# LAVA LAKE ERUPTIVE PROCESSES QUANTIFIED WITH INFRASOUND AND VIDEO AT MOUNT EREBUS, ANTARCTICA

by

Alexander Miller

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Geophysics

Boise State University

August 2016

BOISE STATE UNIVERSITY GRADUATE COLLEGE

# DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Alexander Miller

Thesis Title: Lava Lake Eruptive Processes Quantified with Infrasound and Video at Mount Erebus, Antarctica

Date of Final Oral Examination: 04 March 2016

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

| | |
|---|---|
| Jeffrey B. Johnson, Ph.D. | Chair, Supervisory Committee |
| Brittany Brand, Ph.D. | Member, Supervisory Committee |
| Paul Michaels, Ph.D. | Member, Supervisory Committee |

The final reading approval of the thesis was granted by Jeffrey B. Johnson, Ph.D., Chair of the Supervisory Committee. The thesis was approved for the Graduate College by Jodi Chilson, M.F.A., Coordinator of Theses and Dissertations.

The truth is out there.

# ACKNOWLEDGMENTS

The author wishes to express gratitude to the Mount Erebus Volcano Observatory, efforts of Dr. Philip Kyle, Dr. Richard Aster, Dr. William McIntosh, Jake Anderson, Hugo Ortiz, and the McMurdo personnel.

# ABSTRACT

A natural laboratory exists at Mount Erebus where strombolian activity from the lava lake is directly observed from the crater rim. Lava lake eruptions occur when pressurized bubble slugs distend the lake surface before bursting within a few tenths of a second. The unique setting presents an ideal site to quantify bubble growth through infrasound and video analysis. Two infrasound sensors and one video camera recorded eruptions ∼330 m from the lava lake in 2006. Infrasound waveforms exhibit a high-amplitude bipolar pulse followed by a coda consisting of about five decaying oscillations. Video records are quantified by tracking the expanding bubble edge, which is approximated to a 3-D hemispherical volume. Video-inferred volumes scale closely with infrasound-inferred volumes during the bipolar pulse but deviate during the coda. Volcanic processes responsible for infrasound after the bipolar pulse are therefore unrelated to the inertially traveling lava lake fragments observed in the video during the coda. Two peak frequencies dominate coda spectra and reveal infrasound sources not observed in the video. Frequency peaks recorded at both stations suggest that echoes reflecting off the steep crater walls influence the infrasound codas. After removing echo contamination, both stations exhibit a single frequency peak at 1.48 Hz, which is attributed to a Helmholtz resonance within the evacuated magma conduit. Quantitatively relating infrasound to video thus gives a detailed chronology of a typical Mount Erebus eruption.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

**ARMA** – Auto Regressive Moving Average

**DEM** – Digital Elevation Model

**FLIR** – Forward Looking Infrared Radiometer

**FPS** – Frames Per Second

**FWZPZF** – Finite Window Zero Pressure Zero Flux

**GPS** – Global Positioning System

**MEMS** – Microelectromechanical Systems

**MEVO** – Mount Erebus Volcano Observatory

**RAY** – Infrasound station

**RGB** – Red Green Blue color values

**RMS** – Root Mean Square

**SHK** – Infrasound and video station

**SNR** – Signal to Noise Ratio

**SPS** – Samples Per Second

# LIST OF SYMBOLS

| | |
|---|---|
| $\theta$ | Azimuthal angle used in edge detection ($degrees$) |
| $\Delta\theta$ | Incremental increase in azimuth ($degrees$) |
| $\alpha$ | Radial distance measured at particular azimuth ($m$) |
| $V_{frame}$ | Video inferred frame volume ($kg/m^3$) |
| $\rho_{atm}$ | Atmospheric density ($kg/m^3$) |
| $d$ | Distance from source to receiver ($m$) |
| $c$ | Ambient sound speed ($m/s$) |
| $r$ | Radius of expanding lava bubble ($m$) |
| $V_{vid}$ | Video-inferred cumulative volume ($m^3$) |
| $P_{vid}$ | Video-inferred excess pressure ($Pa$) |
| $P_{inf}$ | Infrasound excess pressure conditioned according to the FWZPZF method ($Pa$) |
| $q_{inf}$ | Infrasound-inferred source volume flux ($m^3/s$) |
| $V_{inf}$ | Infrasound-inferred cumulative volume ($m^3$) |
| $\Delta f$ | Difference in frequency peaks ($Hz$) |
| $\Delta t$ | Difference in primary and echo arrival ($s$) |
| $t_e$ | Echo arrival time ($s$) |

$\alpha_e$ Echo amplitude

$h(t)$ Echo Green's function

$H(z)$ Z transform of h(t)

$D(z)$ Deconvolution function

$Y(z)$ Deconvolved output in z domain

$X(z)$ Contaminated input in z domain

$y(t)$ Deconvolved output in time domain

$x(t)$ Contaminated input in time domain

$\gamma_{mix}$ Weighted heat capacity ratios

$R_{mix}$ Gas constants of the individual gas constituents $(J/kg/K)$

$T$ Absolute temperature measured $(K)$

$c_{res}$ Sound speed in the Erebus resonator $(m/s)$

$v_{res}$ Resonator volume $(m^3)$

$r_{res}$ Resonator neck radius $(m)$

$f_{res}$ Resonator fundamental frequency $(Hz)$

$h_{res}$ Resonator height $(m)$

# CHAPTER 1

# INTRODUCTION

## 1.1 Volcano Infrasound Background

Infrasound waves emanating from volcanic vents yield information on eruptive source processes. Erupted gas and pyroclasts ejected over large spatial scales (tens of meters or greater) accelerate the overlying atmosphere, creating a pressure wave recorded as transient changes in atmospheric pressure (Johnson, 2004). In general, volcano acoustic signals possess low peak frequencies (often between 0.5 and 2 Hz) and may have intense sound pressures (100s of Pa at 1 km) (Johnson & Ripepe, 2011). Typical explosion infrasound waveforms consist of an initial compression, then rarefaction, followed by a longer, more complex coda (Morrissey & Chouet, 1997; Johnson, 2003; Fee & Matoza, 2013).

Volcano infrasound studies can be used to relate atmospheric pressure waves to volumetric source processes. Explosive gas volume was estimated at Erebus (Johnson et al., 2008) and Tolbachik (Firstov & Kravchenko, 1996) volcanoes for short-duration (<5 s) explosions assuming a simple acoustic source. Further, Johnson and Miller (2014) estimated cumulative flux from infrasound records of long-duration (>10 sec) vulcanian eruptions at Sakurajima Volcano, Japan. Although infrasound analysis

can be used to robustly quantify and monitor volcano processes, when possible, a multiparametric approach further constrains volcanic behavior.

## 1.2 Mount Erebus Background

Mount Erebus's unique setting and frequent stombolian-style eruptions create an ideal natural laboratory to study volcanic explosion mechanisms with a diverse suite of geophysical instruments. Located on Ross Island, Antarctica, Mount Erebus is a 3,794 m high stratovolcano whose crater floor is directly observable from the rim ∼200 meters above. Volcanic eruptions commonly come from two sources on the crater floor: a constricted vent that erupts ash-rich gases, and, tens of meters away, an active lava lake exhibiting bubble bursting events (Jones et al., 2008). This research focuses on the strombolian activity from the lava lake known as Ray Lake.

Ray Lake's chemistry and conduit geometry support characteristic strombolian activity. The phonolitic lava is 56% silica with a inferred viscosity of $10^3$ Pa s at 1200° C (Dibble et al., 1984). $CO_2$ and $H_2O$ gas exsolve at depth and coalesce into large slugs with widths comparable to the conduit diameter and lengths greater than the width (Moussallam et al., 2012; Oppenheimer et al., 2011). The slug grows from decompression as it rises through the conduit, producing seismic tremor (Dibble et al., 1984). The bubble reaches the magma-air interface and begins to distend the lava lake surface hemispherically before fragmenting and releasing pressurized gas into the atmosphere. Intact portions of the lava surface continue to accelerate until the bubble fully bursts, after which fragments traveling inertially reach heights of hundreds of meters (Gerst et al., 2013). The duration between first motion of the

lava lake surface and complete bubble bursting varies between eruptions and ranges between 0.5 and 1.5 s. Immediately following bubble bursting, a large void remains in the conduit that represents the pre-burst slug volume (Gerst et al., 2013), which fills to pre-eruption levels after several minutes.

Due to the relatively simple source mechanism and excellent view of vent activity, the regularly exploding lava lake at Mount Erebus is the subject of continuous geophysical monitoring (Aster et al., 2004). The Mount Erebus Volcano Observatory (MEVO), with support from NSF Polar Programs, continuously monitors Mount Erebus through seismic, infrasound, and video surveillance (Aster et al., 2004). Other campaign deployments, including Doppler radar and magnetic surveys, add to the geophysical catalog (Dibble et al., 1984; Gerst et al., 2008). Integrated geophysical data greatly enhance the ability to capture lava lake eruption dynamics and quantify eruption processes.

Erebus's bubble bursting events radiate significantly more infrasonic energy than seismic energy, which highlight the importance of infrasound investigations (Gerst et al., 2013). Displaced lava from the slug accelerates the overlying atmosphere, creating a pressure wave dominated by infrasonic frequencies. Infrasound recordings of eruptions share similarities in shape and duration due to the relatively simple and regular source mechanism. Infrasound transients vary slightly among eruptions but typically contain a bipolar pulse (compression and rarefaction) followed by a $\sim$3 s long coda composed of several decaying oscillations (Fig. 1.1).

Despite the resemblance between bubble bursting events, slight differences in eruptions exist that influence the corresponding infrasound record. Variations in

peak amplitudes and bipolar pulse duration relate to bubble size and growth history. Infrasound source locations from Ray Lake are scattered across an approximate 200 $m^2$ area, indicating that the lava lake's first motions from a rising slug are spatially variable (Jones et al., 2008). Deviations from the typical bipolar pulse include events exhibiting multiple compressional peaks and dilational troughs. Multiple bubble bursts explain the more complex infrasound signals by superimposing a relatively simple infrasound signal with itself at varying lag times (Rowe et al., 2000). The lava lake membrane initially fragments at random locations across the shell and releases directed gas bursts, which explains small dipole components inferred from infrasound analysis (Johnson et al., 2008). Deviations from a monopole source are corroborated by Doppler radars tracking the expanding lake membrane (Gerst et al., 2008).

More recently, Gerst et al. (2013) used Doppler radar to directly relate the expanding bubble to infrasound genesis during the first second of a Ray Lake eruption. A single Doppler radar was used to measure velocity at a point on the expanding membrane, which, at 15 samples per second (SPS), approximates a hemispherical model of the expanding bubble. Superimposing phase and amplitudes from a random distribution of point sources across the shell creates a synthetic non-compact source that compares favorably to recorded infrasound. Double-peak patterns characterize Doppler-inferred infrasound where the first peak often correlates to changes in slope in the infrasound records while the second Doppler peak correlates with the infrasound compressional peak. Doppler to infrasound comparisons are valid during the expansion of the spherical shell before burst. Changes in the Doppler radar echo

spectrum, showing the echo power as a function of particle velocity, indicate bursting times for each event. Specifically, changes from a narrow spectrum to wider spectrum indicate the transition from the lava lake membrane moving as a single cohesive shell to the membrane breaking up into ballistics traveling at a broad range of velocities (Gerst et al., 2013).

Gerst et al. (2013) speculate about the infrasound source that persists after the bubble bursts and consider a resonance in the evacuated conduit. Slug volumes calculated from the Doppler-inferred bubble model and known geometry of the lava lake approximate the evacuated conduit dimensions. The conduit, treated as a $\lambda/4$ resonator (Vidal et al., 2006), yields a resonating frequency of $\sim$1.5 Hz with characteristic damping times ranging between 0.4 s and 3.4 s. Expected resonance and damping times match inter-station infrasound data to varying degrees. Gerst et al. (2013) suggest propagation effects, including echoes off the steep crater walls, as a possible explanation for discrepancies in Doppler and infrasound data $\sim$1 s following the eruption onset.

The MEVO video data recorded at $\sim$ 30 FPS is compared to infrasound data to relate bubble growth, conduit resonance, and echoes to infrasound genesis. Our methods use a novel application of a semi-autonomous image processing algorithm to locate the edges of the expanding bubble. An expanding bubble is modeled by tracking the bubble edge in video data and is used to calculate an acoustic source. Similar to Doppler radar, results from video processing correlate well during the early stages of the infrasound signal, but are less correlated later. Analysis of infrasound signal that is uncorrelated with visual activity from video gives new insights into

volcano processes occurring after the bubble has burst. Infrasound coda suggest path effects influence the waveform after ∼1 s. Analysis of the digital elevation model (DEM) corroborates acoustic path effects, and reveals echoes from initial bubble growth reflect off steep crater walls. Removing the echo contamination from the infrasound coda yields a strong peak in coda spectra that is related to a Helmholtz resonance. This study demonstrates that infrasound is a robust means to describe both visible and invisible vent activity at Mount Erebus's lava lake.

## 1.3 Instrumentation and Deployment

Infrasound and video data come from a period of heightened lava lake activity in January 2006 (Johnson et al., 2008) when three to four explosions occurred per day. In 2006, four infrasound sensors recorded continuously with Guralp digitizers sampling at 40 SPS as part of a telemetered multi-disciplinary geophysical network (Aster et al., 2004). Two acoustic stations, RAY and SHK, recorded infrasound data near the lava lake source (∼330 m to both stations). Station locations were on opposite sides of the lava lake (Fig. 1.2) (azimuthal separation between both stations and Ray Lake was 135°), allowing an investigation for topographical site effects. A homogeneous atmosphere and straight line acoustic propagation is assumed for direct arrivals based on the source-to-receiver proximity (Cannata et al., 2009). Both microphones consisted of MEMS-based differential pressure transducers (Marcillo et al., 2012) with corner frequencies (-3 dB) at 0.05 Hz or lower. The microphone frequency response, from the mechanical filter, is flat up to the nyquist frequency at 20 Hz. Combined instrument/telemetry self-noise and ambient noise was measured as

a root mean square (RMS) amplitude for the 10 s before the arrival of the explosion signal (Tab. 2.1). Noise was well below the eruption pulse amplitudes and RMS signal level for most eruptions.

Of the 350 lava lake events discussed in the Jones et al. (2008) catalog, 13 events were selected that correspond to digital video records with clear views of the lava lake and for which the infrasound signal-to-noise ratio (SNR) exceeds one (Tab. 2.1). All videos were filmed 325 m from the center of Ray Lake on top of the crater rim at station SHK. The infrared-sensitive camera produced false-color images with an image resolution of 640 x 480 and an approximate 300 m field of view of the southeast crater wall (Fig. 1.2). The lava lake center is viewed at 46° from horizontal and spans an area of ∼8100 pixels. Videos were GPS time-stamped at every frame, providing critical timing information for comparison with infrasound records. Each frame of every video was converted into a separate image file for processing. The frame rate ranged from 23 to 29 frames per second (FPS). Ray Lake's radius, estimated at 25 m (Johnson et al., 2008), scaled the size of the explosion sources. Image processing is performed for about 100 video frames, or ∼4 s, which encompassed the duration of bubble growth and the length of the trailing coda.

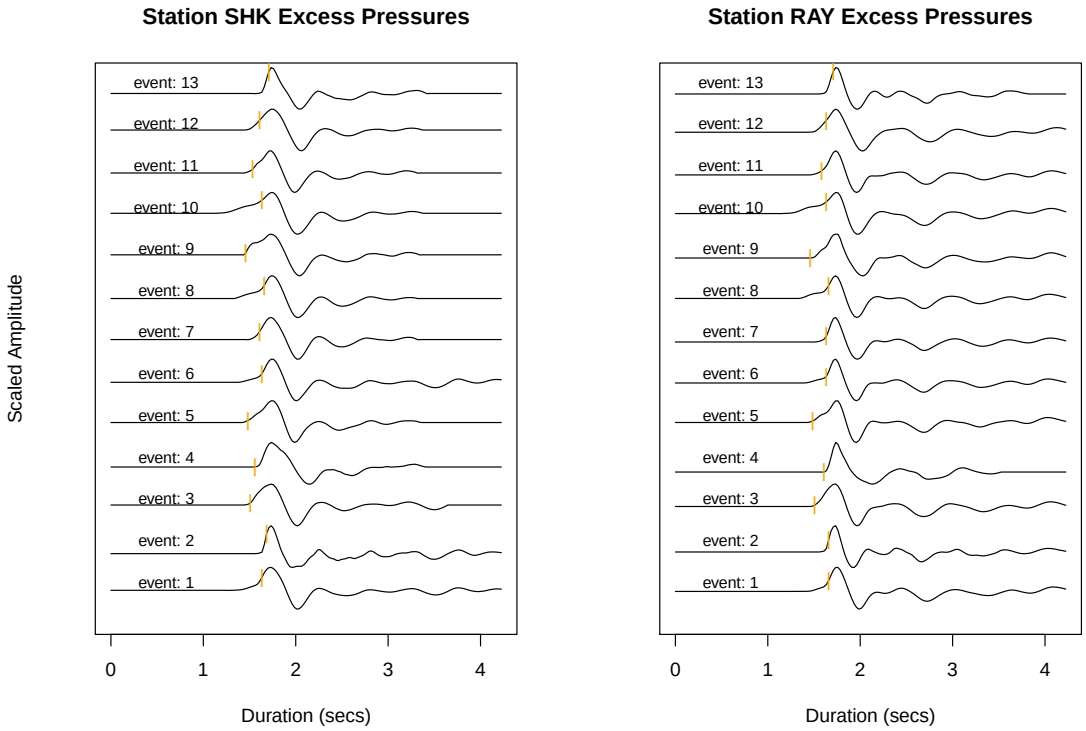


Figure 1.1: Normalized infrasound pressure records. Amplitudes are given in Table 1. Amplitudes are scaled between zero and one. All 13 events exhibit bipolar pulse (compression followed by rarefaction) followed by a trailing coda. Bursting time is marked by the vertical yellow bars. Bipolar shape and length are similar between stations but vary from eruption to eruption. Coda shapes and lengths are similar for each eruption but vary depending on the station.

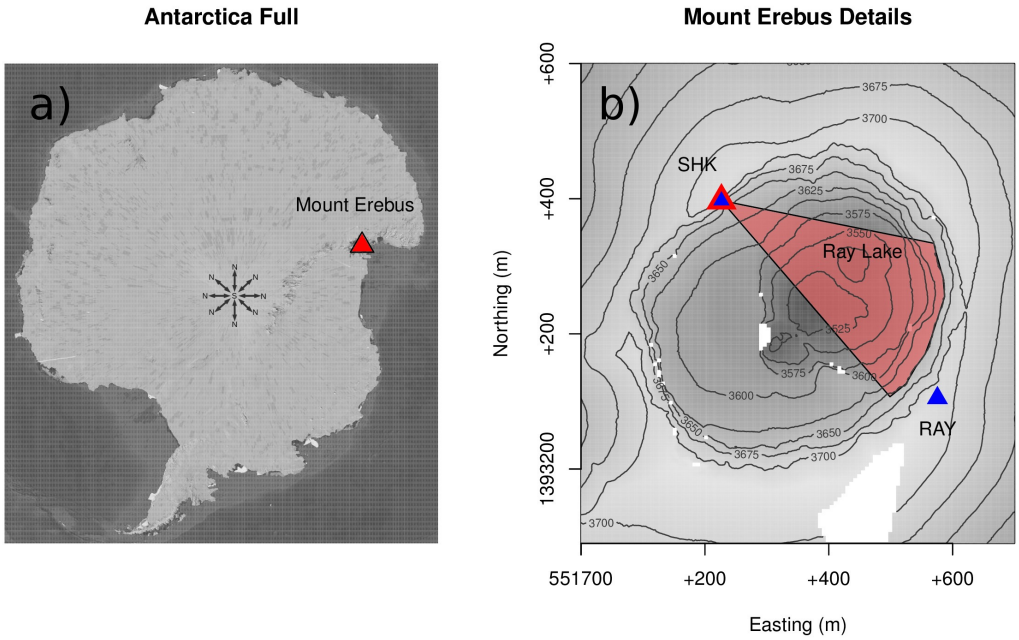


Figure 1.2: Overview of Antarctica and station locations around Mount Erebus. a) Mount Erebus location in Antarctica and b) station locations. Station SHK was co-located with video camera. Approximate field of view is indicated in transparent red. UTM northing and easting is indicated in lower left.

# CHAPTER 2

# METHODS

## 2.1 Image Processing and Quantification

Images of Ray Lake eruptions show an expanding bubble at the magma-air interface prior to fragmentation. Initial lava lake distension centers on the brightest area, where the bubble grows as a spherical cap with an expanding height and relatively constant base radius (Gerst et al., 2013). The bubble bursts when the slug overpressure ruptures the bubble membrane, resulting in directionalized gas jetting at the fragmentation point. Thereafter, intact portions of the bubble membrane expand radially until completely fragmenting into separate ballistics $\sim$1 s later (Gerst et al., 2013). A semi-autonomous image processing algorithm is used to quantify the expansion chronology from the thirteen featured eruptions.

Similar analysis is carried out on each eruption video, but several image processing parameters vary slightly due to shifts in contrast and lighting over the course of video acquisition. First, a 4 s time window is identified, beginning with the background static lava lake conditions. Second, an image analysis area is identified and cropped according to the maximum observed extent of the bubble (Fig. 2.1b). Third, the RGB color value is picked from the pixel centered over the lava lake area during the

onset of an explosion. A spectrum of RGB color values is manually determined to identify the remaining pixels associated with lava. Fourth, each frame is converted to a boolean color map consisting of lava and background (Fig. 2.1c, f). The boolean image is smoothed with a 5x5 pixel box filter to eliminate speckle noise and better define the edge of the bubble. Fifth, an edge detection algorithm is applied after the infrasound onset to track the expanding edges of the bubble. As the eruption progresses, the lava area expands approximately radially about the center point (Fig. 2.1f). This expansion quantifies the explosion process.

Image analysis is used to produce synthetic cumulative volume time series from the video frames. Distances from the lava lake center to the expanding edge are calculated as a function of azimuth, $\theta$, at increments of one degree ($\Delta\theta = 1°$). Radial distances, $\alpha$, are calculated only above the horizontal line that intersects the lake center and are used to quantify volumes of the expanding bubble. A spherical wedge geometry is used to estimate a volume.

$$V_\theta = \frac{2}{3}\frac{\pi}{180}\Delta\theta\alpha_\theta^3 \tag{2.1}$$

Volume at each frame is estimated as the summed volume for every spherical wedge.

$$V_{frame} = \sum_{\theta=0}^{180} V_\theta \tag{2.2}$$

Frame volume is synced to GPS timing to derive a source volume time series, $V_{vid}$. Volume time histories are converted from pixels$^3$ to m$^3$ with a conversion factor

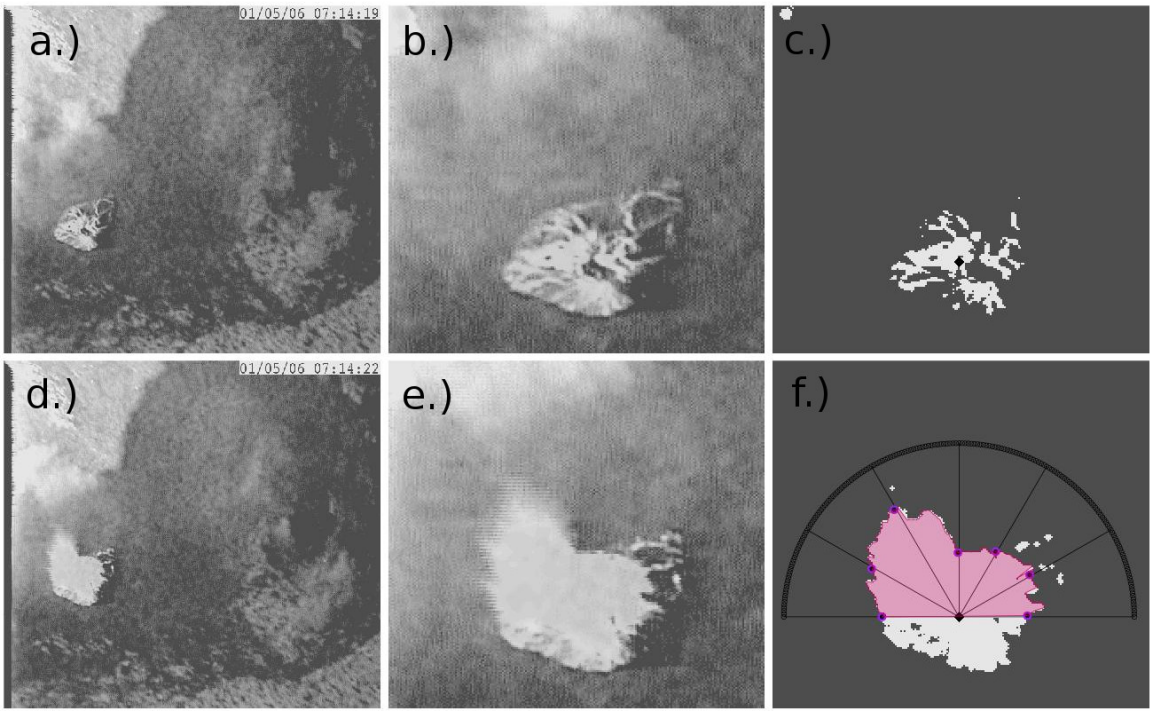


Figure 2.1: Overview of image processing. a-c) Lava lake images corresponding to the first visual indication of motion from a rising bubble, d-f) ∼2 s after first motion, a,d) gray scaled original, b,e) cropped stills respectively, c) boolean mask of lava lake at first motion with calculated center point, and f) Boolean mask with edge detection. Edge distances are indicated for 7 azimuthal values.

of (0.55 m/pixel)$^3$ based upon the the known diameter of Ray Lake (Johnson et al., 2008) (Fig. 2.2i).

Errors in source volume calculations include uncertainties primarily related to geometric or color thresholding issues. Geometric simplifications include the assumption of radial wedge geometry, an oblique viewing angle of a potentially non-spherical source, and the pixel size calibration. Spherical wedge geometries are assumed due to the single camera location, but are considered a reasonable approximation due

to the predominately hemispherical expansion (Gerst et al., 2008). The camera viewing angle of the lava lake is measured at 46° from horizontal; however, we consider the effects negligible as the image processing routine infers volumes from predominantly semi-circle edges detected above the calculated center point. Source detection uncertainties refer to the limitations of the image processing algorithm to adequately detect the edge of expansion. We minimize this uncertainty by tuning step three for each video sequence such that the boolean color map best corresponds to the visual observations of expanding lava and pyroclasts. Manual analysis is also necessary to assure that non-eruptive features, such as snow, steam, or clouds, are not considered as dynamic expansions of lava.

Volume histories are filtered to smooth noise artifacts from edge detection. A two-pole Butterworth lowpass filter of 2.5 Hz applied to the volume time series minimizes high-frequency noise fluctuations, which are outside the spectrum of the associated infrasound. Video-inferred volume flux, $q_{vid}$, is calculated by time differentiating the filtered volume time series.

$$q_{vid} = \dot{V}_{vid}(t) \tag{2.3}$$

This volume flux is used as an input parameter in an assumed simple acoustic source (Lighthill, 1978). A compact monopole source relates volume acceleration to pressure time histories for radial propagation of an acoustic wave. At a given distance ($d$), and using a speed of sound ($c$) of 340 m/s, and atmospheric density of 1.189 kg/m$^2$ ($\rho_{atm}$), the synthetic infrasound derived from a hemispherical volumetric source would be

$$P_{vid} = \frac{1}{2\pi r}\rho_{atm}\ddot{V}_{vid}\Big(t - \frac{d}{c}\Big). \tag{2.4}$$

Video volumes and pressures correspond to the expanding explosion source associated with Erebus lava lake events and provide quantitative information about timing and rates of expansion. To test whether video-derived infrasound approximates recorded infrasound, $P_{vid}$ is calculated from $V_{vid}$ and compared with infrasound data.

## 2.2 Infrasound Conditioning and Processing

The infrasound source is considered a simple monopole (Lighthill, 1978), and the recorded waveform is studied over a longer duration than previously investigated (Johnson et al., 2008; Gerst et al., 2013). Non-linear near-source effects, which are possible for very large volcanic blasts (Morrissey & Chouet, 1997), are not considered in our data at Mount Erebus. Conditioning the pressure waveform according to the Finite Window Zero Pressure Zero Flux (FWZPZF) (Johnson & Miller, 2014) forces both the starting and ending values of excess pressure and flux to zero. The FWZPZF application allows a more comprehensive analysis of the trailing coda that follows the bipolar pulse. The FWZPZF method windows the event length variably where event duration range between 1.9 and 4.7 s over the 13 eruptions (Table 2.1). The conditioned excess pressure waveform is referred to as $P_{inf}$.

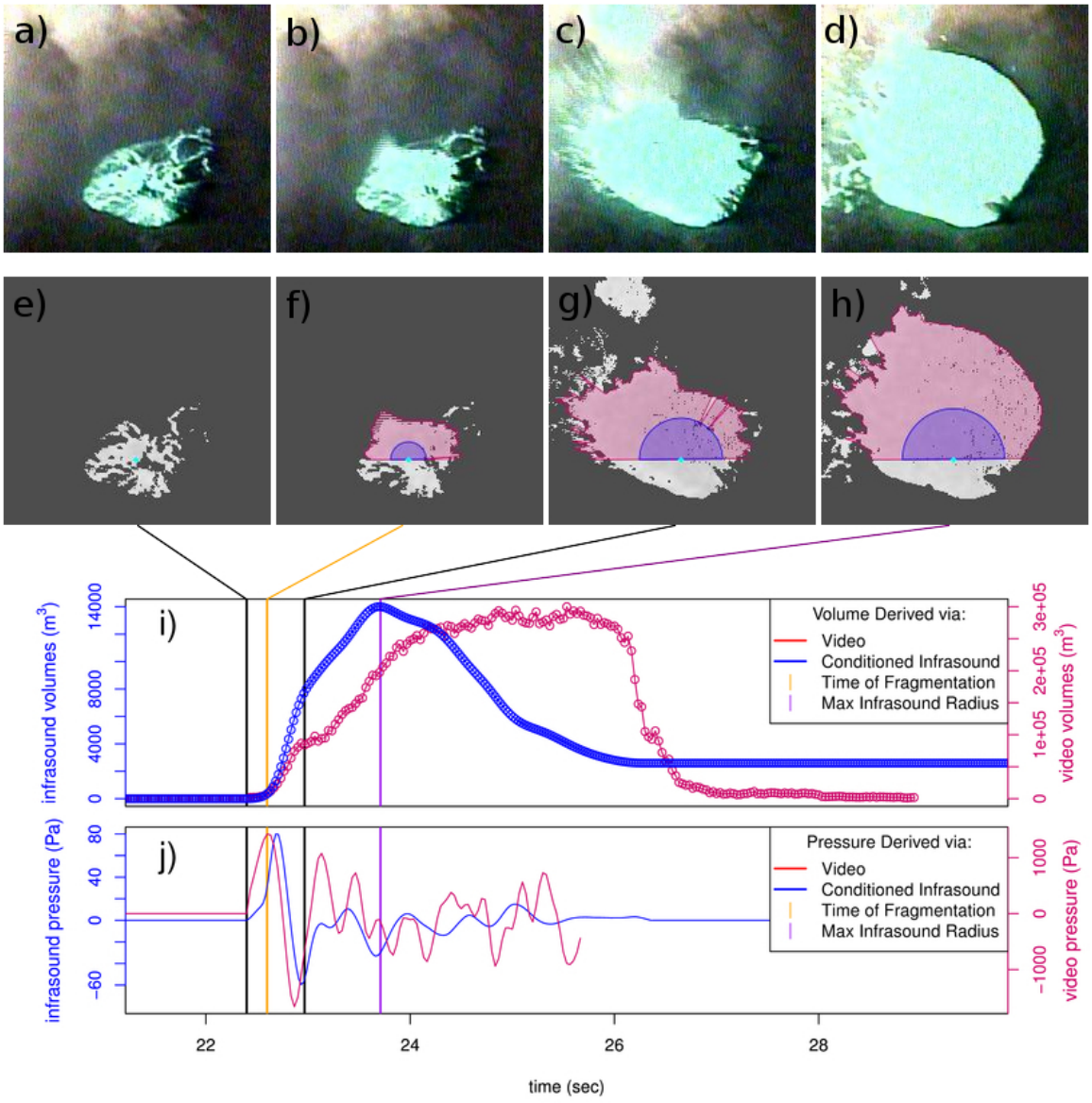


Figure 2.2: Infrasound and video analysis from Event 1. a-d) Raw images. e-h) processed images. Image edge detection is shown in pink and compares with infrasound-inferred hemispherical source projected in blue. i) Infrasound-inferred and video-inferred volumes time histories. j) Conditioned infrasound and video-inferred excess pressure time histories. Note the scale differences on the left and right axes.

Table 2.1: Overview of the 13 infrasound events. All data were recorded in 2006 with the local dates and times given. Signal to noise levels were calculated using RMS values 10 s prior to eruption and RMS values during the eruption (∼4 s). Event duration is the signal window length according to the FWZPZF method (Johnson & Miller, 2014). Bipolar pulse duration is manually picked and contains the infrasound compression and rarefaction. Bursting time is the difference in seconds from observed bubble fragmentation to eruption onset. $R^2$ and slope values correspond to the fitted regression line between infrasound and video volumes during the bipolar pulse.

| **Event** | **Date (2006)** | **Signal to Noise** | | **Infrasound Max Values (Pa)** | | **Event Duration (s)** | | **Bipolar Pulse Duration (s)** | | **Bursting Time (s)** | $V_{inf}$ **to** $V_{vid}$ **Linear Regression** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | RAY | SHK | RAY | SHK | RAY | SHK | RAY | SHK | | $R^2$ Value | Slope |
| 1 | Jan 5, 07:14 | 34.8 | 3.18 | 79.34 | 57.96 | 4.15 | 3.05 | 0.50 | 0.55 | 0.1 | 0.99 | 10.14 |
| 2 | Jan 5, 15:10 | 34.84 | 2.92 | 155.45 | 113.96 | 4.73 | 4.23 | 0.27 | 0.35 | 0.08 | 0.80 | 13.27 |
| 3 | Jan 5, 16:10 | 46.67 | 1.46 | 90.73 | 63.48 | 3.95 | 2.23 | 0.32 | 0.32 | 0.05 | 0.98 | 6.12 |
| 4 | Jan 7, 01:59 | 53.87 | 28.83 | 81.18 | 48.18 | 1.93 | 1.85 | 0.50 | 0.50 | 0.03 | 0.88 | 2.90 |
| 5 | Jan 7, 05:12 | 13.44 | 44.94 | 60.7 | 46.66 | 2.85 | 1.93 | 0.45 | 0.52 | 0.07 | 0.99 | 12.17 |
| 6 | Jan 7, 17:56 | 36.32 | 46.25 | 140.04 | 103.01 | 3.00 | 1.85 | 0.37 | 0.42 | 0.1 | 0.97 | 6.21 |
| 7 | Jan 7, 19:30 | 22.21 | 39.44 | 135.9 | 82.95 | 3.00 | 1.85 | 0.37 | 0.42 | 0.1 | 0.97 | 6.21 |
| 8 | Jan 7, 23:04 | 165.92 | 19.03 | 89.87 | 72.5 | 4.08 | 2.00 | 0.47 | 0.55 | 0.3 | 0.84 | 12.91 |
| 9 | Jan 8, 08:19 | 90.06 | 172.12 | 160.57 | 102.96 | 3.93 | 1.92 | 0.45 | 0.47 | 0.04 | 0.99 | 12.52 |
| 10 | Jan 8, 12:04 | 63.86 | 64.6 | 34.15 | 28.71 | 4.25 | 2.23 | 0.47 | 0.52 | 0.33 | 0.84 | 8.43 |
| 11 | Jan 8, 15:25 | 222.46 | 94.11 | 132.44 | 93.24 | 3.95 | 1.90 | 0.42 | 0.50 | 0.1 | 0.97 | 11.64 |
| 12 | Jan 9, 14:26 | 194.97 | 37.54 | 117.89 | 84.9 | 2.88 | 1.93 | 0.47 | 0.52 | 0.07 | 0.85 | 8.16 |
| 13 | Jan 9, 17:34 | 85.07 | 28.51 | 84.68 | 63.61 | 2.28 | 1.88 | 0.35 | 0.42 | 0.1 | 0.94 | 5.15 |

Infrasound-inferred source volume flux, $q_{inf}$, is the time integration of the infrasound pressure record accounting for retardation time determined by propagation distance $d$ and sound speed $c$. Atmospheric density, $\rho_{atm}$, is used to convert mass flux to volume flux by

$$q_{inf} = \int \frac{2\pi r}{\rho_{atm}} P_{inf}\Big(t + \frac{d}{c}\Big) dt. \tag{2.5}$$

Cumulative volume is the time-integrated $q_{inf}$:

$$V_{inf} = \int q_{inf} dt \tag{2.6}$$

Infrasound-inferred volumes can now be quantitatively compared to video observations of the eruption (Fig. 2.2).

# CHAPTER 3

# RESULTS

## 3.1 Infrasound

Volumes from the two infrasound stations show excellent agreement, both in cumulative size and time coincidence of growth (Fig. 3.1a). Although source-to-receiver propagation times are similar for both stations at 1.02 and 1.03 s (Jones et al., 2008), deployment sites were on nearly opposite sides of the crater (Fig. 1.2). Similar inferred volumes from RAY and SHK indicate a predominantly isotropic radiation pattern, which agrees with previous work (Johnson et al., 2008). However, comparisons of video-inferred and infrasound-inferred volumes show some differences and give insight into how the expanding source volume produces infrasound.

## 3.2 Video and Infrasound

Infrasound-inferred and video-inferred volumes are compared and analyzed for the 13 featured explosions (Fig. 3.1b,c). Video-inferred volumes and infrasound-inferred volumes agree while source volumes are less than $\sim$10000 $m^3$. However, as infrasound volumes increase, the inferred volumes calculated from the two datasets diverge. While this divergence is not clearly marked by a specific time or volume, a divide

is clear after separating infrasound and video volumes into two time periods based upon the shape of the infrasound waveform. All 13 infrasound pressure records exhibit a compression followed by rarefaction, though pulse duration (0.6-1.0 s) and amplitudes (29-160 Pa) are variable. The first time period corresponds to the duration of the infrasound bipolar pulse, during which video and infrasound-derived volumes generally agree. After the bipolar pulse, volumes calculated from video processing continue to increase and disagree with infrasound-inferred volumes, which increase more slowly. Video volumes during the bipolar pulse have $R^2$ values ranging between 0.8 and 0.99 for the 13 events (Fig. 3.2). The scaled relation suggest that video correlates with infrasound volumes throughout the duration of the bipolar pulse, after which video-inferred volume disagrees with infrasound-inferred volume.

Regression lines relating video and infrasound-derived volumes during the infrasound bipolar pulse have slopes ranging between 2.9 and 15.9 over the 13 events (Fig. 3.2). Potential sources of slope variability include both the slight irregularities in bubble growth and physical attributes in the video quality. Gas slugs distend the lava lake surface irregularly, especially after initial fragmentation (e.g., Fig. 2.1). Gerst et al. (2013) classify bubble activity into two types: Type I indicates intact bubble growth before fragmentation while type II occurs when the lava lake is almost immediately punctured by an outflow of gas. In several instances, when bursting times are relatively fast and lava lake distension is minimal (type II), the ejected gas contributes more to the infrasound source. In these circumstances (e.g., events 3, 4, 12), video regression lines often have relatively low slopes. In instances when bursting occurs later (type I), bubble growth dominates both infrasound and video

signals resulting in a higher slope (e.g., events 6, 8, 11). The choice of color to describe the lava lake surface, which is selected based on lighting and visibility (step three in methods), also affects the scaling between visible lava lake components and infrasound signals. In instances with limited visibility or lava lake contrast, the selected color underrepresents the true lava lake surface. Conversely, when contrast or visibility is good, the lava lake color is fully captured and will scale up the video volume slopes. Regardless, for every eruption, a strong scaled relation between infrasound and video exists where $R^2$ values range between 0.8 and 0.99 over the 13 events (Fig. 3.2).

Although video- and infrasound-inferred volumes scale well during the bipolar pulse, video volumes are consistently greater than infrasound volumes. One or the combination of several possibilities result in consistently greater video-inferred volumes. Videos show only the perspective from station SHK and may over represent the true eruptive process. Video data is smoothed (step four in the methods), which slightly exaggerates the bubble radius; however, this error is exacerbated when converting bubble edges to volumes according to Equation 2.1. Video overestimates source volumes during the infrasound coda (after the bipolar pulse) when video tracks inertially traveling ballistics and not the acoustic source.

## 3.3 Infrasound Coda

Infrasound filtered above 0.7 Hz reveals similarities between SHK and RAY that occur during the coda, after the initial bipolar pulse. We define coda as the filtered infrasound signal starting at the first zero-crossing after the bipolar pulse, which

typically is 0.5 to 1.0 s after signal onset. Coda similarities in the time domain include five damped oscillations that diminish to background after 4 s. Peak amplitudes of the coda are similar with the first oscillation's amplitude being about 20 percent of the initial compression peak. Coda spectra are dominated by two distinct peaks in the frequency domain occurring at 1.2 and 1.8 Hz for station RAY and 1.0 and 2.1 Hz for station SHK (Fig. 3.3).

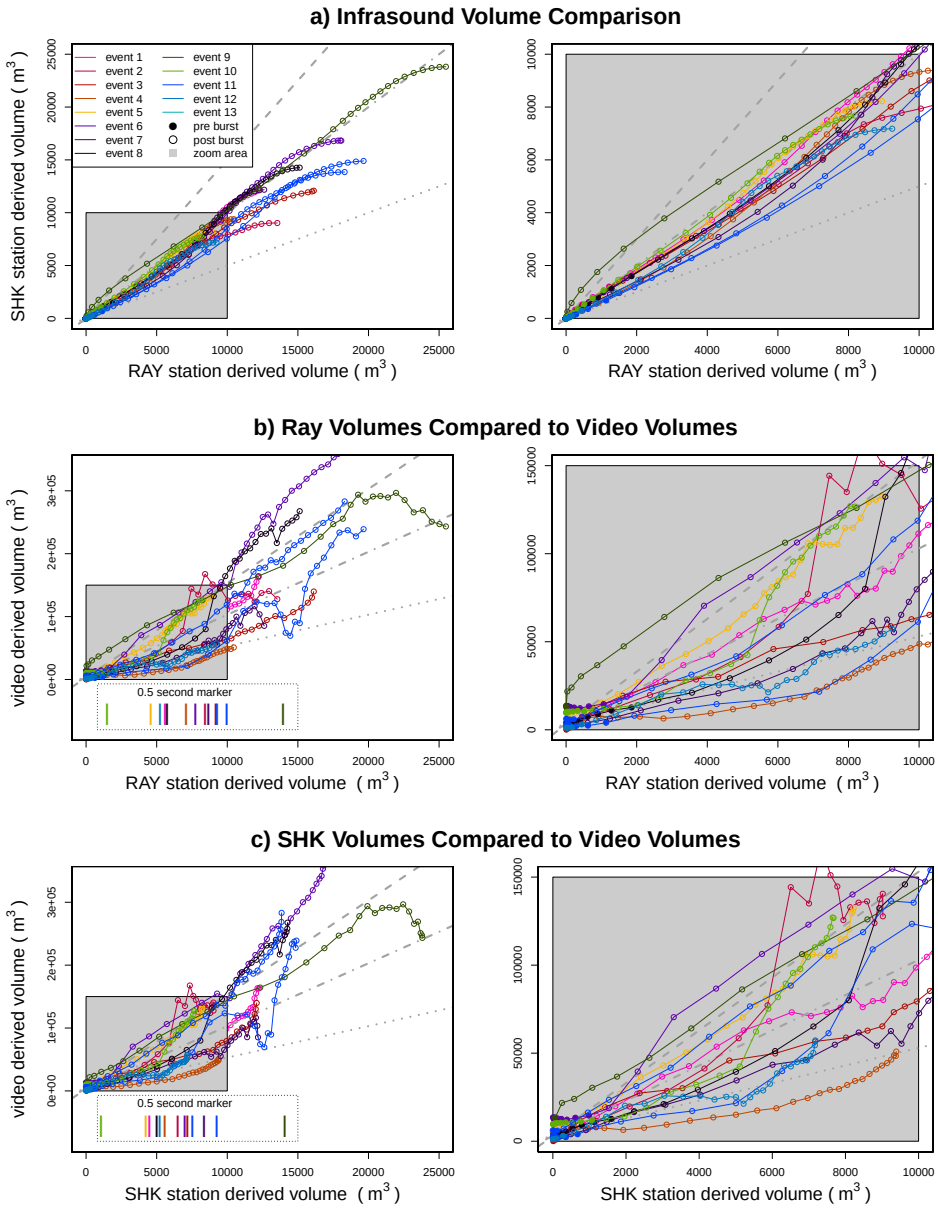


Figure 3.1: Comparisons between video and infrasound. Only values before the maximum infrasound determined volume are shown. Filled circles indicate volumetric measurements when the bubble is expanding 'smoothly 'while open circles indicate values after the bubble has fragmented. Left panels show the entire truncated datasets while right panels show a zoomed-in dataset (gray boxed region). Vertical bars beneath infrasound to video comparisons indicate 0.5 s from infrasound onset for each particular eruption. Infrasound data were decimated to match the sampling rate, or FPS, of the video.

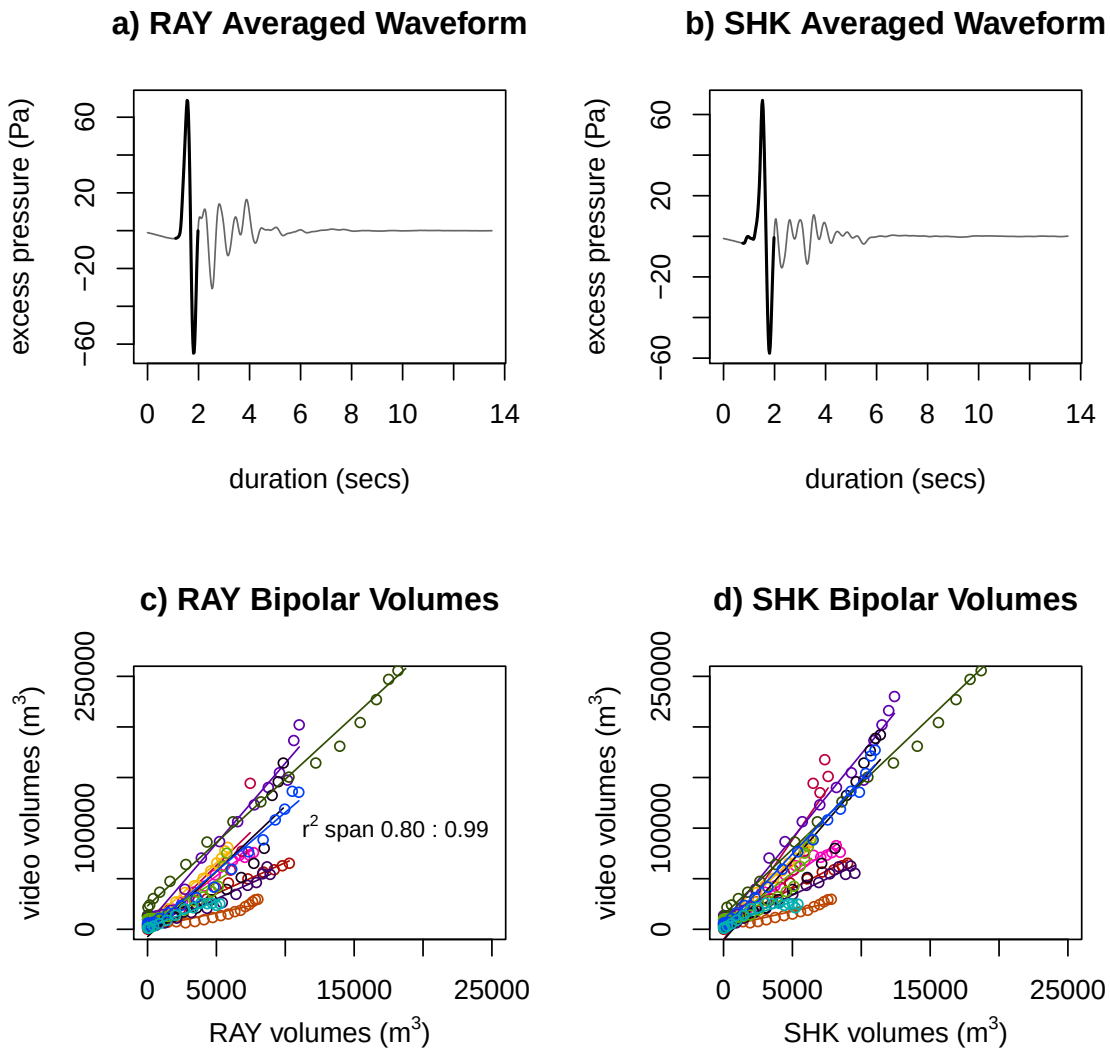


Figure 3.2: Video volume regressions for each eruption. Video volumes calculated during the bipolar pulse (before the second zero crossing) have varying regression slopes when plotted as function of infrasound volume. See Table 2.1 for $R^2$ values and slopes for each eruption

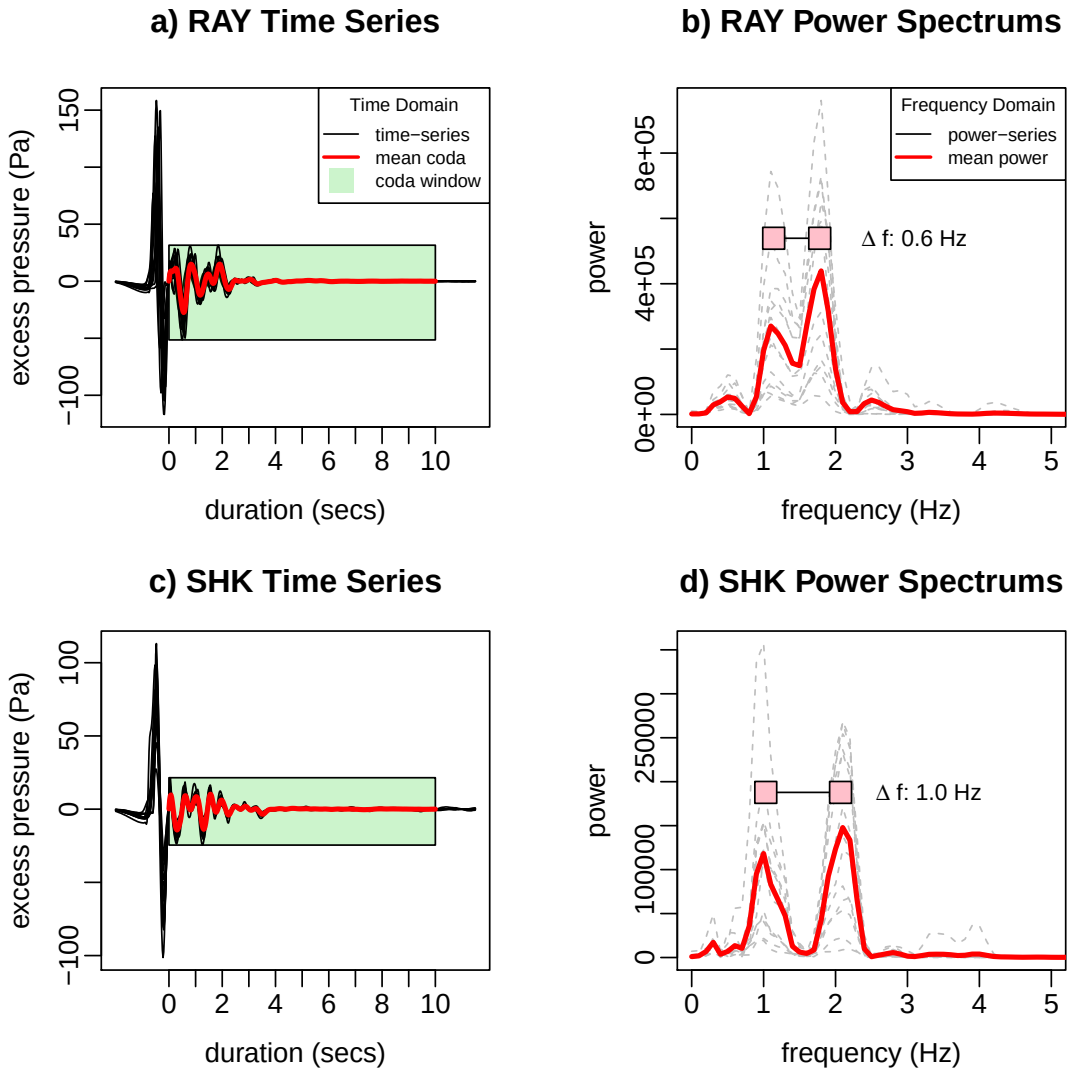


Figure 3.3: Infrasound coda time series and power spectra. All 13 events are plotted on top of each other to show similarities. Codas are windowed (green shading) between the first zero crossing following the bipolar pulse and 10 s after that zero crossing. Power spectra are calculated for time series values within the green region. Mean values are plotted in red.

# CHAPTER 4

# DISCUSSION

## 4.1 Video and Infrasound Analysis and Discussion

Infrasound waveforms at Erebus are controlled by the growth history of large gas slugs at the magma surface. First movement of the lake surface from a rising slug coincides with the initial generation of infrasound signal. Video processing quantifies the visual components of bubble growth that were previously only qualitatively described. The scaling of inferred volumes between both video and infrasound datasets confirm that infrasound is initially generated by bubble growth at the magma surface. Volumetric agreement between datasets persists after the bubble bursts, where large (greater than 10 m) intact portions of the lava lake continue to accelerate and produce infrasound. Though the infrasound bipolar pulse, and inferred volumes, are similar at both stations, the post-fragmentation signal, which is comprised of damped sinusoids, is different at the two stations. An additional acoustic source, associated with resonance of a bubble void and modulated by propagation, is used to explain the infrasound recorded after the expanding bubble ceases to radiate infrasound.

## 4.2 Infrasound Coda Analysis and Discussion

Path effects are used to explain the two frequency peaks in the infrasound coda spectra (Fig. 3.3). Coda spectra are similar across the 13 eruptions for an individual station, but are dissimilar between the two stations. The damped sinusoidal oscillations recorded at both RAY and SHK suggest a resonating source similar to that observed in laboratory experiments by Vidal et al. (2006). The inconsistencies between RAY and SHK suggest infrasound waves traveling from source to RAY experience different propagation effects than those traveling to SHK. Interference from echoes contaminate the coda and must be removed to investigate a potential resonator. After the echoes are deconvolved from the coda, a common frequency between both station's coda is clear that describes a resonator whose geometry matches dimensions of the evacuated magma conduit.

### 4.2.1 Echoes Off the Crater Walls

Array infrasound studies conducted at other volcanoes, e.g., Sakurajima (Yokoo et al., 2014) and Tungurahua (Anderson et al., 2015), have demonstrated that distant echoes off distant topographical features can contaminate coda. The two near-vent infrasound stations at Mount Erebus are not appropriate for array analysis, however, echoes off of near-vent topography are likely to contaminate infrasound coda. A DEM of the crater geometry is used to determine that echoes have a probable influence on infrasound signal.

Crater morphology indicates that coda is influenced by a single strong echo. A simplified model is considered where two Dirac delta functions represent the initial

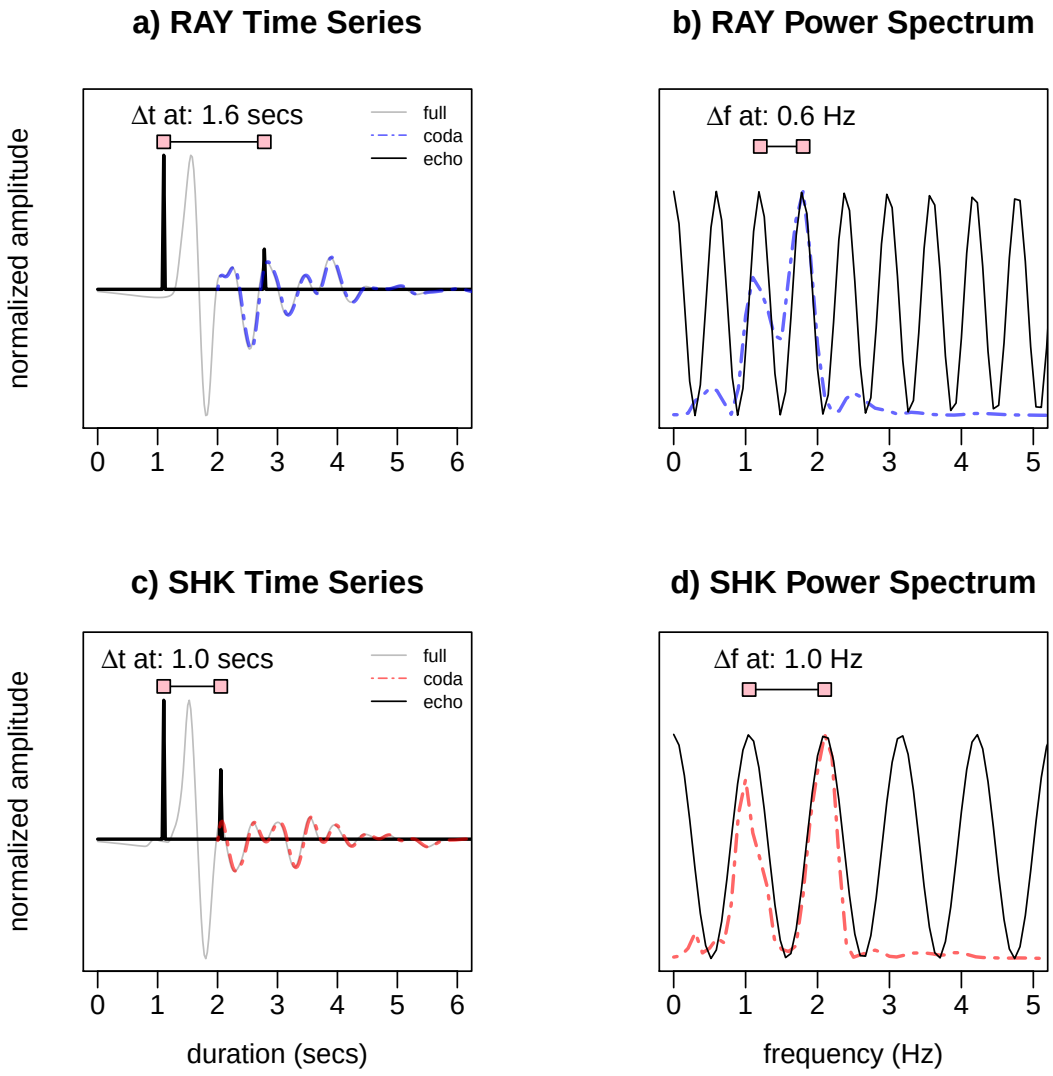


Figure 4.1: Inferring the echo Green's function. Blue and red dashed lines correspond to coda time series and coda frequency spectra. Black lines correspond to time series echo functions and corresponding frequency spectra that best fit the observed coda spectra. Two Dirac delta functions model the impulse followed and the echo. Echo delay times are calculated according to $\Delta t = 1/\Delta f$ from the associated spectrum.

impulse followed by an echo. A time series composed of two delta functions separated by a delay time of $\Delta t$ corresponds to an oscillating frequency spectrum with peaks separated by $1/\Delta t$ (Fig. 4.1). The peaks amplitudes in the frequency domain depend on relative amplitudes of the two impulses in the time domain. It is therefore possible to infer an echo response by calculating the $\Delta f$ from the coda spectra. Green's functions for the echoes are thus generated for the two different stations (Fig. 4.1). Distinct peak frequencies at 1.2 and 1.8 Hz ($\Delta f$ = 0.6 Hz) at RAY and 1.0 and 2.0 Hz ($\Delta f$ = 1.0 Hz) at SHK are attributed to a time series comprising 2 delta functions separated by a lag time of 1.6 and 1.0 s. Total echo travel times are then 2.6 and 2.0 s between Ray Lake and RAY and SHK stations.

Analysis of the DEM validates the echo functions derived from the coda spectra. The DEM is smoothed in order to find surfaces that 1) optimally reflect infrasound and 2) are situated at a distance where two-way travel times match the echo travel time. Low pass filtering the DEM smooths topography that may otherwise misrepresent the broad topographic features of interest. Convolving the DEM with a 2-D Gaussian mask ($\sigma$ = 1 pixels) filters out features with dimensions much less than the wavelengths of interest ($\sim$85 m). Two-way travel distances from the center of Ray Lake to each station are calculated at every point on the filtered DEM. Regions within 12.5 m of the calculated two-way travel distances (864 m for RAY and 628 m for SHK) represent a two-way travel ellipse between source and receiver. Using the law of reflection, the aspects and slopes at every point within each two-way travel time ellipse indicate potential reflectors responsible for the echoes.

Two regions corresponding to the calculated two-way travel distance occur along

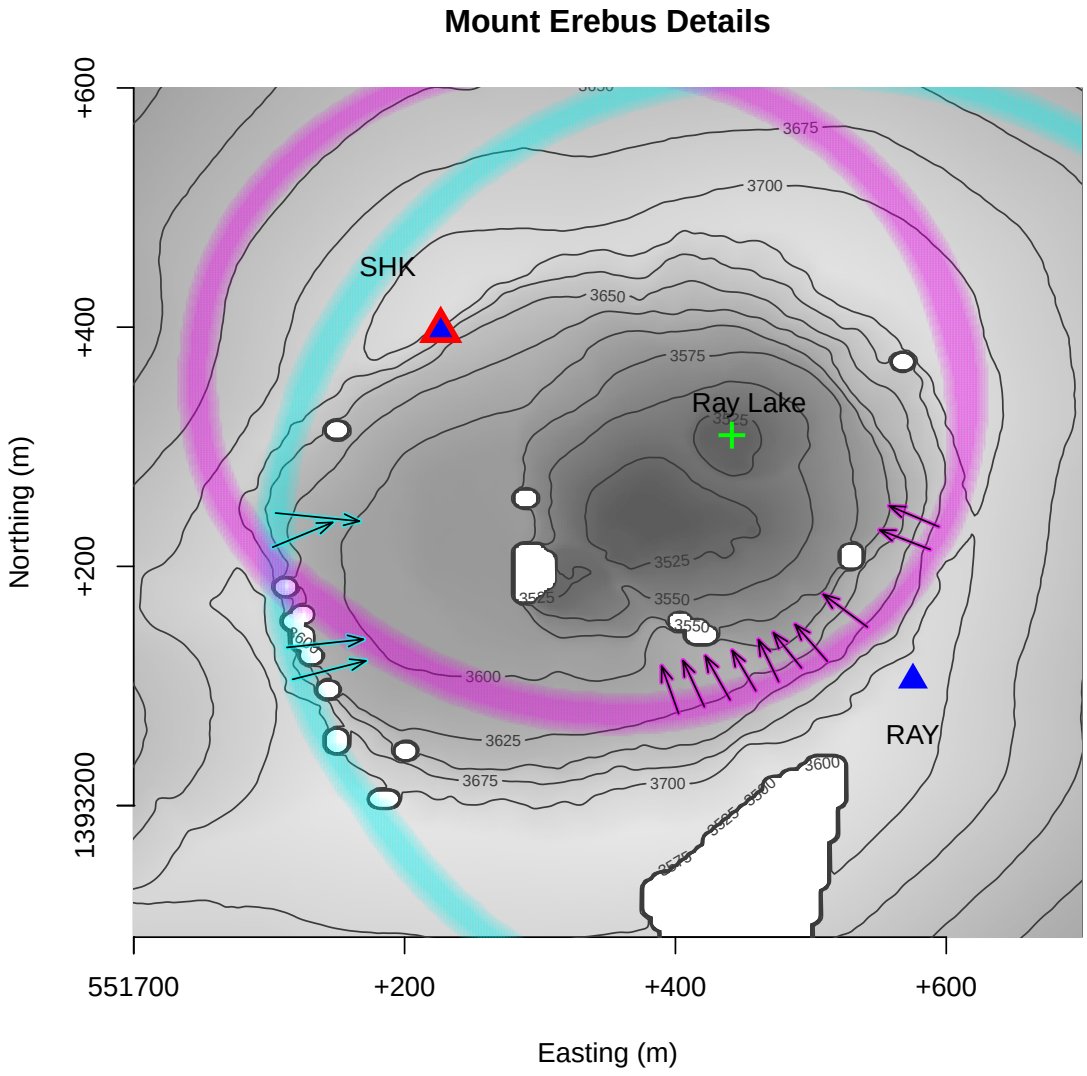


Figure 4.2: Filtered DEM highlighting potential echo locations. Areas in pink and blue overlay locations where two-way travel times from Ray Lake to RAY and SHK are between 2.3 and 1.9 s respectively. Arrows are oriented normal to the slope and are within 20° of the normal to an ideally reflecting surface. Arrows are scaled based on the agreement between the slopes and the ideal vertical angle of incidence.

the steep crater wall (Fig. 4.2). These regions are ideally oriented such that an acoustic source from Ray Lake would reflect towards either infrasound receiver. In total, areas summing to ∼9,000 $m^2$ and ∼26,000 $m^2$ are considered reasonable reflectors as they are within the correct distances and fall within 20° of the ideal incidence angles associated with RAY and SHK respectively (Fig. 4.2 blue and pink arrows). The DEM analysis supports that the high-amplitude pulse, generated by bubble growth and bursting, echoes off the crater walls and reaches RAY and SHK microphones roughly 1.6 and 1.0 s after the direct arrival.

### 4.2.2 Echo Deconvolution

Agreement between DEM analysis and the coda derived echo responses suggests that the echo can be deconvolved from the infrasound coda. An autoregressive moving average (ARMA) operator in the time domain deconvolves the echo from full waveform (bipolar pulse and coda) at both stations.

The time domain echo transfer function, h(t), that contaminates the signal is defined as

$$h(t) = \begin{cases} 1 & \text{if } t = 0 \\ \alpha_e & \text{if } t = t_e \\ 0 & \text{if } t \neq 0 \text{ and } t \neq t_e \end{cases}$$

where $\alpha_e$ is the echo amplitude and $t_e$ is the echo arrival time. Taking the z-transform of $h(t)$ gives

$$H(z) = 1 + \alpha_e z^{t_e}. \tag{4.1}$$

The deconvolution function is defined as the inverse of 4.1,

$$D(z) = \frac{1}{1 + \alpha_e z^{t_e}}, \tag{4.2}$$

and is related to the deconvolved output, Y(z), and contaminated signal input, X(z), by

$$D(z) = \frac{Y(z)}{X(z)}. \tag{4.3}$$

To find the deconvolution ARMA operator, we equate 4.2 and 4.3,

$$Y(z) \times [1 + \alpha_e z^{t_e}] = X(z) \tag{4.4}$$

then take the inverse z-transform before solving for y(t),

$$y(t) = x(t) - \alpha_e y(t - t_e). \tag{4.5}$$

Equation 4.5 is applied once $t > t_e$. The deconvolution is performed over both station's averaged waveform with echo lag times calculated from RAY and SHK spectrum (Fig. 4.1). Echo amplitudes must range between zero and one, where the latter would indicate a perfect reflection. Slopes and aspects of the rough crater walls would not perfectly reflect acoustic energy to either station and therefore have an amplitude less than one. Echoes are scaled such that amplitudes for RAY are less than SHK, which is likely due to the loss in energy with greater travel times. Deconvolution with echo amplitudes of 0.3 and 0.5 for station RAY and

Table 4.1: Echo arrival times and echo amplitude values used in Equation 4.1 through 4.5. Echo arrival times are calculated from coda spectrum. Echo amplitudes are chosen such that SHK values are larger than RAY values and to maximize deconvolved coda similarities.

| station | echo arrival time $t_e$ (s) | echo amplitude $\alpha_e$ |
|---|---|---|
| RAY | 1.7 | 0.3 |
| SHK | 1.0 | 0.5 |

SHK respectively (Table 4.1) produces markedly more similar coda spectra than the original codas contaminated by the echoes (Fig. 4.3). Deconvolved coda spectra at both RAY and SHK are dominated by a single narrow peak at 1.48 Hz (Fig. 4.3).

### 4.2.3 Conduit Resonance

The single peak frequencies at 1.48 Hz for both RAY and SHK correspond to the damped infrasound oscillations that reach background after ∼7 s (Fig. 4.3). Vergniolle and Brandeis (1996) recorded similar acoustic oscillations at Stromboli, which they interpret as bubbles below the magma surface expanding and contracting around their equilibrium volume. Vergniolle & Caplanauerbach (2004) later proposed similar vibrational waves at Shishaldin Volcano in Alaska, where bubbles oscillate once at the surface prior to bursting. In contrast, video from Erebus clearly indicates that infrasound oscillations follow, rather than precede, bubble bursting and the associated bipolar pulse. Notably, these explosions leave behind a sub vertical-walled void (Fig. 4.4). Resonance from the void is considered as the likely mechanism for the damped oscillation.

Resonance occurs in response to pressure perturbations in a bounded region,

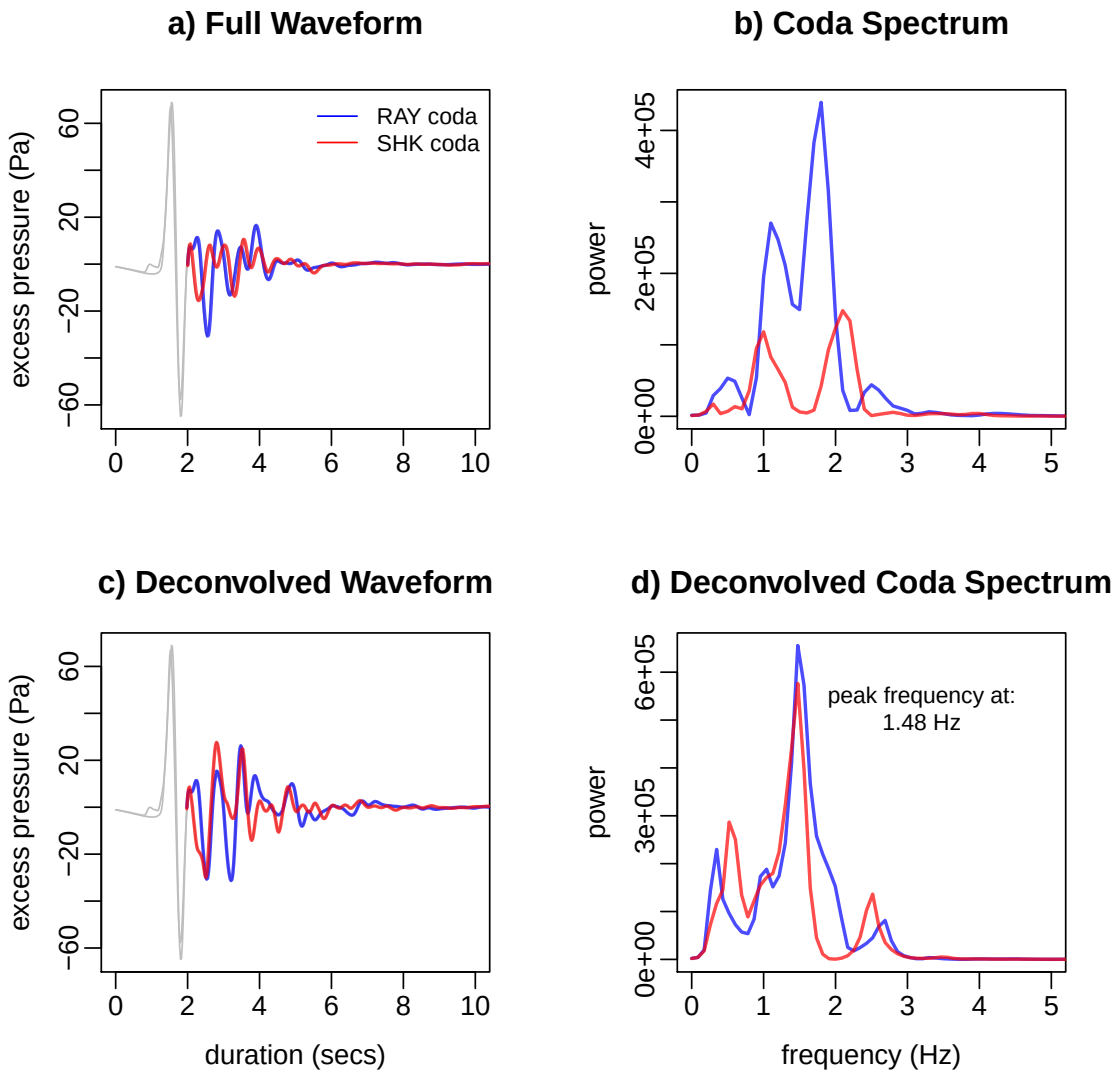


Figure 4.3: Original and deconvolved codas. RAY and SHK waveforms are averaged values of the 13 events.

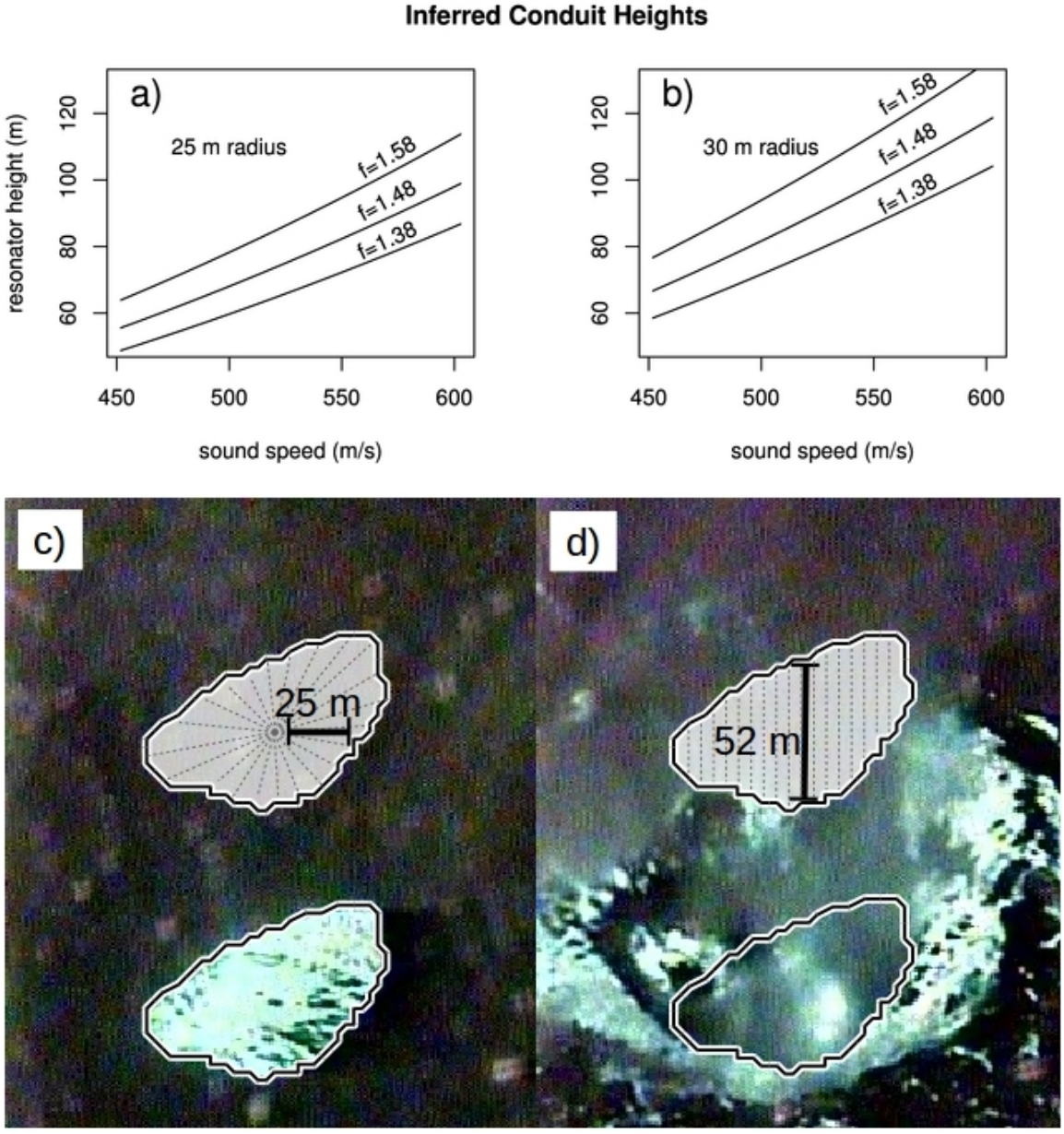


Figure 4.4: Height estimates of evacuated conduit resonator and lava lake images. Panel a) shows heights given a 25 m radius resonator while panel b) heights are calculated with a 30 m radius. Resonating frequencies were taken from the deconvolved frequency peaks from both RAY and SHK codas. Resonator heights are given from the average frequency peaks between both stations. Panel c) shows the full lava lake prior to the arrival of the bubble slug. The radius is taken from Johnson et al. (2008). Panel d) shows the evacuated conduit after the bubble fully bursts.

including magma-filled conduits and gas-filled craters. In both cases, the system behaves as a mass-spring system, where a body of displaced air acts as the mass while the internal pressure is analogous to spring stiffness. Pressure within the resonator rises and falls accordingly until the resonator reaches equilibrium with the ambient pressure. When an acoustic source excites a resonator with a wavelength larger than the cavity's lateral dimension, the fundamental oscillating frequency produced directly relates to the cavity's geometry (Fletcher & Rossing, 1998). For instance, resonators excited by acoustic sources model volcanic conduits at Villarrica volcano, Chile (Goto & Johnson, 2011) and Kilauea, Hawaii (Fee et al., 2010).

Resonating infrasound is attributed to degassing at the magma surface where the signal is modulated by crater morphology (Buckingham & Garces, 1996). For example, both Bessel horn and Helmholtz resonance have been proposed to model continuous 0.5 to 1 Hz tones at Villarrica's crater (Goto & Johnson, 2011; Richardson et al., 2014). Similarly, discrete strombolian-style explosions are proposed as excitation mechanisms for Helmholtz resonance between 0.03 and 0.1 Hz at Kilauea Volcano (Fee et al., 2010). In all three studies, the calculated dimension of the resonators agree with visually inferred measurements of crater size.

We consider Erebus's evacuated conduit a potential Helmholtz resonator because the geometry of the void left by the burst slug is cylindrical in contrast to the flared shape of Villarrica's crater (Richardson et al., 2014). Helmholtz resonance predicts a characteristic frequency that relates to the geometry of the reservoir volume and sound speed. Sound speed in non-homogeneous gas mixtures, such as those at Erebus, are calculated similar to Morrissey et al. & Chouet (2001) by

$$c_{res} = \sqrt{\gamma_{mix} R_{mix} T}, \tag{4.6}$$

where T is the absolute temperature (K) and $\gamma_{mix}$ and $R_{mix}$ are the weighted heat capacity ratios and gas constants (J/kg/K) of the individual gas constituents weighted by their corresponding mole fraction. Gas composition at Erebus is taken from electrochemical measurements by Moussallam et al. (2012), where $H_2O$ and $CO_2$ were found to constitute 48 and 44 mole percent of the plume, respectively. $H_2O$ and $CO_2$ heat capacity ratios vary little across our temperature range and are fixed at 1.324 and 1.281 for 373 K (White, 1999). Gas constants are given at 461.5 and 188.9 J/Kg/K for $H_2O$ and $CO_2$, respectively.

We consider a range of temperatures and corresponding sound speeds (Fig. 4.4) that are reasonable in lava lake volcanic systems. The upper temperature limit is 842 K and is taken from average infrared-inferred temperatures of Ray Lake (Calkins et al., 2008). The lower limit is 473 K and is taken from average temperatures measured from FLIR imagery at Kilauea Volcano, Hawaii (Fee et al., 2010). Thus, the speed of sound ranges between 451 and 602 m/s given the weighted heat capacity ratio, gas constant, and range of temperatures.

A volume calculation of the Ray Lake Helmholtz resonator requires the inferred sound speed and deconvolved coda peak frequency, which for the case of Erebus is determined to be  1.48 Hz. Using similar geometrical assumptions as Goto and Johnson (2011), who consider a resonator with a circular neck of negligible length, then the resonating volume, $v_{res}$, is given by

$$v_{res} = \left(\frac{c_{res}}{2\pi f_{res}}\right)^2 \frac{\pi r_{res}}{1.7} \tag{4.7}$$

where $r_{res}$ is the neck radius (m), $c_{res}$ is the speed of sound (m/s), and f is the resonating frequency (Hz) (Fletcher & Rossing, 1998). For a cylindrical conduit with a radius equal to the neck radius, we estimate the height of the resonator, $h_{res}$, by

$$h_{res} = \left(\frac{c_{res}}{2\pi f_{res}}\right)^2 \frac{1}{1.7 r_{res}}. \tag{4.8}$$

A sound speed of 532 m/s, cylindrical radius of 25 m, and a frequency ranging between 1.38 and 1.58 Hz yields a cylindrical height of 77 m ±10m for the evacuated magma conduit at Erebus. Various cylindrical heights are calculated from the discussed range of sound speeds in Fig. 4.4.

Geometric measurements from imagery of the evacuated conduit agree with these void height values (Fig. 4.4). Given the camera viewing angle and the estimated radius of Ray Lake, the maximum observable conduit depth at station SHK is 52 m. Imagery of the evacuated conduit is often obscured by gas, but when visible the images suggest lava levels drop at least to the observable extent.

# CHAPTER 5

# CONCLUDING REMARKS

Mount Erebus lava lake eruptions exhibit characteristic strombolian attributes and provide an ideal field site to investigate the fundamentals of volcano infrasound source mechanisms. Large gas slugs radially distend the lava lake surface, perturb the atmosphere, and create infrasound. This process produces a characteristic bipolar pulse shape in infrasound recordings, although pulse amplitudes and duration are variable. Once the bubble completely fragments, the inertially driven pyroclasts no longer generate infrasound; however, infrasound signal is still present.

Video and infrasound recorders provide inexpensive and robust means for monitoring volcanic processes and, when quantitatively compared, better describe a typical Mount Erebus lava lake eruption. A simple edge-detection algorithm tracks the expanding edge of the bubble, which is approximated to a 3-D hemispherical acoustic source. Bubble volumes inferred from video correlate with infrasound-inferred volumes during the bipolar pulse. Isolating infrasound unrelated to visual processes allows for a novel investigation into infrasound coda. The spectral content of these codas elucidates two additional acoustic processes: (i) echoes reflecting off crater walls and (ii) Helmholtz resonance from the evacuated conduit. Infrasound records without corresponding video may now be investigated due to the well-understood

eruptive process. Deviations from the characteristic bipolar pulse or coda would indicate changes in the bubble bursting event or in crater morphology.

Going forward, volcanic systems that demonstrate impulsive sources followed by trailing codas, such as those at Villarrica (Goto & Johnson, 2011), Karymsky and Stromboli (Fee & Matoza, 2013), and Tolbachik and Arenal (Johnson, 2003) volcanoes, may warrant similar approaches. Conduit resonance is identified at Villarrica (Goto & Johnson, 2011), Kilauea (Fee et al., 2010), and Shishaldin (Vergniolle & Caplanauerbach, 2004) volcanoes and is likely a common acoustic phenomenon at many others. Propagation effects, such as echoes off steep crater walls, are often ignored in analysis but can modulate coda waveforms. Codas will more accurately represent resonance phenomenon after removing echo contaminations. Tracking echo arrival times and resonating frequencies over time presents a potential means to monitor changes in crater and conduit morphology.

# REFERENCES

Anderson, J., Johnson, J., Steele, A., Ruiz, M., Hall, M. (2015). *Explosions, ballistics, echoes, degassing, and lightning: diverse volcanic processes revealed by infrasound in the July 2013 eruption of Tungurahua.* Manuscript submitted for publication.

Aster, R., McIntosh, W., Kyle, P., Esser, R., Bartel, B., Dunbar, N., . . . Ruiz, M. (2004). *Real-time data received from Mount Erebus Volcano, Antarctica.* Eos Trans. AGU Eos, Transactions American Geophysical Union, 85(10), 97.

Buckingham, M. J., & Garces, M. A. (1996). *Canonical model of volcano acoustics.* J. Geophys. Res.: Solid Earth, 101(B4), 8129-8151.

Calkins, J., Oppenheimer, C., & Kyle, P. (2008). *Ground-based thermal imaging of lava lakes at Erebus volcano, Antarctica.* Journal of Volcanology and Geothermal Research, 177(3), 695-704.

Dibble, R., Kienle, J., Kyle, P., & Shibuya, K. (1984). *Geophysical studies of Erebus volcano, Antarctica, from 1974 December to 1982 January.* New Zealand Journal of Geology and Geophysics, 27(4), 425-455.

Dowling, A. P. (1997). *Steady-State Radiation from Sources, in Encyclopedia of Acoustics, Volume One* (ed M. J. Crocker), John Wiley & Sons, Inc., Hoboken, NJ, USA.

Fee, D., Garcs, M., Patrick, M., Chouet, B., Dawson, P., & Swanson, D. (2010). *Infrasonic harmonic tremor and degassing bursts from Halema'uma'u Crater, Kilauea Volcano, Hawaii.* J. Geophys. Res., 115(B11).

Fee, D., & Matoza, R. S. (2013). *An overview of volcano infrasound: From hawaiian to plinian, local to global.* Journal of Volcanology and Geothermal Research, 249, 123-139.

Firstov, P.P. & Kravchenko, N.M., 1996. *Estimation of the amount of explosive gas released in volcanic eruptions using air waves.* Volcanol. Seismol. 17, 547 560.

Fletcher, N. H., & Rossing, T. D. (1998). *The physics of musical instruments.* New York: Springer.

Gerst, A., Hort, M., Aster, R. C., Johnson, J. B., & Kyle, P. R. (2013). *The first second of volcanic eruptions from the Erebus volcano lava lake, Antarctica-Energies, pressures, seismology, and infrasound.* J. Geophys. Res. Solid Earth, 118(7), 3318-3340.

Gerst, A., Hort, M., Kyle, P. R., & Vge, M. (2008). *4D velocity of Strombolian eruptions and man-made explosions derived from multiple Doppler radar instruments.* Journal of Volcanology and Geothermal Research, 177(3), 648-660.

Goto, A., & Johnson, J. B. (2011). *Monotonic infrasound and Helmholtz resonance at Volcan Villarrica (Chile).* Geophys. Res. Lett., 38(6).

Johnson, J. (2003). *Generation and propagation of infrasonic airwaves from volcanic explosions.* Journal of Volcanology and Geothermal Research, 121(1-2), 1-14.

Johnson, J. B. (2004). *Source location variability and volcanic vent mapping with a small-aperture infrasound array at Stromboli Volcano, Italy.* Bulletin of Volcanology Bull Volcanol, 67(1), 1-14.

Johnson, J., Aster, R., Jones, K. R., Kyle, P., & McIntosh, B. (2008). *Acoustic source characterization of impulsive Strombolian eruptions from the Mount Erebus lava lake.* Journal of Volcanology and Geothermal Research, 177(3), 673-686.

Johnson, J. B., & Ripepe, M. (2011). *Volcano infrasound: A review.* Journal of Volcanology and Geothermal Research, 206(3-4), 61-69.

Johnson, J. B., & Miller, A. J. (2014). *Application of the Monopole Source to Quantify Explosive Flux during Vulcanian Explosions at Sakurajima Volcano (Japan).* Seismological Research Letters, 85(6), 1163-1176.

Jones, K. R., Johnson, J. B., Aster, R., Kyle, P. R., & McIntosh, W. (2008). *Infrasonic tracking of large bubble bursts and ash venting at Erebus Volcano, Antarctica.* Journal of Volcanology and Geothermal Research, 177(3), 661-672.

Lighthill, M. J. (1978). *Waves in fluids.* Cambridge: Cambridge University Press.

Marcillo, O., Johnson, J. B., & Hart, D. (2012). *Implementation, Characterization, and Evaluation of an Inexpensive Low-Power Low-Noise Infrasound Sensor Based on a Micromachined Differential Pressure Transducer and a Mechanical Filter.* J. Atmos. Oceanic Technol., 29(9), 1275-1284.

Matoza, R. S., Fee, D., & Garcs, M. A. (2010). *Infrasonic tremor wavefield of the Pu'u 'O'o crater complex and lava tube system, Hawaii, in April 2007.* J. Geophys. Res. Journal of Geophysical Research, 115(B12).

Matoza, R. S., Fee, D., & Lopez, T. M. (2014). *Acoustic Characterization of Explosion Complexity at Sakurajima, Karymsky, and Tungurahua Volcanoes.* Seismological Research Letters, 85(6), 1187-1199.

Morrissey, M. M., & Chouet, B. A. (1997). *A numerical investigation of choked flow dynamics and its application to the triggering mechanism of long-period events at Redoubt Volcano, Alaska.* J. Geophys. Res.: Solid Earth, 102(B4), 7965-7983.

Morrissey, M., & Chouet, B. (2001). *Trends in long-period seismicity related to magmatic fluid compositions.* Journal of Volcanology and Geothermal Research, 108(1-4), 265-281.

Moussallam, Y., Oppenheimer, C., Aiuppa, A., Giudice, G., Moussallam, M., & Kyle, P. (2012). *Hydrogen emissions from Erebus volcano, Antarctica.* Bulletin of Volcanology Bull Volcanol, 74(9), 2109-2120.

Oppenheimer, C., & Kyle, P. R. (2008). *Probing the magma plumbing of Erebus volcano, Antarctica, by open-path FTIR spectroscopy of gas emissions.* Journal of Volcanology and Geothermal Research, 177(3), 743-754.

Oppenheimer, C., Moretti, R., Kyle, P. R., Eschenbacher, A., Lowenstern, J. B., Hervig, R. L., & Dunbar, N. W. (2011). *Mantle to surface degassing of alkalic magmas at Erebus volcano, Antarctica.* Earth and Planetary Science Letters, 306(3-4), 261-271.

Rowe, C., Aster, R., Kyle, P., Dibble, R., & Schlue, J. (2000). *Seismic and acoustic observations at Mount Erebus Volcano, Ross Island, Antarctica, 1994:1998.* Journal of Volcanology and Geothermal Research, 101(1-2), 105-128.

Richardson, J. P., Waite, G. P., & Palma, J. L. (2014). *Varying seismic-acoustic properties of the fluctuating lava lake at Villarrica volcano, Chile.* J. Geophys. Res.: Solid Earth, 119(7), 5560-5573.

Vergniolle, S., & Brandeis, G. (1996). *Strombolian explosions: 1. A large bubble breaking at the surface of a lava column as a source of sound.* J. Geophys. Res.: Solid Earth, 101(B9), 20433-20447.

Vergniolle, S., & Caplanauerbach, J. (2004). *Acoustic measurements of the 1999 basaltic eruption of Shishaldin volcano, Alaska 2. Precursor to the Subplinian phase.* Journal of Volcanology and Geothermal Research, 137(1-3), 135-151.

Vidal, V., Gminard, J., Divoux, T., & Melo, F. (2006). *Acoustic signal associated with the bursting of a soap film which initially closes an overpressurized cavity.* The European Physical Journal B Eur. Phys. J. B, 54(3), 321-339.

White, F. M. (1999). *Fluid mechanics.* Boston ; Toronto: McGraw-Hill.

Yokoo, A., Suzuki, Y. J., & Iguchi, M. (2014). *Dual Infrasound Sources from a Vulcanian Eruption of Sakurajima Volcano Inferred from Cross-Array Observation.* Seismological Research Letters, 85(6), 1212-1222.

# APPENDIX A

# INFRASOUND SIGNAL PROCESSING FLOWS

Background noise and instrument drift can significantly impact infrasound cumulative volume estimates. Integrating raw waveforms twice, (Eq. 2.6), would erroneously imply a growing volumetric source before the arrival of the gas slug and well after it has fully burst. Conditioning infrasound according to the Finite Window Zero Pressure Zero Flux (FWZPZF) method isolates the source signal without any filtering, which is shown to have adverse affects on cumulative volume analyses (Johnson & Miller, 2014).

Code used to implement the FWZPZF method is below but the general steps are as follows: 1. Pick the beginning of the event (arrival of the bipolar pulse) and select a time range for the end of the event. 2. Fit a trend line from the picked beginning point to every point within the ending time range. 3. Condition the waveform by subtracting each trend line from the corresponding data values between the beginning and ending values. 4. Record the fluxes for every conditioned waveform (Eq. 2.5). 5. Choose the waveform corresponding to the lowest cumulative flux value, then force all values before the beginning and after the corresponding end time to zero.

Although no filters are applied to infer cumulative volumes, infrasound is filtered to investigate the coda. Coda is considered signal after the dilational trough crosses a

zero pressure. However, for several events, the zero crossing occurs well after the the bubble has burst. Investigations into infrasound processes occur after the bubble has burst and therefore consider zero crossing in these instances unreasonable. A two-pole high pass Butterworth filter of 0.6 Hz produces reasonable zero crossings after the dilational trough. Frequency analysis is carried out on these filtered infrasound waveforms using the FFT function in the base R package. Peak amplitudes used to generate the echo Green's function are found after implementing a synchronous stack of the coda waveforms.

```
#########################################################
# ----              Monopole Detrend              ----##
## This is taken from monopole_detrend.m written by ##
# Jeff Johnson for use in R.                          #
#########################################################

monopole_detrend <- function(wv,be,en,en2,k) {

    source("/home/alex/progs/r/linreg.r")

### Inputs ###
    ## wv is initial (unfiltered) waveform
    ## be is the beginning sample
    ## en is ending sample or samples

### Outputs ###
    ## dp is pressure
  if(!exists("md")) {
        md <- NULL
    }

    en_range <- en:en2
    normflux <- rep(NA,length(wv))

    while(en <= en2)
        {
```

```
        wv1 <- wv
        x=c(be,en)
        y=c(wv1[be], wv1[en])
        f <- function(int,slop,x){int + slop*x}
        p <- linreg(x=x, y=y)
        pfit <- sapply(X=1:length(wv1),'f',int=p[1,1],slop=p[2,1])
        trend_variance <- abs((pfit[en]-pfit[be])/(en-be))
        wv2 <- wv1 - pfit
        wv2[c(1:be-1,(en+1):length(wv2))] <- 0
        p_offset <- 0

        if(length(en_range)==1) {
            p_offset = mean(wv2[be:en])
        }
        wv3 = wv2 - p_offset
        wv3[c(1:be-1,(en+1):length(wv2))] <- 0

        normflux[en]=sum(wv3)

        en=en+1
    }

if(length(en_range) == 1) {
    dp = wv3
    e <- c(be, en2)

    md[[k]] <- list(trend_variance,e,dp)
    assign("md", md, envir=.GlobalEnv)

} else {

    ## Second Bit
    flux_indices1 <- which(abs(diff(sign(normflux))) > 1)
    flux_indices2 <- matrix(c(flux_indices1,flux_indices1+1),byrow=TRUE,nrow=2)
    m.normflux <- NULL
    m.normflux <- rbind(m.normflux,normflux[flux_indices2[1,]])
    m.normflux <- rbind(m.normflux,normflux[flux_indices2[2,]])

    Y <- apply(m.normflux,2,FUN=function(x){min(abs(x))})
    I <- apply(m.normflux,2,FUN=function(x){which.min(abs(x))})
```

```
        ## if(length(I) == 0) {
        ##     ##dp <- rep(NA,length.out=length(wv))
        ##     en <- NA
        ## }

        ##rm(bes, ens)

        if(length(I) >= 1) {

            flux_indices = rep(NA,length.out=length(I))
            k = 1
            }
        }

    }

}

            while(k <= length(I)) {
                flux_indices[k] <- flux_indices2[I[k],k]
                monopole_detrend(wv=wv,be=be,en=flux_indices[k],en2=flux_indices[k],k=k)
                ##assign("md", md, envir=.GlobalEnv)
                k=k+1
            }
        }

    }

}
```

# APPENDIX B

# DECONVOLUTION CODE

One R script accomplished the DEM analysis and is found below. No external libraries or functions are required.

```
####################################
#  ---- Echo Decon from Coda ---- #
##                              ##
# Read in a time series for both #
# RAY and SHK stations. Generate #
# an echo based on the delta f  #
# between peaks in the frequency #
#spectrum. Decon the echo respo -#
#-nce from the coda             #
####################################
rm(list=ls())

## read in the mean time series
ray.full.ts <- read.table('~/site/erebus_2006/data/ray_ts.txt')
shk.full.ts <- read.table('~/site/erebus_2006/data/shk_ts.txt')

## read in the mean coda
ray.coda.ts <- read.table('~/site/erebus_2006/data/ray_coda_ts.txt')
shk.coda.ts <- read.table('~/site/erebus_2006/data/shk_coda_ts.txt')

## what are indices corresponding to the bipolar arrival
## these indices are picked manually
r.pad = 45
s.pad = 45
```

```
#############
## INPUTS ##
#############

inf.sps = 40 # samples per second
c = 320 # m/s sound speed
ray.dist = 328 # m distance from source to ray station
shk.dist = 325 # m distance from source to shk station

r.delta.f = 0.60 ##0.60 # Hz between frequency peaks for RAY
s.delta.f = 1.0 # Hz between frequency peaks for SHK

r.echo.dist <- c*r.delta.f # m apprx echo location for RAY
s.echo.dist <- c*s.delta.f # m apprx echo location for SHK


## primary and echo arrival indices
r.prim.indx <- r.pad
r.echo.indx <- round(1/r.delta.f*inf.sps) + r.prim.indx

s.prim.indx <- s.pad
s.echo.indx <- round(1/s.delta.f*inf.sps) + s.prim.indx

## create a greens function that represents an echo
r.echo.coef = 0.5 # RAY echo coef.
s.echo.coef = 0.85 # SHK echo coef.

## RAY
r.green.echo <- rep(0,nrow(ray.full.ts))
r.green.echo[r.prim.indx] <- 1
r.green.echo[r.echo.indx] <- r.echo.coef

## SHK
s.green.echo <- rep(0,nrow(shk.full.ts))
s.green.echo[s.prim.indx] <- 1
s.green.echo[s.echo.indx] <- s.echo.coef

########################
## TIME -> FREQUENCY ##
```

```
#######################

## first set up the frequency axis for both full signal and just coda

## the bipolar and coda
N.f <- length(r.green.echo) # Number of bins
nyq <- inf.sps/2 ## Nyquist
fax.bins.f <- 0:(N.f-1) ## frequency bins
fax.hz.f <- (fax.bins.f*inf.sps/N.f)[1:(N.f/2)] # frequency axis (only
    positive frequencies)

## the coda only
N.c <- nrow(ray.coda.ts)
fax.bins.c <- 0:(N.c-1) ## frequency bins
fax.hz.c <- (fax.bins.f*inf.sps/N.c)[1:(N.c/2)]

## Now calculate the fft
## first the entire infrasound power spectrum
ray.spec <- (fft(ray.full.ts[,2])^2)[1:(N.f/2)]
shk.spec <- (fft(shk.full.ts[,2])^2)[1:(N.f/2)]

## second the coda infrasound power spectrum
ray.coda.spec <- (fft(ray.coda.ts[,2])^2)[1:(N.c/2)]
shk.coda.spec <- (fft(shk.coda.ts[,2])^2)[1:(N.c/2)]

## third the echo power spectrum
r.echo.spec <- (fft(r.green.echo)^2)[1:(N.f/2)]
s.echo.spec <- (fft(s.green.echo)^2)[1:(N.f/2)]

#############
## DECON ##
#############

## we are going to try and do this in the time domain with a loop
## set up the initial time series

## RAY
## Bipolar with coda
ray.decon = ray.full.ts
ray.decon[(r.echo.indx + 1):nrow(ray.full.ts),2] = 0
```

```
## Coda Only
ray.coda.decon = ray.coda.ts
ray.coda.decon[(r.echo.indx - r.pad + 1):nrow(ray.coda.ts),2] = 0

## SHK
## Bipolar with coda
shk.decon = shk.full.ts
shk.decon[(s.echo.indx + 1):nrow(shk.full.ts),2] = 0

## Coda Only
shk.coda.decon = shk.coda.ts
shk.coda.decon[(s.echo.indx - s.pad + 1):nrow(shk.coda.ts),2] = 0

## now lets start a loop for the entire RAY signal
r.i <-r.echo.indx + 1
while(r.i <= nrow(ray.full.ts)) {
    ray.decon[r.i,2] <- ray.full.ts[r.i,2] - (r.echo.coef *ray.decon[r.i-r.
        echo.indx+r.pad,2] )

    r.i = r.i+1
}

## loop for coda RAY signal
r.i <- (r.echo.indx-r.pad) + 1
while(r.i <= nrow(ray.coda.ts)) {
    ray.coda.decon[r.i,2] <- ray.coda.ts[r.i,2] - (r.echo.coef * ray.coda.
        decon[(r.i-(r.echo.indx-r.pad)),2])

    r.i = r.i+1
}


## now a loop for the full SHK signal
s.i <-s.echo.indx + 1
while(s.i <= nrow(shk.full.ts)) {
    shk.decon[s.i,2] <- shk.full.ts[s.i,2] - (s.echo.coef * shk.decon[(s.i-
        s.echo.indx+s.pad),2])

    s.i = s.i+1
}
```

```
## loop for coda SHK signal
s.i <- (s.echo.indx-s.pad) + 1
while(s.i <= nrow(shk.coda.ts)) {
    shk.coda.decon[s.i,2] <- shk.coda.ts[s.i,2] - (s.echo.coef * shk.coda.
        decon[(s.i-(s.echo.indx-s.pad)),2])

    s.i = s.i+1
}

## Move to the frequency domain with an FFT
ray.decon.spec <- (fft(ray.decon[,2])^2)[1:(N.f/2)]
ray.coda.decon.spec <- (fft(ray.coda.decon[,2])^2)[1:(N.c/2)]

shk.decon.spec <- (fft(shk.decon[,2])^2)[1:(N.f/2)]
shk.coda.decon.spec <- (fft(shk.coda.decon[,2])^2)[1:(N.c/2)]


###############
## PLOTTING ##
###############
graphics.off()
close.screen(all.screens=TRUE)

##RAY
##dev.new()
pdf('~/site/erebus_2006/sandbox/full_decon.pdf')

plot.new()
title('Echo Decon from Full Waveform')
split.screen(c(3,2))

screen(1,new=FALSE)
plot(ray.full.ts[,1],ray.full.ts[,2],type='l',xlab=NA,ylab='amplitude (Pa)
    ',col='blue')
lines(shk.full.ts[,1],shk.full.ts[,2],col='red')

screen(2)
plot(fax.hz.f,abs(ray.spec),type='l',xlab=NA,ylab=NA,xlim=c(0,5),col='blue
    ')
lines(fax.hz.f,abs(shk.spec),type='l',col='red')
```

```
screen(3)
plot(ray.full.ts[,1],r.green.echo,type='l',xlab=NA,ylab='amplitude (Pa)',
    col='blue')
lines(shk.full.ts[,1],s.green.echo,col='red')

screen(4)
plot(fax.hz.f,abs(r.echo.spec),type='l',xlab=NA,ylab=NA,xlim=c(0,5),col='
    blue',ylim=c(min(c(abs(s.echo.spec),abs(s.echo.spec))),max(c(abs(s.echo
    .spec),abs(s.echo.spec)))))
lines(fax.hz.f,abs(s.echo.spec),col='red')

screen(5)
plot(ray.decon,type='l',xlab='time (secs)',ylab='amplitude (Pa)',col='blue
    ')
lines(shk.decon,type='l',col='red')

screen(6)
plot(fax.hz.f,abs(ray.decon.spec),type='l',xlab='frequency (Hz)',ylab=NA,
    xlim=c(0,5),col='blue',ylim=c(min(c(abs(ray.decon.spec),abs(shk.decon.
    spec))),max(c(abs(ray.decon.spec),abs(shk.decon.spec)))))
lines(fax.hz.f,abs(shk.decon.spec),col='red')

dev.off()

##coda
##dev.new()
pdf('~/site/erebus_2006/sandbox/coda_decon.pdf')

plot.new()
title('Echo Decon from Coda Waveform')
split.screen(c(2,2))

screen(1,new=FALSE)
plot(ray.coda.ts[,1],ray.coda.ts[,2],type='l',xlab=NA,ylab='amplitude (Pa)
    ',col='blue')
lines(shk.coda.ts[,1],shk.coda.ts[,2],col='red')

screen(2)
plot(fax.hz.c,abs(ray.coda.spec),type='l',xlab=NA,ylab=NA,xlim=c(0,5),col
    ='blue')
```

```
lines(fax.hz.c,abs(shk.coda.spec),col='red')

screen(3)
plot(ray.full.ts[,1],r.green.echo,type='l',xlab=NA,ylab='amplitude (Pa)',
    col='blue')
lines(shk.full.ts[,1],s.green.echo,col='red')

screen(4)
plot(fax.hz.f,abs(r.echo.spec),type='l',xlab=NA,ylab=NA,xlim=c(0,5),col='
    blue',ylim=c(min(c(abs(r.echo.spec),abs(s.echo.spec))),max(c(abs(r.echo
    .spec),abs(s.echo.spec)))))
lines(fax.hz.f,abs(s.echo.spec),col='red')

screen(5)
plot(ray.coda.decon,type='l',xlab='time (secs)',ylab='amplitude (Pa)',col
    ='blue')
lines(shk.coda.decon,type='l',col='red')

screen(6)
plot(fax.hz.c,abs(ray.coda.decon.spec),type='l',xlab='frequency (Hz)',ylab
    =NA,xlim=c(0,5),col='blue')
lines(fax.hz.c,abs(shk.coda.decon.spec),col='red')

dev.off()

\end{verbatim}
\end{vcode}
\label{fig:code}

\chapter{DEM Analysis}\label{app:processingflows}
Two R scripts accomplished the DEM analysis and are found below. Several
    custom built functions are required and found below.

\begin{vcode} % an alternative to 'singlespace', that also shrinks the
             % typewriter font so that it blends better with the text
\begin{verbatim}

#####################################
##----  Find Echo Locations ----##
#                                   #
# Given a DEM and a good distance #
```

```
# formula calculator, we go through #
# and find all the regions that  #
# are potential echo locations!  #
#####################################
rm(list=ls())

library(tiff)
library(raster)

source('~/progs/r/build_gaus.r')
source('~/progs/r/filt2d.r')
source('~/progs/r/calc_dist.r')

###############
### INPUTS ###
###############

c=320##340 ## speed of sound (m/s)
ef=1/0.6 ## 0.9 ## echo delay (Hz)
r.dist=328 ## transient distance from source to ray (m)
s.dist=325 ## transient distance from source to shk (m)

## this is the distance that we will be looking for
r.ed <- c*ef + r.dist ## echo distance for RAY (m)
s.ed <- c*ef + s.dist ## echo distance for SHK (m)

## we need to define a fudge range
fudge=25 ## (m)

## station info
ray.st.old <- c(552320.4066174207,1393270.1165953353)
shk.st.old <- c(551971.6908052489 ,1393562.1693075672 )

x.shift <- -45
y.shift <- 35
ray.st <- c(ray.st.old[1]+x.shift, ray.st.old[2]+y.shift,3766)
shk.st <- c(shk.st.old[1]+x.shift, shk.st.old[2]+y.shift,3774)

## approximate location of ray lake
erebus.loc <- c(552141.7,1393506,3550)
```

```
## read in the DEM
##dem <- readTIFF("/home/alex/site/erebus_2006/images/maps/dem/
    erebus_atm_2001_dem_v5.tif")
##dem <- t(apply(dem, 2, rev))

## make the dem odd so you can filter it
##dem <- dem[-1,]
##dem <- dem[,-1]

## filter the DEM
## first build a gaus
gaus <- build.gaus(x=nrow(dem),y=ncol(dem),sig=1)
## now filter the dem with the gausian matrix
demf <- filt2d(x=dem,mask=gaus)

## do you really want the dem to be filtered?
dem = demf

## some DEM info
ul.x <- 547523.5550433077
ul.y <- 1398911.5184835605
lr.x <- 557029.427655948
lr.y <- 1389381.5765959155


re.calc = FALSE ## do you want to recalculate

if(re.calc == FALSE) {
###############
### READ IN ###
###############

    ## we can read in the data from here
    ray.dmat <- as.matrix(read.table('~/site/erebus_2006/data/
        ray_distance_mat.txt'))
    shk.dmat <- as.matrix(read.table('~/site/erebus_2006/data/
        shk_distance_mat.txt'))
    lake.dmat <- as.matrix(read.table('~/site/erebus_2006/data/
        lake_distance_mat.txt'))

}if(re.calc == TRUE) {
```

```
##############################################
### DISTANCES FROM RAY, SHK, and Lave Lake ###
##############################################

    ## we need a sequence of x, y, and z values
    ## x, y, and z locatiosn will NOT change through our calculations
    ## only the z values will change
    xlocs <- rep(seq(ul.x,lr.x,length.out=dim(dem)[1]),ncol(dem))
    ylocs <- rep(seq(lr.y,ul.y,length.out=dim(dem)[2]),each=nrow(dem))
    zlocs <- as.vector(dem)

    ## now we can apply the distance formula (?)
    ## set up the vectors to contain distances
    ray.dvec <- rep(NA,length(xlocs))
    shk.dvec <- rep(NA,length(xlocs))
    lake.dvec <- rep(NA,length(xlocs))

    ## percent calculator
    perc=0

    i=1
    while(i<=length(xlocs)) {

        ## x, y and zs for ray distances
        xr=c(ray.st[1], xlocs[i])
        yr=c(ray.st[2], ylocs[i])
        zr=c(ray.st[3], zlocs[i])

        ## x, y and zs for shk distances
        xs=c(shk.st[1], xlocs[i])
        ys=c(shk.st[2], ylocs[i])
        zs=c(shk.st[3], zlocs[i])

        ## x, y and zs for shk distances
        xl=c(erebus.loc[1], xlocs[i])
        yl=c(erebus.loc[2], ylocs[i])
        zl=c(erebus.loc[3], zlocs[i])

        ray.dvec[i] <- calc.dist(x=xr,y=yr,z=zr,dim=3)
        shk.dvec[i] <- calc.dist(x=xs,y=ys,z=zs,dim=3)
        lake.dvec[i] <- calc.dist(x=xl,y=yl,z=zl,dim=3)
```

```
        ## calculate the percent done
        perc.new <- round(i/length(xlocs),2)*100
        if(perc.new > perc) {
            print(paste(perc.new, '% done',sep=""))
        }
        perc=perc.new

        i=i+1
    }

    ## now convert dvec into a matrix
    ray.dmat <- matrix(ray.dvec,ncol=ncol(dem),nrow=nrow(dem))
    shk.dmat <- matrix(shk.dvec,ncol=ncol(dem),nrow=nrow(dem))
    lake.dmat <- matrix(lake.dvec,ncol=ncol(dem),nrow=nrow(dem))

    ## write these matrices?
    ##write.table(ray.dmat,'~/site/erebus_2006/data/ray_distance_mat.txt',
        quote=FALSE,row.names=FALSE,col.names=FALSE)
    ##write.table(shk.dmat,'~/site/erebus_2006/data/shk_distance_mat.txt',
        quote=FALSE,row.names=FALSE,col.names=FALSE)
    ##write.table(lake.dmat,'~/site/erebus_2006/data/lake_distance_mat.txt
        ',quote=FALSE,row.names=FALSE,col.names=FALSE)

    ## write these matrices?
    write.table(ray.dmat,'~/site/erebus_2006/data/ray_distance_mat_test.txt
        ',quote=FALSE,row.names=FALSE,col.names=FALSE)
    write.table(shk.dmat,'~/site/erebus_2006/data/shk_distance_mat_test.txt
        ',quote=FALSE,row.names=FALSE,col.names=FALSE)
    write.table(lake.dmat,'~/site/erebus_2006/data/lake_distance_mat_test.
        txt',quote=FALSE,row.names=FALSE,col.names=FALSE)

}

## now we have to do add the distances from the lava lake to each station
str <- lake.dmat + ray.dmat ## source to ray station
sts <- lake.dmat + shk.dmat ## source to shk station

## notice that we shrunk the dem by one row and one col for filtering
## now we have to shrink the str and sts
str = str[-1,]
```

```
str = str[,-1]
sts = sts[-1,]
sts = sts[,-1]

## now hone in on the areas where the echo could occur
str.echo <- str
str.echo[which(str.echo >= (r.ed + fudge))] = NA
str.echo[which(str.echo <= (r.ed - fudge))] = NA

sts.echo <- sts
sts.echo[which(sts.echo >= (r.ed + fudge))] = NA
sts.echo[which(sts.echo <= (r.ed - fudge))] = NA

## limit the DEM to only the areas overlapping the echo locals
dem.echo.ray <- dem
dem.echo.ray[which(is.na(str.echo))] = NA
dem.echo.ray <- matrix(dem.echo.ray,nrow=nrow(dem),ncol=ncol(dem))

dem.echo.shk <- dem
dem.echo.shk[which(is.na(sts.echo))] = NA

## okay now we need to do some dem analysis
## using the terrain function in the raster package

## we need to turn the dems under question into raster objects
ras.echo.ray <- raster(t(dem.echo.ray[,ncol(dem.echo.ray):1]),xmn=ul.x,xmx
    =lr.x,ymn=lr.y,ymx=ul.y)
ras.echo.shk <- raster(t(dem.echo.shk[,ncol(dem.echo.shk):1]),xmn=ul.x,xmx
    =lr.x,ymn=lr.y,ymx=ul.y)

## now need to get the projections.
## not sure if this really matters or not
projection(ras.echo.ray)=CRS("+init=epsg:27700")
projection(ras.echo.shk)=CRS("+init=epsg:27700")

## now we can use terrain() to calculate aspect and slope
echo.ray.slope <- terrain(ras.echo.ray, opt='slope')
echo.ray.aspct <- terrain(ras.echo.ray, opt='aspect',unit='radians')

echo.shk.slope <- terrain(ras.echo.shk, opt='slope')
echo.shk.aspct <- terrain(ras.echo.shk, opt='aspect',unit='radians')
```

```
## Set parameters to plot arrows of slope and aspect
##  we need to get the x1 and y1 values for the arrows() function

## lets use the aspect and slope as a matrix for this calculation
ray.aspct <- as.matrix(echo.ray.aspct) ## ray station
ray.slope <- as.matrix(echo.ray.slope)
shk.aspct <- as.matrix(echo.shk.aspct) ## shk station
shk.slope <- as.matrix(echo.shk.slope)

## we need to do the reverse of t(x[,ncol(x):1])
## which is *maybe* t(x1)[,ncol(t(x1)):1]
ray.aspct <- t(ray.aspct)[,ncol(t(ray.aspct)):1] ## ray station
ray.slope <- t(ray.slope)[,ncol(t(ray.slope)):1]
shk.aspct <- t(shk.aspct)[,ncol(t(shk.aspct)):1] ## shk station
shk.slope <- t(shk.slope)[,ncol(t(shk.slope)):1]

## now lets define the x0 and y0 values for both stations
## basically what we do is take all the x and y values associated
## with the non-na values of the aspct and slope values, multiply them
## by the resolution and then add the x and y offsets
## these will eventually turn into x0 and y0 values for
## the arrows function
## x0,y0,x1,y1 will all be vectors
x0.ray <- (row(ray.aspct)[!is.na(ray.aspct)] * diff(seq(ul.x,lr.x,length.
    out=dim(dem)[1]+1))[1]) + ul.x
y0.ray <- (col(ray.aspct)[!is.na(ray.aspct)] * diff(seq(lr.y,ul.y,length.
    out=dim(dem)[2]+1))[1]) + lr.y
## z0 are just the dem values at the non-na points
z0.ray <- dem.echo.ray[which(!is.na(ray.aspct))]

x0.shk <- (row(shk.aspct)[!is.na(shk.aspct)] * diff(seq(ul.x,lr.x,length.
    out=dim(dem)[1]+1))[1]) + ul.x
y0.shk <- (col(shk.aspct)[!is.na(shk.aspct)] * diff(seq(lr.y,ul.y,length.
    out=dim(dem)[2]+1))[1]) + lr.y
## z0 are just the dem values at the non-na points
z0.shk <- dem.echo.shk[which(!is.na(shk.aspct))]

## now we need the changes in x and y for each station
## we are going to normalize everything to one
alpha = 1
```

```
## delta ray x,y,z
drx <- alpha*sin(ray.aspct)
drx <- drx[which(!is.na(drx))]

dry <- alpha*cos(ray.aspct)
dry <- dry[which(!is.na(dry))]

## now for delta zs, use dz=slope*dx
## we need the slope as a vector
rsv <- ray.slope[which(!is.na(ray.slope))] ## ray slope vector
drz <- rsv*drx

## delta shk x,y
dsx <- alpha*sin(shk.aspct)
dsx <- dsx[which(!is.na(dsx))]

dsy <- alpha*cos(shk.aspct)
dsy <- dsy[which(!is.na(dsy))]

## now for delta zs, use dz=slope*dx
## we need the slope as a vector
ssv <- shk.slope[which(!is.na(shk.slope))] ## shk slope vector
dsz <- ssv*dsx

## finally we can calculate the x1,y1 values
x1.ray <- x0.ray+(drx)
y1.ray <- y0.ray+(dry)
z1.ray <- z0.ray+(drz)

x1.shk <- x0.shk+(dsx)
y1.shk <- y0.shk+(dsy)
z1.shk <- z0.shk+(dsz)


####################
### WRITE FILES ###
####################

##the filtered dem
save(demf,file='~/site/erebus_2006/data/dem_filt.RData')
```

```
## the source to receiver (shk or ray) matrices
write.table(str.echo,'~/site/erebus_2006/data/str_echo_test.txt',quote=
    FALSE,row.names=FALSE,col.names=FALSE)
write.table(sts.echo,'~/site/erebus_2006/data/sts_echo.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)

## the x0,y0,z0 and x1,y1,z1 locations for the arrows
## ray
write.table(x0.ray,'~/site/erebus_2006/data/x0_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)
write.table(y0.ray,'~/site/erebus_2006/data/y0_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)
write.table(z0.ray,'~/site/erebus_2006/data/z0_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)
write.table(x1.ray,'~/site/erebus_2006/data/x1_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)
write.table(y1.ray,'~/site/erebus_2006/data/y1_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)
write.table(z1.ray,'~/site/erebus_2006/data/z1_ray_test.txt',quote=FALSE,
    row.names=FALSE,col.names=FALSE)

## shk
write.table(x0.shk,'~/site/erebus_2006/data/x0_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)
write.table(y0.shk,'~/site/erebus_2006/data/y0_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)
write.table(z0.shk,'~/site/erebus_2006/data/z0_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)
write.table(x1.shk,'~/site/erebus_2006/data/x1_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)
write.table(y1.shk,'~/site/erebus_2006/data/y1_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)
write.table(z1.shk,'~/site/erebus_2006/data/z1_shk.txt',quote=FALSE,row.
    names=FALSE,col.names=FALSE)



###################################
# ----   Plot Echoes     ---- #
##                           ##
```

```
# This is an effort to only read #
# in what we need in order to #
# plot all the necessary aspects #
# in the dem analysis         #
###################################
rm(list=ls())

library(tiff)

source('~/progs/r/build_gaus.r')
source('~/progs/r/filt2d.r')

###############
### INPUTS ###
###############
## load in the filtered dem
load('~/site/erebus_2006/data/dem_filt.RData')
dem=demf
dem[which(dem==-9999.000)] = NA

## some DEM info
ul.x <- 547523.5550433077
ul.y <- 1398911.5184835605
lr.x <- 557029.427655948
lr.y <- 1389381.5765959155

## station info
ray.st.old <- c(552320.4066174207,1393270.1165953353)
shk.st.old <- c(551971.6908052489 ,1393562.1693075672 )

x.shift <- -45
y.shift <- 35
ray.st <- c(ray.st.old[1]+x.shift, ray.st.old[2]+y.shift,3766)
shk.st <- c(shk.st.old[1]+x.shift, shk.st.old[2]+y.shift,3774)

## approximate location of ray lake
erebus.loc <- c(552141.7,1393506,3550)

## read in the source to receiver echo matrices
str.echo <- read.table('~/site/erebus_2006/data/str_echo_test.txt')
sts.echo <- read.table('~/site/erebus_2006/data/sts_echo.txt')
```

```
## read in the x0,y0 and x1,y1 locations for the arrows
x0.ray <- read.table('~/site/erebus_2006/data/x0_ray_test.txt')
y0.ray <- read.table('~/site/erebus_2006/data/y0_ray_test.txt')
z0.ray <- read.table('~/site/erebus_2006/data/z0_ray_test.txt')
x1.ray <- read.table('~/site/erebus_2006/data/x1_ray_test.txt')
y1.ray <- read.table('~/site/erebus_2006/data/y1_ray_test.txt')
z1.ray <- read.table('~/site/erebus_2006/data/z1_ray_test.txt')

x0.shk <- read.table('~/site/erebus_2006/data/x0_shk.txt')
y0.shk <- read.table('~/site/erebus_2006/data/y0_shk.txt')
z0.shk <- read.table('~/site/erebus_2006/data/z0_shk.txt')
x1.shk <- read.table('~/site/erebus_2006/data/x1_shk.txt')
y1.shk <- read.table('~/site/erebus_2006/data/y1_shk.txt')
z1.shk <- read.table('~/site/erebus_2006/data/z1_shk.txt')


##################################
### RADIALLY FIND ECHO VALUES ###
##################################

## first find the center
ray.cent <- c(mean(x0.ray[,1]), mean(y0.ray[,1]))
shk.cent <- c(mean(x0.shk[,1]), mean(y0.shk[,1]))

## define a radial step or delta angle
d.angle <- 4 * pi/180 ## in radians

## what is the initial angle to use
angles = seq(0,2*pi,by=d.angle)

## define a maximum line value
## first find the lower left and uper right points of the echo locations
rll <- c(min(x0.ray[,1]),min(y0.ray[,1]))
rur <- c(max(x0.ray[,1]),max(y0.ray[,1]))

sll <- c(min(x0.shk[,1]),min(y0.shk[,1]))
sur <- c(max(x0.shk[,1]),max(y0.shk[,1]))

## now find the distance between these points
ray.max <- sqrt( (rur[1] - rll[1])^2 + (rur[2] - rll[2])^2 )
```

```
shk.max <- sqrt( (sur[1] - sll[1])^2 + (sur[2] - sll[2])^2 )

## we use a loop for the rest of the midpoint calculations
## loop over all the angles to find the middle of the echo locations

## lets start a plot to note the progress
plot(x0.ray[,1],y0.ray[,1])
points(ray.cent[1],ray.cent[2],pch='+',col='blue',cex=2)

## set up a object to hold the ray echo midpoints calculated in the loop
ray.echo.mids = NULL
ray.echo.inds = NULL

shk.echo.mids = NULL
shk.echo.inds = NULL

## first define a starting angle
angle= 0 * pi/180

j=1
while(j<=length(angles)) {

    ## what angle will we use in the following calculations
    angle=angles[j]

    ## now find the far out point at the angle given and max distance
    ## from the center
    r.fop.x <- ray.cent[1] + (ray.max*cos(angle))
    r.fop.y <- ray.cent[2] + (ray.max*sin(angle))
    r.fop <- c(r.fop.x,r.fop.y)

    s.fop.x <- shk.cent[1] + (shk.max*cos(angle))
    s.fop.y <- shk.cent[2] + (shk.max*sin(angle))
    s.fop <- c(s.fop.x,s.fop.y)


    ## now find the fitted points between the center and the
    ## point away from the center at the given angle
    r.fit.p <- approx(x=c(ray.cent[1],r.fop[1]),y=c(ray.cent[2],r.fop[2]),
        n=500)
```

```
s.fit.p <- approx(x=c(shk.cent[1],s.fop[1]),y=c(shk.cent[2],s.fop[2]),
    n=500)

## calculate the distance at each fitted point to all the ray echo
    locations
r.min.dists <- rep(0,length(r.fit.p$x))
s.min.dists <- rep(0,length(s.fit.p$x))

i=1
while(i<=length(r.min.dists)) {
    r.min.dists[i] <- min(abs(sqrt( (x0.ray[,1] - r.fit.p$x[i])^2 + (y0
        .ray[,1] - r.fit.p$y[i])^2 )))
    s.min.dists[i] <- min(abs(sqrt( (x0.shk[,1] - s.fit.p$x[i])^2 + (y0
        .shk[,1] - s.fit.p$y[i])^2 )))

    i=i+1
}

## now find the 'region' where those distances are very small
r.diff.dists <- diff(r.min.dists)
s.diff.dists <- diff(s.min.dists)

## we are gonna have to have some kind of 'tolerance'
## and some kind of distance threshold away from the center
thresh.c = 100
thresh.f = 800
tol = 0.3

## fit range are the values of the fitted line that fit with range of
    the echo
## location edges according to the threshold and tolerance above
r.fit.range <- which( abs((r.diff.dists - 0)) <= tol & (1:length(r.diff
    .dists)) > thresh.c & (1:length(r.diff.dists)) < thresh.f )
s.fit.range <- which( abs((s.diff.dists - 0)) <= tol & (1:length(s.diff
    .dists)) > thresh.c & (1:length(s.diff.dists)) < thresh.f )

## find the mean fitted range. This will be the closest point to the
    middle of the
## echo areas
r.fit.mean <- round(mean(r.fit.range))
s.fit.mean <- round(mean(s.fit.range))
```

```
    ## find the echo index that is closest to this fitted mean
    ray.echo.ind <- which.min(abs(r.fit.p$x[r.fit.mean] - x0.ray[,1]) + abs
        (r.fit.p$y[r.fit.mean] - y0.ray[,1]))
    ray.echo.inds <- c(ray.echo.inds,ray.echo.ind)

    shk.echo.ind <- which.min(abs(s.fit.p$x[s.fit.mean] - x0.shk[,1]) + abs
        (s.fit.p$y[s.fit.mean] - y0.shk[,1]))
    shk.echo.inds <- c(shk.echo.inds,shk.echo.ind)

    ## quick plotting
    points(ray.cent[1],ray.cent[2],pch='+',col='blue',cex=2)
    lines(x=c(ray.cent[1],s.fop[1]),y=c(ray.cent[2],r.fop[2]),lty=2,col='
        red')
    points(x0.ray[,1][ray.echo.ind],y0.ray[,1][ray.echo.ind],col='blue',pch
        =16)

    j=j+1
}

## HELLS BELLS
## find the x0,y0 and x1,y1 values associated with the index values

x0.ray.d <- x0.ray[,1][ray.echo.inds]
y0.ray.d <- y0.ray[,1][ray.echo.inds]
z0.ray.d <- z0.ray[,1][ray.echo.inds]

x0.shk.d <- x0.shk[,1][shk.echo.inds]
y0.shk.d <- y0.shk[,1][shk.echo.inds]
z0.shk.d <- z0.shk[,1][shk.echo.inds]

x1.ray.d <- x1.ray[,1][ray.echo.inds]
y1.ray.d <- y1.ray[,1][ray.echo.inds]
z1.ray.d <- z1.ray[,1][ray.echo.inds]

x1.shk.d <- x1.shk[,1][shk.echo.inds]
y1.shk.d <- y1.shk[,1][shk.echo.inds]
z1.shk.d <- z1.shk[,1][shk.echo.inds]


##############################
```

```
### CALCULATE SOME ANGLES ###
#############################

## lets do this over a loop

##############
### INPUTS ###
##############


## these are points used in every calculation in the loop
## they are source and receiver (ray lake and ray station) locs
p2 <- erebus.loc
p3.r <- ray.st
p3.s <- shk.st

## set up a vector of source to echo location to ray 's.e0.r.xy'
s.e0.r.xy <- rep(0,length(ray.echo.inds))
s.e0.s.xy <- rep(0,length(shk.echo.inds))

## set up a vector of source to normal dem location to ray 's.e1.r.xy'
s.e0.e1.r <- rep(0,length(ray.echo.inds))
s.e0.e1.s <- rep(0,length(shk.echo.inds))

## set up a vector of scaling lengths for all the vectors
s.lengths.r <- rep(0,length(ray.echo.inds))
s.lengths.s <- rep(0,length(shk.echo.inds))

i=1
while(i<length(ray.echo.inds)) {



###################
### XY ANALYSIS ###
###################

    ## first define two vectors based on the three points
    ## source, echo location, receiver (p2,p1,p3)
    p1.r <- c(x0.ray[,1][ray.echo.inds[i]], y0.ray[,1][ray.echo.inds[i]],z0
        .ray[,1][ray.echo.inds[i]])
```

```
    p1.s <- c(x0.shk[,1][shk.echo.inds[i]], y0.shk[,1][shk.echo.inds[i]],z0
        .shk[,1][shk.echo.inds[i]])

    ## now make two vectors from the three points
    v1.r <- c( (p1.r[1]-p2[1]),(p1.r[2]-p2[2]) )
    v2.r <- c( (p1.r[1]-p3.r[1]),(p1.r[2]-p3.r[2]) )

    v1.s <- c( (p1.s[1]-p2[1]),(p1.s[2]-p2[2]) )
    v2.s <- c( (p1.s[1]-p3.s[1]),(p1.s[2]-p3.s[2]) )

    ## calculate the angle between v1 and v2 in radians
    theta.r <- acos( (v1.r%*%v2.r)/(sqrt(sum(v1.r^2))*sqrt(sum(v2.r^2))) )
    theta.s <- acos( (v1.s%*%v2.s)/(sqrt(sum(v1.s^2))*sqrt(sum(v2.s^2))) )

    s.e0.r.xy[i] <- theta.r
    s.e0.s.xy[i] <- theta.s

###########
###########

    ## while we are here, lets calculate the source to normal dem location
        angles
    ## define some new points
    ## the normal dem location
    p4.r <- c(x1.ray[,1][ray.echo.inds[i]], y1.ray[,1][ray.echo.inds[i]],z1
        .ray[,1][ray.echo.inds[i]])
    p4.s <- c(x1.shk[,1][shk.echo.inds[i]], y1.shk[,1][shk.echo.inds[i]],z1
        .shk[,1][shk.echo.inds[i]])

    ## now make two vectors from the three points
    v1.r <- c( (p1.r[1]-p2[1]),(p1.r[2]-p2[2]) )
    v2.r <- c( (p1.r[1]-p4.r[1]),(p1.r[2]-p4.r[2]) )

    v1.s <- c( (p1.s[1]-p2[1]),(p1.s[2]-p2[2]) )
    v2.s <- c( (p1.s[1]-p4.s[1]),(p1.s[2]-p4.s[2]) )

    ## calculate the angle between v1 and v2 in radians
    theta.r <- acos( (v1.r%*%v2.r)/(sqrt(sum(v1.r^2))*sqrt(sum(v2.r^2))) )
    theta.s <- acos( (v1.s%*%v2.s)/(sqrt(sum(v1.s^2))*sqrt(sum(v2.s^2))) )

    ## save the angles
```

```
    s.e0.e1.r[i] <- theta.r
    s.e0.e1.s[i] <- theta.s

###################
### XZ ANALYSIS ###
###################

    ## define the vectors
    v2.r <- (p3.r-p1.r)
    v3.r <- (p4.r-p1.r)

    v2.s <- (p3.s-p1.s)
    v3.s <- (p4.s-p1.s)

    ## find the angle between v3 and v2
    v3.theta.v2.r <- acos( (v3.r%*%v2.r)/(sqrt(sum(v3.r^2))*sqrt(sum(v2.r
        ^2))) )
    v3.theta.v2.s <- acos( (v3.s%*%v2.s)/(sqrt(sum(v3.s^2))*sqrt(sum(v2.s
        ^2))) )

    ## find the projection of v3 onto v2
    ## this is the length of the projection
    v3.proj.v2.r <- sqrt(sum(v3.r^2)) * cos(v3.theta.v2.r)
    v3.proj.v2.s <- sqrt(sum(v3.s^2)) * cos(v3.theta.v2.s)

    ## populate the scaling length vector
    s.lengths.r[i] <- v3.proj.v2.r
    s.lengths.s[i] <- v3.proj.v2.s

    i=i+1
}


################
### analysis ###
################

## the s.e0.r.xy/2 that are close to s.e1.r.xy should indicate
## good possibility for echo locations
## given by a tolerance angle
angle.tol = 10 * pi/180 ## in radians
```

```
ray.good.inds <- which( abs((s.e0.r.xy/2) - s.e0.e1.r) < angle.tol)
shk.good.inds <- which( abs((s.e0.s.xy/2) - s.e0.e1.s) < angle.tol)

## again, there is an error but it is understood.
## we take away the good indices that fall within the degree tolerance
## but in the wrong direction!

##ray.good.inds = ray.good.inds[1:3]
shk.good.inds = shk.good.inds[-c(1,2,3,14:18)]

## now we have to shift/scale our x1,y1s according to our length vector
## echo locations (xy1), for ray (ray), decimated, and scaled (ds)
## first the scaling factor

## define two vector matrices
v1.mat.r <- matrix(c(x1.ray.d-x0.ray.d, y0.ray.d-y0.ray.d),ncol=2)
v2.mat.r <- matrix(c(x1.ray.d-x0.ray.d, y1.ray.d-y0.ray.d),ncol=2)

v1.mat.s <- matrix(c(x1.shk.d-x0.shk.d, y0.shk.d-y0.shk.d),ncol=2)
v2.mat.s <- matrix(c(x1.shk.d-x0.shk.d, y1.shk.d-y0.shk.d),ncol=2)

## find the angle between the two vectors
## use a loop for now but there has to be another way

## set up a vector to hold the angles, thetas
thetas.r <- rep(0,nrow(v1.mat.r))
thetas.s <- rep(0,nrow(v1.mat.s))

i=1
while(i<=nrow(v1.mat.r)) {

    thetas.r[i] <- acos( (v2.mat.r[i,] %*% v1.mat.r[i,])/(sqrt(sum(v2.mat.r
        [i,]^2)) * sqrt(sum(v1.mat.r[i,]^2))) )

    thetas.s[i] <- acos( (v2.mat.s[i,] %*% v1.mat.s[i,])/(sqrt(sum(v2.mat.s
        [i,]^2)) * sqrt(sum(v1.mat.s[i,]^2))) )

    i=i+1
}

## set up a scaling factor
```

```
s.factor <- 50

## determine which way the slope is going in both x and y directions
x.sign.r <- (x0.ray.d - x1.ray.d)/abs(x0.ray.d-x1.ray.d)
y.sign.r <- (y0.ray.d - y1.ray.d)/abs(y0.ray.d-y1.ray.d)

x.sign.s <- (x0.shk.d - x1.shk.d)/abs(x0.shk.d-x1.shk.d)
y.sign.s <- (y0.shk.d - y1.shk.d)/abs(y0.shk.d-y1.shk.d)

## use thetas to find dy and dx
ray.dy <- abs(s.lengths.r)*sin(thetas.r) * s.factor * y.sign.r
ray.dx <- abs(s.lengths.r)*cos(thetas.r) * s.factor * x.sign.r

shk.dy <- abs(s.lengths.s)*sin(thetas.s) * s.factor * y.sign.s
shk.dx <- abs(s.lengths.s)*cos(thetas.s) * s.factor * x.sign.s

## add (or rather subtract) dy,dx to x0,y0
x2.ray <- x0.ray.d-ray.dx
y2.ray <- y0.ray.d-ray.dy

x2.shk <- x0.shk.d-shk.dx
y2.shk <- y0.shk.d-shk.dy

## limit all the vectors one more time
## to those with a positive lenght

x0.ray.d2 <- x0.ray.d[which(s.lengths.r > 0)]
y0.ray.d2 <- y0.ray.d[which(s.lengths.r > 0)]
x1.ray.d2 <- x2.ray[which(s.lengths.r >0 )]
y1.ray.d2 <- y2.ray[which(s.lengths.r > 0)]

x0.shk.d2 <- x0.shk.d[which(s.lengths.s > 0)]
y0.shk.d2 <- y0.shk.d[which(s.lengths.s > 0)]
x1.shk.d2 <- x2.shk[which(s.lengths.s >0 )]
y1.shk.d2 <- y2.shk[which(s.lengths.s > 0)]


############################
### CALCULATE GOOD AREAS ###
############################
```

```
c=340 ## speed of sound (m/s)
ef=1.1 ## echo delay (Hz)
r.dist=328 ## transient distance from source to ray (m)
s.dist=325 ## transient distance from source to shk (m)

## this is the distance that we will be looking for
r.ed <- c*ef + r.dist ## echo distance for shk (m)
s.ed <- c*ef + s.dist ## echo distance for shk (m)

## we need to define a fudge range
fudge=25 ## (m)

r.radsm <- r.ed-fudge ## ray radius big
r.radbg <- r.ed+fudge ## ray radius small

s.radsm <- s.ed-fudge
s.radbg <- s.ed+fudge

## find the total area under investigation
r.area <- (pi*r.radbg^2) - (pi*r.radsm^2)
s.area <- (pi*s.radbg^2) - (pi*s.radsm^2)

## now find how much area each of the points occupies
r.arw.area <- r.area/length(ray.echo.inds)
s.arw.area <- s.area/length(shk.echo.inds)

## and how much of the total area do that good arrows represent?
## there is a HACK in here! Be sure and fix it later
r.gd.arrws <- r.arw.area*(length(ray.good.inds)-6)
s.gd.arrws <- s.arw.area*(length(shk.good.inds)-4)


################
### PLOTTING ###
################
graphics.off()
close.screen(all.screens=TRUE)

pdf('~/site/erebus_2006/production_figs/dem_analysis/echo_dem.pdf')

## first the DEM
```

```
image(x=seq(ul.x,lr.x,length.out=dim(dem)[1]+1), y=seq(lr.y,ul.y,length.
    out=dim(dem)[2]+1),z=dem,zlim=c(3500,max(dem,na.rm=TRUE)),xlim=c
    (551700,552400),ylim=c(1393090,1393800), col=gray.colors(200),xlab="
    Easting (m)",ylab="Northing (m)",xaxt='n',yaxt='n',main='Mount Erebus
    Details')

## throw some axis lables on there
axis(1,at=551700, labels=551700)
axis(2,at=1393200, labels=1393200)
axis(1,at=c(551700 ,551900 ,552100 ,552300 ) ,labels=c("","+200" ,"+400"
    ,"+600"))
axis(2,at=c( 1393200 ,1393400 ,1393600 ,1393800), labels=c("","+200"
    ,"+400" ,"+600"))

## and contour DEM
contour(x=seq(ul.x,lr.x,length.out=dim(dem)[1]), y=seq(lr.y,ul.y,length.
    out=dim(dem)[2]),z=dem,xlim=c(551700,552400),ylim=c(1393090,1393790),
    xlab="",ylab="",xaxt='n',yaxt='n',add=TRUE,levels=c
    (3700,3675,3650,3625,3600,3575,3550,3525,3500),col='gray23')

## add the ray and shk x0,y0 points
points(x0.ray[,1],y0.ray[,1],col=rgb(0,1,1,0.05))
points(x0.shk[,1],y0.shk[,1],col=rgb(1,0,1,0.05))

## add the good gradient arrows
arrows(x0=x0.ray.d[ray.good.inds],y0=y0.ray.d[ray.good.inds],x1=x2.ray[ray
    .good.inds],y1=y2.ray[ray.good.inds],code=2,length=0.1,angle=20,cex=2,
    col=rgb(0,1,1,0.6),lwd=3)
arrows(x0=x0.ray.d[ray.good.inds],y0=y0.ray.d[ray.good.inds],x1=x2.ray[ray
    .good.inds],y1=y2.ray[ray.good.inds],code=2,length=0.1,angle=20,cex=2,
    col='black',lwd=1)

arrows(x0=x0.shk.d[shk.good.inds],y0=y0.shk.d[shk.good.inds],x1=x2.shk[shk
    .good.inds],y1=y2.shk[shk.good.inds],code=2,length=0.1,angle=20,cex=2,
    col=rgb(1,0,1,0.6),lwd=3)
arrows(x0=x0.shk.d[shk.good.inds],y0=y0.shk.d[shk.good.inds],x1=x2.shk[shk
    .good.inds],y1=y2.shk[shk.good.inds],code=2,length=0.1,angle=20,cex=2,
    col='black',lwd=1)

## add the gradient arrows
```

```
##arrows(x0=x0.ray.d2,y0=y0.ray.d2,x1=x1.ray.d2,y1=y1.ray.d2,code=2,length
    =0.1,angle=20,cex=2,col='gray10',lwd=0.8)
##arrows(x0=x0.shk.d2,y0=y0.shk.d2,x1=x1.shk.d2,y1=y1.shk.d2,code=2,length
    =0.1,angle=20,cex=2,col='gray10',lwd=0.8)

## plot the stations
points(shk.st[1],shk.st[2],pch=17,col='red',cex=2.6)
points(shk.st[1],shk.st[2],pch=17,col='blue',cex=1.4)
points(ray.st[1],ray.st[2],pch=17,col='blue',cex=1.8)

## plot the lava lake location
points(erebus.loc[1],erebus.loc[2],col='green',pch='+',cex=2)

## add some descriptive text
text(552154.4,1393535,"Ray Lake")
text(551887.6,1393651, "SHK")
text(552275.3,1393260, "RAY")

dev.off()
```