Algorithm comparisons for nonstationary phase shift

Robert J. Ferguson and Gary F. Margrave

ABSTRACT

We explore the practical aspects of wavefield extrapolation by nonstationary phase shift (NSPS), specifically runtime. We examine each of the three NSPS domains, space, dual and Fourier, to find which is the most computationally efficient. Our analysis shows that the dual domain is faster than in either the space or Fourier domains for equal operator lengths.

When operator lengths are reduced, that is when the input velocity field is smoothed, the computational effort in the Fourier domain can be greatly reduced. We demonstrate that, for a typical 2-D experiment, significant runtime improvements (8:1) are obtainable when the velocity field is desampled from an interval of 10 m to 160 m (16:1).

A high level programming language (MATLAB) was used to create the prototype of the NSPS extrapolator. Coding a particularly long running program step in C resulted in no run time advantages, though small improvements have been observed for very small data sets (128 traces, 128 samples/trace).

Approximate square root and exponent functions were examined for accuracy and potential for runtime improvement. A 45-degree solution for the square root and the bilinear transform for the exponent were used. Both of these approximations returned disappointing runtimes and, particularly for the bilinear transform, they are prone to error. Error for the square root approximation results from the need to limit scattering angles to less than 45-degrees. Error for the bilinear transform results in incorrect shifting of the higher frequencies (>20 Hz).

INTRODUCTION

Propagation of the seismic wavefield, as recorded at the surface of the earth, back into a simulated earth is the essence of seismic migration. Sometimes described as inverse extrapolation (Berkhout, 1985, pg. 20), migration uses a wavefield extrapolator to achieve an image. Seismic modeling can be described in similar terms and is often thought of as forward extrapolation (Berkhout, 1985, pg. 21, Claerbout, 1985, pg. 6). In both processes, data are extrapolated to a set of new recordings registered at a set of new recording planes (Berkhout, 1985, pg. 20, Claerbout, 1985, pg. 44). To create a post stack migrated section, an imaging condition is invoked such that the parts of the extrapolated recordings at t=0, corresponding to the recording planes inside the simulated earth, together form the migrated result (Berkhout, 1985, pg. 20, Claerbout, 1985, pg. 51). The modeled result is similarly computed using z=0 as the imaging condition (Berkhout, 1985, pg. 20, Claerbout, 1985, pg. 51).

In a recursive Kirchhoff migration a wavefield extrapolator is derived using the wave equation and the Kirchhoff integral for plane surfaces (Berkhout, 1985, Appendix G). In finite difference migration the extrapolator is obtained directly from a discretised version of the wave equation (Berkhout, 1985, Appendix I). In phase shift migration it is obtained from the Fourier-transform of the wave equation (Berkhout, 1985, pg. 20). All extrapolators are theoretically equivalent, but there are many practical differences.

To form a migration algorithm, the extrapolator is applied recursively beginning with the recorded wavefield. The output from the first extrapolation step is then used as the input to the next; the output from this step is used as input to the third. The process is repeated until the imaging condition is satisfied for all the required depths (Berkhout, 1985, pg. 20, Claerbout, 1985, pg. 51).

Each of the extrapolation methods described above have a number of advantages and disadvantages. For example phase shift extrapolation requires constant lateral velocity but will return unconditional stability and high accuracy for scattering angles up to 90 degrees (Holberg, 1988, Stoffa et al., 1989).

As a remedy to the constant velocity requirement of phase shift extrapolation, Gazdag and Squazerro (1984) presented phase shift plus interpolation (PSPI). This method, while attractive intuitively, has been shown to have obvious instabilities (Margrave and Ferguson, 1997, Etgen, 1991). A number of authors have proposed a nonstationary phase shift method, which provides for continuous lateral variation in velocity (Black et al., 1984, Wapenaar, 1992, Margrave, 1997, Margrave and Ferguson, 1997). Black et al. (1984) describe, but do not derive, an f-k wavefield extrapolation technique based on the equivalence of diffraction of optical plane-waves and depth migration. Wapenaar (1992) also proposes nonstationary phase shift in spirit, by proposing that there is a phase shift extrapolation that may provide runtime advantage if the velocity field is smooth. Margrave and Ferguson (1997) derive nonstationary phase shift (NSPS) as a natural outcome of considering wavefield extrapolation in the framework of Margrave's (1997) linear theory of nonstationary filters. They give explicit integral forms for both NSPS and PSPI and show that NSPS has a superior impulse response.

This paper focuses on the implementation of NSPS as a wavefield extrapolator suitable for use in a seismic migration or modeling algorithm. The MATLAB programming environment was chosen due to its ease of use and potential to export code in the C programming language.

METHODOLOGY

Following the linear theory of nonstationary filters (Margrave, 1997) NSPS is examined in three domains, space, space/Fourier and Fourier, to compare computational effort. Computational effort is established by counting the number of operations that are required to extrapolate a monochromatic wavefield one depth step.

Then, based on suggestions by Wapenaar (1992) and Margrave and Ferguson (1997), the computational effect of smoothing the input velocity field is analyzed. We present a number of examples that show the Fourier domain extrapolation of a wavefield through a progressively smoother velocity field, and the runtime benefits that result.

We break the computational effort of Fourier NSPS into individual steps and analyze runtimes. We find that three main programming steps, of a total of seven, form a large portion of the total runtime. The first, the application of the NSPS extrapolator to the input spectrum, is re-coded in the C language and examined for improvement in runtime. The two other expensive steps are the computation of a square root, and the exponentiation of the depth wavenumber. Reasoning that, in a computer, the square root and exponent operations are computationally expensive, we look at the runtime results of approximating them, the former using what is called a 45 degree solution, the later using a 2nd order bilinear transform. Using analytic examples, and a small synthetic model, we compare impulse responses of the square root and exponential approximations to the exact extrapolator to provide a description of the errors.

COMPARISON OF DOMAINS

The linear theory of nonstationary filters (Margrave, 1997) describes three domains in which the application of a nonstationary filter, such as the wavefield extrapolation, is possible. The three domains are space, dual and Fourier. We will simply write down the appropriate descriptions here, noting that the primed variables are used to separate 'input' and 'output' stages of computation. For example, in the case of the space domain x refers to the space variable upon output, and x' refers to the space variable upon input.

The space domain form of NSPS is (Margrave and Ferguson, 1997):

$$\Psi(x,\Delta z,\omega)=\int_{-\infty}^{\infty} a(x-x',x',\omega)\Psi(x',0,\omega)dx', \tag{1}$$

where $a(x-x',x',\omega)$ is the wavefield extrapolator (a convolution matrix) expressed in the space domain, $\Psi(x',0,\omega)$ is the input wave-field at a reference depth of z=0, and $\Psi(x,\Delta z,\omega)$ is the resulting wavefield at a depth Δz from the reference depth z=0. Because there is no exact space domain expression for $a(x-x',x',\omega)$ (Black et al., 1984) we choose to compute the exact expression in the Fourier domain and then inverse Fourier-transform (IFT) it back to the space domain. The nonstationary extrapolator is (Margrave and Ferguson, 1997):

$$\alpha(k_x,x',\omega)=e^{i\Delta z\sqrt{\frac{\omega^2}{v^2(x')}-k_x^2}}. \tag{2}$$

The nonstationarity of the extrapolator is expressed by its having a different set of values in k_x for each input position x'; later we will refer to this extrapolator as residing in the 'dual' domain. Applying the 1-D inverse Fourier transform (IFT) to the first coordinate of equation (2) gives the space domain expression for $a(x,x',\omega)$.

A shifting process is then applied to $a(x, x', \omega)$ which places the zero lag of each column at a location corresponding to the diagonal of the matrix, resulting in the convolution kernel $a(x - x', x', \omega)$.

The dual domain form of NSPS is (Margrave and Ferguson, 1997):

$$\Psi(x, \Delta z, \omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \Psi(x', 0, \omega) \alpha(k_x, x', \omega) e^{-ik_x x'} dx' e^{ik_x x} dk_x \, . \tag{3}$$

By describing this equation as a ‘dual domain’ expression we are referring to the form of the extrapolator, and the fact that it is a function of coordinates k_x and x', thereby existing simultaneously in both the space and wavenumber domains.

In the Fourier domain the NSPS expression is (Margrave and Ferguson, 1997):

$$\Psi(x, \Delta z, \omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \varphi(k_x', \Delta z, \omega) A(k_x, k_x - k_x', \omega) dk_x' e^{ik_x x} dk_x \, , \tag{4}$$

where, $\varphi(k_x', \Delta z, \omega)$ is the FT of the input wavefield along the space axis (x'). The extrapolator in equation (4), $A(k_x, k_x - k_x', \omega)$, is formed from the FT along the x' axis of the extrapolator in equation (2). The FT is followed by a shifting procedure that places the zero lag of each column at a location corresponding to the diagonal of the matrix, followed by a matrix transpose. The result is a purely Fourier domain extrapolator $A(k_x, k_x - k_x', \omega)$, hence the descriptive title ‘Fourier domain NSPS’.

Using the equations for NSPS in the space, dual and Fourier domains we will now consider them in terms of numbers of computer operations. That is, we will count the number of operations each version of NSPS takes to propagate a monochromatic wavefield a single depth step. To describe this process consider a recorded wavefield, for example a poststack seismic section recorded at z=0, consisting of N traces. The Fourier-transform along the time axis, and extraction of a single frequency ω, results in an $N \times 1$ dataset, $\Psi(x', 0, \omega)$, suitable for input to NSPS in the space domain (equation 1) or dual domain (equation 3). Fourier-transform of $\Psi(x', 0, \omega)$ over x', results in an $N \times 1$ vector, $\varphi(k_x', 0, \omega)$, suitable for input into the Fourier domain NSPS (equation 4).

Computation of the extrapolator (equation 2), results in an $N \times N$ matrix, $\alpha(k_x, x', \omega)$ for multiplication with $\Psi(x', 0, \omega)$ in the dual domain. In the space and Fourier domains, $\alpha(k_x, x', \omega)$ is either inverse Fourier-transformed (space) or forward Fourier-transformed (Fourier) and then phase shifted prior to application.

Comparison of NSPS proceeds by breaking the processes into their component operations and tabulating the results (Figure 1).

Dual	Fourier	Space
Build: $\left[\alpha e^{ik_x x'}\right]$	Build: $\left[\alpha e^{i\delta k_x x'}\right]$	Build: $\left[\alpha e^{ik_x \delta x'}\right]$
Multiply: $\left[\alpha e^{ik_x x'}\right]\left[\varphi_o\right]=\left[\varphi_{\Delta z}\right]$	$FT\{[\alpha]\}\rightarrow[A]$, $N^2 logN$	$IFT\{[\alpha]\}\rightarrow[a]$, $N^2 logN$
$IFT\{[\varphi_{\Delta z}]\}$, $NlogN$	Multiply: $[A][\varphi_o]=[\varphi_{\Delta z}]$	Multiply: $[a][\Psi_o]=[\Psi_{\Delta z}]$
	$IFT\{[\varphi_{\Delta z}]\}=[\Psi_{\Delta z}]$, $NlogN$	

Fig 1. Comparison of operations required by NSPS in the dual, Fourier and Space domains to compute one extrapolation step. The rows represent the main processing steps. FTs of vectors are distinguished from FTs of matrices by the number of operations required as a function of the number of input x locations (N).

Figure (1) is summarized as follows (note this summary is valid only if the operator lengths (*N*) in each domain are the same):

1) We will ultimately apply NSPS recursively in a migration or modeling algorithm, over a large number of depth steps, so we ignore computation of $\varphi(k_x',0,\omega)$ in terms of runtime. Then, NSPS in the Fourier domain is slower than the space domain by an IFT of a vector $(\varphi_{\Delta z})$ at a cost of $NlogN$ operations every Δz.

2) If we neglect the cost of calculation of $e^{ik_x x'}$ required by NSPS in the dual domain then, NSPS in the Fourier domain is slower by one FT over the columns of a matrix $(FT\{[\alpha]\})$ every Δz, at a cost of $N^2 logN$ operations.

3) NSPS in the space domain is slower that the dual domain by the difference between an IFT over the columns of a matrix $(IFT\{[\alpha]\})$ and a 1-D IFT $(IFT\{[\varphi_{\Delta z}]\})$ every Δz, at a cost of *NlogN(N-1)* operations.

Given the proceeding observations above we have in order of increasing speed: Fourier $\rightarrow$ space $\rightarrow$ dual, the fastest NSPS domain being the dual domain. Of course these observations are based solely upon the assumptions in the analysis above, and include no attempts to reduce computational effort by manipulation of the input seismic data or velocity profile.

THE ADVANTAGE OF FOURIER DOMAIN NSPS: THE EFFECT OF SMOOTHING THE VELOCITY FIELD

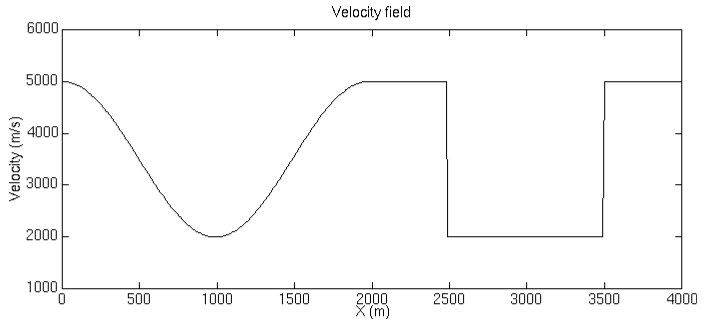
As the previous section showed, NSPS in the dual domain is conceptually the fastest of the three domains, according to the assumptions that were listed. What was hinted at, and what we will now explore, is the possibility of reducing the computational effort of NSPS by smoothing the velocity field. Smoothing the velocity field amounts to reducing the k_x' bandwidth, or alternatively, desampling the velocity field in x'.

The effect of velocity smoothing on NSPS in the Fourier domain is a reduction of the k_x' Nyquist of the velocity field. This results in a reduction in the number of terms lying outward of the main diagonal of the matrix representing $A(k_x, k_x - k_x', \omega)$ (Margrave and Ferguson, 1997, Wapenaar (1992) has also proposed the spirit if not the method of such a scheme). By reducing the number of non-zero elements in $A(k_x, k_x - k_x', \omega)$, a very efficient algorithm is possible which only computes and applies nonzero elements.

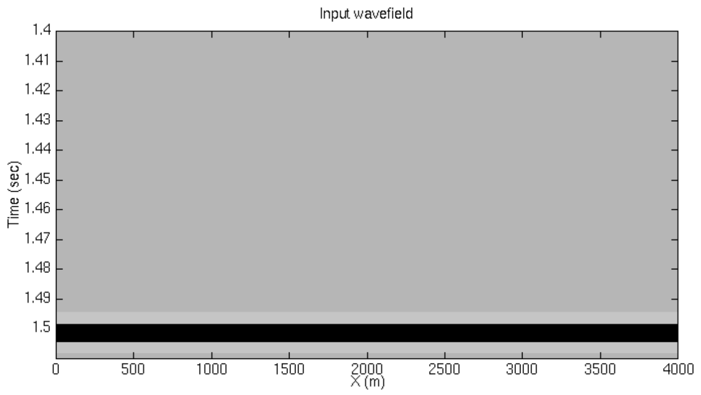
RUNTIME EXAMPLES FOR FOURIER DOMAIN NSPS

To test the effect of smoothing the input velocity field, and to quantify the runtimes of the NSPS algorithm in MATLAB, a synthetic data set was devised. The velocity field of Figure (2a) was chosen such that there was large velocity variation across the line. One side of the velocity field varies smoothly and the other side varies abruptly to demonstrate the effect of smoothing.

Because we intend to develop a 2-D depth migration algorithm based on NSPS, as a precursor to prestack migration, we chose the data dimensions to be typical of common post stack experiments. Thus, our synthetic wavefield (Figure 2b) is 400 traces by 1001 samples, with a CDP interval of 10 m and sample rate of .001 ms.



(a)

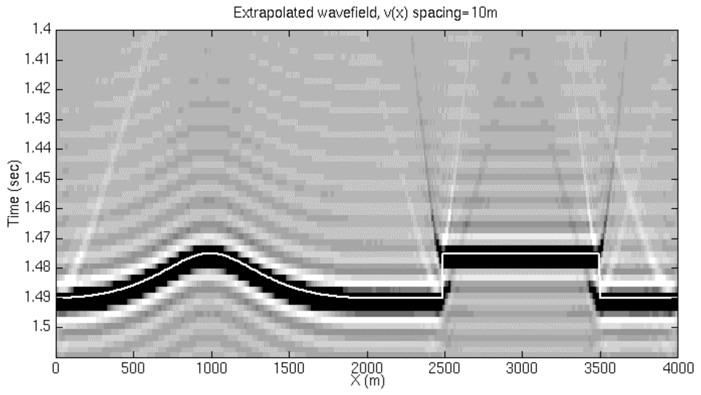


(b)

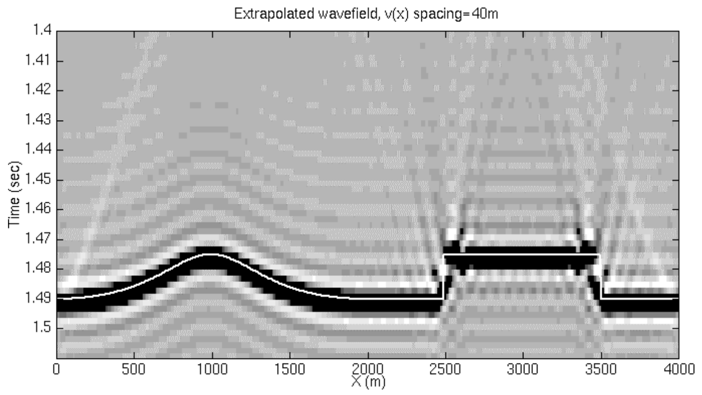
Fig 2. (a) Velocity field. (b) Input wave field.

The following Figures and runtimes are based on a single extrapolation (one Δz step), of the wavefield of Figure (2b), through the velocity field of Figure (2a), using the Fourier domain form of NSPS. The runtimes specifically correspond to execution of NSPS in MATLAB.

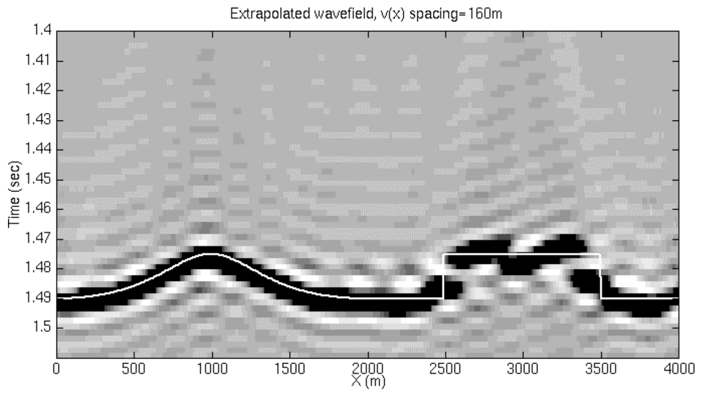
Figure (3a) shows the extrapolated wavefield using a velocity interval of 10 m (a velocity sample for CDP location); the exact zero offset result is plotted as a white line. Clearly, NSPS has done a very good job by coping with both the smoothly varying velocities (0-2500 m), and discontinuously varying velocities (2500-4000 m). Figure (3b) shows the result using a velocity interval of 40-m. Again; NSPS has done a very good job with only minor disruption of the diffractions at the velocity discontinuities (2500 m, 3400 m). Figure (3c) is the result using a 160-m velocity interval. When we confine out attention to the step feature between 2000 and 4000 m we see that have gone too far in desampling the velocity field. But, where our velocity function is smooth, as is the case between 0 and 2000 m, NSPS works very well, and will withstand even more smoothing.



(a)



(b)



(c)

Fig 3. NSPS extrapolation of the wavefield of figure (2b) using the velocity field of Figure (2a). (a) One velocity per trace (b) One velocity every 40 m, (c) One velocity every 160 m.

Figure (4) gives a breakdown of the runtimes of each of the above extrapolation experiments into what we define as the major components in terms of runtime. We see that, for a single extrapolation step, the fully sampled runtime is a little less than 24 minutes in MATLAB. As we smooth the velocity field we see that the runtime is reduced to a little more than seven minutes for a 40-m spacing, and less than three minutes for a 160-m spacing.

Further examination of Figure (4) shows that there are four dominant programming steps in terms of run time. The first two, the square root (step 2) and exponent (step 4), are related of the computation of the extrapolator $\alpha(k_x, x', \omega)$. The next two, (6) and (7), correspond, respectively, to the FFT of $\alpha(k_x, x', \omega)$ (step 6), and the construction/application of $A(k_x, k_x - k_x', \omega)$ to the input spectra (step 7). As we compare the four runtimes to the total times, as the velocity function is smoothed, we see that they all, save the FFT, remain consistently dominant relative to the total time, particularly the $A(k_x, k_x - k_x', \omega)$ step.

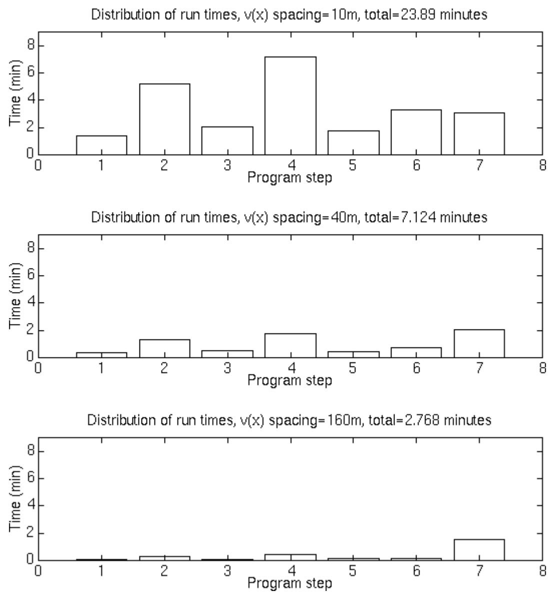


Fig 4. Break down of runtimes for NSPS, using three velocities corresponding to those of Figure (3). The program steps are as follows: (1) compute k_z^2, (2) square root, (3) damping of evanescent energy, (4) exponent, (5) apply absorbing boundaries, (6) Fast Fourier transform of $\alpha(k_x, x', \omega)$ along x', (7) application of $A(k_x, k_x - k_x', \omega)$.

We considered weather we could reduce the runtime of step 7 by rewriting it in C. We then re-executed the three experiments, and the runtime results are given in Figure (5). As we can see, the compiled version of step seven has not noticeably reduced its execution time. We have had some success in reducing the runtime of a much smaller dataset, 128 traces by 128 time samples, but such a small size is hardly representative of a real seismic experiment.

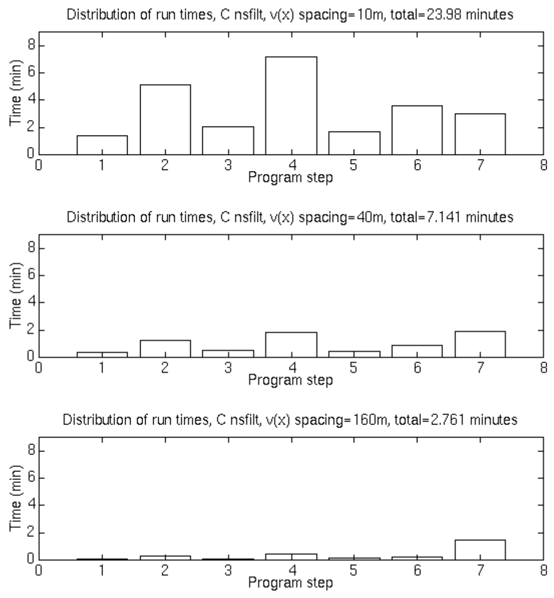


Fig 5. Runtime results for compiled step 7. No runtime gains are apparent. See Figure (6) for a description of each step.

SQUARE ROOT APPROXIMATION

We then considered whether improvements could be made in the computation of the square root, (step 2). We reasoned that, in a computer, the computation of the square root, in this case the square root of a matrix, was a computationally expensive step. We further reasoned that a series approximation might be computationally faster without sacrificing accuracy in geologic conditions that would tolerate, for example, a 45-degree scattering angle. Thus, based on the 45-degree solution (Claerbout, 1985, pg. 79) we replaced the square root with the following approximation:

$$k_z(k_x, x', \omega) = \sqrt{\frac{\omega^2}{v^2(x')} - k_x^2} \approx \frac{\omega}{v(x')} - \frac{k_x^2}{2\frac{\omega}{v(x')} - \frac{v(x')k_x^2}{2\omega}} \ . \qquad (5)$$

Figure (6) shows a comparison between the 45 degree solution and the exact solution for $v(x') = 3000m/s$ and $\omega = 120Hz$. We see that they are equal between $k_x = 0$ and $k_x \approx 0.02$, beyond which the approximation over estimates k_z. These values correspond approximately to zero and 45 degrees (Claerbout, 1985, pg. 78).

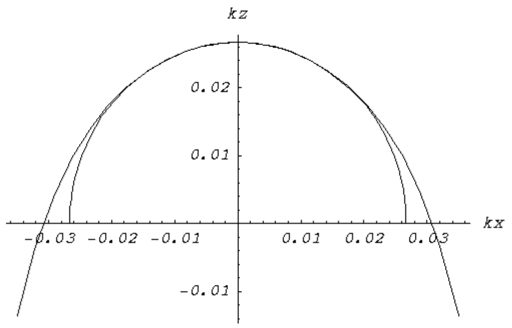
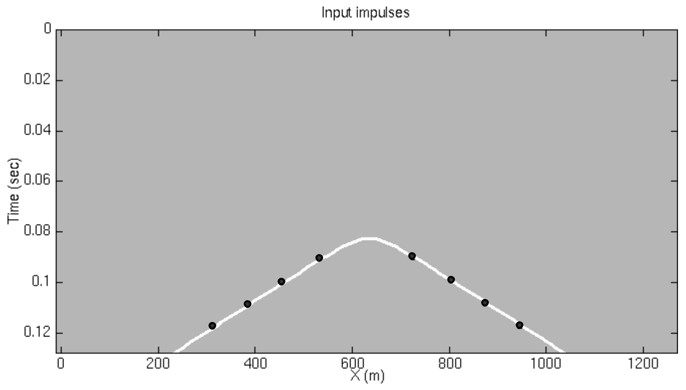


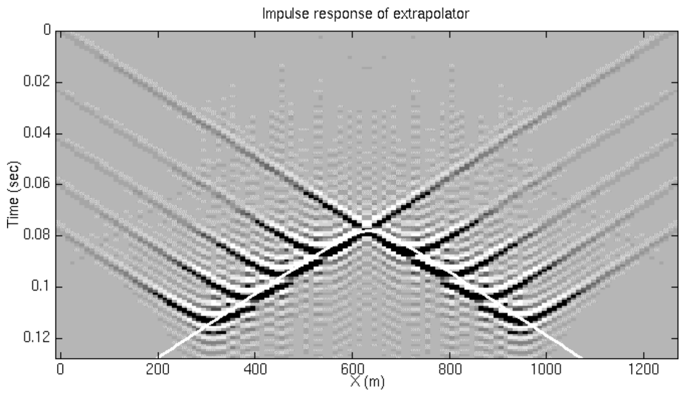
Fig 6. Comparison of solutions to $k_z(k_x, x', \omega)$, where $v(x') = 3000 m/s$ and $\omega = 120 Hz$. The inner curve is the exact solution. The outer curve is the 45-degree solution.

Examples of a set of impulse responses for 45-degree solution are compared to those of the exact expression in Figure (7). In these examples the impulses in Figure (7a), if extrapolated correctly, should be focused to the apex of the hyperbola corresponding to that shown as a white curve drawn on Figure (7a) (the curve has been time shifted to aid comparison). Extrapolation of the impulses by the exact expression is shown in Figure (7b) and shows the expected focused result.

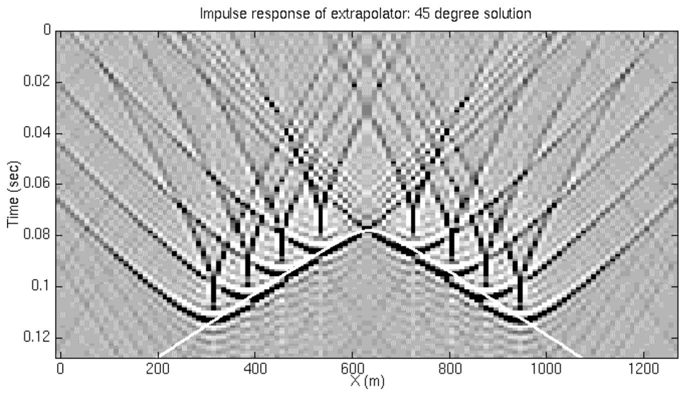
The 45-degree version of the experiment is shown in Figure (7c) and, as would be expected, the focusing of the diffraction energy is reliable only to a certain point along the diffraction hyperbolas, about the 45-degree point. We notice also that the ‘heart shape’ of the 45 degree operator (see Claerbout, 1985, pg. 232) is evident for each diffraction response (the heart shapes would be more obvious using larger axes – here we are only seeing the cleft in the heart).



(a)



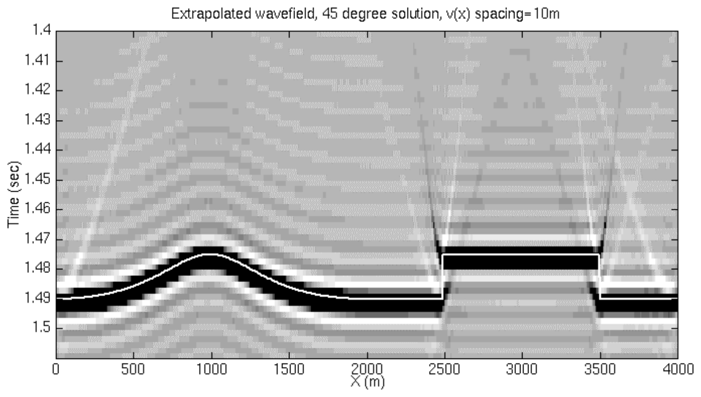
(b)



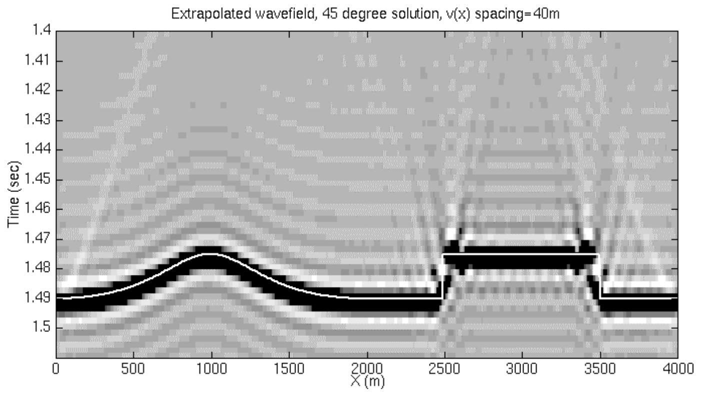
(c)

Fig 7. Comparison of impulse responses of exact extrapolator to 45 degree solution. (a) input impulses, (b) exact extrapolation focusses at apex of hyperbola, (c) extrapolation using 45 degree solution is dispersive for dips beyond 45 degrees.

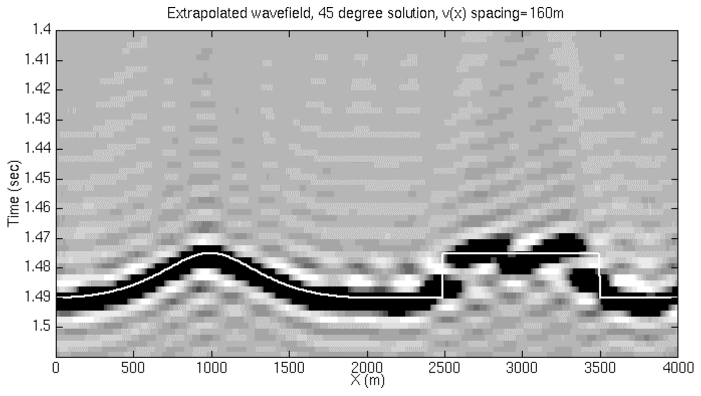
Extrapolation of the wavefield of Figure (2a) is computed using the above approximate square root method. Figure (9) shows that, for a flat event, we have retained accuracy in extrapolation for all three-velocity fields, however, as Figure (10) shows, we have not reduced the total runtime.



(a)



(b)



(c)

Fig 8. NSPS extrapolation (45 degree solution) of the wavefield of figure 2(b) using the velocity field of Figure 2(a). (a) One velocity per trace, (b) One velocity every 40 m. (c) One velocity every 160 m.

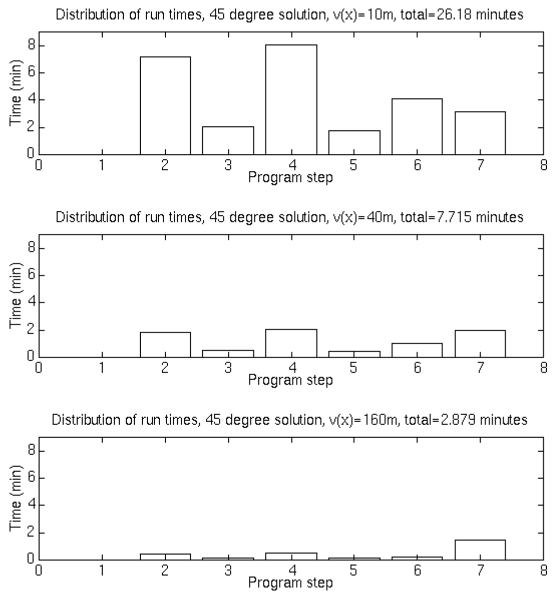


Fig 9. Runtime results for 45-degree solution. No runtime gains are apparent. See Figure (6) for a description of each step.

BILINEAR TRANSFORM

We next considered ways in which we could speed the computation of the exponent (step 4). At first one might consider a series approximation hopeful that, in this case, a runtime improvement might result. However, such an approximation leads to the amplitude spectrum of the extrapolator being less than or greater than unity. After recursive application, for example in a migration routine, the amplitudes will either grow or decrease, ultimately having the potential to overflow or underflow machine precision. Thus, a bilinear transform (a good description is given by Karl, 1989, pg. 54) was used which approximates the exponential, and has an amplitude spectrum of unity for all orders of approximation. The form of the bilinear transform of the extrapolator that we use her is:

$$\alpha(k_x, x', \omega) = e^{i\xi} \approx \frac{e^{i\xi/2}}{e^{-i\xi/2}} = \frac{1 + i\xi/2 + (i\xi/2)^2/2!\ldots}{1 - i\xi/2 + (i\xi/2)^2/2!\ldots}, \xi = \Delta z\sqrt{\frac{\omega^2}{v^2(x')} - k_x^2} \ . \qquad (6)$$

The set of impulse responses, described above (Figure 7a), are used to find the appropriate order for the bilinear transform. Figure (10) shows the extrapolation using the second order approximation to equation (6). (The first order approximation was found to inadequately focus the hyperbola, and the third order is expected to be too computationally expensive i.e. each order in the expansion requires a separate matrix.)

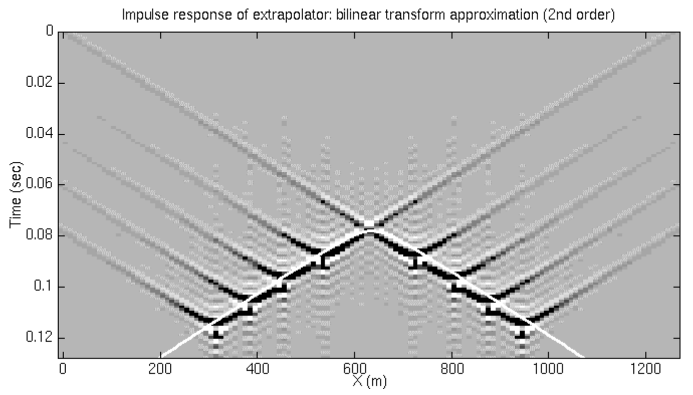
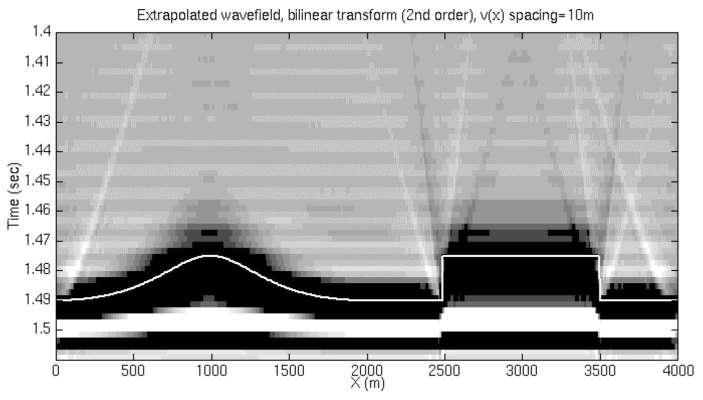
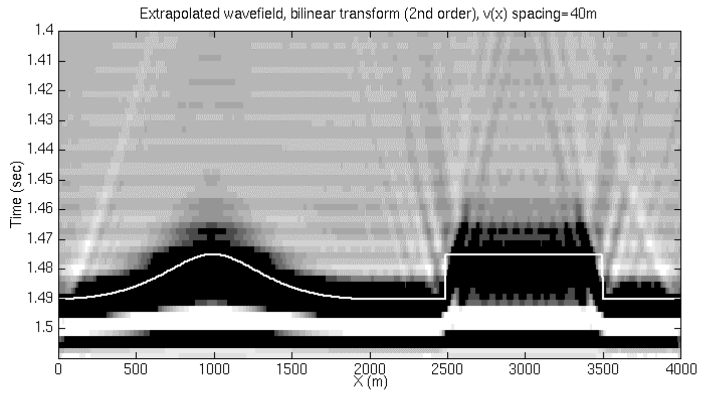


Fig 10. Extrapolation of impulses of Figure 7(a) by bilinear transform. The second order solution, used here, is found to be the lowest approximation that focusses this image.

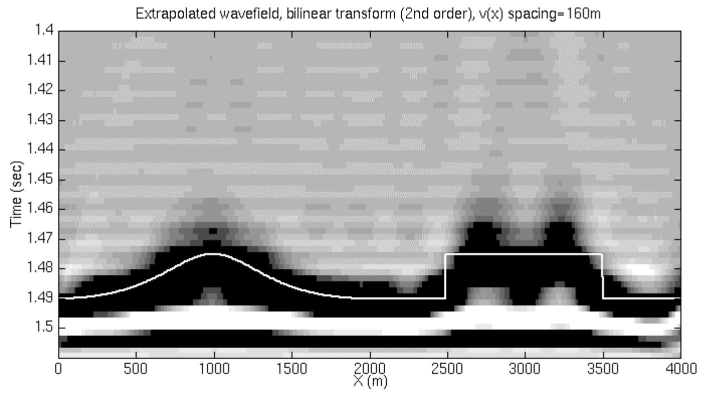
Extrapolation of the wavefield of Figure (2) results in very low-frequency results (Figure 11), and an increase, instead of a decrease, in runtime (Figure 12). The low-frequency look of the extrapolated output can be understood by comparing the phase spectra of the exact and approximate extrapolators. Figure (13) shows clearly that, at low-frequencies (0-20 Hz), the exact and approximate extrapolators (Figures 13a and 13b) are the same. Beyond about 20 Hz the lateral detail in the phase spectrum of the exact extrapolator becomes constant in the approximate one. This is possibly due to the non-convergence of the approximate extrapolator at higher frequencies, where ξ in equation (6) exceeds unity.



(a)



(b)



(c)

Fig 11. NSPS extrapolation (bilinear transform – second order) of the wavefield of figure 2(b) using the velocity field of Figure 2(a). (a) One velocity per trace, (b) One velocity every 40 m. (c) One velocity every 160 m.

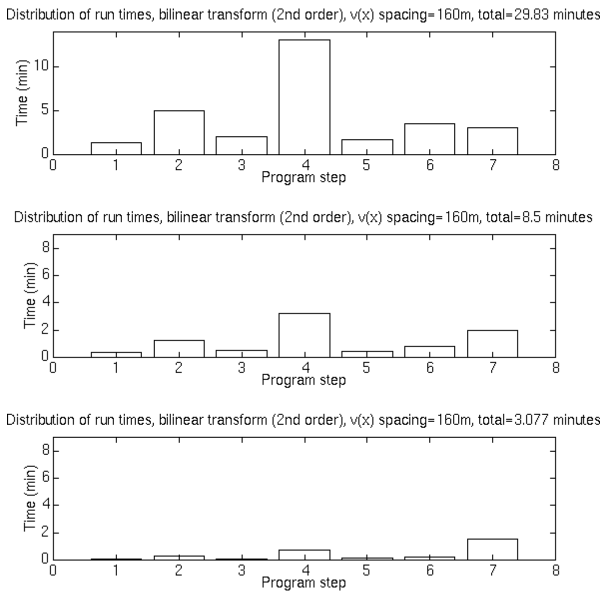
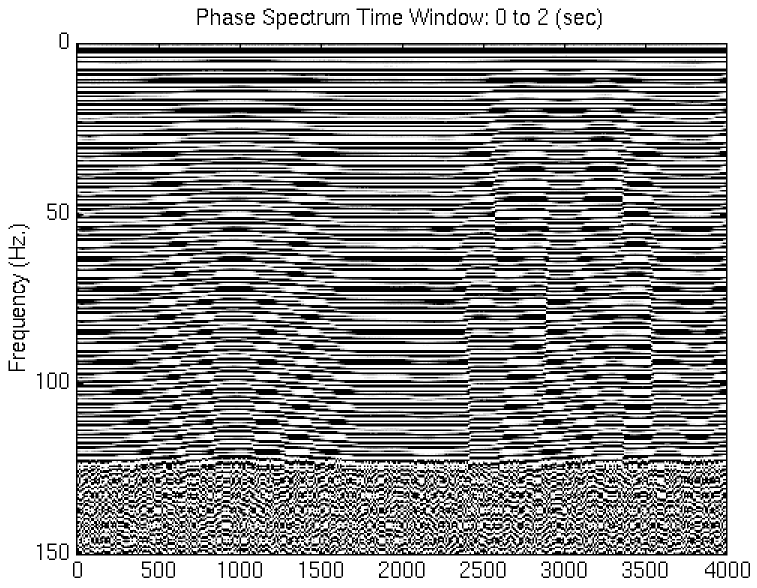
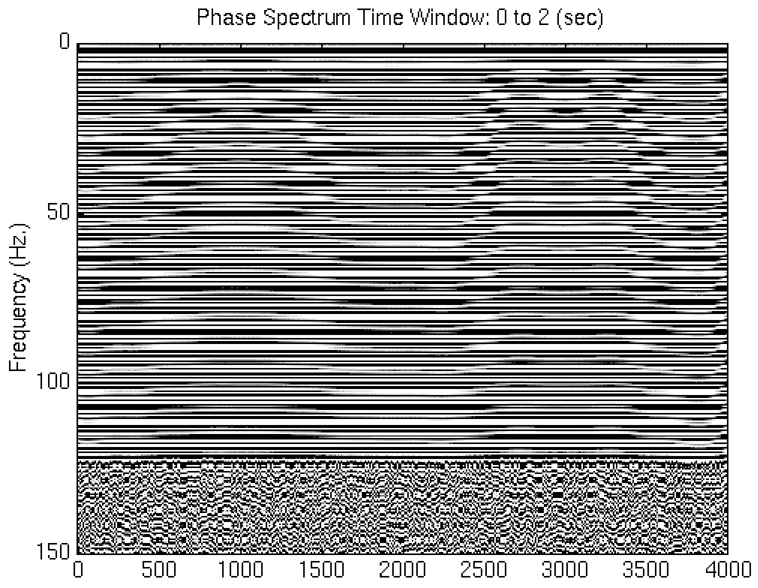


Fig 12. Runtime results for the bilinear transform. No runtime gains are apparent. See Figure (6) for a description of each step.



(a)



(b)

Fig 13. Comparison of the phase spectra of extrapolated wavefields, (a) by the exact extrapolator and (b) using the bilinear transform. Below about 20 Hz the phase spectra are the same. Above 20 Hz they become increasingly different, the bilinear transform spectrum becoming increasingly constant.

CONCLUSIONS

In this paper we have explored a number of issues relevant to the development of wavefield extrapolation by nonstationary phase shift (NSPS) as a first step toward a full migration/modeling algorithm. Because NSPS can be cast in three distinct domains, the space, dual and Fourier domains, we asked which of the three expressions for NSPS is the most computationally efficient. We discovered, by analysis, that NSPS in the dual domain is faster than in either the space or Fourier domains. We find also that NSPS in the space domain is faster than in the Fourier domain.

We then examined the effect of smoothing the input velocity field on computational effort. We find that, though smoothing reduces the computational effort in all domains, the space and dual domain expressions suffer from having to desample the input wavefield in space (i.e. dip limit) to realize these savings. The Fourier domain however, can benefit greatly from velocity smoothing without sacrificing geologic dip. For a typical 2-D experiment we find that significant runtime improvements, an almost 6:1 reduction, are obtainable when the velocity field is desampled from an interval of 10 m to 160 m. Depending on the desired smoothness of the velocity field, even greater runtime savings are available.

The effect of coding a particularly long running program step in the C language, and then externally compiling it resulted in no run time advantages, though we had previously found small improvements when using very small data sets, i.e. 128 traces each having 128 time samples.

When we approximate other long running processes, particularly the square root and exponent, using series; the 45-degree solution for the square root and the bilinear transform for the exponent, we find that both of these approximations returned disappointing runtimes. We find that the square root and exponent approximations were not faster than the exact method solution and, particularly for the bilinear transform, they were prone to error. Errors for the square root approximation are manifest in the need to limit the geologic dip of the data. Errors for the bilinear transform are evident in incorrect shifting of the higher frequencies (>20 Hz) possibly due to the non-convergence of the operator.

REFERENCES

Berkhout, A, J., 1985, Seismic Migration: Imaging of acoustic energy by wavefield extrapolation: A., Theory, in Developments in Solid Earth Geophysics 14A, Elsevier.

Black, J. L., Su, C. B., Wason, C. B., 1984, Steep-Dip depth migration: Expanded abstracts, **54**th Annual Mtg. Soc. Expl. Geophy., 456-457.

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

Margrave, G. F., and Ferguson, R. J., 1997, Wavefield extrapolation by nonstationary phase shift: : Expanded abstracts, **67**th Annual Mtg. Soc. Expl. Geophy., 1599-1602.

Holberg, O., 1988, Towards optimum one-way wave propagation: Geophys. Prosp., **36**, 99-114.

Margrave, G. F., 1997, Theory of nonstationary linear filtering in the Fourier domain with application to time variant filtering: Geophysics, *in press*.

Stoffa, P. L., Fokkema, J. T., de Luna Freire, R. M., and Kessinger, W. P., 1990, Split-step Fourier migration: Geophysics, **55**, 410-421.

Wapenaar, C. P. A., 1992, Wave equation based seismic processing: In which domain?: EAGE **54**th Conference Extended Abstracts, B019.