Analysis of T. J. Hughes Seismic Data

Semester Project

GEOL 4068 – Reflection Seismology

Dr. Juan Lorenzo

December 9, 2010

Introduction

Reflection seismology is, through the analysis of mechanical waves, the science of examining the earth's interior. It uses the principles of seismology to estimate the properties of the Earth's subsurface from reflected seismic waves. The method requires a controlled seismic source of energy, such as an explosion, an impact, or a seismic vibrator. The waves are bent, reflected, refracted, diffracted, and scattered. These waves are then received by hydro or geophones located in water or on the ground surface/seafloor, respectively.

Acquisition

Data acquisition is most often done by a string of sensors extending away from the seismic source. For land data, the singular receiver unit is a geophone sensitive to vertical ground motion (single units that measure motion in 2 or 3 axial directions are also available). As the geophone moves with the ground, a conductor is moving in a magnetic field inside the geophone. By doing so, the device generates a current proportional to the particle velocity of the earth surface. The result is electrical currents going ultimately to an analog-to-digital converter. When a shot fires, each group of geophones will begin sending analog data, which is digitized to form digital data. Where the analog-digital conversion takes place depends on the equipment in use. Data is ultimately stored in a data recording unit. Often, data is collected for a shot then the geophones and the source signal are moved systematically in a direction that is aligned axially with the geophone layout. Doing this achieves signals from a common depth point (CDP) which is termed common mid-point (CMP) shooting.

About the Data

The T.J. Hughes dataset was obtained from a land seismic reflection survey along a Mississippi River levee. The data was located on the machine LGC01 in the seismic reflection laboratory. The network address of the machine was 'lgc01@geol.lsu.edu'. The original data was found in the directory '/home/refseis10/LSU1_1999_TJHughes/seismics/data/1999/Z/dat'. The first shotpoint gather was more northerly than the last shotpoint gather. Below is a table showing information about the experimental layout and the data.

Number of shot gathers:	100
Offset of 1st geophone from shot:	4.5 meters
Geophone spacing:	3 meters
Number of traces per shot:	24
Shift between successive shots:	3 meters
Recording duration:	1s
Samples per trace:	2048
Delay between shot and recording start:	10 milliseconds before shot
Time between samples:	500 microseconds

Processing

Seismic Unix

Seismic Un*x (SU) is a free seismic reflection analysis software package distributed by the Center for Wave Phenomena (CWP) at the Colorado School of Mines (CSM). SU runs on several operating systems, including Linux, Microsoft Windows, and Apple Macintosh, In this class we used seismic unix in Linux. SU is maintained and regularly updated by CWP at CSM. The SU home page at the CWP is: http://www.cwp.mines.edu/cwpcodes . SU can be downloaded from the CWP ftp server and installed on almost any Unix system. It is possible for one to adjust and expand SU since the source code is also included.

Converting .dat to .su

Before the data can be processed by Seismic Unix it must be converted from .dat format to .su format. This is accomplished by using the script convert.pl to convert data from .dat (.sg2) format to .segy format to .su format. Essentially, .segy data is the same as .su data, but .segy has a 3200 byte header and 400 byte header at the beginning of the shot gather. The script convert.pl can be found in the appendix.

Deleting Bad Traces

Before the frequency spectrum can be analyzed bad traces should be deleted. Bad traces often have noise at high frequencies which will induce false high frequencies into the frequency power spectrum. Below is a table showing trace deletions:

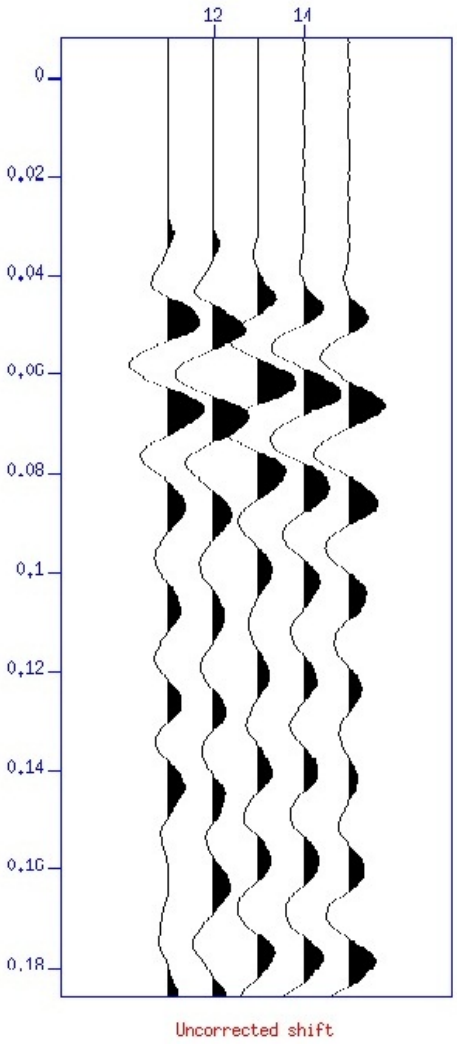
File	Killed	File	Killed	File	Killed	File	Killed	File	Killed	File	Killed
1001	2	1014	19	1050	20	1079	11,12	1087	11,12	1095	11,12
1002	1	1015	18	1051	19	1080	11,12	1088	11,12	1096	11,12
1006	All	1039	23	1052	18	1081	11,12	1089	11,12	1097	11,12
1009	24	1040	22	1053	17	1082	11,12	1090	11,12	1098	1-12
1010	23	1046	All	1075	11,12	1083	11,12	1091	11,12		
1011	22	1047	23	1076	11,12	1084	11,12	1092	11,12		
1012	21	1048	22	1077	11,12	1085	11,12	1093	11,12		
1013	20	1049	21	1078	11,12	1086	11,12	1094	11,12		

Killing bad traces was accomplished using the program sukill. File names were changed on good files to be consistent with files that needed traces killed (e.g. 1002.k.su , 1003.k.su) Trace kills and file name changes were done manually.

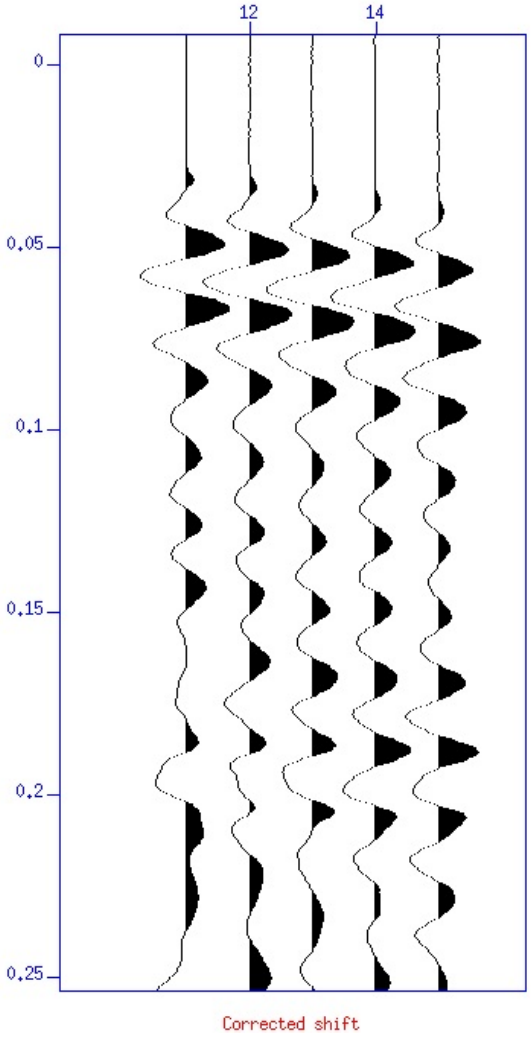
Reversing Traces

Due to a hardware error one half of the traces within a gather were out of phase with the other half. This caused an abrupt shift in the data within each gather. This would also cause

errors later in processing, such as when traces of the same CMP were stacked. Correcting this problem was accomplished by windowing the good and bad half of the data then reversing (phase shift of 180°) the bad data. The good data and reversed bad data are then concatenated to form a single corrected file. The script used to do this (rev.sh) can be found in the appendix. Shown below is an example of the uncorrected and corrected shift in the data.



Uncorrected shift

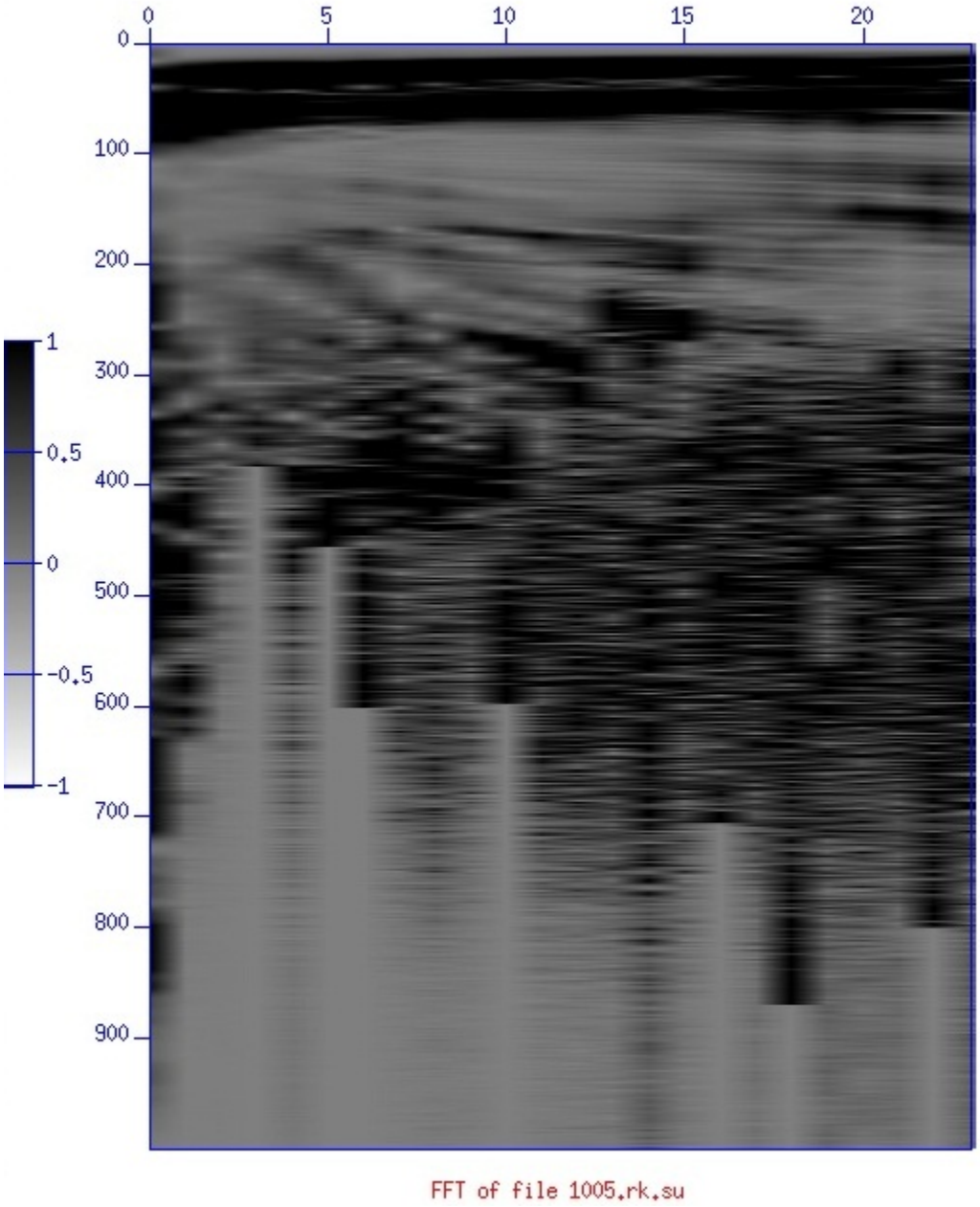


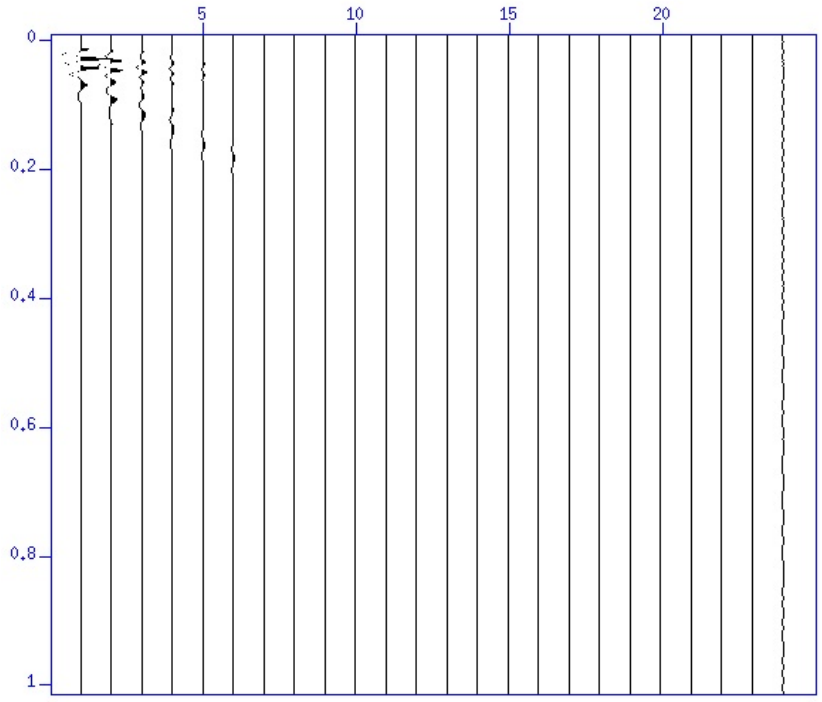
Corrected shift

Gain and Bandpass Filtering

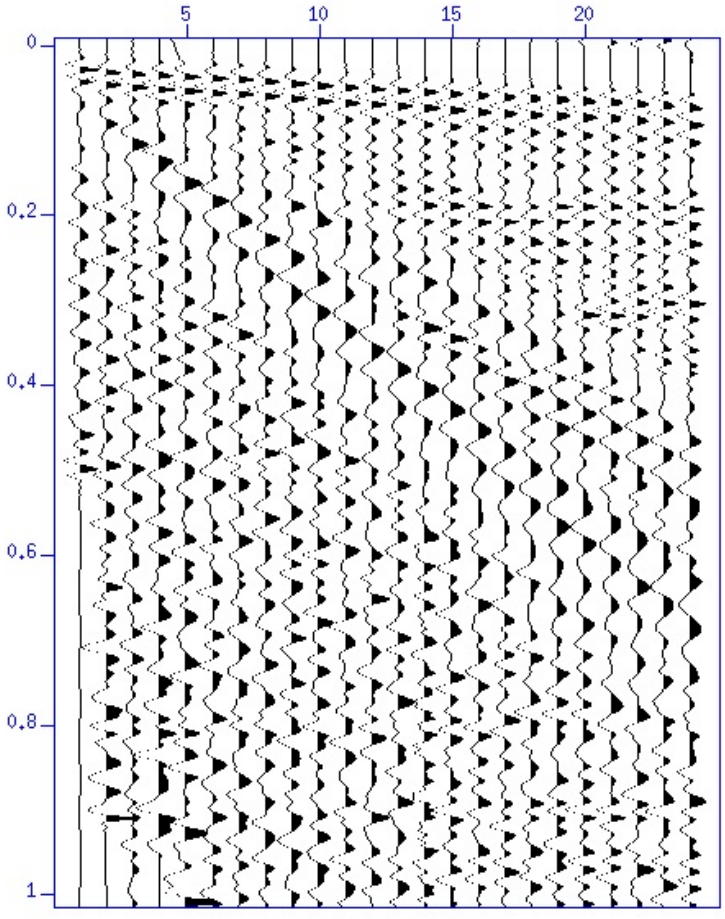
The next step in processing was gaining and bandpass filtering the data. Gaining the data corrects for the loss of signal due to spherical divergence by adjusting signal for each trace to a consistent level for the whole gather or .su file. Without gaining, the file is virtually unviewable. Gaining is achieved by utilizing the sugain command with automatic gain control on, agc=1, and a wagc value of 0.1 (should be approximately 1/10 length of dataset in seconds) Filtering is needed to take out unwanted frequencies from the signal. Most of the signal was concentrated in frequencies between 10 and 100 Hz. (See Fast Fourier Transform below) The

frequencies chosen for the bandpass filter were 5, 10, 100, 150 (f=5,10,100,150) Also shown below is data pre and post gaining and filtering. (gainandfilter.sh in appendix)





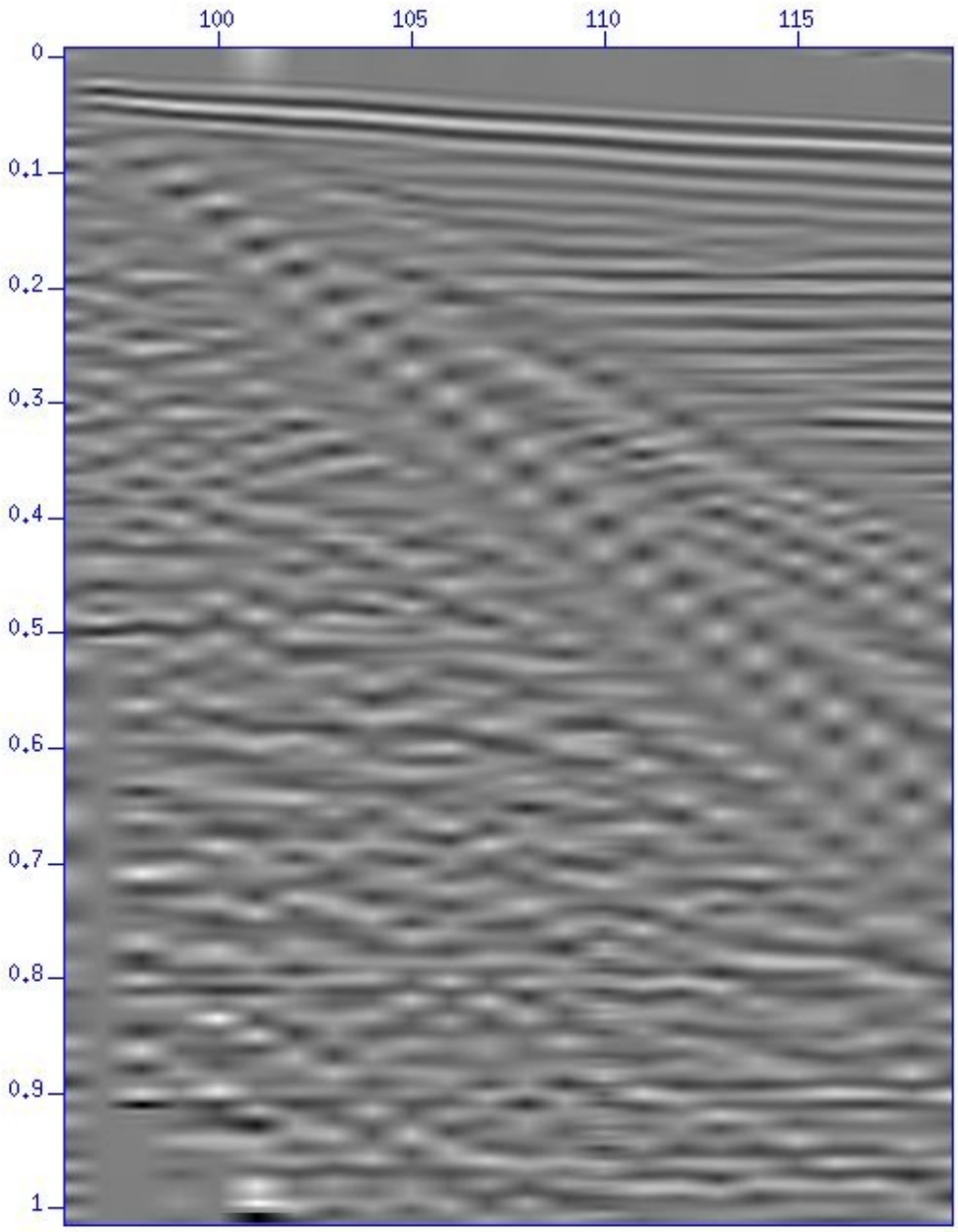
Data with no gain or filter



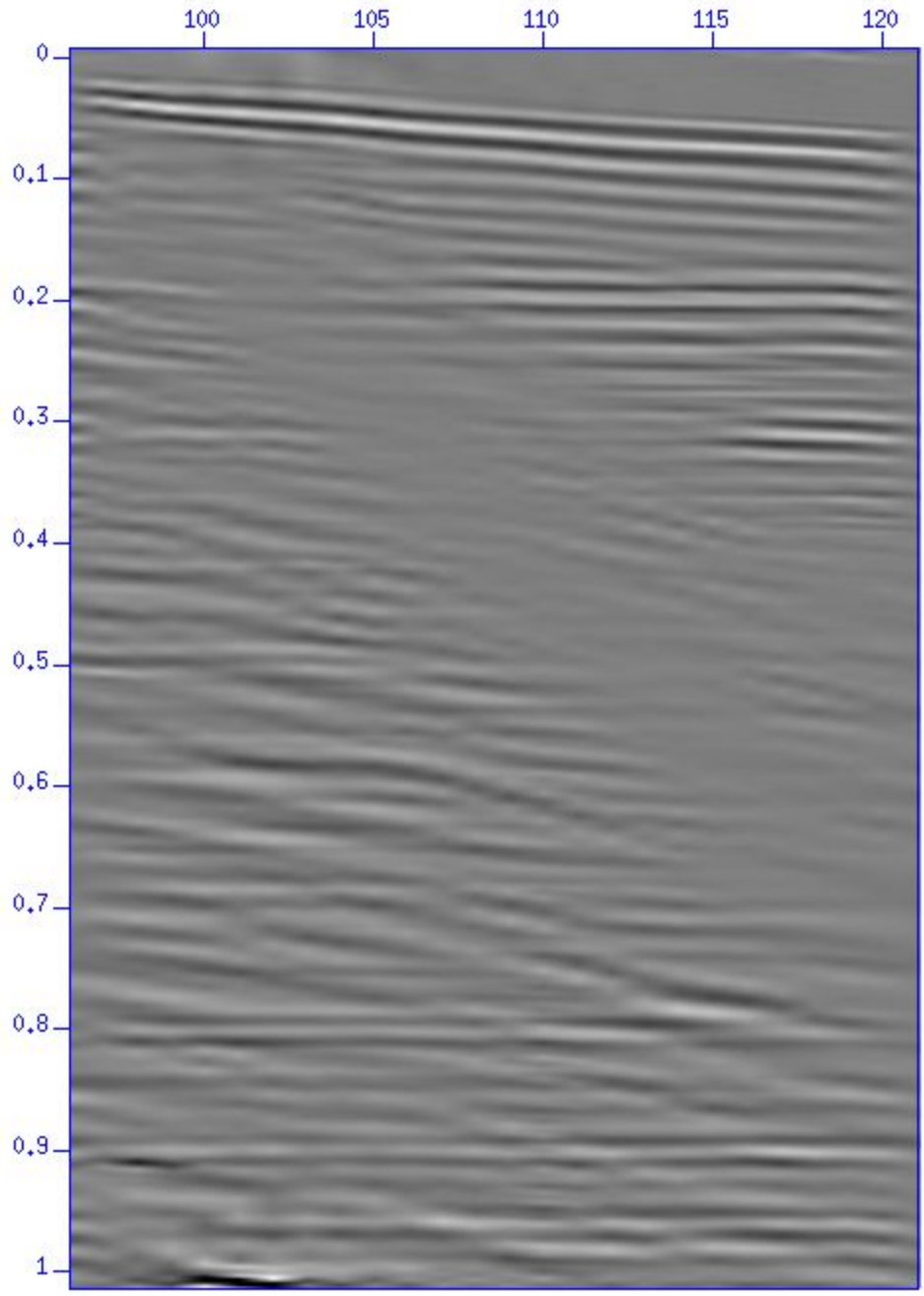
Data gained and filtered

Minimizing Ground Roll

Ground roll is the direct arrival of a Rayleigh wave. This often cuts directly through the data diagonally making it difficult to see features under the ground roll arrival. Ground roll is best taken care of using f-k filtering. Filtering in the f-k domain is based on samples per trace slopes that you wish to eliminate from the data. The slopes I chose to eliminate were those from 30 samples per trace to 60 samples per trace. Most all negative slopes were eliminated from -100 to 0. Below are images of data before and after f-k filtering. Filtering isachieved by the use of the program sudipfilter.pl (in appendix). Shown below are pre and post f-k filtering.



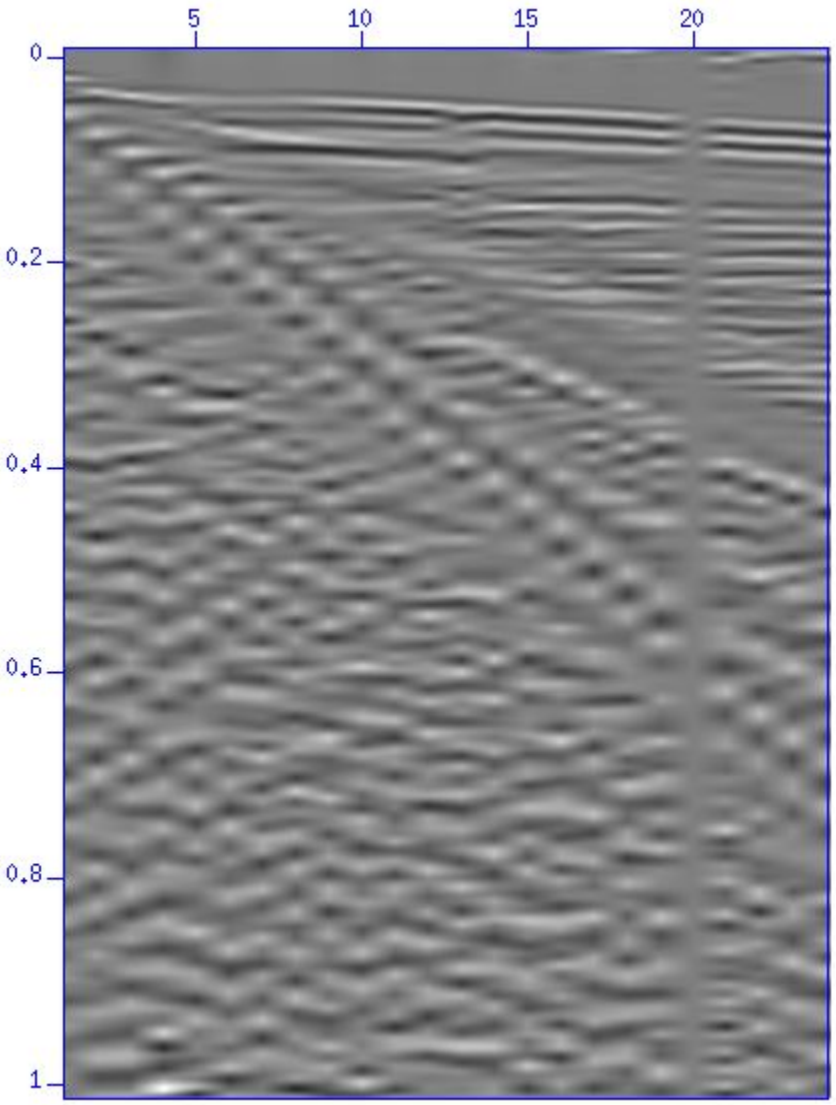
No f-k filtering



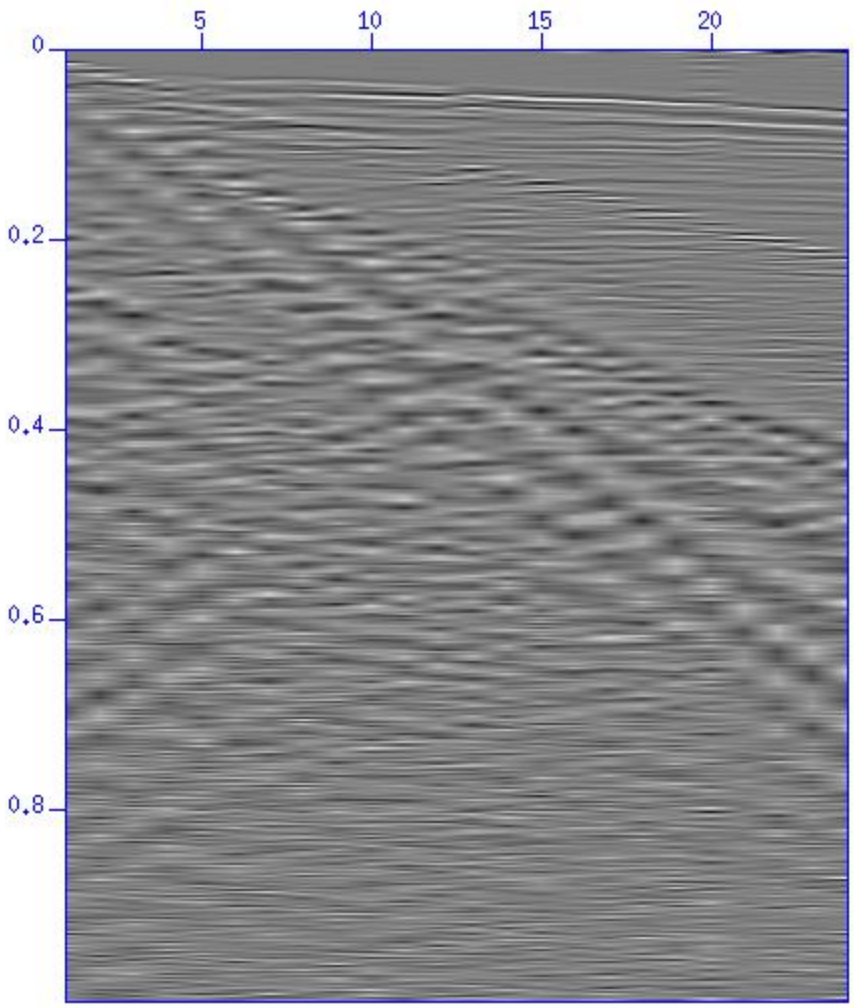
Whitening Spectrum

Spectrum whitening is achieved using spiking de-convolution. Spiking de-convolution makes peaks in the data more narrow and taller. This is done in the hope of improving the level of detail within the data. Using the program SpikDecon_P.pl script which uses the SU program called supef with values for minlag and maxlag equal to 0.00050s and 0.0125, respectively, no

improvement in visualizing the data was seen. Therefore, I chose not to include spiking de-convolution in processing. Below are images of files before and after spiking de-convolution.



Without Spiking Deconvolution



Header Geometry

Header geometry was created using the script make_header_geometry.sh which utilizes the program sushw. The correct values were created for shot location (sx), geophone location (gx), and offset, which equal to gx-sx. These values are needed to calculate appropriate values for common depth point (CDP). Units for these values were in centimeters due to the fact that values must be expressed as whole integers. This was taken into account later when calculating stacking velocities.

Calculating CMP's

Common mid-points (CMPs) are equal to CDP's when dealing with horizontal strata. CDP's were calculated using the script makeCMP.sh which utilizes the suchw program. CDP's are needed to determine stacking velocities which is the next step.

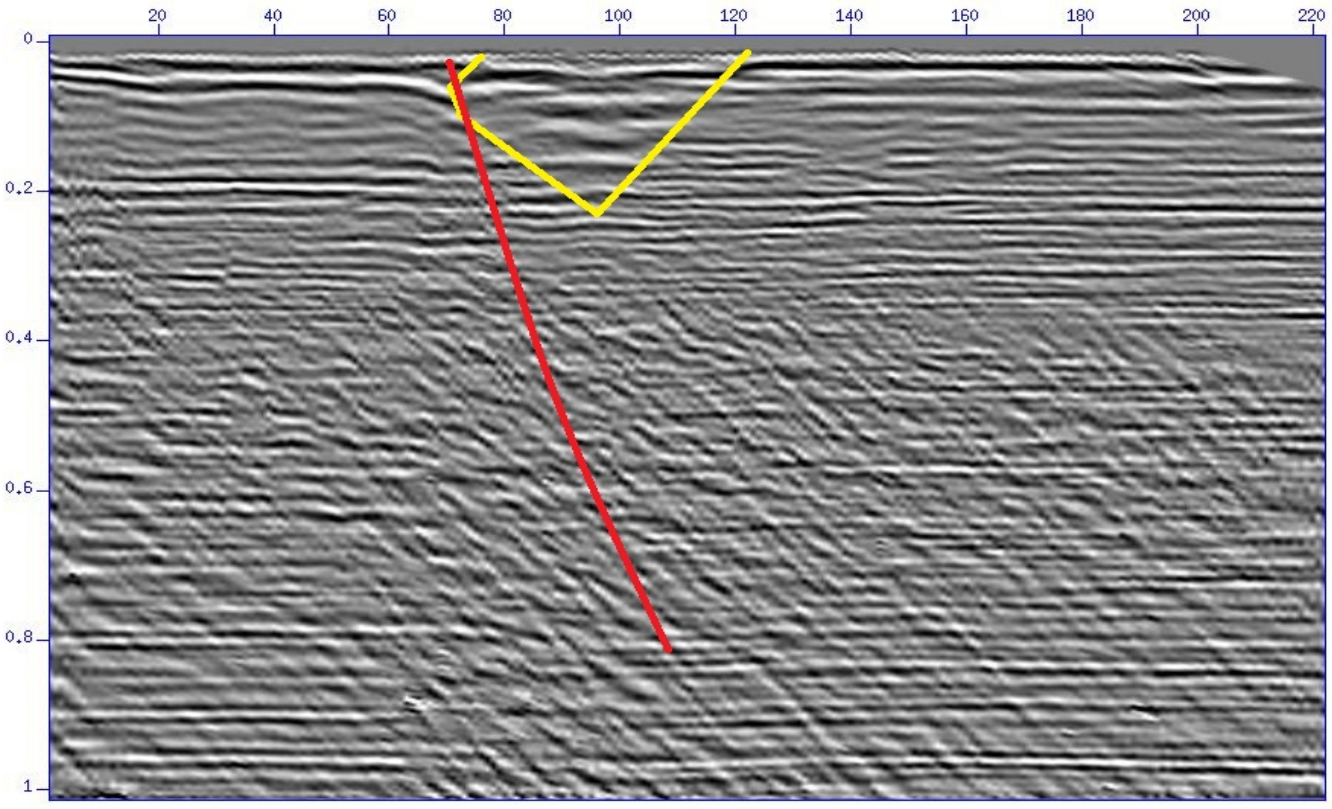
Stacking Velocities

Stacking Velocities are needed to correct the curvature of seismic data called normal moveout (NMO). Seismic velocities usually increase with depth so a few velocities are chosen for a few travel times in the dataset. Using the script nmo_test.sh stacking velocities were choosen using a CDP directly in the middle of the dataset (cdp=18375). Velocities from 100000 cm/s to 200000 cm/s were examined every 10000 cm/s. The table below shows the values that were tnmo and vnmo:

Vnmo (m/s)	Tnmo (s)
1100	0.8093
1300	0.1525
1500	0.3150
1800	0.6000

Stacking Final Image

The final image was created from the script simple_stack.sh. The concatenated traces were sorted by CDP and offset, stacking velocities were applied, and the traces were stacked using the time and velocity values in the previous section. Stacking traces of the same CDP removes noise from the trace. The final image is then analyzed for key features. Below is the final image.



Final Image

Conclusions

The final image is lacking significant features and at first glance appears that not much information can be taken from the image. However, a clear failed wedge of soil (yellow) is found at the very surface. There also may be a fault line extending from the left (northern) side of the failure wedge down into the soil below. Reflections are almost perfectly horizontal indicating that choices for stacking velocities were good. This concludes the analysis of the T. J. Hughes dataset.

References

Liner, Christopher L. (2004). *Elements of 3D Seismology – Second Edition.* Tulsa, Oklahoma: PennWell Corporation

David Forel, Thomas Benz, and Wayne D. Pennington (2005). *Seismic Data Processing with Seismic Un*x – A 2D Seismic Data Processing Primer*. Tulsa, Oklahoma: Society of Exploration Geophysicists

Lorenzo, Juan. GEOL 4068 Seismic Reflection – Lecture Notes

Reflection seismology - Wikipedia. Retrieved December 8, 2010, from http://en.wikipedia.org/wiki/Reflection_seismology

Appendix

Convert.pl

```
#! /usr/bin/perl


# PROGRAM NAME : convert.pl
# SEG2SU
# This file does the following:
# It runs perl scripts which convert
# a SUnix seg2 binary file to segy file for
# a PC

# INPUT SEG2 DIRECTORY
$DATA_seismics_SEG2 =
'/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/dat';


# OUTPUT SU DIRECTORY
$DATA_seismics_SU   =
     '/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su';

# OUTPUT SEGY DIRECTORY
```

```
$DATA_seismics_SEGY =
     '/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/sgy';

     @mv2segyDIR = (" mv                          \\
                  $DATA_seismics_SEG2/*.sgy   \\
                  $DATA_seismics_SEGY         \\
                 ");

# if number of files =8 but first file is "1004.su"
$number_of_files = 1;

# INPUT FILE NAMES
for ($i=1,$j=1002; $i <=$number_of_files  ; $i += 1,$j +=1){
     $j_char = sprintf("%u",$j);
     $file_name[$i]        = $j_char;
    }

# CONVERT SEG2 FILES TO SEGY FILES
for ($i= 1 ; $i <= $number_of_files  ; $i += 1) {

     @seg2segy = (" ./seg2segy $file_name[$i].SG2   1 ");

     system @seg2segy;

     system 'echo', @seg2segy;

    }

# MOVE SEGY FILES TO SEGY DIRECTORY
     system @mv2segyDIR;

     system 'echo', @mv2segyDIR;



# CONVERT SEGY FILES TO SU FILES
for ($i= 1 ; $i <= $number_of_files ; $i += 1) {

     @seg2su =  (" segyread                                          \\
                tape=$DATA_seismics_SEGY/$file_name[$i].sgy  \\
                endian=0
\\
                > $DATA_seismics_SU/$file_name[$i].su             \\
               ");

     system @seg2su;

     system 'echo', @seg2su;

    }
```

Rev.sh

```
#!/bin/sh
set -x
# rev.sh
# Reverses traces 13 through 24
# James Chatagnier
# November 22, 2010

# set up working directories

SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'

first=1001
last=1100


#rm $SU_DIR/$output_file.su

for ((file=$first; file<=$last; file=$file + 1))
     do

   suwind <$SU_DIR/$file.su \
          key=tracf              \
          min=1 max=12           \
         > $SU_DIR/$file.temp_1to12.su

   suwind <$SU_DIR/$file.su  \
          key=tracf min=13 max=24 \
     |                     \
   suop    op=neg                   \
         > $SU_DIR/$file.temp_13to24.su

  cat $SU_DIR/$file.temp_1to12.su \
      $SU_DIR/$file.temp_13to24.su \
      > $SU_DIR/$file.r.su

     done

rm -f $SU_DIR/*temp*

# plotting concatenated data

#  sugain <$SU_DIR/$output_file.rev.su           \
#          agc=1 wagc=0.1             \
#       |                        \
#  sufilter f=3,6,100,160             \
#       | suxwigb title="$output_file'_polrev'.su"
```

Sufft.sh

```
#!/bin/sh

# Program Name: SUFFT script
# Programmer:James Chatagnier
# Purpose:Displays Fast Fourier Transform for a file
# Version:1
# Date: December 3, 2010

# DATA DIRECTORY
SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'


# fourier analysis fo data
file_name='1010.su'

sufft <$SU_DIR/$file_name        \
| suamp      mode=amp            \
| sugain     wagc=.1   agc=1          \
| suximage legend=1   clip=1
#| suxwigb legend=1 clip=5
```

Gainandfilter.sh

```
#! /bin/sh
set -x

#Filename:gainandfilter.sh
#Purpose: Gains and filters files, makes new file
#Written by: James Chatagnier
#Date:December 5, 2010

#Set up working Directory and files

SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'

first=1001
last=1100

#Loop to apply gain and filter

for ((file=$first; file<=$last; file=$file + 1))
      do
      sugain < $SU_DIR/$file.rk.su               \
            agc=1                 \
            wagc=0.1              \
            | sufilter            \
            f=5,10,100,150              \
            > $SU_DIR/$file.rkgf.su
      done
```

concatall.sh

```
#!/bin/sh
set -x
# cat.sh
#March 24 2010
# Program to concatenate many files together
# and review the results
# Juan M. Lorenzo

# set up working directories
SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'
output_file='1001_1100.rk'

rm  $SU_DIR/$output_file.su
touch $SU_DIR/$output_file.su


# cat all S from E and W

first=1001
last=1100

for ((file_num=$first; file_num<=$last; file_num=$file_num+1))
   do
     cat $SU_DIR/$file_num.rk.su >> $SU_DIR/$output_file.su
   done


#sugain <$SU_DIR/$output_file.su agc=1 wagc=0.5        \
#|                                      \
#sufilter f=3,6,1000,1500
#suximage  clip=5 &
```

---

sudipfilt.pl

```
#! /usr/bin/perl

# SCRIPT NAME
# Suspecfk.pl
# Purpose: f-k spectral analysis
# Juan M. Lorenzo
# Feb 15 2008


# Use shell transparently to locate home directory before compilation

        my $library_location;

        BEGIN {
                use Shell qw(echo);

                $home_directory = ` echo \$HOME`;
                chomp $home_directory;
```

```
                $library_location = $home_directory.'/lsu/libAll';
        }

# LOAD GENERAL PERL LIBRARY
  use lib $library_location;

# library path
  use lib './libAll';

# use library
  use System_Variables2;

# import system variables
  my ($DATA_SEISMIC_SU) = System_Variables2::DATA_SEISMIC_SU();

        #sample rate = us
        # d1 = sample rate in s = .000XXX

# sufile names
  $sufile_in[1]       = '1001_1100.rk';
  $sufile_out[1]           = $sufile_in[1].'gf_fk';
  $inbound [1]        = $DATA_SEISMIC_SU.'/'.$sufile_in[1].'.su';
  $outbound [1]            =
$DATA_SEISMIC_SU.'/'.$sufile_out[1].'.su';
#print("$sufile_in[1]\n");


# GAIN DATA
      @sugain[1]  =  (" sugain                          \\
           pbal=1                               \\
               ");

# GAIN DATA
      @sugain[2]  =  (" sugain                          \\
           wagc=0.1                        \\
           agc=1                           \\
               ");

# FILTER  DATA
      @sufilter[1] =  (" sufilter                               \\
           f=5,10,100,150                  \\
               ");



# WINDOW  DATA by shot point
# in this case fldr
#  is equivalent to sequential shot point gather number
      @suwind[2] =  (" suwind                           \\
           key=fldr                        \\
           min=1001                        \\
           max=1100                        \\
```

```
              ");

# F-K SPECTRAL ANALYSIS
     @suspecfk[1] =  (" suspecfk                                \\
               dt=1 dx=1                  \\
              ");

# LINEAR MOVEOUT
     @sureduce[1] =  (" sureduce                                \\
               rv=1.5                          \\
              ");

# LINEAR MOVEOUT
     @sureduce[2] =  (" sureduce                                \\
               rv=-1.5                         \\
              ");


# APPLY DIP FILTER
     @sudipfilter[1] =  (" sudipfilt                            \\
          dt=1 dx=1                          \\
          amps=1,0,0,1                            \\
          bias=0                                  \\
          slopes=15,25,65,75                        \\
              ");

# APPLY DIP FILTER
     @sudipfilter[2] =  (" sudipfilt                            \\
          dt=1 dx=1                          \\
          amps=1,0,0,1                            \\
          bias=0                                  \\
          slopes=-100,-90,-5,0                    \\
              ");


# DISPLAY DATA
          #key=offset                             \\
     @suxwigb[1] =  (" suxwigb                               \\
          title=$sufile_in[1]                     \\
          label1='No. samples'                    \\
          label2='No. traces'                     \\
          d1=1 d2=1 f1=1 f2=1                     \\
          wbox=300 hbox=370 xbox=370 ybox=0            \\
          n2tic=1 d2num=20                \\
          va=1                            \\
          xcur=3                                  \\
          clip=2.5                        \\
          ");

# DISPLAY DATA
          #key=offset                             \\
     @suxwigb[5] =  (" suxwigb                               \\
```

```
            title=$sufile_in[1]                    \\
            label1='No. samples'                   \\
            label2='No. traces'                    \\
            d1=1 d2=1 f1=1 f2=1                    \\
            wbox=300 hbox=370 xbox=370 ybox=440          \\
            n2tic=1 d2num=20                  \\
            va=1                              \\
            xcur=3                                 \\
            clip=3                                 \\
            ");

# DISPLAY DATA
      @suximage[1] =  (" suximage                  \\
            title=$sufile_in[1]                    \\
            style=seismic                          \\
            x1beg=0.5 x1end=0.                     \\
            label1='Frequency (Hz) dt=1 Nf=0.5'          \\
            label2='k (1/m) dx=1 Nk=0.5'                 \\
            n2tic=1 d2num=0.2 f2num=-0.5                 \\
            n1tic=1 d1num=0.1                      \\
            wbox=300 hbox=370 xbox=0 ybox=0              \\
            ");

# DISPLAY DATA
      @suximage[2] =  (" suximage                  \\
            title=$sufile_in[1]                    \\
            label1='Time (s)'                      \\
            label2='No. traces'                    \\
            n2tic=1 d2num=20                  \\
            wbox=300 hbox=370 xbox=670 ybox=0            \\
            ");

# DISPLAY DATA
      @suximage[4] =  (" suximage                  \\
            title=$sufile_in[1]                    \\
            x1beg=0.5 x1end=0.                     \\
            label1='Frequency (Hz) dt=1 Nf=0.5'          \\
            label2='k (1/m) dx=1 Nk=0.5'                 \\
            n2tic=1 d2num=0.2 f2num=-0.5                 \\
            n1tic=1 d1num=0.1                      \\
            wbox=300 hbox=370 xbox=0 ybox=440            \\
            ");

# DISPLAY DATA
      @suximage[6] =  (" suximage                  \\
            title=$sufile_in[1]                    \\
            label1='Time (s)'                      \\
            label2='No. traces'                    \\
            d1=XX f1=0                        \\
            n2tic=1 d2num=20                  \\
            wbox=300 hbox=370 xbox=670 ybox=440          \\
            ");
```

```
#  DEFINE FLOW(S)
     @flow[1] = ("                                  \\
           @suwind[2]                               \\
           < @inbound[1] |                          \\
           @sugain[2] |                             \\
           @sufilter[1] |                           \\
           @suspecfk[1] |                           \\
           @suximage[1]                             \\
           &                                  \\
           ");

#  DEFINE FLOW(S)
     @flow[2] = ("                                  \\
           @suwind[2]                               \\
           < @inbound[1] |                          \\
           @sugain[2] |                             \\
           @sufilter[1] |                           \\
           @suxwigb[1]                                        \\
           &                                  \\
           ");

#  DEFINE FLOW(S)
     @flow[3] = ("                                  \\
           @suwind[2]                               \\
           < @inbound[1] |                          \\
           @sugain[2] |                             \\
           @sufilter[1] |                           \\
           @suximage[2]                             \\
           &                                  \\
           ");

#  DEFINE FLOW(S)
     @flow[4] = ("                                  \\
           @suwind[2]                               \\
           < @inbound[1] |                          \\
           @sugain[2] |                             \\
           @sufilter[1] |                           \\
           @sudipfilter[1] |                        \\
           @sudipfilter[2] |                        \\
           @suspecfk[1] |                           \\
           @suximage[4]                             \\
           &                                  \\
           ");


#  DEFINE FLOW(S)
     @flow[5] = ("                                  \\
           @suwind[2]                               \\
           < @inbound[1] |                          \\
           @sugain[2] |                             \\
           @sufilter[1] |                           \\
```

```
          @sudipfilter[1] |                    \\
          @sudipfilter[2] |                    \\
          @suxwigb[5]                          \\
          &                              \\
          ");

#  DEFINE FLOW(S)
     @flow[6] = ("                             \\
          @suwind[2]                           \\
          < @inbound[1] |                      \\
          @sugain[2] |                         \\
          @sufilter[1] |                       \\
          @sudipfilter[1] |                    \\
          @sudipfilter[2] |                    \\
          @suximage[6]                         \\
          &                              \\
          ");


#  DEFINE FLOW(S)
     @flow[7] = ("                             \\
          @suwind[2]                           \\
          < @inbound[1] |                      \\
          @sugain[2] |                         \\
          @sufilter[1] |                       \\
          @sudipfilter[1] |                    \\
          @sudipfilter[2]              \\
          > @outbound[1]                       \\
          &                              \\
          ");

# RUN FLOW(S)
       system @flow[1];
       #system 'echo', @flow[1];

       system @flow[2];
       #system 'echo', @flow[2];

       system @flow[3];
       #system 'echo', @flow[3];

       system @flow[4];
       #system 'echo', @flow[4];

       system @flow[5];
       #system 'echo', @flow[5];

       system @flow[6];
       #system 'echo', @flow[6];

       system @flow[7];
       #system 'echo', @flow[7];
```

SpikeDecon_P.pl

```
#! /usr/bin/perl
# -w
# SCRIPT NAME
# SpikDecon_Vertical.pl
# PURPOSE:
# Spiking Deconvolution
# DATE:
#    Feb 21 2008
# VERSION NUMBER:
#    V1
# DATE
#    May 4 2009
# VERSION NUMBER:
#    V1.1
# AUTHOR:
#    J Lorenzo

# Use shell transparently to locate home directory before compilation

        my $library_location;

        BEGIN {
                use Shell qw(echo);

                $home_directory = ` echo \$HOME`;
                chomp $home_directory;
                $library_location = $home_directory.'/lsu/libAll';
        }

# LOAD GENERAL PERL LIBRARY
  use lib $library_location;

# library path
  use lib './libAll';

# use library
  use System_Variables2;

# import system variables
  my ($PL_SEISMIC)    = System_Variables2::PL_SEISMIC();
  my ($DATA_SEISMIC_SU)    = System_Variables2::DATA_SEISMIC_SU();
  my ($date)          = System_Variables2::date();

# sufile names
   $sufile_in[1]      = '1001.rk_fk';
   $inbound[1]        = $DATA_SEISMIC_SU.'/'.$sufile_in[1].'.su';
```

```
   $outbound[3]       =
$DATA_SEISMIC_SU.'/'.$sufile_in[1].'_spikdecon'.'.su';

  use lib './libAll';


# GAIN DATA
# sugain data
      $text_sugain[1]='pbal ';
      @sugain[1]  =  (" sugain                          \\
           pbal=1                               \\
               ");

# GAIN DATA
# sugain data
      $wagc         = 0.1;
      $text_sugain[2]    = 'wagc='.$wagc;

      @sugain[2]  =  (" sugain                          \\
           wagc=$wagc                      \\
           agc=1                           \\
               ");

# FILTER  DATA
      @bandpass[1]       = '0,10,100,150';
      $text_sufilter[1] = 'bpf '.@bandpass[1];

      @sufilter[1] =  (" sufilter                                \\
           f=@bandpass[1]                  \\
               ");


# WINDOW  DATA by time
      @suwind[1] =  (" suwind                           \\
           tmin=0                          \\
           tmax=1                          \\
               ");

# DECONVOLUTION
# deconvolution data
      # 1 sample=500 us
      $min_lag   = 0.00050;
      $max_lag   = 0.0125;
        $text_supef[1]     = 'Prediction Lag (s) '.$min_lag.' Operator
Lag(s) '.$max_lag;
      @supef[1] =  (" supef                             \\
           minlag=$min_lag                 \\
           maxlag=$max_lag                 \\
               ");

# DISPLAY DATA
# display data
```

```
  $windowtitle  = @sufile_in[1].' '.$date.' Spiking Deconvolution';
           $title1    = 'P';
           $xlabel    = 'offset (m)';
           $tlabel    = 'Time(s)';
           $X0        = 0;
           $widthbox  = 300;
           $xbox_shift     = $widthbox;

# DISPLAY DATA
# display data
      @suxwigb[1] =  (" suxwigb                    \\
           key=offset                         \\
           title='$text_sugain[2] $text_sufilter[1] '  \\
           label1='$tlabel'                   \\
           label2='$xlabel'                   \\
           xbox=$X0                           \\
           wbox=$widthbox                          \\
           windowtitle='$windowtitle'              \\
           clip=5                                  \\
           ");

#modify variables
              $X0    = $X0 + $xbox_shift;

# DISPLAY DATA
# display data
      @suxwigb[2] =  (" suxwigb                    \\
           key=offset                         \\
           title='$text_sugain[2] $text_sufilter[1] $text_supef[1]'
      \\
           label1='$tlabel'                   \\
           label2='$xlabel'                   \\
           xbox=$X0                           \\
           wbox=$widthbox                          \\
           windowtitle='$windowtitle'              \\
           clip=5                                  \\
           ");

#modify variables
              $X0    = $X0+$xbox_shift;

# DISPLAY DATA
# display data
           $windowtitle= @sufile_in[1].' '.$date.' Spiking
Deconvolution';
           $title1 = 'P';
           $xlabel = 'Trace number';
           $tlabel = 'Time(s)';
           $X0   = $X0;
           $widthbox= 300;

      @suximage[1] =  (" suximage                  \\
```

```
          title='$text_sugain[2] $text_sufilter[1] '  \\
          label1='$tlabel'                  \\
          label2='$xlabel'                  \\
          windowtitle='$windowtitle'             \\
          xbox=$X0                          \\
          wbox=$widthbox                         \\
               perc=99                                    \\
               va=1                                       \\
               xcur=3                                     \\
               clip=3                                     \\
          ");

#modify variables
              $X0    = $X0+$xbox_shift;


     @suximage[2] =  (" suximage                \\
          title='$text_sugain[2] $text_sufilter[1] $text_supef[1]'
     \\
          label1='$tlabel'                  \\
          label2='$xlabel'                  \\
          windowtitle='$windowtitle'             \\
          xbox=$X0                          \\
          wbox=$widthbox                         \\
               perc=99                                    \\
               va=1                                       \\
               xcur=3                                     \\
               clip=3                                     \\
          ");

#  DEFINE FLOW(s)
     @flow[1] = (" @suwind[1]                   \\
          < @inbound[1]|                        \\
          @sufilter[1] |                    \\
          @sugain[2] |                          \\
          @suximage[1]                          \\
          &                                 \\
          ");

#  DEFINE FLOW(s)
     @flow[2] = (" @suwind[1]                   \\
          < @inbound[1]|                        \\
          @sufilter[1] |                    \\
          @sugain[2] |                          \\
          @suxwigb[1]                           \\
          &                                 \\
          ");

#  DEFINE FLOW(s)
     @flow[3] = (" @suwind[1]                   \\
          < @inbound[1]|                        \\
          @supef[1] |                           \\
```

```
          @sufilter[1] |                    \\
          @sugain[2] |                           \\
          @suximage[2]                           \\
          &                                 \\
          ");

#  DEFINE FLOW(s)
     @flow[4] = (" @suwind[1]                    \\
          < @inbound[1]|                         \\
          @supef[1] |                            \\
          @sufilter[1] |                    \\
          @sugain[2] |                           \\
          @suxwigb[2]                            \\
          &                                 \\
          ");

#  DEFINE FLOW(s)
     @flow[5] = (" @suwind[1]                    \\
          < @inbound[1]|                         \\
          @supef[1] |                            \\
          @sufilter[1] |                    \\
          @sugain[2]                             \\
          > @outbound[3]                         \\
          &                                 \\
          ");

# RUN FLOW(s)
       system @flow[1];
       system 'echo', @flow[1];

       system @flow[2];
       system 'echo', @flow[2];

       system @flow[3];
       system 'echo', @flow[3];

       system @flow[4];
       system 'echo', @flow[4];

       system @flow[5];
       system 'echo', @flow[5];
```

Make_header_geometry.sh

```
#! /bin/sh
set -x
#Author:James Chatagnier
#Date: December 6, 2010
#Purpose: Generate appropriate headers before generating CMPs (see
make_CMP.sh)
#    offset is defined as sx-gx
```

```
# DATA/$file DIRECTORY
SU_DATA='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'
FILE_NAME='1001_1100.rkgf_fk'

 sushw <$SU_DATA/$FILE_NAME.su   \
      key=sx,gx,offset      \
      a=0,450,450           \
      b=0,300,300           \
      c=300,300,0           \
      j=24,24,24      \
      >$SU_DATA/1001_1100.rkgf.fk.geom.su
```

Makecmp.sh

```
#! /bin/sh
set -x
# Purpose: To generate CMP values in the headers
#     Headers must already have the correct geometry values inserted
for
#     the seismic experiment (See header_geom.sh for this)
#     We use the basic relation that
#
#               CMP = (sx+gx)/2
#
#       where sx is the shot location, and gx is the receiver
location.
#
#     We use suchw to calculate the CMP using offset and other key
words as
#     input.
#            value(key1) = (a + b * value(key2) + c * value(key3)) / d
#     can be rewritten as:
#
#     If we choose the first CMP to be equal to ,say, 101
#       then a = 304
#       a = ( 101 (first CMP number) + 51 (absolute value of
#     half the longest offset on the first shot gather))/2
#       Because d=2 we have to double the size to get a and make
#     our first CMP=101.  You can choose other numbers to be the first
CMP.
#
#            value(cdp)   =(304 +1 * value(sx)   + 1 * value(gx)  ) /
2

# Date: Oct. 25 2007
# Juan Lorenzo


# DATA/$file DIRECTORY
DATA_IN=/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su
```

```
DATA_OUT=/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su
 suchw <$DATA_IN/1001_1100.rkgf.fk.geom.su \
      key1=cdp   \
      key2=sx          \
      key3=gx          \
      a=0        \
      b=1        \
      c=1        \
      d=2        \
 >$DATA_OUT/1001_1100.rkgf.fk.hd.su
```

Nmo_test.sh

```
#!/bin/sh
set -x
# nmo_test.sh
# Oct. 29, 2007
# Program to test nmo's
# several constant velocity moveouts are tested
# starting at 600 m/s and ending at 1000 m/s
# STEP 1: Data is sorted by cdp and offset
#STEP 1A: Data is windowed
# STEP 2: DATA is moved out
# STEP 3: data is filtered
# STEP 4: data is gained
# STEP 5: data is displayed
# Author: Juan M. Lorenzo

# set up working directories
SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'

this_file='1001_1100.rkgf.fk.hd'
counter=0
vel_start=100000
vel_last=200000
vel_inc=10000
first_cmp=18375
last_cmp=18375

for ((vel=$vel_start; vel<=$vel_last; vel=$vel+$vel_inc))

do

      echo $vel
      susort  <$SU_DIR/$this_file.su  cdp offset  \
            |
      suwind key=cdp min=$first_cmp max=$last_cmp  \
            |                              \
      sunmo  vnmo=$vel      \
            |                              \
#     sufilter  f=0,3,400,600                      \
#           |                              \
```

```
#     sugain agc=1 wagc=0.1                 \
#          |
      suximage                    \
                                        \
      xbox=$[$counter*200] ybox=0 wbox=200 hbox=600    \
      title="$vel m/s" &                    \
      counter=$[$counter+1]
done
```

simple_stack.sh

```
#!/bin/sh
set -x

#Purpose: Sorts data by cdp and offset then stacks
#Author: James Chatagnier
#Date: December 7, 2010

SU_DIR='/home/jamec/LSU1_1999_TJHughes/seismics/data/1999/Z/su'
FILE_IN='1001_1100.rkgf.fk.hd'

susort < $SU_DIR/$FILE_IN.su                    \
cdp offset                                \
| sunmo                                 \
      vnmo=110000,130000,150000,180000          \
      tnmo=0.0809,0.1525,0.3150,0.6000      \
| sustack                                \
| sugain agc=1 wagc=.2                         \
| suximage title-‘Final Image’ clip=3
```