Parallel 3D prestack depth migration using recursive Kirchhoff extrapolation

Hugh D. Geiger, Gary F. Margrave, Darren S. Foltinek[*], and J. Marc Langlois[**]

ABSTRACT

We have designed and implemented a parallel 3D prestack depth migration algorithm based on recursive Kirchhoff extrapolators. Recursive Kirchhoff wavefield extrapolation in the frequency-space domain allows us to use the Weyl formulation, which should give better estimates of the phase than either the GPSPI (generalized phase shift plus interpolation) or NSPS (non-stationary phase shift) formulations, and hence reconstruct the extrapolated wavefield with greater accuracy. The algorithm has been coded for single- and multiple-node processing in both MATLAB and C. Simple synthetics tests have greatly accelerated the testing and debugging phase of the software development. This project lays the foundation for the POTSI research-friendly framework for parallel seismic processing and imaging, which will allow us to efficiently investigate a variety of compute-intensive problems in seismic processing and imaging. Key features of the research framework are: full integration with existing software including the CREWES MATLAB seismic toolbox, Seismic Unix, USP, and ProMAX; concurrent design and coding in MATLAB and C/FORTRAN; and semi-automated archiving. Research plans for the upcoming year include depth migration of selected 2D and 3D synthetic datasets (e.g. SEG/EAGE Salt and SIGSBEE dataset), followed by a shift in focus to depth imaging of 3D land seismic datasets.

INTRODUCTION

It has long been recognized that seismic data processing is amenable to parallel computation. Many processing algorithms are ‘embarrassingly parallel’ in that they can take advantage of independent data structures, such as traces and records in the acquired data coordinates or frequency in transformed data coordinates. Thus, large data volumes can be split into a number of sub-volumes for concurrent computation. 3D recursive or ‘wave-equation’ prestack depth migration algorithms are particularly compute-intensive and are thus natural candidates for parallelization. In this paper, we discuss initial results of our efforts to design parallel 3D prestack depth migration algorithm based on the 3D recursive Kirchhoff extrapolators introduced by Margrave and Daley (2001).

Up until recently, the high cost of parallel computational facilities has limited their use in university research to high-level problems. Access to parallel computation facilities, such as the MACI cluster at the University of Calgary, is typically limited and often available only on an assessed-needs basis. In addition, parallel implementations require advanced skills in parallel programming and seismic data management. Researchers weaned on MATLAB need to hone their skills in languages such as C or FORTRAN and develop a good understanding of MPI (the Message Passing Interface). Implementation often requires a top-down redesign of algorithm architecture, extensive debugging and

[*] Front Range Publishing; www.frontrange.ca
[**] Marc Langlois Consulting; marc.langlois@shaw.ca

testing, reworking to achieve desired levels of optimization, and communication with parallel-unfriendly processing software such as SU (Seismic Unix) or ProMAX. The result is that the use of parallel computation is pushed much later in the research cycle, or doesn't happen at all.

One of the primary goals of the POTSI project (Pseudodifferential Operator Theory in Seismic Imaging) is to develop a research-friendly framework that encourages the use of parallel computation facilities. It is now possible to purchase a modest PC/Linux cluster for less than the cost of a research salary. CREWES, as an in kind sponsor of POTSI, plans to acquire a cluster consisting of ~20 processors within the next few months. Although a dedicated parallel cluster will remove some of the barriers to access mentioned above, it does not address the most important challenge – how to design a research framework for parallel computation to ensure that the cluster becomes an efficient research tool.

Three essential components of the research framework were identified at the outset: first, that it will build upon the CREWES MATLAB seismic toolbox, which has proven to be both a powerful tool for high-level research and easy to modify for distributed parallel computation; second, that the basic parallel architecture be designed, developed, and coded concurrently in MATLAB and C/FORTRAN, as this will allow efficient testing and debugging; and third, that the legacy of reusable code and associated documentation be archived as automatically as possible, so that incoming students and researchers do not have to continually 'reinvent the wheel.'

A well-designed research framework that encourages efficient use of a parallel computational environment will allow us to tackle a host of problems in the larger context of 3D depth imaging, including:

- cost-optimized acoustic and elastic wavefield extrapolation,
- anisotropic true-amplitude imaging and inversion (Ferguson and Margrave 2002),
- recursive Kirchhoff extrapolation (Margrave and Daley, 2001; Margrave and Geiger, 2002),
- recursive GPSPI (generalized phase shift plus interpolation) and NSPS (nonstationary phase shift) extrapolation (Margrave & Ferguson, 1998 and 1999; Mi and Margrave, 2001),
- Gabor extrapolation (Grossman et al. 2002a and 2002b),
- Recursive wavefield extrapolation using the most effective algorithm for each step,
- least-squares imaging (Kuehl and Sacchi, 2001. Sacchi is a POTSI participant at University of Alberta),
- imaging velocity analysis,
- Gabor deconvolution (Margrave and Lamoureux, 2001; Margrave et al., 2002),
- high-resolution imaging,

- irregular/undersampled land and OBS geometries (Margrave and Geiger, 2002),
- land and OBS acquisition design,
- true depth processing (imaging = deconvolution + migration),
- rough topography (Margrave and Yao, 2000; Margrave and Geiger, 2002),
- multiple suppression,
- imaging through complex near-surface velocity structures (e.g. permafrost), and
- P-S imaging and P-P and P-S joint inversion (Larsen et al, 1998; Larsen, 1999; Margrave et al, 2001).

For our initial research project, we chose to implement parallel 3D prestack depth migration using the recursive Kirchhoff extrapolators introduced by Margrave and Daley (2001). The migration code is being developed in both MATLAB and C with careful attention to modular design. This will allow us to use the parallel framework and libraries to quickly implement new algorithms; for example, the GPSPI and NSPS extrapolators of Margrave and Ferguson (1998, 1999), Mi and Margrave (2001), and Ferguson and Margrave (2002).

In this paper, we provide a brief review of the theory of recursive Kirchhoff extrapolators, describe the basic structure of the parallel algorithm, discuss some preliminary results from simple synthetic seismic tests of the MATLAB and C-code, and finally, present our plans for future research.

THEORY REVIEW: RECURSIVE KIRCHHOFF EXTRAPOLATORS

We consider wavefield extrapolation with the 3D Kirchhoff implementation of the GPSPI extrapolator (Margrave and Daley, 2001). If $\psi(x,y,z=0,\omega)$ is an upgoing wavefield in the space-frequency (x,y,ω) domain at depth level $z=0$, then its backward-extrapolated value at depth z is estimated as

$$\psi(x,y,z,\omega)=\frac{-i\omega}{2\pi}\int_{-\infty}^{\infty}\psi(\hat{x},\hat{y},z=0,\omega)\frac{\cos\theta}{v\tilde{r}}e^{i\omega\tilde{r}/v}\left(1+\frac{iv}{\omega\tilde{r}}\right)d\hat{x}d\hat{y}, \qquad (1)$$

where the integration is over the input lateral coordinates $\hat{x}$ and $\hat{y}$, v is the laterally-variable velocity, x and y are the output lateral coordinates, z is positive downward, $\tilde{r}$ is the 3D radius vector given by

$$\tilde{r}=\sqrt{(x-\hat{x})^2+(y-\hat{y})^2+z^2}, \qquad (2)$$

and $\cos\theta = z/r = \partial r/\partial z$ is the cosine of the *scattering angle.*

Equation (1) consists of two terms, one that goes as r^{-1} and is called the *far field term* and another that goes as r^{-2} and is called the *near field term*. The phase shift $e^{i\omega\tilde{r}/v}$ is precisely that required to give the time-advanced output as a sum along a constant-velocity diffraction hyperboloid[*]. Thus, the equation is a spatial convolution of the wavefield with an operator that achieves geometric scaling and summation along a diffraction hyperboloid. Accordingly, equation (1) is interpretable as a Kirchhoff-style wavefield extrapolation operation in the space-frequency domain.

Equation (1) can be expressed in a simplified form as

$$\psi(x,y,z,\omega)=\int_{-\infty}^{\infty}\psi(\hat{x},\hat{y},z=0,\omega)W_{3D}(x-\hat{x},y-\hat{y},z,v,\omega)d\hat{x}d\hat{y}\,, \tag{3}$$

where

$$\overleftarrow{W}_{3D}(x-\hat{x},y-\hat{y},z,v,\omega)=-\frac{i\omega}{2\pi}\left[\frac{|z|}{v\tilde{r}^2}e^{i\omega\tilde{r}/v}\left(1+\frac{iv}{\omega\tilde{r}}\right)\right] \tag{4}$$

is the backward 3D space-frequency wavefield extrapolation operator or, alternatively,

$$\vec{W}_{3D}(x-\hat{x},y-\hat{y},z,v,\omega)=\frac{i\omega}{2\pi}\left[\frac{|z|}{v\tilde{r}^2}e^{-i\omega\tilde{r}/v}\left(1-\frac{iv}{\omega\tilde{r}}\right)\right] \tag{5}$$

is the forward 3D space-frequency wavefield extrapolation operator. Equation (3) shows that W_{3D} is applied to the wavefield by a 2D convolution over the lateral spatial coordinates.

The expressions for W_{3D} (equations 4 and 5) were developed from a constant velocity phase shift extrapolator (see Margrave and Daley, 2001) but are easily modified to give phase shift operators that approximately accommodate lateral velocity variations. The backward extrapolator for 3D GPSPI is

$$\psi_{GPSPI}(x,y,z,\omega)=\frac{-i\omega}{2\pi v(x,y)}\int_{-\infty}^{\infty}\psi(\hat{x},\hat{y},z=0,\omega)\frac{z}{\tilde{r}^2}e^{i\omega\tilde{r}/v(x,y)}\left(1+\frac{iv(x,y)}{\omega\tilde{r}}\right)d\hat{x}d\hat{y} \tag{6}$$

where $v(x,y)$ the velocity at the output point (Figure 1). The backward 3D NSPS expression is

$$\psi_{NSPS}(x,y,z,\omega)=\frac{-i\omega}{2\pi}\int_{-\infty}^{\infty}\psi(\hat{x},\hat{y},z=0,\omega)\frac{z}{v(\hat{x},\hat{y})\tilde{r}^2}e^{i\omega\tilde{r}/v(\hat{x},\hat{y})}\left(1+\frac{iv(\hat{x},\hat{y})}{\omega\tilde{r}}\right)d\hat{x}d\hat{y} \tag{7}$$

where $v(\hat{x},\hat{y})$ the velocity at the input points (Figure 2). Finally, for the 3D Weyl operator, the backward extrapolator is

[*] Fourier transform sign convention: $h(\mathbf{x},t)=\mathrm{Re}[1/2\pi\int_0^{\infty}2H(\mathbf{x},\omega)e^{i\omega t}d\omega]$, where only positive frequencies are operated on to preserve Hermitian symmetry (see CREWES MATLAB seismic toolbox function ifftrl).

$$\psi_{Weyl}\left(x,y,z,\omega\right)=\frac{-i\omega}{2\pi}\int_{-\infty}^{\infty}\psi\left(\hat{x},\hat{y},z=0,\omega\right)\frac{z}{\overline{v}\left(x,y;\hat{x},\hat{y}\right)\tilde{r}^{2}}e^{i\omega\tilde{r}/\overline{v}\left(x,y;\hat{x},\hat{y}\right)}\left(1+\frac{i\overline{v}\left(x,y;\hat{x},\hat{y}\right)}{\omega\tilde{r}}\right)d\hat{x}d\hat{y} \tag{8}$$

where $\overline{v}\left(x,y;\hat{x},\hat{y}\right)=.5\left(v\left(x,y\right)+v\left(\hat{x},\hat{y}\right)\right)$ is an average of the velocities at the input and output points[*] (Figure 3). The forward extrapolators are identical except for the sign changes shown in equations (4) and (5).

SHOT-RECORD 3D PRESTACK DEPTH MIGRATION

A basic prestack depth migration algorithm for a single shot record consists of three components, forward extrapolation of the source wavefield to depth z, backward extrapolation of the receiver wavefield to depth z, and an imaging condition that estimates the reflectivity at depth z from the two extrapolated wavefields. The Kirchhoff wavefield extrapolators given by equations (6), (7), and (8) can be coded as a one module that handles the GPSPI, NSPS or Weyl velocity calculation as an option. The imaging condition is implemented as either a stabilized deconvolution or weighted crosscorrelation at zero time (Claerbout 1971).

A number of additional components are required to make the basic algorithm useful for seismic processing, including input and output of seismic data and velocity fields, and weighted summation of migrated shot records. In addition, we plan to run the migration algorithm in a standard seismic processing environment such as Seismic Unix, USP or ProMAX. To facilitate this, an application framework was developed consisting of a number of functional packages with well-designed, public, application programmer interfaces (API's). The low-level details of the packages can be changed, or tailored to a specific environment, without requiring any modifications to the migration code. To speed up development, we used existing software that was already available as open-source (e.g. Seismic Unix) where possible. Conversely, new migration algorithms can also be plugged into the framework with relative ease.

Architecture of the prestack migration algorithm

The following packages form the core of the application framework:

- **Seismic IO**: Reading and writing of seismic trace files. The public interface abstracts the concepts of a data source and sink (e.g. a disk file), and trace objects (header and data samples) that can be read or written sequentially or randomly. Currently only SEGY is supported, but other data formats or sources could be easily implemented.

- **Velocity Model Access**: A set of routines to read and select velocity values from standard format files. The only format currently supported is the Gocad gridded binary file, used in the SEG SALT dataset.

[*] Note that the notation of Margrave and Daley (2001, equation 28) incorrectly suggests that the Weyl velocity is determined at a lateral position half way between the input and output points, rather than as the average of the velocities at the input and output points.

- **Matrix Manipulation**: A suite of routines to manipulate 1D, 2D and 3D matrices of either real or complex data. Much of the functionality duplicates what is available in MATLAB, which made the translation from MATLAB to C relatively straight forward.

- **Message Logging**: A standard way of writing progress and debug messages to the terminal, or to a file, that also supports a parallel processing environment. Debugging parallel programs can be difficult, so a message logging system is essential.

Parallel Implementation

Parallel computing involves the use of more than one computer simultaneously to tackle a single problem. There are a variety of different parallel architectures and schemes in use but they all share some common concepts. A parallel computing environment is made up of multiple compute nodes, which are single machines or CPUs that perform computations. It is often advantageous to designate one or more compute nodes to perform data distribution, collection, and management functions while the other nodes perform the work. In this case the nodes are divided into "master" and "worker" nodes.

One of the primary goals for the implementation of the parallel recursive Kirchhoff migration was to make use of existing computer hardware, including both Unix and Windows workstations. A major advantage is the ready availability of this often under-utilized resource. Disadvantages include slow network connections, different compute nodes speeds and memory availability, and different operating systems.

One measure of parallel computing performance is scalability. This refers to how the performance scales with the number of compute nodes. The best case is linear, where, for example, the program will run 10 times faster on 10 compute nodes than on a single node. A major difficulty in dealing with 3D seismic is the data volume. It turns out that one of the key factors determining the scalability of parallel computing algorithms is the ratio of compute time to data transfer time. In general, the greater the compute/data transfer ratio, the better the algorithm will scale to large numbers of compute nodes. The challenge when processing seismic data is to not saturate the network with data transfer. If this is allowed to happen, the workers will spend most of their time waiting for data to arrive instead of performing computations.

The 3D prestack migration algorithm works in five dimensions: shot, frequency, and the spatial coordinates of x, y, and z. Data in the independent dimensions of shot and frequency can be distributed to worker nodes for computation and then recombined for final output. Shot parallelization is relatively simple. Each compute node (worker) is given a single shot to migrate. The resulting (x,y,z) data cube then returned to the master where they are stacked together to produce the final 3D image. However, the relatively long time required for migration of a shot record makes it desirable to produce a finer-grained parallelism.

Frequency parallelization works because each frequency component is completely independent. The traces from a single shot are transformed from time to frequency, forming a data cube with dimensions of frequency, x and y (see Figure 4). This (f,x,y) cube is sliced along the frequency axis, forming what we call "frequency slabs". Each frequency slab is then migrated by a worker node, resulting in a (x,y,z) image cube covering a subset of the frequency range. These cubes are then returned to the master, where they are combined to create the image. A benefit of frequency parallelization is that the memory requirements for each worker node are greatly reduced. This is important because a single shot record in a 3D seismic survey can contain 1000s of traces.

Concurrent development in MATLAB and C/MPI

Software development followed two concurrent paths. Initial development was done in MATLAB because of its excellent and highly interactive compute and visualization environment. As soon as the core extrapolator was proven out in MATLAB, a C-language framework was constructed that would allow us to quickly port the algorithm from MATLAB to C. The algorithm was then parallelized, first in MATLAB and then in C.

All the initial development was done in a single-processor (non-parallel) environment. This was necessary to maximize simplicity and to keep the debugging process manageable. Parallel computing is inherently more complex than single-node computing, and the results of a parallel algorithm need to be compared to single-node results.

All C code was written using object-oriented programming methodologies. The main benefit of object-oriented programming is better complexity management, resulting in rapid development of code that is both robust and relatively easy to understand. The resulting modular design also allows for a simple parallel implementation, where only 3 out of 20 major modules are aware that they are in a parallel environment. Another major benefit of a modular design is that efforts to optimize speed can be based on runtime statistics. A common programming mistake is to begin optimization for speed before the code is shown to work. This results in increased complexity and development time, often without a measurable benefit because there is no non-optimized code available to compare with the optimized code.

Use of MPI-CH

The Message Passing Interface (MPI) is a specification for the interface to a subroutine library that was formally defined in 1994. MPI quickly became a standard way of writing parallel software since it combined the best features of a number of earlier parallel programming models. The basic function of the library is to provide a means of communication between processes running in a distributed environment using messages (data plus its description). The lowest-level routines are "message_send" and "message_receive", which can be used on their own, but are also used to create a large number of very useful, higher level functions.

There are a number of implementations of the MPI standard that are available. One of the most common is MPICH, currently maintained at Argonne National Laboratory (www.mcs.anl.gov/mpi/mpich). MPICH was chosen because it is free, widely used, and available for a large number of computer architectures and operating systems.

The design of the migration application required that it would run primarily in parallel in a distributed environment, but also on a single processor without an MPI library. The modular design allows switching between the two modes simply by re-compiling and re-linking the application.

3D NUMERICAL IMPLEMENTATION- PRELIMINARY RESULTS

We have implemented equations (6), (7), and (8) in both MATLAB and C as part of our parallel 3D depth migration algorithm. As discussed in the previous section, the overall architecture and detailed modular design is common to both. This has greatly accelerated the coding, testing, and debugging of the C-code. The concurrent development and testing has also helped us identify key geophysical effects related to operator aliasing, operator aperture, and operator tapering. In this section, we present some preliminary results from simple synthetic datasets.

The basic structure of the shot-record migration algorithm consists of a forward extrapolation of the source wavefield, a backward extrapolation of the receiver wavefield, and an imaging condition applied at each depth step. Each of these steps has been tested in isolation. First, we examine the impulse response of the forward and backward extrapolators, and compare results from the MATLAB and C codes. Next, we create a simple synthetic for a shallow horizontal reflector and image the reflector using the MATLAB code. In both tests, the wavefields can be normalized for spherical divergence (and, in the case of impulse responses, for the cosine of the scattering angle). The amplitudes are then extracted along exact hyperbolic diffraction or reflection curves, and the values compared to the exact expected value of 1.0. Variations from the expected value are caused by errors or oversights in the algorithm, coding bugs, or limited aperture effects, while differences between the MATLAB and C-code help us to focus our debugging efforts.

Figure 5 shows the impulse response of the forward extrapolator. The impulse was seeded at $(x, y, z, t) = (0, 0, 0, 200\text{ms})$, with $dx = dy = 25\text{m}$, and $v = 2500\text{ms}^{-1}$. A frequency range of 0-50/60 Hz was extrapolated in a single depth step of $dz = 50\text{m}$. The impulse response is plotted as VA-wiggle traces with the MATLAB result on the left (Figure 5a) and the C-code result on the right (Figure 5b). The center traces at $(x, y) = (0, 0)$ show the expected time shift of +20ms. The laterally varying phase rotation is caused by the near field term (see equation 1). In Figures 5c and 5d, the complex trace amplitudes along the phase-delayed hyperbola are plotted. The amplitude scales in this and subsequent impulse response tests differ because of a constant multiplier in the C-code. The amplitude spectra of the respective center traces are plotted in Figures 5e and 5f. The slightly lower frequency content in the impulse response of the C-code compared to the MATLAB code is caused by differences in the implementation of the 50-60Hz raised cosine bell taper.

In Figure 6, the impulse response is generated using the same parameters as in Figure 5, but corrected for spherical divergence and the cosine of the scattering angle (i.e. normalized by r^2/z). The MATLAB results are plotted on the left and C-code results on the right. The VA-wiggle trace displays in Figure 6a and 6b helped us identify a trace polarity error at $(x, y) = (-400, 0)$ in the C-code caused by an indexing error in the C-code, a bug that has now been fixed. In Figures 6c and 6d, the normalized complex trace amplitudes along the phase-delayed hyperbola are plotted. For Figures 6e and 6f, the near-field term has been removed so the graphs of the normalized complex trace amplitudes are flat, as expected. The vertical scales in Figures 6c, 6d, 6e, and 6f have the same relative percentage extent, showing slight numerical error in the C-code at long offsets (Figure 6f).

Figure 7 shows the impulse response of the backward extrapolator. The impulse is generated using the same parameters as in Figure 5, and plotted as VA-wiggle traces with the MATLAB result on the left (Figure 7a) and the C-code result on the right (Figure 7b). The center traces at $(x, y) = (0, 0)$ show the expected time shift of -20ms. The normalized VA-wiggle trace displays in Figures 7c and 7d show the C-code trace polarity error at $(x, y) = (-400, 0)$, as discussed in the previous paragraph. For Figures 7e and 7f, the near field term has been removed so the graphs of the normalized complex trace amplitudes along the phase-advanced hyperbola are flat, as expected. The vertical scales in Figures 7e and 7f have the same relative percentage extent (and the same as in Figures 6c, 6d, 6e, and 6f). A comparison of Figure 6f (C-code forward extrapolator) with Figure 7f (C-code backwards extrapolator) reveals a slightly larger (max ~0.15%) numerical error at longer offsets in the backward extrapolator.

In Figure 8, a depth step of 10m is used to accentuate the near field term of equation (1). All other parameters are the same Figure 5. Only the C-code results are shown - the corresponding MATLAB results are identical. The impulse responses for the forward extrapolator (left) and the backward extrapolator (right) are plotted as VA-wiggle traces in Figures 8a and 8b, as VA-wiggle traces after amplitude normalization for r^2/z in Figures 8c and 8d, and as normalized complex trace amplitudes extracted along hyperbolic diffraction curves in Figures 8e and 8f. No inconsistencies were found in the near field results.

The second suite of synthetic tests uses exact synthetic data for a 300m deep horizontal reflector in a constant velocity medium $(v = 3000\text{ms}^{-1})$. The recorded wavefield is modelled as a constant velocity zero-phase 0-60Hz bandlimited Green's function (dominant frequency of 15 Hz) at an image source location of $(x, y, z) = (0, 0, 600\text{m})$. This replicates the bandlimited source impulse along a hyperbolic reflection curve with appropriate spherical divergence. The source is seeded at a depth of 25m (Figure 8a) using the same procedure as for the receiver wavefield, but with a Green's function at the true source location of $(x, y, z) = (0, 0, 0)$. In Figure 8b, the surface data has been extrapolated to 25m depth. As with the impulse response tests, the complex trace amplitudes can be extracted along exact hyperbolic diffraction curves and normalized for spherical divergence to give an expected value of 1.0. The r- normalized complex amplitudes for the source and receiver wavefields at 25m are plotted in Figures 8c and 8d, and after extrapolation to 50m in Figures 8e and 8f. Operator aperture, operator taper, and data taper result in errors in the expected normalized amplitudes, which should be 1.0 everywhere.

Figure 10 shows the source and the receiver wavefields for the 300m deep horizontal reflector test of Figure 9, after extrapolation to the reflector depth of 300m. The source wavefield (Figure 10a) has been extrapolated with good amplitude preservation, as indicated by the plot of *r*-normalized complex amplitudes (Figure 10c). The receiver wavefield (Figure 10b) has a smaller aperture of preserved amplitudes (as expected given the limited extent of the survey) with slight operator aliasing and wraparound effects visible below the extrapolated hyperbola and the expected 'infinite' aperture effects visible above the hyperbola. The *r*-normalized complex amplitudes (Figure 10d) are slightly improved by using a different set of extrapolation parameters (Figure 10e), but the corresponding extrapolated wavefield (Figure 10f) shows a slightly larger 'infinite' aperture effect.

Figure 11 is the depth section for the 300m deep horizontal reflector created by applying an imaging condition to the extrapolated source and receiver wavefields at each 25m depth step between 0 and 400m (see Figures 9 and 10). A deconvolution imaging condition is used with a stabilization factor in the denominator. The depth section is plotted in Figure 11a after sinc function interpolation to 5m depth increments. In Figure 11b, the maximum amplitude along the reflector is plotted. Although the expected value of 1.0 is not recovered exactly, the errors appear reasonable given the limited aperture of the data and the large operator apertures required to extrapolate shallow wavefields. Further testing of the algorithms is planned.

CONCLUSIONS

One of the primary goals of the POTSI project is to develop a research-friendly framework that encourages the use of parallel computation facilities. This framework will allow us to efficiently investigate a variety of compute-intensive problems in seismic processing and imaging. Key features of the research framework are: full integration with existing software including the CREWES MATLAB seismic toolbox, Seismic Unix, USP, and ProMAX, concurrent design and coding in MATLAB and C/FORTRAN, and semi-automated archiving.

As our first major research project, we have chosen to design and code a parallel 3D prestack depth migration algorithm based on the 3D recursive Kirchhoff extrapolators introduced by Margrave and Daley (2001). Recursive Kirchhoff wavefield extrapolation in the frequency-space domain allows us to use the Weyl formulation, which should give better estimates of the phase than either the GPSPI or NSPS formulations, and hence reconstruct the extrapolated wavefield with greater accuracy. The algorithm has been coded for single- and multiple-node processing in both MATLAB and C. Simple synthetics tests have greatly accelerated the testing and debugging phase of the software development.

RESEARCH PLANS

We are in the process of creating a parallel 3D imaging facility. We intend to fully test the parallel prestack migration algorithm using simple constant and variable velocity synthetics, first on a 3-node cluster of 2 GHz PC/Linux machines, and then on the ~20 node CREWES cluster to be purchased shortly. The current 3D C-code is designed for the SEG/EAGE Salt data sets. Our first major test will be the 45-shot SEG/EAGE Salt data set, kindly provided by Curtis Oberg at Sandia National Laboratories. Statoil has offered an OBS data set similar to the 45-shot data set, which we intend to process as our first real data set. The 2D recursive Kirchhoff algorithm, now proven in MATLAB, will be converted to parallel C-code. This will allow us to process standard 2D synthetic data sets (e.g. SIGSBEE) and implement GPSPI and NSPS extrapolators. Our focus will then shift to imaging 3D land seismic datasets including real data, and the associated problems of irregular acquisition geometries, sub-optimal resolution in depth migrated images, and near-surface velocity inhomogeneities such as permafrost.

ACKNOWLEDGEMENTS

Funding for this work came from CREWES, POTSI, NSERC, MITACS, and PIMS and we thank all of these organizations. Special thanks to BP, Esso, and C&C Systems, corporate sponsors of the POTSI project.

REFERENCES

Claerbout, J.F., 1971, Towards a unified theory of reflector mapping: Geophysics, **36**, 467-481.

Ferguson, R.J. and Margrave, G.F., 2002, Depth imaging in anisotropic media by symmetric nonstationary phase shift: Geophysical Prospecting, **50**, 281-288.

Grossman, J.P., Margrave, G.F., and Lamoureux, M.P., 2002a, Constructing adaptive, nonuniform Gabor frames from partitions of unity: CREWES Research Report, **14** (this volume).

Grossman, J.P., Margrave, G.F., and Lamoureux, M.P., 2002b, Fast wavefield extrapolation by phase-shift in the nonuniform Gabor domain: CREWES Research Report, **14** (this volume).

Kuehl, H. and Sacchi, M., 2001, Generalized least-squares DSR migration using a common angle imaging condition, 71st Ann. Internat. Mtg: Soc. of Expl. Geophys., 1025-1028.

Larsen, J.A., Margrave, G.F., Lu, H., and Potter, C.C., 1998, Simultaneous P-P and P-S inversion by weighted stacking applied to the Blackfoot 3C-3D survey, CREWES Research Report, **10**, 50, 1-23.

Larsen, J.A., 1999, AVO inversion by simultaneous P-P and P-S inversion: M.Sc. Thesis, University of Calgary.

Margrave, G.F., Henley, D.C., Lamoureux, M.P., Iliescu, V., and Grossman, J.P., 2002: An update on Gabor deconvolution: CREWES Research Report, **14** (this volume).

Margrave, G.F. and Geiger, H.D., 2002: Wavefield resampling during Kirchhoff extrapolation: CREWES Research Report, **14** (this volume).

Margrave, G.F. and Daley, P., 2001, Recursive Kirchhoff extrapolators: CREWES Research Report, **13**

Margrave, G.F. and Lamoureux, M.P., 2001, Gabor deconvolution: CREWES Research Report, **13**.

Margrave, G.F., Stewart, R.R., and Larsen, J.A., 2001, Joint PP and PS seismic inversion: The Leading Edge, **20**, 1048-1052.

Margrave, G.F. and Yao, Z., 2000, Downward continuation from topography with a laterally variable depth step: 70th Annual SEG meeting, Calgary, AB., 481-484.

Margrave, G.F. and Ferguson, R.J., 1998, Explicit Fourier wavefield extrapolators: CREWES Research Report, **10**.

Margrave, G.F. and Ferguson, R.J., 1999, Wavefield extrapolation by nonstationary phase shift: Geophysics, **64**, 1067-1078.

Mi, Y. and Margrave, G.F., 2001, Dual extrapolation algorithms for Fourier shot record migration, 71st Ann. Internat. Mtg: Soc. of Expl. Geophys., 1021-1024.

FIGURES

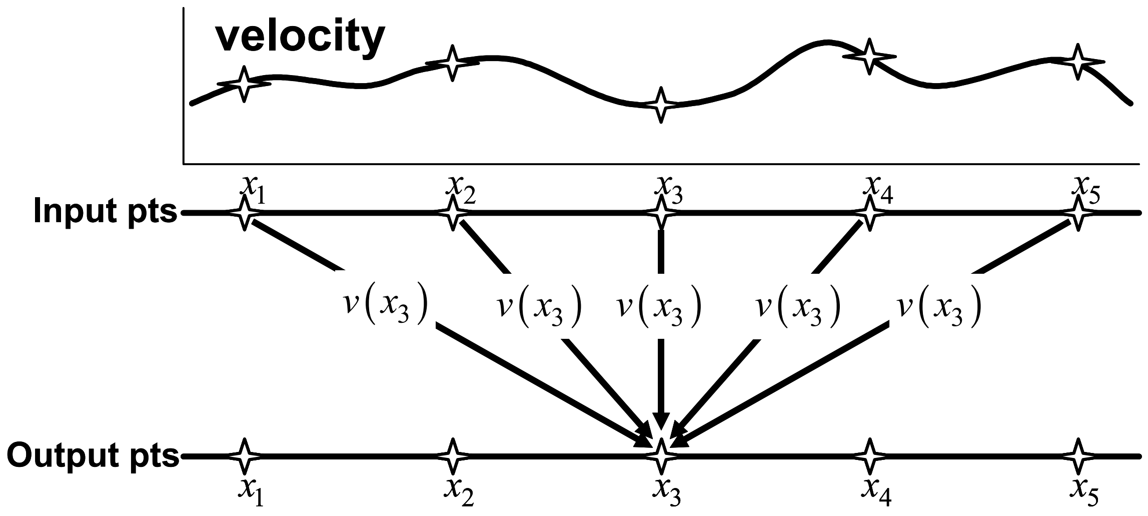


FIG. 1. GPSPI wavefield extrapolation calculates a single output point using straight raypaths from each input point and the velocity at the output point.

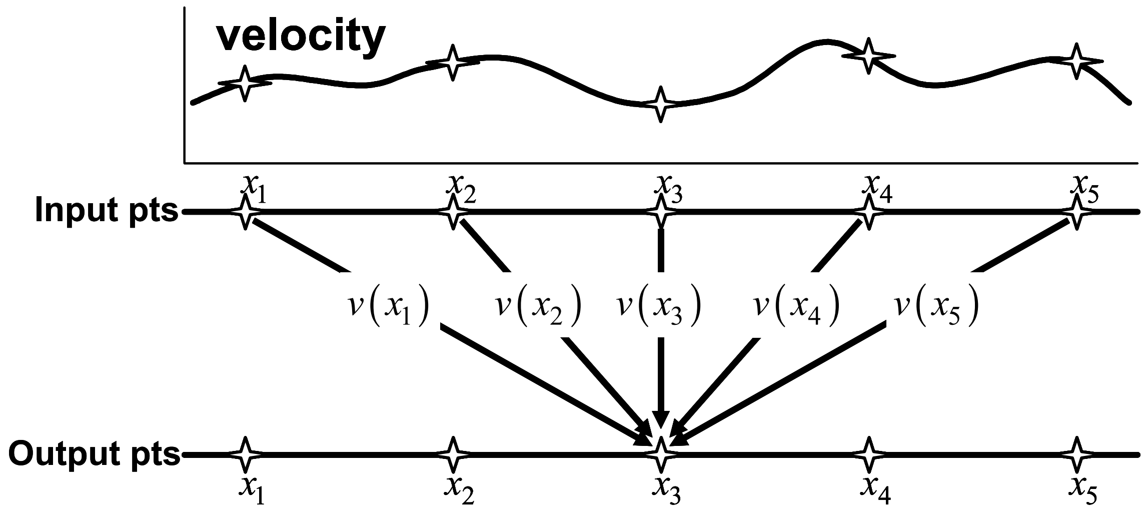


FIG. 2. NSPS wavefield extrapolation calculates a single output point using straight raypaths from each input point and the velocity at the input points.

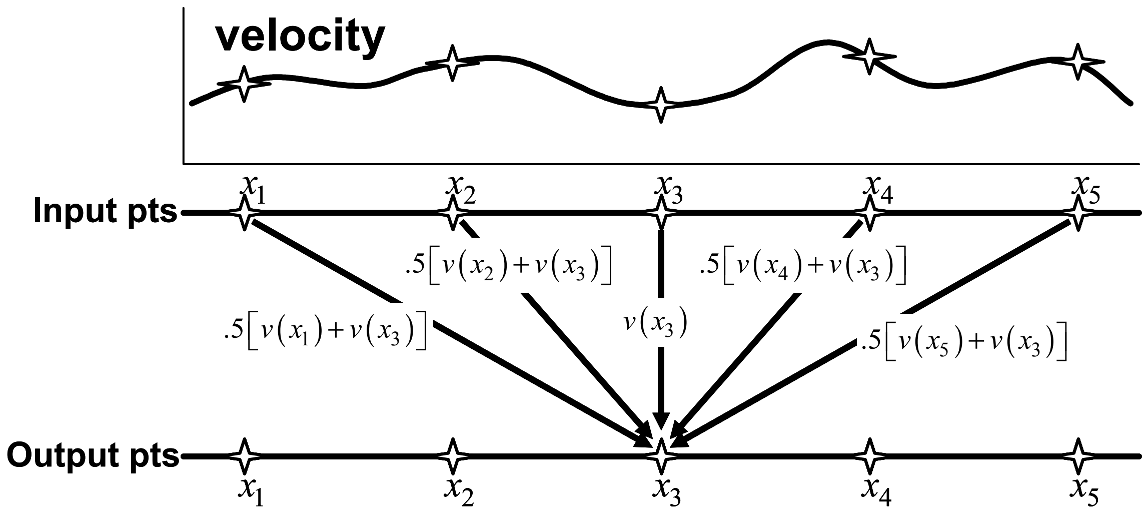
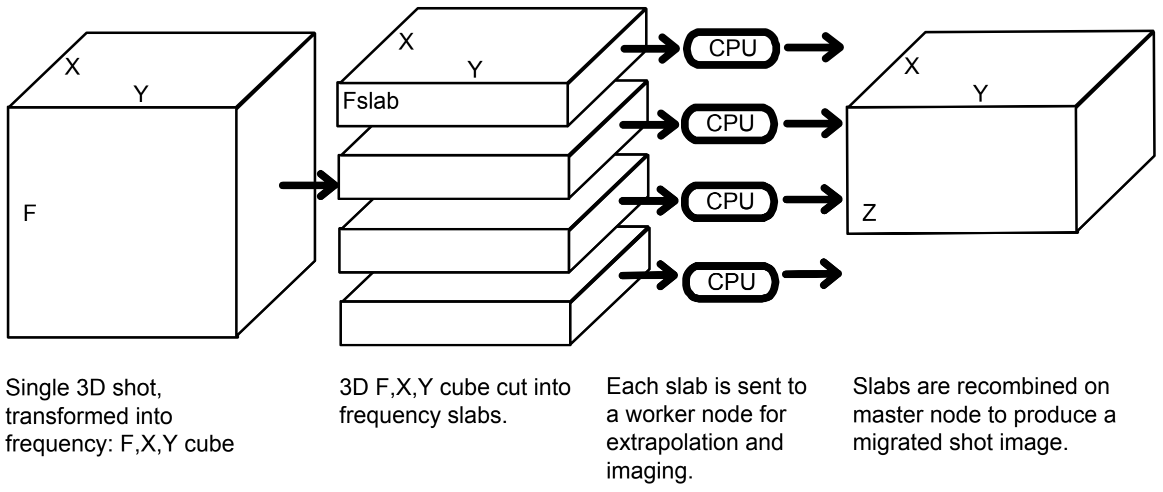


FIG. 3. Weyl wavefield extrapolation calculates a single output point using straight raypaths from each input point and the average of the velocity at the input and output points.



Single 3D shot, transformed into frequency: F,X,Y cube

3D F,X,Y cube cut into frequency slabs.

Each slab is sent to a worker node for extrapolation and imaging.

Slabs are recombined on master node to produce a migrated shot image.

FIG. 4. Prestack (f,x,y) data volume for single prestack shot record is divided into frequency slabs by the master node. The slabs are distributed to worker nodes for forward and backward extrapolation and imaging, then recombined on the master node to create the final image volume.

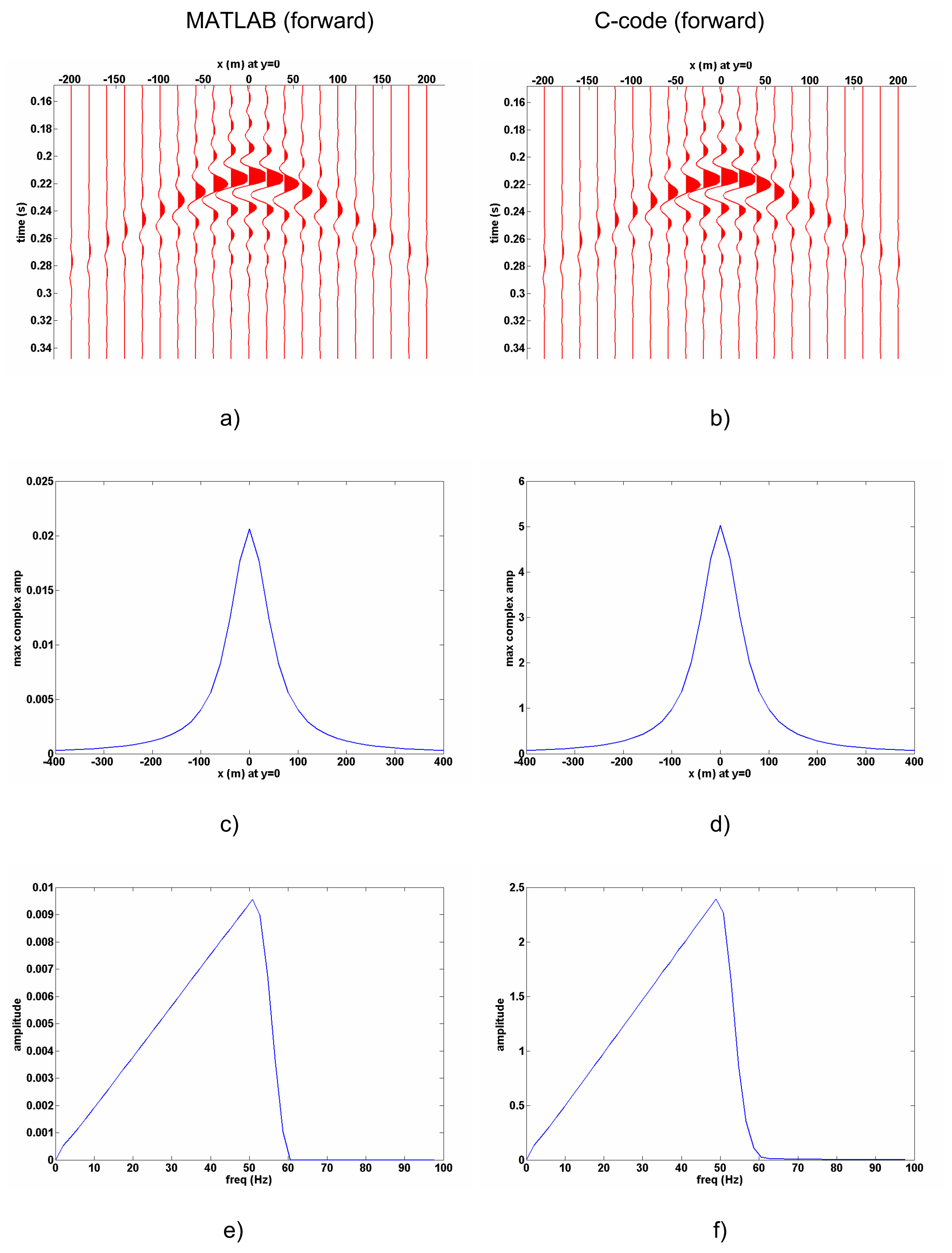
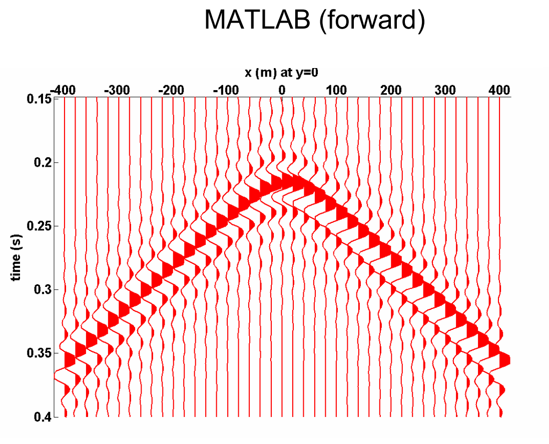
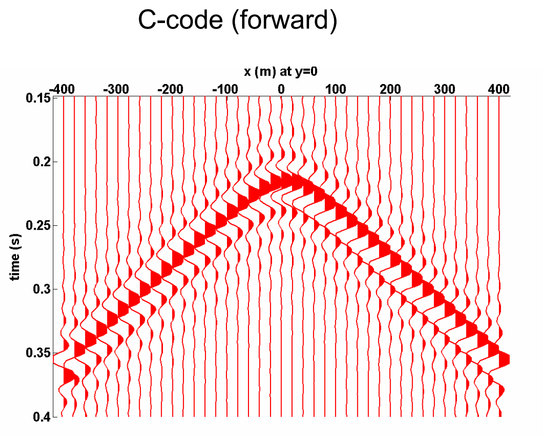


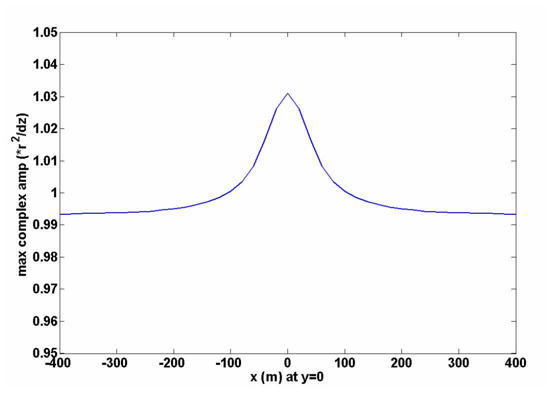
FIG. 5. The MATLAB code (left) and C-code (right) yield almost identical results for the impulse response of forward extrapolator. The impulse was seeded at (x,y,z,t)=(0,0,0,200ms), with dx=dy=25m, v=2500m/s, and dz=50m. In a) and b), the 0-50/60Hz impulse responses at depth 50m are plotted as VA-wiggle traces from x=(-200,200). The centre traces at (x,y)=(0,0) show the expected time shift of +20ms. The laterally-varying phase rotation is caused by the near field term (see equation 1). In c) and d), the complex trace amplitudes along the phase delayed hyperbola are plotted from x=(-400,400). The amplitude scales differ because of a constant multiplier in the C-code. The amplitude spectra of the respective center traces are plotted in e) and f).

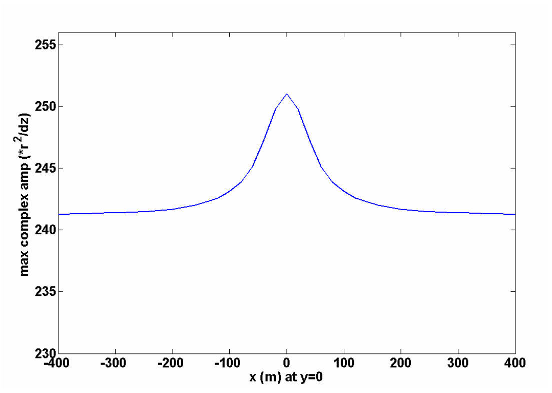




a)

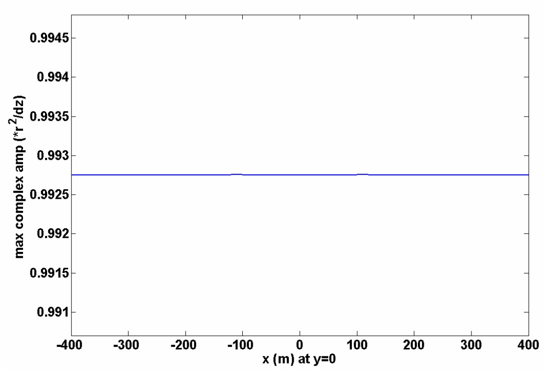
b)

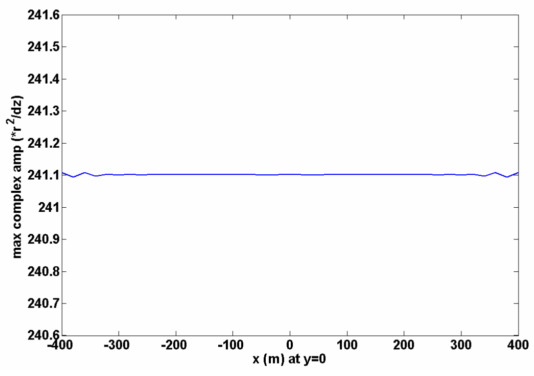




c)

d)

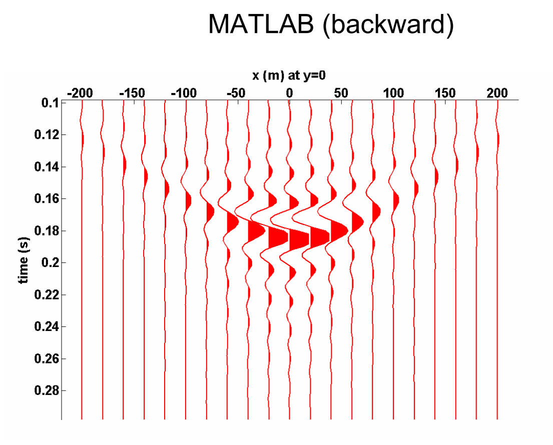


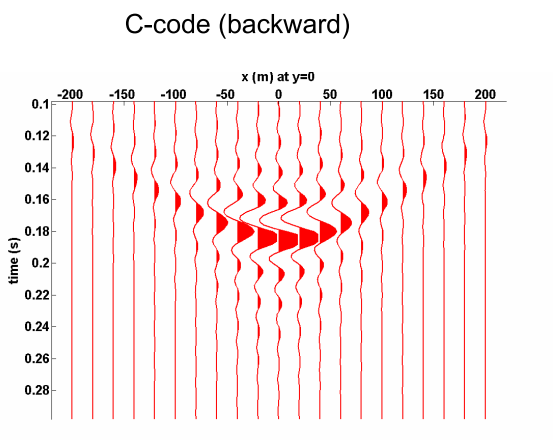


e)

f)

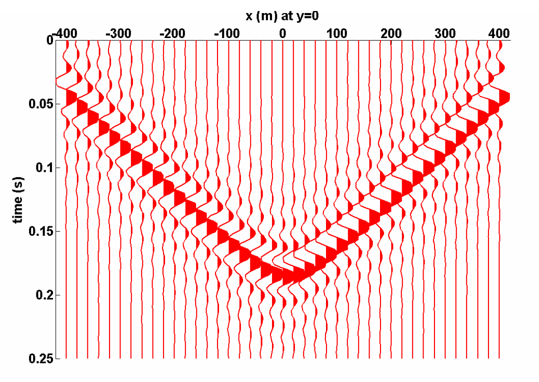
FIG. 6. The impulse response for the forward extrapolator, after amplitude normalization for r^2/z. The impulse is generated using the same parameters as in Figure 5, with MATLAB results on the left and C-code results on the right. The VA-wiggle trace displays in a) and b) reveal a trace polarity error at (x,y)=(-400,0) for the C-code (now fixed). In c) and d), the normalized complex trace amplitudes along the phase delayed hyperbola are plotted from x=(-400,400). In e) and f), the near-field term has been removed (VA-Wiggle traces not shown), so the normalized complex trace amplitudes are flat, as expected. The vertical scales in c) and d) and in e) and f) have the same relative percentage extent, showing slight numerical error in the C-code at long offsets (f).

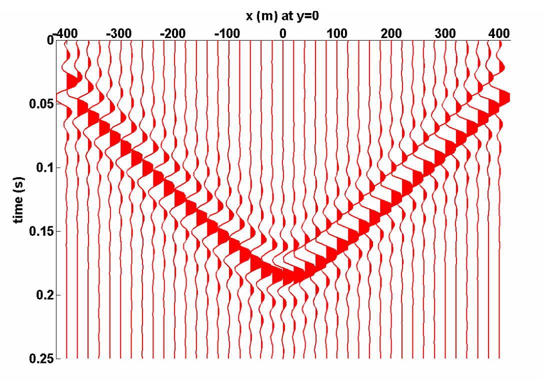




a)

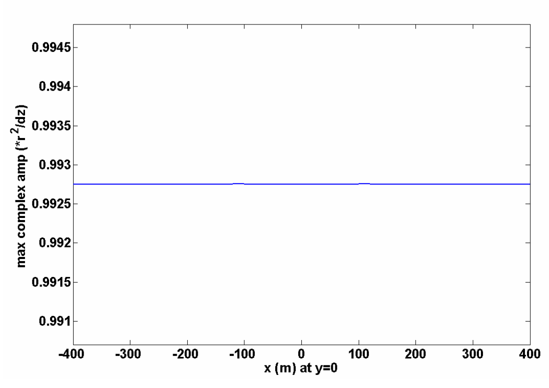
b)

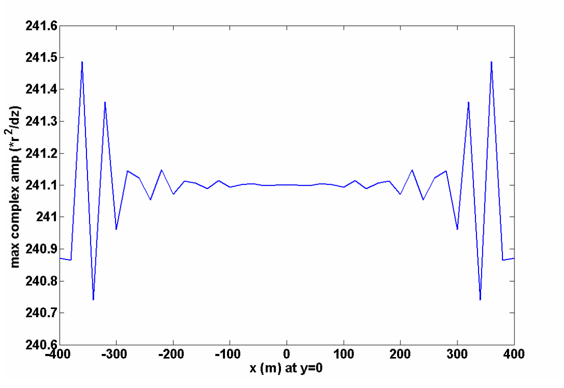




c)

d)



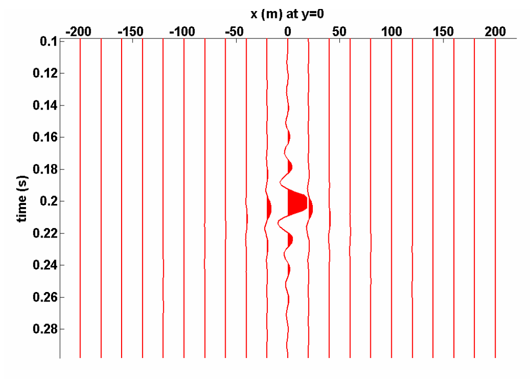


e)

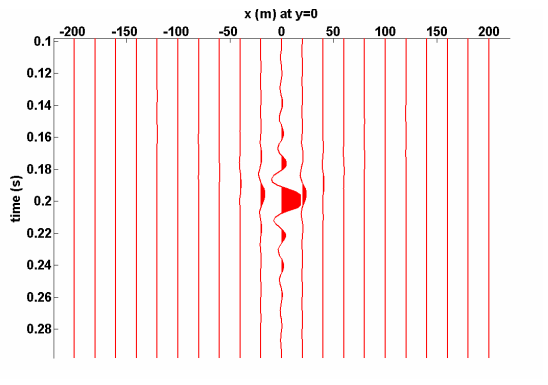
f)

FIG. 7. The MATLAB code (left) and C-code (right) yield almost identical results for the impulse response of the backward extrapolator. The impulse is generated using the same parameters as in Figure 5. In a) and b), the centre traces at (x,y)=(0,0) show the expected time shift of -20ms. The VA-wiggle trace displays in c) and d) show a trace polarity error at (x,y)=(-400,0) for the C-code. In e) and f), the normalized complex trace amplitudes (near-field term removed) along the phase advanced hyperbola are plotted from x=(-400,400). The vertical scales in e) and f) have the same relative percentage extent, showing in (f) a ~0.15% max numerical error in the C-code at long offsets.

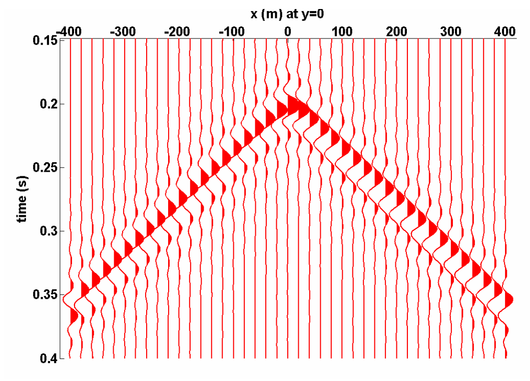
C-code (forward) C-code (backward)



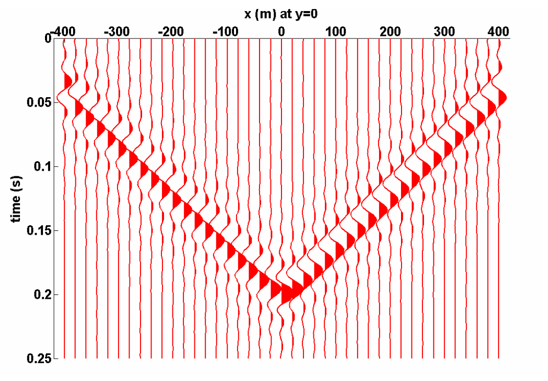
a)



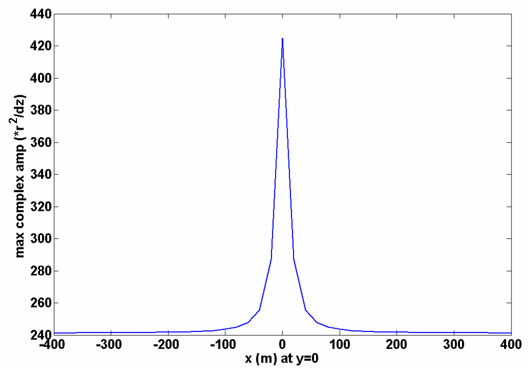
b)



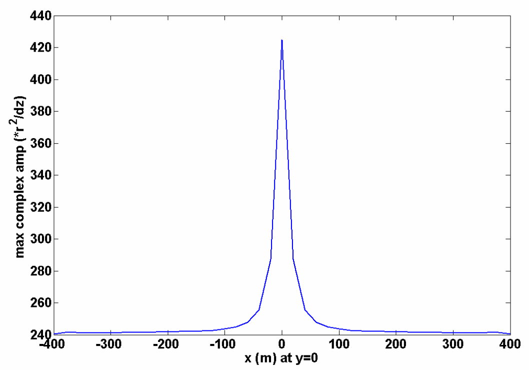
c)



d)

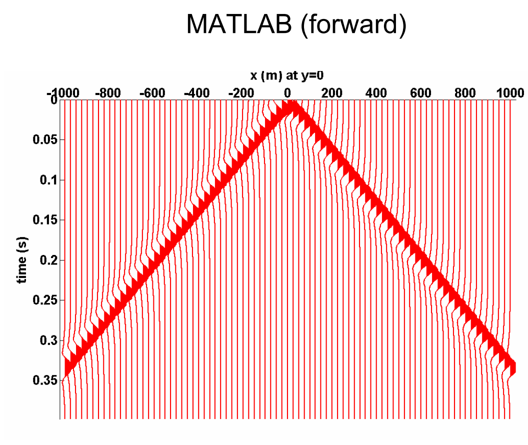


e)

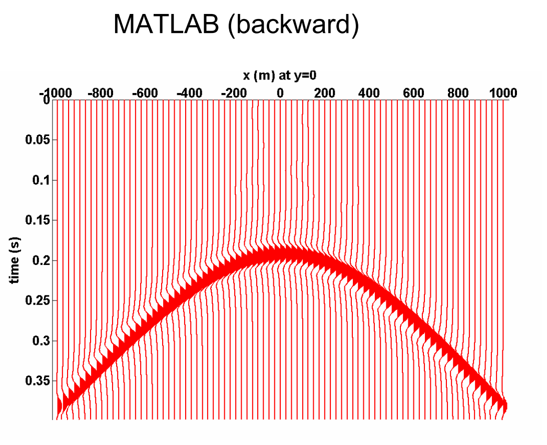


f)

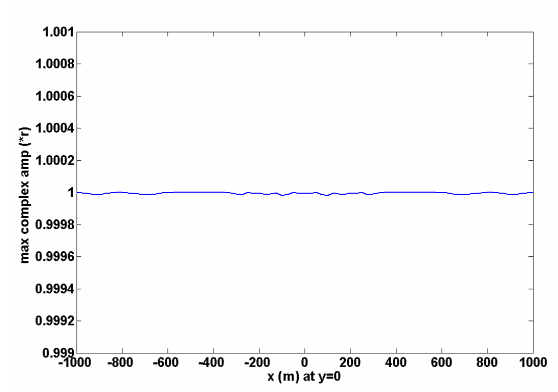
FIG. 8. A depth step of 10m accentuates the near field term in equation (1). All other parameters as in Figure 5. Only the C-code results are shown. The corresponding MATLAB results are identical. The impulse responses for the forward extrapolator are on the left and for the backward extrapolator on the right. In a) and b), the 0-50/60Hz impulse responses at depth 10m are plotted as VA-wiggle traces from x=(-200,200). The impulse responses after amplitude normalization for r^2/z are plotted in c) and d), and the normalized complex trace amplitudes in e) and f).



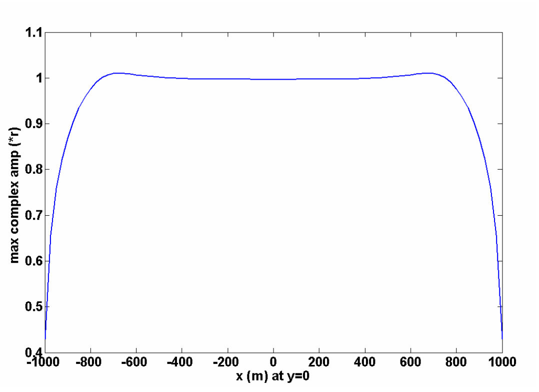
a)



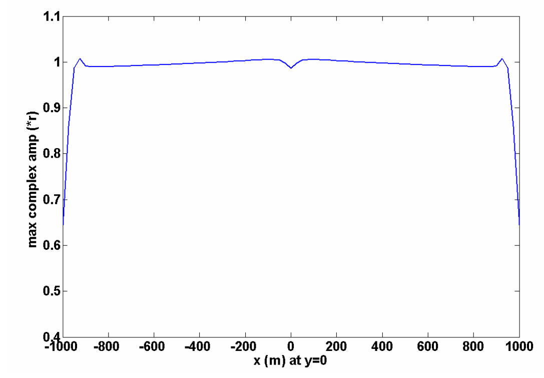
b)



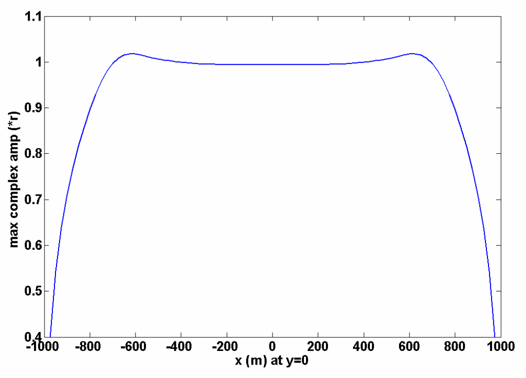
c)



d)

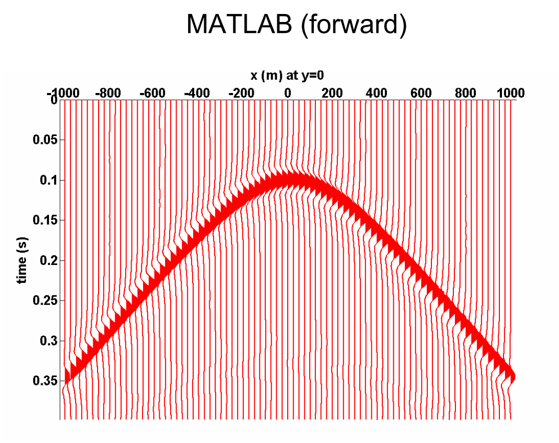


e)

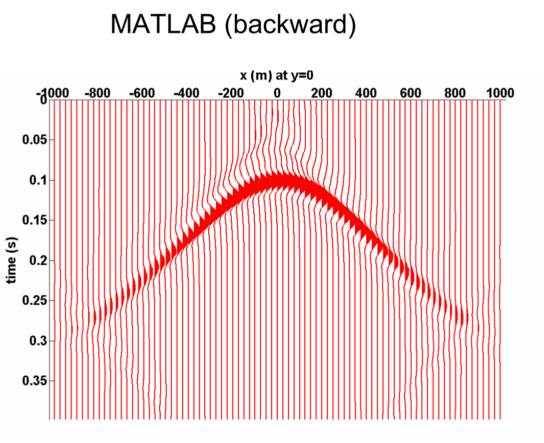


f)

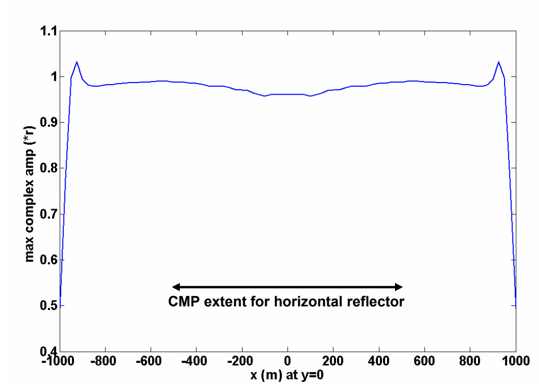
FIG. 9. Imaging a horizontal reflector at 300m depth (constant velocity 3000m/s) requires forward extrapolation of the source wavefield (left) and backward extrapolation of the receiver wavefield (right). 25m depth steps were used. (a) The source is seeded at depth 25m by a constant velocity Green's function at (0,0). The receiver wavefield is seeded as a constant velocity Greens function at an image source depth of 600m. In (b), the surface data has been extrapolated to 25m. The r-normalized complex amplitudes for the source and receiver wavefields at 25m are plotted in c) and d), and after extrapolation to 50m in e) and f). Operator aperture, operator taper, and data taper result in errors in the expected normalized amplitudes, which should be 1.0 everywhere.



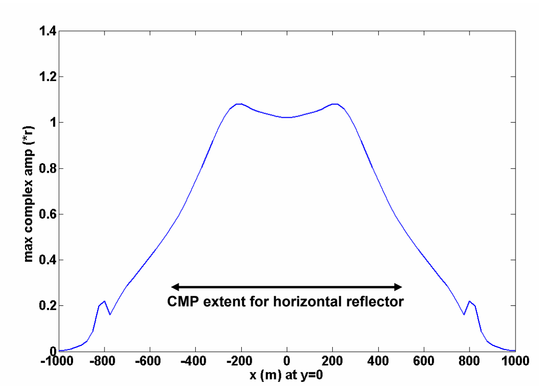
a)



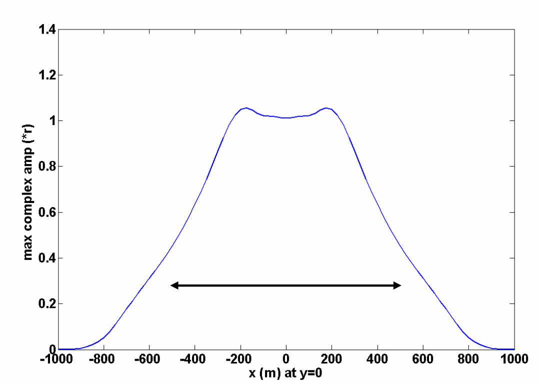
b)



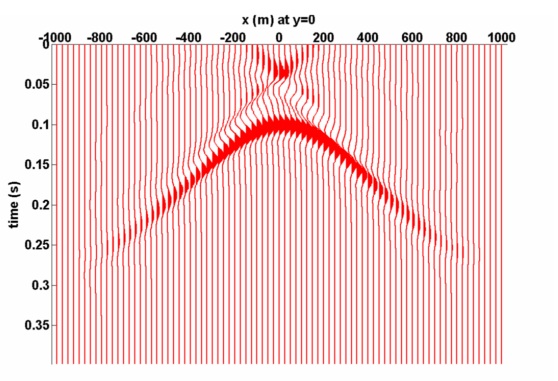
c)



d)

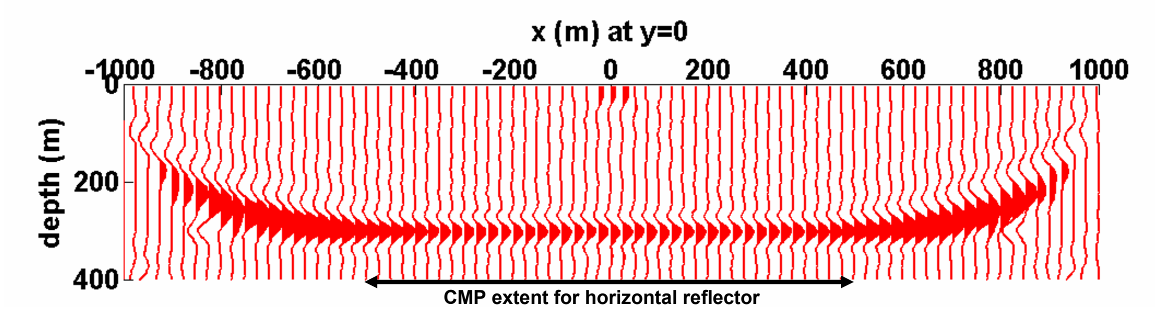


e)

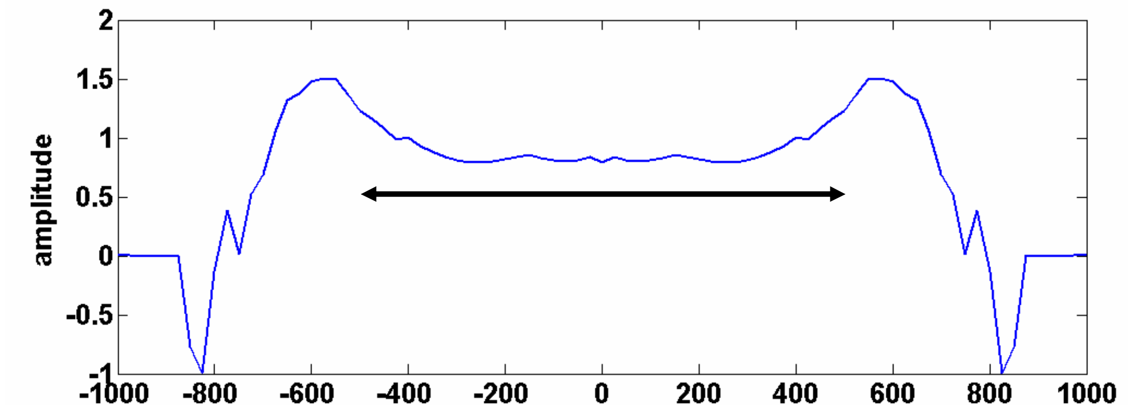


f)

FIG. 10. Source and the receiver wavefields for the 300m deep horizontal reflector test of Figure 9, after extrapolation to 300m depth. The source wavefield (a) has been extrapolated with good amplitude preservation, as indicated by (c) the plot of r-normalized complex amplitudes. The receiver wavefield (b) has a smaller aperture of preserved amplitudes, as expected given the limited extent of the survey, with slight operator aliasing and wraparound effects visible below the extrapolated hyperbola, and expected 'infinite' aperture effects visible above the hyperbola. The r-normalized complex amplitudes (d) are slightly improved with different set of extrapolation parameters (e), but the extrapolated wavefield (f) shows a slightly larger 'infinite' aperture effect.



a)



b)

FIG. 11. The 300m deep horizontal reflector, after imaging the extrapolated source and receiver wavefields at each 25m depth step between 0 and 400m (see Figures 9 and 10). A deconvolution imaging condition was used with a stabilization factor in the denominator. The depth section is plotted in (a), with a sinc function interpolation to 5m depth increments. In b), the amplitude along the reflector at depth 300m is plotted. Testing of the algorithms is in its initial stages. We plan to investigate parameters that control edge and aperture effects visible outside the CMP extent (double-ended black arrow).