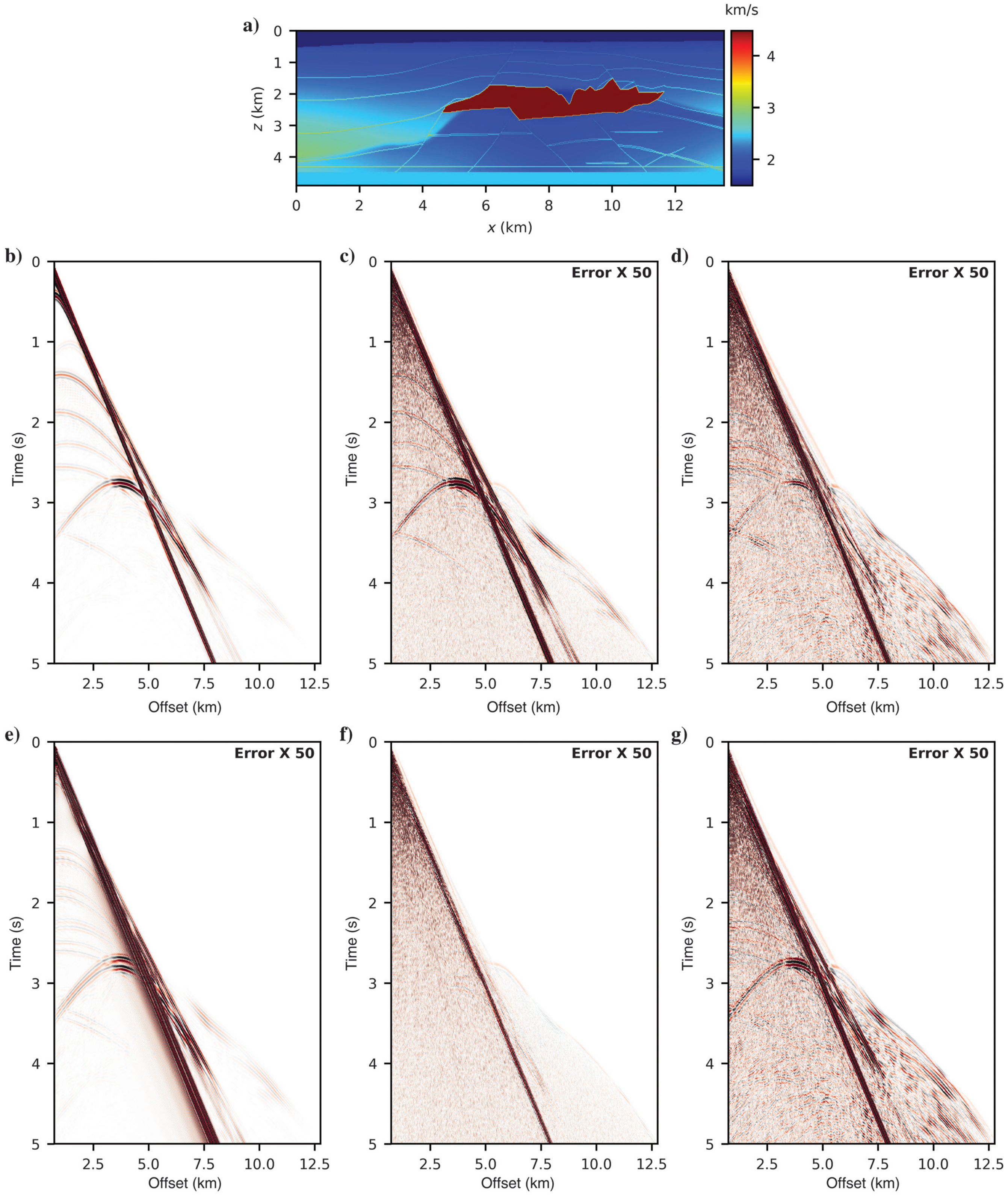

Figure 6. Comparison between modeling with FP16 and FP32 for the 3D SEG-EAGE salt benchmark model: (a) the V_P velocity, (b) the FP32 shot gather, (c) the error of FP16 IO, (d) of FP16 COMP, (e) the error between FP32 modeling with 12-point and 8-point Holberg coefficients, and the error between FP32 with models truncated to (f) FP16 and FP16 IO and (g) FP16 COMP.

considered quite small, given that both types of FD coefficients are deemed accurate enough for seismic imaging.

Finally, to better understand the source of the error, we computed the solution for a modified Marmousi model, obtained by truncating parameters a, b, and c to the nearest FP16 number. The error is shown in Figures 3f and 4f for FP16 IO and Figures 3g and 4g for FP16 COMP. In the case of FP16 IO, most of the coherent error is no longer visible, showing that it is controlled mostly by the rather small truncation error of the material parameters. Such truncation errors are, however, much smaller than the accuracy of the models used in imaging or inversion and should thus be of no practical consequence. In the case of FP16 COMP, the coherent energy is reduced, but still visible. Indeed, in this case, the dispersion relation will be affected by the reduced accuracy of the FD operator. For instance, the fourth-order Taylor coefficients are (1.125 and −0.041656) in FP16 and (1.125 and −0.041666668) in FP32. The small difference of the second coefficient will affect the numerical phase velocity, thus creating coherent errors between both solutions. Such errors, as shown earlier, remain rather small compared to using different FD coefficients.

3D propagation

The analytical solution for a point force in the z-direction for an elastic 3D homogeneous space is given by Pilant (2012). We computed the solution for $V_P = 3500$ m/s, $V_S = 2000$ m/s, and $\rho = 2000$ kg/m^3. The numerical solution was obtained with a 6 m grid spacing and a 0.6 ms time step, for a source with a peak frequency of 20 Hz. To be able to compute the solution for the largest offset possible with a 3D model fitting in the memory of a single GPU, we used for this test a 12-point stencil with Holberg coefficients for a maximum group velocity error of 0.1%. The grid size was $700 \times 700 \times 700$ with a 112 point thick absorbing boundary. The largest offset measured is thus 2558 m (or 40× the P-wave wavelength at 40 Hz). The number of grid points per wavelength is 14 at 40 Hz, well over the 2.91 value needed to keep the dispersion at less than 0.1%. The large absorbing boundary width and the large number of grid points per wavelength were used to obtain the most accurate numerical solution possible for this FD stencil.

Figure 5 shows the comparison of the analytical solution (the black lines), the numerical solution of three different versions of our code (the red lines), and the error (the green line). Note that the error has been magnified by 10 for the traces in the time domain (Figure 5a–5c). The traces shown are the particle velocity in the z-direction v_z, for an offset of 2558 m, on which the P-wave arrival is dominant.

Figure 5a shows the numerical results for the FP32 version of the code. The error represents 0.32% of the energy of the trace, which can be attributed to dispersion and the imperfect absorbing boundary. Such an error represents the degree of accuracy attainable with a standard FDTD code using a high-order stencil and a 32-bit floating-point representation. Figure 5d shows the amplitude spectrum of the same trace, the green line shows again the spectrum of the error. The amplitude of the error is smaller than the amplitude of the signal for frequencies between 2 and 50 Hz. Outside those frequencies, the solution loses its accuracy, mostly because of dispersion and of the signal truncation caused by the finite length of the signal.

Figure 5b and 5e shows the numerical results for the FP16 IO version of the code. The error contains a part that is correlated with the main arrival, which is similar to and of the same amplitude as the FP32 version. Hence, using FP16 only for storage does not seem to introduce further dispersion. The error contains an additional component compared to the FP32 solution: a random error in the time domain that is present after the first P-wave arrival. This error is

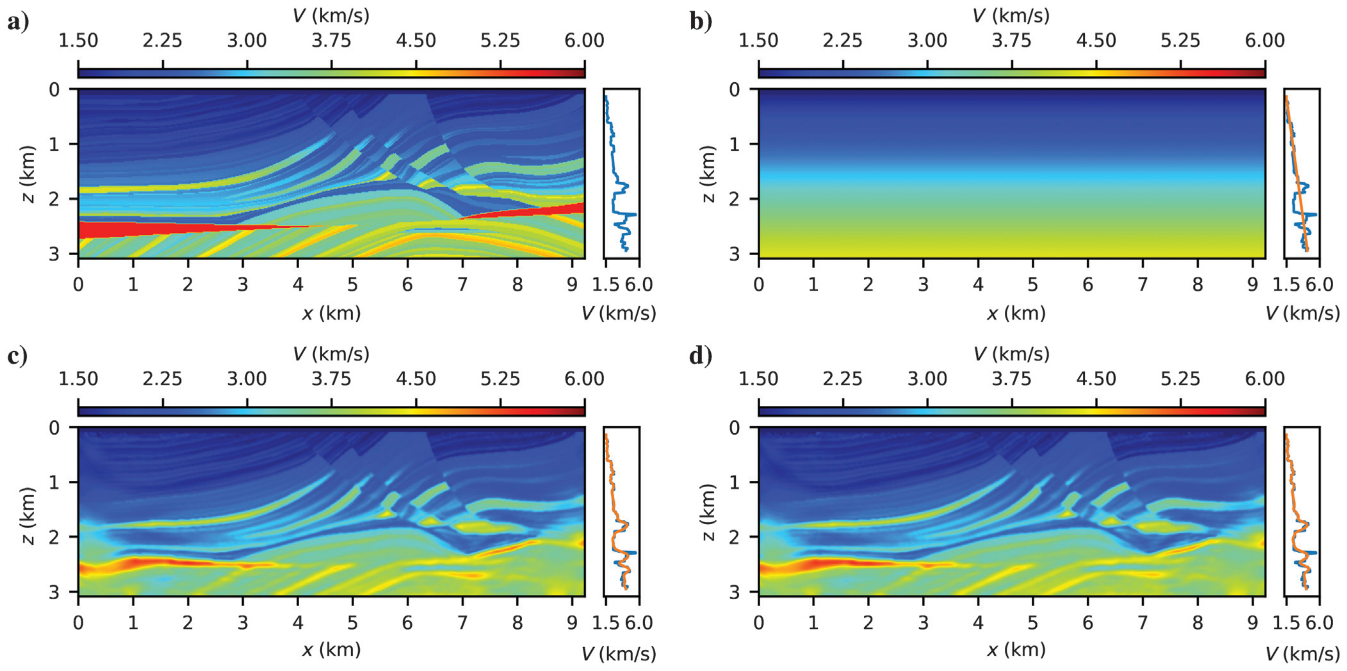

Figure 7. Inversion for the Marmousi model: (a) true V_P model, (b) initial model, (c) inverted model in FP32, and (d) inverted model with FP16 COMP. The insets are virtual boreholes located at a distance of 7 km, the blue lines represent the true model, and the orange lines represent the inverted models.

caused by the truncation error associated with the narrow dynamic range of half-precision, which propagates along the desired signal. The amplitude spectrum in Figure 5e shows that this random noise propagates at larger frequencies with an amplitude commensurate with the smallest FP16 normal number of 2^{-14}. Comparing Figure 5d and 5e, we see that the bandwidth where the solution is accurate is largely similar between FP16 IO and FP32, with the lowest accurate frequency slightly larger for FP16 IO at approximately 3 Hz instead of 2 Hz. The difference between the FP32 and FP16 IO solutions is 0.21% of the energy of the trace, below the error introduced by the FDTD stencil.

Figure 5c and 5f shows the numerical results for the FP16 COMP version of the code. As for the FP16 IO version, two components of the noise can be seen: one attributable to the accuracy of the FDTD stencil and the other attributable to the limited accuracy of half-precision. However, computations in half-precision introduced additional noises in the lower part of the spectrum. The lower limit of the usable bandwidth of the solution is increased to 6 Hz, but the higher limit remains at 50 Hz. The error between FP16 COMP and FP32 solution is now 1.03% of the total energy. The accuracy of the solution is reduced significantly by computing in FP16, but it may still be high enough for migration and inversion in most cases.

Finally, we computed a seismic shot for the SEG-EAGE salt model. The seismic gather is shown in Figure 6. We used the same 12-point stencil as for the analytic solution. Once again, we tried to reduce dispersion to a minimum, with a grid spacing of 20 m, a source peak frequency of 9 Hz, leading to four grid points per wavelength at 18 Hz for a minimum velocity of 1500 m/s. As shown in Figures 3 and 4, Figure 6a shows the shot gather obtained with the FP32 version, and Figure 6b and 6c shows the error between FP32 and FP16 IO and FP16 COMP, respectively. Note again that the errors have been magnified by 50. The same error patterns as in 2D propagation are observed: a coherent noise component following the main arrivals and a random noise component introduced by the limited dynamic range of FP16. As before, when the velocities are converted to the nearest representable FP16 value, the coherent error mostly disappears for FP16 IO, as can be seen in Figure 6f, confirming that storing the wavefields in FP16 does not introduce further numerical dispersion. Computing with FP16 does introduce some dispersion, as seen in Figure 6f. The energy of this error is even lower in three dimensions than in two dimensions, with 0.0018% and 0.0012% of the total energy for FP16 IO and FP16 COMP, respectively. Figure 6e compares those errors to the error between a 12-point and an 8-point stencil. Coherent noise is present in this gather and has a magnitude comparable to the FP16 versions of the code. This confirms that the FP16 solutions in three

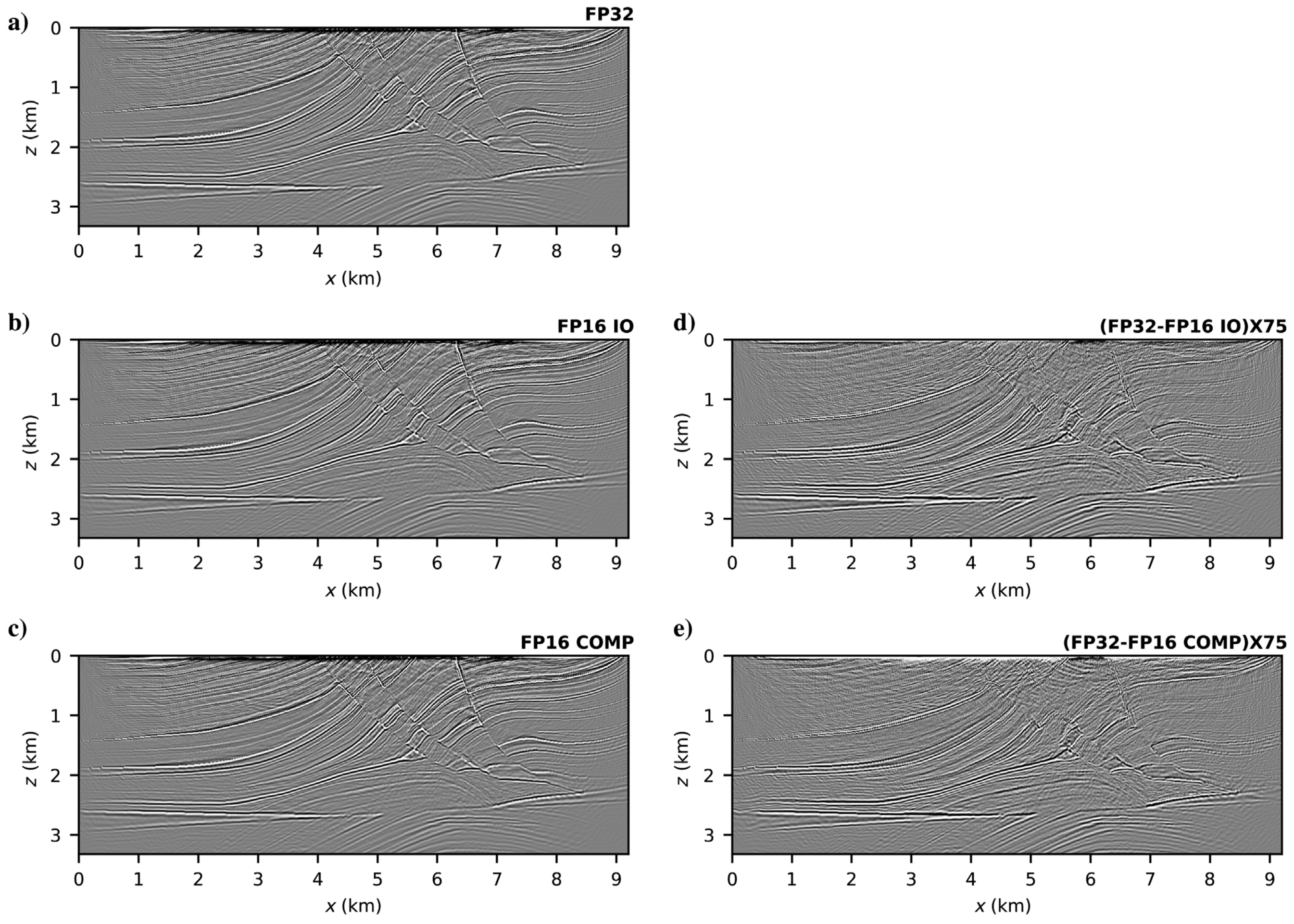

Figure 8. RTM migration for the Marmousi model with the different versions of the code: (a) FP32, (b) FP16 IO, and (c) FP16 COMP. The errors between FP32 and FP16 IO and FP16 COMP, multiplied by 75, are shown in (d and e), respectively.

dimensions have a level of accuracy that is similar to variations caused by using different stencils. Because the 12-point and 8-point stencils are deemed accurate enough for inversion and migration, the FP16 versions should thus be usable.

Impact on FWI and RTM

As a final test, we performed FWI and RTM on the original Marmousi benchmark model (Versteeg, 1994). The goal is to determine if the quality of the inverted model and image is adversely affected by noises introduced by FP16 modeling. To do so, a reference data set was computed with the FP32 code. FWI and RTM were carried out with the different versions of our code. Any changes in the inverted models should be caused by the reduced precision of FP16.

The data set for FWI contains 283 shots (one every two grid cells), each containing 283 receivers. The source is a Ricker wavelet with a central frequency of 7.5 Hz. All inversion tests were carried out with the exact same methodology. The starting model corresponds to the linear trend in depth of the true model (Figure 7a and 7b). We used stochastic limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) (Fabien-Ouellet et al., 2017a), in which each iteration uses a random subset of the sources to compute the gradient. To be able to compare different inversions results, the random subsets schedules are the same for each code version. In this experiment, six sources out of the 283 were used at each iteration. The inversion follows the frequency continuation strategy of Bunks et al. (1995): 14 frequency bands between 1.2 and 17 Hz were extracted via a Butterworth band-pass filter and inverted sequentially. We performed 40 iterations per frequency band, and we used the resulting model as a starting model for the next frequency band. Only V_P was inverted, and we kept the density and source signature fixed and known.

The results of the inversion are shown in Figure 7c and 7d for FP32 and FP16 COMP versions, respectively. The inversion has converged to a very accurate representation of the true model, for the full depth of the model. Inaccuracies on both sides of the model are caused by the acquisition footprint and are to be expected. As can be visually appraised, the FP32 and FP16 COMP inversion converged to very similar models. Some very minor differences exist: The root-mean-square error (rms error) between the true and the FP32 inverted models is 391 m/s, whereas the rms error between the FP32 and FP16 COMP inverted models is 61 m/s (43 m/s for FP16 IO). Thus, the error produced by FP16 usage is much smaller than the error caused by inversion, and, in this case, FWI can be performed either in FP32 or FP16.

RTM was also applied to the acoustic Marmousi model. In this case, the source wavelet was a 25 Hz Ricker wavelet. Using a smoothed version of the true model, 39 shots were migrated. We used a conventional RTM approach based on Laplace filters and a correlation imaging condition. Migrated sections obtained with the FP32, the FP16 IO, and FP16 COMP codes are shown in Figure 8a–8c, respectively. The three migrated sections are visually identical. Figure 8d and 8e shows the error between the FP32 and the FP16 versions, magnified by 75, to see the error pattern. The largest errors are found near the sources and receivers on the top of the model, where RTM results are inaccurate. The error is correlated with reflection events, although magnification is required to see it on the migrated sections. The deeper portion seems somewhat more affected, which is probably caused by the phase velocity mismatch produced by converting the velocity model to FP16. The error remains very small in both cases (0.05% of the energy for FP16 IO and 0.13% for FP16 COMP), indicating that FP16 is a viable option for migration.

CONCLUSION

We have shown that it is possible to use half-precision floating-point numbers to solve the isotropic elastic wave equation with the FDTD method. We have further shown that the solution remains accurate enough for FWI and RTM in realistic scenarios, in two dimensions and three dimensions. Note that FP16 usage can be switched on and off easily, in cases requiring higher accuracy. Advantages of using FP16 are (1) a reduction by a factor of two of memory usage and (2) an acceleration by a factor between 1.7 and two of the code execution on recent GPUs. The acceleration is mainly caused by the twofold increase of the number of cell elements per cycle that can be accessed in memory, our FD code being memory bound. This means a significant acceleration can be attained even with older GPU models that only support FP16 reading and writing from/to memory. Even CPU implementations should reap benefits from FP16 because they are too memory bound.

However, the use of FP16 with the FDTD method required extensive rewriting of our GPU kernel in CUDA, which is a minor inconvenience. Most importantly, the wave equation has to be rescaled so that operations involve terms respecting the reduced dynamic range of half-precision floating-point numbers. This scaling is performed once before actual computations happen and has a negligible impact on execution time. Still, great care must be taken to prevent any catastrophic overflow, which happens too easily when working in FP16. Even without overflow, FP16 precision may not be enough to compute an accurate solution of other PDEs, especially if the order of the accuracy of the numerical scheme is low. This is the case for common convolutional perfectly matched layer implementations or some FDTD solutions of the viscoelastic wave equation.

Finally, the possibility of usefully solving the wave equation in reduced precision opens the way to benefit from more sophisticated accelerators that are now being produced for FP16 computations, such as Nvidia's Tensorcores. As chip makers shift their priorities to follow the burgeoning machine learning industry, a major computing evolution is expected to take the form of FP16 accelerators. This work shows that the seismic community can benefit from such advancements, albeit with some effort on rethinking existing codes.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of the Nvidia Corporation for providing access to the different GPU models used in our numerical tests. We would also like to thank R. Clapp and B. Biondi from the Stanford Exploration Project for the valuable discussions on seismic modeling and inversion.

DATA AND MATERIALS AVAILABILITY

Data associated with this research are available and can be accessed via the following URL: https://github.com/GeoCode-polymtl/Seis_float16.git and 10.5281/zenodo.3551150.

REFERENCES

Abma, R., 2006, Numerical precision and triangle filter antialiasing in Kirchhoff migrations: The Leading Edge, **25**, 20–23, doi: 10.1190/1.2164748.

Aminzadeh, F., N. Burkhard, T. Kunz, L. Nicoletis, and F. Rocca, 1995, 3-D modeling project: 3rd report: The Leading Edge, **14**, 125–128, doi: 10.1190/1.1437102.

Bailey, D. H., 2005, High-precision floating-point arithmetic in scientific computation: Computing in Science & Engineering, **7**, 54–61, doi: 10.1109/MCSE.2005.52.

Baysal, E., D. D. Kosloff, and J. W. C. Sherwood, 1983, Reverse time migration: Geophysics, **48**, 1514–1524, doi: 10.1190/1.1441434.

Billette, F., and S. Brandsberg-Dahl, 2005, The 2004 BP velocity benchmark: 67th Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: https://doi.org/10.3997/2214-4609-pdb.1.B035.

Bunks, C., F. M. Saleck, S. Zaleski, and G. Chavent, 1995, Multiscale seismic waveform inversion: Geophysics, **60**, 1457–1473, doi: 10.1190/1.1443880.

Cerjan, C., D. Kosloff, R. Kosloff, and M. Reshef, 1985, A nonreflecting boundary condition for discrete acoustic and elastic wave equations: Geophysics, **50**, 705–708, doi: 10.1190/1.1441945.

Clapp, R. G., H. Fu, and O. J. T. l. E. Lindtjorn, 2010, Selecting the right hardware for reverse time migration: The Leading Edge, **29**, 48–58, doi: 10.1190/1.3284053.

Fabien-Ouellet, G., E. Gloaguen, and B. Giroux, 2017a, A stochastic L-BFGS approach for full-waveform inversion: 87th Annual International Meeting, SEG, Expanded Abstracts, 1622–1627, doi: 10.1190/segam2017-17783222.1.

Fabien-Ouellet, G., E. Gloaguen, and B. Giroux, 2017b, Time-domain seismic modeling in viscoelastic media for full waveform inversion on heterogeneous computing platforms with OpenCL: Computers & Geosciences, **100**, 142–155, doi: 10.1016/j.cageo.2016.12.004.

Fabien-Ouellet, G., E. Gloaguen, and B. Giroux, 2017c, Time domain viscoelastic full waveform inversion: Geophysical Journal International, **209**, 1718–1734, doi: 10.1093/gji/ggx110.

Fu, H., W. Osborne, R. G. Clapp, and O. Pell, 2008, Accelerating seismic computations on FPGAs — From the perspective of number representations: 70th Annual International Conference and Exhibition Incorporating SPE EUROPEC, EAGE, Extended Abstracts, doi: https://doi.org/10.3997/2214-4609.20147868.

Graves, R. W., 1996, Simulating seismic wave propagation in 3D elastic media using staggered-grid finite differences: Bulletin of the Seismological Society of America, **86**, 1091–1106.

Ho, N.-M., and W.-F. Wong, 2017, Exploiting half precision arithmetic in Nvidia GPUs: High Performance Extreme Computing Conference (HPEC), IEEE, 1–7.

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, doi: 10.1016/j.jcp.2010.06.024.

Levin, S. A., 2004, Numerical precision in 3D prestack Kirchhoff migration: 74th Annual International Meeting, SEG, Expanded Abstracts, 1119–1122, doi: 10.1190/1.1851078.

Martin, G. S., R. Wiley, and K. J. Marfurt, 2006, Marmousi2: An elastic upgrade for Marmousi: The Leading Edge, **25**, 156–166, doi: 10.1190/1.2172306.

Medeiros, V., A. Barros, A. Silva-Filho, and M. E. de Lima, 2013, High performance implementation of RTM seismic modeling on FPGAs: Architecture, arithmetic and power issues: Springer, 305–334.

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.

Pilant, L., 2012, Elastic waves in the earth: Elsevier.

Raknes, E. B., and B. Arntsen, 2015, A numerical study of 3D elastic time-lapse full-waveform inversion using multicomponent seismic data: Geophysics, **80**, no. 6, R303–R315, doi: 10.1190/geo2014-0472.1.

Versteeg, R., 1994, The Marmousi experience: Velocity model determination on a synthetic complex data set: The Leading Edge, **13**, 927–936, doi: 10.1190/1.1437051.

Virieux, J., 1986, P-wave propagation in heterogeneous media: Velocity-stress finite-difference method: Geophysics, **51**, 889–901, doi: 10.1190/1.1442147.

Virieux, J., and S. Operto, 2009, An overview of full-waveform inversion in exploration geophysics: Geophysics, **74**, no. 6, WCC1–WCC26, doi: 10.1190/1.3238367.

Warner, M., A. Ratcliffe, T. Nangoo, J. Morgan, A. Umpleby, N. Shah, V. Vinje, I. Štekl, L. Guasch, C. Win, G. Conroy, and A. Bertrand, 2013, Anisotropic 3D full-waveform inversion: Geophysics, **78**, no. 2, R59–R80, doi: 10.1190/geo2012-0338.1.

Yang, P., R. Brossier, L. Métivier, and J. Virieux, 2016, A review on the systematic formulation of 3-D multiparameter full waveform inversion in viscoelastic medium: Geophysical Journal International, **207**, 129–149, doi: 10.1093/gji/ggw262.

Zuras, D., M. Cowlishaw, A. Aiken, M. Applegate, D. Bailey, S. Bass, D. Bhandarkar, M. Bhat, D. Bindel, and S. Boldo, 2008, IEEE standard for floating-point arithmetic: IEEE Std 754-2008, 1–70.