GEOPHYSICS, VOL. 72, NO. 5 (SEPTEMBER-OCTOBER 2007); P. SM223–SM230, 10 FIGS.
10.1190/1.2753753

Computational methods for large-scale 3D acoustic finite-difference modeling: A tutorial

John T. Etgen[1] and Michael J. O'Brien[2]

ABSTRACT

We present a set of methods for modeling wavefields in three dimensions with the acoustic-wave equation. The primary applications of these modeling methods are the study of acquisition design, multiple suppression, and subsalt imaging for surface-streamer and ocean-bottom recording geometries. We show how to model the acoustic wave equation in three dimensions using limited computer memory, typically using a single workstation, leading to run times on the order of a few CPU hours to a CPU day. The structure of the out-of-core method presented is also used to improve the efficiency of in-core modeling, where memory-to-cache-to-memory data flow is essentially the same as the data flow for an out-of-core method. Starting from the elastic-wave equation, we develop a vector-acoustic algorithm capable of efficiently modeling multicomponent data in an acoustic medium. We show that data from this vector-acoustic algorithm can be used to test upgoing/downgoing separation of P-waves recorded by ocean-bottom seismic acquisition.

INTRODUCTION

Seismic modeling has long been used as a tool to help interpret recorded data and to assist in the design and testing of seismic-processing methods. A variety of different techniques for modeling wave propagation are available, but the finite-difference method is popular because it combines the ability to model wave propagation accurately in general media with reasonably straightforward computational algorithms. Although direct numerical solutions to the governing equations are often considered expensive, we prefer them because of the notion that the modeled data should have higher fidelity than the processing techniques we wish to test.

In addition to being useful for studying processing and interpretation issues, modeling can be used to study seismic-acquisition design. Although it is more common to use physical modeling to study acquisition design (Blacquière et al., 1999), computational modeling is also a proven approach (Etgen and Regone, 1998). The requirements for wavefield fidelity in modeling for acquisition design are stringent because we are often studying subtle or complex wave phenomena. Some of these phenomena we consider noise because of the difficulties they cause in later stages of processing or interpretation of seismic data using otherwise proven methods; but we must model them accurately to understand fully how they impact field data processing. Of course, the goal of being able to replicate all geometrical details of the acquisition geometry and all wave phenomena involved in modern seismic acquisition is still difficult to fully achieve, and we must make compromises.

The immediate application for the modeling techniques that we describe below is the design of surface-streamer and ocean-bottom acquisition geometries to improve seismic reflection images beneath salt (Regone, 2006). Other applications include the general improvement of wavefield separation and seismic imaging in any area of complex overburden. Critical to our primary task is the ability to model wave propagation in large 3D volumes of the subsurface using a frequency band similar to that observed in field data, incorporating the relevant physical phenomena. It would be ideal to carry out modeling computations using broadband, high-frequency waves in an anisotropic anelastic model, but the physical scale of the problems we wish to investigate makes it impractical or too expensive to do so routinely. Modeling must cost significantly less than field-acquisition trials. In addition, the difficulty in obtaining reasonable parameter estimates beyond P-wave velocity and density often leads to difficulties and delays in model construction.

As a compromise, we use the acoustic wave equation to study the design and processing of novel seismic-acquisition methods for subsalt imaging. Although there are obvious trade-offs in treating the earth as a fluid, this task is within the computational capacity available today on a cluster of workstations. We focus here on methods to make large-scale acoustic modeling of the exploration reflection seismology experiment practical, but many aspects of these techniques have broader application.

Manuscript received by the Editor December 5, 2006; revised manuscript received February 20, 2007; published online August 23, 2007.
[1]BP America, Houston, Texas. E-mail: john.etgen@bp.com.
[2]Allied Geophysics, Evergreen, Colorado. E-mail: mobrien@alliedgeo.com.

The computational techniques we use to optimize memory usage have existed for some time and have been applied to 3D elastic modeling (Yomogida and Etgen, 1993). They were introduced by Levin (1993) in the context of solving Laplace's equation and are described in detail for earthquake modeling by Graves (1996) and Moczo et al. (1999). For an excellent general reference on modern finite-difference methods applied to seismic-wave propagation, see Moczo et al. (2006). In addition to enabling efficient out-of-core computations, we use these techniques to optimize data flow from memory to cache, increasing the efficiency of modeling calculations on any computing platform.

To model P-waves recorded by a surface-streamer, seismic acquisition method in typical exploration settings, it is usually sufficient to model using the acoustic wave equation for variable-velocity, variable-density media. However, to model ocean-bottom recording with two (or more) components, it is necessary to generate and/or record vector motion in addition to pressure. We describe a method that is designed to propagate acoustic waves in a variable-velocity, variable-density medium and capable of generating and recording the directional components of particle motion such that multicomponent ocean-bottom recording can be modeled. In practice, we model ocean-bottom recording as reciprocal shots, where directional sources are placed at receiver locations and a pressure field is recorded at source locations.

EFFICIENT OUT-OF-CORE 3D FINITE-DIFFERENCE MODELING

One approach for modeling large-scale wave-propagation problems where the computational domain is too large to fit in the addressable memory of a single computer is to distribute the calculation across multiple individual computers (Villarreal and Scales, 1996). In this approach, the model is decomposed into subdomains and a joint computation is carried out by communicating subdomain boundary information between the separate processors at each time step. When high-speed communication is possible between the processors, this approach has merit. However, the implementation of these algorithms in available computer languages is often complicated. In addition, the use of many computers on a single problem leads to greater sensitivity to individual computer failures.

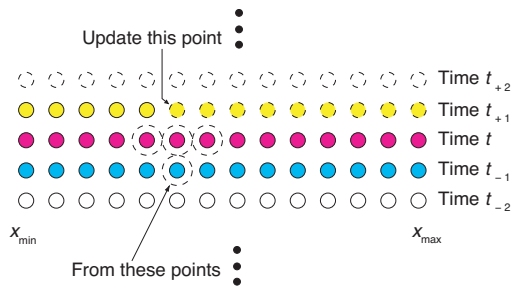

Figure 1. Order of calculations in a conventional time-marching, second-order-in-time, finite-difference scheme for the 1D acoustic wave equation. The magenta circles represent the wavefield values on the spatial grid at the current time step. The cyan circles represent the previous time level. The yellow circles represent the wavefield values at the new time level being computed.

We take another approach and compute individual experiments, typically a single shot, on a single computer using an out-of-core algorithm. The main drawback of an out-of-core algorithm is that it relies on high-speed input and output (I/O) to and from the disk to keep the entire computation from being I/O bound. To ameliorate the fact that on most modern computing systems I/O is far slower than memory access, we use a pipelined algorithm that computes multiple time-step updates per I/O pass over the computational volume. This strategy is very similar to the strategy used to get effective use of array processors in the days before vector computers were available; the latter was described by Stewart Levin many years ago (personal communication, 1987).

With this pipelined algorithm, the available physical memory stores only a subset of the computational volume at any instant in time. We make use of the fact that a time step for any given point in the computational volume only depends on the values of the wavefield at neighboring locations some finite distance away. The distance depends on the length of the numerical difference operators. A copy of the wavefield covering the entire computational domain resides on disk. Computations proceed by sweeping a multiple-time-step operator through the computational volume. This technique can be applied to any explicit time-marching method where the spatial derivative operators have finite length. Although this approach is not applicable to pseudospectral methods (Kosloff and Baysal, 1982; Reshef et al., 1988) because they need to access the entire computational domain along each spatial axis for every time step, it is applicable to high-order, finite-difference methods (Dablain, 1986; Holberg, 1987). In principle, the pseudospectral method could be used on space axes that are kept in memory and a finite-difference operator used only on the axis involved in the pipeline. Like the domain decomposition approach, this algorithm can become quite complicated when applied in conjunction with high-order spatial-difference operators. Indeed, it becomes even more complicated with the application of high-order, time-difference corrections (Dablain, 1986).

To illustrate, we start with the constant density 1D acoustic-wave equation,

$$\frac{\partial^2 u}{\partial t^2} = v^2(x)\frac{\partial^2 u}{\partial x^2}. \tag{1}$$

Figure 1 shows a schematic implementation of a finite-difference algorithm using a second-order-in-time, second-order-in-space, conventional, time-marching, finite-difference procedure:

$$\begin{aligned} u(x,t+\Delta t) &= v^2(x)\frac{\Delta t^2}{\Delta x^2}[u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)] \\ &\quad + 2u(x,t) - u(x,t-\Delta t). \end{aligned} \tag{2}$$

In the algorithm depicted in Figure 1, wavefield values from times t_{-1} and t are used to compute the entire spatial extent (x_{min} to x_{max}) of t_{+1}. In the figure, points that have been updated to time t_{+1} are shown with solid yellow circles, and points yet to be updated are shown with dashed yellow circles; the algorithm is updating points along the spatial axis in serial order. We can see from Figure 1 that once several of the leftmost values of time t_{+1} are available, all of the information needed to proceed to time t_{+2} is present for part of the spa-

tial domain. The conventional algorithm makes no use of that fact and brings the entire spatial extent of the wavefield to time t_{+1} before beginning to update any spatial locations to time t_{+2}.

Figure 2 shows a schematic of a pipelined algorithm for a second-order-in-time, fourth-order-in-space, 1D wave equation; now, wavefield values up to two gridpoints away are required to update any particular wavefield value. We rearrange the order of operations in the conventional finite-difference, time-marching method to begin the update of a wavefield value at a given x-coordinate as soon as all values needed from the prior time step are available. The filled black and green circles represent wavefield values present in memory for the two time levels, t_{-1} and t, currently stored on disk. Open black and green circles denote wavefield values for time levels t_{-1} and t residing on disk.

In the figure, a pipelined time step begins by reading from disk the wavefield values for the dashed black and green circles. Given those wavefield values, we can update the wavefield to time t_{+1} for the rightmost (dashed-circle) magenta gridpoint, two spatial increments to the left. Because the pipeline is moving left to right, the four magenta wavefield values to the right have already been updated to time t_{+1} and are available to create an update for the wavefield values denoted by the dashed yellow circle to time t_{+2}. This update allows the rightmost cyan wavefield value (dashed circle) to be updated to time t_{+3}. Finally, the leftmost cyan wavefield value and the leftmost yellow wavefield value are written to disk, and the entire pipeline moves one gridpoint to the right.

This simple schematic illustrates a three time-step update operator that would require 23 wavefield values to be stored in memory at any instant in the computation. In practice, there are opportunities for reducing the memory used by the pipeline substantially — for example, by not storing obsolete wavefield values that are no longer needed by the computations, such as the leftmost two black circles.

The extension of this method to 3D acoustic modeling is conceptually straightforward and illustrated in Figure 3. The computational pipeline of Figure 3 sweeps through constant y-planes of the computational space, performing four time steps per I/O pass through the wavefield volume. The number of time steps performed per pass is limited by the number of wavefield slices that fit in memory. In practice, we use most of the available memory to minimize I/O. By adapting the number of time steps computed in each pass over the model to the model's dimensions and available memory, the pipelined algorithm is capable of propagating wavefields in model volumes that are much larger than those we can fit in the memory of a single node. We generalize the pipelined algorithm outlined above to use high-order, variable-length differencing operators for the variable-velocity, variable-density wave equation in three dimensions, including the fourth-order, time-error correction method of Dablain (1986).

A problem analogous to the above out-of-core problem, where limited memory forces the use of a disk-to-memory-to-disk data flow, occurs with modern microprocessors where the amount of high-speed cache is limited. Normally, we think of access to main memory as being fast enough to allow full use of the CPU. In reality, access to main memory is often much slower than access to cache. The pipelined algorithm can increase computational performance when the modeling problem is also viewed as an out-of-cache problem, with data flowing from memory to cache and back to memory. Wavefield values in cache advance the time level of the calculation as far as possible before they are flushed or sent back to main memory.

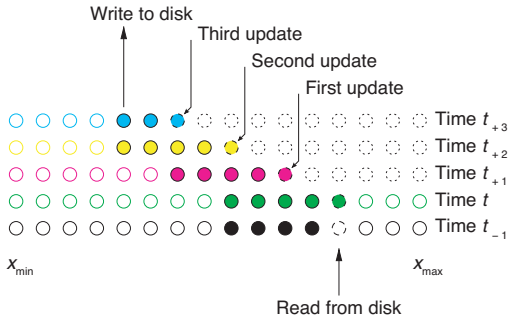

Figure 2. Order of calculations for a pipelined time-marching scheme for the 1D acoustic wave equation that uses five-point second spatial differences. The black and green circles represent the two most recent time levels, stored on disk. The magenta, yellow, and cyan circles represent the next three time levels that are computed in one pass over the computational volume.

APPLICATION OF HIGH-ORDER DIFFERENCING TO THE VARIABLE-VELOCITY, VARIABLE-DENSITY ACOUSTIC WAVE EQUATION

When modeling seismic reflection experiments, we often desire to separate the kinematics of wave propagation from the structural features that will generate reflections, so we need to allow both velocity and density to vary arbitrarily. Following Claerbout (1985), we write the acoustic wave equation in three dimensions as

$$\frac{\partial^2 u}{\partial t^2} = K\left[\frac{\partial}{\partial x}\left(\frac{1}{\rho}\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{1}{\rho}\frac{\partial u}{\partial y}\right) + \frac{\partial}{\partial z}\left(\frac{1}{\rho}\frac{\partial u}{\partial z}\right)\right], \quad (3)$$

where the bulk modulus is the product of the velocity squared times the density,

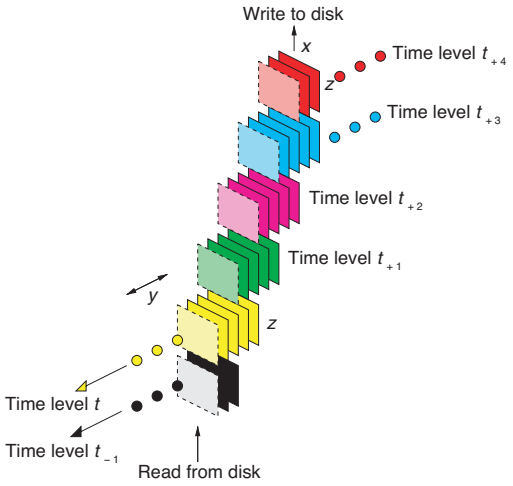

Figure 3. Depiction of a four-time-step computational pipeline for simple acoustic wave propagation in three dimensions. The computation holds the entire x-z extent of the model in memory for a subset of the y-axis. As in Figure 2, this multiple-time-step operator is passed over the entire spatial volume, advancing the state of the wavefield multiple (four) time levels.

$$K = \rho v^2. \tag{4}$$

Rather than derive our difference equations by applying the product rule for derivatives to equation 3, we write a cascaded first-order spatial difference expression to compute the second time difference of the wavefield:

$$\frac{\delta^2 u}{\delta t^2} = K\left[\frac{\delta_+}{\delta x}\left(\frac{1}{\rho}\frac{\delta_- u}{\delta x}\right) + \frac{\delta_+}{\delta y}\left(\frac{1}{\rho}\frac{\delta_- u}{\delta y}\right) + \frac{\delta_+}{\delta z}\left(\frac{1}{\rho}\frac{\delta_- u}{\delta z}\right)\right]. \tag{5}$$

The δ_+ and δ_- symbols denote spatial difference operators that are centered halfway between gridpoints, either forward or backward in the direction of the spatial derivative we approximate as denoted by their subscript. For example, suppressing the dependence of u on time and the other spatial coordinates, write the first derivative of u with respect to x evaluated halfway between gridpoints ix and $ix + 1$ as

$$\frac{\delta_+}{\delta x}u(ix + 1/2) \approx \sum_{j=0}^{n-1} a_j[u(ix + 1 + j) - u(ix - j)], \tag{6}$$

where a_j are the coefficients of the first difference operator of half-length n. These coefficients can be determined by Taylor expansion, as in Dablain (1986), or by an optimization approach, as in Holberg (1987).

In practice, we create difference coefficients for operators of a few different lengths — for example 2, 4, 8, or 12 points — and then choose an operator for each spatial axis based on the sample increment of that axis, desired accuracy, and run time. For example, in many reflection seismology problems, it is critical to achieve accurate, dispersion-free wave propagation in the vertical or near-vertical direction, although we might be willing to sacrifice some accuracy on waves that propagate large horizontal distances, such as refractions. In that case, we choose a longer and thus more accurate and more expensive operator for the z-axis than we do for the x- and y-axes.

VECTOR-ACOUSTIC MODELING

Conventional acoustic finite-difference modeling generates the wavefield from an omnidirectional pressure source. This computation is appropriate for modeling data generated by an air-gun source and recorded in the water column with hydrophones. When data are recorded on the ocean bottom, however, we almost always record at least two components: pressure and vertical particle velocity. This approach is used with field data to enable upgoing/downgoing wavefield separation, which is used for partial suppression of multiple reflections. Because an important contributing factor to the quality of images below salt is the degree to which the multiple reflections can be removed, no study of ocean-bottom recording would be complete without modeling both pressure and vertical particle velocity.

Propagating a vector wavefield in a liquid medium with the elastic-wave equation is an effective way to generate multicomponent data for acoustic problems. To improve calculation efficiency in this case where the shear modulus is zero, we can remove many terms from the calculation and derive a vector–acoustic wave equation rather than just using an elastic-finite-difference method with all shear parameters set to zero. We begin with the isotropic elastic wave equation expressed in particle displacements u_i, stresses S_{ij}, and accelerations $\partial^2 u_i/\partial t^2$:

$$\begin{aligned}
S_{xx} &= (\lambda + 2\mu)\frac{\partial}{\partial x}u_x + \lambda\frac{\partial}{\partial y}u_y + \lambda\frac{\partial}{\partial z}u_z,\\
S_{yy} &= (\lambda + 2\mu)\frac{\partial}{\partial y}u_y + \lambda\frac{\partial}{\partial x}u_x + \lambda\frac{\partial}{\partial z}u_z,\\
S_{zz} &= (\lambda + 2\mu)\frac{\partial}{\partial z}u_z + \lambda\frac{\partial}{\partial y}u_y + \lambda\frac{\partial}{\partial x}u_x,\\
S_{xy} &= \frac{\mu}{2}\left[\frac{\partial}{\partial x}u_y + \frac{\partial}{\partial y}u_x\right],\\
S_{xz} &= \frac{\mu}{2}\left[\frac{\partial}{\partial z}u_x + \frac{\partial}{\partial x}u_z\right],\\
S_{yz} &= \frac{\mu}{2}\left[\frac{\partial}{\partial z}u_y + \frac{\partial}{\partial y}u_z\right],\\
\rho\frac{\partial^2 u_x}{\partial t^2} &= \frac{\partial}{\partial x}S_{xx} + \frac{\partial}{\partial y}S_{xy} + \frac{\partial}{\partial z}S_{xz},\\
\rho\frac{\partial^2 u_y}{\partial t^2} &= \frac{\partial}{\partial y}S_{yy} + \frac{\partial}{\partial x}S_{xy} + \frac{\partial}{\partial z}S_{yz},\\
\rho\frac{\partial^2 u_z}{\partial t^2} &= \frac{\partial}{\partial z}S_{zz} + \frac{\partial}{\partial y}S_{yz} + \frac{\partial}{\partial x}S_{xz}.
\end{aligned} \tag{7}$$

To implement the acoustic approximation, we set the shear modulus μ to zero and remove all terms that have no remaining contribution. This leaves the coupled system of equations for vector motion in a fluid medium:

$$\begin{aligned}
\rho\frac{\partial^2 u_x}{\partial t^2} &= \frac{\partial}{\partial x}\lambda\left[\frac{\partial}{\partial z}u_z + \frac{\partial}{\partial x}u_x + \frac{\partial}{\partial y}u_y\right],\\
\rho\frac{\partial^2 u_y}{\partial t^2} &= \frac{\partial}{\partial y}\lambda\left[\frac{\partial}{\partial z}u_z + \frac{\partial}{\partial x}u_x + \frac{\partial}{\partial y}u_y\right],\\
\rho\frac{\partial^2 u_z}{\partial t^2} &= \frac{\partial}{\partial z}\lambda\left[\frac{\partial}{\partial z}u_z + \frac{\partial}{\partial x}u_x + \frac{\partial}{\partial y}u_y\right].
\end{aligned} \tag{8}$$

Note that the quantity in brackets is the same for all three equations.

This system of equations is essentially a reduced form of the elastic equations, so we use the staggered-grid technique of Virieux (1986). Rather than apply time differencing to both particle velocities and stresses, we use the modification proposed by Igel et al. (1995) to apply second-order time differencing to displacements because they appear directly in equation 8. This technique uses the first difference operators that are centered halfway between gridpoints described in equation 6 in the same space-differencing scheme used for the scalar acoustic equation. For isotropic acoustic models, the computational requirements to implement this system of vector equations are essentially the same as for the conventional scalar equations. In practice, propagating a single scalar wavefield is somewhat faster because there are fewer variables to move from disk to memory to CPU and back.

Our current application for vector-acoustic modeling is to generate synthetic multicomponent recording experiments that simulate seismic acquisition with pressure sources in the water column and pressure plus particle-velocity receivers on the ocean bottom. Because the typical seismic acquisition techniques that we wish to model call for many more pressure sources than multicomponent receiver stations, ocean-bottom recording is most efficiently modeled in reciprocal mode. We place directional and pressure sources at the relatively few seafloor locations that serve as receivers in field data, and pressure receivers are placed on a fine grid at the surface where, in the field, we would place pressure sources. Using a vector-acoustic propagator is also an efficient way to compute anisotropic wavefields when shear data are not an integral part of the study.

EXAMPLES

Scalar wave propagation

Figure 4 shows velocity and density in a model typical of acquisition design studies for subsalt imaging in the Gulf of Mexico. The model is approximately 1600 km^2 in size, realistic for this type of study. We computed shots at approximately 8000 surface locations to produce a data set that can be decimated into several different acquisition geometries. The data sets from those acquisition geometries are processed using prestack depth migration and then compared to determine which recording geometry and/or processing strategy produces the best image.

Figure 5 displays the raw output from the modeling code, an areal shot record sampled at 30 m in both x- and y-axes for the entire $16\times16\times10$-km submodel. The shot location for Figure 5 is over an uncomplicated part of the model, giving a relatively simple set of events in the shot record. Figure 6 displays the areal shot record data for a shot taken over the salt body in the model. In Figure 6, the events are much more complicated than in Figure 5because of distortions in the wavefield caused by salt and the presence of complex multiple reflections. The computational model for this example has $640\times640\times400$ gridpoints at 25-m spacing in all directions (x,y,z), which means that a single time level of the entire wavefield requires approximately 164 Mwords of memory. Using a simple implementation of an acoustic wave equation that requires five 3D arrays (velocity, density, and the wavefield for three time levels), the domain for this computational model would barely fit in a machine with 4 Gbytes of memory. The vector-acoustic method, which requires 14 3D arrays (velocity, density, three wavefield components at three time levels each, and three intermediate stress variables), requires almost three times as much memory and would require a single machine with greater than 10 Gbytes of addressable memory. In contrast, a pipelined algorithm that computes 20 time steps per I/O pass through the model requires approximately 1.4 Gbytes of memory to compute the data in Figures 5 and 6.

In practice, we tend not to fill memory completely because most operating systems can use memory to improve the performance of the I/O through the use of buffer cache, thus improving overall performance. This also partially mitigates the fact that we made no attempt to use asynchronous I/O in the implementation of the pipelined algorithm. Figures 5 and 6 were computed in 6600 time steps with an increment of 0.0018 s using eight-point spatial difference operators on the z-axis and four-point spatial difference operators on the x- and y-axes for estimating the second-order time derivatives of the wavefields and two-point spatial difference operators for the fourth-order time-difference-error correction step.

For this example, 76 floating-point operations are needed per gridpoint per time step. The scalar acoustic code requires between 64 and 154 floating-point operations per gridpoint per time step, depending on the accuracy level chosen by the user. This may seem high, but we prefer to use accurate, high-order differences to produce the highest-fidelity results we can afford. Total run time for a single shot in this example is almost exactly one CPU day on the workstations we use, and about 80% of that time is CPU time. Therefore, a cluster of 800 workstations can compute all the required shots in about 10 days.

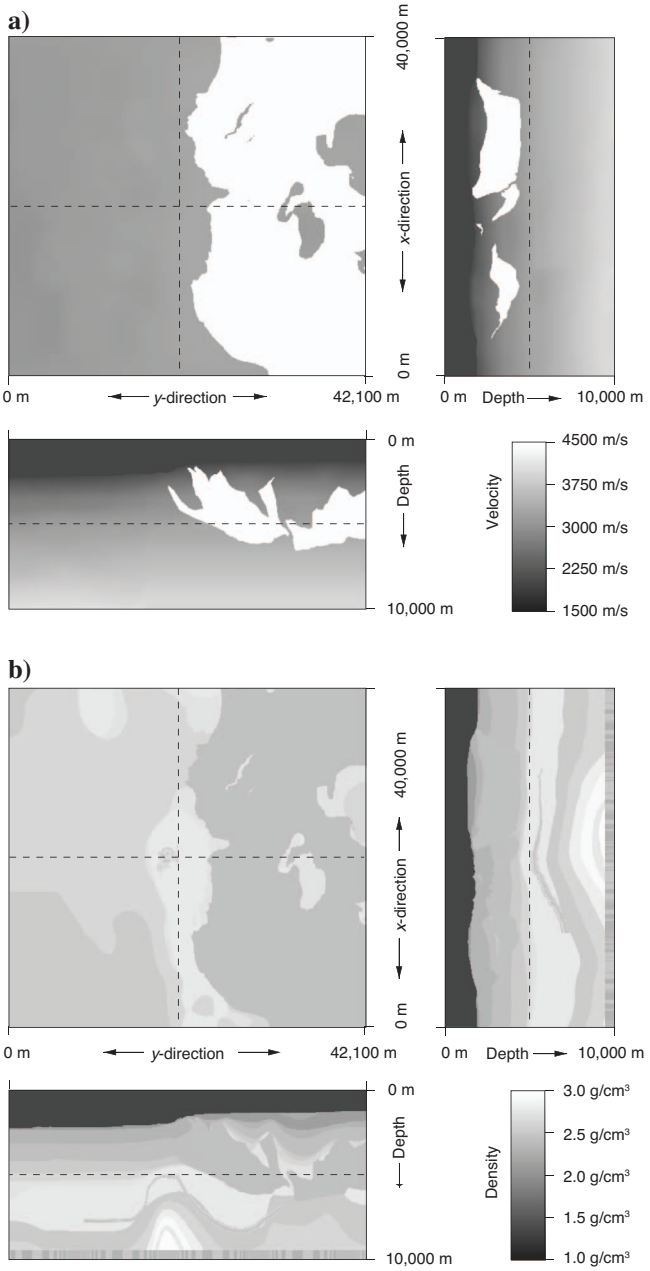

Figure 4. (a) Velocity and (b) density model in a 3D finite-difference modeling study of wide-azimuth acquisition design.

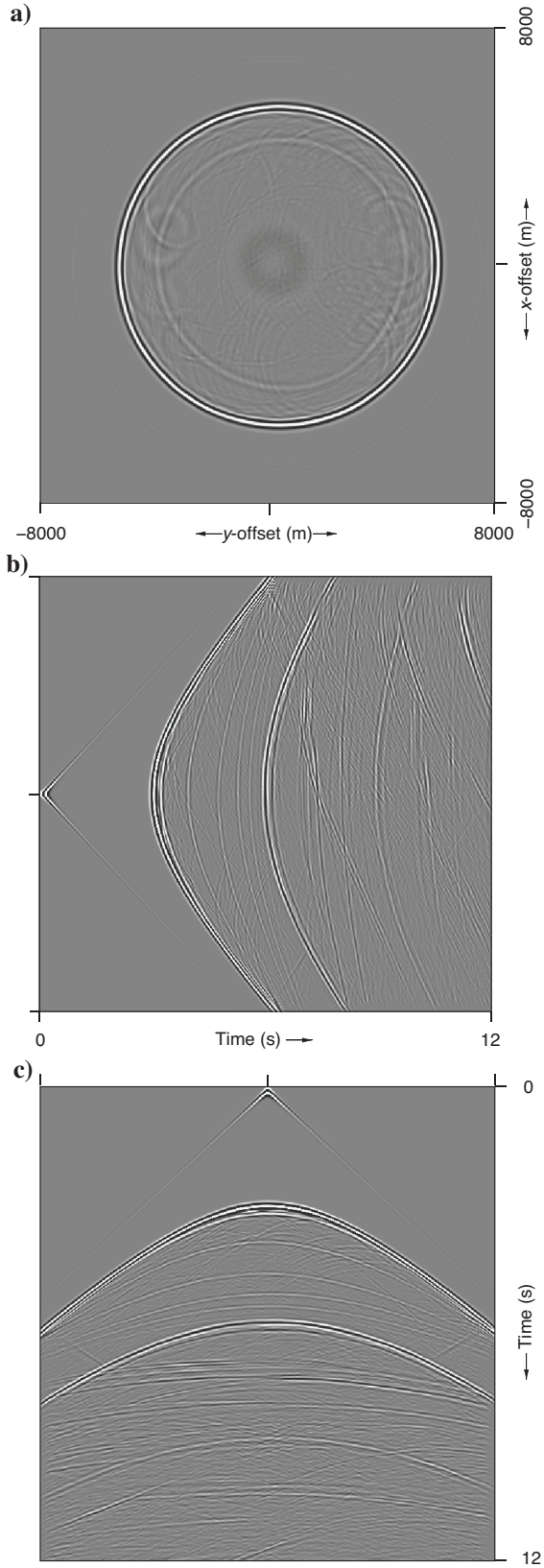

Figure 5. Areal shot record data output from 3D finite-difference modeling for a shot in the Figure 4 model. (a) Time slice; (b) time section through the source location in the *x*-direction; (c) time section through the source location in the *y*-direction.

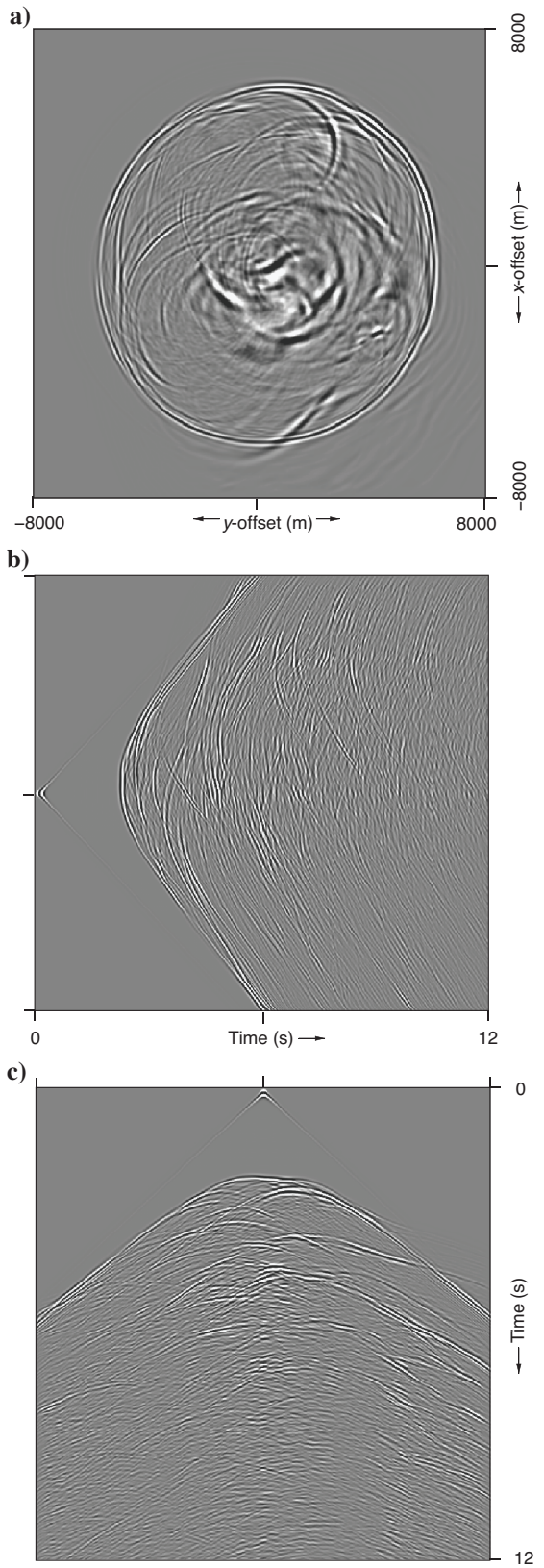

Figure 6. Areal shot record data output from 3D finite-difference modeling for a shot in the Figure 4 model. (a) Time slice; (b) time section through the source location in the *x*-direction; (c) time section through the source location in the *y*-direction.

Vector-acoustic wave propagation

A simple model (Figure 7) is used to demonstrate the capabilities of the vector-acoustic equation to model an ocean-bottom experiment. The source is set on the seafloor at a depth of 750 m, and the receivers are placed just below the surface of the model at a depth of 20 m. Two shots are computed: one using a pressure source and the other using a vertical displacement source. Snapshots of the wavefield from both shots are placed side by side in Figure 8. Because we use an equation that propagates displacements, the vertical component wavefield shown in Figure 8 does not directly give the data for upgoing/downgoing wavefield separation by PZ summation. We differentiate the recorded wavefield with respect to time to create vertical particle velocity data. Nevertheless, comparing the displacement field and pressure field snapshots illustrates that the two wavefields contain free-surface reflections with opposite polarity. This difference in polarity for events reflected downward from the free surface is the phenomenon that allows summation to separate upgoing and downgoing fields.

Time records for the two shots are shown in Figure 9, where the vertical displacement data have been converted to vertical velocity through differentiation with respect to time. We treat these records as common receiver records where surface sources are recorded into ocean-bottom receivers. From that perspective, and counting events from top to bottom, the first, third, and sixth events to arrive at the seafloor are propagating downward and contain energy that PZ summation should remove. The second and fourth events are from waves propagating upward and contain energy that should be enhanced. The fifth event is an internal multiple with contributions from both upgoing and downgoing waves. The right panel of Figure 7 is a schematic of raypaths for these six events.

Figure 10 shows the results of rudimentary wavefield separation performed by adding these two sections together. In Figure 10, the direct arrival is attenuated on the near-offsets where waves are propagating nearly vertically and the direction of particle motion is primarily vertical. At farther offsets where particle velocities have a large horizontal component, the attenuation is less effective. At later times (events three and six), attenuation is more successful at farther offsets because the energy from these compound multiples is traveling mostly downward. The simple summation performed to create Figure 10 demonstrates that vector-acoustic modeling does generate data useful for upgoing/downgoing separation tests. In practice, we use a more sophisticated wavelet matching technique to improve the quality of the up/down separation analogous to the techniques used on field data.

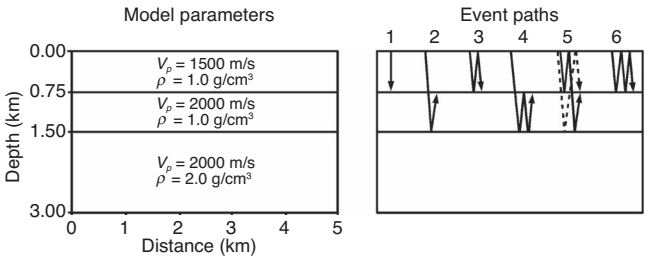

Figure 7. Medium parameters of the diagnostic model are shown on the left. The source for a reciprocal shot is placed on the seafloor at z = 0.75 km. Receivers are at the surface. The reciprocal paths (from receiver back to source) of the first six events are shown on the right.

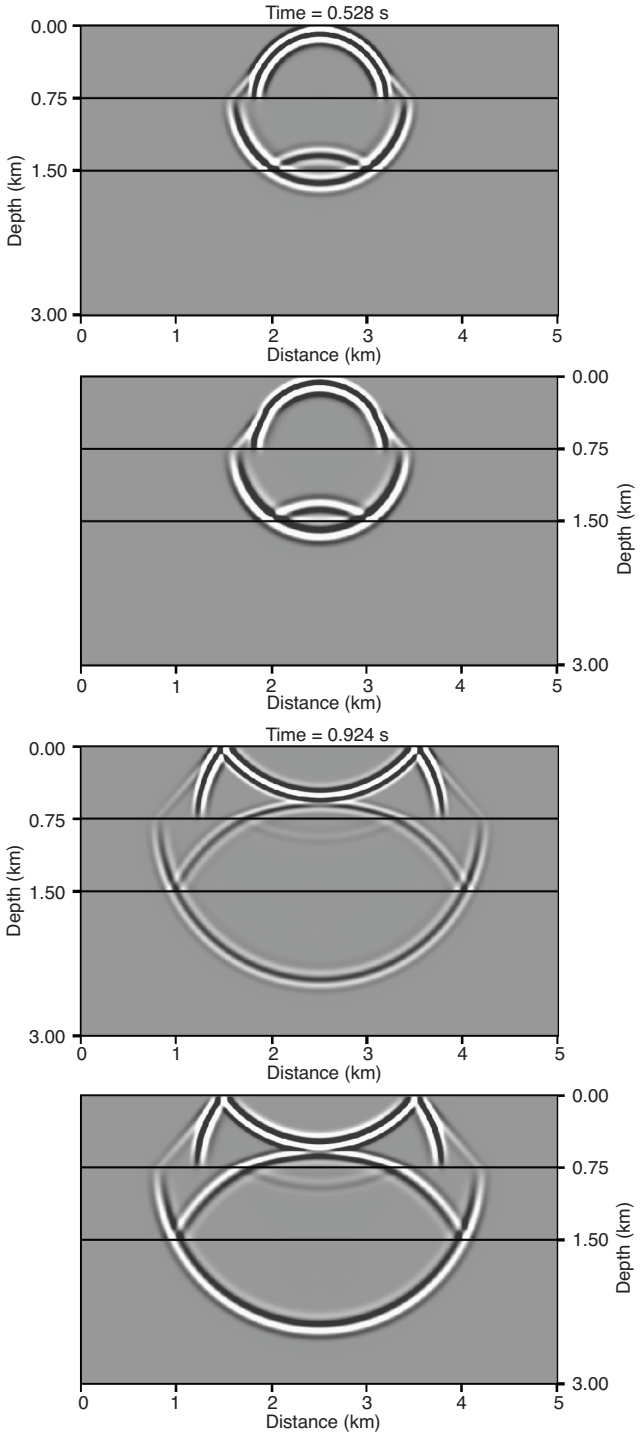

Figure 8. Snapshots of the wavefield from a pressure source (left) and from a vertical displacement source (right). The polarity of the waves reflected upward from the boundary at 1.5 km is approximately the same for the two types of wavefields. These add constructively upon PZ summation. The polarity of waves reflected downward from the free surface have approximately opposite polarity for the two types of wavefields, resulting in partial cancellation upon PZ summation. Note the small discrepancies in the phase of these wavefields because we are plotting pressure and displacement here, rather than pressure and vertical velocity. Recorded vertical displacement data are differentiated with respect to time to produce vertical particle velocity traces prior to actual PZ summation.

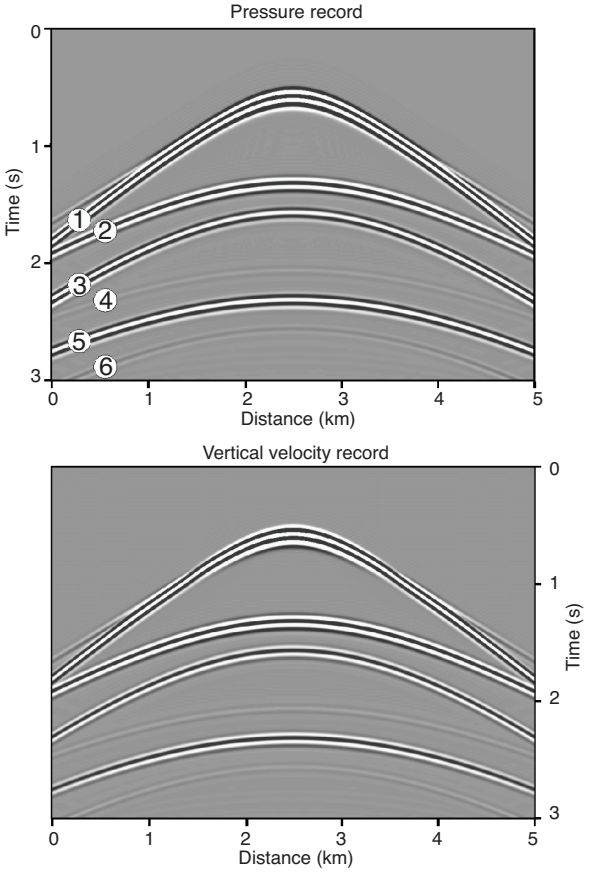

Figure 9. Ocean-bottom common-receiver gathers (computed by invoking reciprocity) for a pressure receiver and a vertical velocity receiver. The first, third, and sixth events contain purely downgoing energy and have opposite polarity on the two receiver components. The second and fourth events contain purely upgoing energy and have the same polarity. Event numbers annotated on the pressure record correspond with raypaths shown in Figure 7.

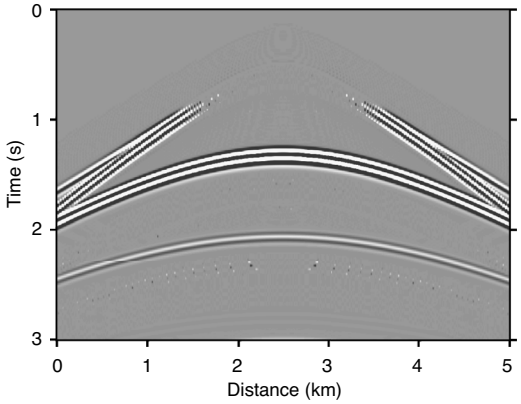

Figure 10. Rudimentary PZ summation of the receiver gathers in Figure 9 results in attenuation of the downward-traveling waves reflected from the free surface. The far offsets of the first event contain a large component of horizontal particle velocity, preventing complete attenuation. The second and fourth events are enhanced relative to Figure 9.

CONCLUSIONS

The pipelined algorithm described here is general in the sense that it can improve the speed of and/or the size of wave-equation modeling computations. We show that it can be adapted to high-order space- and time-differencing operators and discuss using the same general pipeline method to enhance performance of in-core calculations. In addition, we develop a vector-acoustic system of equations equivalent to the elastic-wave equation in a liquid medium and show that z-component data generated by this method can be combined with pressure data to separate upward- and downward-traveling waves.

In our experience, the run time of the out-of-core version of our code is about 1.4 times that of an in-core version when the problem will fit in memory. While it may be possible to get better speed from a domain-decomposed modeling code executing on a cluster of machines, the inherent robustness of running single shots on individual machines may compensate for any potential loss in efficiency caused by having to do I/O during the calculations.

ACKNOWLEDGMENTS

We thank BP for permission to publish this work and the BP High-Performance Computing team in Houston for their support.

REFERENCES

Blacquière, G., A. Volker, and L. Ongkiehong, 1999, 3-D physical modeling for acquisition geometry studies: 69th Annual International Meeting, SEG, Expanded Abstracts, 665–668.

Claerbout, J. F., 1985, Imaging the earth's interior: Blackwell Scientific Publications, Inc.

Dablain, M. A., 1986, The application of high-order differencing to the scalar wave equation: Geophysics, **51**, 54–66.

Etgen, J., and C. Regone, 1998, Strike shooting, dip shooting, widepatch shooting — Does prestack depth migration care? A model study: 68th Annual International Meeting, SEG, Expanded Abstracts, 66–69.

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.

Holberg, O., 1987, Computational aspects of the choice of operator and sampling interval for numerical differentiation in large-scale simulation of wave phenomena: Geophysical Prospecting, **35**, 629–655.

Igel, H., P. Mora, and B. Riollet, 1995, Anistotropic wave propagation through finite-difference grids: Geophysics, **60**, 1203–1216.

Kosloff, D. D., and E. Baysal, 1982, Forward modeling by a Fourier method: Geophysics, **47**, 1402–1412.

Levin, S. A., 1993, High performance computing: Scientific Computing and Computational Mathematics (SCCM) 240 Lecture Notes, Stanford University.

Moczo, P., M. Lucká, J. Kristek, and M. Kristeková, 1999, 3D displacement finite differences and a combined memory optimization: Bulletin of the Seismological Society of America, **89**, 69–79.

Moczo, P., J. O. A. Robertsson, and L. Eisner, 2006, The finite-difference time-domain method for modeling of seismic wave propagation, *in* R.-S. Wu and V. Maupin, eds., Advances in wave propagation in heterogeneous earth: Advances in Geophysics 48, Elsevier-Pergamon Press.

Regone, C. J., 2006, A modeling approach to wide-azimuth design for subsalt imaging: The Leading Edge, **25**, 1467–1475.

Reshef, M., D. Kosloff, M. Edwards, and C. Hsiung, 1988, Three-dimensional acoustic modeling by the Fourier method: Geophysics, **53**, 1175–1183.

Villarreal, A., and J. A. Scales, 1996, 3D finite difference modeling via domain decomposition: 66th Annual International Meeting, SEG, Expanded Abstracts, 1231–1234.

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

Yomogida, K., and J. T. Etgen, 1993, 3D wave propagation in the Los Angeles basin for the Whittier-Narrows earthquake: Bulletin of the Seismological Society of America, **83**, 1325–1344.