MULTI-COMPONENT ACTIVE SOURCE RAYLEIGH WAVE ANALYSIS

by

Gabriel Gribler

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Geophysics

Boise State University

August 2015

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Gabriel Gribler

Thesis Title: Multi-Component Active Source Rayleigh Wave Analysis

Date of Final Oral Examination: 15 May 2015

The following individuals read and discussed the thesis submitted by student Gabriel Gribler and they evaluated his presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Lee M. Liberty, Ph.D.	Chair, Supervisory Committee
Paul Michaels, Ph.D.	Member, Supervisory Committee
John Bradford, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Lee M. Liberty, Chair of the Supervisory Committee. The thesis was approved for the Graduate College by John R. Pelton, Ph.D., Dean of the Graduate College.

ABSTRACT

Determining how a building site will respond to earthquake ground shaking plays a critical role in proper construction practices. One critical constraint on how a site responds is the near surface shear wave seismic velocity distribution. One commonly used method for indirectly estimating shear wave velocities is Multichannel Analysis of Surface Waves (MASW), which utilizes a spread of vertical geophones to measure Rayleigh wave dispersion. With this approach, phase velocity vs. frequency dispersion curve picks can be used to estimate shear wave velocities with depth. I investigate the use of two (vertical and horizontal inline) component seismic signals to record the elliptical Rayleigh wave motion for improved constraints on the phase velocity vs. frequency relationship in a process I term Multi-Component Analysis of Surface Waves (MCASW). Using MCASW allows me to better constrain Rayleigh wave dispersion at lower frequencies, leading to more accurate estimates of shear wave velocities at greater depths compared to the traditional MASW approach. I can also use multiple seismic components to determine particle motions to identify and remove select Rayleigh wave modes. I show that my polar mute approach leads to a further improvement of shear wave velocity estimates from Rayleigh wave signals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

MCASW	Multi Component Analysis of Surface Waves
MASW	Multi Channel Analysis of Surface Waves
SASW	Spectral Analysis of Surface Waves
ReMi	Refraction Microtremor Analysis
AGC	Automatic Gain Control
PMT	Particle Motion Transform
PM	Polar Mute

CHAPTER ONE: INTRODUCTION

A seismic site response analysis provides an estimate of how a building or structure will respond to earthquake ground motion. Two critical constraints for site response are the stiffness of sediments overlying a hard boundary (bedrock) and depth to that hard boundary (bedrock) (Rodriguez-Marek et al., 2001). The shear modulus, which correlates to sediment stiffness, is directly related to shear wave velocity under small strain conditions (Craig, 1992); thus, shear wave velocity estimates play a critical role in determining earthquake site response. The depth to bedrock, or other high velocity boundary, will determine the frequency at which the overlying sediment column will resonate, causing local or regional site amplifications (Rodriguez-Marek et al., 2001). For determining subsurface geologic structure and for indirectly estimating depth to hard boundaries, seismic reflection or refraction (body wave) methods are often used (e.g., Pelton, 2005; Chapter 8).

Shear wave velocities determined by borehole measurements (Pickett, 1963) offer a way to directly estimate shear wave velocities at depth. However, boreholes offer a one dimensional measurement and are expensive to drill. Dorman and Ewing (1962) first documented that surface waves (Rayleigh waves and Love waves) can be utilized to estimate shear wave velocities at depth (also Black et al., 1989). Rayleigh waves are guided waves that propagate in an elliptical (two-dimensional) motion along the surface of the earth (Strutt, 1885). For a simple earth model, with seismic velocities increasing with increasing depth, Rayleigh waves become dispersive, meaning that different seismic frequencies propagate at different seismic velocities. This phenomenon is due to longer wave length (lower frequency) Rayleigh waves probing to greater depths, thus propagating at greater phase velocities. It was determined by Dorman and Ewing (1962) that shear wave velocity profiles could be estimated by inversion of Rayleigh wave phase velocities at a range of frequencies. The ability to get estimates of shear wave velocities vs. depth by the inversion of phase velocity vs. frequency (which can be measured at the surface of the earth) allows for indirect estimates of shear wave velocities at depth.

Spectral Analysis of Surface Waves (SASW) was the first active source surface method developed to measure Rayleigh wave dispersion (phase velocity vs. frequency) and utilizes two vertically oriented geophones and a vertically polarized source (Stokoe & Nazarian, 1983). A more robust method for measuring Rayleigh wave dispersion, known as Multichannel Analysis of Surface Waves (MASW) utilizes a linear spread of 24 or more evenly spaced, vertical geophones (Park et al., 1999). Both of these methods (SASW and MASW) measure frequency dependent phase velocities from about 5-100 Hz, limited by the Rayleigh wave frequencies that can be generated by a simple active source (Stephenson et al., 2005). A passive source alternative to MASW, known as Refraction Microtremor (ReMi) analysis, employs a linear array of vertical geophones to record ambient noise (often car traffic) to measure surface wave dispersion from about 2-30 Hz (Louie, 2001; Stephenson et al., 2005). These methods offer an opportunity to indirectly estimate subsurface shear wave velocity distribution that are comparable to borehole measurements (Stephenson et al., 2005).

Traditional SASW, MASW, and ReMi methods, used by the engineering seismology community, utilize single component, vertical geophones and only capture one component of ground motion. Recent advancements of ground-coupled seismic land streamers allow for seismic data collection without the time and effort of planting geophones (van der Veen et al., 2001). While these seismic land streamers allow for rapid deployment and data collection for single component (generally vertically polarized geophones), no additional field effort is required to collect three-component data, to record the entire wave field (e.g., Pugin et al., 2004). While multicomponent seismic data have been utilized by industry and earthquake seismologist for decades, this approach has seen limited use within the engineering seismology community. For example, the oil and gas industry utilize three component active source seismic data to improve subsurface physical property estimates and to improve imaging capabilities for targets located many thousands of meters below the earth's surface (e.g., Hardage, 2011). Earthquake seismologists employ three component broadband sensors to determine earthquake magnitude, location, and source direction (e.g., Brune, 1970), and to estimate whole earth properties by estimating p-wave and s-wave velocity distributions (e.g., Aki & Richards, 1980). Seismologists also utilize multiple seismic components by examining the spectral ratio of the horizontal and vertical (H/V) components to estimate the fundamental site period that may correspond to large ground motion amplification (Nakamura, 1989; SESAME, 2004). Horizontal component data approaches in the near surface community are limited, with some s-wave reflection and refraction studies to estimate subsurface velocity distributions and to characterize subsurface stratigraphy (e.g. Hasbrouck, 1986; Carr et al., 1998; Dasios et al., 1999; Pugin et al., 2004) and Love wave methods to estimate shear wave velocities using horizontally polarized surface waves (Black et el., 1989). It is currently not an established practice to record and analyze multiple component surface (Rayleigh) wave data within the near surface seismology community.

Here, I explore the use of multi-component Rayleigh wave data to improve shear wave velocity estimates and earthquake site response analysis for a typical sedimentary basin velocity distribution. Although my models and data are valid for estimating high gradient shear wave velocities within the upper 50 meters, my approach is also applicable to any geologic environment and scale. I first introduce Rayleigh wave theory and describe particle motions measured from two components (horizontal in-line, or radial, and vertical components). I then examine the integration of these two components into the traditional MASW approach that I have termed Multi-Component Analysis of Surface Waves (MCASW). I also utilize multiple components to determine Rayleigh wave motion, either retrograde or prograde motion, to selectively isolate the fundamental or higher modes. I include both modeled data and field records to show that improved shear wave velocity estimates can be obtained by the utilization of the complete Rayleigh wave motion. My results suggest that by capturing the complete particle motion of the Rayleigh waves, improved shear wave velocity estimates can be obtained to greater depth with greater confidence.

Rayleigh Wave Theory

Rayleigh waves are guided surface waves that arise from seismic energy propagating in an elastic medium with a free surface (Strutt, 1885; Kramer, 1996). Solving the elastic wave equation for a plane wave in a homogenous half space with a free boundary (vanishing stress) condition (the earth’s surface) yields a coupled p-wave and vertically polarized shear wave that propagates with both vertical and horizontal inline displacement (Figure 1) (Strutt, 1885). The depth of Rayleigh wave propagation is dependent on the period or frequency of propagation. A series of Rayleigh waves propagating through the shallow subsurface at a given velocity can have different frequencies and these are known as different modes of Rayleigh waves, with the lowest frequency defined as the fundamental mode while all of the higher frequencies are known as higher modes (e.g., Garland, 1979; Aki and Richards, 1980; Xia et al., 2003). These higher modes are especially common where alternating high and low velocity layers appear (e.g., Gucunski and Woods, 1991). On a semblance plot of frequency vs. phase velocity, the fundamental mode generally shows the highest coherence in the lower frequencies and the higher modes follow a similar, but shifted frequency/velocity relationship with high coherence at the higher frequencies (Xia et al., 2003).

Rayleigh wave motion in a homogenous half space is the simplest earth model case. For example, for a phase velocity of 675 m/s, the vertical and horizontal displacements decay with depth (Figure 1). At the surface, Rayleigh wave motions are in retrograde or counter clockwise motion with negative vertical and positive horizontal displacements (Strutt, 1885). At a depth of approximately 1/5 the dominate wavelength, the horizontal displacement vanishes, corresponding to a vertical oscillation (Figure 1). At greater depths, the horizontal displacement resumes with negative displacement (reverse motion from above) while the vertical displacement remains negative. This deeper Rayleigh wave motion is referred to as prograde, or clockwise, motion. The particle motion transition depth from retrograde to prograde motion for a given velocity distribution is frequency dependent, and lower frequencies (longer wavelengths) will transition from retrograde to prograde motion at greater depths (Figure 1). This transition also occurs with higher modes and is important to separate the fundamental and higher mode signals when examining particle motions. With increasing depth, both the vertical and horizontal component motion approaches zero. This implies that a typical active source Rayleigh wave approach to estimating shear wave velocities with depth is limited to about the upper 100 meters of the subsurface, often significantly shallower. However, this active source approach is typically acceptable to obtain Vs30 measurements that are a standard approach to site characterization (Building Seismic Safety Council of the National Institute of Building Sciences, 2009).

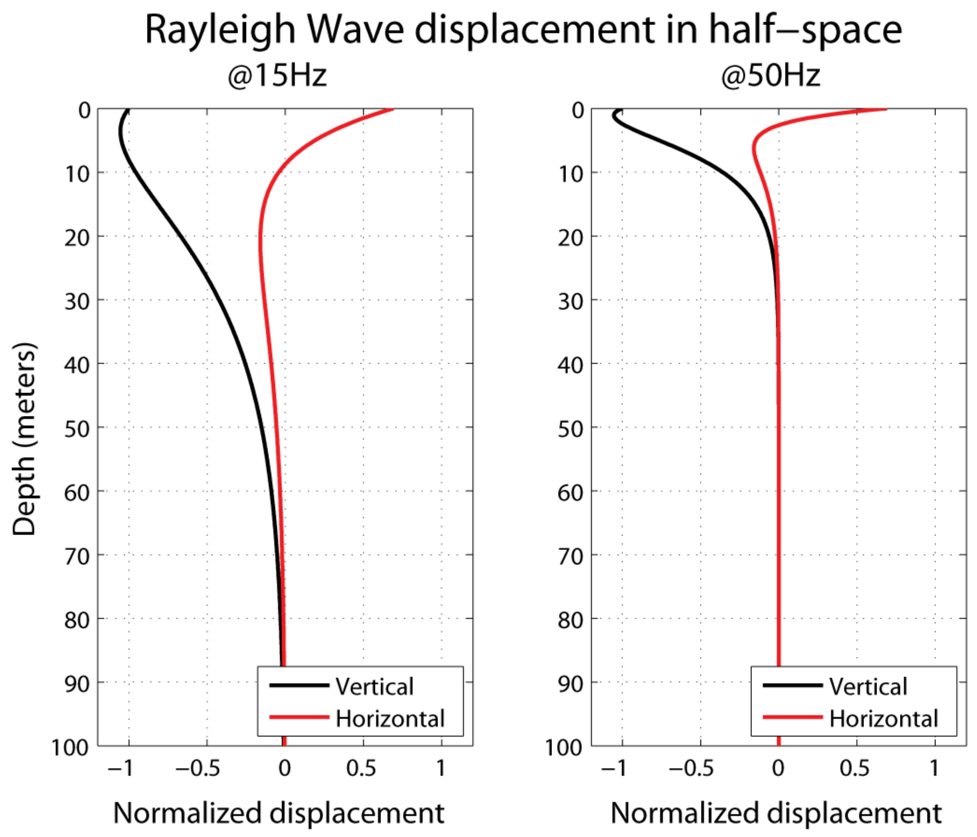

Figure 1 Vertical and horizontal particle displacements versus depth for Rayleigh wave in a half space with a phase velocity of 675 m/s. (Left) displacement at 15 Hz (Right) displacement at 50 Hz. Sign convention from Aki and Richards (1980).

CHAPTER TWO: MULTI COMPONENT ANALYSIS OF SURFACE WAVES (MCASW)

Rayleigh Wave Modeling

To explore elliptical Rayleigh wave displacement and the benefit of recording both vertical and horizontal ground motions from active source seismic surveys, I create synthetic shot gathers that contain only the Rayleigh wave displacements and exclude all body wave energy. The one dimensional algorithm to generate these synthetic shot gathers was derived from Aki and Richards (1980, Ch. 7) and was coded by Michaels and Smith (1997). The velocity distribution used in this model represents an unsaturated soil column with a high shear wave velocity gradient increasing from 250 m/s to 1700 m/s from the earth's surface to 100 m depth (Figure 2). P-wave velocities increase from 400 m/s to 2500 m/s for that same depth range (Figure 2). It should be noted that the elastic moduli and mass density are linearly interpreted in the model (Figure 2) and the velocities were derived from:

$$P - wave\ velocity = \sqrt{\frac{\lambda + 2\mu}{\rho}} \qquad (1)$$

$$S - wave\ velocity = \sqrt{\frac{\mu}{\rho}} \qquad (2)$$

This relationship will yield a velocity gradient for both P and S waves that follow a square root function (Figure 2; Aki & Richards, 1980).

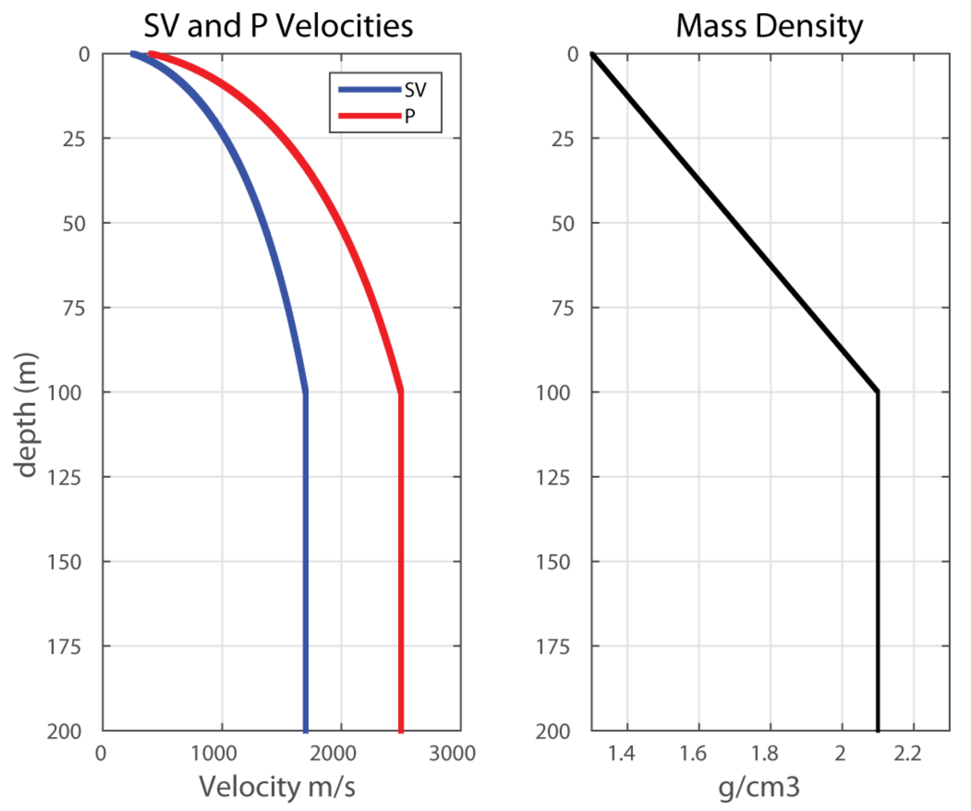

Figure 2 Left) Velocity profile representing an unsaturated, unconsolidated soil column. Blue line denotes S-wave velocities, increasing from 250 m/s at the surface to 1700 m/s at a depth of 100 meters and P-wave velocities increasing from 400 m/s at the surface to 2500 m/s at 100 meters. Right) Mass density with depth ranging from 1.3 g/cm^3 at the surface to 2.1 g/cm^3 at 100 meters depth.

For this velocity model, I investigate particle displacement versus depth at 12 Hz for both the vertical and horizontal component as well as the motion stress vector versus depth (Figure 3). This frequency choice is arbitrary and is used to determine if the model is correctly solving for the necessary boundary conditions (Strutt, 1885). For particle displacement, there is a larger amount of displacement on the horizontal component in the upper few meters relative to the vertical component. At 20 meters depth, the horizontal component reverses direction, marking the transition from retrograde motion to prograde motion. Below this depth, both the vertical and horizontal displacements diminish with an increase in depth. The motion stress vector versus depth plot confirms that this model solves for the boundary condition of vanishing stress at the surface, which will give rise to Rayleigh waves.

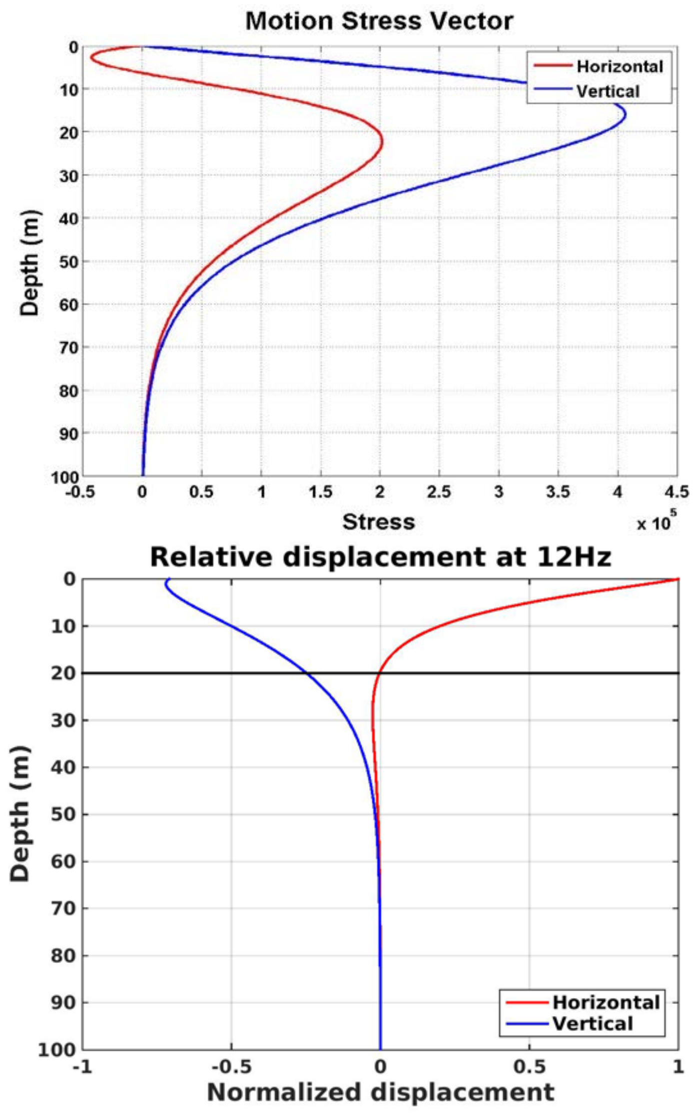

Figure 3 (top) Normalized particle placement with depth in the vertical (blue) and horizontal (red) direction at 12 Hz. bottom) Motion stress vector for the vertical and horizontal component with depth showing that the model is correctly solving for the boundary conditions of vanishing stress at the surface and with depth. The horizontal lines denote where particle motion changes from retrograde to prograde motion. Sign convention from Aki and Richards (1980).

To emphasize the use of multicomponent surface wave measurements, I generate synthetic shot gathers containing only Rayleigh waves for both the vertical and horizontal (radial inline) components (Figure 4). For this model, there is no cross-line (transverse) surface wave motion. These individual shot gathers were converted into the frequency-phase velocity (dispersion) domain via the phase shift transform which sums the amplitudes that lie along different search velocities at every frequency (Park et al., 1998) (Figure 4). I take into account cylindrical spreading, to individually gain traces dependent on the distance from the source. At low frequencies, the dispersion images are typically very low amplitude relative to higher frequencies. To improve low frequency dispersion trends, I gain the amplitudes at a given frequency by the maximum amplitude in a 2 Hz window above the given frequency. I overlay the theoretical dispersion curve (Aki & Richards, 1980; Ch. 7) on my dispersion images to show that the modeled shot gathers accurately represent the expected phase velocity versus frequency relationship above 10Hz (figure 5). Below 10 Hz, the high amplitude zone diverges from the expected dispersion curve from limited sampling of the low frequency, long wave length arrivals from having a finite aperture. By comparing the vertical and horizontal dispersion images, I show nearly identical dispersion above 10 Hz for both the vertical and horizontal component data and either dispersion curve can be used to accurately pick Rayleigh wave dispersion for this model, above 10 Hz.

Particle Motion Transform (PMT)

For simplicity, I investigate elliptical (in-line) particle motion as a single time series through a Particle Motion Transform (PMT). I first calculate the maximum vector amplitude for each sample and for each recorded component:

$$Max\ Amplitude = \sqrt{Vertical^2 + Horizontal^2} \quad (3)$$

I retain the sign of only one component to maintain the proper ground motion period (frequency). To weight the resulting dispersion domain to the vertical component, I retain the sign of the vertical component:

$$Vertical\ PMT = sign(Vertical) * \sqrt{Vertical^2 + Horizontal^2} \quad (4)$$

For greater depths and lower frequencies, where horizontal component motion dominates, I maintain the sign of the horizontal component:

$$Horizontal\ PMT = sign(Horizontal) * \sqrt{Vertical^2 + Horizontal^2} \quad (5)$$

With elliptical particle motion, the horizontal and vertical components each have two zero crossings, when particle motion in that plane transitions from positive to negative and back again. When generating the PMT, these four zero crossings must be mapped into two zero crossings, which creates truncations in the data. These truncations can introduce both low and high frequency artifacts, as seen in the unfiltered PMT shot gather as well as the power spectra (Figure 4). However, for a typical subsurface velocity model, these artifacts appear to predominately affect frequencies outside the frequency range of typical SASW or MASW surveys. When I apply a 60 Hz low pass Butterworth filter to my PMT shot gather example, the surface wave dispersive trend matches the trend seen with a single component, indicating predominately a high frequency effect (Figure 4). The power spectra shows a two order of magnitude increase at low frequencies, indicating some sort of low frequency effect. I will explore how these differences in the power spectra present themselves in the phase velocity vs. frequency (dispersion) domain.

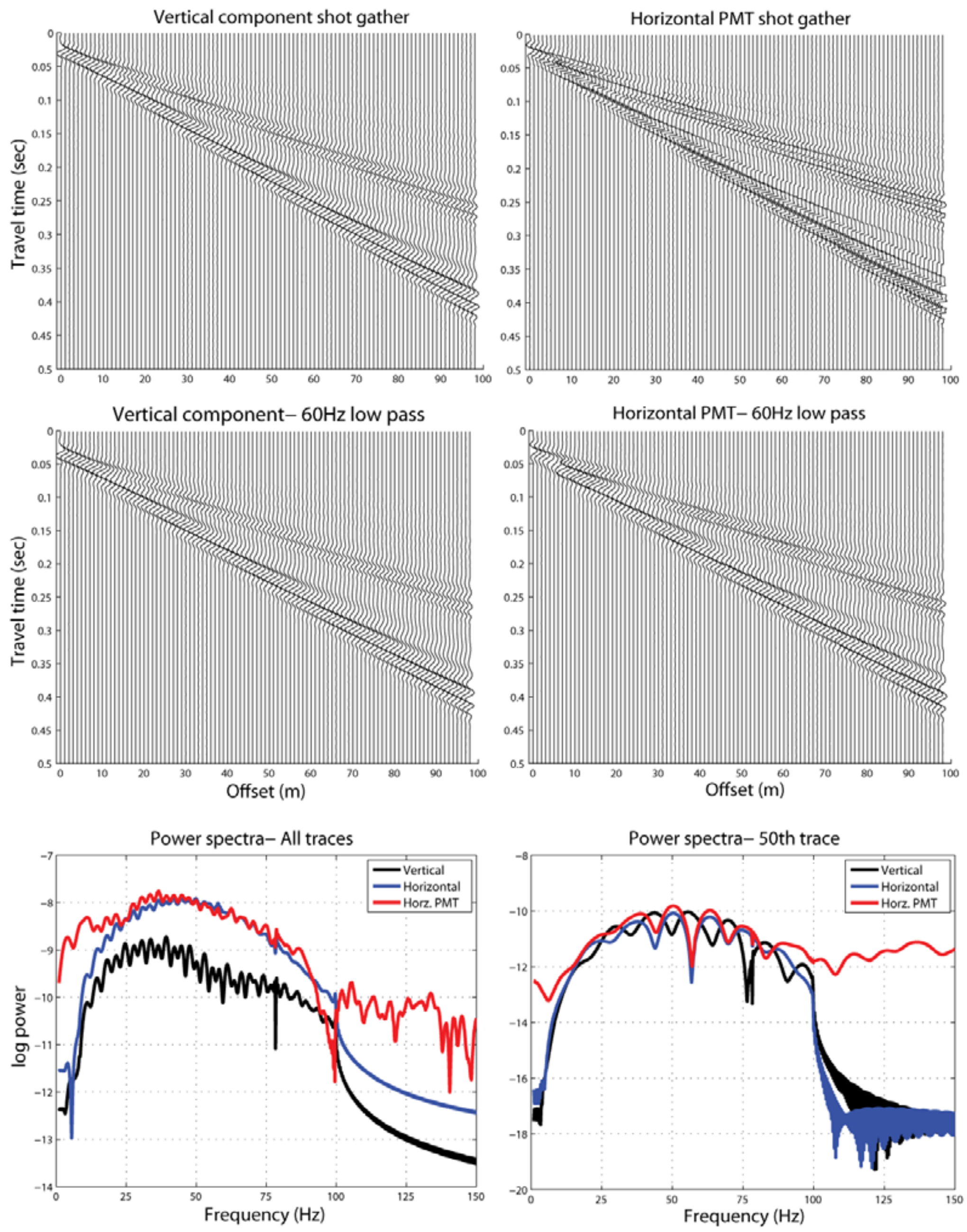

Figure 4 Top left: Vertical component shot gather. Top right: Horizontal PMT shot gather. Middle left: Vertical component shot gather with two pole low pass Butterworth filter with a 60 Hz cutoff. Middle right: Horizontal PMT with identical low pass filter. Bottom left: Power spectra for all traces for the vertical (black) horizontal (blue) and horizontal PMT (red). Bottom right: Power spectra for 50th trace only.

PMT Dispersion Analysis

As noted before, phase velocities for this model show high coherence to as low as 10 Hz using only vertical or horizontal component dispersion curves (Figure 5). To investigate if a PMT approach can improve our confidence in dispersion curve picks at lower frequencies to produce improved shear wave velocity estimates with depth, I create dispersion images for both the horizontal and vertical PMT data sets (Figure 5). I define an improvement in pick confidence by an increase in relative amplitude (increased coherence) and/or a decrease in the width of the semblance envelope of the fundamental mode at the correct phase velocity at a given frequency when compared to only vertical (standard approach) or horizontal semblance plots.

For the vertical PMT, the maximum amplitude for my model matches the theoretical dispersion curve above about 17 Hz (Figure 5). Between about 13 Hz and 17 Hz, the semblance diminishes, creating a low confidence zone when compared to single component semblance plots (Figure 5). This low semblance zone results from an interference effect from higher modes and will be explored later in this chapter. At frequencies less than approximately 13 Hz, the vertical PMT coherence increases again, however the high coherence zone does not track the anticipated dispersion.

For the horizontal PMT, the maximum semblance matches the theoretical dispersion curve (Figure 5). Below approximately 15 Hz, the peak coherence follows the theoretical dispersion curve with a tighter envelope compared to the single component dispersion plots, leading to higher confidence phase velocity picks at lower frequencies (Figure 5).

To determine if my PMT approach can improve confidence in dispersion curve picks when compared to a single component approach, I present cross sections through the non-gained vertical, horizontal, vertical PMT, and horizontal PMT dispersion plots (Figure 6). At 25 Hz, all four dispersion plots show a distinct peak in amplitude at the predicted velocity of the fundamental mode, with the highest amplitude observed on the horizontal PMT. At this frequency, the only improvement from PMT over vertical component data is an increase in maximum amplitude of about 45% without any noticeable change in the fundamental mode envelope. At 15 Hz, I observe a clear peak on the vertical and horizontal component amplitude at the predicted velocity. Note that the horizontal component has approximately 30% higher amplitude when compared to the vertical component accompanied with a decrease in the envelope, indicating higher confidence on the horizontal component over the vertical component signal. The horizontal PMT has the highest amplitude, but slightly underestimates the predicted phase velocity (Figure 5). The vertical PMT has the lowest maximum amplitude and shows no peak in semblance at the predicted velocity, correlating to the low semblance zone in the dispersion plot (Figure 5). Both PMT fail to improve confidence at 15 Hz due to the effects seen in an around this low amplitude zone on the vertical PMT (13-17 Hz). At 10 Hz, a subtle peak can be seen on both the vertical and horizontal amplitude correlating to the predicted phase velocity. The horizontal PMT shows an increase in peak amplitude of approximately 230%, compared to the vertical component, and correlates to the predicted phase velocity. At 7 Hz, the single component amplitude cross sections shows a very small peak at the predicted velocity leading to a very low confidence dispersion pick. The horizontal PMT semblance cross section shows a larger amplitude signal that correlates to the predicted phase velocity, with an increase in peak amplitude of approximately 600% compared to the vertical component semblance. For this model, by using the horizontal PMT, I can have higher confidence dispersion curve picks below 10 Hz. By utilizing the PMT dispersion plots, I have shown that I can more accurately and confidently estimate the phase velocity vs. frequency relationship at lower frequencies, thus, more accurately estimate shear wave velocities to greater depths compared to single component data. The horizontal PMT best matches the anticipated dispersion for this model because the horizontal motion is larger than the vertical motion (Figure 3). For other models where the vertical motion may be larger, the vertical PMT would likely best match the theoretical dispersion.

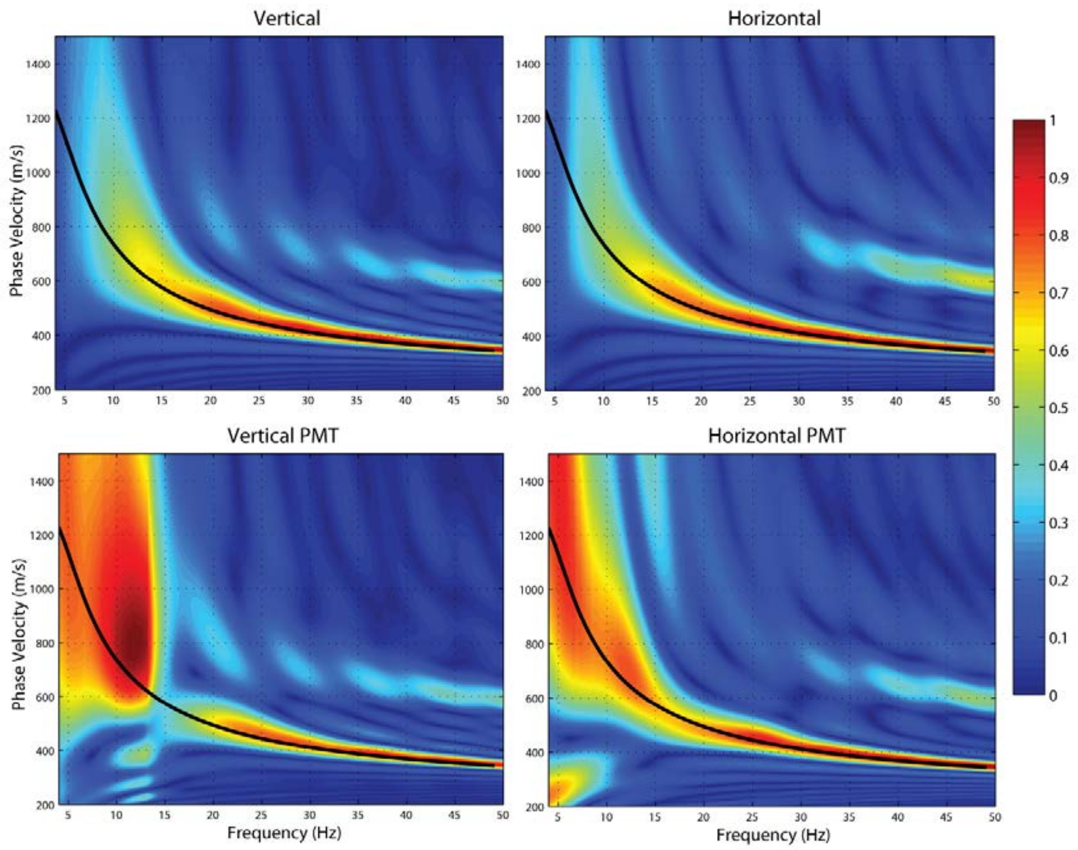

Figure 5 Dispersion images for vertical, horizontal, vertical PMT and horizontal PMT. Gained at each frequency to the maximum amplitude in a leading 2 Hz window. Black line denotes theoretical dispersion curve.

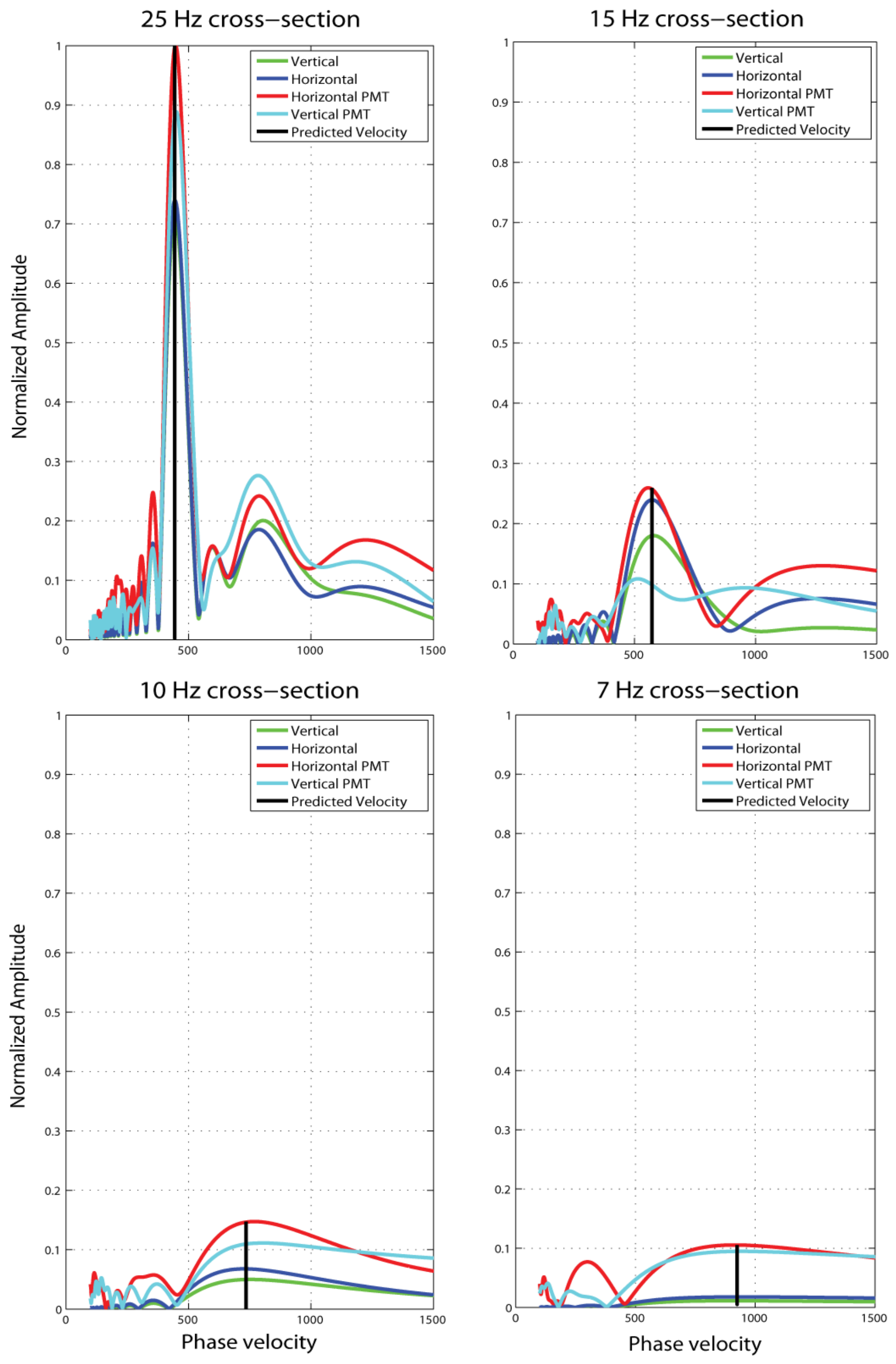

Figure 6 Cross sections through vertical (green), horizontal (dark blue), horizontal PMT (red) and vertical PMT (baby blue) dispersion semblance (figure 5). Vertical black line denotes predicted phase velocity from theoretical dispersion curve. All cross sections normalized to the maximum semblance for 25 Hz cross section.

Vertical PMT Interference Effect

I now investigate the low amplitude region between 13-17 Hz, seen on the vertical PMT and found along the fundamental dispersion (Figure 5). Since the vertical PMT maintains the sign of the vertical component, it may be related to a particle motion transition, as a change in particle motion from retrograde to prograde results from a change in sign of the horizontal component. Since the fundamental mode is always in retrograde motion (Figure 3), this low semblance area may be due to a particle motion transition of a higher order mode.

To explore this potential higher mode effect, I generate synthetic shot gathers containing the fundamental and differing higher order modes. With only the fundamental mode present, the vertical PMT does not contain a low amplitude zone from 13-17 Hz, indicating there is no particle motion transition on the fundamental mode (Figure 7). When the first higher mode is included, there is also no noticeable effect on the fundamental mode causing the low amplitude zone. While the second higher mode appears to create a slight effect in the region of the low amplitude zone, no prominent low amplitude zone has developed in the effected region. However, when I include the fundamental, first and second higher mode, the low amplitude zone is prominent and indicates an interference effect caused by the first two higher modes. This higher mode effect may be related to a change in particle motion of the first and/or second higher mode for this model (Figure 7).

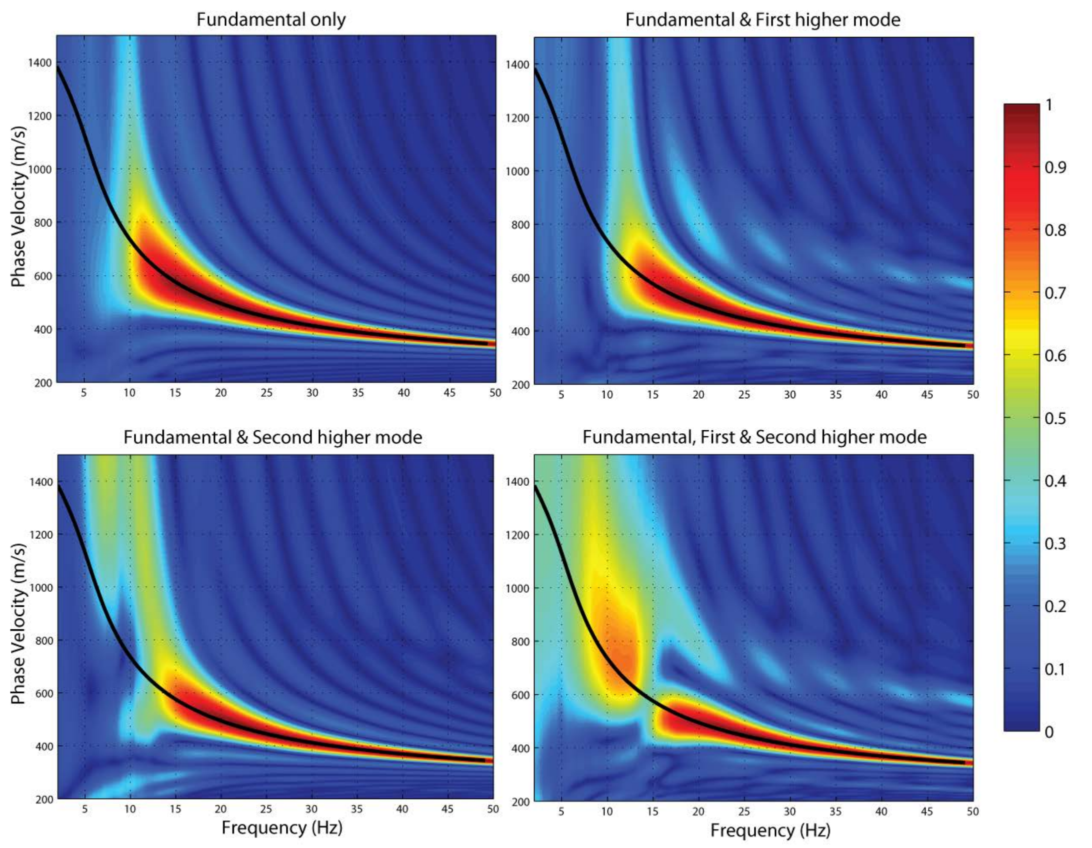

Figure 7 Normalized vertical PMT dispersion images for shot gathers containing energy from select modes of propagation. Top left includes only the fundamental mode. Top right includes the fundamental mode and first higher mode only. Bottom left contains the fundamental and second higher mode only. Bottom right contains fundamental, first and second higher mode only. Black line denotes the theoretical dispersion curve for this velocity model. Note the low amplitude area between 13-17 Hz on the bottom right figure.

PMT Dispersion Domain Artifacts

As previously stated, my PMT approach can introduce both high frequency (>75 Hz) and low frequency (<20 Hz) spectral changes (Figure 4). To determine if these spectral changes introduce artifacts that could degrade the fundamental or higher modes in the dispersion domain, I take the difference between the horizontal PMT and the vertical dispersion plots derived from my unfiltered synthetic shot gathers (Figure 8). After differencing the two dispersion domains, the highest amplitude (greatest difference) follows the anticipated dispersion of the fundamental, first and second higher modes.

This indicates that the horizontal PMT creates an increase in amplitude of coherent Rayleigh wave energy. The low frequency spectral changes arise from a combination of two effects. Between 7 Hz and 25 Hz, the increase in amplitude is predominately from an increase in amplitude of the fundamental mode. Below 10 Hz, the increase in amplitude occurs from an artifact that can be seen from 5 Hz to 10 Hz and between 200 m/s to 400 m/s. This artifact is below the fundamental mode window and does not appear to affect dispersion along the fundamental mode path. This analysis suggests that the artifacts introduced from the PMT do not significantly affect the surface wave dispersion in the frequencies of interest (5-50 Hz) for this model.

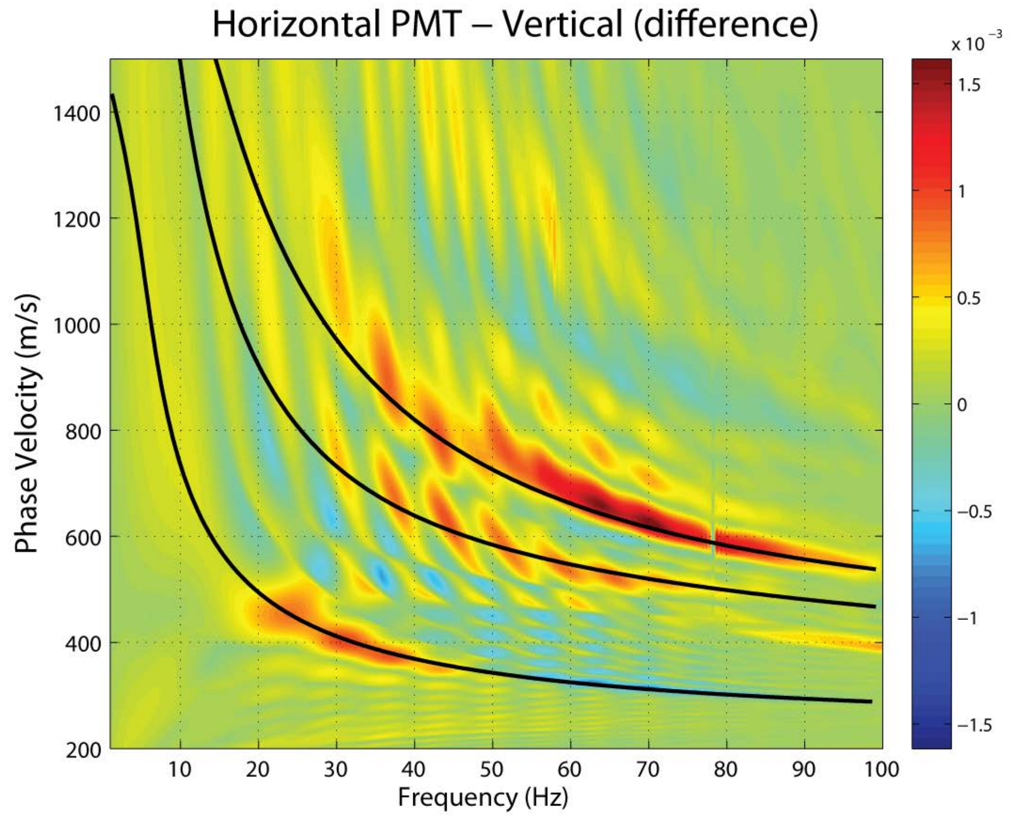

Figure 8 Difference dispersion image derived by taking the ungained horizontal PMT dispersion plot and subtracting the ungained vertical dispersion plot derived from my unfiltered synthetic shot gathers. Positive color values denote higher amplitudes on horizontal PMT dispersion plot while negative values denote higher amplitudes on the vertical dispersion plot. The slowest velocity/frequency black line denotes the fundamental mode theoretical dispersion curve, followed by the first then the second higher mode dispersion curves.

Polar Mute (PM)

The use of two seismic components allows me to record and analyze elliptical Rayleigh wave motion. The fundamental Rayleigh wave mode propagates almost exclusively in retrograde motion, except in a few rare cases at very low frequencies (Mooney and Bolt, 1966) and well below the range of frequencies measured with active source methods. Depending on the near surface velocity distribution, higher Rayleigh wave modes can propagate in prograde motion and I will show using my modeled data that the higher modes are predominantly in prograde motion for my model. By using multiple components to determine the particle motion direction (retrograde or prograde), I can differentiate between fundamental and higher mode arrivals. By differentiating the different particle motion directions, I can selectively remove the fundamental or higher order modes with a polar mute (PM) approach.

To determine Rayleigh wave particle motion using vertical and horizontal inline time series, I first convert my time series from Cartesian coordinates into polar coordinates, using the following equations:

$$Amplitude = \sqrt{Vertical^2 + Horizontal^2} \quad (6)$$

$$Phase\ Angle = \tan^{-1}\left(\frac{Vertical}{Horizontal}\right) \quad (7)$$

The sign convention used is a downward displacement (positive depth/ negative height) is a positive vertical displacement and motion away from the source location on the horizontal axis would represent a positive horizontal displacement. With the above sign convention, Rayleigh waves propagating with retrograde motion would travel counter-clockwise and the phase angle would decrease with increasing time. Prograde Rayleigh waves would propagate clockwise, with the phase angle increasing with time. Using polar coordinates, I determine the dominant particle motion by calculating the phase angle for every time sample, then I calculate a running sum of the phase angles to produce a total phase angle vs. time series (Figure 9). Retrograde motion (mostly fundamental mode energy) will yield a negative slope and prograde motion (mostly higher mode energy) will yield a positive slope (Figure 9). I fit a high degree polynomial (50th degree) to the total phase angle vs. time to smooth the trend and to remove the higher frequency fluctuations (Figure 9). The amplitude information, derived when converting from

Cartesian to polar coordinates, can also be used to remove low amplitude arrivals relative to the fundamental Rayleigh wave arrivals, such as body wave signals (reflections and refractions) and other coherent or random noise sources. However, I do not address this possible benefit as my modeled shot gathers contain only Rayleigh waves without any body waves or other noise sources.

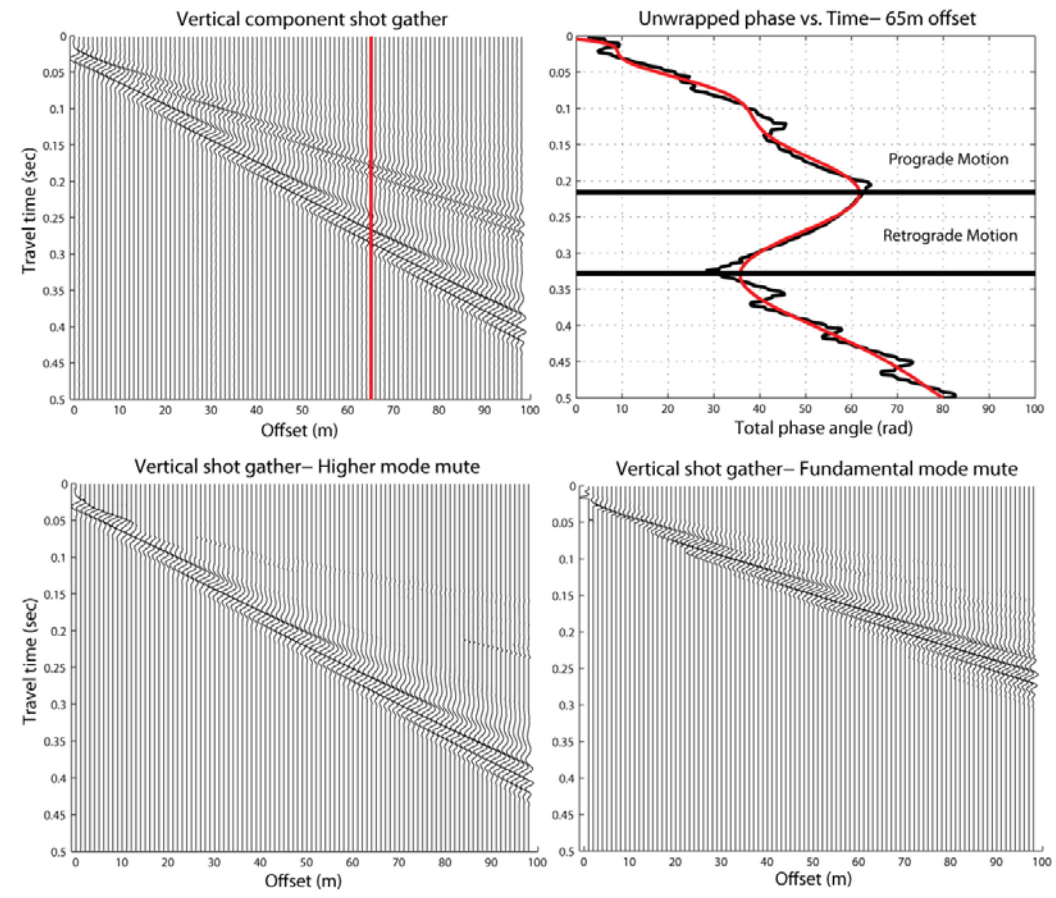

Figure 9 Top left: Vertical component Rayleigh wave synthetic shot gather that is trace normalized. Red line denotes 65th trace where the example analysis was done. Top right: Unwrapped phase angle (summed phase angle) over time. Black line denotes unwrapped phase calculation while red line is a 50th degree polynomial fitted line to get the overall trend. Horizontal black line denotes the changes in slope of the fitted line denoting changes in dominate particle motion. Bottom left: Vertical component shot gather with the higher modes selectively removed using the polar mute. Bottom right: Vertical component shot gather with the fundamental mode removed.

To better isolate the fundamental mode, I utilize the total phase vs. time for each trace to zero out samples that have a positive trend (prograde motion) (Figure 9). I will refer to this prograde polar mute as a higher mode mute. To determine the validity of this surgical mute, I compare the vertical PMT dispersion plots with (Figure 10) and without (Figure 5) the higher mode mute. As discussed in this chapter, the low semblance zone along the fundamental dispersion on the vertical PMT is due to an interference effect from the first and second higher modes (Figure 7). Once the higher mode mute is applied, this interference effect disappears, indicating that the surgical mute is capable of attenuating the higher mode arrivals. All four dispersion plots with the higher mode mute (Figure 10) show similar dispersion, however; below 15 Hz, the high amplitude region diverges from the expected fundamental dispersion for all plots. This low frequency, high velocity information is lost when the higher modes are muted because these high velocity arrivals fall in the region that is dominated by the higher modes. One downside of working in the time domain is the fundamental and higher modes cannot always be separated.

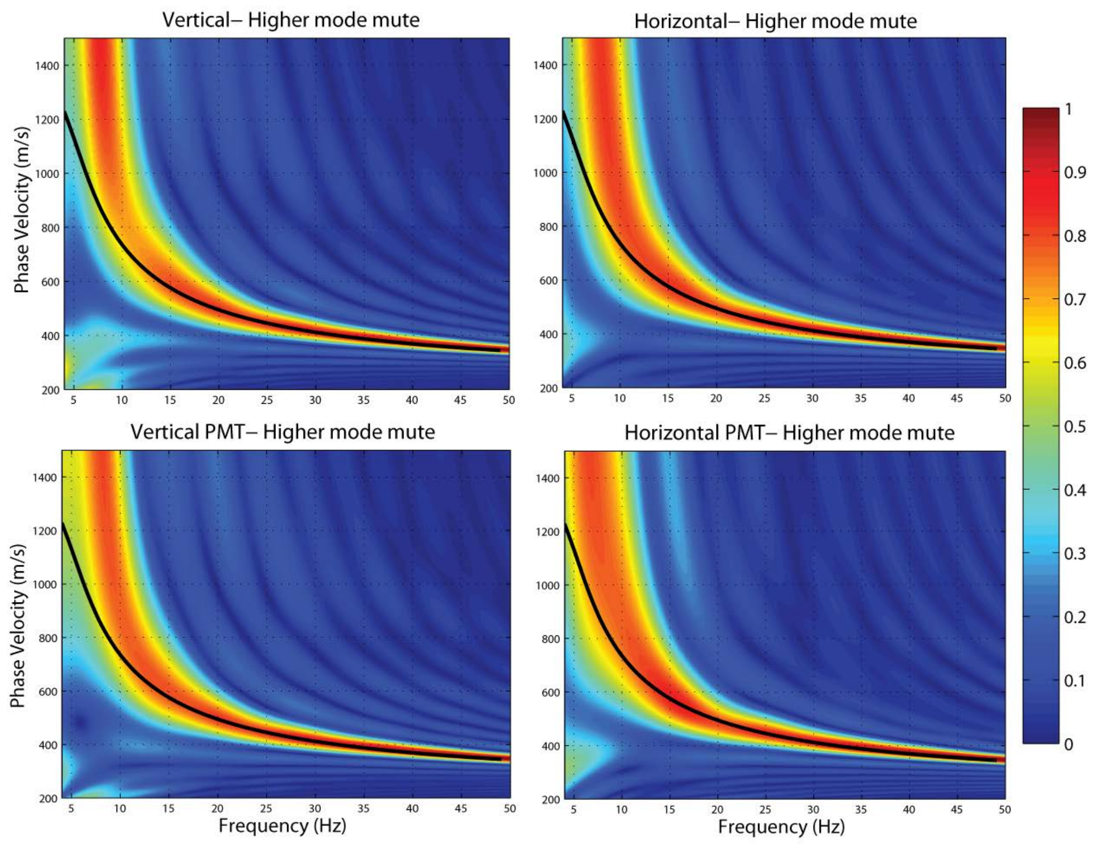

Figure 10 Vertical, horizontal, vertical PMT and horizontal PMT gained dispersion plots with higher mode mute applied. Black line denotes theoretical dispersion curve.

To isolate many of the higher modes, I again utilize the total phase vs. time for each trace to zero out samples that have a negative trend (retrograde motion) (Figure 9). I will refer to this retrograde polar mute as a fundamental mode mute. Figure 11 shows that the higher modes have been predominately isolated and the majority of the fundamental mode has been removed. The low frequency, high velocity portion of the fundamental mode remains because these arrivals lie within the zone where the higher modes and fundamental mode signals cannot be separated (Figure 11). It can be seen that the vertical and vertical PMT dispersion plots best captures the expected dispersion of the first higher mode above approximately 20 Hz, but very little energy from the second higher mode. The horizontal and horizontal PMT predicts the expected dispersion of the first higher mode above approximately 30 Hz and also predicts the dispersion of the second higher mode above 60 Hz. By selectively removing the fundamental mode, the higher modes have a higher relative amplitude compared to the dispersion plots without the fundamental mute applied (Figure 5).

With a standard approach to generate dispersion plots, the dispersion domain becomes dominated by the side lobes of the fundamental mode, which mask many higher mode signals (Figure 5). These fundamental mode side lobes are an artifact of frequency domain dispersion analysis and may be mistaken as higher order mode signal. In Figure 5, the higher modes appear as slight amplitude modulations of the fundamental side lobes. By removing most of the fundamental mode energy, these lobes can be attenuated, allowing the higher modes to become more apparent. By using the fundamental mute, I have the ability to correctly identify and characterize higher mode signals.

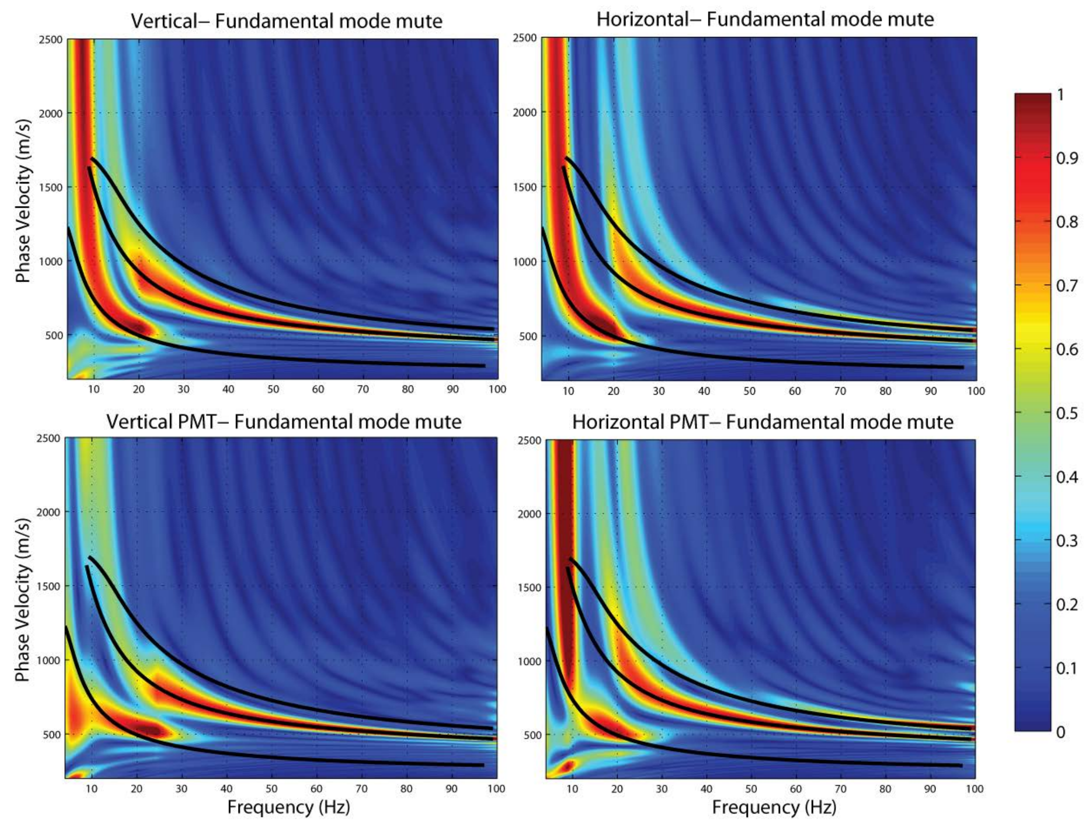

Figure 11 Vertical, horizontal, vertical PMT and horizontal PMT gained dispersion plots with fundamental mode mute applied. Black lines denotes fundamental, first higher order and second higher order modes.

CHAPTER THREE: MULTI-COMPONENT SEISMIC LAND STREAMER

Seismic land streamers were developed as an alternative to planting geophones. This approach allows for more rapid acquisition of land based active source seismic data with a smaller field crew for a variety of field conditions (e.g. van der Veen et al., 2001). Land streamers are composed of a fixed line of active source seismic sensors (geophones) that can be pulled behind a vehicle and seismic source, similar to marine hydrophone streamers pulled behind a boat. Multi-component data can also be acquired with no additional field time and effort over single channel data (e.g., Pugin et al., 2004), as the additional components can be mounted, oriented, and leveled in the streamer.

I have designed and constructed a multi-component land streamer for acquiring seismic data in an urban environment for a variety of road surfaces. The main structural component of the streamer is recycled fire hose that acts as non-stretch material to keep the geophones at a fixed spacing, yet flexible to allow for the streamer to be easily deployed and retrieved. The fire hose also houses and protects geophone cables from any wear and tear as the streamer is pulled along the road surface. The geophones are mounted to steel shoes, which consist of a threaded plate that is bolted through the fire hose to a length of four inch channel steel, which couples the geophones to the road surface (Figure 12). Up to three geophones can be mounted and oriented in any fashion for each shoe, depending on the type of survey being carried out. My streamer design consists of 48 shoes that are mounted one meter apart and can be pulled at variable fixed offsets behind a vehicle. The Boise State multi-component land streamer system with a field crew of three people has shown to collect up to a kilometer an hour of high fold, high density, seismic data to be used for both reflection and surface wave processing procedures (Liberty and Gribler, 2014).

For field tests presented here, I utilize a trailered 200 kg accelerated weight drop and a 96 channel (47 m aperture) land streamer system with one meter receiver spacing and a source to first geophone separation of five meters. For my Rayleigh wave analysis, I utilize two component data acquisition with a 4.5 Hz vertical geophone and a 4.5 Hz inline horizontal geophone to record in-plane elliptical Rayleigh wave motion produced from a vertically polarized source.

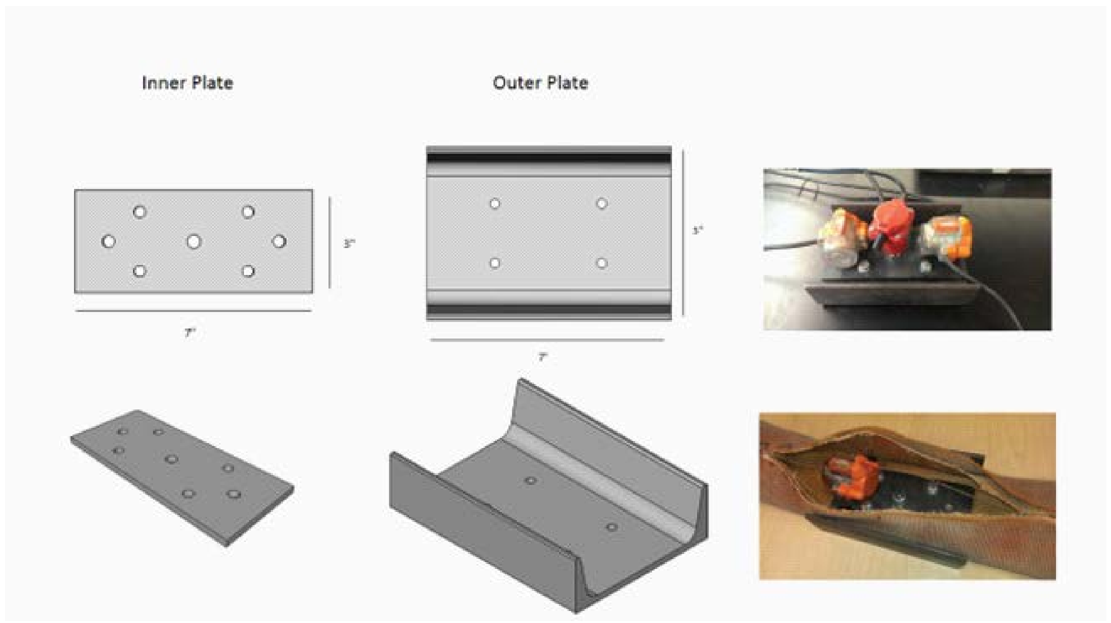

Figure 12 Seismic land streamer shoe design that allows for the use of up to three independent geophone elements that can be oriented in any fashion. Shoes are mounted into three inch diameter recycled firehose.

CHAPTER FOUR: MCASW FIELD EXAMPLE

To confirm that the MCASW technique can be used to generate improved dispersion images and thus provide an improved estimate of subsurface shear wave velocity estimates at greater depths compared to single component (vertical-only) data, I apply the PMT and polar mute approaches to field data that include surface wave and body wave signals as well as cultural noise sources. The data were collected near a school in the town of Donnelly Idaho (Liberty and Gribler, 2014) where unconsolidated fluvial and overbank flood deposits lie above a lacustrine sedimentary basin (e.g., Giorgis et al., 2006). Here, slow seismic velocities at the surface were documented with a positive velocity gradient increasing with depth. The velocity gradient is much lower than my modeled example, however I will show that MCASW works for a variety of cases. A Boise State team collected data using the multi-component land streamer system (Liberty and Gribler, 2014). The streamer and source were located on a compact dirt road surface above unconsolidated soil that allowed for good coupling between the ground and the streamer shoes and clear and dispersive Rayleigh wave arrivals (Figure 13).

The first step in processing the field data is to apply a time domain top mute to the shot records to remove body wave arrivals. I then generate dispersion curves for the vertical, horizontal, vertical PMT, and horizontal PMT shot gathers (Figure 13). The vertical and horizontal dispersion plots show similar coherence on the dispersion plots at frequencies above 10 Hz. At frequencies below 10 Hz, the dispersion for the vertical and horizontal components begin to diverge relative to each other, with the high coherence region on the horizontal component indicating slightly higher phase velocity compared to the vertical component coherence. The horizontal dispersion plot also appears to have higher coherence at lower frequencies (< 10 Hz) when compared to the vertical component, indicating that the horizontal component may provide better constraints at the lower frequencies (greater depths) for this field site. Having higher amplitudes on the horizontal component at lower frequencies indicates that this field site has a larger amplitude horizontal component compared to the vertical component, indicating that I should use the horizontal PMT. Investigating the PMT dispersion results, as expected the horizontal PMT shows better constraints on phase velocity picks at lower frequencies compared to the vertical PMT. The horizontal PMT shows a clear high amplitude (orange) pick down to approximately 7 Hz while the last high coherence pick on the vertical PMT is only down at approximately 13 Hz (Figure 13).

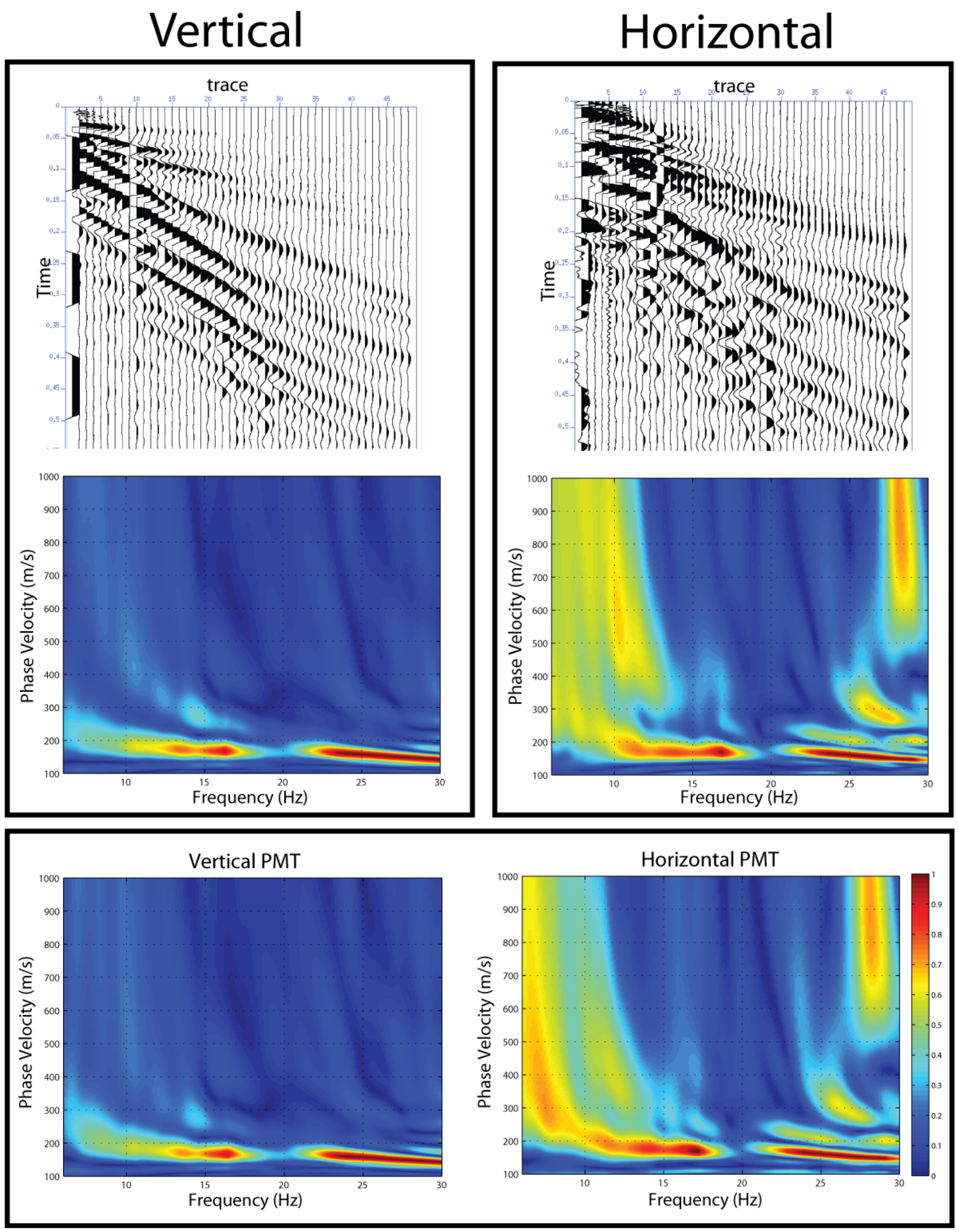

Figure 13 Top left: Vertical shot gather and accompanying dispersion image from field data. Top right: Horizontal (radial inline) shot gather and accompanying dispersion image. Bottom: Both vertical and horizontal PMT dispersion plots.

As stated in Chapter One, Rayleigh waves have different modes of propagation that can be separated in the frequency domain, because at a given phase velocity these modes propagate at different frequencies. The fundamental mode is the lowest frequency at a given phase velocity and higher modes arrive at higher frequencies. When the difference in frequencies between the fundamental and first higher mode at a given phase velocity is small, the fundamental mode can become contaminated by the first higher mode. In the dispersion domain, this contamination can cause the fundamental and first higher mode to merge, leading to an overestimation in the fundamental phase velocity at a given frequency. The horizontal PMT from my field site (Figure 13) contains a small amount of contamination of the fundamental mode, between 10 Hz and 15 Hz, where the fundamental and first higher mode merge and the fundamental envelope, or region, is slightly pulled up to higher phase velocities. At 20 Hz, there is a lower amplitude region along the fundamental mode and the phase velocities increase slightly, indicating either a velocity inversion in the subsurface or contamination of the fundamental by higher modes (Figure 13). If higher mode contamination is present in the region above 20 Hz, phase velocities may be overestimated, resulting in an overestimation of the shear wave velocities for shallow layers.

To address higher mode contamination below 15 Hz, I apply the higher mode polar mute to attenuate the prograde higher mode signal in the time domain. With the higher mode mute applied, the first higher mode is greatly reduced relative to the fundamental mode and the fundamental envelope no longer appears to be effected by the first higher mode (Figure 14). Above 20 Hz, the higher mode mute does not remove or diminish the high amplitude region previously discussed, indicating that it may in fact be part of the fundamental mode (Figure 14). To improve my confidence that this high amplitude region above 20 Hz is part of the fundamental mode and not a higher mode, I apply a fundamental mute. With the fundamental mode (retrograde signal) attenuated, I can isolate many of the higher modes (Figure 14). With the fundamental mode removed, the high amplitude region above 20 Hz is removed, indicating that this coherent signal is part of the fundamental mode or a higher mode propagating in retrograde motion. With the polar mute approach, the surface wave modes become more clear and continuous, allowing for higher confidence in correctly picking the phase velocity-frequency relationship and correct identification of each mode.

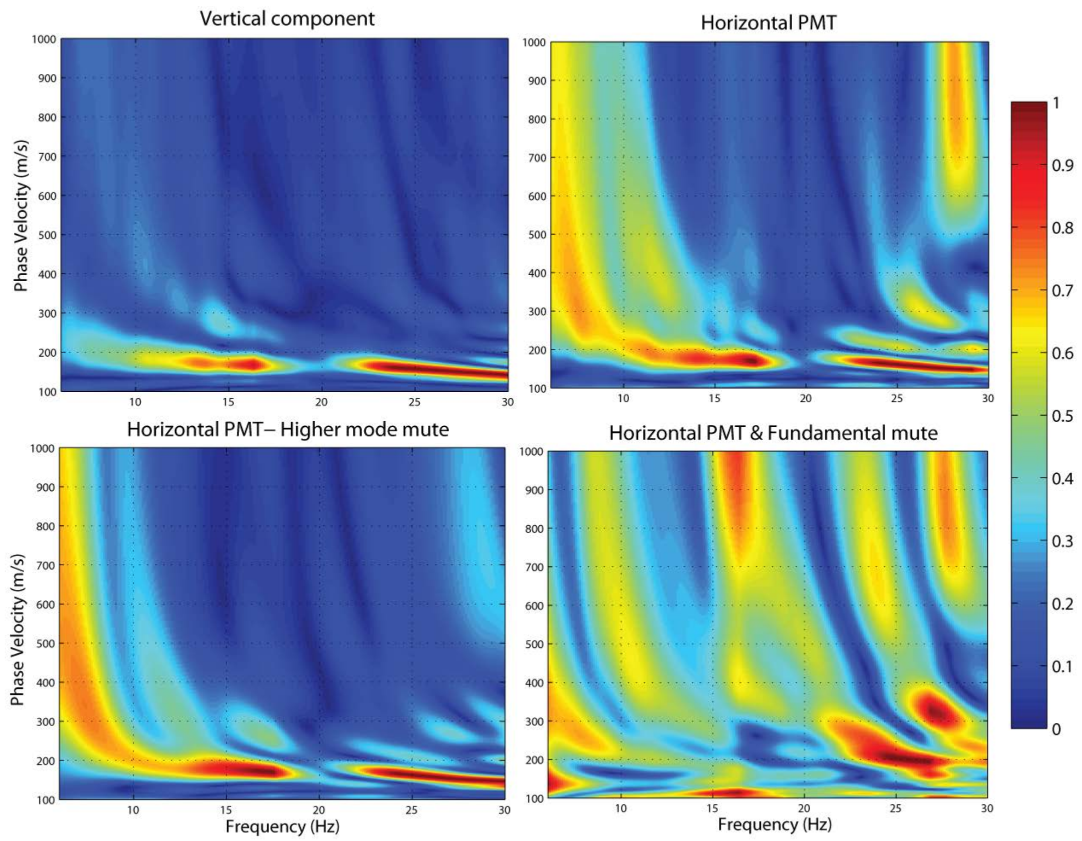

Figure 14 Top left: Vertical component dispersion plots (traditional MASW approach). Top right: Horizontal PMT dispersion plot. Bottom left: Horizontal PMT with the higher mode mute dispersion plot (MCASW approach). Bottom right: Horizontal PMT with fundamental mute dispersion plot.

By improving my confidence in dispersion curve picks to lower frequencies when compared to a single component approach, I can improve my confidence in estimating shear wave velocities to greater depths. To evaluate the benefit of utilizing multicomponent data, I compare dispersion curve picks derived from vertical component dispersion plots (traditional MASW approach) with the horizontal PMT and higher mode mute dispersion plot Ffigure 15). To provide an unbiased approach to dispersion plot picks, I calculate the highest amplitude of the non-gained dispersion plot (only accounting for cylindrical spreading) and determine the frequency at which the amplitude is 25% of the maximum. For the vertical only dispersion, picks are valid to as low as 11.4 Hz and the horizontal PMT with a higher mode mute (MCASW) to 6.5 Hz. After I pick the dispersion curve, I calculate a lower and upper dispersion curve bound that is at 85% of the maximum amplitude at every picked frequency. This is done to determine how the inverted shear wave velocity profile may vary and to give me an idea of confidence in shear wave velocity estimates at different depths.

I invert the dispersion curve picks and the 85% confidence (error) bounds for both the vertical component and MCASW results. Both were inverted using a 20 layer model with the depth to half space being dependent on the lowest frequency picked for the individual data sets. The vertical component, only picking down to 11.4 Hz, yields a maximum depth of 7 meters (Figure 15). The horizontal PMT and higher mode mute, with dispersion picks down to 6.5 Hz, yields a maximum depth of 32 meters, 4.5 times deeper than the vertical component alone (Figure 15). The error bounds on the vertical component velocity profile match very closely to the inverted picks. The MCASW lower bounds follow the same general trend as the inverted picks, with the difference in shear wave velocities increasing with depth, indicating less confidence with an increase in depth. The higher error bound does not follow the trend of the inverted picks due to the inversion being unable to converge (Figure 15). Comparing the vertical component and horizontal PMT with the polar mute derived velocity profiles, both follow the same general trend except in the upper 2.5 meters (Figure 15). This divergence is predominately due to a limitation of 20 layers in my inversion for both analyses. Therefore, the vertical only component, with a total depth of 7 meters, has much finer depth resolution in the upper few meters compared to the MCASW results, which has 20 layers spread over a total depth of 32 meters.

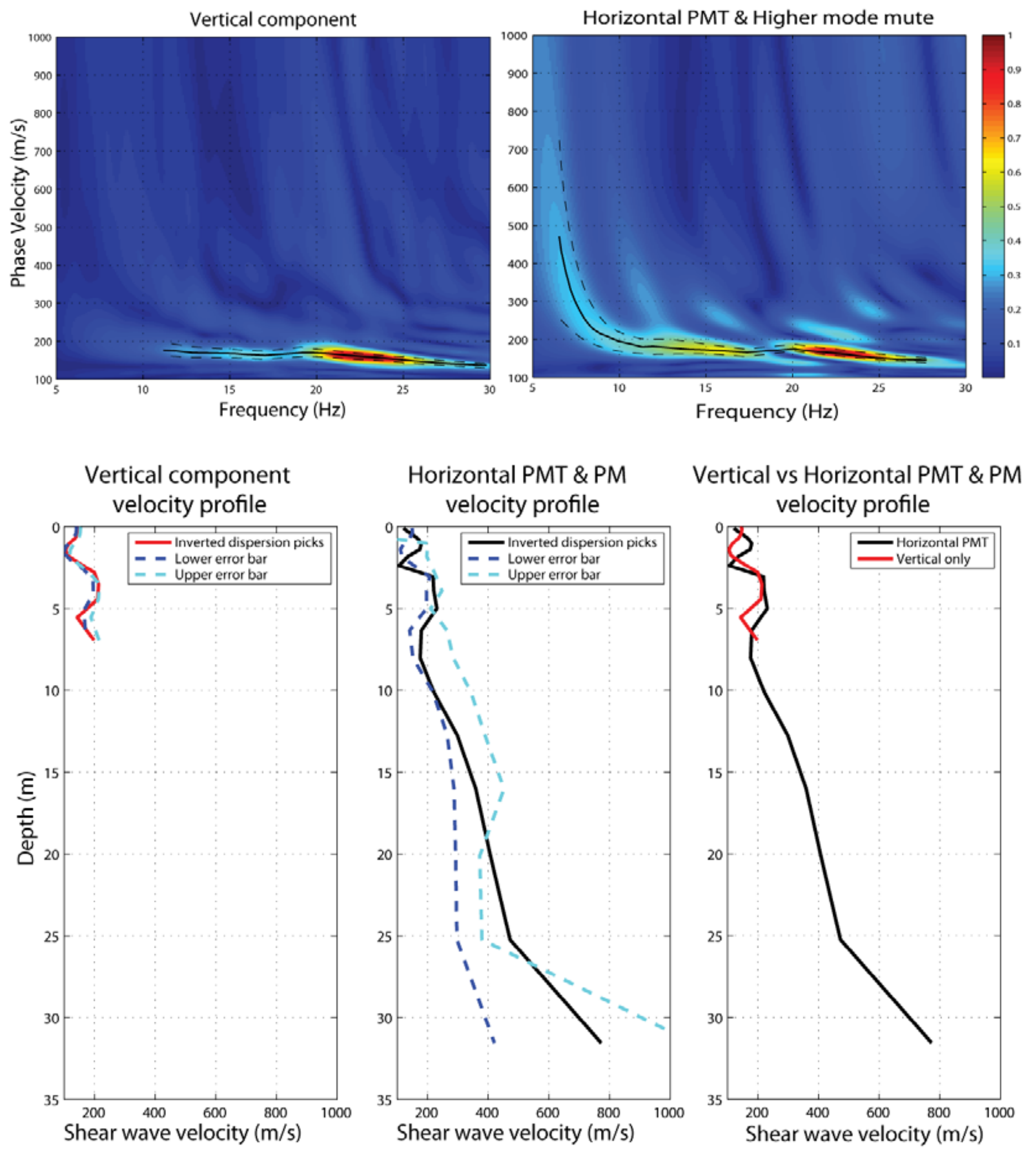

Figure 15 Top: Vertical component (MASW approach) and horizontal PMT with higher mode mute (MCASW approach) dispersion plots. Solid black line denotes dispersion curve picks for each and dashed black line are error bars that are at 85% of the picked amplitudes. Bottom: Velocity profiles from inverting dispersion curve picks and error bars. Solid red line denotes inverted vertical component picks, solid black from MCASW dispersion curve picks, dashed dark blue is from lower error bounds for the respective dispersion plots and dashed light blue line is from upper error bounds.

Multi-component multichannel analysis of surface waves (MCASW) provides for an improved semblance at lower frequencies. This can be seen by an increase in maximum semblance, a decrease in the width of the fundamental mode and a more distinctive fundamental envelope for both my synthetic and field example. This analysis enables me to more accurately estimate shear wave velocities to greater depths without having to increase my receiver aperture or source size. For field data, it also allows me to increase my signal to noise ratio without having to stack multiple shots together at a single source location, which would slow down acquisition in the field. Thus, the MCASW approach using a land streamer can result in improved shear wave velocity estimates with lower costs for data acquisition.

CHAPTER FIVE: CONLCUSIONS

The use of multi-component active source data allows for improved estimates of near surface shear wave velocity distributions to greater depths. Multi-Component Analysis of Surface Waves (MCASW) offers increased confidence in phase velocity vs frequency picks at lower frequencies over a single channel MASW approach, allowing for more accurate velocity estimates at greater depths. MCASW also allows me to determine elliptical particle motion direction to correctly identify fundamental and higher Rayleigh wave modes, leading to more confident shear wave velocity estimates. Multi-component data collection and analysis should be employed for geotechnical applications, as it offers greater versatility and data processing options for improved estimates of site specific seismic response. Future work will involve using multiple components to directly estimate site resonances, using an active source H/V approach. Further modeling will also be done to determine the benefits of multi-component Rayleigh wave analysis for a variety of geologic conditions.

REFERENCES

Aki, K., & Richards, P. G. (1980). Quantitative Seismology (2 volumes).

Black, R. A., Ralph, H., & Knupp, K. G. S. (1989). Sensitivity of near-surface shear-wave velocity determination from Rayleigh and Love waves.

Brune, J. N. (1970). Tectonic stress and the spectra of seismic shear waves from earthquakes. Journal of geophysical research, 75(26), 4997-5009.

Building Seismic Safety Council of the National Institute of Building Sciences (NIBS) , 2009, NEHRP recommended seismic provisions for new buildings and other structures (2009 edition), Federal Emergency Management Agency (FEMA)

Carr, B. J., Hajnal, Z., & Prugger, A. (1998). Shear-wave studies in glacial till. Geophysics, 63(4), 1273-1284.

Craig, R.F., 1992. Soil mechanics, 5th ed. Chapman & Hall, New York.

Dasios, A., McCann, C., Astin, T. R., McCann, D. M., & Fenning, P. (1999). Seismic imaging of the shallow subsurface: shear-wave case histories. Geophysical Prospecting, 47(4), 565-591.

Dorman, J., Ewing, M., 1962. Numerical inversion of seismic surface wave dispersion data and crust–mantle structure in the New York-Pennsylvania area. J. Geophys.Res.67,5227–5241.

Garland, G.D., 1979. Introduction to Geophysics: Mantle, Core and Crust, 2nd ed. W.B. Saunders, Philadelphia

Giorgis, S., Tikoff, B., Kelso, P., & Markley, M. (2006). The role of material anisotropy in the neotectonic extension of the western Idaho shear zone, McCall, Idaho. Geological Society of America Bulletin, 118(3-4), 259-273.

Gucunski, N., & Woods, R. D. (1991). Use of Rayleigh modes in interpretation of SASW test. In Second International Conference on Recent Advances in Geotechnical Earthquake Engineering and Soil Dynamics (1991: March 11-15; St. Louis, Missouri). Missouri S&T (formerly the University of Missouri--Rolla).

Hardage, B., DeAngelo, M., Murray, P., Sava, D., (2011). Multicomponent Seismic Technology. Society of Exploration Geophysicists

Hasbrouck, W.P., 1986. Hammer-impact, shear-wave studies. In: 254 Danbom, S.H., Domenico, S.N. (Eds.), Shear-Wave Exploration. Society of Exploration Geophysicists, pp. 97– 121.

Kramer, S. L. (1996). Geotechnical earthquake engineering (Vol. 80). Upper Saddle River, NJ: Prentice Hall.

Louie, J. N., 2001, Faster, Better: Shear-wave velocity to 100 meters from Refraction Microtremor analysis, Bulletin of the Seismological Society of America, 2001, vol. 91, no. 2 (April), p. 347-364.

Liberty, L.M. and Gribler, G., (2014). Shear wave seismic velocity profiling and depth to water table – earthquake site response measurements for Valley County, Idaho, CGISS technical report 14-01, 69 p.

Michaels, P., Smith, R. B., (1997). Surface Wave Inversion by Neural Networks (Radial Basis Functions) for Engineering Applications, 2(1), 65-76. doi: 10.4133/JEEG2.1.65

Mooney, H. M., & Bolt, B. A. (1966). Dispersive characteristics of the first three Rayleigh modes for a single surface layer. *Bulletin of the Seismological Society of America*, *56*(1), 43-67.

Nakamura, Y., 1989, A Method for Dynamic Characteristics Estimation of Subsurface using Microtremor on the Ground Surface, Quarterly report of Railway Technical Research Institute, v. 30

Park, C. B., Miller, R. D., & Xia, J. (1998, January). Imaging dispersion curves of surface waves on multi-channel record. In SEG Expanded Abstracts (Vol. 17, No. 1, pp. 1377-1380).

Park, C. B., Miller, R. D., & Xia, J. (1999). Multichannel analysis of surface waves. Geophysics, 64(3), 800-808.

Pelton, J. R. (2005). Near-surface seismology: Surface-based methods. 2005) Near-surface geophysics. Society of Exploration Geophysicists, Tulsa OK, 219-263.

Pickett, G. R. (1963). Acoustic character logs and their applications in formation evaluation. Journal of Petroleum technology, 15(06), 659-667.

Pugin, A.J.M., Larson, T.H., Sargent, S.L., McBride, J.H., and Bexfield, C.E., 2004, Near-surface mapping using SH-wave and P-wave seismic land-streamer data acquisition in Illinois, U.S.: The Leading Edge, v. 23, doi: 10.1190/1.1776740.

Rodriguez-Marek, A., Bray, J. D., & Abrahamson, N. A. (2001). An empirical geotechnical seismic site response procedure. Earthquake Spectra, 17(1), 65-87.

SESAME, 2004, Guidelines for the implementation of the H/V spectral ratio technique on ambient vibrations: Measurements, processing and interpretation, SESAME European research project WP12- Deliverable D23.12

Stephenson, W. J., Louie, J. N., Pullammanappallil, S., Williams, R. A., & Odum, J. K. (2005). Blind shear-wave velocity comparison of ReMi and MASW results with boreholes to 200 m in Santa Clara Valley: implications for earthquake ground-motion assessment. Bulletin of the Seismological Society of America, 95(6), 2506-2516.

Stokoe II, K.H., Nazarian, S., 1983. Effectiveness of ground improvement from Spectral Analysis of Surface Waves. Proceeding of the Eight European Conference on Soil Mechanics and Foundation Engineering, Helsinki, Finland.

Strutt, J. W. (Rayleigh, L.) (1885). On waves propagated along the plane surface of an elastic solid. Proceedings of the London Mathematical Society, 17, 4-1.

van der Veen, M., Spitzer, R., Green, A.G., and Wild, P., 2001, Design and application of a towed land-streamer system for cost-effective 2-D and pseudo-3-D shallow seismic data acquisition: GEOPHYSICS, v. 66, doi: 10.1190/1.1444939.

Xia, J., Miller, R. D., Park, C. B., & Tian, G. (2003). Inversion of high frequency surface waves with fundamental and higher modes. Journal of Applied Geophysics, 52(1), 45-57.

APPENDIX A

Model Input and Field Acquisition Parameters

Table A.1 Model input parameters

Receiver spacing	1 meter
Number of stations	99 stations
Source to first receiver distance	5 meters
Source frequency range	1- 100 Hz
Depth steps	0.5 meters
Sample rate	0.001 seconds

Table A.2 Field acquisition parameters

Receiver spacing	1 meter
Number of stations	48 stations
Source to first receiver distance	5 meters
Source	200 kg accelerated weight drop
Vertical geophones	4.5 Hz
Horizontal geophones	4.5 Hz
Sample rate	0.005 seconds

APPENDIX B

MCASW and Dispersion Generation Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%Rayleigh wave component rotation for field records
%
%Coded by Gabriel Gribler for M.S. in geophysics thesis
%
%Modified from code written by Matt Haney (4 April, 2014)
%
% Code reads in vertical and horizontal inline shot gathers to complete
% Multi-Component Analysis of Surface Waves (MCASW)

clear all
close all

% Input data

% read in vertical component SU file
[vert,SuTraceHeaders,SuHeader]=ReadSu('don_sch_v.su','endian','b');

% read in horizontal component SU file
[horz,SuTraceHeaders,SuHeader]=ReadSu('don_sch_r.su','endian','b');

vert=fliplr(vert);
horz=fliplr(horz);

rec=(2:48);                      %Stations to use in dispersion generation

vert=vert(:,rec);
horz=horz(:,rec);


% Input parameters from headers

[v1 v2] = size(vert);
dt=SuHeader.dt/1000000;                   %sample rate from header
nd=SuTraceHeaders.offset;                 %near offset (source-1st receiver)
from header
nd=5;
trace_spac=SuTraceHeaders(2).offset-SuTraceHeaders(1).offset;   %trace spacing
trace_spac=1;
noffs = ([0:(v2-1)]*trace_spac+nd)/1;      %offsets in meters for each trace



%Velocity range for dispersion curve generation
vmin = 100;       %m/s
vmax = 1500;      %m/s
dv = 1;           %m/s
vsv=[vmin:dv:vmax];
```

```
% autoset the frequency range based on Nyquist frequency and time length
Nyquist=1/(dt*2);
fmin = round(v1*dt*2);
df = Nyquist/1000;
fmax = Nyquist/10;
fmin=1;
fmax=50;
df=0.1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Applying a top mute to shot gathers

mut_min=200;                            %sample to mute at trace 1
mut_max=1400;                           %sample to mute at last trace
mut_slp=round((mut_max-mut_min)/v2);

mute=ones(v1,v2);

for x=1:v2;
    mut_ind(x)=mut_min+((x-1)*mut_slp);
   for y=1:v1;

       if y<mut_ind(x);
           mute(y,x)=0;
       else mute(y,x)=1;

       end

   end
end


vert=vert.*mute;                 %apply boolean mute matrix to shot gathers
horz=horz.*mute;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%Polar mute: determine particle motion to slectively mute out higher modes
%or fundamental mode
%
t=(1:1:v1);      %time sample vector
amp_th=0.1;      %amplitude threshold to mute low amplitude arrivals (optional)

for jj=1:v2;

    for ii=1:v1;

        Amp(ii,jj)=sqrt((vert(ii,jj)^2)+(horz(ii,jj)^2)); %calculate amplitude
        rad(ii,jj)=atan2(vert(ii,jj),horz(ii,jj));   %calculate phase
    end

Amp_n(:,jj)=Amp(:,jj)./max(Amp(:,jj));   %trace normalized amplitude
unrad(:,jj)=unwrap(rad(:,jj));           %unwrapped phase
coef=polyfit(t',unrad(:,jj),50);
unrad_fit(:,jj)=polyval(coef,t);         %fit polynomail to smooth phase

end

buf=1;      %gap in time samples to calculate unwraped phase slope

% Loop determines if slope is positive or negative and assigns a 1 or 0
% accordingly, to create a boolean mute matrix to be applied to the
% original shot gathers. > sign can be changes to < sign to turn this
% fundamental mute into higher mode mute

for jj=1:v2;

    for ii=1:(v1-buf);

        if unrad_fit(ii+buf,jj)>unrad_fit(ii,jj) %&& Amp_n(ii,jj)>amp_th
            pm(ii,jj)=1;

        else pm(ii,jj)=0;
        end

    end

    pm((v1-buf+1):v1,jj)=0;
end

pm=pm.*Amp_n;

vert=pm.*vert;
horz=pm.*horz;

% figure
% plot(unrad(:,40))
% hold on
% plot(unrad_fit(:,40))
%
% figure
% imagesc(pm)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Particle Motion Transforms (PMT)

for q=1:numel(horz(1,:));
for p=1:numel(horz(:,1));

rot_r(p,q)=sqrt((vert(p,q)^2)+(horz(p,q)^2));   %Vertical PMT
rot_p(p,q)=sqrt((vert(p,q)^2)+(horz(p,q)^2));   %Horizontal PMT


if horz(p,q)>=0
    rot_p(p,q)=rot_p(p,q);
else rot_p(p,q)=-rot_p(p,q);
end

if vert(p,q)>=0
    rot_r(p,q)=rot_r(p,q);
else rot_r(p,q)=-rot_r(p,q);
end

end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Vertical only dispersion plot creation


% round fmax up to a multiple of the desired frequency spacing
fmax = ceil(fmax/df)*df;
% round fmin down to a multiple of the desired frequency spacing
fmin = floor(fmin/df)*df;
% vector of frequencies
fsv=[fmin:df:fmax];

% length of data needed for desired frequency resolution
n = round(1/(df*dt)); % must be an integer

% cut out the frequencies desired, depending on even or odd samples
if (floor(n/2) == ceil(n/2))

    % for even samples
    n2=n/2;
    n2m1=(n/2)-1;
    nfrs = [-n2:n2m1]*(1/n2)*(1/(2*dt));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([vert(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            vf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        else
            dum = fftshift(fft([ vert(1:n,ii) ]));
            % cut out the part I want
            vf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        end

    end

else

    % for odd samples
    nm12 = (n-1)/2;
    nfrs = [-nm12:nm12]'*(1/nm12)*((1/(2*dt))-(1/(2*n*dt)));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([vert(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            vf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        else
            dum = fftshift(fft([ vert(1:n,ii) ]));
            % cut out the part I want
            vf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        end

    end

end

% make matrices that represent the frequency and offset grid
frqsv = repmat(fsv',1,v2);
xsv = repmat(noffs,length(fsv),1);

% form the phase velocity spectrum
for ii=1:length(vsv)
    vv = vsv(ii);

    % the whitening technique of Park et al. (1998), SEG abstract
    dfw_v(:,ii) = abs(sum((vf./abs(vf)).*exp(i*2*pi*frqsv.*(xsv/vv)),2));

    % geometrical spreading and no whitening
    dff_v(:,ii) = abs(sum(vf.*sqrt(xsv).*exp(i*2*pi*frqsv.*(xsv/vv)),2));
end

% Apply a dispersion domain AGC to gain low amplitude arrivals
```

```
f_buf=2;          % AGC window
fn_end=5;         % frequency to end gain at


for ii=(numel(fsv):-1:(find(fsv==fn_end)+f_buf/df));

    dfnv_v(ii,:)=dff_v(ii,:)./max(dff_v(ii,:));
    dfn_v(ii-(f_buf/df),:)=dff_v(ii-(f_buf/df),:)./...
        max(max(dff_v((ii:-1:(ii-(f_buf/df))),:)));

end


[v_dmax1 v_dmax2]= max(dff_v');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Radial only dispersion plot creation

% round fmax up to a multiple of the desired frequency spacing
fmax = ceil(fmax/df)*df;
% round fmin down to a multiple of the desired frequency spacing
fmin = floor(fmin/df)*df;
% vector of frequencies
fsv=[fmin:df:fmax];

% length of data needed for desired frequency resolution
n = round(1/(df*dt)); % must be an integer

% cut out the frequencies desired, depending on even or odd samples
if (floor(n/2) == ceil(n/2))

    % for even samples
    n2=n/2;
    n2m1=(n/2)-1;
    nfrs = [-n2:n2m1]*(1/n2)*(1/(2*dt));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([horz(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            hf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        else
            dum = fftshift(fft([horz(1:n,ii) ]));
            % cut out the part I want
            hf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        end

    end

else

    % for odd samples
    nm12 = (n-1)/2;
    nfrs = [-nm12:nm12]'*(1/nm12)*((1/(2*dt))-(1/(2*n*dt)));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([horz(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            hf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        else
            dum = fftshift(fft([ vert(1:n,ii) ]));
            % cut out the part I want
            hf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        end

    end

end

% make matrices that represent the frequency and offset grid
frqsv = repmat(fsv',1,v2);
xsv = repmat(noffs,length(fsv),1);

% form the phase velocity spectrum
for ii=1:length(vsv)
    vv = vsv(ii);

    % the whitening technique of Park et al. (1998), SEG abstract
    dfw_h(:,ii) = abs(sum((hf./abs(hf)).*exp(i*2*pi*frqsv.*(xsv/vv)),2));

    % geometrical spreading and no whitening
    dff_h(:,ii) = abs(sum(hf.*sqrt(xsv).*exp(i*2*pi*frqsv.*(xsv/vv)),2));
end


for ii=(numel(fsv):-1:(find(fsv==fn_end)+f_buf/df));

    dfnv_h(ii,:)=dff_h(ii,:)./max(dff_v(ii,:));
    dfn_h(ii-(f_buf/df),:)=dff_h(ii-(f_buf/df),:)./...
        max(max(dff_h((ii:-1:(ii-(f_buf/df))),:)));
```

```
end

h_dmax= max(dff_h');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Vertical PMT dispersion generation

% cut out the frequencies desired, depending on even or odd samples
if (floor(n/2) == ceil(n/2))

    % for even samples
    n2=n/2;
    n2m1=(n/2)-1;
    nfrs = [-n2:n2m1]*(1/n2)*(1/(2*dt));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([rot_r(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            rrf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        else
            dum = fftshift(fft([rot_r(1:n,ii) ]));
            % cut out the part I want
            rrf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        end

    end

else

    % for odd samples
    nm12 = (n-1)/2;
    nfrs = [-nm12:nm12]'*(1/nm12)*((1/(2*dt))-(1/(2*n*dt)));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([rot_r(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            rrf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        else
            dum = fftshift(fft([rot_r(1:n,ii) ]));
            % cut out the part I want
            rrf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        end

    end

end

% make matrices that represent the frequency and offset grid
frqsv = repmat(fsv',1,v2);
xsv = repmat(noffs,length(fsv),1);

% form the phase velocity spectrum
for ii=1:length(vsv)
    vv = vsv(ii);

    % the whitening technique of Park et al. (1998), SEG abstract
    dfw_rr(:,ii) = abs(sum((rrf./abs(rrf)).*exp(i*2*pi*frqsv.*(xsv/vv)),2));

    % geometrical spreading and no whitening
    dff_rr(:,ii) = abs(sum(rrf.*sqrt(xsv).*exp(i*2*pi*frqsv.*(xsv/vv)),2));
end


for ii=(numel(fsv):-1:(find(fsv==fn_end)+f_buf/df));

    dfnv_rr(ii,:)=dff_rr(ii,:)./max(dff_v(ii,:));
    dfn_rr(ii-(f_buf/df),:)=dff_rr(ii-(f_buf/df),:)./...
        max(max(dff_rr((ii:-1:(ii-(f_buf/df))),:)));

end

rr_dmax= max(dff_rr');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Horizontal PMT dispersion generation

% cut out the frequencies desired, depending on even or odd samples
if (floor(n/2) == ceil(n/2))

    % for even samples
    n2=n/2;
    n2m1=(n/2)-1;
    nfrs = [-n2:n2m1]*(1/n2)*(1/(2*dt));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([rot_p(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            rpf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        else
            dum = fftshift(fft([rot_p(1:n,ii) ]));
            % cut out the part I want
            rpf(:,ii) = dum((n2+1+(fmin/df)):(n2+1+(fmax/df)),1);
        end

    end

else

    % for odd samples
    nm12 = (n-1)/2;
    nfrs = [-nm12:nm12]'*(1/nm12)*((1/(2*dt))-(1/(2*n*dt)));

    % Fourier transform data which has been padded to obtain the desired
    % frequency resolution and then cut out data
    for ii=1:v2

        if (n > v1)
            dum = fftshift(fft([rot_p(:,ii) ; zeros(n-v1,1)]));
            % cut out the part I want
            rpf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        else
            dum = fftshift(fft([rot_p(1:n,ii) ]));
            % cut out the part I want
            rpf(:,ii) = dum((nm12+1+(fmin/df)):(nm12+1+(fmax/df)),1);
        end

    end
end

% make matrices that represent the frequency and offset grid
frqsv = repmat(fsv',1,v2);
xsv = repmat(noffs,length(fsv),1);

% form the phase velocity spectrum
for ii=1:length(vsv)
    vv = vsv(ii);

    % the whitening technique of Park et al. (1998), SEG abstract
    dfw_rp(:,ii) = abs(sum((rpf./abs(rpf)).*exp(i*2*pi*frqsv.*(xsv/vv)),2));

    % geometrical spreading and no whitening
    dff_rp(:,ii) = abs(sum(rpf.*sqrt(xsv).*exp(i*2*pi*frqsv.*(xsv/vv)),2));
end


for ii=(numel(fsv):-1:(find(fsv==fn_end)+f_buf/df));
```

```
    dfnv_rp(ii,:)=dff_rp(ii,:)./max(dff_v(ii,:));
    dfn_rp(ii-(f_buf/df),:)=dff_rp(ii-(f_buf/df),:)./...
        max(max(dff_rp((ii:-1:(ii-(f_buf/df))),:)));

end

rp_dmax= max(dff_rp');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Find max amplitude and determine at what frequency the amplitude falls
%below a certain amplitude threshold

for pp=1:numel(fsv);

max_amp_v(pp)=max(dff_v(pp,:));
max_amp_v_ind(pp)=find(dff_v(pp,:)==max_amp_v(pp));
max_amp_h(pp)=max(dff_h(pp,:));
max_amp_h_ind(pp)=find(dff_h(pp,:)==max_amp_h(pp));
max_amp_rr(pp)=max(dff_rr(pp,:));
max_amp_rr_ind(pp)=find(dff_rr(pp,:)==max_amp_rr(pp));
max_amp_rp(pp)=max(dff_rp(pp,:));
max_amp_rp_ind(pp)=find(dff_rp(pp,:)==max_amp_rp(pp));

end

ampcut=.25;           %amplitude threshold

max_amp_v_low=fsv(max(find(find(max_amp_v<=(ampcut*max(max_amp_v)))...
    <=find(max_amp_v==max(max_amp_v)))));
max_amp_h_low=fsv(max(find(find(max_amp_h<=(ampcut*max(max_amp_h)))...
    <=find(max_amp_h==max(max_amp_h)))));
max_amp_rr_low=fsv(max(find(find(max_amp_rr<=(ampcut*max(max_amp_rr)))...
    <=find(max_amp_rr==max(max_amp_rr)))));
max_amp_rp_low=fsv(max(find(find(max_amp_rp<=(ampcut*max(max_amp_rp)))...
    <=find(max_amp_rp==max(max_amp_rp)))));


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Dispersion picks for field records

vert_disp=[29.7984,  136.8852;
   27.6094,  141.4250;
   25.6125,  148.2346;
   23.4236,  157.3140;
   22.0795,  161.8537;
   20.6586,  167.5284;
   19.7561,  169.7982;
   19.1993,  169.7982;
   17.0296,  162.9887;
   15.9543,  165.2585;
   15.2631,  167.5284;
   12.8629,  169.7982;
   11.9220,  174.3380;
   11.1732,  176.6078];

pro_disp=[27.7247,  145.9647;
   26.0349,  148.2346;
   24.5373,  152.7743;
   23.0396,  160.7188;
   22.0987,  165.2585;
   21.0042,  169.7982;
   20.1018,  175.4729;
   17.3752,  166.3934;
   15.7047,  170.9332;
   14.9174,  173.2030;
   14.0534,  174.3380;
   12.8821,  178.8777;
   11.9220,  182.2825;
   11.3268,  180.6873;
   10.6356,  188.7667;
    9.9251,  195.8462;
    9.2915,  208.6003;
    8.8882,  215.6797;
    8.4658,  228.1084;
    8.2162,  241.2673;
    7.9666,  255.8310;
    7.7170,  275.3153;
    7.4866,  295.7440;
    7.1793,  330.9823;
    6.8913,  381.1097;
    6.6417,  433.1021;
    6.5265,  470.6028];


% vertical dispersion error bounds

err_b=0.85;      %percent of max amplitude to calculate error bars for

for pp=1:numel(vert_disp(:,1));

    dis=vert_disp;

    fd_ind(pp)=max(find(fsv<=dis(pp,1)));
    v_ind(pp)=max(find(vsv<=dis(pp,2)));

    dis_err=(find(dff_v(fd_ind(pp),:)>=(err_b*dff_v(fd_ind(pp),v_ind(pp)))));
    dis_jump=diff(dis_err);
    dis_jump_ind=find(dis_jump>1);
    val=dis_err(dis_jump_ind)-v_ind(pp);


if isempty(val);
            vert_err_low(pp)=vsv(min(dis_err));
            vert_err_high(pp)=vsv(max(dis_err));

  elseif numel(val)>=2

    for kk=1:numel(dis_jump_ind);

        if  val(kk)<0
            vert_err_low(pp)=vsv(dis_err(dis_jump_ind(kk)));


        elseif val(kk)>0

            vert_err_high(pp)=vsv(min(dis_err(dis_jump_ind)));
            vert_err_low(pp)=vsv(min(dis_err));

        end
    end

  elseif val<0
      vert_err_low(pp)=vsv(dis_err(dis_jump_ind));
      vert_err_high(pp)=max(vsv);

  elseif val>0
      vert_err_low(pp)=vsv(min(dis_err));
      vert_err_high(pp)=vsv(dis_err(dis_jump_ind));

  end
end

vert_err_low=horzcat(fsv(fd_ind)',vert_err_low');
vert_err_high=horzcat(fsv(fd_ind)',vert_err_high');

for pp=1:numel(pro_disp(:,1));

    dis=pro_disp;

    fd_ind(pp)=max(find(fsv<=dis(pp,1)));
    v_ind(pp)=max(find(vsv<=dis(pp,2)));

    dis_err=(find(dff_rp(fd_ind(pp),:)>=(err_b*dff_rp(fd_ind(pp),v_ind(pp)))));
    dis_jump=diff(dis_err);
    dis_jump_ind=find(dis_jump>1);
    val=dis_err(dis_jump_ind)-v_ind(pp);
  if isempty(val);
            pmt_err_low(pp)=vsv(min(dis_err));
            pmt_err_high(pp)=vsv(max(dis_err));

  elseif numel(val)>=2

    for kk=1:numel(dis_jump_ind);

        if  val(kk)<0
            pmt_err_low(pp)=vsv(dis_err(dis_jump_ind(kk)));


        elseif val(kk)>0

            pmt_err_high(pp)=vsv(min(dis_err(dis_jump_ind)));
            pmt_err_low(pp)=vsv(min(dis_err));

        end
    end

  elseif val<0
      pmt_err_low(pp)=vsv(dis_err(dis_jump_ind));
      pmt_err_high(pp)=max(vsv);

  elseif val>0
      pmt_err_low(pp)=vsv(min(dis_err));
      pmt_err_high(pp)=vsv(dis_err(dis_jump_ind));

  end

end


pmt_err_low=horzcat(fsv(fd_ind)',pmt_err_low');
pmt_err_high=horzcat(fsv(fd_ind)',pmt_err_high');




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Inverted shear wave velocity profiles derived from the previously picked
%dispersion plots and the high and low error bounds. Inversion was done
%within SurfSeis.

vert_profile=load('vert_don(Vs).txt');
vert_low=load('vert_don_low(Vs).txt');
vert_high=load('vert_don_high(Vs).txt');
pmt_profile=load('pmt_don(Vs).txt');
```

```
pmt_low=load('pmt_don_low(Vs).txt');
pmt_high=load('pmt_don_high(Vs).txt');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plots

%source power spectra

% figure
% plot(fsv,(abs(vf(:,2))),'linewidth',3)
% ylabel('Amplitude','FontSize',14,'Fontweight','bold')
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold')
% title('Amplitude spectrum of near trace','FontSize',14,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on

% Vertical shot gather
% figure
% hold on
% t=(1:1:v1);
% for k=1:v2
%  tr=vert(:,k);
%  tr=tr./max(tr);
%   plot(tr+(k+6),t.*dt,'k')
% end
% view(0,-90)
% axis([6 53 0 1])
% ylabel('Travel time (sec)','FontSize',14,'Fontweight','bold')
% xlabel('Offset (m)','FontSize',14,'Fontweight','bold')
% title({'Horizontal component shot gather' 'Top Mute & Polar
Mute'},'FontSize',14,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')


%Vertical dispersion plot
% figure
% imagesc(fsv,vsv,transpose(dff_v)); axis xy; axis([ fmin fmax vmin vmax ]);
% hold on
% plot(vert_disp(:,1),vert_disp(:,2),'k','linewidth',2)
% hold on
% plot(vert_err_low(:,1),vert_err_low(:,2),'k--
',vert_err_high(:,1),vert_err_high(:,2),'k--')
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold');
% ylabel('Phase Velocity (m/s)','FontSize',16,'Fontweight','bold');
% title('Vertical component','FontSize',20,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on
% axis([fn_end,30,100,1000])
%caxis([0 11E8])
```

```
% Horizontal (radial inline) dispersion plot
% figure
% imagesc(fsv,vsv,transpose(dfn_h)); axis xy; axis([ fmin fmax vmin vmax ]);
% hold on
% %plot(c1(:,1),c1(:,2),'k','linewidth',2)
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold');
% ylabel('Phase Velocity (m/s)','FontSize',16,'Fontweight','bold');
% title('Horizontal component','FontSize',20,'Fontweight','bold')
% axis([fn_end,30,100,1000])

% Vertical PMT dispersion plot
% figure
% imagesc(fsv,vsv,transpose(dfn_rr)); axis xy; axis([ fmin fmax vmin vmax ]);
% hold on
% plot(c1(:,1),c1(:,2),'k','linewidth',2)
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold');
% ylabel('Phase Velocity (m/s)','FontSize',16,'Fontweight','bold');
% title('Vertical PMT','FontSize',20,'Fontweight','bold')
% axis([fn_end,30,100,1000])

% Horizontal PMT dispersion plot
% figure
% imagesc(fsv,vsv,transpose(dff_rp)); axis xy; axis([ fmin fmax vmin vmax ]);
% hold on
% plot(pro_disp(:,1),pro_disp(:,2),'k','linewidth',2)
% hold on
% plot(pmt_err_low(:,1),pmt_err_low(:,2),'k--
',pmt_err_high(:,1),pmt_err_high(:,2),'k--')
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold');
% ylabel('Phase Velocity (m/s)','FontSize',16,'Fontweight','bold');
% title('Horizontal PMT & Higher mode mute','FontSize',20,'Fontweight','bold')
% axis([fn_end,30,100,1000])
%caxis([0 11E8])

%Difference between retrograde and prograde dispersion plots
% figure
% imagesc(fsv,vsv,transpose(diff)); axis xy; axis([ fmin fmax vmin vmax ]);
% hold on
% set(gca,'Fontsize',12,'Fontweight','bold')
% grid on
% xlabel('Frequency (Hz)','FontSize',14,'Fontweight','bold');
% ylabel('Phase Velocity (m/s)','FontSize',16,'Fontweight','bold');
% title('Retrograde - Prograde (difference)','FontSize',20,'Fontweight','bold')
% Frequency cross sections through dispersion curves
% f_cr=[9];
%
% conf_max=0.85;
% fund_max=1000;
% maxind=find(vsv==fund_max);
% for p=1:numel(f_cr)
%
%     f_ind(p)=find(fsv==f_cr(p));
%    %c_ind(p)=min(find(c1(:,1)>=f_cr(p)));
%    f_ind(p)=find(fsv==f_cr(p));
%
%    max_amp_v=max(dff_v(f_ind(p),(1:maxind)));
%    conf1_v=max(find(dff_v(f_ind(p),(1:maxind))>conf_max*max_amp_v));
%    conf2_v=max(find(dff_v(f_ind(p),(1:conf1_v-1))<conf_max*max_amp_v));
%    conf_v(p)=vsv(conf1_v)-vsv(conf2_v);
%
% %       max_amp_v=max(dfw_v(f_ind(p),(1:maxind)));
% %    conf1_v=max(find(dfw_v(f_ind(p),(1:maxind))>conf_max*max_amp_v));
% %    conf2_v=max(find(dfw_v(f_ind(p),(1:conf1_v-1))<conf_max*max_amp_v));
% %    conf_v(p)=vsv(conf1_v)-vsv(conf2_v);
%
%    max_amp_h=max(dff_h(f_ind(p),(1:maxind)));
%    conf1_h=max(find(dff_h(f_ind(p),(1:maxind))>conf_max*max_amp_h));
%    conf2_h=max(find(dff_h(f_ind(p),(1:conf1_h-1))<conf_max*max_amp_h));
%    conf_h(p)=vsv(conf1_h)-vsv(conf2_h);
%
%
% %    max_amp_h=max(dfw_h(f_ind(p),(1:maxind)));
% %    conf1_h=max(find(dfw_h(f_ind(p),(1:maxind))>conf_max*max_amp_h));
% %    conf2_h=max(find(dfw_h(f_ind(p),(1:conf1_h-1))<conf_max*max_amp_h));
% %    conf_h(p)=vsv(conf1_h)-vsv(conf2_h);
%
%    max_amp_rp=max(dff_rp(f_ind(p),(1:maxind)));
%    conf1_rp=max(find(dff_rp(f_ind(p),(1:maxind))>conf_max*max_amp_rp));
%    conf2_rp=max(find(dff_rp(f_ind(p),(1:conf1_rp-1))<conf_max*max_amp_rp));
%    conf_rp(p)=vsv(conf1_rp)-vsv(conf2_rp);
%
% %    max_amp_rp=max(dfw_rp(f_ind(p),(1:maxind)));
% %    conf1_rp=max(find(dfw_rp(f_ind(p),(1:maxind))>conf_max*max_amp_rp));
% %    conf2_rp=max(find(dfw_rp(f_ind(p),(1:conf1_rp-1))<conf_max*max_amp_rp));
% %    conf_rp(p)=vsv(conf1_rp)-vsv(conf2_rp);
%
%    max_amp_rr=max(dff_rr(f_ind(p),(1:maxind)));
%    conf1_rr=max(find(dff_rr(f_ind(p),(1:maxind))>conf_max*max_amp_rr));
%    conf2_rr=max(find(dff_rr(f_ind(p),(1:conf1_rr-1))<conf_max*max_amp_rr));
%    conf_rr(p)=vsv(conf1_rr)-vsv(conf2_rr);
%
```

```
% %    max_amp_rr=max(dfw_rr(f_ind(p),(1:maxind)));
% %    conf1_rr=max(find(dfw_rr(f_ind(p),(1:maxind))>conf_max*max_amp_rr));
% %    conf2_rr=max(find(dfw_rr(f_ind(p),(1:conf1_rr-1))<conf_max*max_amp_rr));
% %    conf_rr(p)=vsv(conf1_rr)-vsv(conf2_rr);
%
%
%
%    figure
%
%
plot(vsv(1:901),dff_v(f_ind(p),(1:901)),'g',vsv(1:901),dff_h(f_ind(p),(1:901)),
'b',...
%
vsv(1:901),dff_rp(f_ind(p),(1:901)),'r',vsv(1:901),dff_rr(f_ind(p),(1:901)),'c'
,'linewidth',2)
%     hold on
%
plot(vsv(conf2_v:conf1_v),conf_max*max_amp_v*ones(numel(conf2_v:conf1_v),1),'g'
,'linewidth',2)
%     hold on
%
plot(vsv(conf2_h:conf1_h),conf_max*max_amp_h*ones(numel(conf2_h:conf1_h),1),'b'
,'linewidth',2)
%     hold on
%
plot(vsv(conf2_rp:conf1_rp),conf_max*max_amp_rp*ones(numel(conf2_rp:conf1_rp),1
),'r','linewidth',2)
%     hold on
%
plot(vsv(conf2_rr:conf1_rr),conf_max*max_amp_rr*ones(numel(conf2_rr:conf1_rr),1
),'c','linewidth',2)
%
%     xlabel('Phase velocity','FontSize',18,'Fontweight','bold')
%     ylabel('Semblance amplitude','FontSize',18,'Fontweight','bold')
%     title(sprintf('%g Hz cross-
section',f_cr(p)),'FontSize',20,'Fontweight','bold')
%     set(gca,'Fontsize',12,'Fontweight','bold')
%     %leg=legend(sprintf('Vertical (%d m/s)',conf_v(p)),sprintf('Horizontal
(%d m/s)',conf_h(p)),...
%       sprintf('Prograde rot. (%d m/s)',conf_rp(p)),sprintf('Retrograde rot.
(%d m/s)',conf_rr(p)));
%     %set(leg,'fontsize',14,'fontweight','bold')
%     grid on
%
% end

% Vertical component velocity profile
% figure('PaperPosition',[0.25 0.25 4 8])
% plot(vert_profile(:,3),abs(vert_profile(:,2)),'r',vert_low(:,3),...
%  abs(vert_low(:,2)),'b--',vert_high(:,3),abs(vert_high(:,2)),'c--
','linewidth',2)
% axis([100,1000,0,35])
% view(0,-90)
% grid on
% xlabel('Shear wave velocity (m/s)','FontSize',18,'Fontweight','bold')
% ylabel('Depth (m)','FontSize',18,'Fontweight','bold')
% title({'Vertical component' 'velocity
profile'},'FontSize',20,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')
% leg=legend('Inverted dispersion picks','Lower error bar','Upper error bar');
% set(leg,'fontsize',10,'fontweight','bold','location','northeast')

%Horizontal PMT & PM velocity profile
% figure('PaperPosition',[0.25 0.25 4 8])
%
plot(pmt_profile(:,3),abs(pmt_profile(:,2)),'k',pmt_low(:,3),abs(pmt_low(:,2)),
'b--',...
%   pmt_high(:,3),abs(pmt_high(:,2)),'c--','linewidth',2)
% axis([100,1000,0,35])
% view(0,-90)
% grid on
% xlabel('Shear wave velocity (m/s)','FontSize',18,'Fontweight','bold')
% ylabel('Depth (m)','FontSize',18,'Fontweight','bold')
% title({'Horizontal PMT & PM' 'velocity
profile'},'FontSize',20,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')
% leg=legend('Inverted dispersion picks','Lower error bar','Upper error bar');
% set(leg,'fontsize',10,'fontweight','bold','location','northeast')

% Vertical vs Horizontal PMT & PM velocity profile
% figure('PaperPosition',[0.25 0.25 4 8])
%
plot(pmt_profile(:,3),abs(pmt_profile(:,2)),'k',vert_profile(:,3),abs(vert_prof
ile(:,2)),'r','linewidth',2)
% axis([100,1000,0,35])
% view(0,-90)
% grid on
% xlabel('Shear wave velocity (m/s)','FontSize',18,'Fontweight','bold')
% ylabel('Depth (m)','FontSize',18,'Fontweight','bold')
% title({'Vertical vs Horizontal PMT & PM' 'velocity
profile'},'FontSize',20,'Fontweight','bold')
% set(gca,'Fontsize',12,'Fontweight','bold')
% leg=legend('Horizontal PMT','Vertical only');
% set(leg,'fontsize',10,'fontweight','bold','location','northeast')
```

APPENDIX C

ReadSu Code

```
% ReadSu : Reads a SU formatted file (Seismic Unix)
%
% Call :
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename);
%
% To read in big endian format (default):
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'endian','b');
% To read in little endian format :
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'endian','l');
%
%
% To read in trace data as 'int32' :
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'DataFormat','int32');
% To read time slice 0.5<t<5 :
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'trange',.5,3);
% Skip every 5th trace :
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'jump',5);
% Read data in a CDP header range : 5000<cdp<5800
% (change cdp to any other valid TraceHeader value)
% [Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'minmax','cdp'5000,5800);
%
% Combine any combination of the above
%
[Data,SuTraceHeaders,SuHeader]=ReadSu(filename,'jump',1,'minmax','cdp',5300,540
0);
%
%



%
% (C) 2001-2004 Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
%    This program is free software; you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by
%    the Free Software Foundation; either version 2 of the License, or
%    (at your option) any later version.
%
%    This program is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of
%    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
%    GNU General Public License for more details.
%
%    You should have received a copy of the GNU General Public License
%    along with this program; if not, write to the Free Software
%    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
%
%
%

%
```

```
% 0.1 : INitial Release
% 0.2 : Added SkipData var, to skip reading of data.
% 0.3 : May 01, 2002
%       Added ability to read in ever 'jump' traces.
%       Added ability to read in time range.
%       Added abiliy to read in header range (ex. mincdp to maxcdp).
%

function [Data,SuTraceHeaders,SuHeader,HeaderInfo]=ReadSu(filename,varargin);

dsf=[];

if ~(exist(filename)==2'),
  SegymatVerbose([mfilename,' : ', filename,' does not exist !'])
  Data=[];SuTraceHeaders=[];SuHeader=[];HeaderInfo=[];
  return
end


% SET THE NEXT FLAG TO 1, IF YOU ARE SURE ALL TRACES HAVE THE SAME LENGT
% THIS WILL GREATLT INCREASE READING WHEN USING 'JUMP'
FixedTraceLength=1;

% NEXT TWO LINES TO ENUSRE THAT VARARGIN CAN BE PASSED TO FUNCTION
if nargin==2
    % CALL USING VARARGIN
    local_nargin=1+length(varargin{1});
    varargin=varargin{1};
else
    % DIRECT CALL
    local_nargin=length(varargin);
end


%
% HERE SHOULD BE A CALL TO A DEFAULT SEGY HEADER !!!!
%

SegyHeader.SegyFormatRevisionNumber=1;
SegyHeader.DataSampleFormat=5;
SegyHeader.Rev=GetSegyHeaderBasics;

% TRANSFORM VARARGING INTO PARAMETERS
cargin=1;
while (cargin<local_nargin)

   if strcmp(varargin{cargin},'endian')
       cargin=cargin+1;
       eval(['endian=char(varargin{cargin});'])
       if strcmp(endian,'b'),SegymatVerbose(['Reading BIG ENDIAN STYLE']);end
       if strcmp(endian,'l'),SegymatVerbose(['Reading LITTLE ENDIAN
STYLE']);end
    end

    if strcmp(varargin{cargin},'DataFormat')
       cargin=cargin+1;
       eval(['DataFormat=char(varargin{cargin});'])

       if strcmp(DataFormat,'float32'),
                SegyHeader.DataSampleFormat=5; % IEEE
       end

       if strcmp(DataFormat,'int32'),
                SegyHeader.DataSampleFormat=2; % 4 Byte, two's
                                          % complement integer
       end
       if strcmp(DataFormat,'int16'),
            SegyHeader.DataSampleFormat=3; % 2 Byte, two's
                                          % complement integer
       end
       if strcmp(DataFormat,'int8'),
            SegyHeader.DataSampleFormat=8; % 2 Byte, two's
                                          % complement integer
       end
       SegymatVerbose(['DataFormat : ',num2str(SegyHeader.DataSampleFormat)])
    end

    if strcmp(varargin{cargin},'dsf')
      cargin=cargin+1;
      eval(['dsf=',num2str(varargin{cargin}),';']);
      SegymatVerbose(['USING Data Sample Format : dsf=',num2str(dsf)])
    end

    if strcmp(varargin{cargin},'jump')
       cargin=cargin+1;
       eval(['jump=',num2str(varargin{cargin}),';']);
       SegymatVerbose(['JUMP : Read only every ',num2str(jump),'th trace'])
    end

    if strcmp(varargin{cargin},'minmax')
       cargin=cargin+1;
       eval(['header=''',varargin{cargin},''';']);
       cargin=cargin+1;
       eval(['headermin=',num2str(varargin{cargin}),';']);
       cargin=cargin+1;
       eval(['headermax=',num2str(varargin{cargin}),';']);
       SegymatVerbose(['MIN MAX : Using header ',header,' from
',num2str(headermin),' to ',num2str(headermax)])
    end
    if strcmp(varargin{cargin},'trange')
       cargin=cargin+1;
       eval(['tmin=',num2str(varargin{cargin}),';']);
       cargin=cargin+1;
       eval(['tmax=',num2str(varargin{cargin}),';']);
       SegymatVerbose(['TRANGE : tmin=',num2str(tmin),' tmax=',num2str(tmax)])
    end

    cargin=cargin+1;

end



%
% MAYBE DATA FORMATS CAN VARY ?
% but for now we use float32 if nothing else is set
%
if exist('SegyHeader')==0,
  SegyHeader.DataSampleFormat=5; % IEEE
end

if isempty(dsf)==0,
    SegyHeader.DataSampleFormat=dsf;
end

%
% SU DATA CAN BE EITHER BIG OR SMALL ENDIAN
% DEFAULT IS BIG ENDIAN
%

if exist('endian')==0,
  segyid = fopen(filename,'r');   % USE LOCAL BUTE ORDER AS DEFAULT
else
  segyid = fopen(filename,'r',endian);
end

if ~(exist(filename)==2'),
  SegymatVerbose([mfilename,' : ', filename,' does not exist !'])
  Data=[];SegyTraceHeaders=[];SegyHeader=[];
  return
end




if exist('SkipData')==0,
    SkipData=0; % [0] READ ONLY HEADER VALUES, [1] READ IN ALL DATA
end
```

```
if SkipData==1, SegymatVerbose(['Not reading data - headers only']), end

% SEGY HEADER FORMAT INFO
Revision=SegyHeader.SegyFormatRevisionNumber;
if Revision>0, Revision=1; end
Format=SegyHeader.Rev(Revision+1).DataSampleFormat(SegyHeader.DataSampleFormat)
.name;
BPS=SegyHeader.Rev(Revision+1).DataSampleFormat(SegyHeader.DataSampleFormat).bp
s;
SegymatVerbose([mfilename,' : ',Format])

% GET SIZE OF FILE

fseek(segyid,0,'eof'); DataEnd=ftell(segyid);
fseek(segyid,0,'bof');       % Go to the beginning of the file

%txt=['SegyRevision ',sprintf('%0.4g',Revision),',
',Format,'(',num2str(SegyHeader.DataSampleFormat),')'];
txt=[Format,'(',num2str(SegyHeader.DataSampleFormat),')'];


hw=waitbar(0,['Reading SU-file -',txt]);
traceinfile=0;
outtrace=0;
existJump=exist('jump');
existHeader=exist('header');
existtmin=exist('tmin');
existtmmax=exist('tmax');

tic;
while (~(ftell(segyid)>=DataEnd))
  traceinfile=traceinfile+1;
  usetrace=1;

  %if exist('jump')
  if existJump==1;
    if (traceinfile/jump)~=round(traceinfile/jump), usetrace=0; end
  end

  if ((usetrace==0)&(FixedTraceLength==1)&(exist('ns')==1))
    skip=240+(SegyHeader.BytesPerSample/8)*ns;
    fseek(segyid,skip,'cof');
  else
    % Read Trace Header
    SingleSuTraceHeader=GetSegyTraceHeader(segyid);

    ns=SingleSuTraceHeader.ns;

    % IF HEADER MIN MAX HAS BEEN CHOSEN, THEN CHECK THAT TRACE IS GOOD ENOUGH
    if ((existHeader==1)&(usetrace==1))
```

```
      headervalue=getfield(SingleSuTraceHeader,header);
       if ((headervalue<headermin)|(headervalue>headermax))
           usetrace=0;
       end
    end

    if usetrace==1;
      % Read Segy Trace Data
      SkipData=0;
    else
      SkipData=1;
    end


SingleSuData.data=GetSegyTraceData(segyid,SingleSuTraceHeader.ns,SegyHeader,Ski
pData);

  end % END READ TRACE OR NO FIXED LENGTH

  if (usetrace==1)
    % IF TIME RANGE IS SPECIFIED, THEN EXTRACT THIS
    if (existtmin)&(existtmax)
          % NEXT LINE SHOULD CONSIDER THAT ns in Trace and Segy Header could
vary !!!
          origtrange=[1:1:SingleSuTraceHeader.ns].*SingleSuTraceHeader.dt.*1e-
6+SingleSuTraceHeader.DelayRecordingTime;
          gooddata=find(origtrange>tmin & origtrange<tmax);
          SingleSuData.data=SingleSuData.data(gooddata);
          % CHECK NEXT LINE TAHT DelatRec... is in micro seconds
          SingleSuTraceHeader.DelayRecordingTime=tmin;
          SingleSuTraceHeader.ns=length(gooddata);
          ns=length(gooddata); %  for use below
    end
    outtrace=outtrace+1;
    SuTraceHeaders(outtrace)=SingleSuTraceHeader;
    SuData(outtrace).data=SingleSuData.data;

    %    keyboard
  end


  waitbar(ftell(segyid)/DataEnd,hw);
end
close(hw)
SegymatVerbose([mfilename,' : Elapsed time ',num2str(toc)]);

ns=SuTraceHeaders(1).ns;
dt=SuTraceHeaders(1).dt;
nt=length(SuData);
```

```
% DefaultSegyHeader
SuHeader=SegyHeader;
SuHeader.ns=ns;
SuHeader.nsOrig=ns;
SuHeader.dt=dt;
SuHeader.dtOrig=dt;
SuHeader.FixedLengthTraceFlag=1; % CHECK THAT THIS IS TRUE
SuHeader.SegyFormatRevisionNumber=1;
SuHeader.NumberOfExtTextualHeaders=0;


Data=zeros(ns,nt);
try
  for it=1:nt
    Data(:,it)=SuData(it).data;
  end
catch
  SegymatVerbose([mfilename,' Could not collect data in one matrix - check byte
order'])
end

[HeaderInfo]=TraceheaderToInfo(SuTraceHeaders);
return
SegymatVerbose([mfilename,' : Elapsed time ',num2str(toc)]);

ns=SuTraceHeaders(1).ns;
dt=SuTraceHeaders(1).dt;
nt=length(SuData);

% DefaultSegyHeader
SuHeader=SegyHeader;
SuHeader.ns=ns;
SuHeader.nsOrig=ns;
SuHeader.dt=dt;
SuHeader.dtOrig=dt;
SuHeader.FixedLengthTraceFlag=1; % CHECK THAT THIS IS TRUE
SuHeader.SegyFormatRevisionNumber=1;
SuHeader.NumberOfExtTextualHeaders=0;


Data=zeros(ns,nt);
try
  for it=1:nt
    Data(:,it)=SuData(it).data;
  end
catch
  SegymatVerbose([mfilename,' Could not collect data in one matrix - check byte
order'])
end
```

```
[HeaderInfo]=TraceheaderToInfo(SuTraceHeaders);
return
```

APPENDIX D

SegymatVerbose Code

```
% SegymatVerbose : Writes out verbose information to the screen
%
%
% Call :
%   SegymatVerbose(text,verboselevel)
%   prints out ‘text’ to screen if verboselevel is higher than threshold
%   set in m-file.
%

%
% © 2001-2004, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
%    This program is free software; you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by
%    the Free Software Foundation; either version 2 of the License, or
%    (at your option) any later version.
%
%    This program is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of
%    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
%    GNU General Public License for more details.
%
%    You should have received a copy of the GNU General Public License
%    along with this program; if not, write to the Free Software
%    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA


function SegymatVerbose(txt,level)

  if nargin==0, return, end
  if nargin==1,
    level=0;
  end

  VerboseLevel=1;

  % Only print information if at or above VerboseLevel
  if level<=VerboseLevel
    disp(75print(‘%s  : %s’,’SegyMAT’,txt))
  end
```

APPENDIX E

GetSegyTraceHeader Code

```
% GetSegyTraceHeader : Reads a seg y trace, data and header
%
% [SegyTraceHeader]=GetSegyTraceHeader(segyid,TraceStart,DataFormat,ns);
%
%
% (C) 2001-2004 Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
%    This program is free software; you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by
%    the Free Software Foundation; either version 2 of the License, or
%    (at your option) any later version.
%
%    This program is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of
%    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
%    GNU General Public License for more details.
%
%    You should have received a copy of the GNU General Public License
%    along with this program; if not, write to the Free Software
%    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
%

function
[SegyTraceHeader]=GetSegyTraceHeader(segyid,TraceStart,DataFormat,ns,SegyTraceH
eader);

if exist('DataFormat')==0, DataFormat='float32'; end
if exist('TraceStart')==0, TraceStart=ftell(segyid); end

if exist('SegyTraceHeader')
    if isempty('SegyTraceHeader');
        clear SegyTraceHeader;
    end
end


fseek(segyid,TraceStart,'bof');

% GET POSITION FOR EASY LATER LOCALIZATION
SegyTraceHeader.SegyMAT_TraceStart = ftell(segyid);

SegyTraceHeader.TraceSequenceLine=fread(segyid,1,'int32');    % 0
SegyTraceHeader.TraceSequenceFile=fread(segyid,1,'int32');    % 4
SegyTraceHeader.FieldRecord=fread(segyid,1,'int32');          % 8
SegyTraceHeader.TraceNumber=fread(segyid,1,'int32');          % 12
SegyTraceHeader.EnergySourcePoint=fread(segyid,1,'int32');    % 16
SegyTraceHeader.cdp=fread(segyid,1,'int32');                  % 20
SegyTraceHeader.cdpTrace=fread(segyid,1,'int32');             % 24

SegyTraceHeader.TraceIdenitifactionCode=fread(segyid,1,'int16'); % 28
SegyTraceHeader.NSummedTraces=fread(segyid,1,'int16'); % 30
SegyTraceHeader.NStackedTraces=fread(segyid,1,'int16'); % 32
SegyTraceHeader.DataUse=fread(segyid,1,'int16'); % 34
SegyTraceHeader.offset=fread(segyid,1,'int32');             %36

SegyTraceHeader.ReceiverGroupElevation=fread(segyid,1,'int32');             %40
SegyTraceHeader.SourceSurfaceElevation=fread(segyid,1,'int32');             %44
SegyTraceHeader.SourceDepth=fread(segyid,1,'int32');             %48
SegyTraceHeader.ReceiverDatumElevation=fread(segyid,1,'int32');             %52
SegyTraceHeader.SourceDatumElevation=fread(segyid,1,'int32');             %56
SegyTraceHeader.SourceWaterDepth=fread(segyid,1,'int32');  %60
SegyTraceHeader.GroupWaterDepth=fread(segyid,1,'int32');  %64
SegyTraceHeader.ElevationScalar=fread(segyid,1,'int16');  %68

% Multiply/divide next number for following 4 values
SegyTraceHeader.SourceGroupScalar=fread(segyid,1,'int16');  %70
SegyTraceHeader.SourceX=fread(segyid,1,'int32');  %72
SegyTraceHeader.SourceY=fread(segyid,1,'int32');  %76
SegyTraceHeader.GroupX=fread(segyid,1,'int32');  %80
SegyTraceHeader.GroupY=fread(segyid,1,'int32');  %84

SegyTraceHeader.CoordinateUnits=fread(segyid,1,'int16');  %88
SegyTraceHeader.WeatheringVelocity=fread(segyid,1,'int16');  %90
SegyTraceHeader.SubWeatheringVelocity=fread(segyid,1,'int16');  %92

SegyTraceHeader.SourceUpholeTime=fread(segyid,1,'int16');  %94
SegyTraceHeader.GroupUpholeTime=fread(segyid,1,'int16');  %96
SegyTraceHeader.SourceStaticCorrection=fread(segyid,1,'int16');  %98
SegyTraceHeader.GroupStaticCorrection=fread(segyid,1,'int16');  %100
SegyTraceHeader.TotalStaticApplied=fread(segyid,1,'int16');  %102
SegyTraceHeader.LagTimeA=fread(segyid,1,'int16');  %104
SegyTraceHeader.LagTimeB=fread(segyid,1,'int16');  %106
SegyTraceHeader.DelayRecordingTime=fread(segyid,1,'int16');  %108

SegyTraceHeader.MuteTimeStart=fread(segyid,1,'int16');  %110
SegyTraceHeader.MuteTimeEND=fread(segyid,1,'int16');  %112

SegyTraceHeader.ns=fread(segyid,1,'uint16');  %114
SegyTraceHeader.dt=fread(segyid,1,'uint16');  %116

SegyTraceHeader.GainType=fread(segyid,1,'int16');  %118
SegyTraceHeader.InstrumentGainConstant=fread(segyid,1,'int16');  %120
SegyTraceHeader.InstrumentInitialGain=fread(segyid,1,'int16');  %%122

SegyTraceHeader.Correlated=fread(segyid,1,'int16');  %124

SegyTraceHeader.SweepFrequenceStart=fread(segyid,1,'int16');  %126
SegyTraceHeader.SweepFrequenceEnd=fread(segyid,1,'int16');  %128
SegyTraceHeader.SweepLength=fread(segyid,1,'int16');  %130
SegyTraceHeader.SweepType=fread(segyid,1,'int16');  %132
SegyTraceHeader.SweepTraceTaperLengthStart=fread(segyid,1,'int16');  %134
SegyTraceHeader.SweepTraceTaperLengthEnd=fread(segyid,1,'int16');  %136
SegyTraceHeader.TaperType=fread(segyid,1,'int16');  %138

SegyTraceHeader.AliasFilterFrequency=fread(segyid,1,'int16');  %140
SegyTraceHeader.AliasFilterSlope=fread(segyid,1,'int16');  %142
SegyTraceHeader.NotchFilterFrequency=fread(segyid,1,'int16');  %144
SegyTraceHeader.NotchFilterSlope=fread(segyid,1,'int16');  %146
SegyTraceHeader.LowCutFrequency=fread(segyid,1,'int16');  %148
SegyTraceHeader.HighCutFrequency=fread(segyid,1,'int16');  %150
SegyTraceHeader.LowCutSlope=fread(segyid,1,'int16');  %152
SegyTraceHeader.HighCutSlope=fread(segyid,1,'int16');  %154

SegyTraceHeader.YearDataRecorded=fread(segyid,1,'int16');  %156
SegyTraceHeader.DayOfYear=fread(segyid,1,'int16');  %158
SegyTraceHeader.HourOfDay=fread(segyid,1,'int16');  %160
SegyTraceHeader.MinuteOfHour=fread(segyid,1,'int16');  %162
SegyTraceHeader.SecondOfMinute=fread(segyid,1,'int16');  %164
SegyTraceHeader.TimeBaseCode=fread(segyid,1,'int16');  %166
if SegyTraceHeader.TimeBaseCode==1, SegyTraceHeader.TimeBaseCodeText='Local';
elseif SegyTraceHeader.TimeBaseCode==2, SegyTraceHeader.TimeBaseCodeText='GMT';
elseif SegyTraceHeader.TimeBaseCode==3,
SegyTraceHeader.TimeBaseCodeText='Other';
elseif SegyTraceHeader.TimeBaseCode==4, SegyTraceHeader.TimeBaseCodeText='UTC';
else SegyTraceHeader.TimeBaseCodeText=''; end

SegyTraceHeader.TraceWeightningFactor=fread(segyid,1,'int16');  %170
SegyTraceHeader.GeophoneGroupNumberRoll1=fread(segyid,1,'int16');  %172
SegyTraceHeader.GeophoneGroupNumberFirstTraceOrigField=fread(segyid,1,'int16');
%174
SegyTraceHeader.GeophoneGroupNumberLastTraceOrigField=fread(segyid,1,'int16');
%176
SegyTraceHeader.GapSize=fread(segyid,1,'int16');  %178

SegyTraceHeader.OverTravel=fread(segyid,1,'int16');  %178

SegyTraceHeader.cdpX=fread(segyid,1,'int32');  %180
SegyTraceHeader.cdpY=fread(segyid,1,'int32');  %184

SegyTraceHeader.Inline3D=fread(segyid,1,'int32');  %188
SegyTraceHeader.Crossline3D=fread(segyid,1,'int32');  %192

SegyTraceHeader.ShotPoint=fread(segyid,1,'int32');  %196
SegyTraceHeader.ShotPointScalar=fread(segyid,1,'int16');  %200

SegyTraceHeader.TraceValueMeasurementUnit=fread(segyid,1,'int16');  %202
if SegyTraceHeader.TraceValueMeasurementUnit==-1,
SegyTraceHeader.TraceValueMeasurementUnitText='Other';
```

```
elseif SegyTraceHeader.TraceValueMeasurementUnit==0,
SegyTraceHeader.TraceValueMeasurementUnitText='Unknown';
elseif SegyTraceHeader.TraceValueMeasurementUnit==1,
SegyTraceHeader.TraceValueMeasurementUnitText='Pascal (Pa)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==2,
SegyTraceHeader.TraceValueMeasurementUnitText='Volts (v)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==3,
SegyTraceHeader.TraceValueMeasurementUnitText='Millivolts (v)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==4,
SegyTraceHeader.TraceValueMeasurementUnitText='Amperes (A)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==5,
SegyTraceHeader.TraceValueMeasurementUnitText='Meters (m)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==6,
SegyTraceHeader.TraceValueMeasurementUnitText='Meters Per Second (m/s)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==7,
SegyTraceHeader.TraceValueMeasurementUnitText='Meters Per Second squared
(m/&s2)Other';
elseif SegyTraceHeader.TraceValueMeasurementUnit==8,
SegyTraceHeader.TraceValueMeasurementUnitText='Newton (N)';
elseif SegyTraceHeader.TraceValueMeasurementUnit==8,
SegyTraceHeader.TraceValueMeasurementUnitText='Watt (W)';
else SegyTraceHeader.TraceValueMeasurementUnitText='Undefined'; end

SegyTraceHeader.TransductionConstantMantissa=fread(segyid,1,'int32');  %204
SegyTraceHeader.TransductionConstantPower=fread(segyid,1,'int16'); %208

SegyTraceHeader.TransductionUnit=fread(segyid,1,'int16');  %210

SegyTraceHeader.TraceIdentifier=fread(segyid,1,'int16');  %212

SegyTraceHeader.ScalarTraceHeader=fread(segyid,1,'int16');  %214

SegyTraceHeader.SourceType=fread(segyid,1,'int16');  %216

SegyTraceHeader.SourceEnergyDirectionMantissa=fread(segyid,1,'int32');  %218
SegyTraceHeader.SourceEnergyDirectionExponent=fread(segyid,1,'int16');  %222

SegyTraceHeader.SourceMeasurementMantissa=fread(segyid,1,'int32');  %224
SegyTraceHeader.SourceMeasurementExponent=fread(segyid,1,'int16');  %228

SegyTraceHeader.SourceMeasurementUnit=fread(segyid,1,'int16');  %230

SegyTraceHeader.UnassignedInt1=fread(segyid,1,'int32');  %232
SegyTraceHeader.UnassignedInt2=fread(segyid,1,'int32');  %236





if exist('ns')==0,
  ns=SegyTraceHeader.ns;
end


% GO TO POSITION OF DATA
fseek(segyid,TraceStart+240,'bof');
SegyTraceHeader.SegyMAT_TraceDataStart = ftell(segyid);
```

APPENDIX F

GetSegyTraceData Code

```
% GetSegyTraceData : Get Segy trace data if filehandle
%
% Call :
%
%   tracedata=GetSegyTraceData(segyid,ns,SegyHeader,SkipData
%

%
% (C) 2001-2004 Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
%    This program is free software; you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by
%    the Free Software Foundation; either version 2 of the License, or
%    (at your option) any later version.
%
%    This program is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of
%    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
%    GNU General Public License for more details.
%
%    You should have received a copy of the GNU General Public License
%    along with this program; if not, write to the Free Software
%    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
%
function tracedata=GetSegyTraceData(segyid,ns,SegyHeader,SkipData)
%
% Get Segy Trace Data.
%
%

SkipData=[];

if nargin==0,
    SegymatVerbose(exist('segyid'))
    SegymatVerbose([mfilename,' : SEGYID not specified - exiting'])
    tracedata=[];
    return
end
if nargin==1
  SegymatVerbose([mfilename,' : NS not specified - exiting'])
  tracedata=[];
  return
end
if nargin==2
    SegyHeader.DataFormat='float32';
    SegyHeader.BytesPerSample=32;
    SegyHeader.DataSampleFormat=5; % IEEE
    SegymatVerbose(['Dataformat not specified -  dataformat-
>',SegyHeader.DataFormat])
end;
```

```
if isempty(SkipData)==1, SkipData=0; end

Revision=SegyHeader.SegyFormatRevisionNumber;
if Revision>0, Revision=1; end
Format=SegyHeader.Rev(Revision+1).DataSampleFormat(SegyHeader.DataSampleFormat)
.format;

BPS=SegyHeader.Rev(Revision+1).DataSampleFormat(SegyHeader.DataSampleFormat).bp
s;
if (SkipData==1)
    SkipBytes=ns*BPS/8;
    fseek(segyid,SkipBytes,'cof');
    tracedata=[];
else
    % SegymatVerbose([mfilename,' : ',Format,'. ns=',num2str(ns)])
    try
      tracedata=fread(segyid,ns,Format);
    catch
      SegymatVerbose([mfilename,' : Error using fread - Possibly ''ns'' is
negative -' ...
             ' check byteorder-'])
      tracedata=[];
    end


    if (strcmp(Format,'uint32')==1), % IBM FLOATING POINT
        % CONVERT FROM FLOATING POINT
        verbose=1;
        if verbose>1, SegymatVerbose([mfilename,'Converting from IBM,
DataFormat :',SegyHeader.DataFormat]); end
        try
          tracedata=ibm2num(uint32(tracedata));
        catch
          % SegymatVerbose([mfilename,' : SOMETHING BAD HAPPENED WHEN
CONVERTING FROM IBM FLOATS TO IEEE. ARE YOU SURE DATA ARE IBM FLOAT FORMATTED
?' ])
          % tracedata=0.*tracedata;
          % return

        end
    end;

end
```

APPENDIX G

GetSegyHeaderBasics Code

```
% GetSegyHeaderBasics : Default Segy Header Header settings
%
% Call :
% Rev=GetSegyHeaderBasics
%
%

%
% (C) 2001-2004, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
%    This program is free software; you can redistribute it and/or modify
%    it under the terms of the GNU General Public License as published by
%    the Free Software Foundation; either version 2 of the License, or
%    (at your option) any later version.
%
%    This program is distributed in the hope that it will be useful,
%    but WITHOUT ANY WARRANTY; without even the implied warranty of
%    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
%    GNU General Public License for more details.
%
%    You should have received a copy of the GNU General Public License
%    along with this program; if not, write to the Free Software
%    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
%
function Rev=GetSegyHeaderBasics;

%
%  TODO
%  Fixed Point ???
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Revision 0
Rev(1).name='Revision 0 (1975)';
Rev(1).SegyFormatRevisionNumber=0;
% DataSampleFormat
Rev(1).DataSampleFormat(1).name='4-byte IBM Floating Point';
Rev(1).DataSampleFormat(2).name='4-byte Fixed Point - (UNSUPPORTED)';
Rev(1).DataSampleFormat(3).name='2-byte Fixed Point - (UNSUPPORTED)';
Rev(1).DataSampleFormat(4).name='4-byte Fixed Point with gain - (UNSUPPORTED)';

Rev(1).DataSampleFormat(1).format='uint32'; %
Rev(1).DataSampleFormat(2).format='int32'; %
Rev(1).DataSampleFormat(3).format='int16'; %
%Rev(1).DataSampleFormat(2).format=''; % 'uint32'; %
%Rev(1).DataSampleFormat(3).format=''; % 'uint16'; %
Rev(1).DataSampleFormat(4).format=''; % 'uint32'; %

Rev(1).DataSampleFormat(1).bps=32;
Rev(1).DataSampleFormat(2).bps=32;
Rev(1).DataSampleFormat(3).bps=16;
%Rev(1).DataSampleFormat(4).bps=32;
%Rev(1).DataSampleFormat(3).bps=0; % NOT IMPLEMENETD
Rev(1).DataSampleFormat(4).bps=0; % NOT IMPLEMENETD

% TraceSorting
Rev(1).TraceSorting(1).name='as recorded (no sorting)';
Rev(1).TraceSorting(2).name='CDP ensemble';
Rev(1).TraceSorting(3).name='single fold continous profile';
Rev(1).TraceSorting(4).name='horizontally stacked';
Rev(1).TraceSorting(1).value=1;
Rev(1).TraceSorting(2).value=2;
Rev(1).TraceSorting(3).value=3;
Rev(1).TraceSorting(4).value=4;

% SweepType
Rev(1).SweepType(1).value=1;
Rev(1).SweepType(1).name='Linear';
Rev(1).SweepType(2).value=2;
Rev(1).SweepType(2).name= 'Parabolic';
Rev(1).SweepType(3).value=3;
Rev(1).SweepType(3).name='Exponential';
Rev(1).SweepType(4).value=4;
Rev(1).SweepType(4).name= 'Other';

% TaperType
Rev(1).TaperType(1).value=1;
Rev(1).TaperType(1).name='Linear';
Rev(1).TaperType(2).value=2;
Rev(1).TaperType(2).name= 'Cos2';
Rev(1).TaperType(3).value=3;
Rev(1).TaperType(3).name='Other';

% CorrelatedDataTraces
Rev(1).CorrelatedDataTraces(1).value=1;
Rev(1).CorrelatedDataTraces(1).name='No';
Rev(1).CorrelatedDataTraces(2).value=2;
Rev(1).CorrelatedDataTraces(2).name= 'Yes';

% BinaryGain
Rev(1).BinaryGain(1).value=1;
Rev(1).BinaryGain(1).name='Yes';
Rev(1).BinaryGain(2).value=2;
Rev(1).BinaryGain(2).name= 'No';

% AmplitudeRecoveryMethod
Rev(1).AmplitudeRecoveryMethod(1).value=1;
Rev(1).AmplitudeRecoveryMethod(1).name='none';
Rev(1).AmplitudeRecoveryMethod(2).value=2;
Rev(1).AmplitudeRecoveryMethod(2).name= 'Spherical Divergence';
Rev(1).AmplitudeRecoveryMethod(3).value=3;
Rev(1).AmplitudeRecoveryMethod(3).name= 'AGC';
Rev(1).AmplitudeRecoveryMethod(4).value=4;
Rev(1).AmplitudeRecoveryMethod(4).name= 'Other';

% MeasurementSystem
Rev(1).MeasurementSystem(1).value=1;
Rev(1).MeasurementSystem(1).name='Meters';
Rev(1).MeasurementSystem(2).value=2;
Rev(1).MeasurementSystem(2).name= 'Feets';

% ImpulseSignalPolarity
Rev(1).ImpulseSignalPolarity(1).value=1;
Rev(1).ImpulseSignalPolarity(1).name='Negative number';
Rev(1).ImpulseSignalPolarity(2).value=2;
Rev(1).ImpulseSignalPolarity(2).name= 'Positive Number';

% VibratoryPolarityCode
Rev(1).VibratoryPolarityCode(1).value=1;
Rev(1).VibratoryPolarityCode(1).name='337.5 - 22.5';
Rev(1).VibratoryPolarityCode(2).value=2;
Rev(1).VibratoryPolarityCode(2).name= '22.5-67.5';
Rev(1).VibratoryPolarityCode(3).value=3;
Rev(1).VibratoryPolarityCode(3).name= '67.5-112.5';
Rev(1).VibratoryPolarityCode(4).value=4;
Rev(1).VibratoryPolarityCode(4).name= '112.5-157.5';
Rev(1).VibratoryPolarityCode(5).value=5;
Rev(1).VibratoryPolarityCode(5).name= '157.5-202.5';
Rev(1).VibratoryPolarityCode(6).value=6;
Rev(1).VibratoryPolarityCode(6).name= '202.5-247.5';
Rev(1).VibratoryPolarityCode(7).value=7;
Rev(1).VibratoryPolarityCode(7).name= '247.5-292.5';
Rev(1).VibratoryPolarityCode(8).value=8;
Rev(1).VibratoryPolarityCode(8).name= '292.5-337.5';

% FixedLengthTraceFlag
Rev(1).FixedLengthTraceFlag(1).value=0;
Rev(1).FixedLengthTraceFlag(1).name='Varying Trace Length';
Rev(1).FixedLengthTraceFlag(2).value=1;
Rev(1).FixedLengthTraceFlag(2).name= 'Fixed Trace Length';



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Revision 1
Rev(2).name='Revision 1 (2002)';
Rev(2).SegyFormatRevisionNumber=100;
% DataSampleFormat
Rev(2).DataSampleFormat(1).format='uint32'; %
Rev(2).DataSampleFormat(2).format='int32'; %
Rev(2).DataSampleFormat(3).format='int16'; %
Rev(2).DataSampleFormat(4).format=''; %
Rev(2).DataSampleFormat(5).format='float32'; %
Rev(2).DataSampleFormat(6).format=''; %
Rev(2).DataSampleFormat(7).format=''; %
Rev(2).DataSampleFormat(8).format='int8'; %

Rev(2).DataSampleFormat(1).bps=32;
Rev(2).DataSampleFormat(2).bps=32;
Rev(2).DataSampleFormat(3).bps=16;
Rev(2).DataSampleFormat(4).bps=0;
Rev(2).DataSampleFormat(5).bps=32;
Rev(2).DataSampleFormat(6).bps=0;
Rev(2).DataSampleFormat(7).bps=0;
Rev(2).DataSampleFormat(8).bps=8;

Rev(2).DataSampleFormat(1).name='4-byte IBM Floating Point';
Rev(2).DataSampleFormat(2).name='4-byte two''s complement';
Rev(2).DataSampleFormat(3).name='2-byte two''s complement';
Rev(2).DataSampleFormat(4).name='4-byte fixed-point with gain (obsolete) - NOT
IMPLEMENTED';
Rev(2).DataSampleFormat(5).name='4-byte IEEE floating-point';
Rev(2).DataSampleFormat(6).name='DSF Not currently used';
Rev(2).DataSampleFormat(7).name='DSF Not currently used';
Rev(2).DataSampleFormat(8).name='1-byte, two''s complement integer';

% TraceSorting
Rev(2).TraceSorting(1).value=-1;
Rev(2).TraceSorting(1).name='Other (should be explained in user Extended
Textual File Header stanza)';
Rev(2).TraceSorting(2).value=0;
Rev(2).TraceSorting(2).name= 'Unknown';
Rev(2).TraceSorting(3).value=1;
Rev(2).TraceSorting(3).name= 'As recorded (no sorting)';
Rev(2).TraceSorting(4).value=2;
Rev(2).TraceSorting(4).name= 'CDP ensemble';
Rev(2).TraceSorting(5).value=3;
Rev(2).TraceSorting(5).name= 'Single fold continuous profile';
Rev(2).TraceSorting(6).value=4;
Rev(2).TraceSorting(6).name= 'Horizontally stacked';
Rev(2).TraceSorting(7).value=5;
Rev(2).TraceSorting(7).name= 'Common source';
Rev(2).TraceSorting(8).value=6;
Rev(2).TraceSorting(8).name= 'Common receiver';
Rev(2).TraceSorting(9).value=7;
Rev(2).TraceSorting(9).name= 'Common offset';
Rev(2).TraceSorting(10).value=8;
Rev(2).TraceSorting(10).name= 'Common mid-point';

% SweepType
```

```
Rev(2).SweepType(1).value=1;
Rev(2).SweepType(1).name='Linear';
Rev(2).SweepType(2).value=2;
Rev(2).SweepType(2).name= 'Parabolic';
Rev(2).SweepType(3).value=3;
Rev(2).SweepType(3).name='Exponential';
Rev(2).SweepType(4).value=4;
Rev(2).SweepType(4).name= 'Other';

% TaperType
Rev(2).TaperType(1).value=1;
Rev(2).TaperType(1).name='Linear';
Rev(2).TaperType(2).value=2;
Rev(2).TaperType(2).name= 'Cos2';
Rev(2).TaperType(3).value=3;
Rev(2).TaperType(3).name='Other';

% CorrelatedDataTraces
Rev(2).CorrelatedDataTraces(1).value=1;
Rev(2).CorrelatedDataTraces(1).name='No';
Rev(2).CorrelatedDataTraces(2).value=2;
Rev(2).CorrelatedDataTraces(2).name= 'Yes';

% BinaryGain
Rev(2).BinaryGain(1).value=1;
Rev(2).BinaryGain(1).name='Yes';
Rev(2).BinaryGain(2).value=2;
Rev(2).BinaryGain(2).name= 'No';

% AmplitudeRecoveryMethod
Rev(2).AmplitudeRecoveryMethod(1).value=1;
Rev(2).AmplitudeRecoveryMethod(1).name='none';
Rev(2).AmplitudeRecoveryMethod(2).value=2;
Rev(2).AmplitudeRecoveryMethod(2).name= 'Spherical Divergence';
Rev(2).AmplitudeRecoveryMethod(3).value=3;
Rev(2).AmplitudeRecoveryMethod(3).name= 'AGC';
Rev(2).AmplitudeRecoveryMethod(4).value=4;
Rev(2).AmplitudeRecoveryMethod(4).name= 'Other';

% MeasurementSystem
Rev(2).MeasurementSystem(1).value=1;
Rev(2).MeasurementSystem(1).name='Meters';
Rev(2).MeasurementSystem(2).value=2;
Rev(2).MeasurementSystem(2).name= 'Feets';

% ImpulseSignalPolarity
Rev(2).ImpulseSignalPolarity(1).value=1;
Rev(2).ImpulseSignalPolarity(1).name='Negative number';
Rev(2).ImpulseSignalPolarity(2).value=2;
Rev(2).ImpulseSignalPolarity(2).name= 'Positive Number';
```

```
% VibratoryPolarityCode
Rev(2).VibratoryPolarityCode(1).value=1;
Rev(2).VibratoryPolarityCode(1).name='337.5 - 22.5';
Rev(2).VibratoryPolarityCode(2).value=2;
Rev(2).VibratoryPolarityCode(2).name= '22.5-67.5';
Rev(2).VibratoryPolarityCode(3).value=3;
Rev(2).VibratoryPolarityCode(3).name= '67.5-112.5';
Rev(2).VibratoryPolarityCode(4).value=4;
Rev(2).VibratoryPolarityCode(4).name= '112.5-157.5';
Rev(2).VibratoryPolarityCode(5).value=5;
Rev(2).VibratoryPolarityCode(5).name= '157.5-202.5';
Rev(2).VibratoryPolarityCode(6).value=6;
Rev(2).VibratoryPolarityCode(6).name= '202.5-247.5';
Rev(2).VibratoryPolarityCode(7).value=7;
Rev(2).VibratoryPolarityCode(7).name= '247.5-292.5';
Rev(2).VibratoryPolarityCode(8).value=8;
Rev(2).VibratoryPolarityCode(8).name= '292.5-337.5';

% FixedLengthTraceFlag
Rev(2).FixedLengthTraceFlag(1).value=0;
Rev(2).FixedLengthTraceFlag(1).name='Varying Trace Length';
Rev(2).FixedLengthTraceFlag(2).value=1;
Rev(2).FixedLengthTraceFlag(2).name= 'Fixed Trace Length';
```

APPENDIX H

TraceHeaderToInfo Code

```
% TraceHeaderToInfo : Extract traceheader values from structures into an array
%
%[HeaderInfo]=TraceheaderToInfo(SegyTraceHeaders)

%
% (C) 2001-2002, Thomas Mejer Hansen, tmh@gfy.ku.dk/thomas@cultpenguin.com
%
% OBSOLETE
%
function [HeaderInfo]=TraceheaderToInfo(SegyTraceHeaders)

     SingleSegyTraceHeaders=SegyTraceHeaders(1);

     S=size(SegyTraceHeaders,2);
     cdp=zeros(1,S);
     offset=zeros(1,S);

     HeaderInfo.cdp=[SegyTraceHeaders.cdp];
     HeaderInfo.offset=[SegyTraceHeaders.offset];
     HeaderInfo.Inline3D= [SegyTraceHeaders.Inline3D];
     HeaderInfo.Crossline3D=[SegyTraceHeaders.Crossline3D];
     HeaderInfo.SourceX=[SegyTraceHeaders.SourceX];
     HeaderInfo.SourceY=[SegyTraceHeaders.SourceY];
     HeaderInfo.GroupX=[SegyTraceHeaders.GroupX];
     HeaderInfo.GroupY=[SegyTraceHeaders.GroupY];
     HeaderInfo.cdpX=[SegyTraceHeaders.cdpX];
     HeaderInfo.cdpY=[SegyTraceHeaders.cdpY];

  HeaderInfo.t = [1:1:SingleSegyTraceHeaders.ns]*SingleSegyTraceHeaders.dt*1e-6
- SingleSegyTraceHeaders.DelayRecordingTime./1000;
```