Gathering
2005.022

This document describes the general procedure that was used for a particular project. This project had about 600 Texans, about 15 Reftek RT130's, and had both a few land shots and thousands of ship shots per deployment. The Texans were split into three different groups: yellow, red, and green. The different groups recorded at different times during a deployment. These are the colors that are referred to in the instructions. The RT130's recorded continuously, but were deployed and picked up at different times during the whole project.

(TEX) 125proci.sh

Collect all of the raw .TRD files into a directory such as /data/67 (67 was a deployment line). Run 125proci.sh using

```
125proci.sh -p0
```

This script will create several directories and run the program 125_segy on each >TRD file which will convert the raw data to SEGY.

(RT130) ref2segy

The program ref2segy is used to convert the raw RT130 data files to SEGY after the data has been downloaded from the CompactFlash cards. Collect all of the .ref files into one directory (they should be if you use -p0 above) and enter the following lines:

```
foreach file (`ls -1 *.ref`)
ref2segy -f $file
end
```

This small shell program may need to be different if you are not using the tcsh shell on your computer. What it does is loop through the current directory, place the file name of each *.ref file into the variable "file", execute ref2segy on that file, then repeats for the next .ref file. The single quotes in the first line are 'backward accents' (not single quotes), and the "-1" is 'minus-one', not 'minus ell'. ref2segy will create an "R" directory for each Julian day that the raw data file spans. A directory like R035.02 is the data from data stream 2 for day 35. The SEGY files will be in these R-directories. If you are going to use the rgatherer.py program to make receiver gathers you will need to collect all of the SEGY files from all of the R-directories into one directory.

(TEX) clockcor

Once 125proci.sh has finished use the program clockcor, and the 125_SEGY.PCF file created by 125_segy in the step above to time correct the data. In the example /data/67 directory use the command

```
clockcor -d CSEGY PCF/125_SEGY.PCF SEGY/*
```

to correct all of the SEGY files in the /data/67/SEGY directory and place the corrected versions into the /data/67/CSEGY directory. The program accesses each SEGY file in the SEGY directory and then checks the entries in the PCF file to figure out what correction to make. Therefore, if there is a SEGY file for a particular event, for a particular Texan, but no entry in the PCF file for that Texan, then a line stating "No Correction Found" will mean that there was no timing information for that time period/Texan in the PCF file. Sometimes SEGY files are created for Texans with bad data that have a start time in the year 1980.

These will usually cause clockcor to crash. If this happens remove those files from the SEGY directory and restart clockcor.

(TEX) txn2segy

Creates shot gathers. To run txn2segy for shot gathering we need three files:

1. surveyfile (follows the -s command line argument)

The format of this file is:

```
# Survey File
#flag latitude longitude elevation
71001 12.15598 -69.91330 -5.40000
71002 12.13348 -69.90017 -4.40000
71003 12.11412 -69.89505 1.80000
71004 12.09348 -69.88505 0.80000
.
.
71053 12.98706 -70.90876 1.000987
```

The flag, latitude, longitude, and elevation for the SHOT POINTS must also be included in this list in addition to all of the receiver locations. The flag numbers must be in ascending order. The comment lines are optional.

2. dasfile (-d)

```
# DAS file
# flag DAS/sn channel
71001 10793 1
71002 10847 1
71003 10767 1
71004 11028 1
```

The flag numbers must be in ascending order. The comment lines are optional.

3. eventfile (-e)

This file contains the information about a each shot’s start time and points to the dasfile which describes where everything was when the shot went off. The format is:

```
# eventfile
#Flag startime_trace das_file
71053 2004:115:04:31:45.000 dasfile
```

The flag number of the event/shot must match the flag number of the shot's location in the surveyfile. The comment lines are optional. It may be necessary to specify the path (relative or absolute) to the dasfile such as:

```
71053 2004:115:04:31:45.000 ./dasfile
```

This need seems to be dependent on shell settings.

Running txn2segy

To run txn2segy set an environment variable named "SEGYDATA" to the location of the SEGY files to be used. Use the following command (works for tcsh):

```
setenv SEGYDATA /<directory where your SEGY (corrected or uncorrected) files are>
```

for example:

setenv SEGYDATA /data/67/CSEGY/

Run tx2nsegy:

txn2segy –s <surveyfile> -g –l 60 –e <eventfile> –d <dasfile> <output_filename>

- Use -g only if the survey file contains longitude and latitude in decimal degrees and that offset should be calculated using a geodetic inverse routine with WGS-84 as the reference ellipsoid. The positions that end up in the headers of the gather files might be the latitude and longitude in seconds of arc (there may be bugs in this part of the program).
- The -l 60 tells txn2segy to cut the traces to a length of 60 seconds following the start time listed in the eventfile.
- output_filename: will have the extension .SGY added. This file may be looked at using segyVista, Promax, etc.

txn2segy reads through all of the SEGY files it finds and then checks the other lists (dasfile, surveyfile) to see if there is anything to do for that SEGY file.

- THIS MAY HAVE CHANGED -
(RT130 or TEX) segyvista
To run segyvista just enter segyvista on the command line. The graphical user interface is pretty straightforward.

(TEX) segygather

segygather - merges and cuts SEGY files into a common receiver gather.

To run segygather we need four files:

1. Receiver geometry file

Example:
number DAS/C lon lat elevation
72002 10760/1 -69.94257 12.17730 0.10000
72003 10523/1 -69.94257 12.17730 0.10000
72004 10997/1 -69.92583 12.16518 -0.40000
72005 10996/1 -69.91303 12.15617 -1.60000
72006 10739/1 -69.90017 12.13348 -1.80000

It is better to have this file without the azimuth, depth, and wdepth since it makes "funny" conversions that don't seem to have the correct values in the headers of the gather. This file is supposed to contain several geometry lines but it does not seem to work so, it’s better to have an individual file for each DAS. The program rgatherer.py (see below) solves this problem by reading each line of the geomerty file for each DAS. The file must include the header shown (it is used by segygather) and no other comment lines.

2. Geometry shot file

To avoid complications files with the specific shots associated with each color (green, red, yellow) should be generated.

Example (all of the shots during the time range of one "color"):

shot time delay lon lat
1389 04:120:01:35:46.798 0 -69.736040 12.661730
1390 04:120:01:36:52.271 0 -69.735940 12.660370
1391 04:120:01:37:54.913 0 -69.735940 12.659020

1392 04:120:01:38:58.272 0 -69.735920 12.657660

The header line is required and there should be no other comment lines.

3. The segy file list

A list with the path and filename of the SEGY files the program should use and the absolute path to those file, or the path relative to the directory where segygather will be executed from.

Example:
../CSEGY/2004_119_22_45_00_10662_1.RSY
../CSEGY/2004_119_23_45_00_10662_1.RSY
../CSEGY/2004_120_00_45_00_10662_1.RSY
../CSEGY/2004_120_01_45_00_10662_1.RSY

On the command line execute segygather with the necessary parameters

segygather -G –n 60 –I <segyfilelist> –s <shotfilename> –g <receiverfilename> > <output_filename>

-G does the distance conversion in the new version of segygather.
-n 60 is the number of seconds there will be data for after each shot. This may not be 60 for your project.

(RT130 or TEX) rgatherer.py

Since segygather only seems to be able to handle receiver geometry files with one receiver in them, rgatherer.py will read the receiver geometry file with all of the receivers in it, create a geometry file with just one receiver, create any other files that segygather needs, and then run segygather to create a .SGY (gather file) for each receiver.

To run this program you will need the following files:

1. "grouplist.txt"-Contains a list of all of the DASs to be processed
AND which color group they are in

Example:
11123 yellow
11132 green
11121 red

The color is used to create the beginning of the filename for the shot list file (see below) and must be separated from the DAS number by one blank space. There should be no leading blanks on the lines. It is this list that will determine which Texans are processed. If a Texan is not in grouplist.txt then no processing will be done for that Texan.

2. "geometry.txt"-Contains the position and flag number information for all of the DASs.

Example:
number DAS/C lon lat elevation
72002 10760/1 -69.94257 12.17730 0.10000
72003 10523/1 -69.94257 12.17730 0.10000
72004 10997/1 -69.92583 12.16518 -0.40000
72005 10996/1 -69.91303 12.15617 -1.60000

rgatherer.py program will append "/1" to the DAS number from the grouplist.txt file (see above) to find the line in geometry.txt file that it needs. There should be

no leading blanks on the lines and everything should be separated by one blank space. If the DASs are RT130s the lines for channels 2, and 3 may also be included in this file (i.e. specified by /2 and /3 following the DAT ID).

3. "<color>shots.txt"-Contains the list of shots and their parameters that occurred while a color group's Texans were recording

   Example "redshots.txt":

   ```
   shot time delay lon lat
   1389 04:120:01:35:46.798 0 -69.736040 12.661730
   1390 04:120:01:36:52.271 0 -69.735940 12.660370
   1391 04:120:01:37:54.913 0 -69.735940 12.659020
   1392 04:120:01:38:58.272 0 -69.735920 12.657660
   ```

4. "segylist.txt"-Contains a list of all of the SEGY files to be used to make the gathers for all of the DASs. The path should be relative to where rgatherer.py will be executed. rgatherer.py will create lists of the SEGY files for each DAS using this file.

   Example:

   ```
   ../CSEGY/2004_119_22_45_00_10662_1.RSY
   ../CSEGY/2004_119_22_45_00_10663_1.RSY
   ../CSEGY/2004_119_22_45_00_10664_1.RSY
   ../CSEGY/2004_119_22_45_00_10666_1.RSY
   ```

Just run rgatherer.py in the same directory with the above files, answer the questions it asks, and it should do the rest.