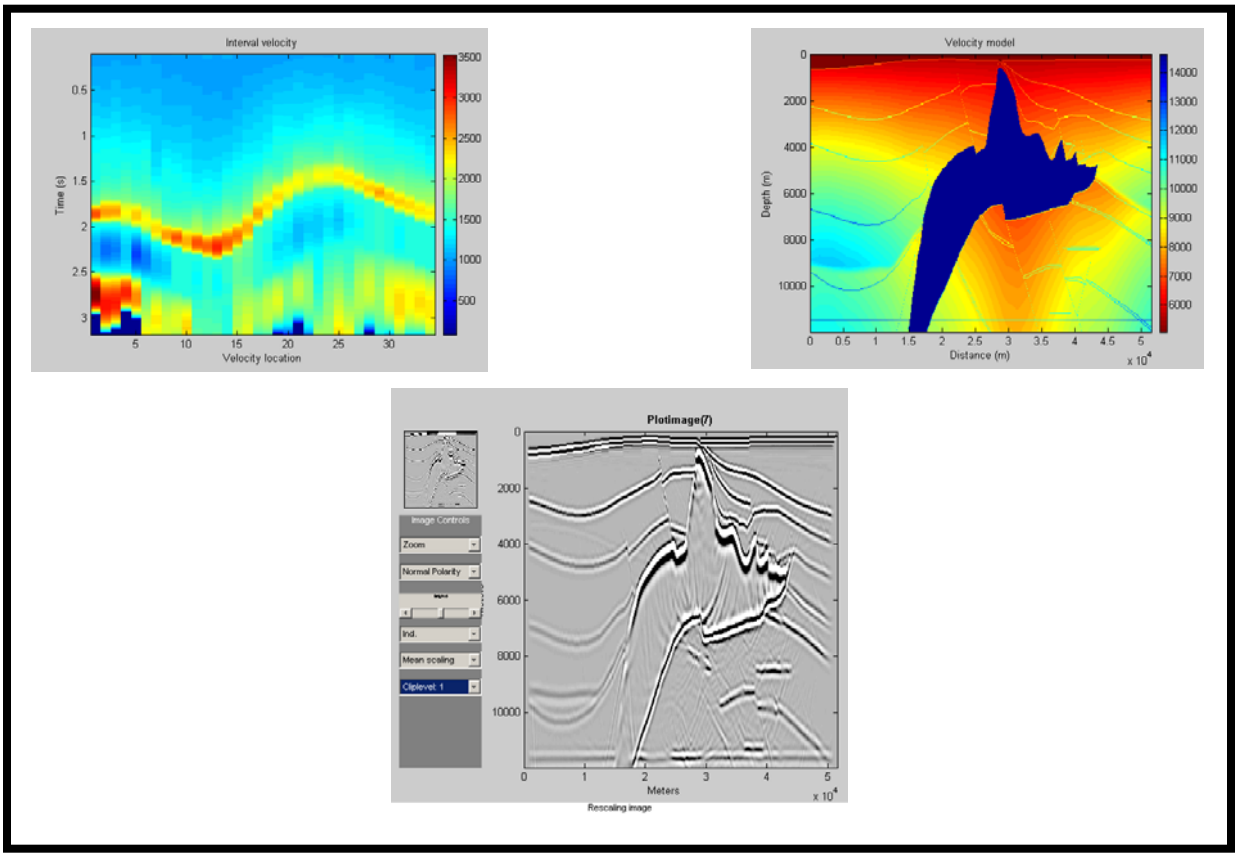
RESERVOIR GEOPHYSICS GRADUATE PROGRAM – UNIVERSITY OF INDONESIA
A report submitted in partial fulfillment of the requirements for the class of Computational Geophysics

Velocity Interpolation, Raytracing in V(z) Medium, AVO/AVA Inversion, Acoustic Wave Finite Difference Modeling and Migration



by

Befriko S. Murdianto
6304210026

Table of Contents

Introduction

Seismology often deals with very small deformations (infinitesimal strain theory); this assumption simplifies the mathematical treatment of many seismological problems. Seismology is also a complex technology that blends advanced physics, mathematics, and computation. For this purpose, a bunch of approximations and assumptions are derived to simplify the problem. As we will see within the subjects covered in this report, approximations and assumptions has been very much employed for the methodology to be computationally efficient and applicable. For example, the v(z) approximation used in raytracing problem. This is not far from the appearance of seismic sections from many of the world's sedimentary basins. Such basins are good v(z) environments and the normal rays for flat events are all vertical. However, when the same method was taken into thrust belts, continental margins, and other areas where the v(z) approximation is not a good model, it was gradually realized that this method may require another approximation that respect v(x,z) environments as well. For this reason, this report should be considered as a first step to understand the real geology, which is very complex and how to implement it into computational settings with some approximations and assumptions that can help in our understanding of the real problem.

There is a fundamental distinction between physical velocities and velocity measures. The former refers to velocities that are actually the speed at which some physical wave propagates. On the other hand, velocity measures are typically quantities derived from data analysis that have the physical dimensions of velocity but are related to physical velocities in some direct (and possibly unknown) fashion. In contrast to physical velocities, it cannot generally be expected that a physical wave actually propagates at the speed of one of these velocity measures.

Using the measured data for the analysis of velocity and the application to the data of corrections that depend upon velocity are fundamental to seismic processing. Velocity, together with the geometry of the seismic acquisition and the shapes of the reflectors, causes the characteristic traveltime shapes (such as hyperbolae) in the data. Spatial variations in velocity cause distortions from the simple, canonical shapes, and these distortions must be accounted for in processing. Sometimes velocity information can be obtained from supporting data such as well logs, core measurements, and geologic maps but this information is inherently sparse with respect to the coverage of the seismic data itself. Ultimately, the only way to obtain the spatially dense velocity information required in processing is from the data itself. This is known as velocity analysis and is the source of some of the velocity measures such that they can be converted into measurements of physical velocities is called velocity inversion and is an ongoing problem. A careful discussion of the definitions of these measures is essential to gain understanding of the complexities of velocity inversion.

For simplicity, consider the case of P-wave propagation in a heterogeneous earth (S-waves can be treated similarly). A velocity measure will be defined by either relating it mathematically to the actual P-wave speed or by describing how it can be derived from seismic data.

Instantaneous velocity

Instantaneous velocity generally refers to the speed of propagation of seismic waves in the earth. The word instantaneous refers to the local wave speed as the wave evolves from one instant of time to the next. The term can be applied to any wave type (P, S, surface, etc.) though here we specialize to P-wave for clarity.

Like most seismic velocities, v_{ins} is not usually a vector quantity and so is not a velocity as physicist would use the term. Rather it is a scalar, which can be thought of as the magnitude of a velocity vector.

For practical seismic experiments, v_{ins} must be acknowledged to be a highly variable function of position in the earth. In the most general case of anisotropic media, it also depends upon direction, but this will be ignored for now. It is not a function of time as that would imply that the earth's properties changed during the seismic experiment. Only rarely can it be assumed to be spatially constant but often there is significant variation in one direction only. This is commonly the case in sedimentary basins where velocity varies strongly with depth but only weakly in the horizontal direction.

Consider the instantaneous velocity function that is linear with depth z

$$v_{ins} = v_o + cz\ . \tag{1}$$

The universal velocity function is the name given to this function when v_o=1800 m/sec and c=0.6 sec^{-1}.

Average velocity

The typical industry definition of v_{ave} is that it is a particular depth divided by the vertical traveltime from the surface to that depth. Given any $z(\tau)$ curve, pick a point (τ_o,z_o) and v_{ave} is the slope of the line connecting the origin point (τ_o,z_o) while v_{ins} is the tangent to the curve at that point. Mathematically, $v_{ave}(z)$ is expressed as

$$v_{ave}(z) = \frac{z}{\tau(z)} = \frac{z}{\int_0^z \frac{dz}{v_{ins}(z)}}, \tag{2}$$

while $v_{ave}(\tau)$ is

$$v_{ave}(\tau) = \frac{z(\tau)}{\tau} = \frac{1}{\tau}\int_0^\tau v_{ins}(\tau)d\tau\ . \tag{3}$$

Equation (3) shows that the average velocity is a mathematical average only with respect to vertical traveltime. When considered as a function of depth, the average velocity is not a mathematical average.

For the running example of linear variation of velocity with depth, $v_{ave}(z)$ becomes

$$v_{ave}(z) = \frac{cz}{\ln\left[1+\frac{cz}{v_o}\right]}, \tag{4}$$

and $v_{ave}(\tau)$ is given by

$$v_{ave}(\tau) = \frac{v_o}{c\tau}\left[e^{c\tau} - 1\right]. \tag{5}$$

Mean velocity

Since v_{ave} is a mathematical average of v_{ins} over traveltime and not depth, it is useful to define another velocity measure that is a depth average. The mean velocity v_{mean} is defined as

$$v_{mean}(z) = \frac{1}{z}\int_0^z v_{ins}(z)dz\ . \tag{6}$$

For the linear variation of $v_{ins}(z)$, $v_{mean}(z)$ becomes

$$v_{mean}(z) = \frac{1}{z}\int_z^0 \left[v_o + cz\right]dz = v_o + \frac{1}{2}cz\ , \tag{7}$$

which shows that the mean velocity increases at half the rate of $v_{ins}(z)$. Of course $v_{mean}(z)$ can be re-expressed as a function of τ by substituting $z(\tau)$, this result in

$$v_{mean}(\tau) = \frac{v_o}{2}\left[1 + e^{c\tau}\right]. \tag{8}$$

RMS velocity

The mean velocity, discussed previously, is not commonly used in seismology. The reason is that another velocity measure, v_{rms} is used and only two of v_{ave}, v_{mean}, and v_{rms} are independent. The root-mean-square velocity is defined as

$$v_{rms}^2(\tau) = \frac{1}{\tau}\int_0^{\tau} v_{ins}^2(\tau)d\tau\,. \tag{9}$$

Two important relationships exist between the three velocity measures: v_{ave}, v_{mean}, and v_{rms}. First, they are linked by the relation

$$v_{rms}^2 = v_{ave}v_{mean}\,, \tag{10}$$

which means that only two of the three velocity measures can be considered independent.

The second relationship follows from a mathematical inequality known as Schwartz's Inequality. This result is quite general and confirms the intuition that the square root of the average of the squares of a set of numbers is always greater than the direct average of the numbers themselves.

Interval velocity

Corresponding to any of the velocity averages, an interval velocity can be defined that is simply the particular average applied across a small interval rather than from the surface (z=0) to depth z. For example, the average and rms velocities across an interval defined by τ_1 and τ_2 are simply

$$v_{ave}(\tau_2,\tau_1) = \frac{1}{\tau_2-\tau_1}\int_{\tau_1}^{\tau_2} v_{ins}(\tau)d\tau\,, \tag{11}$$

and

$$v_{rms}^2(\tau_2,\tau_1) = \frac{1}{\tau_2-\tau_1}\int_{\tau_1}^{\tau_2} v_{ins}^2(\tau)d\tau\,. \tag{12}$$

These quantities are function of both the upper and lower bounds of the integrals. In the limit, as the interval shrinks so small that $v_{ins}(\tau)$ can be considered constant, both interval velocity approach the instantaneous velocity.

It follows directly from the definition of average velocity (equation (2)) that the average velocity across a depth interval is just the ratio of the depth interval to the vertical traveltime across the interval. That is

$$v_{ave}(\tau_2,\tau_1) = \frac{z_2-z_1}{\tau_2-\tau_1} = \frac{\Delta z}{\Delta\tau}\,. \tag{13}$$

Thus, if the range from 0 to z is divided into n finite intervals defined by z_0, z_1, z_2,…,z_{n-1},z_n (where z_0=0 and z_n=z) then

$$v_{ave}(\tau) = \frac{z(\tau)}{\tau} = \frac{\sum_{k=1}^{n}\left[z_k - z_{k-1}\right]}{\tau} = \frac{1}{\tau}\sum_{k=1}^{n}\left[\tau_k - \tau_{k-1}\right]v_{ave}(\tau_k,\tau_{k-1})\,, \tag{14}$$

where $\tau_k=\tau(z_k)$. Defining $\Delta\tau_k=\tau_k-\tau_{k-1}$ then gives

$$v_{ave}(\tau) = \frac{1}{\tau}\sum_{k=1}^{n} v_k \Delta\tau_k \ , \tag{15}$$

where $v_k \equiv v_{ave}(\tau_k, \tau_{k-1})$ and $\tau = \sum_{k=1}^{n} \Delta\tau_k$. Comparing equation (15) with equation (3) suggest that the former is just a discrete version of the latter. However, the velocity v_k in equation (15) is the average velocity of the k^{th} finite interval and is thus the time average of v_{ins} across the interval. Of course, if $v_{ins}(\tau)$ is constant across each interval then v_k is an instantaneous velocity and this distinction vanishes.

Equation (15) can be used in a number of ways. Most obviously, it shows how to combine a set of local average velocities to obtain the macro average velocity across a larger interval. Also, it can be used to estimate a local average velocity given two macro-average velocities. Suppose the average velocities v_{ave1} and v_{ave2} from z=0 to depths z_1 and z_2 are known. Then an expression for the average velocity, from $z_1 \rightarrow z_2$ or equivalently from $\tau_1 \rightarrow \tau_2$, follows from equation (15) and is

$$v_{ave}(\tau_2, \tau_1) = \frac{1}{\tau_2 - \tau_1}\left[\tau_2 v_{ave2} - \tau_1 v_{ave1}\right]. \tag{16}$$

The two macro averages are each weighted by their time intervals and then the shallower average is subtracted from the deeper. This difference is then divided by the time interval of the local average.

If v_{ave1}, τ_1, v_{ave2}, and τ_2 are all measured without error, then this process (i.e. equation (16)) works perfectly. However, in a real case, errors in the measurements can lead to wildly incorrect estimates of $v_{ave}(\tau_2,\tau_1)$. With noisy data, there is no guarantee that the term $[\tau_2 v_{ave2} - \tau_1 v_{ave1}]$ will always be positive leading to the possibility of negative velocity estimates. Also, the division by $\tau_2-\tau_1$ can be unstable if the two times are very close together.

The discussion so far has been only for average velocities across an interval. The entire derivation above can be repeated for rms velocity with similar results but a few more subtleties arise in interpretation. Rather than repeating the derivation just given with 'rms' in place of 'ave', it is instructive to consider an alternative approach. It is a property of integrals that $\int_a^c = \int_a^b + \int_b^c$ where a<b<c. Given $\tau_0<\tau_1<\tau_2$ and applying this rule to equation (9) results in

$$v_{rms}^2(\tau_2, \tau_0) = \frac{1}{\tau_2 - \tau_0}\left[\int_{\tau_0}^{\tau_1} v_{ins}^2(\tau)d\tau + \int_{\tau_1}^{\tau_2} v_{ins}^2(\tau)d\tau\right]. \tag{17}$$

Recognizing the integral in […] as rms interval velocities squared multiplied by their interval times leads to

$$v_{rms}^2(\tau_2, \tau_0) = \frac{1}{\tau_2 - \tau_0}\left[(\tau_1 - \tau_0)v_{rms}^2(\tau_1, \tau_0) + (\tau_2 - \tau_1)v_{rms}^2(\tau_2, \tau_1)\right]. \tag{18}$$

For the case of n subdivisions between τ_2 and τ_0 this generalizes to

$$v_{rms}^2(\tau_2, \tau_0) = \frac{1}{\tau_2 - \tau_0}\sum_{k=1}^{n} v_k^2 \Delta\tau_k \ , \tag{19}$$

where $v_k=v_{rms}(\tau_k,\tau_{k-1})$ and, as before, $\Delta\tau_k=\tau_k-\tau_{k-1}$ and $\tau=\sum_{k=1}^{n}\Delta\tau_k$. This is the rms equivalent to equation (15) and all of the comments made previously about v_{ave} apply here for v_{rms}. In particular, equation (19) should be thought of a combining interval rms velocities into a macro rms velocity. Only in the case when v_{ins} does not vary significantly across an interval the v_k in equation (19) be considered to be instantaneous velocities.

Equation (18) is the addition rule for rms velocities. To combine rms velocities, the squared velocities are added and each must be weighted by its time interval. Equation (18) can be rearranged to give an expression for estimating a local rms velocity from two macro velocities

$$v_{rms}^2(\tau_2,\tau_1)=\frac{1}{\tau_2-\tau_1}\left[(\tau_2-\tau_0)v_{rms}^2(\tau_2,\tau_0)-(\tau_1-\tau_0)v_{rms}^2(\tau_1,\tau_0)\right], \tag{20}$$

or, with simplified notation

$$v_{rms}^2(\tau_2,\tau_1)=\frac{1}{\tau_2-\tau_1}\left[\tau_2 v_{rms2}^2-\tau_1 v_{rms1}^2\right]. \tag{21}$$

In this expression, τ_0 has been set to 0 and $v_{rms2}=v_{rms}(\tau_2,\tau_0)$ and similarly for v_{rms1}. Equation (21) is often called the Dix equation for interval rms velocities because it was C. H. Dix (Dix, 1955) who first recognized the connection between stacking velocities and rms velocities and showed how to calculate an interval velocity from two stacking velocity measurements.

The application of equation (21) in practice requires the measurement of stacking velocities and vertical traveltimes to two closely spaced reflectors. Under the assumption that stacking velocities are well approximated by rms velocities (Dix, 1955 or Taner and Koehler, 1969), the rms velocity of the interval is then estimated with equation (21). However, as with average velocities, errors in measurement lead to problems with the estimation of $v_{rms}(\tau_2,\tau_1)$. If the term $\left[\tau_2 v_{rms2}^2-\tau_1 v_{rms1}^2\right]$ becomes negative then imaginary interval velocity estimates result. Thus one essential condition for the use of this technique is that

$$v_{rms2}^2>v_{rms1}^2\frac{\tau_1}{\tau_2}. \tag{22}$$

Since $\tau_1/\tau_2<1$, this is a constraint upon how fast v_{rms} estimates can decrease with increasing time. There is no mathematical basis to constrain the rate at which v_{rms} can increase, however, it is reasonable to formulate a constraint on physical grounds. Since P-wave seismic velocities are not expected to exceed some v_{max} (say 7000 m/s), a constraint would be

$$v_{rms2}^2<v_{rms1}^2\frac{\tau_1}{\tau_2}+v_{\max}^2\frac{\tau_2-\tau_1}{\tau_2}. \tag{23}$$

Velocity interpolation implemented in Matlab

The following m-file script will interpolate a set of picked velocities using shape-preserving cubic interpolation. Picked velocities are saved to an m by n matrix as a t_{nmo} and v_{nmo} pairs, with m is the number of picked velocity functions on one location and n is the number of velocity locations. The interpolation is accomplished using Matlab's built-in function interp1 with shape-preserving cubic interpolation, rather than linear interpolation in order to get a smooth function between the picks. The interpolated velocities are then converted to interval velocity using vrms2vint function with the assumption that $v_{nmo}=v_{rms}$ which is valid for flat, shallow reflectors and for small offsets. For interval velocity, sometimes it is easier for us to

view it as a “block constant” within a certain interval. To achieve this, the interval velocities from the interpolated functions are then converted back to rms velocities but using a block constant of 10 equally-spaced time intervals. These rms velocities are then converted back to interval velocities to get the block constant v_{int}.

```
% This script interpolates a given tnmo and vnmo. Uses shape-preserving cubic
% interpolation, rather than linear interpolation for a smooth function

clear all

% load tnmo and vnmo
velpick;

[m,n] = size(tnmo);
tmax = max(max(tnmo));
tmin = min(min(tnmo));

% interpolated time vector
dt = 0.002;
t_interp = [tmin:dt:tmax]';

% copy interpolated time for other velocity locations
t_ones = ones(1,n);
t_interp = t_interp*t_ones;

% cubic interpolation
for i = 1:n
    v_interp(:,i) = interp1(tnmo(:,i),vnmo(:,i),t_interp(:,i),'pchip');
    % change to interval velocity
    vint(:,i) = vrms2vint(v_interp(:,i),t_interp(:,i));
end

figure;
subplot(2,2,1);plot(v_interp(:,1),t_interp(:,1),vnmo(:,1),tnmo(:,1),'ro');
axis ij;title('Velocity function for location 1');legend('interpolated','picked');
xlabel('Velocity (m/s)');ylabel('Time (s)');
subplot(2,2,2);plot(v_interp(:,17),t_interp(:,17),vnmo(:,17),tnmo(:,17),'ro');
axis ij;title('Velocity function for location 17');legend('interpolated','picked');
xlabel('Velocity (m/s)');ylabel('Time (s)');
subplot(2,2,3);plot(v_interp(:,18),t_interp(:,18),vnmo(:,18),tnmo(:,18),'ro');
axis ij;title('Velocity function for location 18');legend('interpolated','picked');
xlabel('Velocity (m/s)');ylabel('Time (s)');
subplot(2,2,4);plot(v_interp(:,34),t_interp(:,34),vnmo(:,34),tnmo(:,34),'ro');
axis ij;title('Velocity function for location 34');legend('interpolated','picked');
xlabel('Velocity (m/s)');ylabel('Time (s)');

% block constant vint
tblock=linspace(min(t_interp(:,1)),max(t_interp(:,1)),10);
for j = 1:n
    vrmsblock(:,j) = vint2vrms(vint(:,j),t_interp(:,j),tblock);
    vintblock(:,j) = vrms2vint(vrmsblock(:,j),tblock);
end

figure;
subplot(2,2,1);plot(v_interp(:,1),t_interp(:,1));drawvint(tblock,vintblock(:,1),'r')
;
axis ij;title('Velocity function for location 1');legend('rms','interval');
xlabel('Velocity (m/s)');ylabel('Time (s)');
subplot(2,2,2);plot(v_interp(:,17),t_interp(:,17));drawvint(tblock,vintblock(:,17),'
r');
axis ij;title('Velocity function for location 17');legend('rms','interval');
xlabel('Velocity (m/s)');ylabel('Time (s)');
```

```
subplot(2,2,3);plot(v_interp(:,18),t_interp(:,18));drawvint(tblock,vintblock(:,18),'
r');
axis ij;title('Velocity function for location 18');legend('rms','interval');
xlabel('Velocity (m/s)');ylabel('Time (s)');
subplot(2,2,4);plot(v_interp(:,34),t_interp(:,34));drawvint(tblock,vintblock(:,34),'
r');
axis ij;title('Velocity function for location 34');legend('rms','interval');
xlabel('Velocity (m/s)');ylabel('Time (s)');

% display interpolated vrms
figure;imagesc(1:n,t_interp(:,1),v_interp);colorbar;
xlabel('Velocity location');ylabel('Time (s)');
title('Interpolated rms velocity');

% display vint
figure;imagesc(1:n,t_interp(:,1),vint);colorbar;
xlabel('Velocity location');ylabel('Time (s)');
title('Interval velocity');
```

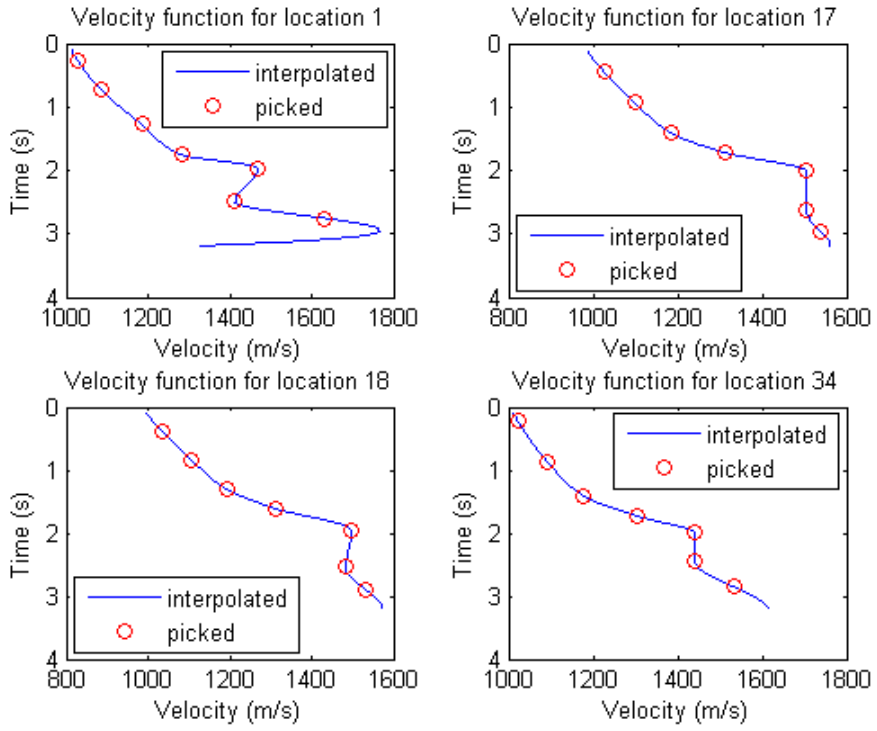


Figure 1. Velocity interpolation results from 4 velocity locations. Red circle is the picked velocity, while blue line is the interpolated velocity using shape-preserving cubic interpolation.

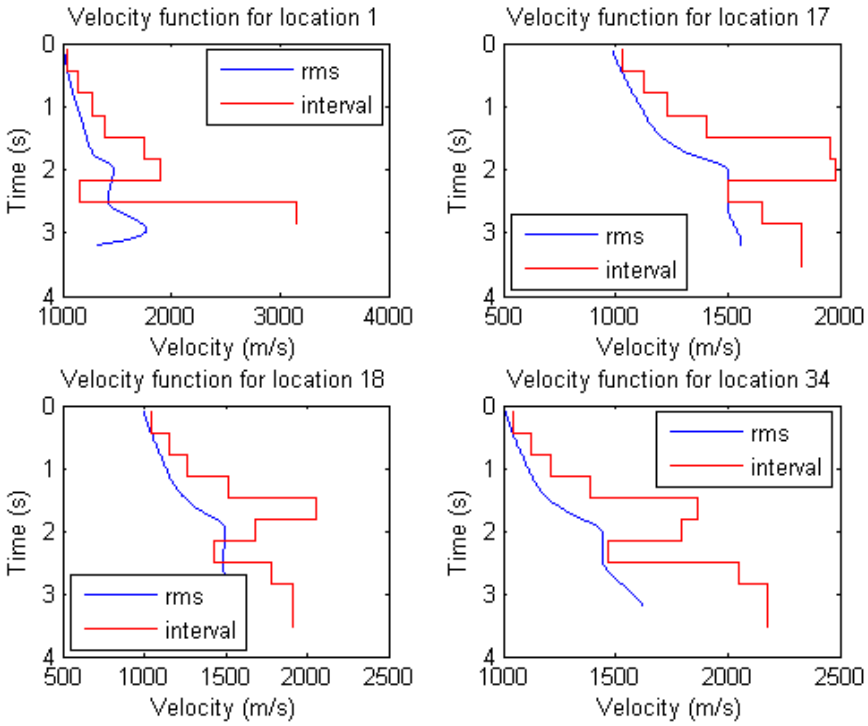


Figure 2. Comparison between interpolated v_{rms} (blue) and block constant v_{int} (red) on 4 velocity analysis locations.

Figure 1 shows the result of the velocity interpolation on 4 velocity analysis locations. Interpolation is only done in temporal direction, while keeping the spatial direction un-interpolated. Red circle is the picked velocity, while blue line is the interpolated velocity using shape-preserving cubic interpolation in the temporal direction. Figure 2 displays the comparison between the interpolated v_{rms} (blue line) and block constant v_{int} (red line). Essentially, this is just a de-sampled version of the finely sampled v_{int} that was calculated in the first for-loop. Figure 3 shows the interpolated v_{rms} for all 34 velocity analysis locations. Interpolation is done in the temporal direction, no interpolation is done in the spatial direction. Figure 4 shows the interpolated v_{int} for all 34 velocity analysis locations. Conversion from v_{rms} to v_{int} is done on the finely sampled v_{rms}, therefore the block constant appearance may not be too obvious.

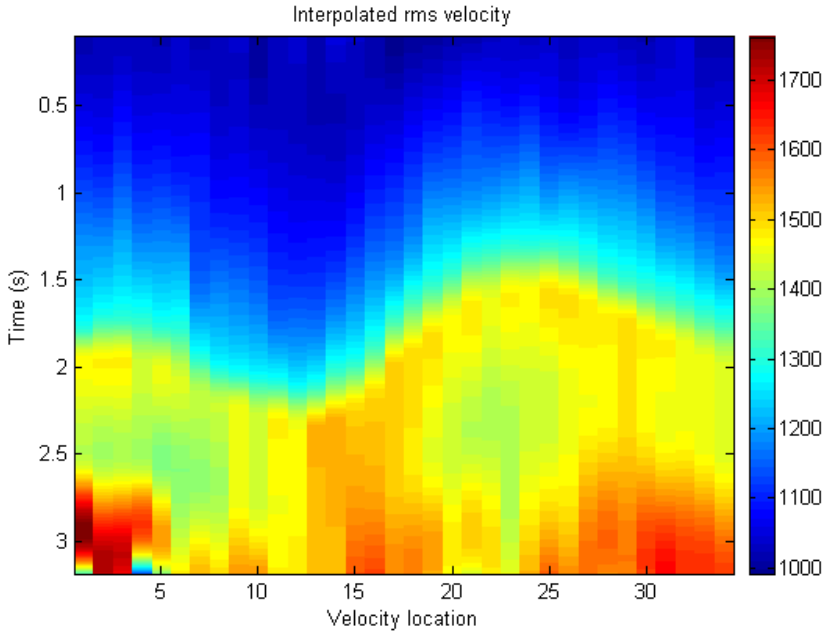


Figure 3. Interpolated v_{rms} for all velocity analysis locations.

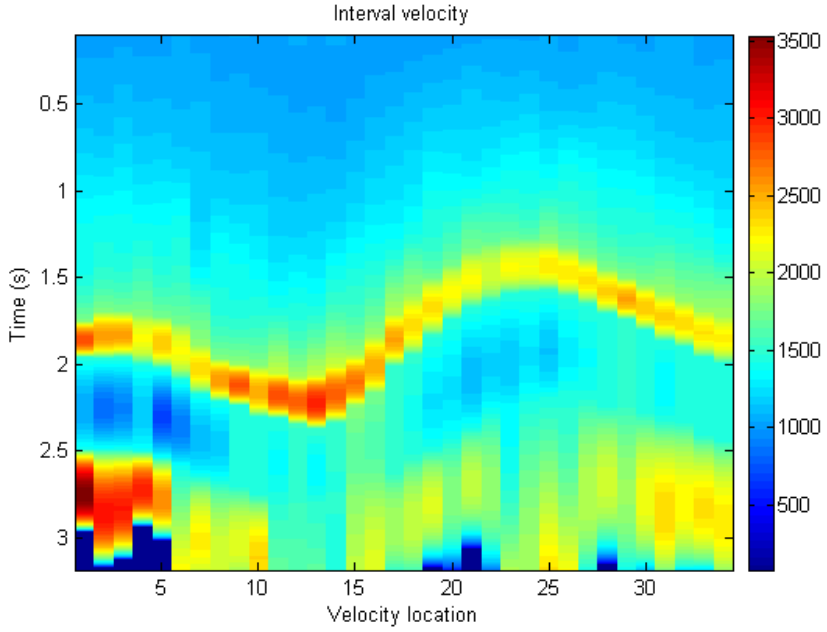


Figure 4. Interval velocity for all velocity analysis locations. Conversion from v_{rms} to v_{int} is done on the finely sampled v_{rms}.

Before dealing with v(z) raytracing, an understanding of the Snell's law is required. Snell's law is the relation that governs the angles of reflection and refraction of wavefronts (or equivalent raypaths) at velocity interfaces.

Snell's law and Fermat's principle

To understand its origins, consider Figure 5 that shows a periodic plane wave propagating across a planar interface between two media of velocity v_1 and v_2. In the first medium, the wavelength is $\lambda_1=v_1/f$ while in the second medium it is $\lambda_2=v_2/f$. As the wavefronts cross the interface, they must do so in a continuous fashion, otherwise, the wavefronts in the second medium would either lead or lag those in the first medium. Either possibility violates considerations of causality. The only way for the two wavefront systems to maintain continuity across the interface, and still have different wavelengths, is for them to have different angles with respect to the interface. The relationship between these angles follows from a consideration of apparent velocity. Suppose an array of receivers were placed along the interface. Then the requirement of wavefront continuity means that the wavelength component measured along the interface must be the same when evaluated in either medium. Since the frequency, f, does not change (this is a property of linear wave propagation), the apparent velocities measured along the interface must be the same in both media. Working in the rotated coordinates (x',z'), the apparent velocity v_x' must give the same result when evaluated using v_1 and θ as when using v_2 and ϕ. Thus

$$v_{1x'} \equiv \frac{v_1}{\sin\theta} = v_{2x'} \equiv \frac{v_2}{\sin\phi}. \tag{24}$$

This result is called Snell's law. The angles involved in Snell's law can be considered as the angles between the wavefronts and the interface or between the raypaths and the normal to the interface.

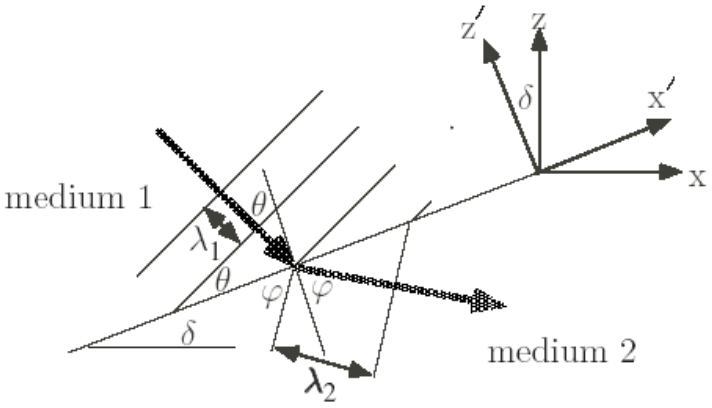


Figure 5. Snell's law results from the physical requirements that the apparent velocity along an interface be conserved.

Snell's law can be derived in other ways, perhaps the most common being to demonstrate that it is a consequence of requiring the raypaths correspond to stationary traveltimes. This is known as Fermat's principle. Physically, waves can propagate energy from point A to point B along infinitely many different paths. Fermat's principle says that the most important paths are those for which the traveltime is stationary with respect to slight variations in the path. Here, stationary most often means a minimum though there are important physical situations for which traveltime is maximized.

The derivation of Snell's law given here makes it clear that it applies to reflected as well as transmitted energy. In an acoustic medium, where there is only one mode of wave propagation, this results in the angle of incidence and the angle of refraction being equal. However, in elastic media, the incident and reflected waves can be either P-waves or S-waves. For example, in the case of an incident P-wave reflecting as an S-wave, Snell's law states

$$\frac{\sin\theta_p}{v_p} = \frac{\sin\theta_s}{v_s}. \tag{25}$$

As a final summary statement, Snell's law states that wavefront propagation across a velocity interface always conserves the apparent velocity along the interface. Though it is not proven here, this holds for arbitrary wave types and for non-planar wavefronts and interfaces.

Raytracing theory for a v(z) medium

A v(z) medium is one where $\partial_x v = \partial_y v = 0$ and all velocity interfaces are horizontal. In this case, Snell's law says that the horizontal apparent velocity of the wavefront associated with the ray is conserved. This may be stated, for j^{th} interface, as

$$\frac{\sin\theta_{j-1}}{v_{j-1}} = \frac{\sin\phi_j}{v_j}. \tag{26}$$

Since all of the velocity interfaces are horizontal, $\phi_j=\theta_j$ so that

$$\frac{\sin\theta_{j-1}}{v_{j-1}} = \frac{\sin\theta_j}{v_j}. \tag{27}$$

This analysis may be repeated at any other interface with similar conclusion. Thus the quantity $p=\sin\theta_j/v_j$ is conserved throughout the entire wave propagation. p is generally referred to as the ray parameter since it is a unique constant for any ray and so parameterizes (or names) the possible rays. The conservation of p and its identification as the horizontal apparent slowness is a direct consequence of Snell's law. This analysis generalizes to a continuous variation of v with z such that

$$p \equiv \frac{\sin(\theta(z))}{v(z)} = \text{ a constant for any particular ray.} \tag{28}$$

General expression for the traveltime and horizontal distance traveled by any ray in a v(z) medium can be easily derived. Figure 6 shows a differential element of ray. From the geometry, it follows that

$$dx = \tan(\theta(z))dz, \tag{29}$$

and

$$dt = \frac{ds}{v(z)} = \frac{dz}{v(z)\cos(\theta(z))}. \tag{30}$$

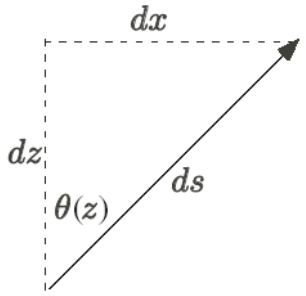


Figure 6. A differential ray element ds, at depth z, and traveling at an angle $\theta(z)$ with respect to the vertical.

Snell's law is incorporated by replacing the trigonometric functions using $pv(z)=\sin(\theta(z))$ so that

$$dx = \frac{pv(z)}{\sqrt{1-p^2v^2(z)}}\,dz\,, \tag{31}$$

and

$$dt = \frac{dz}{v(z)\sqrt{1-p^2v^2(z)}}\,. \tag{32}$$

Expression for macroscopic raypaths are obtained by simply integrating these results. For a ray traveling between depths z_1 and z_2, the horizontal distance traveled becomes

$$x(p) = \int_{z_1}^{z_2} \frac{pv(z)}{\sqrt{1-p^2v^2(z)}}\,dz\,, \tag{33}$$

and the total traveltime is

$$t(p) = \int_{z_1}^{z_2} \frac{dz}{v(z)\sqrt{1-p^2v^2(z)}}\,. \tag{34}$$

Given a velocity function and a ray parameter, we can compute exactly the horizontal distance (offset) and traveltime for a ray that traverses between two depths. The difficulty is that it is usually desired to trace a ray with a specific offset and there is no simple way to determine the ray parameter that will do this. Generally the process is iterative. The offsets produced by a fan of rays (i.e. p values) are examined and hopefully two p values will be found that bracket the desired offset. Then a new, refined fan of rays can be constructed and the process repeated until a ray is found that produces the desired offset within a specified tolerance (called the capture radius).

If the v(z) medium is discretely layered rather than continuously variable, then summation forms of equations (33) and (34) are more appropriate. These are easily seen to be

$$x(p) = \sum_{k=1}^{n} \frac{pv_k}{\sqrt{1-p^2v_k^2}}\,\Delta z_k\,, \tag{35}$$

and

$$t(p) = \sum_{k=1}^{n} \frac{\Delta z_k}{v_k\sqrt{1-p^2v_k^2}}\,. \tag{36}$$

Another way to find the ray parameter is by finding the root of equation, also known as the bisection method. This method will be discussed shortly in the Matlab implementation of v(z) raytracing.

Inversion of P-wave reflection coefficient for quantitative AVO/AVA analysis

In modern seismic exploration, acquisition and efficient utilization of high-quality three-dimensional seismic data becomes necessary to properly account for the complexity of the subsurface structure. Amplitude variation with offset (AVO) analysis is one of the seismic techniques that try to extract elastic properties of the subsurface from surface measurement of seismic data.

The beginning of the modern era of AVO analysis can be dated back to 1982 when, in his ground-breaking presentation, Ostrander (1982) demonstrated on real data how the variation of reflected amplitudes with offset can be used for direct detection of hydrocarbons. Since then, AVO has become an important tool of seismic exploration, complementary to seismic imaging. Primarily used as a purely qualitative method for revealing amplitude anomalies, AVO indeed proved to be useful not only for detection of hydrocarbon reservoirs (Ostrander, 1982; Chiburis, 1984; Hwang and Lellis, 1988; Burnett, 1990), but also for basic reservoir characterization (Smith and Gidlow, 1987; Ursin et al, 1996; Castagna et al, 1998) and, more recently, for reservoir monitoring. A remarkable development of AVO techniques over the past two decades, along with the development of acquisition techniques providing high-quality data, resulted in an increasing capability of AVO analysis. Today, the AVO methodology is more quantitative (Mallick, 1995; Cambois, 2000), and can be used to provide estimates of important reservoir characteristics, such as porosity, saturation, crack density, and orientation and contents of the cracks (Rüger, 1998).

Although the theoretical potential of quantitative AVO is promising, many practical and theoretical problems make successful implementation of AVO inversion difficult. One of the most troublesome theoretical problems is that conventional P-wave AVO inversion is poorly constrained (Jin and Beydoun, 1993; Cambois, 2000). As a consequence, individual parameters of the reservoir cannot be found uniquely and the recovered quantities may not be sufficiently accurate.

In AVO, the reflection coefficients at the target horizon (such as the top of a hydrocarbon reservoir) are the fundamental quantities to be analyzed as function of incidence angle. It is beyond the scope of this report to address many other practical and theoretical issues closely related to AVO analysis, such as amplitude-preserving data processing, correction of AVO amplitudes for noise, amplitude calibration, and propagation phenomena.

Reflection and transmission coefficients at a welded boundary

Reflection and transmission coefficients generally are complicated nonlinear functions of medium parameters. Thus, exact expressions of the coefficients usually do not provide useful analytic insight for AVO. Therefore, simplified approximations for the reflection and transmission coefficients are of great importance in practical AVO analysis (Bortfeld, 1961; Aki and Richards, 1980; Shuey, 1985).

The most general approximations for P-wave reflection coefficient for isotropic media started from the work of Knott (1899) and Zoeppritz (1919), which described the relationship between incident and reflection/transmission amplitudes for plane waves at a welded elastic interface

(Figure 7). The boundary conditions for these equations vary depending upon the elastic media on either side of the welded elastic interface. These boundary conditions for a solid-solid interface include continuity of tangential and normal displacement, as well as continuity of normal and tangential stress. The equations are fully described in matrix form by Aki and Richards (1980).

$$\begin{bmatrix} \sin\theta_1 & \cos\eta_1 & -\sin\theta_2 & \cos\eta_2 \\ -\cos\theta_1 & \sin\eta_1 & -\cos\theta_2 & -\sin\eta_2 \\ \sin 2\theta_1 & \frac{v_1}{w_1}\cos 2\eta_1 & -\frac{\rho_2 v_1 w_2^2}{\rho_1 v_2 w_1}\sin 2\theta_2 & \frac{\rho_2 v_1 w_2}{\rho_1 w_1^2}\cos 2\eta_2 \\ \cos 2\eta_1 & -\frac{w_1}{v_1}\sin 2\eta_1 & -\frac{\rho_2 v_2}{\rho_1 v_1}\cos 2\eta_2 & -\frac{\rho_2 v_2}{\rho_1 w_1}\sin 2\eta_2 \end{bmatrix} \begin{bmatrix} A_{PR} \\ A_{SR} \\ A_{PT} \\ A_{ST} \end{bmatrix} = \begin{bmatrix} -\sin\theta_1 \\ -\cos\theta_1 \\ \sin 2\theta_1 \\ -\cos 2\eta_1 \end{bmatrix}, \quad (37)$$

where A_{PR}, A_{SR}, A_{PT}, and A_{ST} are the P-wave reflection coefficient, S-wave reflection coefficient, P-wave transmission coefficient, and S-wave transmission coefficient, and v, w, and ρ are the P-wave velocity, S-wave velocity, and density, respectively.

For normal incidence condition (θ=0), reflection and transmission coefficient from equation (37) becomes

$$Ro = \frac{\rho_2 V_2 - \rho_1 V_1}{\rho_2 V_2 + \rho_1 V_1} \qquad To = \frac{2\rho_1 V_1}{\rho_2 V_2 + \rho_1 V_1}. \quad (38)$$

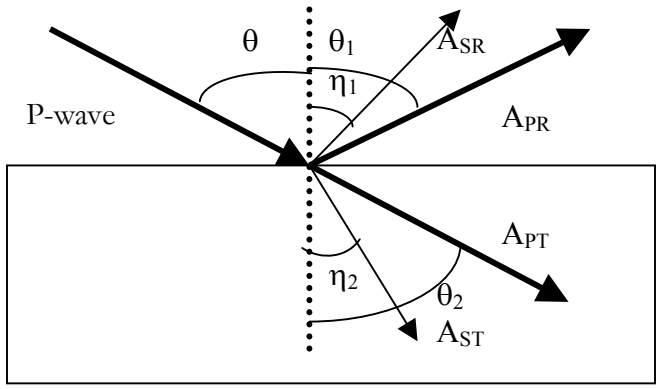


Figure 7. Reflection and transmission coefficients at a boundary.

Shuey (1985) further simplified these equations into a series of first order Zoeppritz equation approximations for plane-wave reflection and transmission coefficient at a welded interface, between two solid half-spaces. The linearized approximation for P-P reflection coefficient is

$$R(\theta) = Ro + \left[A_o R_o + \frac{\Delta\sigma}{(1-\sigma)^2} \right] \sin^2\theta + \frac{1}{2}\frac{\Delta Vp}{Vp}\left(\tan^2\theta - \sin^2\theta\right), \quad (39)$$

where Vp, σ, ρ are the average P-wave, Poisson's ratio and density values across an interface, ΔVp, $\Delta\sigma$, $\Delta\rho$ are the P-wave, Poisson's ratio and density contrast across an interface, θ is the average of the P-wave angle of incidence and transmission across the interface. Ro and Ao are defined by

$$Ro = \frac{1}{2}\left[\frac{\Delta Vp}{Vp} + \frac{\Delta\rho}{\rho} \right], \quad (40)$$

$$Ao = B - 2(1+B)\left(\frac{1-2\sigma}{1-\sigma}\right), \tag{41}$$

$$B = \frac{\dfrac{\Delta Vp}{Vp}}{\left(\dfrac{\Delta Vp}{Vp} + \dfrac{\Delta\rho}{\rho}\right)}, \tag{42}$$

$$Vp = \frac{Vp_2 + Vp_1}{2} \qquad \Delta Vp = Vp_2 - Vp_1, \tag{43}$$

$$\rho = \frac{\rho_2 + \rho_1}{2} \qquad \Delta\rho = \rho_2 - \rho_1, \tag{44}$$

$$\sigma = \frac{\sigma_2 + \sigma_1}{2} \qquad \Delta\sigma = \sigma_2 - \sigma_1. \tag{45}$$

Index 1 and 2 denotes upper and lower layer across the interface respectively.

Gardner et al (1974) derived a relationship between density and Vp using empirical observations

$$\rho` = 0.31 Vp^{0.25}. \tag{46}$$

By using equation (46), the dependency of density in equation (39) can be eliminated by deriving density to Vp

$$\frac{\Delta\rho}{\rho} = \frac{1}{4}\frac{\Delta Vp}{Vp}, \tag{47}$$

$$Ro = \frac{5}{8}\frac{\Delta Vp}{Vp}. \tag{48}$$

Hence, B in equation (42) becomes 4/5. Substituting this into (41) gives

$$Ao = \frac{4}{5} - \frac{18}{5}\left(\frac{1-2\sigma}{1-\sigma}\right). \tag{49}$$

Substituting equations (48) and (49) into (38) gives

$$R(\theta) = Ro + \left[\frac{4}{5} - \frac{18}{5}\left(\frac{1-2\sigma}{1-\sigma}\right) + \frac{\Delta\sigma}{(1-\sigma)^2}\right]\sin^2\theta + \frac{4}{5}Ro\left(\tan^2\theta - \sin^2\theta\right),$$

$$R(\theta) = Ro - \frac{18}{5}\left[\frac{1-2\sigma}{1-\sigma}\right]Ro\sin^2\theta + \frac{\Delta\sigma}{(1-\sigma)^2}\sin^2\theta + \frac{4}{5}Ro\tan^2\theta,$$

$$R(\theta) = Ro\left(1 + \frac{4}{5}\tan^2\theta\right) - \frac{18}{5}\left(\frac{1-2\sigma}{1-\sigma}\right)Ro\sin^2\theta + \frac{\Delta\sigma}{(1-\sigma)^2}\sin^2\theta. \tag{50}$$

where $\sigma = \dfrac{\left(\dfrac{Vp}{Vs}\right)^2 - 2}{2\left[\left(\dfrac{Vp}{Vs}\right)^2 - 1\right]}$.

Non-linear least squares

If we have a set of observation data represented by

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}, \tag{51}$$

and the model response is defined by

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{bmatrix}. \tag{52}$$

The model that we use is a function of *p* parameters which is the elements of vector $\vec{\theta}$

$$\vec{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \\ \theta_n \end{bmatrix}. \tag{53}$$

Also, let us assume that θ_j^0 is the initial prediction of θ_j parameters (j=1,2,…,p), and $\mathbf{f^0}$ is the initial model response. If model response **f** is a linear function of parameters, hence the perturbation of the model response around $\vec{\theta}^0$ can be expressed by using first order Taylor's series expansion

$$\mathbf{f} = \mathbf{f}^0 + \sum_{j=1}^{p} \left. \frac{\partial \mathbf{f}}{\partial \theta_j} \right|_{\vec{\theta}=\vec{\theta}^0} \left(\theta_j - \theta_j^0\right), \tag{54}$$

or, in matrix notation

$$\mathbf{f} = \mathbf{f}^0 + \mathbf{Z}\vec{\delta}, \tag{55}$$

with **Z** is the Jacobian matrix n x p obtained from the partial derivatives with the following elements

$$Z_{ij} = \frac{\partial f_i}{\partial \theta_j}, \tag{56}$$

and $\vec{\delta} = \vec{\theta} - \vec{\theta}^0$ is the vector of parameter changes with elements δ_j which describes the changes or perturbation in θ_j parameters

$$\delta_j = \theta_j - \theta_j^0, \tag{57}$$

with j=1,2,…,p.

If we define **e** as an error vector that describes the difference between model response **f** and observed data **y**

$$\mathbf{e} = \mathbf{y} - \mathbf{f} \,. \tag{58}$$

By combining equations (55) and (58), we get

$$\mathbf{e} = \mathbf{y} - \left(\mathbf{f}^0 + \mathbf{Z}\vec{\delta}\right), \tag{59}$$

or

$$\mathbf{y} - \mathbf{f}^0 = \mathbf{Z}\vec{\delta} + \mathbf{e} \,. \tag{60}$$

Vector $\mathbf{y}\text{-}\mathbf{f}^0$ is called discrepancy vector, defined as **g**, so that

$$\mathbf{g} = \mathbf{y} - \mathbf{f}^0 \,, \tag{61}$$

and

$$\mathbf{e} = \mathbf{g} - \mathbf{Z}\vec{\delta} \,. \tag{62}$$

Gauss – Newton method

The problem of geophysical inversion is not as simple as one's might think. The problem arises since the matrix **Z** usually is not a square matrix and doesn't have a full rank. In reality, most geophysical problems are over determined, that is, more observations than the number of model parameters. For such cases, one can use the Gauss – Newton method to find the solution.

$$\mathbf{S} = \mathbf{e}^T\mathbf{e} = (\mathbf{g} - \mathbf{Z}\vec{\delta})^T(\mathbf{g} - \mathbf{Z}\vec{\delta}) \,. \tag{63}$$

A condition for **S** to be minimum is

$$\begin{aligned} &\frac{\partial \mathbf{S}}{\partial \vec{\delta}} = 0 \\ &\frac{\partial}{\partial \vec{\delta}}(\vec{\delta}^T\mathbf{Z}^T\mathbf{Z}\vec{\delta} - \mathbf{g}^T\mathbf{Z}\vec{\delta} - \vec{\delta}^T\mathbf{Z}^T\mathbf{g} + \mathbf{g}^T\mathbf{g}) = 0 \end{aligned} \,. \tag{64}$$

The solution to this differential equation is

$$\mathbf{Z}^T\mathbf{Z}\vec{\delta} = \mathbf{Z}^T\mathbf{g} \,. \tag{65}$$

The above equation is called "normal equation". Hence, **x** can be found by

$$\vec{\delta} = (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T\mathbf{g} \,. \tag{66}$$

Marquardt – Levenberg method

To avoid singularities, we use an alternative solution as described in Lines and Treitel (1984) by using a constrained solution

$$\mathbf{S}(\vec{\delta}, \beta) = \mathbf{e}^T\mathbf{e} + (\vec{\delta}^T\vec{\delta} - \delta_0^2) \,, \tag{67}$$

where β is a Lagrange multiplier.

Differentiation of vector δ gives a modified normal equation

$$(\mathbf{Z}^T\mathbf{Z} + \beta\mathbf{I})\vec{\delta} = \mathbf{Z}^T\mathbf{g} \,, \tag{68}$$

such that

$$\vec{\delta} = (\mathbf{Z}^T\mathbf{Z} + \beta\mathbf{I})^{-1}\mathbf{Z}^T\mathbf{g} \,. \tag{69}$$

v(z) raytracing and AVA inversion implemented in Matlab

Raytracing in v(z) medium can be implemented by using bisection method which finds the root of equation of the ray parameter. In Matlab, this can be achieved using Matlab's built-in function fzero. fzero tries to find a zero of a function near a scalar. It first finds an interval containing the scalar where the function values of the interval endpoints differ in sign, then searches that interval for a zero. The following m-file script describes how to use this to find the ray parameter for a v(z) medium. The medium consists of 9 layers with elastic parameters described in Table 1. The calculated ray parameters are then used to calculate traveltime and incidence angles for each interface. These traveltimes are plotted on Figure 8, along with the raypaths within the medium (Figure 9). Both traveltime and incidence angle can be utilized to create a synthetic seismogram shown in Figure 10 by convolving the reflectivity with 25 Hz Ricker wavelet. Reflectivity computed based on Zoeppritz's work (1919) for solid – solid interface (equation (37)) and then plotted as a function of incidence angle (Figure 11). The medium on each side of a given interface is perfectly elastic.

```
% This script creates synthetic seismogram. Traveltimes are calculated
% using raytracing and amplitudes are calculated using Zoeppritz's
% equation. AVA inversion is run on synthetics to get elastic parameters
% (Poisson's ratio) using Shuey's (1985) approximation.

clear all;
% model parameters for raytracing : velocity, thickness, and offset
v = [3.000 3.000 2.000 3.000 3.500 3.300 4.000 3.400 3.600];
z = [0 .400 .600 .900 1.100 1.500 1.600 1.800 2.000];
h = diff(z);
D = [0.05:0.05:3];

% the following parameters are not required in raytracing, they're
% required to calculate reflectivity
rho = [2.5 2.5 1.9 2.5 2.1 2.5 2.2 1.5 2];
vs = 0.5*v;
pois = (vs.^2-0.5*v.^2)./(vs.^2-v.^2);

% number of traces and number of interfaces
ntrace = length(D);
nint = min([length(h) length(v)]);

% find ray parameters using MATLAB command : fzero
disp('Calculating ray parameters...');
for j = 1:ntrace
   offset = D(j);
   for i = 1:nint
      hh = h(1:i);
      vv = v(1:i);
      er = 0.0001;
      x  = min(1./vv) - er;
      p(i,j) = fzero('fun_ray',[-er x],[],offset,hh,vv);
      theta(i,j) = (asin(p(i,j)*vv(i)))*180/pi;

      % calculate traveltime based on ray parameters
      t(i,j) = 2*sum(hh./(vv.*sqrt(1-p(i,j)^2*vv.^2)));

   end
end
disp('Ray parameter calculation completed');

% plot traveltime
figure
```

```
hold on
for i = 1:nint
   plot(D,t(i,:)); axis ij;
   axis([min(D) max(D) 0 1.25*max(max(t))]);
   title('Traveltime plot');
   xlabel('Offset (km)');ylabel('Time (s)');
end
hold off

% source and receiver depths
zsrc = z(:,1); zrec = zsrc;
figure; flipy;
% trace the rays, iterate over each layer
disp('Tracing rays...');
for k = 1:length(z)
    zd = z(:,k);
    [t2,p2] = traceray_pp(v,z,zsrc,zrec,zd,D,10,-1,10,1,1,-gcf);
end
title(['v(z) medium, P-P mode zsrc=' num2str(zsrc) ' zrec=' num2str(zrec)])
line(D,zrec*ones(size(D)),'color','b','linestyle','none','marker','v')
line(0,zsrc,'color','r','linestyle','none','marker','*')
grid;xlabel('Offset (km)');ylabel('Depth (km)');
disp('Ray tracing completed');

% create zeros matrix for spike's location
tmax = max(max(t)); dt = 0.002;
ta = [0:dt:tmax];
spikes = zeros(length(ta),ntrace);

% add n+1th layer to calculate reflectivity
v(nint+1) = 3.8; rho(nint+1) = 2.5; vs(nint+1) = 0.5*v(nint+1);

% put spikes into location
disp('Creating synthetics...');
for i = 1:nint
    for j = 1:ntrace
        ir(i,j) = fix(t(i,j)/dt+0.1)+1;
        refl(i,j) = zoeppritz(rho(:,i),v(:,i),vs(:,i),...
            rho(:,i+1),v(:,i+1),vs(:,i+1),1,1,1,theta(i,j));
        refl(i,j) = real(refl(i,j));
        spikes(ir(i,j),j) = spikes(ir(i,j),j)+refl(i,j);
    end
end

% convolve spikes with 25Hz ricker wavelet
[w,tw] = ricker(0.002,25);
for i = 1:ntrace
    seis(:,i) = convz(spikes(:,i),w);
end
disp('Synthetics done');

% plot synthetics
figure;plotseis(seis,ta,D,1,2,1,1,'k');
title('Synthetic seismogram');set(gca,'xaxislocation','bottom');
xlabel('Offset (km)');ylabel('Time (s)');

% plot AVO response
xmin = min(min(theta)); xmax = max(max(theta));
ymin = min(min(refl)); ymax = max(max(refl));
figure;plot(theta(1,:),refl(1,:),theta(2,:),refl(2,:),'r',theta(3,:),...
    refl(3,:),'k',theta(4,:),refl(4,:),'g',theta(5,:),refl(5,:),'c',...
    theta(6,:),refl(6,:),'m',theta(7,:),refl(7,:),'y',theta(8,:),refl(8,:),'b:');
axis([xmin xmax ymin ymax]);
```

```
xlabel('Incidence angle (degree)');ylabel('Reflection coefficient');
title('AVA response');legend('Layer 1-2','Layer 2-3','Layer 3-4',...
    'Layer 4-5','Layer 5-6','Layer 6-7','Layer 7-8','Layer 8-9');
```

AVA inversion is applied here using Shuey's (1985) approximation to the exact expression of the particle displacement reflection and transmission coefficients (the relative displacement amplitudes). Since we are only using 2-term approximation, which is valid only for incidence angle below 30 degree, the maximum incidence angle used in the inversion is set to 25 degree. This is because we used the full exact expression to create the forward model. Inversion result (Poisson's ratio for each interface) is then plotted on Figure 12, along with the true model.

```
% run AVA inversion for each interface
% initial model : Poisson's ratio for upper and lower layer
% max. incidence angle is set to 25 deg. since we're using 2-term Shuey
% approximation
pois1 = 0.4; pois2 = 0.2; thetamax = 25;
disp('Starting AVA inversion using Shuey''s (1985) approximation');
disp(sprintf('Maximum incidence angle is set to %d degree',thetamax));
disp('============================================================');
% stopping criteria
stop = 1e-4;
for k = 2:2:nint
    ao = refl(k,1);
    fprintf('Running AVA inversion for layer %d and %d\n',k,k+1);
    fprintf('True parameters for these layers :\n');
    fprintf('ao = %0.3f\n',ao);
    fprintf('Poisson''s ratio for upper layer = %0.3f\n',pois(:,k));
    fprintf('Poisson''s ratio for lower layer = %0.3f\n',pois(:,k+1));
    itheta = find(theta(k,:)<=thetamax);
    thetan = theta(k,1:length(itheta)); % incidence angle
    Robs = refl(k,1:length(itheta))'; % observation data, from forward modeling
    poisson = [pois1 pois2];
    p = [ao poisson]'; thetan = thetan';
    Rcal = shuey(ao,poisson,thetan);
    a = deriv_avo(ao,poisson,thetan); % jacobian matrix
    beta = 1e-5; % damping factor, to prevent singular matrix
    delta = marquardt(Robs,Rcal,a,p,beta); % parameter increment
    p = p+delta; % update initial model
    poisr(k-1) = p(2); poisr(k-1+1) = p(3);
    error = sum((Robs-Rcal).^2); % sum of squared errors
    % iterative computation
    count=0;
    while stop<error % stop iteration if error < stopping criteria
        ao = p(1); poisson = [p(2) p(3)];
        Rcal = shuey(ao,poisson,thetan);
        a = deriv_avo(ao,poisson,thetan);
        delta = marquardt(Robs,Rcal,a,p,beta);
        p = p+delta;
        poisr(k-1) = p(2); poisr(k-1+1) = p(3);
        error = sum((Robs-Rcal).^2);
        count = count+1; % update no. of iteration
    end
    pois1 = p(2); pois2 = p(3);
    % display result
    fprintf('Inversion result for these layers :\n',k,k+1);
    disp(sprintf('ao = %0.3f',p(1)));
    disp(sprintf('Poisson''s ratio for upper layer = %0.3f',p(2)));
    disp(sprintf('Poisson''s ratio for lower layer = %0.3f',p(3)));
    disp(sprintf('Sum of squared errors = %0.5f',error));
    disp(sprintf('No. of iterations = %d\n',count));
```

```
end

% plot result
figure;drawvint(z(:,2:length(z)),poisr);drawvint(z(:,2:length(z)),pois(:,2:length(z)
),'r');flipy;
ymin = min(z(:,2:length(z))); ymax = max(z(:,2:length(z)));
xmin = 0.2; xmax = 0.5;
axis([xmin xmax ymin ymax]);
xlabel('Poisson''s ratio');ylabel('Depth (km)');
legend('Inverted','True','Location','NorthWest');
```

Function file depicting the numeric function used to calculate ray parameter

```
function f = fun_ray(x,D,h,v);
f = 2*sum(x*h.*v./sqrt(1-x^2*v.^2)) - D;
```

Function file to calculate linearized P-wave reflection coefficient using Shuey's (1985) approximation

```
function R=shuey(Ro,pois,theta);
% SHUEY: calculate reflectivity as a function of incidence angles using
% Shuey's (1985) approximation
%
% R = shuey(ao,pois,theta)
%
% Ro .. reflectivity at normal incidence
% pois ... 2x1 column vector of Poisson's ratio for the upper and lower
% layer
% theta ... column vector of incidence angles

% Convert angles from radians to degrees
theta=theta.*pi./180;

C=(-18/5*(1+.5*(pois(2)+pois(1))^2-3/2*(pois(2)+pois(1)))+...
    ((pois(2)-pois(1))/Ro))/((1-.5*(pois(2)+pois(1)))^2);
R=Ro.*(1+4/5.*(tan(theta)).^2)+Ro.*C.*(sin(theta)).^2;
```

Function file to calculate Jacobian matrix. Basically, this is obtained from the partial derivatives of R(θ).

```
function drdp=deriv_avo(ao,pois,theta)
% Create jacobian matrix for AVO inversion
drdp(:,1)=1+4/5.*tan(theta).^2+(18/5-9/5.*(pois(2)+pois(1)).^2+27/5.*pois(2)+...
    27/5.*pois(1)+(pois(2)-pois(1))./ao)./(1-1/2.*pois(2)-1/2.*pois(1)).^2.*...
    sin(theta).^2-1./ao.*(pois(2)-pois(1))./(1-1/2.*pois(2)-1/2.*pois(1)).^2.*...
    sin(theta).^2;
drdp(:,2)=ao.*(-18/5.*pois(2)-18/5.*pois(1)+27/5-1./ao)./(1-1/2.*pois(2)-1/2.*...
    pois(1)).^2.*sin(theta).^2+ao.*(-18/5-9/5.*(pois(2)+pois(1)).^2+27/5.*...
    pois(2)+27/5.*pois(1)+(pois(2)-pois(1))./ao)./(1-1/2.*pois(2)-1/2.*...
    pois(1)).^3.*sin(theta).^2;
drdp(:,3)=ao.*(18/5.*pois(2)-18/5.*pois(1)+27/5+1./ao)./(1-1/2.*pois(2)-1/2.*...
    pois(1)).^2.*sin(theta).^2+ao.*(-18/5-9/5.*(pois(2)+pois(1)).^2+27/5.*...
    pois(2)+27/5.*pois(1)+(pois(2)-pois(1))./ao)./(1-1/2.*pois(2)-1/2.*...
    pois(1)).^3.*sin(theta).^2;
```

Function file to run least squares inversion using Marquardt – Levenberg optimization method (Lines and Treitel, 1984) described in equation (69).

```
function [delta] = marquardt(mo,mc,a,p,beta)
% MARQUARDT : Least squares inversion using Marquardt-Levenberg
```

```
% optimization method, as described in "Lines, L. R. and Treitel, S.
% (1984): A review of least squares inversion and its application to
% geophysical problems, Geophys. Prosp., vol. 32, pp. 159-186."
%
% [delta] = marquardt(mo,mc,a,p,beta)
%
% mo ... column vector of original model parameters
% mc ... column vector of calculated model from initial parameters
% a ... jacobian matrix
% p ... column vector of initial parameters
% beta ... damping factor

% calculate the discrepancy matrix
g=mo-mc;

l=length(p);
i=eye(l); % identity matrix
at=a';
ata=at*a;

% parameter increment
delta=inv(ata+beta*i)*at*g;
```

Layer	Vp (m/s)	Density (gr/cc)	Vs (m/s)	Depth (m)
1	3000	2.5	1500	0
2	3000	2.5	1500	400
3	2000	1.9	1000	600
4	3000	2.5	1500	900
5	3500	2.1	1750	1100
6	3300	2.5	1650	1500
7	4000	2.2	2000	1600
8	3400	1.5	1700	1800
9	3600	2.0	1800	2000

Table 1. Elastic parameters for a 9-layer medium.

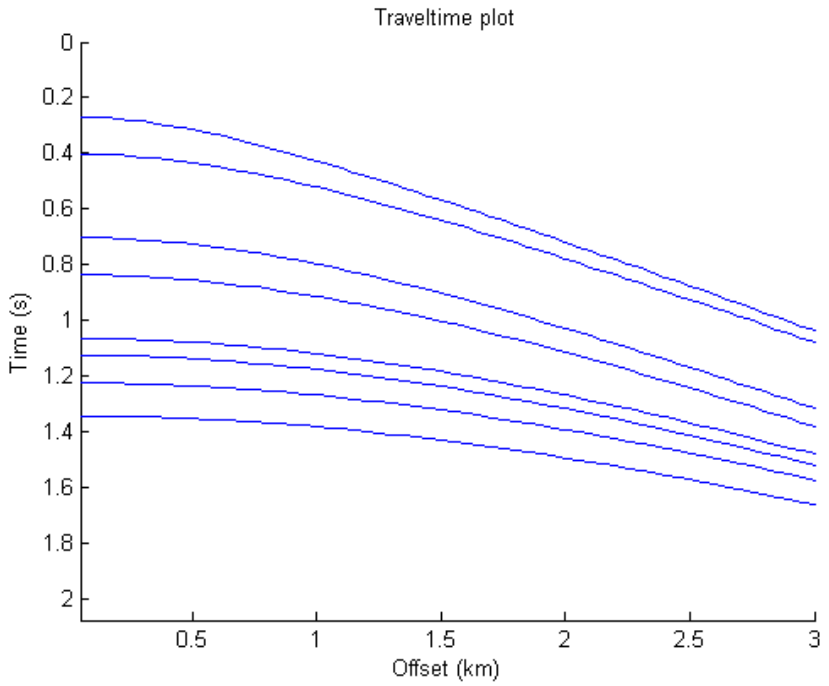


Figure 8. Traveltime plot calculated from the ray parameters obtained from ray tracing.

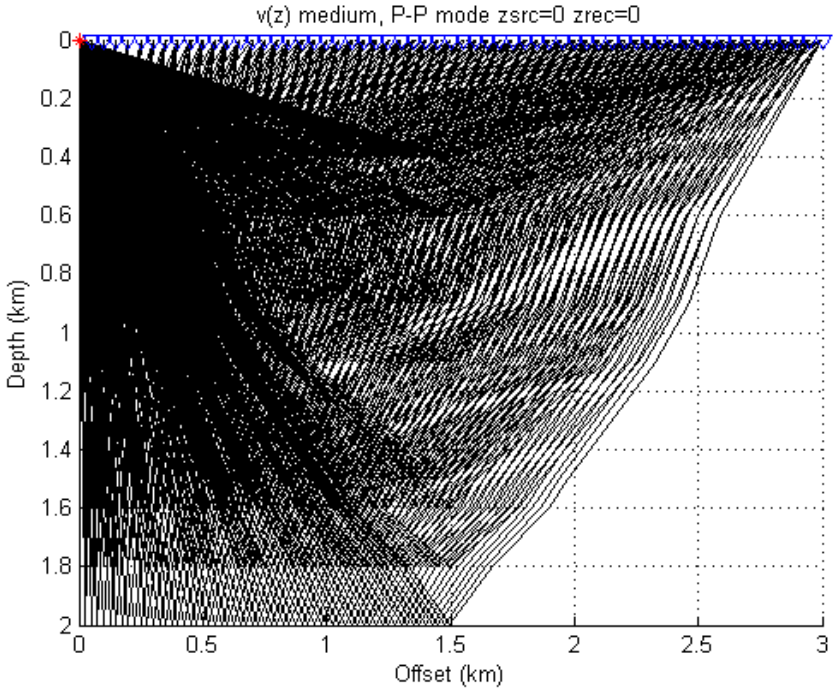


Figure 9. Raypaths within the medium for all interfaces. Red asterisk denotes shot location, blue triangles denote receiver locations.

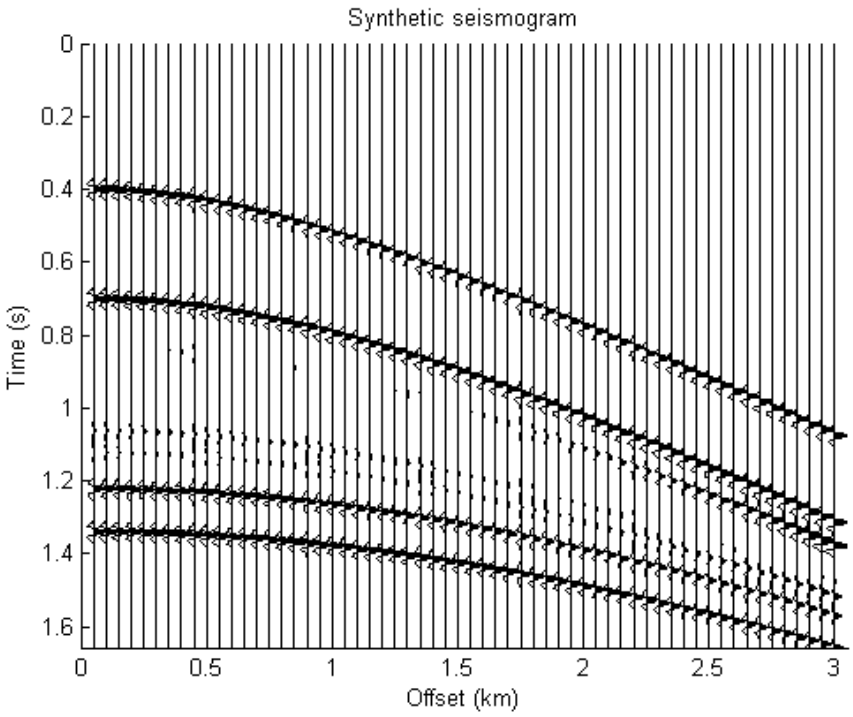


Figure 10. Synthetic seismogram created by convolving the reflectivity calculated from incidence angle with 25 Hz Ricker wavelet.

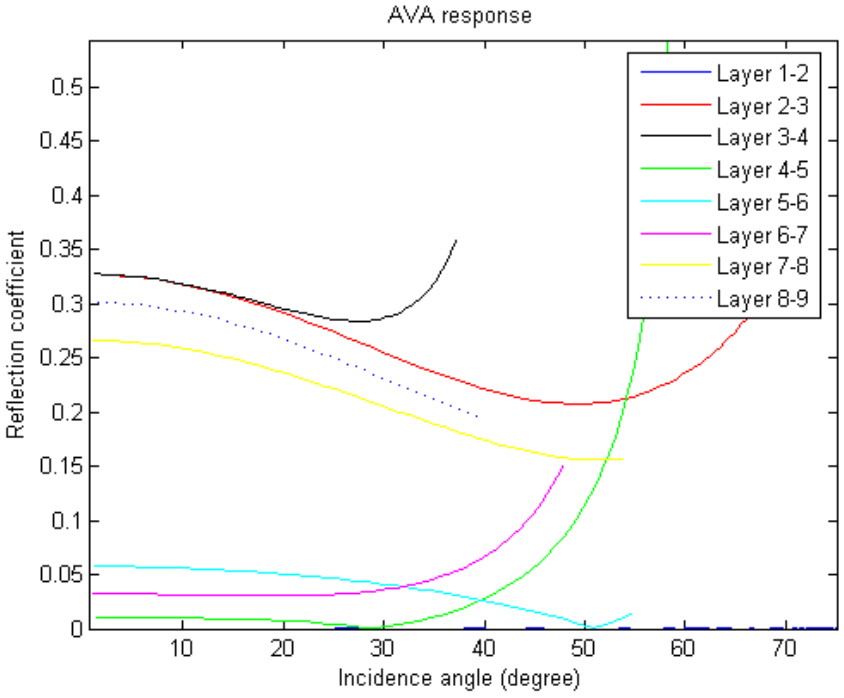


Figure 11. Reflectivity plotted as a function of incidence angle for each interface.

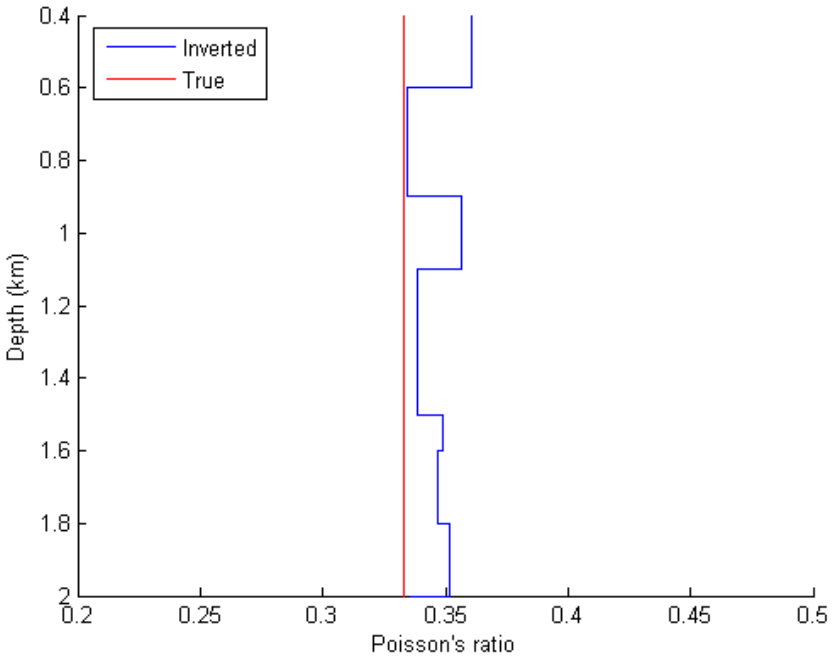


Figure 12. Poisson's ratio for true (red) and inverted (blue) models.

Acoustic wave modeling is an important tool to understand the behavior of acoustic wave in a medium. This can be achieved by using finite difference (FD) scheme to time step the wavefield and create snapshots of the wavefield, shot records, as well as exploding reflector models. FD can be implemented using second order and fourth order approximation to the scalar wave equation. Additionally, absorbing boundary can also be implemented to reduce the edge effect artifacts.

2D scalar wave equation

Consider the scalar wave equation in two dimensions

$$\nabla^2\psi(x,z,t) = \frac{1}{v^2(x,z)}\frac{\partial^2\psi(x,z,t)}{\partial t^2}, \tag{70}$$

where ∇^2 is the Laplacian operator, and is given by

$$\nabla^2\psi = \frac{\partial^2\psi}{\partial x^2} + \frac{\partial^2\psi}{\partial z^2}. \tag{71}$$

A second order approximation for the Laplacian operator can be implemented as (Margrave, 2001)

$$\begin{aligned}\nabla^2\psi(x,z,t) = &\frac{\psi(x+\Delta x,z,t) - 2\psi(x,z,t) + \psi(x-\Delta x,z,t)}{\Delta x^2} \\ &+ \frac{\psi(x,z+\Delta z,t) - 2\psi(x,z,t) + \psi(x,z-\Delta z,t)}{\Delta z^2}\end{aligned}, \tag{72}$$

while the fourth order approximation is

$$\begin{aligned}&\nabla^2\psi(x,z,t) = \\ &\frac{1}{12\Delta x^2}\left[-\psi(x+2\Delta x,z,t) + 16\psi(x+\Delta x,z,t) - 30\psi(x,z,t) + 16\psi(x-\Delta x,z,t) - \psi(x-2\Delta x,z,t)\right] \\ &+ \frac{1}{12\Delta z^2}\left[-\psi(x,z+2\Delta z,t) + 16\psi(x,z+\Delta z,t) - 30\psi(x,z,t) + 16\psi(x,z-\Delta z,t) - \psi(x,z-2\Delta z,t)\right]\end{aligned} \tag{73}$$

The use of these equations in time stepping a wavefield is known to be unstable, which can cause into the growing of amplitudes of the wavefield without bound as it is stepped through time. Lines et al (1999) showed that a stability condition to this problem could be achieved by applying

$$\frac{v_{\max}\Delta t}{\Delta x} \le \frac{1}{\sqrt{2}}, \tag{74}$$

for second order Laplacian, and

$$\frac{v_{\max}\Delta t}{\Delta x} \le \sqrt{\frac{3}{8}}, \tag{75}$$

for fourth order Laplacian.

Finite difference modeling with the acoustic wave equation

Generally, FD operators need many more samples than the Nyquist criterion of ten per wavelength. Technically, this is because the operators cause an artificial dispersion called grid dispersion. A good rule of thumb is about five to ten samples per wavelength. Typically, in the creation of a model, a desired temporal frequency range is known. Then, the minimum wavelength is given by $\lambda_{min}=v_{min}/f_{max}$ and the spatial sample rate can be chosen to achieve a desired number of samples-per-wavelength. Finally the temporal sample rate is chosen to achieve stability.

Acoustic wave finite difference implemented in Matlab

FD scheme is used to generate synthetic shot records in Matlab. The velocity model is taken from SEG/EAGE salt model as shown in Figure 19. The same model is used for testing migration algorithms. The shot record is generated by time stepping the wavefield from time zero to 1 second with time step interval of 1 ms. Before attempting to do so, bin spacing must be defined. The AFD package from CREWES requires the bin spacing to be equal vertically and horizontally (Youzwishen and Margrave, 1999). The velocity model that is used to generate shot record has a bin spacing that spans 30 meters vertically and horizontally. Next step is to define source and receiver positions. The source is placed on the upper left of the velocity model (x=0) at the surface (z=0). Receivers are placed along the surface (z=0) with the farthest offset located 4500 m away to the right from the source. Both source and receiver intervals are 30 m.

Acoustic wave finite difference modeling requires inputs giving: the temporal and spatial sample sizes, the maximum record time, the velocity matrix, the receiver positions, the wavelet, the desired Laplacian (five point or nine point), and the two initial snapshots of the wavefield (snap1 and snap2 in the m-file script). The snapshots should be initialized to matrices of zeros the same size as the velocity model. Then the source configuration is described by placing appropriate impulses in these two snapshots. A simple strategy is to leave snap1 as all zeros and simply place impulses in snap2.

```
% This script generates single shot gather for a SEG/EAGE salt model. The
% shot is located on the upper left of the model (first location on the
% 10 gathers version). Receivers are set with off-end geometry,
% farthest offset is 4500m. Both source and receivers are located on the
% surface. Record length is 1s with 4ms sample interval.

clear all

load vmodel;
[m,n]=size(vmodel);dx=30;dz=dx;
dtstep=.001;dt=.004;tmax=1;
x=[0:n-1]*dx;z=[0:m-1]*dz;
a=find(x<=4500);
xrec=x(:,1:length(a));zrec=zeros(size(xrec));
snap1=zeros(size(vmodel));
snap2=snap1;
snap2(1,1)=1;
% second order laplacian
[seismogram2,seis2,t]=afd_shotrec(dx,dtstep,dt,tmax,...
    vmodel,snap1,snap2,xrec,zrec,[5 10 30 40],0,1);

% display velocity model
figure;imagesc(x,z,vmodel);
```

```
colormap(flipud(colormap));colorbar;
xlabel('Distance (m)');ylabel('Depth (m)');
title('Velocity model');
%display shot record
plotimage(seismogram2,t,xrec);
xlabel('Offset (m)');ylabel('Time (s)');
```

Figure 13 is the synthetic shot record generated using FD modeling with second-order approximation to the Laplacian operator, while Figure 14 is the same shot generated with fourth-order approximation to the Laplacian operator. Both figures show the effect of using second-order and fourth-order approximation to the Laplacian operator by comparing their frequency contents (Figure 15 and Figure 16). It can be seen that the fourth-order approximation had increased the bandwidth as expected, although not significantly.

Figure 17 is the corresponding 10 shot gathers generated with second-order approximation where both source and receivers are moved to the right. All figures are plotted with a highly clipped display, otherwise, the direct wave near the source would be the only visible arrival. Note the artifacts reflecting from both upper and lower edges of the model. These are typical artifacts known as edge effects. They arise because a computer simulation of wave propagation must always operate in a finite domain with definite boundaries. When a wave encounters such a boundary, it behaves as though it has met a perfect reflector. These boundary reflections would be much worse if absorbing boundaries are not incorporated (Clayton and Engquist, 1977). Obviously, absorbing boundaries are not perfect but they do help. The absorbing boundary conditions are optimized for a small range of wavefront incidence angles around normal incidence. That is, a wavefront that encounters the boundary at normal incidence is completely absorbed, while one making an angle is partially reflected.

```
% This script generates 10 synthetic shot gathers for an SEG/EAGE salt
% model. First shot is on the upper left and then moving to the right with
% 30m shotpoint increment. Receivers are set with off-end geometry,
% farthest offset is 4500m. Both source and receivers are located on the
% surface. Record length is 1s with 4ms sample interval.

clear all

% load SEG/EAGE velocity model
load vmodel;

% set grid size, time step, sample rate and recording length
[m,n]=size(vmodel);dx=30;dz=dx;
dtstep=.001;dt=.004;tmax=1;

% x and z coordinates of the model
x=[0:n-1]*dx;z=[0:m-1]*dz;

% where's offset 4500?
a=find(x<=4500);

% receivers coordinates
xrec=x(:,1:length(a));zrec=zeros(size(xrec));

% create wavefield snapshots at t=0 and t=dtstep. At t=0, the snapshot
% simply all blanks (no shot exploded), at t=dtstep, the snapshot is blank
% everywhere except on the shot location.
snap1=zeros(size(vmodel));
snap2=snap1;

% create 10 gathers, set zeros matrix for recorded seismogram
```

```
ng=10;seismogram=zeros((tmax/dt+1),length(xrec)*ng);
b=length(xrec)-1;seis=zeros(size(seismogram));

% generate shot records, iterate over each record
for i=1:ng
    % where's my shot?
    xpos=find(x==(i-1)*dx);
    % put the explosion on the shot location and then move the receivers
    snap2(1,xpos)=1;xrec=xrec+((i-1)*dx);
    % second order laplacian
    [seismogram(:,i+b*(i-1):i+b*i),seis(:,i+b*(i-
1):i+b*i),t]=afd_shotrec(dx,dtstep,dt,tmax,...
        vmodel,snap1,snap2,xrec,zrec,[5 10 30 40],0,1);
    fprintf('Finished generating shot gather no. %d \n',i);
end

% display velocity model
figure;imagesc(x,z,vmodel);
colormap(flipud(colormap));colorbar;
xlabel('Distance (m)');ylabel('Depth (m)');
title('Velocity model');
% display shot records
plotimage(seismogram,t,1:length(xrec)*ng);
xlabel('Trace no.');ylabel('Time (s)');
```

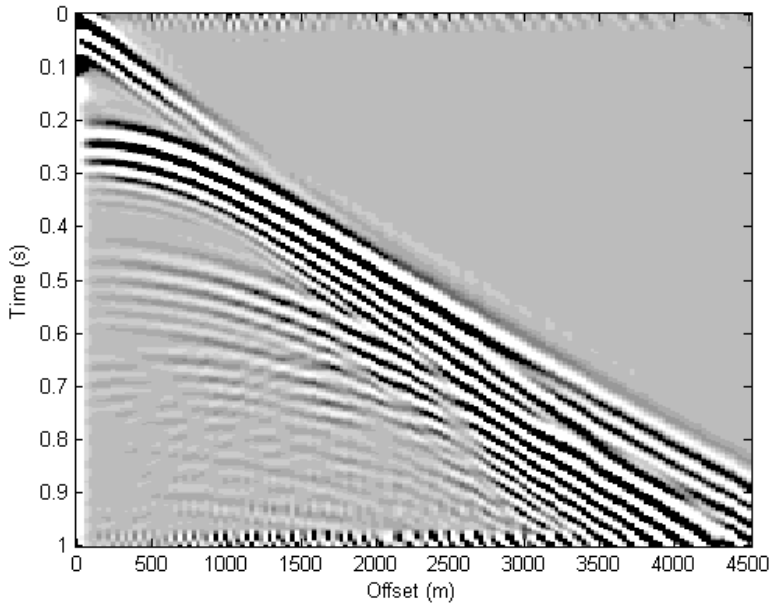


Figure 13. Synthetic shot record for a single gather correspond to the first gather in Figure 17 generated using finite difference with second-order approximation to the Laplacian operator.

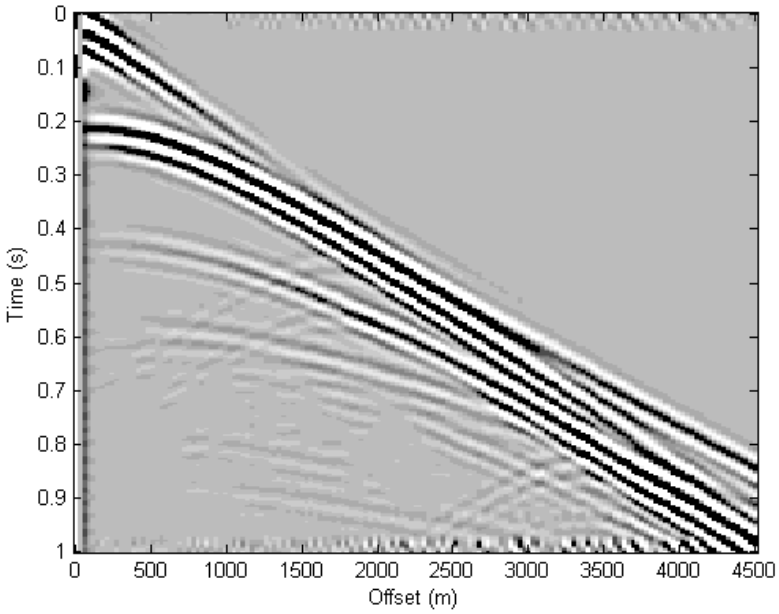


Figure 14. Same as Figure 13, except that fourth-order approximation was used.

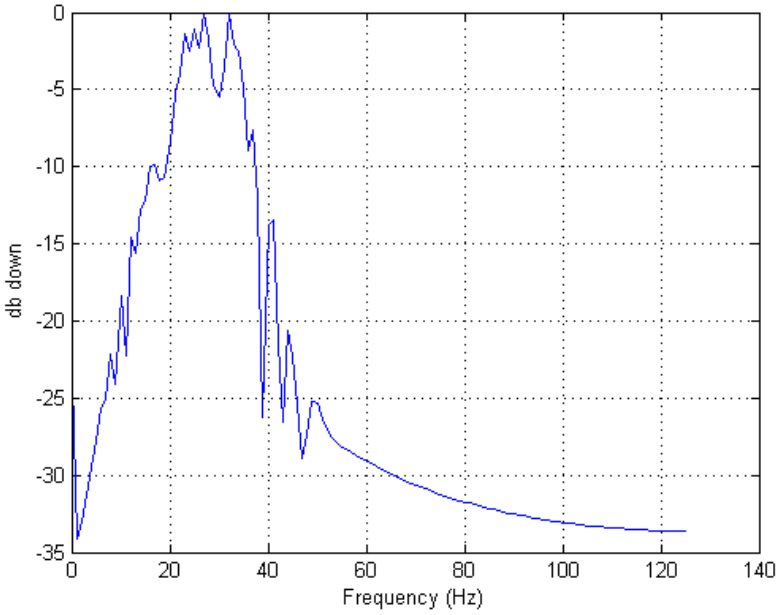


Figure 15. db spectrum of the second-order approximation to the Laplacian operator.

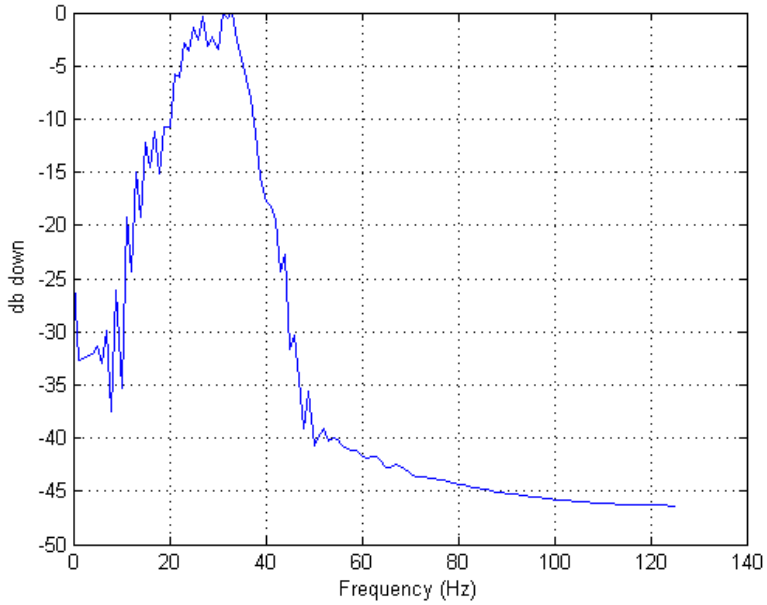


Figure 16. db spectrum of the fourth-order approximation to the Laplacian operator.

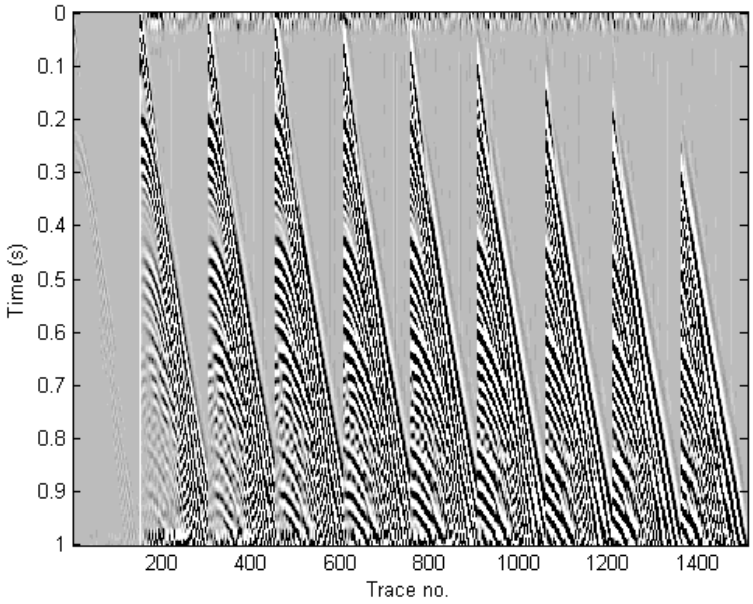


Figure 17. 10 consecutive shot gathers generated using finite difference with second-order approximation to the Laplacian operator.

Migration

In exploration seismology, migration refers to a multi-channel processing step that attempts to spatially re-position events and improve focusing. The term is essentially synonymous with imaging (though the latter term has a specific secondary meaning as a sub-step in a migration process). Before migration, seismic data is usually displayed with traces plotted at the surface location of the receivers and with a vertical time axis. This means that dipping reflections are systematically mispositioned in the lateral coordinate and the vertical time axis needs a transformation to depth. Also problematic is the unfocused nature of seismic data before migration. It is very much like looking through an unfocused camera lens. Just as the construction of a camera lens requires knowledge of the physics of light propagation, migration algorithms incorporate the physics of acoustic and elastic wave propagation. In addition to spatial positioning and focusing, migration algorithms perform amplitude and phase adjustments that are intended to correct for the effects of the spreading (or convergence) of raypaths as wave propagate.

Migration can be viewed as an approximate solution to the general elastic wavefield inversion problem (Gray, 1997). Full elastic inversion uses the entire seismic wavefield as input into a mathematical process that seeks to estimate the elastic parameters of the earth. This is called an inverse problem because it is opposite to the classical forward modeling problem of predicting the seismic wavefield response of a known elastic earth model. Generally, a forward problem can be reduced to finding the solution to a partial differential equation, in this case the elastic wave equation, given specifications of the coefficients of the equation as functions of position and given appropriate boundary conditions. Though often this is a very difficult process, it is usually more easily accomplished than the corresponding inverse problem. The inverse problem usually amounts to estimating the coefficients of a partial differential equation given its response to a known source. Often, these inverse problems are ill-posed, which is a mathematical term for a problem whose inputs are insufficient to determine all of its expected outputs. In addition to being ill-posed, inversion problems are generally nonlinear and practical schemes are typically linearized approximations. This means that they are very sensitive to an assumed initial model. This is unfortunate because knowledge of the subsurface is very limited and the construction of initial models is extremely difficult.

Thus, for many reasons, migration amounts to a very approximate solution to a general, nonlinear inverse problem. The need to find approximate solutions has led to a large number of different migration algorithms that are generally distinguished by the kind of approximations made. As a result there are many different, overlapping ways to categorize migration algorithms. Given that one-way waves will be considered, there are: finite difference algorithms that offer great flexibility with spatial variations of velocity but suffer from grid dispersion and angle limitations, Kirchhoff methods that can easily accommodate arbitrary variations in seismic recording geometry but require raytracing to guide them, and Fourier (or spectral) techniques that offer high fidelity but have difficulty coping with rapid variations in either velocity or recording geometry. All of these techniques attempt, in some manner, to downward continue measurements made at z=0 into the subsurface. There are also useful distinctions about whether the estimation of R(x,y,z) is made directly from z=0, direct methods, or whether R(x,y,z) is estimated from $z-\Delta z$, recursive methods. Finally, perhaps the most common categorization comes under the confusing labels of time migration and depth migration. The former is an earlier technology that remains viable because it is very

insensitive to velocity errors even though it is known to be accurate only if $\partial_x v \approx 0$. On the other hand, depth migration is the only currently available imaging technology that is accurate when $\partial_x v$ varies strongly; however it requires a very accurate specification of the velocity model.

These last remarks hint at the general chicken-and-egg nature of modern migration methods. A true inversion technique would derive the velocity model from the data as part of the inversion. Migration requires the velocity model as input and merely attempts to reposition, focus, and adjust amplitudes. These really all turn out to be the same thing. In order to be a useful process, it should be true that migration requires only an approximate, or background, velocity model and that more detailed velocity information can be extracted from a successful migration than was input to it. This is generally the case though it can be very difficult to achieve in structurally complex areas. Especially for depth migration, the construction of the velocity model is the central difficulty in achieving a successful result.

Exploding reflectors

This is a powerful analogy that can be used to simplify various wave propagation problems. Figure 18 shows a "field experiment" where the data have been acquired via repeated zero-offset experiments. In fact, this is obtained after processing CMP gathers.

In Figure 18 we show a fictitious experiment where the sources are located on top of the reflector. This is the exploding reflector case. Both experiments are equivalent in the sense that they will produce the same wavefield. If the traveltime in the first experiment is divided by two, the two wavefields obtained by these two experiments are equal. We can also assume that the velocity of the medium in the real experiment is half its true value (v=v/2). The exploding reflector analogy permits us to migrate post-stack data ($D(x_m,h=0,t)$). The exploding reflector analogy assumes that the real experiment can be replaced by a fictitious one where only up-going waves generated by the hypothetical explosion of the reflector are considered.

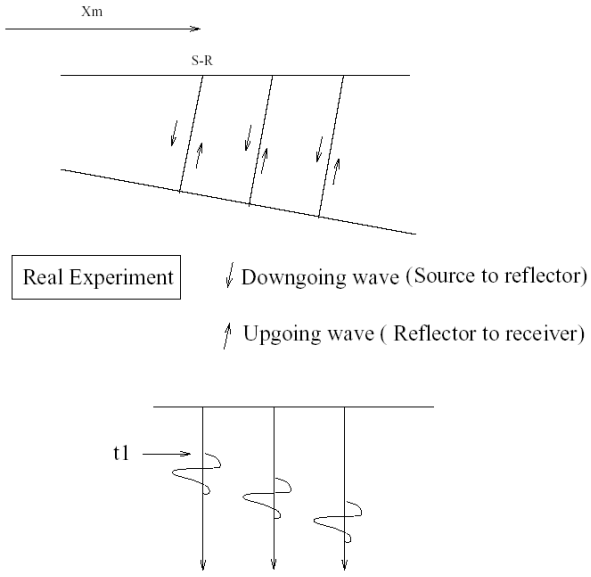


Figure 18. A zero-offset experiment.

We can now define migrations as the operation that maps the data recorded at the surface of the earth (z=0) into an image of the subsurface. If the wavefield recorded at z=0 is denoted by p(t,x,z=0) the migration process is used to compute the image as follows

$$I(x,z) = p(t=0,x,z), \tag{76}$$

where I(x,z) is an estimate of the reflector strength (the reflectivity of the layer). The exploding reflector analogy implies that the reflector explodes at t=0. Therefore, the wavefield at t=0 is nothing else than the image of our fictitious sources located on the reflector. The core of the migration process is an algorithm that permits to propagate our wavefield recorded at z=0 (data) into the subsurface.

In general, one can say that a migration algorithm is composed of two components:

1. A procedure to extrapolate wavefields in depth.
2. An imaging condition.

The imaging condition is described above: reflectors explode at t=0. Next, we will describe a simple procedure to extrapolate wavefields in depth.

The wave equation in (k_x,ω): Introduction to one-way propagators

We will start our analysis by considering the acoustic wave equation in a 2D earth

$$p_{xx} + p_{zz} = \frac{1}{v^2} p_{tt}, \tag{77}$$

where $p_{xx} = \frac{\partial^2 P}{\partial x^2}$, $p_{zz} = \frac{\partial^2 P}{\partial z^2}$ and $p_{tt} = \frac{\partial^2 P}{\partial t^2}$.

The wavefield p(t,x,z) can be transformed into the Fourier space as follows

$$P(\omega,x,z) = \frac{1}{2\pi}\int p(t,x,z)e^{-i\omega t}dt. \tag{78}$$

So far we have only applied the Fourier transform to the temporal variable t. The wave equation in the (ω,x,z) is given by

$$P_{xx} + P_{zz} = \frac{\omega^2}{v^2}P, \; P = P(\omega,x,z). \tag{79}$$

Now we apply again the Fourier transform but with respect to the spatial variable x

$$P(\omega,k_x,z) = \frac{1}{2\pi}\int P(\omega,x,z)e^{-ik_x x}dx. \tag{80}$$

The wave equation in (ω,k_x,z) is given by

$$-k_x^2 P + P_{zz} = -\frac{\omega^2}{v^2}P, \; P = P(\omega,k_x,z). \tag{81}$$

We can rearrange the above equation as follows

$$P_{zz} = -\left(\frac{\omega^2}{v^2} - k_x^2\right)P. \tag{82}$$

The dispersion equation of the scalar wave equation is given by

$$k_x^2 + k_z^2 = \frac{\omega^2}{v^2}, \tag{83}$$

where k_z is the vertical wavenumber. In equation (82) we can recognize the vertical wavenumber $k_z^2 = \frac{\omega^2}{v^2} - k_x^2$

$$P_{zz} + k_z^2 P = 0\,,\ P = P(\omega, k_x, z)\,. \tag{84}$$

The solution to the last equation is a simple exponential of the form

$$P(\omega, k_x, z) = Ae^{i\pm k_z(\omega,k_x)z}\,. \tag{85}$$

The wavefield recorded at the surface is denoted by p(t,x,z=0) (the seismic data). In the (ω,k_x,z) domain, the wavefield is given by P(ω,k_x,z=0). The constant A can be evaluated by considering the wavefield at z=0 (the data).

$$P(\omega, k_x, z) = P(\omega, k_x, z=0)e^{i\pm k_z(\omega,k_x)z}\,. \tag{86}$$

we have found the expression for the downward continuation operator. This is an operator that extrapolates the wavefield at the surface into a depth z below the surface. The exploding reflector idea uses up-going waves. This is why we will choose the minus sign in the extrapolator.

$$P(\omega, k_x, z) = P(\omega, k_x, z=0)e^{-ik_z(\omega,k_x)z}\,, \tag{87}$$

$$k_z = -\sqrt{\left(\frac{\omega^2}{v^2} - k_x^2\right)}\,. \tag{88}$$

The story doesn't end here. It is clear that if we know the velocity v, we can extrapolate our wavefield recorded at z=0 down into the earth, in other words, we can compute $P(\omega, k_x, z)\forall z$. The imaging condition is obtained by transforming back to time and space using the inverse Fourier transform

$$P(\omega, k_x, z) \to F_t^{-1} \to P(t, k_x, z) \to F_x^{-1} \to p(t, x, z)\,, \tag{89}$$

where F_t^{-1} is the inverse Fourier transform over t. Finally, we apply the imaging condition

$$I(x, z) = p(0, x, z)\,. \tag{90}$$

Wavefield extrapolation can be also done in the t,x,z space. Note that wavefield extrapolation in (ω,k_x) is equivalent to a convolution in the (ω,x) domain.

F-K migration

Stolt (1978) showed that the migration problem can be solved by Fourier transform. Here, Stolt's solution will be developed from a formal solution of the wave equation using Fourier transforms. Let ψ(x,z,t) be a scalar wavefield that is a solution to

$$\nabla^2\psi - \frac{1}{v^2}\frac{\partial^2\psi}{\partial t^2} = 0\,, \tag{91}$$

where v is the constant ERM velocity. It is desired to calculate ψ(x,z,t=0) given knowledge of ψ(x,z=0,t). The wavefield can be written as an inverse Fourier transform of its (k_x,f) spectrum as

$$\psi(x, z, t) = \int_{V_\infty} \phi(k_x, z, f)e^{2\pi i(k_x x - ft)}dk_x df\,, \tag{92}$$

where cyclical wavenumbers and frequencies are used and the Fourier transform convention uses a + sign in the complex exponential for spatial components and a – sign for temporal components. If equation (92) is substituted into equation (91), the various partial derivatives can be immediately brought inside the integral where they can be readily evaluated. The result is

$$\int_{V_\infty}\left\{\frac{\partial^2\phi(z)}{\partial z^2}+4\pi^2\left[\frac{f^2}{v^2}-k_x^2\right]\phi(z)\right\}e^{2\pi i(k_x x-ft)}dk_x df=0\,, \tag{93}$$

where the (k_x,f) dependence in $\phi(z)$ has been suppressed for simplicity of notation. The derivation of equation (93) does not require that v be constant; however, the next step does. If v is constant, then the left-hand-side of equation (93) is the inverse Fourier transform of the term in curly brackets. The uniqueness property of Fourier transforms (that there is a unique spectrum for a given function and vice-versa) guarantees that is a function vanishes everywhere in one domain, it must do so in another. Put another way, the zero function has a zero spectrum. Thus, it results that

$$\frac{\partial^2\phi(z)}{\partial z^2}+4\pi^2k_z^2\phi(z)=0\,, \tag{94}$$

where the wavenumber k_z is defined by

$$k_z^2=\frac{f^2}{v^2}-k_x^2\,. \tag{95}$$

Equation (95) is called the dispersion relation for scalar waves though the phrase is somewhat misleading since there is no dispersion in this case.

Post-stack time and depth migration implemented in Matlab

Consider the velocity model in Figure 19. This model is taken from the SEG/EAGE salt model, which has laterally variant velocity and the corresponding stack section is shown in Figure 20. Post-stack time migration is applied first using both constant and vertically varying (v(z)) velocity. Constant velocity is applied using Stolt's (1978) algorithm, and the v(z) migration is applied using F-K migration of Margrave (1998).

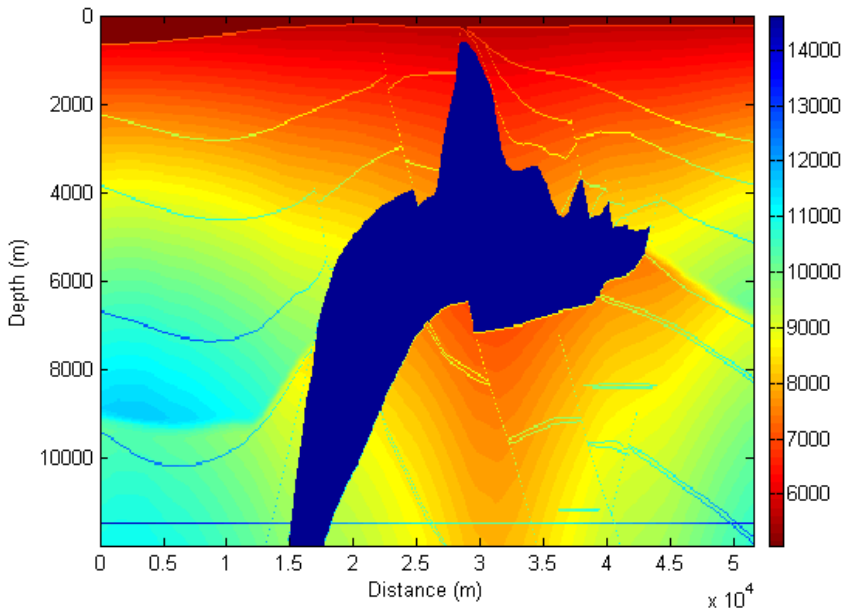


Figure 19. Velocity model used for testing migration algorithms. The model is taken from the SEG/EAGE salt model.

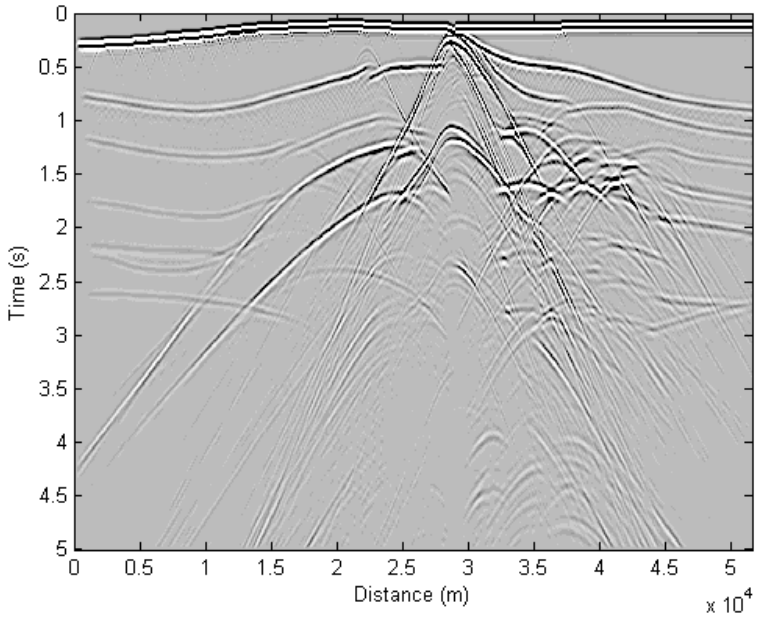


Figure 20. Input stack data of the SEG/EAGE salt model.

```
% constant velocity Stolt's migration
clear all
[seis,dt,header]=altreadsegy('D:\Documents\Projects\etc\data.le.sgy','textheader','y
es');
[m,n]=size(seis);
t=[0:dt:(m-1)*dt]';
dx=40;x=[0:dx:(n-1)*dx];

% constant velocity 7000 m/s
v=7000;
params=NaN*ones(1,13);
params(:,1)=62.5;params(:,3)=90;params(:,4)=10;params(:,12)=1;

[seismig,tmig,xmig]=fkmig(seis,t,x,v,params);
plotimage(seismig,t,x);
xlabel('Distance (m)');ylabel('Time (s)');
```

Constant velocity migration is formalized in the Matlab function fkmig from CREWES. This function has the external interface: [`seismig,tmig,xmig]=fkmig(seis,t,x,v,params)`. Here the first four input variables are simply the seismic matrix, its time coordinate vector, its x coordinate vector, and a scalar giving the velocity of migration. The final input variable is a vector of parameters that control various aspects of the migration. This vector has length thirteen and affords control over the spatial and temporal zero pads, the maximum dip to migrate, the maximum frequency to migrate, the type of spectral interpolation, and the kind of progress messages that are written to the screen. Figure 21 to Figure 23 shows the result of Stolt's migration using constant velocity of 6000, 7000 and 8000 m/s respectively. Note how diffractions are still exists using constant velocity of 6000 m/s and how they collapse properly using constant velocity of 7000 m/s. However, using constant velocity of 8000 m/s, the diffractions turn into smiles indicating over-migration.

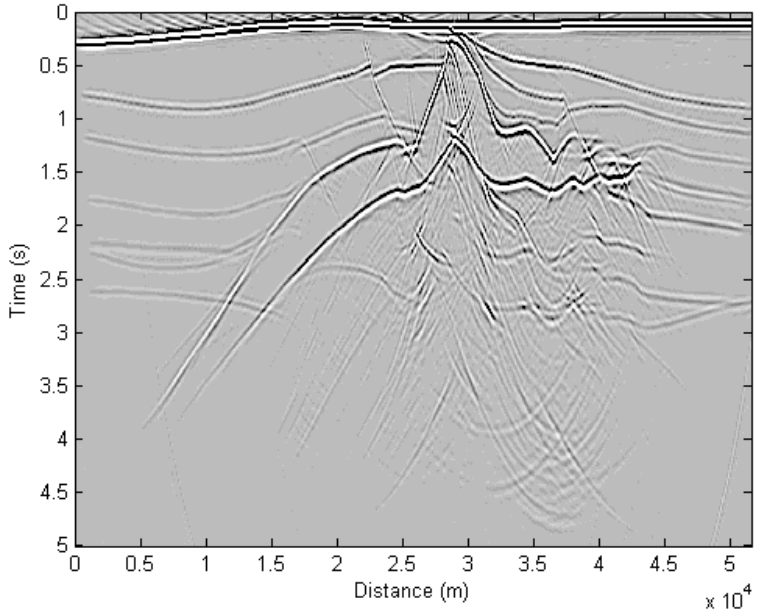


Figure 21. Post-stack time migration using Stolt's algorithm with constant velocity of 6000 m/s. Note how diffractions are still exists indicating under-migration.

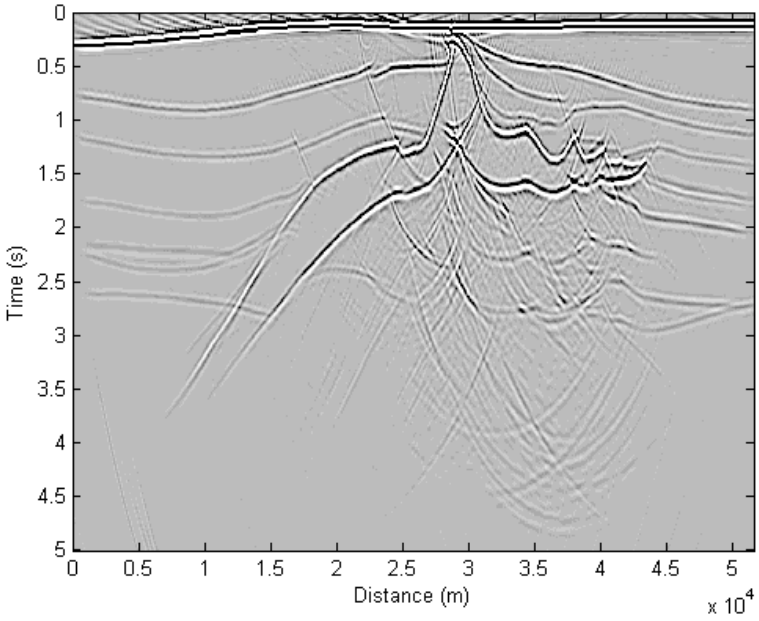


Figure 22. Post-stack time migration using Stolt's algorithm with constant velocity of 7000 m/s. Note how diffractions had been properly collapsed.

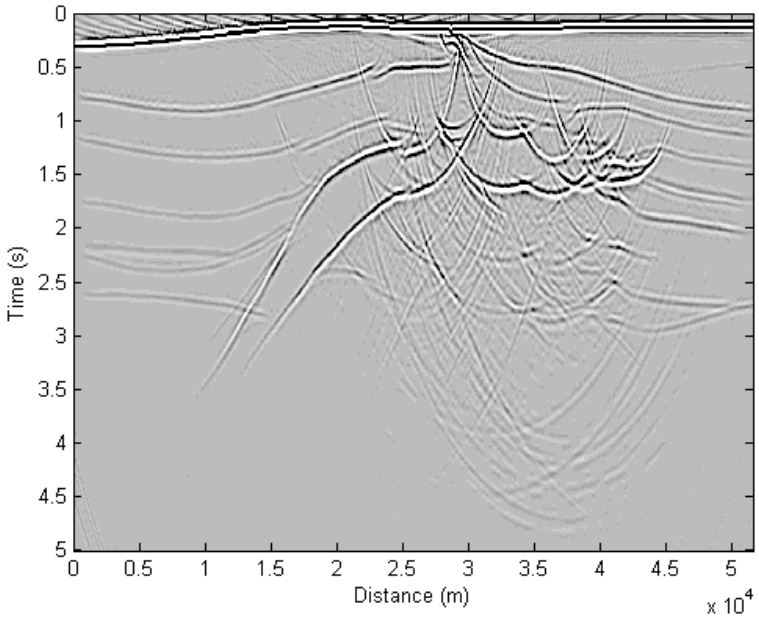


Figure 23. Post-stack time migration using Stolt's algorithm with constant velocity of 8000 m/s. Note how diffractions turn into smiles indicating over-migration.

```
% V(z) F-K migration
% There are 3 options for creating v(z) function, the first one is chosen
% because it gives the best result.

clear all
[seis,dt,header]=altreadsegy('D:\Documents\Projects\etc\data.le.sgy','textheader','y
es');
[m,n]=size(seis);
t=[0:dt:(m-1)*dt]';
dx=40;x=[0:dx:(n-1)*dx];

load vmodel
[mm,nn]=size(vmodel);
dz=40;z=[0:dz:(mm-1)*dz]';
t1=vint2t(vmodel,z);
t2=2*t1;

% option 1: use vint on first location
v=interp1(t2,vmodel(:,1),t,'pchip');

% option 2: use average vint
%vmean=mean(vmodel')';
%v=interp1(t2,vmean,t,'pchip');

% option 3: use a velocity function
%v=1528.6*t+7077;

params=NaN*ones(1,13);
params(:,1)=62.5;params(:,3)=90;params(:,4)=10;
```

```
[seismig,tmig,xmig]=vz_fkmig(seis,v,t,x,params);
plotimage(seismig,tmig,xmig);
```

Figure 24 shows the result of post-stack time migration using v(z) F-K migration algorithm from Margrave (1998). The result gives slightly better imaging due to v(z) approximation, but still doesn't comprehend lateral velocity variation, hence the image is distorted in the subsalt region.

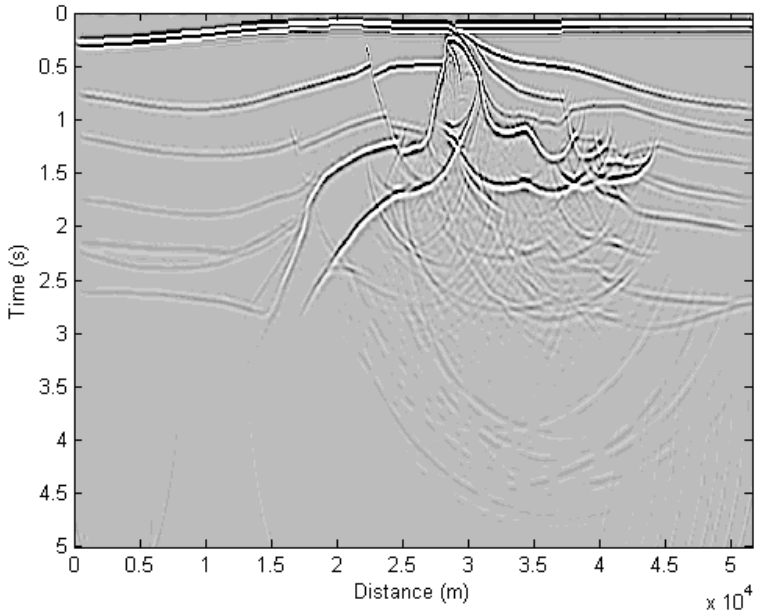


Figure 24. Post-stack time migration using v(z) F-K migration algorithm.

In Figure 25 and Figure 26, the data was migrated with post-stack depth migration using split-step Fourier (Stoffa et al, 1990) and phase-shift plus interpolation (PSPI) algorithm, which is a modification to Gazdag's (1978) v(z) algorithm that can handle lateral velocity variation. Both algorithms are implemented in Seismic Un*x (SU) available from the Center for Wave Phenomena, Colorado School of Mines. This can be implemented using sumigsplit and sumigpspi in SU.

```
sumigsplit <data.le.su vfile=velocities.le.transp nz=300 dz=40 dx=40 >migsplit.le.su
sumigpspi <data.le.su vfile=velocities.le.transp nz=300 dz=40 dx=40 >migpspi.le.su
```

The input velocity file consists of C-style binary floats. The structure of this file is vfile[iz][ix]. Note that this means the x-direction is the fastest direction instead of z-direction. Such a structure is more convenient for the downward continuation type migration algorithm than using z as fastest dimension as in other SU programs.

In PSPI, the wavefield is downward continued using several velocities. The algorithm is equivalent to run Gazdag migration for a set of reference velocities and then interpolate to compute the local wavefield by interpolating wavefield obtained from propagating the data using reference velocities. In general, migration velocities are also called macro velocities or

macro model. These velocities are usually a smooth representation of the real velocities of the subsurface.

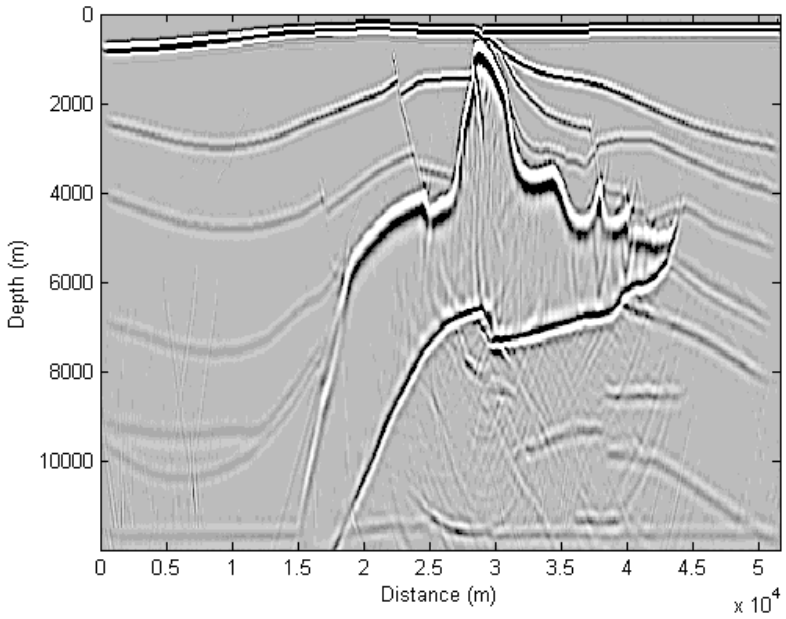


Figure 25. Post-stack depth migration using split step Fourier algorithm. Note the reflector "pull-up" and artifacts in the subsalt.

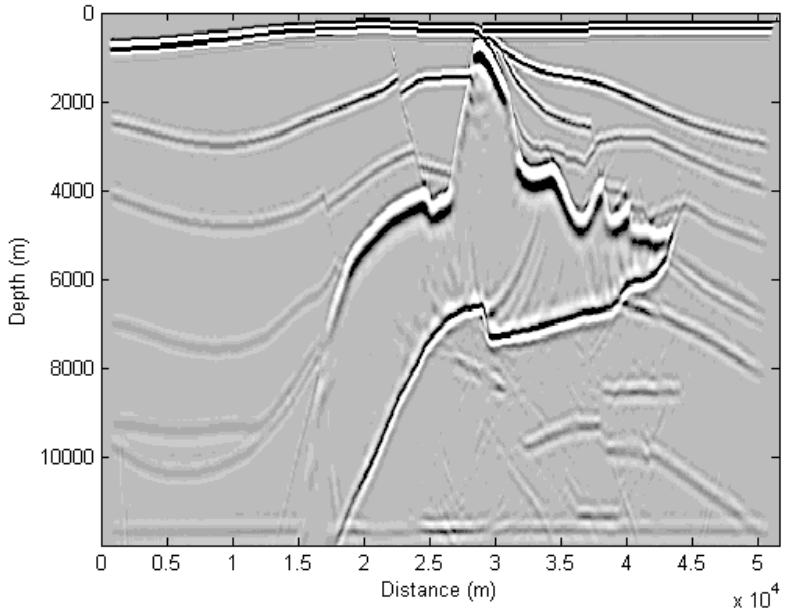


Figure 26. Post-stack depth migration using phase-shift plus interpolation algorithm. Steep dips are not better imaged compared to Figure 25, but the subsalt and salt body region contains much less artifacts.

References

1. Aki, K. and Richards, P. G., 1980, *Quantitative Seismology: Theory and methods*, W. N. Freeman & Co., San Fransisco.
2. Bortfeld, R., 1961, Approximation to the reflection coefficients of plane longitudinal and transverse waves, *Geophys. Prospecting*, **9**, 485 – 502.
3. Burnett, R. C., 1990, Seismic amplitude anomalies and AVO analysis at Mestena Grande field, *Geophysics*, **55**, 1015 – 1025.
4. Cambois, G., 2000, Can P-wave AVO be quantitative?, *The Leading Edge*, **19**, 1246 – 1251.
5. Castagna, J. P., Swan, H. W. and Foster, D. J., 1998, Framework for AVO gradient and intercept interpretation, *Geophysics*, **63**, 948 – 956.
6. Chiburis, E. F., 1984, Analysis of amplitude versus offset to detect gas-oil contacts in the Arabia Gulf, *54th Annual Internat. Mtg., Soc. Expl. Geophys. Extended Abstract*, 669 – 670.
7. Clayton, R. and Engquist, B., 1977, Absorbing boundary conditions for acoustic and elastic wave equations, *Bull. Seis. Soc. Am.*, **67**, 1529 – 1540.
8. Dix, C. H., 1955, Seismic velocities from surface measurements, *Geophysics*, **20**, 68 – 86.
9. Gardner, G. H. F., Gardner, L. W. and Gregory, A. R., Formation velocity and density: The diagnostic basis for stratigraphic traps, *Geophysics*, **39**, 770 – 780.
10. Gazdag, J., 1978, Wave equation migration with the phase-shift method, *Geophysics*, **43**, 1342 – 1351.
11. Gray, S. H., 1997, True amplitude seismic migration: A comparison of three approaches, *Geophysics*, **62**, 929 – 936.
12. Hwang, L. F. and Lelis, P. J., 1988, Bright spots related to high GOR oil reservoir in Green Canyon, *58th Annual Internat. Mtg., Soc. Expl. Geophys. Extended Abstract,* 761 – 763.
13. Jin, S. and Beydoun, W., 1993, A stable elastic inversion for marine data, *63th Annual Internat. Mtg., Soc. Expl. Geophys. Extended Abstract,* 665 – 668.
14. Knott, C. G., 1899, Reflection and refraction of elastic waves with seismological applications, *Philosophical Magazine*, **48**, 64 – 97.
15. Lines, L. R. and Treitel, S., 1984, A review of least squares inversion and its application to geophysical problems, *Geophys. Prospecting*, **32**, 159 – 186.
16. Lines, L. R., Slawinski, R. and Bording, R. P., 1999, A recipe for stability of finite-difference wave-equation computations, *Geophysics*, **64**, 967 – 969.
17. Mallick, S., 1995, Model-based inversion of amplitude variations with offset data using a genetic algorithm, *Geophysics*, **60**, 939 – 954.
18. Margrave, G. F., 1998, Direct Fourier migration for vertical velocity variations, *CREWES Research Report*, **10**.
19. Margrave, G. F., 2001, *Numerical Methods of Exploration Seismology with algorithms in MATLAB*, Department of Geology and Geophysics, University of Calgary.
20. Ostrander, W. J., 1982, Plane wave reflection coefficients for gas sand s at non-normal angles of incidence, *52nd Annual Internat. Mtg., Soc. Expl. Geophys. Extended Abstract*, 216 – 218.
21. Rüger, A., 1998, Variation of P-wave reflectivity with offset and azimuth in anisotropic media, *Geophysics*, **63**, 935 – 947.
22. Shuey, R. T., 1985, A simplification of Zoeppritz equations, *Geophysics*, **50**, 609 – 614.

23. Smith, G. C. and Gidlow, P. M., 1987, Weighted stacking for rock property estimation and detection of gas, *Geophys. Prospecting*, **35**, 993 – 1014.
24. Stoffa, P. L., Fokkema, J. T., Freire, R. M. and Kessinger, W. P., 1990, Split-step Fourier migration, *Geophysics*, **55**, 410 – 421.
25. Stolt, R. H., 1978, Migration by Fourier transform, *Geophysics*, **43**, 23 – 48.
26. Taner, M. T. and Koehler, F., 1969, Velocity spectra-digital computer derivation and applications of velocity functions, *Geophysics*, **34**, 859 – 881.
27. Ursin, B., Ekren, O. and Tjaland, E., 1996, Linearized elastic parameter sections, *Geophys. Prospecting*, **44**, 427 – 455.
28. Youzwishen, C. F. and Margrave, G. F., 1999, Finite difference modeling of acoustic waves in Matlab, *CREWES Research Report*, **11**.
29. Zoeppritz, K., 1919, Erdbebenwellen, on the reflection and penetration of seismic waves through unstable layers, *Göttinger Nachrichten*, **1**, 66 – 84.