MADAGASCAR DATASETS

Trevor Irons
# DATA — TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Author</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trevor Irons</td>
<td>Amoco model</td>
<td>1</td>
</tr>
<tr>
<td>Trevor Irons</td>
<td>Marmousi Model</td>
<td>9</td>
</tr>
<tr>
<td>Trevor Irons</td>
<td>Marmousi2 model</td>
<td>17</td>
</tr>
<tr>
<td>Trevor Irons</td>
<td>Pluto Model</td>
<td>27</td>
</tr>
<tr>
<td>Trevor Irons</td>
<td>Sigsbee2 Models</td>
<td>33</td>
</tr>
</tbody>
</table>
Amoco model

Trevor Irons

Data Type: 2D subset of a synthetic 3D acoustic model
Source: British Petroleum
Location: http://www.software.seg.org
Format: Native
Date of origin: Model was produced for an SEG convention presented in 1998

INTRODUCTION

The Amoco dataset found within the Madagascar repository was created in 1997 and presented formally at the talk *Strike shooting, dip shooting, widpatch shooting – Does prestack migration care? A model study* given by John Etgen and Carl Regone at the 1998 SEG convention. Its creators describe the model as the Carpathians thrusting over the North Sea. The model was specifically created to illustrate the limitation of Kirchhoff migration. The model presented here is a single 2D line from the 3D model presented at the talk. The information presented here was taken from the abstract to their paper which can be found at the SEG website.

The Madagascar Amoco repository contains all the files listed in table 1. The repository contains several velocity models of varying smoothness as well as a shot record.

| 1 | -rwxr-xr-x | 1 root  root | 352292 2005-04-20 07:34 velsmooth.hH |
| 2 | -rwxr-xr-x | 1 root  root | 352278 2005-04-20 07:34 velsmoother.hH |
| 3 | -rwxr-xr-x | 1 root  root | 2269415 2005-04-20 07:34 velmodel.hH |
| 4 | -rwxr-xr-x | 1 root  root | 151395933 2005-04-20 07:34 shots.hH |

Table 1: A list of files contained within the Madagascar amoco dataset repository

MODEL

This model is a 2D subset of a 3D model, however the model does not vary perpendicularly to this line. The velocity model is 22 km across and 4 km in depth.

The *velmodel.hH* file did not need to be updated appreciably in this example. However, the appropriate header settings are found in table 2. Datums were spread every 12.5 meters to produce a 22km by 4 km grid.
Table 2: Amoco unsmoothed velocity model header information

A python SConstruct script that fetches the data sets, appends the header slightly and plots the velocity model can be found in Table 3. An image of the velocity profile is found in Figure 1.

Table 3: Scons script that generates RSF formatted Amoco velocity model

Typing Command 1 within the amoco/model directory runs the script.

bash-3.1$ scons view

Figure 1: Amoco velocity model. amoco/model velmodel
SHOTS

A synthetic off-end survey was performed on the model. Shots were fired every 50 meters along the model while 256 receivers with 25 meter spacing were pushed to the right. Each receiver recorded 384 time samples per shot with gates every 9.9 milliseconds. The header should be formatted as is shown in Table 5:

| n1 = 384 | d1 = 0.0099 | o1 = 0 | label1 = Time | unit1 = s |
| n2 = 256 | d2 = 0.025  | o2 = 0 | label2 = Offset | unit2 = km |
| n3 = 385 | d3 = 0.05   | o3 = 0 | label3 = Shot   | unit3 = km |

Table 4: Amoco shot header information

The file `amoco/shots/SConstruct` is presented in Table 5. This file fetches the shot data, appends the header slightly and produces several images from the record.

```bash
1 from r s f . proj import *
2 # Fetch Files from repository
3 Fetch("shots.hh","Amoco")
4 # Convert Files to RSF
5 Flow('shots','shots.hh','dd form=native |
6     put label1=time label2=offset label3=Shot unit1=km unit2=km unit3=km |
7     scale rscale =0.003048 ' ' ')
8 # Plotting Section
9 Result('zeroOne','shots','''window $SOURCE min2=0 max2=0 size2=1 |
10     max3=10.6 |
11     grey pclip=96 color=l screenratio=1.5 gainpanel=a |
12     label2=Position label1=Time title= label3= unit2=km unit1=s |
13     labelsz=6' ' ')
14 Result('zeroTwo','shots','''window $SOURCE min2=0 max2=0 size2=1 |
15     min3=10.6 |
16     grey pclip=97 color=l screenratio=1.5 gainpanel=a |
17     label2=Position label1=Time title= label3= unit2=km unit1=s |
18     labelsz=6' ' ')
19 Result('shot40','shots','''window $SOURCE min3=14 max3=14 size3=1 |
20     grey pclip=99 color=l gainpanel=a wantframenum=y unit1=s label1=Time |
21     label2=Offset unit2=km label3=Shot unit3=km title= |
22     screenratio =1.35 labelsz=3' ' ')
23 End()
```

Table 5: Scons script that generates RSF formatted Amoco velocity model

Typing Command 2 within the `amoco/shots` directory runs the script.

```bash
bash-3.1$ scons view
```

Several plots are produced. Figures 2(a) and 2(b) show the zero offset data acquired on the Amoco model. The plots are split as the large velocity contrast between the left and right side of the model muddles the image when the gainpanel filter plots the data.

FD MODELING

Madagascar can perform finite difference modeling on the Amoco Velocity model. This is done using the function fdmod found within the program. The raw velocity model needs to be formatted in a similar fashion to the Model Section of this paper.
Figure 2: Amoco zero offset shot data, plot (a) goes from 0 to 10.6 km and (b) goes from 10.6 to 20 km. The plots were split as the large velocity contrast in the model makes it difficult to plot both sides on one scale.
Figure 3: Amoco shot number 280; the source is 14 km from origin.

[amoco/shots shot40]
For the purposes of this example a shot will be fired at 10 km along the horizontal coordinate and at a depth of 10 meters. Receivers are spread at a depth of 25 meters every 12.5 meters along the entire scope of the model. This 22 km long receiver cable is impractical but useful for these purposes. Data is recorded on every receiver at time increments of 1 ms 5000 times resulting in 5 seconds of data.

An SConstruct file located within amoco/fdmod/ properly formats the model and inputs necessary parameters to perform a shot on the Amoco model. This file is reproduced below in Table 9.

Typing Command 4 within the amoco/fdmod/ directory runs the FD modeling script.

```
bash-3.1$ scons view
```

This script first constructs the survey acquisition geometry as was previously mentioned. An image of the survey is created and presented in Figure 9.

Figure 4: FD model geometry as performed on the Amoco velocity model. The X represents the shot while the smaller * symbols represent receivers. The receivers extent along the right hand side although it is not clear in this image.

Firing the shot results the propagation of a wavefield which can be seen in the movie wfl.vpl that is generated. Typing Command 5 within the amoco/fdmod directory displays the wavefield movie.

```
bash-3.1$ scons wfl.vpl
```

Four frames from this movie are presented in Figure 5 illustrating the propagation of the wavefield in the model.

The resulting data is then presented in the file dat.vpl. This plot is reproduced here in Figure 11. This shot is interesting as it clearly illustrates the different moveout witnessed on the two sides of the model.
Table 6: Scons script that performs a finite difference synthetic shot on the Amoco model.
Figure 5: Images of the propagating wavefield in the Amoco model generated by a finite difference model.  

Figure 6: Data gathered by the receivers in the FD model survey.
INTRODUCTION

The Marmousi model was created in 1988 by the Institut Français du Pétrole (IFP) in 1988. The geometry of this model is based on a profile through the North Quenguela trough in the Cuanza basin. The geometry and velocity model were created to produce complex seismic data which require advanced processing techniques to obtain a correct earth image. The Marmousi dataset was used for the workshop on practical aspects of seismic data inversion at the 52nd EAEG meeting in 1990.

Since its inception in 1990 Marmousi has come be a sort of industry standard and almost classic dataset. The Madagascar repository contains the Marmousi files shown in Table 1.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>rw-r-xr-x</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>2</td>
<td>rw-r-xr-x</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>3</td>
<td>rw-r-xr-x</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>4</td>
<td>rw-r-xr-x</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>5</td>
<td>rw-r-r-r-r</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>6</td>
<td>rw-r-r-r-r</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
<tr>
<td>7</td>
<td>rw-r-r-r-r</td>
<td>1</td>
<td>root</td>
<td>root</td>
</tr>
</tbody>
</table>

Table 1: A list of all files contained in the Marmousi repository

MODEL

The Marmousi model contains 158 horizontally layered horizons. A series of normal faults and resulting tilted blocks complicates the model towards its center. The model sits under approximately 32 m of water and is 9.2 km in length and 3 km in depth.

The velocity model found in the Madagascar repository, marmvel.hh can easily be displayed. This grid contains 751 data points in the Z direction and 2301 data points in the x direction. Table 2 displays the proper header configuration.

n1=751 d1=.004 o1=0 label= Depth unit1=km
n2=2301 d2=.004 o2=0 label2=Position unit2=km

Table 2: Header information for Marmousi velocity models

The script found at marmousi/model/SConstruct was written to obtain the Marmousi model datasets, append the headers as necessary and display the data. This file is presented in Table 3 while the velocity model image is displayed in Figure 1.
from rsf.proj import *

# Fetch Files from repository
raw=['marmvel.hh', 'marmsmooth.HH']
for file in raw:
    Fetch(file,'marm')
    if file is 'marmvel.hh':
        fileOut='marmvel'
        t='Velocity\ Model'
        d=.004
        fileOu t='marmvel'
        label1=Depth label2=Position unit1=km unit2=km
        title=Velocity unit1=km unit2=km
    elif file is 'marmsmooth.HH':
        fileOut='marmsmooth'
        t='Smoothed\ Velocity\ Model'
        d=.024
        fileOut='marmsmooth'
        label1=Depth label2=Position unit1=km unit2=km
        title=Velocity unit1=km unit2=km

# Convert Files to RSF and update headers
Flow(fileOut, file, 'dd form=native | scalerscaler =.001 | put

label1=Depth label2=Position unit1=km unit2=km

d1=%f d2=%f' % (d, d))

# Plotting Section
Result(fileOut, 'window $SOURCE |

grey color=I gainpanel=a allpos=y scalebar=y

title=Velocity barlabel=Velocity barunit=km/s screenratio=.326

screenht=3 wheretitle=t labellablez=4 titlesz=6 ' ' ' % t)

End()

Table 3: SConstruct script generating the Marmousi velocity model images

Typing Command 1 within the Marmousi/model directory runs the script.

bash-3.1$ scons view

Figure 1: Velocity model marmousi/model marmvel

SHOT RECORDS

The file marmrefl.hh contains the shot data collected on the Marmousi model. The survey was an off end survey with receivers to the left of the source being pulled towards the right. Receiver as well as shot spacing is every 25 meters. Near offset is 425 meters from the source. 726 time gates were recorded with 4ms spacing for roughly 3 seconds of data collection. Madagascar correctly converts this file according to its header; however, the correct shot header values are reproduced in Table 5.
Table 4: Shot header information for Marmousi.

The file *marmousi/shots/SConstruct* gathers shot data, appends the header as necessary and produces several plots of the data. This file is reproduced here in Figure 4.

Table 5: *SConstruct* script generating the Marmousi shot images

To run the script type Command 1 within the *marmousi/shots* directory.

bash-3.1$ scons view

FINITE DIFFERENCE MODELING

Madagascar can perform finite difference modeling on the Amoco Velocity model. This is done using the function fdmod. The raw velocity model needs to be formatted in a similar fashion to the Model Section of this paper.

For the purposes of this example a shot will be fired at 5 km along the horizontal coordinate and at a depth of 10 meters. Receivers are spread at a depth of 25 meters every 12.5 meters along the entire scope of the model. This long receiver cable is impractical but useful for these purposes. Data is recorded on every receiver at time increments of 1 ms 5000 times resulting in 5 seconds of data. In practice it would be necessary to perform longer running models, but this number of time gates is sufficient for this introduction.

An *SConstruct* file located within *marmousi/fdmod/* properly formats the model and inputs necessary parameters to perform a shot on the Marmousi model. This file is reproduced below in Table 9.
Figure 2: Shot number 20 of Marmousi synthetic survey dataset. Shot position in km is shown in the lower left hand corner. marmousi/shots shot20
Figure 3: Near offset data for Marmousi model [marmousi/shots nearOffset]
from rsf.proj import *  
import fdmod  

# Fetch Files from repository  
raw=['marmvel.hh', 'marmsmooth.HH']  
for file in raw:  
    Fetch(file, "marm")  
if file is 'marmvel.hh':  
d = .004  
fileOut='marmvel'  
t='Velocity\ Model'  
else:  
fileOut='marmsmooth'  
t='Smoothed\ Velocity\ Model'  
# Convert Files to RSF and update headers  
Flow(fileOut, file, "dd form=native | 
scale rscale=0.01 | put 
label1=Depth label2=Position unit1=km unit2=km 
d1=0.00025 d2=0.0005 % (d, d)")  
# Plotting Section  
Result(fileOut, """window $SOURCE | 
grey color=l gainpanel=a allpos=y scalebar=y 
title=\% b label1=\(km/s\) screenratio=.326 
screenht=3 where=left t labelsz=6 titlksz=6 % t)  
par = {  
'nt':10000, 'dt':0.00025, 'ot':0, 'lt':t, 'ut':t", 'a',  
'nx':2301, 'ox':0, 'dx':.004, 'lx':"x", 'ux':"km",  
'nx':751, 'ox':0, 'dx':.004, 'lx':"z", 'ux':"km",  
'nc':400 # wavelet delay  
}  
# add F-D modeling parameters  
fdmod.param(par)  
# density  
Flow('vel', 'marmvel',  
'\math n1=%(nt)d d1=%(dt)d o1=%(ot)g k1=%(kt)d | 
\math output=1 | ' 
icht title=t label1=t label2=unit2=' )  
Plot('vel', fdmod.rplot('\', par))  
# source positions  
Flow('sx', 'sz', 'math output=0.1')  
Flow('rx', 'rz', 'math output=0.5')  
Flow('rr', '[rx, rz, sr]', 'cat axis=2 space=n 
\$[SOURCES[0]] $[SOURCES[1]] | transp 
\$[SOURCES[2]] s c r e e n r a t i o =.326' )  
Plot('sx', 'sz', 'ssplot('', par))  
# finite differences modeling  
fdmod.awedl('dat', 'wfl', 'wav', 'vel', 'den', 'sx', 'rr', 'free=y dense=y', par)  
Plot('wfl', fdmod.wgrey('pclip=99', par), view=1)  
Result('dat', \"window j2=5 \+ fdmod.dgrey('pclip=99 title=Data\ Record label2=Offset 
where=left titlksz=6 labelsz=4', par))  
times=[-1.5, -1.0, -1.5, -2.0]  
cnt=0  
for item in [\"20\", \"40\", \"60\", \"80\"]:  
Result('time'+item, 'wfl',  
'\math n1=%(nt)d min1=0 min2=0 | grey gainpanel=a 
pclip=99 wantframenum=1 title=Wavefield at \%s s labelsz=4 
labell1=unit1-km label2=x unit2=km 
titlksz=6 screenratio=.18 screenht=2 where=left 
\math output=1 | (item, times[cnt])')  
cnt=cnt+1  
End()  

Table 6: Scons script that performs a finite difference synthetic shot on the Marmousi model.
Typing Command 4 within the `marmousi/fdmod/` directory runs the FD modeling script.

```
bash-3.1$ scons view
```

This script first constructs the survey acquisition geometry as was previously mentioned. An image of the survey is created and presented in Figure 9.

**Figure 4:** FD model geometry as performed on the Marmousi velocity model. The X represents the shot while the * symbols represent receivers. 

Firing the shot results the propagation of a wavefield which can be seen in the movie `wfl.vpl` that is generated. Typing Command 5 within the `marmousi/fdmod` directory displays the wavefield movie.

```
bash-3.1$ scons wfl.vpl
```

Four frames from this movie are presented in Figure 5 illustrating the propagation of the wavefield in the model.

**Figure 5:** Images of the propagating wavefield in the Marmousi model generated by a finite difference model.
The resulting data is then presented in the file `dat.vpl`. This plot is reproduced here in Figure 11.

![Data Record](marmousi/fdmod.dat)
INTRODUCTION

The Marmousi2 dataset is an extension and elastic upgrade of the classic Marmousi model. It was created by Allied Geophysical Laboratories (AGL). The Marmousi2 model has enjoyed widespread use and has been particularly insightful in amplitude versus offset (AVO) analysis, impedance inversion, multiple attenuation, and multi-component imaging. AGL has publically released the data for research use around the world.

Table 1 contains all the Marmousi2 files contained within the Madagascar repository.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Attributes</th>
<th>Size</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>density_marmousi-ii.segy</td>
<td>rw-r-r-r-w</td>
<td>155653444</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>vp_marmousi-ii.segy</td>
<td>rw-r-r-r-w</td>
<td>155653444</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>vs_marmousi-ii.segy</td>
<td>rw-r-r-r-w</td>
<td>155653444</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_curl_v_1.segy</td>
<td>rw-r-r-r-w</td>
<td>4525264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_curl_v_2.segy</td>
<td>rw-r-r-r-w</td>
<td>4525264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_div_v_1.segy</td>
<td>rw-r-r-r-w</td>
<td>226264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_div_v_2.segy</td>
<td>rw-r-r-r-w</td>
<td>226264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_vs_1.segy</td>
<td>rw-r-r-r-w</td>
<td>4525264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>obc_vs_2.segy</td>
<td>rw-r-r-r-w</td>
<td>4525264000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>surface_p1.segy</td>
<td>rw-r-r-r-w</td>
<td>3393949000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>surface_p2.segy</td>
<td>rw-r-r-r-w</td>
<td>3393949000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>surface_p3.segy</td>
<td>rw-r-r-r-w</td>
<td>4459728000</td>
<td>2005-05-05</td>
</tr>
<tr>
<td>surface_p4.segy</td>
<td>rw-r-r-r-w</td>
<td>2229866000</td>
<td>2005-05-05</td>
</tr>
</tbody>
</table>

Table 1: A list of all files contained in the Marmousi2 repository

MODEL

The Marmousi2 model completely encapsulates the original Marmousi model which was based on the Northern Quenguela Trough in the Quanza Basin of Angola. Lithologies include sandstones, shales, limestones and marls.

In total the Marmousi2 model is 3.5 km in depth and 17 km across. The model contains 199 horizons which make the model stratigraphically more complex than its predecessor. Additionally the water layer was extended to 450 meters.

As Marmousi2 is an elastic model both shear and primary velocities must be defined across the entire model. Additionally a density model is included. The files vp_marmousi-ii.segy, vs_marmousi-ii.segy, and density_marmousi-ii.segy contain the velocity and density models for Marmousi2. These three files all share the same data
The file *marmousi2/model/SConstruct* is a SCons script that fetches the three model files (VP, VS, and Density), appends the header information as necessary and produces plots of the models. This file is reproduced in table 3 and the models themselves are shown in figures 1, 2, and 3.

```python
from rsf.proj import *
# Fetch Files from repository
modelFiles=['vp_marmousi-ii.segy', 'vs_marmousi-ii.segy', 'density_marmousi-ii.segy']
outputFiles=['vp', 'vs', 'density']
for file in modelFiles:
    Fetch(file, "marm2")
# Convert Files to RSF
counter=0
for file in modelFiles:
    if file is 'vp_marmousi-ii.segy' or file is 'vs_marmousi-ii.segy':
        Flow(outputFiles[counter], file, ' segread tape=SOURCE | put
d1=0.001249 d2=0.011249 o1=0 o2=0 label1=Depth label2=Distance
unit1=km unit2=km')
    if file is 'density_marmousi-ii.segy':
        Flow(outputFiles[counter], file, ' segread tape=SOURCE | put
d1=0.001249 d2=0.011249 o1=0 o2=0 label1=Depth label2=Distance
unit1=km unit2=km')
    counter = counter+1
# Plotting Section
title=['Compressional\ Velocity\ Model', 'Shear\ Velocity\ Model', 'Density\ Model']
counter=0
for file in outputFiles:
    if file is 'vp' or file is 'vs':
        Result(file, file+' small',
        ' grey color=I gainpanel=a allpos=y title=%s
scalebar=y screenratio=.205 screenht=2
labelsz=4 whereititle=t titlel=6 b arreverse=y
... % title [counter])
    if file is 'density':
        Result(file, file+' small',
        ' grey color=I gainpanel=a allpos=y title=%s
scalebar=y screenratio=.205 screenht=2
scalebar=y barlabel=Density barunit="g/cm\^3\ ",
labelsz=4 whereititle=t titlel=6 b arreverse=y
... % title [counter])
    counter = counter+1
End()
```

Table 3: SCons script generating images of the Marmousi2 model

**SHOTS**

Three sets of data were collected over this model. A near surface streamer survey, a vertical sounding profile (VSP), and an ocean bottom cable (OBC) survey. Several sets of shot records are included in the Marmousi2 repository; multicomponent OBC data found in *obc_vx_.segy* and *obc_vz_.segy*, reduced data from the OBC cable
Figure 1: Marmousi2 P-wave velocity model

Figure 2: Marmousi2 S-wave velocity model

Figure 3: Marmousi2 Density Model
found in `obc_div_v.#.segy` and `obc_curl_v.#.segy`, and streamer cable data found in `surface_p.#.segy`. Each of these files was split into components to make them more manageable. The `#` symbol above corresponds to either part number 1 or 2.

In all cases the source was an airgun located on a ship at depth of 10 m. The source began firing at 3 000 m along the horizontal x coordinate and continued firing every 25 m until 14 975 m.

**OBC Surveys**

The OBC cable was placed on the ocean floor at a depth of roughly 450 m. Multicomponent phones were spaced every 12.32 m along the entire length of the model. As the model is 2D only the x and z components of the wavefield were measured. Marmousi2 OBC survey data should have header information configured as shown in table 4.

| n1=2500 | o1=0 | d1=0.002 | label1=Depth Z | unit1=s |
| n2=1381 | o2=0 | d2=12.32 | label2=Position X | unit2=m |
| n3=480  | o3=3000 | d3=25 | label3=Shot-Coord | unit2=m |

Table 4: Header information for Marmousi2 ocean bottom cable surveys

**OBC Vz data**

The file `marmousi2/vz/SConstruct` contains a SCons script that fetches the Vz component data files from the OBC survey, concatenates the segments, appends the header making a three axis file, (time, offset, and shot) and produces several plots of the data. This file is reproduced in table 5.

**OBC Vx data**

Similar to the Vz data the file `marmousi2/vx/SConstruct` contains a list of rules that tell Madagascar to gather the Vx data files, append the header and produce plots of the data. This script is reproduced in table 6.

**OBC div data**

The divergence operator was applied to the multicomponent OBC Pluto dataset. These files are `obc_div_v.1.segy` and `obc_div_v.2.segy`. Taking the divergence separates out the acoustic component of the data.
from rsf.proj import *

# Fetch Files from repository
Fetch("obc_vs_1.segy","marm2")
Fetch("obc_vs_2.segy","marm2")

# Convert Files to RSF and update header
Flow("obc_vs_1",'obc_vs_1.segy',
'"segyread read=d|
  put n2=1381 n3=240 o1=0 o2=0 o3=3000
d2=12.32 d3=25")
Flow("obc_vs_2",'obc_vs_2.segy',
'"segyread read=d|
  put n2=1381 n3=240 o1=0 o2=0 o3=11000
d2=12.32 d3=25')

# Concatinate Datasets
Flow("vz",['obc_vs_1','obc_vs_2'],
'cat $SOURCES[0:2] axis=3 stdin=0')

# Plot Data
Result('zero', 'vz',
'"window $SOURCE min2=0 max2=0 size2=1 |
grey color=I gainpanel=a label2=Position X unit2=m
title=Zero\ Offset\ Data'')
Result('zero2', 'vz',
'"window $SOURCE min2=0 max2=0 size2=1 |
grey color=I gainpanel=a label2=Position X unit2=m
title=Zero\ Offset\ Data'')

End()
The file `marmousi2/div/SConstruct` contains a list of rules that tell Madagascar to gather the div data files, append the header and produce plots of the data. This script is reproduced in table 7 and a plot of shot 50 is shown in figure ??

```python
from rse import *

# Fetch Files from repository
Fetch("obc_div_v1.segy", "obc_div_v2.segy")

# Convert Files to RSF and update header
Flow('obc_div_v1', 'obc_div_v1.segy', 'segyread tape=$SOURCE |
  put n2=1232 d2=12.32 d3=25 label1=Depth \ Z label2=Distance \ X label3=Shot-Cord
  unit1=m unit2=m unit3=m')

Flow('obc_div_v2', 'obc_div_v2.segy', 'segyread tape=$SOURCE |
  put n2=1232 d2=12.32 d3=25 label1=Depth \ Z label2=Distance \ X label3=Shot-Cord
  unit1=m unit2=m unit3=m')

# Concatinate Datasets
Flow('div', ['obc_div_v1', 'obc_div_v2'], 'cat {SOURCES[0:2]} axis=3', stdin=0)

# Plot Data
Result('movie', 'div', 'window $SOURCE |
  min3=4250 max3=4250 n3=1 |
  grey color=1 gainpanel=a
  title=OBC \ Div \ Shot \ 50')
```

Table 7: SCons script generating images of the Marmousi2 Vx data

**OBC curl data**

Similarly the curl operator was applied to the Pluto OBC data. These files are `obc_curl_v1.segy` and `obc_curl_v2.segy` These curl data contain only data generated by the elastic component of the field.

The file `marmousi2/curl/SConstruct` contains a list of rules that tell Madagascar to gather the curl data files, append the header and produce plots of the data. This script is reproduced in table 8

**Streamer Surveys**

The streamer survey was not traditional in the sense that it employed a 17 km long static streamer which spanned the entire model. In total there were 1 361 single component hydrophones spaced every 12.5 m at a depth of 5 m. This unrealistic geometry was chosen both for simplicity and to allow maximum utility of the data. The table 9 outlines the values that streamer data files headers should have.

**FINITE DIFFERENCE MODELING**

Madagascar may be used to perform finite difference modeling of the wavefield and receiver data. The tools to perform these tasks are found in the fdmod package.
Figure 4: Marmousi2 shot 50 of div data. [marmousi2/div movie]
from rsf.proj import *
import fdmod

# Fetch Files from repository
Fetch("oob_curl_v1.segy","marm2")
Fetch("oob_curl_v2.segy","marm2")

# Convert Files to RSF and update header
Flow("oob_curl_v1","oob_curl_v1.segy","segread tape=$SOURCE 
| put n2=1381 n3=320 o1=0 o2=0 o3=3000 
d2=12.32 d3=25 label1=Z label2=X label3=Shot 
unit1=s unit2=m unit3=",stdin=0)
Flow("oob_curl_v2","oob_curl_v2.segy","segread tape=$SOURCE 
| put n2=1381 n3=160 o1=0 o2=0 o3=3000 
d2=12.32 d3=25 label1=Z label2=X label3=Shot 
unit1=s unit2=m unit3=",stdin=0)

# Use fdmod for a graphing function
par = 
\{ nt':2500, dt':0.002, ot':0, 'lt':"t", 'ut':"s", 
'n':1381, 'ox':0, 'dx':12.32, 'lx':"x", 'ux':"km", 
'ny':480, 'oy':0, 'dy':12.32, 'ly':"y", 'uy':"km", 
\}

# add F-D modeling parameters
fdmod.param(par)

# Concatinate Datasets
Flow("curl","[oob_curl_v1","oob_curl_v2"]","cat ${SOURCES[0:2]} axis=3",stdin=0)

# Plot Data
Result("curlShot50","curl","window $SOURCE 
min3=4250 max3=4250 size3=1 | 
grey color=I gainpanel=a 
title=OBC\ Curl\ Shot\ 50")

Result("movie","curl","window $SOURCE 
j3=20 | 
grey color=I gainpanel=a 
title=OBC\ Curl\ Shot\ 50")

Result("curlFD","curl",fdmod.cgrey("j3=20 bias=1.5 pclip=98",par))

End()

Table 8: SCons script generating images of the Marmousi2 curl data

| n1=2500 | o1=0 | d1=0.002 | label1=Depth Z | unit1=s |
| n2=1381 | o2=0 | d2=12.5  | label2=Position X | unit2=m |
| n3=480  | o3=3000 | d3=25 | label3=Shot-Coord | unit2=m |

Table 9: Header information for Marmousi2 streamer surveys
Note, these processes are somewhat computationally intensive. I performed the majority of these models on a machine with a 3 GHz processor and 1.5 MB of RAM and most of the models took on the order of 3 hours to perform.
Pluto Model

Trevor Irons

Data Type: Synthetic
Source: SMAART Consortium
Location: http://www.delphi.tudelft.nl/SMAART/pluto15.htm
Format: SEGY and Native
Date of origin: Publicly released November 2000

INTRODUCTION

The Pluto dataset is one of several that The Subsalt Multiples Attenuation and Reduction Technology Joint Venture (SMAART JV) publicly released between September 2001 and November 2002. Additional information may be found at: http://www.delphi.tudelft.nl/SMAART/. The data remain the property of SMAART and are used under the agreement found at the aforementioned web address.

The Pluto 1.5 dataset is a 2D elastic dataset released in November 2000, designed to emulate deep water subsalt prospects as found in the Gulf of Mexico. It contains realistic free surface and internal multiples over a structure that is relatively easy to image. Table 1 shows the files contained within the Pluto repository of Madagascar.

Table 1: A list of all files contained in the Pluto repository

<table>
<thead>
<tr>
<th>Number</th>
<th>Permissions</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Date and Time</th>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>1094022342</td>
<td>2005-04-20 07:46</td>
<td>pluto.shot.hh</td>
</tr>
<tr>
<td>2</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>35109840</td>
<td>2005-04-20 07:46</td>
<td>int_depth_vp.sgy</td>
</tr>
<tr>
<td>3</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>576162400</td>
<td>2005-04-20 07:48</td>
<td>P15shots150fendon0ph1stHlf.SEGY</td>
</tr>
<tr>
<td>4</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>576162400</td>
<td>2005-04-20 07:48</td>
<td>P15shots150fendon0ph2ndHlf.SEGY</td>
</tr>
<tr>
<td>5</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>3067</td>
<td>2005-04-20 07:48</td>
<td>readme.txt</td>
</tr>
<tr>
<td>6</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>473944</td>
<td>2005-04-20 07:48</td>
<td>Pluto1.5_series.gif</td>
</tr>
<tr>
<td>7</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>35328</td>
<td>2005-04-20 07:48</td>
<td>Pluto15HeaderCoordinates.xls</td>
</tr>
<tr>
<td>8</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>26112</td>
<td>2005-04-20 07:48</td>
<td>Pluto1.5_acq_parameters.doc</td>
</tr>
<tr>
<td>9</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>22016</td>
<td>2005-04-20 07:48</td>
<td>Pluto1.5Const.doc</td>
</tr>
<tr>
<td>10</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>40677840</td>
<td>2005-04-20 07:48</td>
<td>P15VPint25f_padded.SEGY</td>
</tr>
<tr>
<td>11</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>39004355</td>
<td>2005-04-20 07:48</td>
<td>pluto.velo.hh</td>
</tr>
<tr>
<td>12</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>947523094</td>
<td>2005-12-13 18:55</td>
<td>data.H</td>
</tr>
<tr>
<td>13</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>564007299</td>
<td>2005-12-13 18:56</td>
<td>data.art.H</td>
</tr>
<tr>
<td>14</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>947536358</td>
<td>2005-12-13 18:57</td>
<td>mult.H</td>
</tr>
<tr>
<td>15</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>116</td>
<td>2005-12-13 19:02</td>
<td>readme-antoine.txt</td>
</tr>
<tr>
<td>16</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>1094021600</td>
<td>2005-12-13 18:59</td>
<td>pluto-shot.H</td>
</tr>
<tr>
<td>17</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>2086</td>
<td>2005-12-13 18:59</td>
<td>pluto_cmps.H</td>
</tr>
<tr>
<td>18</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>146</td>
<td>2005-12-13 19:02</td>
<td>readme-antoine.txt</td>
</tr>
<tr>
<td>19</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>1340390400</td>
<td>2005-12-13 19:02</td>
<td>pluto_cmps.H</td>
</tr>
<tr>
<td>20</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>947523393</td>
<td>2005-12-13 19:04</td>
<td>sign.H</td>
</tr>
<tr>
<td>21</td>
<td>rw-r-x-x-x</td>
<td>root</td>
<td>root</td>
<td>947523599</td>
<td>2005-12-13 19:07</td>
<td>sign_imp.H</td>
</tr>
</tbody>
</table>

VELOCITY MODELS

The Pluto model was designed to offer a complex environment to test multiple attenuation algorithms. The model is 32 km (105,000 ft) long and 9.14 km (30,000 ft) in
The velocity model file \textit{int\_depth\_vp.sgy} has 1201 datapoints in the vertical direction and 6960 datums in the horizontal direction. The actual synthetic surveys were conducted on a padded model which contains constant velocity cells outside of the model boundaries.

To assure the proper geometry Pluto velocity model headers should be formatted as shown in table 2. Values are listed for both metric and standard units. This article will display metric units exclusively.

<table>
<thead>
<tr>
<th></th>
<th>Standard</th>
<th>Metric</th>
<th>Padded</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n_1)</td>
<td>1201</td>
<td>1201</td>
<td>1401</td>
</tr>
<tr>
<td>(n_2)</td>
<td>6960</td>
<td>6960</td>
<td>6960</td>
</tr>
<tr>
<td>(d_1)</td>
<td>0.025</td>
<td>0.0076</td>
<td>0.025 or 0.0076</td>
</tr>
<tr>
<td>(d_2)</td>
<td>0.025</td>
<td>0.0076</td>
<td>0.025 or 0.0076</td>
</tr>
<tr>
<td>(o_1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(o_2)</td>
<td>-34.875</td>
<td>-10.629</td>
<td>-34.875 or -10.629</td>
</tr>
</tbody>
</table>

Table 2: Header information for Pluto velocity models

The SConstruct file found within \texttt{rsf/book/data/pluto} is shown in table 3. This SConstruct file produces both metric and standard plots of the velocity model. However, only the metric one is presented here in figure 1. Additionally, the padded model found in file \texttt{P15VPint\_25f\_padded.SEGY}, is displayed in figure 2 for reference.

Typing command 1 within the \texttt{pluto} directory runs the script.

bash-3.1$ scons view

![Figure 1: Pluto P-wave velocity model in metric units](pluto/model velocityProfileMetric)
from rsf.proj import *
# Fetch Files from repository
Fetch("int_depth_vp.sgy","pluto")
Fetch("P15VPint_25f_padded.SEGY","pluto")

# Convert Files to RSF
Flow('velocityProfileStd','int_depth_vp.sgy',"
    segyread read=d |
    put d2=0.25 label1=Depth o2=-34.875
    label2=Position unit1=kft unit2=kft
    label=Velocity unit=kft/s |
    scale rscale=0.001"
)

Flow('velocityProfileMetric','int_depth_vp.sgy',"
    segyread read=d |
    put d1=0.0076 d2=0.0076 o2=-10.629
    label1=Depth label2=Position label=Velocity
    unit1=km unit2=km unit=km/s |
    scale rscale=0.0003048"
)

Flow('velocityProfilePadded','P15VPint_25f_padded.SEGY',"
    segyread read=d |
    put d1=0.0076 d2=0.0076 o2=-10.629 label1=Depth
    label2=Position unit1=km unit2=km label=Velocity |
    scale rscale=0.0003048"
)

# Plotting Section
mins=[0,0,-10.5]
maxs=["105",'32','42.5']

for item in ['Std','Metric']:
    Result('velocityProfile'+item,"
    window j1=2 j2=2 |
    grey scalebar=y color=j allpos=y bias=1 title=P-Wave\ Velocity\ Profile
    max2=%s min2=0 screenratio=.28125 screenht=2
    labelsz=4 wanttitle=n barreverse=y % maxs[counter])
    counter=counter+1

Result('velocityProfilePadded',"
    window j1=2 j2=2 |
    grey scalebar=y color=j allpos=y bias=1 gainpanel=a title=P-Wave\ Velocity\ Profile
    screenratio=.28 125 screenht=2 labelsz=4 wanttitle=n barreverse=y"
)
End()

Table 3: SConstruct script generating the velocity model images

![Figure 2: Padded velocity model that surveys were conducted on](pluto/model velocityProfilePadded)
SHOT RECORDS

BP performed a fourth order finite differencing modeling code on the padded velocity model. Madagascar can easily be used to display and manipulate the data. The script `pluto/shot/SConstruct` presented in table 4 fetches the dataset and constructs the RSF formatted dataset `plutoShots.rsf`.

As written this script outputs two images; figure 4 shows the Pluto zero offset shot gather while figure 4 shows shot 30.

```python
from rsf.proj import *

# Fetch Files from repository
fetch( "P15shots150f_endon_0ph-1stHlf.SEGY", "pluto" )
fetch( "P15shots150f_endon_0ph-2ndHlf.SEGY", "pluto" )

# Convert Files to RSF and append headers
files = ["P15shots150f_endon_0ph-1stHlf.SEGY", "p15shots150f_endon_0ph-2ndHlf.SEGY"]
counter=0 #o2=360
for item in ["rsf1", "rsf2"]: Flow( item, files[counter], ""
segread tape=$SOURCE | put
  o1=0 o2=0 o3=0 d2=0.02286 d3=0.0457 n2=350 n3=347
  label1=Time label2=Position unit1=s unit2=km
  label3=Shot "", stdin=0)
counter = counter + 1

# Concatinate Files
Flow( "plutoShots", ["rsf1", "rsf2"], ""
cat ${SOURCES[0:2]} axis=3 "", stdin=0)

# Plotting Section
result( "zero", "plutoShots", ""
  window $SOURCE min2=0 max2=0 size2=1 |
  grey color=I gainpanel=a label2=Position X unit2=km
  title=Zero\ Offset\ Data label2=Distance "")
result( "shot30", "plutoShots", ""
  window $SOURCE min3=1.371 max3=1.371 size3=1 | grey color=I wantframenum=y
gainpanel=a title=Shot\ #\ 30 label2=Offset "")
end()
```

Table 4: `Scons` script that generates RSF formatted pluto shot data

Shot data should be formatted as shown in table 5. Again both metric and standard units are shown.

<table>
<thead>
<tr>
<th>Standard</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1=1126 d1=.008</td>
<td>n1=1126 d1=0.008</td>
</tr>
<tr>
<td>o1=0 label1=Z</td>
<td>o1=0 label1=“Depth”</td>
</tr>
<tr>
<td>Depth unit1=s</td>
<td>unit1=s</td>
</tr>
<tr>
<td>n2=350 d2=75</td>
<td>n2=350 d2=0.02286</td>
</tr>
<tr>
<td>o2=0 label2=X</td>
<td>o2=0 label2=“Position”</td>
</tr>
<tr>
<td>unit2=ft</td>
<td>unit2=km</td>
</tr>
<tr>
<td>n3=694 d3=150</td>
<td>n3=694 d3=0.0457</td>
</tr>
<tr>
<td>o3=0 label3=Shot</td>
<td>o3=0 label3=“Shot”</td>
</tr>
</tbody>
</table>

Table 5: Header information for Pluto velocity models
Figure 3: Zero offset data for Pluto synthetic dataset [pluto/shot zero]
Figure 4: Shot 30 of Pluto dataset
Sigsbee2 Models

Trevor Irons

Data Type: 2D model and acoustic finite difference synthetic data set with constant density
Source: SMAART consortium comprised of BHPBilliton Petroleum, BP, and the ChevronTexaco Exploration and Production Technology Company
Location: http://www.delphi.tudelft.nl/SMAART/sigsbee2a.htm
Format: SEGY and Native
Date of origin: Data were publicly released between September 2001 and November 2002.

INTRODUCTION

The Subsalt Multiples Attenuation and Reduction Technology Joint Venture (SMAART JV) publicly released several data sets between September 2001 and November 2002. These synthetic data model the geologic setting found on the Sigsbee escarpment in the deep water Gulf of Mexico. Additional information may be found at: www.delphi.tudelft.nl/SMAART/
The data sets remain the property of SMAART and are used under the agreement found at the SMAART site listed above.

The file sigsbee/FILES lists all files contained in the Sigsbee2 repository of Madagascar and is reproduced below in Table 1. Any of these files may be downloaded to local machines using ftp protocols.

The Sigsbee2 data are separated into two distinct categories, A and B. They share the same general model geometry and structure, however, the A model has a soft water to seafloor boundary while the B model features a more realistic hard boundary. As a result data produced in the B model features multiple events. The Sigsbee2B data set was featured in paper SP3.8 ”Observations from the Sigsbee2B synthetic data set” at the 2002 SEG meeting in Salt Lake City.

SIGSBEE MODELS

This model contains a sedimentary sequence broken up by a number of normal and thrust faults. Additionally, there is a complex salt structure found within the model that results in illumination problems when processing and migrating the data.

The Sigsbee2A model features an absorbing free surface condition and a weaker than normal water bottom reflection, resulting in data do not contain free surface
multiples and less than normal internal multiples. The Sigsbee 2B model uses the same structural model as Sigsbee2A but the velocity contrast at the water bottom has been increased to a normal level thus generating significant internal and free surface multiples. Modeling on the 2B model was performed with both free and non free surface boundary conditions.

The Sigsbee2 models found in the Madagascar repository share the same dimensions and sampling rate. The model is 9.144 km (30 000 ft) in depth and 24.384 km (80 000 ft) in length. All the models contain 7.62 m (25 ft) grid spacing except for the migrated models that have 11.43 m (37.5 ft) lateral grid spacing. Throughout this article both standard and metric units will be presented in tabular form but all figures will exclusively utilize metric units.

Table 2 displays the correct values that Sigsbee2 model headers should contain.

### Sigsbee 2A Models

The Sigsbee2A velocity and reflection coefficient models are easily viewed using Madagascar. There are 2 velocity models, a smooth migrated model and a true stratigraphic model. The SCons script `sigsbee/model2A/SConstruct` contains a set of rules that tell Madagascar to fetch the data append the headers and make several plots. This script is reproduced in Table 3.

Typing Command 1 within the `sigsbee/model2A` directory runs the script.

```bash
bash-3.1$ scons view
```

![Image of a table showing file list](image-url)

**Table 1:** List of all available files in the Madagascar Sigsbee2 repository
Table 2: Header information for Sigsbee2 models, note the initial offset in the horizontal direction and the coarse lateral sampling of the migrated models.

The Sigsbee2A migrated and stratigraphic velocity models are shown in Figures 1(a) and 1(b) respectively. A plot of the reflection coefficients are shown in Figure 1(c).

**Sigsbee 2B Models**

The Sigsbee 2B model contains the same general geometry as the 2A model except for a more realistic water to floor boundary which results in multiple generation when shots are modeled on it. However, dealing with the files is basically identical the headers should also be calibrated as shown in Table 2.

Table 4 shows the contents of the `sigsbee/model2b/SConstruct` script. This file is quite similar to the one found in the Sigsbee 2A section and contains a list of rules that fetch the datasets and plot them.

Typing Command 2 within the `sigsbee/model2B` directory runs the script.

bash-3.1$ scons view  

A plot of the migrated velocity model is shown below Figure 2(b) while the stratigraphic model can be seen in Figure 2(a). A plot of the reflection coefficients are shown in Figure 2(c).
from rsf.proj import *

# Sigbee 2A velocity model construct #

PRFX = 'sigbee2a_sgy'
SUFX = '.sgy'

for c in ('migration_velocity', 'stratigraphy', 'reflection_coefficients'):
    if(c=='migration_velocity'):
        v='vmig2A'
        o=3.055
        d=0.01143
        s=0.003048
        l='Migration Velocity'
        a='y'
        u='km/s'
    if(c=='stratigraphy'):
        v='vstr2A'
        o=3.048
        d=0.00762
        s=0.003048
        l='Stratigraphic Velocity'
        a='y'
        u='km/s'
    if(c=='reflection_coefficients'):
        v='reflectionCoefficients'
        o=3.048
        d=0.00762
        s=1
        l='Reflection Coefficients'
        a='n'
        u=''
    h = c + ' head'
    t = PRFX + c + SUFX
    Fetch(t, 'sigbee')

Flow([v,h], t,
    segyread tfile=${TARGETS[1]} |
    put o1=0 d1=.00762 label1=Depth unit1=km
    o2=%f d2=%f label2=Distance unit2=km
    | scale rscale=%f
    Result(v,v, 'window f1=1 | grey color=1 scalebar=y allpos=y screenratio=.375 screenht=3 barreverse=y
    wheretitle=t title=%s labels=4 titlex=6 barlabel=Velocity barunit=%s ' % (a,l,u))

End()
Figure 1: Sigsbee 2A contains a stratigraphic velocity model (a) a migrated smoothed model (b) and a reflection coefficient model (c).

```
sigsbee/model2A vmig2A,vstr2A,reflectionCoefficients
```

Figure 2: Sigsbee 2B contains two velocity models, a stratigraphic model (a) and a migrated model (b). The resulting reflection coefficients are shown in (c).

```
sigsbee/model2B vstr2B,vmig2B,reflectionCoefficients2B
```
Table 4: Contents of model2B/SConstruct script.
SHOT RECORDS

Several sets of data were acquired on the Sigsbee models. The Madagascar repository contains one survey taken on the Sigsbee2A model which was performed with an absorbing surface boundary condition. Two surveys were conducted on the Sigsbee2B models one with a free surface boundary and one without.

The three surveys shared the same acquisition geometry. Each receiver recorded data every .008 seconds for 1 500 timesteps resulting in 12 seconds of data. A 7 950 m (26 100 ft) long streamer cable was deployed with 348 hydrophones spaced 22.86 m (75 ft) apart. Shots were fired every 45.72 m (150 ft) starting at 3 330 m (10 925 ft). Table 5 shows the values that Sigsbee shot headers should contain.

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Standard} & & & & \text{Metric} & \\
\text{n1=1500} & \text{d1=0.008} & \text{o1=0} & \text{label1=Time} & \text{unit1=s} & \\
\text{n2=348} & \text{d2=75} & \text{o2=0} & \text{label2=Offset} & \text{unit2=ft} & \\
\text{n3=500 or 496} & \text{d3=150} & \text{o3=10925} & \text{label3=Shot-Coord} & \text{unit3=ft} & \\
\hline
\end{array}
\]

Table 5: Appropriate header values for Sigsbee shot records. The number of shots, \( n3 \), varies slightly between the surveys.

Sigsbee 2A shot records

The survey performed on the Sigsbee2A model had an infinite surface boundary condition. The script found at data2A/SConstruct whose contents are displayed in Table 6 generates a Madagascar formatted data file shots.rsf and also produces several shot images.

Typing Command 3 within the sigsbee/data2A directory runs the script.

\[\text{bash-3.1$ scons view (3)}\]

A plot of the 70th shot, made 6.5 km in to the model is produced by the SConstruct script and is shown below in Figure 3 The zero offset data is presented in Figure 4.

Sigsbee 2B Shot Records

The Sigsbee 2B library contains two sets of shot data, nfs and fs. These shots were modeled with free and non free surface boundary conditions.
from rsf.proj import *

define Variables and Filenames:

data = 'sigsbee2a.nfs.sgy'

Import Data:

Fetch('data', 'sigsbee2a')

Convert Data:

Flow('zzdata tzdata ./dhead ./bdhead', data, ...
  segyread
  tfile=${TARGETS[1]}
  bfile=${TARGETS[3]}
  ' ' ')

create sraw(t,o,s): o=full offset, s=shot position, t=time

Flow('as', 'zzdata', 'dd type=float | headermath output="10925+fldr*150" | window')
Flow('oo', 'zzdata', 'dd type=float | headermath output="offset" | window')
Flow('si', 'as', 'math output=input/150')
Flow('oi', 'oo', 'math output=input/75')
Flow('os', 'oi si', 'cat axis=2 space=n ${TARGETS[1]} | transp | dd type=int')
Flow('raw', 'zzdata os', ...
  inbin head=${TARGETS[1]} xkey=0 ykey=1 ' ' ')
Flow('shot', 'raw', ...
  put
  d2=0.02286 o3=0 label2=Offset unit2=km
  d3=0.04572 o3=3.330 label3=Shot coord unit3=km |
  mutter half=false t0=1.0 v0=6000 ' ' ')

Plot Data:

Result('zero', 'shot', ...
  window min2=0 max2=0 size2=1 |
  grey pcmap=98 color=I screenratio=1.5 gainpanel=a
  label2=Position label1=Time title= label3= unit2=km unit1=s
  labels3=3)
Figure 3: Snapshot of shot number 70 performed on *sigsbee 2A* the position of the source in km is in the lower left hand corner of the plot. [sigsbee/data2A shot70]
Figure 4: Sigsbee2A zero offset data
Free surface model

A SConstruct script found at sigsbee/data2B/fs/ is presented in Table 7. This script reads the segy source file and converts it to Madagascar’s RSF format and appends the header as necessary. The free surface boundary present within this model allows for the generation of reflections at the model edges.

```python
from rsf.proj import *

# Define Variables and Filenames
data = 'sigsbee2b_fs.segy'

# Import Data
Flow('tdata tdata ../dhead ../bhead', data, 'segread',
    tape=SOURCE,
    tfile=TARGETS[1],
    hfile=TARGETS[2],
    bfile=TARGETS[3],
    '\.\.\.',
    stdin=0)

# Convert Data
Flow('zdata tzdata ../dhead ../bhead', data,
    'segread',
    tape=SOURCE,
    tfile=TARGETS[1],
    hfile=TARGETS[2],
    bfile=TARGETS[3],
    '\.\.\.',
    stdin=0)

Flow('ss', 'tzdata', 'dd type=float | headermath output="10925+fldr*150" | window')
Flow('si', 'ss', 'math output=input/150')
Flow('oi', 'oo', 'math output=input/75')
Flow('os', 'oi', 'cat axis=2 space=n $\{SOURCES[1]\} | transp | dd type=int')
Flow('sraw', 'tzdata oo',
    'intbin head=\{SOURCES[1]\} xkey=0 ykey=1 ')
Flow('shotFs2B', 'sraw', '
    put
    d2=02286 o3=0 label2=Offset unit2=km
    d3=04572 o3=3330 label3=Shotcoord unit3=km |
    matter half=false t0=1.0 v0=6000 ')

# Plot Data
Result('zero2Bfs', 'shotFs2B', 'window $\{SOURCE\} min2=0 max2=0 size2=1 |
    grey pclip=98 color=1 screenratio=1.5 gainpanel=a |
    label2=Position label1=Time unit1=s label2=km unit2=km ')
Result('shot702Bfs', 'shotFs2B', 'window $\{SOURCE\} n3=1 f3=70 |
    grey pclip=99 color=1 gainpanel=a wantframenum=y unit1=s label1=Time |
    label2=Offset unit2=km label3=Shotcoord unit3=km |
    screenratio=1.35 label3=3')
```

Table 7: Contents of data2B/fs/SConstruct script.

Again shot number 70, fired 6.5 km into the model, is plotted in Figure 5. The precise coordinates of the shot are shown in the lower left hand corner of the figure. The zero offset data is presented in Figure 6.
Figure 5: Shot number 70 performed on sigsbee 2B FS model. The shot location is presented in the lower left hand corner of the plot sigsbee/fs2B_shot70Bfs
Figure 6: Sigsbee2B free surface boundary zero offset data.
Infinite Surface Model

This data was prepared with the boundaries of the model extending forever; as such multiples are not created as a result of the model edges.

A \texttt{SConstruct} script found at \texttt{sigsee/data2B/fs/} is presented in Table 8. This script translates the \texttt{segy} source data file and converts it into \texttt{rsf} format.

```
from \texttt{rsf.proj} import *

data = 'sigsee2b_nfs.segy'

tape=SOURCE

Flow ("ss", 'tzdata', 'dd type=float | headermath output="10925-fldr*150" | window')
Flow ("oo", 'txdata', 'dd type=float | headermath output="offset" | window')
Flow ("oi si", 'cat axis=2 space=n $(SOURCES[1]) | transp | dd type=int')
Flow ("sraw", 'zdata os',
```

# create sraw(t,o,s): o=full offset, s=shot position, t=time

Flow ("ss", 'tzdata', 'dd type=float | headermath output="10925-fldr*150" | window')
Flow ("oo", 'txdata', 'dd type=float | headermath output="offset" | window')
Flow ("oi si", 'cat axis=2 space=n $(SOURCES[1]) | transp | dd type=int')
Flow ("sraw", 'zdata os',
```

# S Construct Script

Table 8: Contents of \texttt{data2B/nfs/SConstruct} script.

Similar plots are produced for this model. Figure 7 shows an image of shot number 70 taken at 6.5 km into the model. Figure 8 displays the zero offset data acquired on this model.
Figure 7: Shot 70 performed in Sigsbee 2B NFS model.
Figure 8: Sigsbee 2B reflecting surface zero offset data, notice the decreased multiples from the free surface model [sigsbee/nfs2B zero2Bnfs]
FINITE DIFFERENCE MODELING

Finite difference (FD) shot and data modeling can be performed on the Sigsbee models using Madagascar. This example will use the Sigsbee2A model but it could be easily extended to perform modeling on Sigsbee2B.

For the purposes of this example a shot will be fired at 10 km along the horizontal coordinate and at a depth of 10 meters. Receivers are spread at a depth of 0 meters every 7.62 m (25 ft) along the entire scope of the model. This long receiver cable is impractical but useful for these purposes. Data is recorded on every receiver at time increments of 1 ms 3000 times resulting in 3 seconds of data.

An SConstruct file located within sigsbee/fdmod2A/ properly formats the model and inputs necessary parameters to perform a shot on the Sigsbee model. This file is reproduced below in Table 9.

Typing Command 4 within the sigsbee/fdmod2A/ directory runs the FD modeling script.

\[
\text{bash-3.1$ scons view}
\] (4)

This script first constructs the survey acquisition geometry as was previously mentioned. An image of the survey is created and presented in Figure 9.

![Survey Design Image](sigsbee/fdmod2A vel)

Figure 9: FD model geometry as performed on the Sigsbee 2A velocity model. The X represents the shot while the smaller * symbols represent receivers. The receivers extent along the right hand side although it is not clear in this image.

Firing the shot results the propagation of a wavefield which can be seen in the movie wfl.vpl that is generated. Typing Command 5 within the sigsbee/fdmod2A directory displays the wavefield movie.

\[
\text{bash-3.1$ scons wfl.vpl}
\] (5)
from rsf.proj import *
import fdmod
#
# Sigsbee2A
#
data='sigsbee2a_stratigraphy.sgy'
Flow(['vstr2A','vstr2Ahead'], data, 
... segyread tape=$SOURCE tfile=${TARGETS[1]} |
... put o1=0 d1=25 label1=z unit1=ft o2=10000 d2=25 label2=x unit2=ft
...)
#
par = {
... 'nt':7000, 'dt':0.001, 'ot':0, 'lt':'t', 'ut':'s',
... 'kt':100, # wavelet delay
... 'nx':3201, 'ox':0, 'dx':0.00762, 'lx':'x', 'ux':'km',
... 'nz':1201, 'oz':0, 'dz':0.00762, 'lz':'z', 'uz':'km',
}
#
# add F-D modeling parameters
fdmod.param(par)
#
# Velocity
Flow('vel', 'vstr2A', 
... scale rscale=0.003048 |
... put o1=%%(oz)g d1=%%(dz)g o2=%%(oz)g d2=%%(dz)g |
...)
Plot('vel', fdmod.cgrey('allpos=y bias=1.5 pclip=100 color= Survey \ Design labelsz=4 titlesz=6 wheretitle=t', par))
#
Result('vel', ['vel', 'rr', 'ss'], 'Overlay')
#
# density
Flow('den', 'vel', 'math output=1')
#
# finite-differences modeling
fdmod.awefd('dat', 'wfl', 'wav', 'vel', 'den', 'ss', 'rr', 'free=y dens=y', par)
Plot('wfl', fdmod.wgrey('pclip=99 title=Wavefield \ Movie labelsz=4 titlesz=6 wheretitle=t', par), view=1)
# times=[1,2,3,4] ctntr=0
for item in ['9', '19', '29', '39']:
... Result('time'+item, 'wfl', 
... window f3=5s a3=1 min1=0 min2=0 | grey gainpanel=a 
pclip=99 wantframenum-y title=Wavefield \ at \ %s \ s labelsz=4 
titlesz=6 screenratio=357 screenh=2 wheretitle= 
label1=x label2=y unit1=ft unit2=ft 
% (item, times[ctntr])
... ctntr = ctntr + 1
Result('dat', 'window j2=4 j1=2 | transp | ' + fdmod.dgrey('pclip=99', par))
End()
Four frames from this movie are presented in Figure 10 illustrating the propagation of the wavefield in the model.

![Wavefield at 1 s](image1)

![Wavefield at 2 s](image2)

![Wavefield at 3 s](image3)

![Wavefield at 4 s](image4)

Figure 10: Images of the propagating wavefield in the Sigsbee model generated by a finite difference model. sigsbee/fdmod2A time9,time19,time29,time39

The resulting data is then presented in the file `dat.vpl`. This plot is reproduced here in Figure 11.

FD models can be performed on the Sigsbee2B model in a similar fashion. The primary change would be in appending line six, the model input file, in the `SConstruct` file shown in Table 9.
Figure 11: Data gathered by the receivers in the FD model survey.
sigsbee/fdmod2A.dat