

3D geometry loading in ProMAX, a practical crPERL example

Kevin W. Hall, David C. Henley and Hanxing Lu

ABSTRACT

Last year, the radial component data from the Manitou Lake 3D-3C dataset were processed with geometry copied from vertical component data (geometry in the trace headers from the original processor), which have 2.3% fewer traces than the radial component data. This was done in an attempt to expedite the process. Although the difference in the total number of traces seems small, poor radial component results were obtained, reflecting a mismatch of trace headers to trace data. To quickly and accurately load geometry information into ProMAX trace headers from observers notes and shot log files, a Perl script was developed for use with the CREWES Perl Process module. After merging survey data and binning, a visual quality control method was used to spot any problems. A comparison of selected results reaffirms that it is crucial to have geometry correct, in order to obtain good final processing results.

INTRODUCTION

The Manitou Lake 3C-3D data was received as four separate SEG-Y files: vertical component shot gathers with geometry in the trace headers, vertical, inline and crossline shot gathers with no geometry in the trace headers. The vertical component shots with geometry were processed first, before realizing that this dataset had 2.3% fewer traces than the others, likely due to trace kills by the original processor. In the interests of getting a result quickly, the geometry was copied to the radial component anyway. The results were: 1) a consistent horizon could not be interpreted for hand-statics on the radial component receiver stack, 2) imaging resulted in migrated sections with discontinuous reflections, and 3) time slices through the migrated volume contained high-frequency amplitude variations, resulting in a ‘wormy’ appearance (Lu et al., 2006).

In an attempt to improve on the results presented last year, the vertical, inline and crossline datasets were re-processed without reference to the vertical-with-geometry dataset. To do this, we needed a way to quickly and accurately combine and load information from multiple ASCII files on disk to ProMAX trace headers. It was decided that the fastest way to do this was to utilize the CREWES Perl Process module (Bland, 2002).

Geometry loading

We have four sources of geometry information for this survey. The SEG-Y files have field file ID (FFID) in the standard SEG-Y trace headers, as well as source line (S_LINE) and source station (SOU_SLOC) in the optional SEG-Y trace headers. The observers notes have FFID, Record Number (S_LINE*1000+SOU_SLOC; referred to as ‘script’ in the shot log file), ShotID, S_LINE, SOU_SLOC and max/min R_LINE and max/min SRF_SLOC. Alone, the observers notes do not contain enough information to set R_LINE and SRF_SLOC in the ProMAX trace headers, because the receiver patches did not have a consistent number of traces from shot to shot, or from receiver line to receiver line. Fortunately, we also have an I/O shot log report (Appendix A), which has all the

information we need except for FFID. Finally, we have a SEGP file with all the survey information indexed to line and station numbers.

FFID and ShotID were saved as a CSV (comma separated values) file from the observers notes. The ProMAX flow then consists of:

- Disk Data Input (sorted by FFID, excluding bad shots)
- Trace Header Math (create and initialize new trace headers)
- Trace Header Math (fix existing trace headers based on comments in the observers notes)
- CREWES Perl process (runs perlcode.pl; Appendix B)
- Extract Database Files (final run only)

Perlcode.pl (Appendix B) first reads the entire CSV file into an array. For each ensemble from Disk Data Input, it finds the ShotID based on FFID, scans the shot log file until it obtains a match for ShotID, reads all R_LINE and SRF_SLOC information from the corresponding 'Active Seis Map Packet', and sets the ProMAX ShotID, R_LINE and SRF_SLOC trace headers. Debugging consists of printing warnings to debug.csv when 1) the ProMAX and shot log 'script' numbers do not match and 2) when the number of traces in the ProMAX ensemble does not match 'trace count' in the shot log (Figure 1). Note that this process does not require any editing of the shot log file.

This ProMAX flow was run iteratively until all issues were resolved. Most errors were due to repeated shots, and could be avoided by editing the FFID list in Disk Data Input. Finally, the SEGP survey information was merged into the ProMAX database using the 3D land geometry spreadsheet.

QUALITY CONTROL

Accuracy in matching geometry information to seismic traces is of primary importance for the subsequent processing of the traces into useful images, especially for 3D surveys. As shown in the next section, seemingly small errors can lead to severe imaging problems. For this reason, a quality control step applied to the raw seismic data just after loading the geometry is highly recommended. Considering the sheer volume of data handled by many processors, there will always be a temptation to either skip this step, or minimize the time devoted to it.

Many processors have a favorite way of doing quality control on raw data; and hopefully most of these methods are effective. One rudimentary way of doing QC on 3D data, which displays all the traces in a survey, and yet allows the QC step to be done in a reasonable period of time, is to simply read and display the entire data set a shot gather at a time. The key to effectiveness in this procedure is to plot no more than 3 or 4 seconds of data and to plot 3 or 4 key trace headers, in colour, along the top of the trace plot using a tool like Trace Display in ProMAX. Figure 2 shows a display panel from the Manitou Lake 3D seismic survey with source-receiver offset, source-receiver azimuth and receiver line plotted above the seismic.

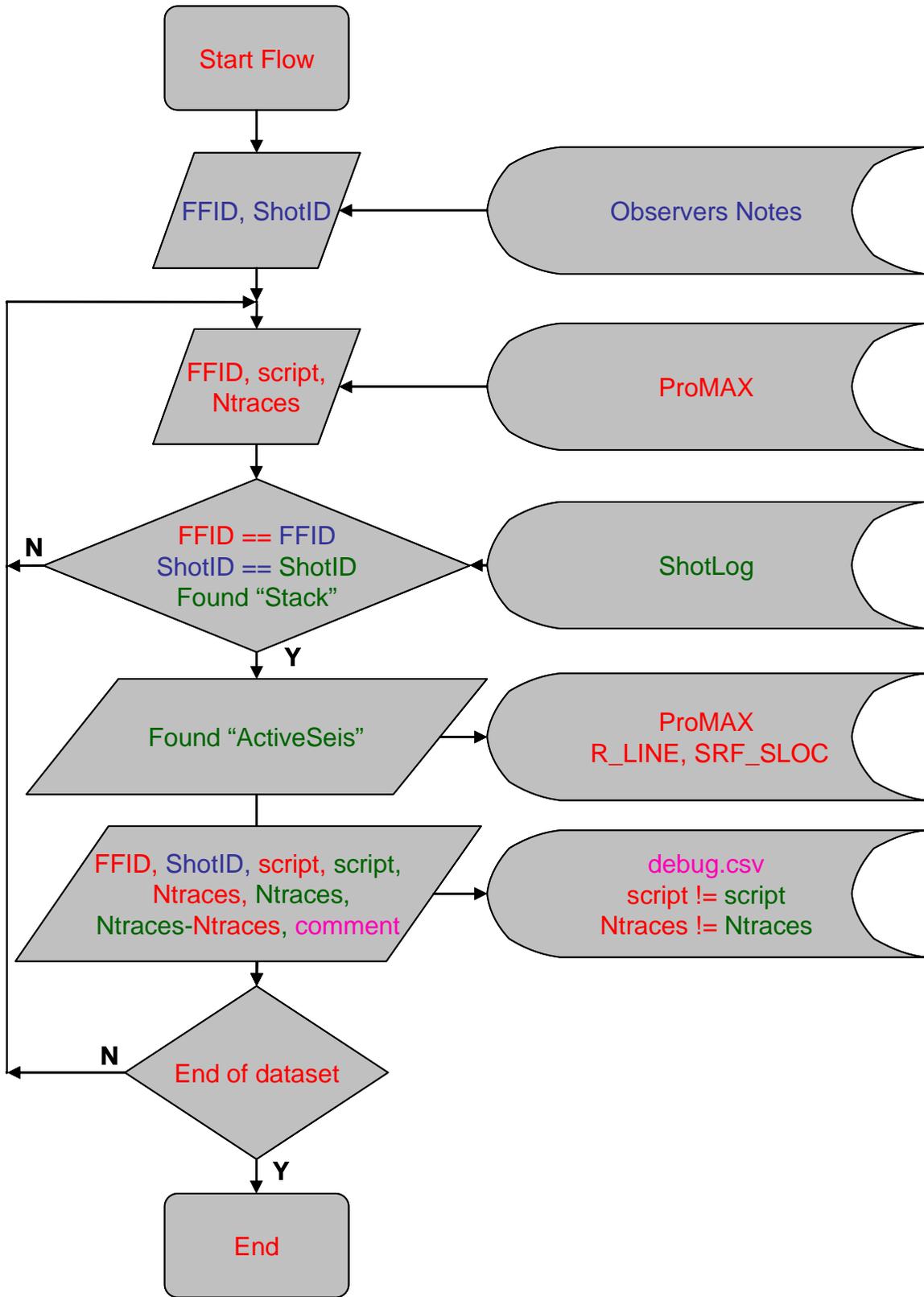
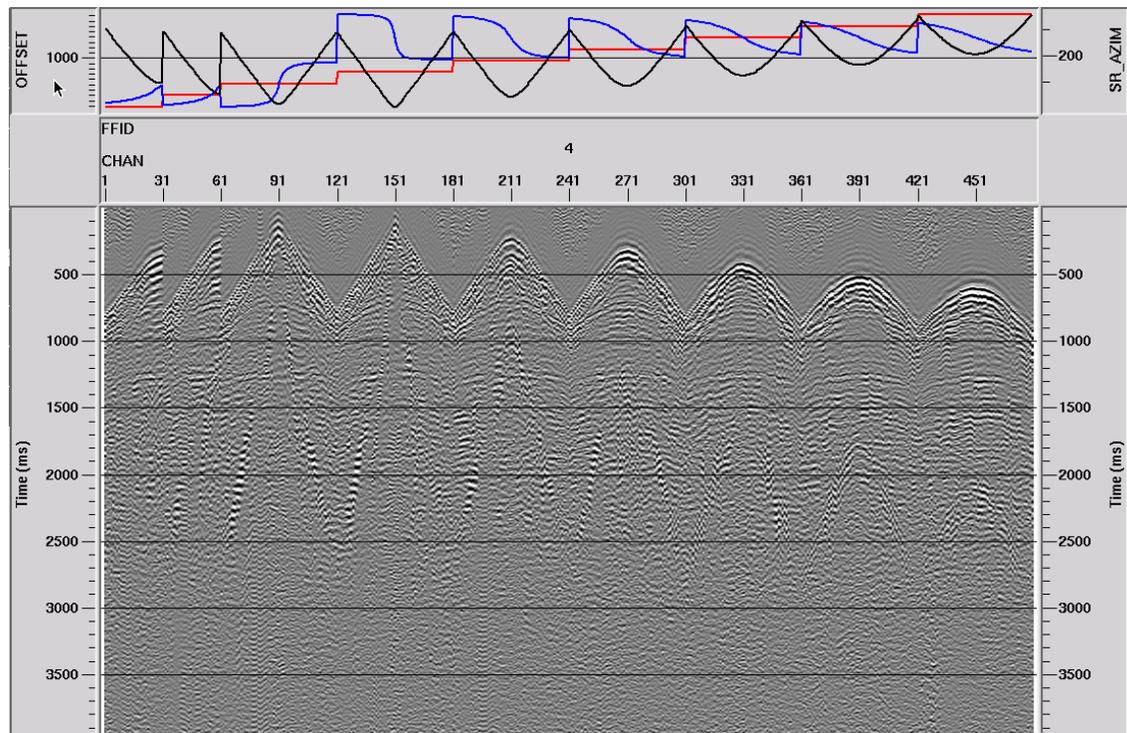


FIG. 1. Flow chart for perlcode.pl (see Appendix B).



Source gather from Manitou Lake 3D survey showing three trace headers plotted above the data. Shown are OFFSET, R_LINE, and SR_AZIM.

FIG. 2. Quality control plot of a shot gather from the Manitou Lake 3D survey. Three trace headers are plotted here, Source-Receiver offset in black, Source-Receiver azimuth in blue and Receiver Line (R_LINE) in red. Any irregularities in these header plots within receiver lines, and any mismatches at receiver line boundaries indicate geometry problems in the data set that must be corrected before further processing.

There are several things to look for on such a display:

- Patterns of first arrivals should be smooth, with no large discontinuities in them except at receiver line boundaries.
- Plots of receiver line number should be flat (constant value) within each receiver line and should show steps only at receiver line ends (red line; Figure 2).
- Plots of source-receiver offset should be smooth curves within receiver lines, unless the offsets have been modified to signed offset (by **rad3d** for application of radial filtering), in which case, they may have discontinuities at the nearest offsets where the sign changes (black line; Figure 2).
- Plots of source-receiver azimuth should be smoothly varying within each receiver line and discontinuous at receiver line boundaries (blue line; Figure 2).

- Plots of other headers, such as X or Y coordinates should be smoothly varying within receiver lines.

The compelling feature of a display like that in Figure 2 is that, with a little practice, it only takes a few seconds to quickly scan such a display and spot problems. Using contrasting colors in the header plots is part of the key to this. With a little practice, a moderately sized 3D survey (say 1000 shot gathers) can be scanned in an hour with a high probability of spotting serious problems. Typically, a problem spotted on one gather will be manifested on more than one gather; so if it isn't spotted on the first gather on which it occurs, it will be seen on subsequent gathers. A problem which occurs only on a single gather is unlikely to create a serious problem for imaging, even if missed by the QC scan.

CONSEQUENCES

Shear statics are notoriously difficult to calculate. One way to deal with the problem is to interpret horizons on a P - S receiver stack generated using P -shot statics and P -receiver statics scaled by V_p/V_s . Time differences required to flatten this horizon can be used as an additional correction term (hand-statics) for the S -receiver statics. Figure 3 shows a comparison between the Manitou Lake P - S receiver stack with an incorrect geometry (as discussed above), and one with a correct geometry. When the geometry is correct, the reflections are higher frequency, noticeably more continuous, and therefore easier to interpret.

An immediate consequence of improved S -receiver statics can be seen in time-slices through the migrated volume at the target time. The time-slice through data with a correct geometry (Figure 4b; Lu et al., 2007) looks more geologically reasonable than our initial result (Figure 4a; Lu et al., 2006).

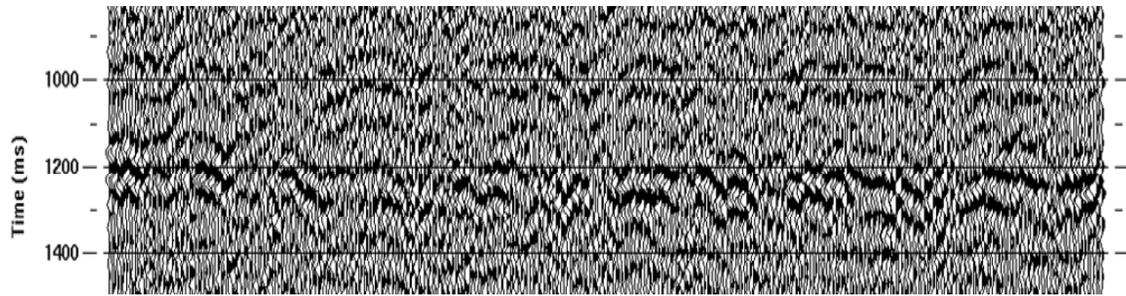


FIG. 3a. *P-S* receiver stack obtained using *P-P* shot statics and *P-P* receiver statics scaled by $V_P/V_S = 2.3$. (Incorrect geometry; Lu et al., 2006).

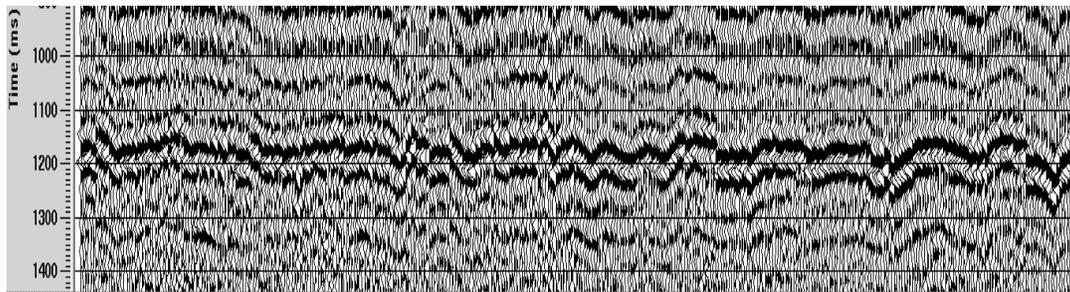


FIG. 3b. *P-S* receiver stack obtained using *P-P* shot statics and *P-P* receiver statics scaled by $V_P/V_S = 2.3$. (Correct geometry; Lu et al., 2007).

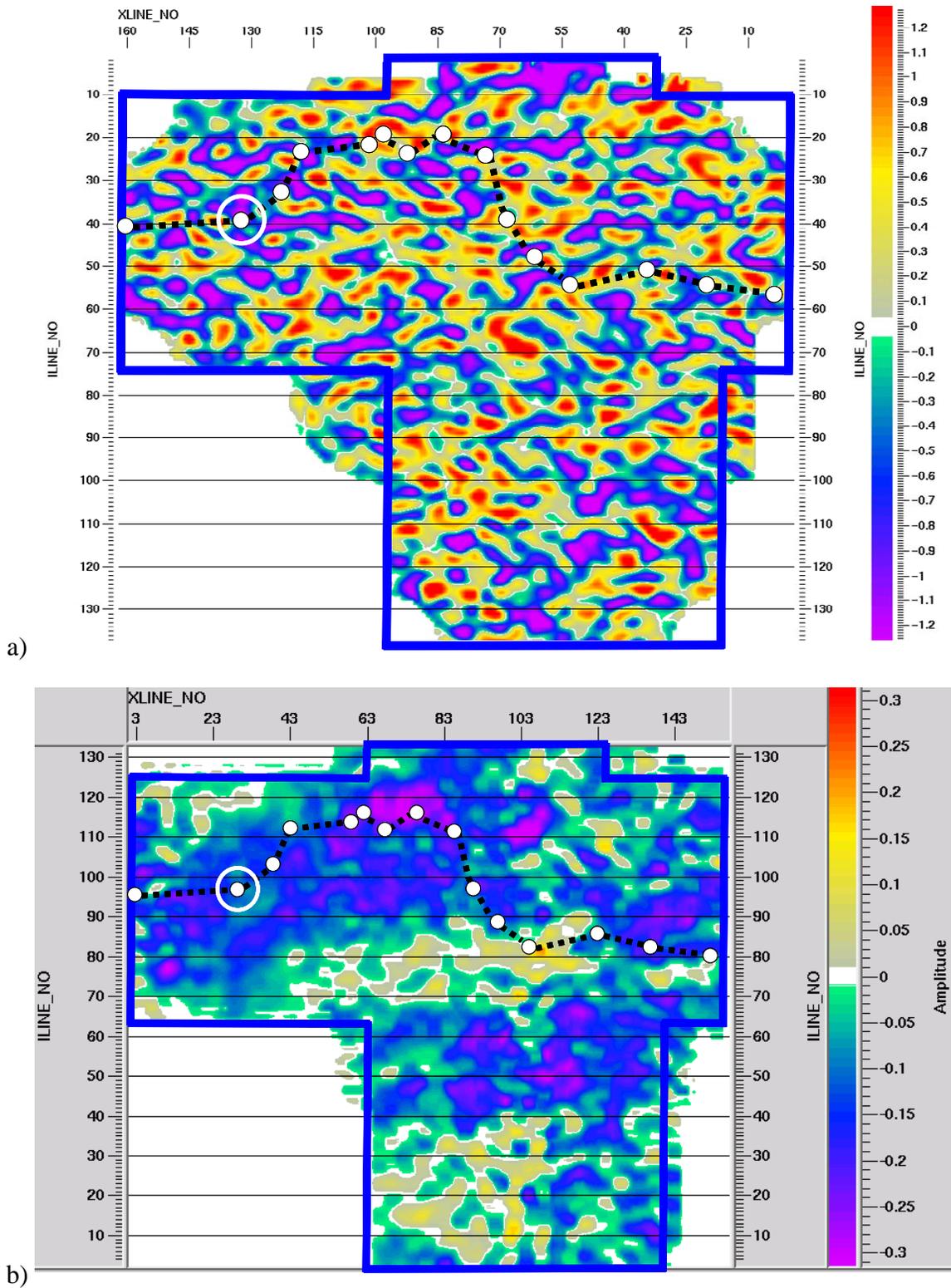


FIG. 4. Time-slices near the target zone through migrated P - S volumes. Dashed line and circles (white) show location of wells along a Colony member sand channel. a) Lu et al. (2006); geometry copied from vertical component shots (2.3% fewer traces). b) Lu et al. (2007); geometry loaded into ProMAX as described in this report.

DISCUSSION

A Perl script was developed to quickly and accurately load geometry information into ProMAX trace headers from a variety of sources. After binning, a visual quality control step was applied. Total development/quality control time was about one week. A seemingly trivial difference in the number of traces between two SEG-Y files (2.3%) had major consequences for image quality in the final migrated volume.

FUTURE WORK

The Perl script is fairly specific to files available for the Manitou Lake survey, but should be easily modifiable as needed for other surveys. Because of the relatively small size of the dataset, no attempt was made to optimize the Perl script for speed. For larger datasets, a re-write may be in order.

ACKNOWLEDGEMENTS

We would like to Calroc Energy Inc. for providing this dataset to CREWES, Landmark Graphics Corporation and Hampson-Russell (a CGG Veritas company) for the use of donated software (ProMAX and GLI3D respectively). We also would like to thank all CREWES sponsors and NSERC for funding.

REFERENCES

- Bland, H.C., 2002, The fusion of ProMAX and Perl: CREWES Research Report, **14**, 19 pages.
- Lu, H., Hall, K.W., Stewart, R.R., Feuchtwanger, D., and Szatkowski, B., 2006, Searching for sand reservoirs: Processing 3C-3D data from Manitou Lake, Saskatchewan: CREWES Research Report, **18**.
- Lu, H., Hall, K.W., Stewart, R.R., 2007, Re-processing the 3C-3D seismic data from Manitou Lake, Saskatchewan: CREWES Research Report, **19**.

APPENDIX B: PERLCODE.PL

```

$| = 1;

#Read observers notes
open(FFID, "<manitou_shotid.csv") or die "Error: $\n";
while(<FFID>) {
    /(\d+),(\d+)/;
    $ShotID[$1] = $2;
}
close(FFID);

#Open output file for debug information
open(OUT, ">debug.csv") or die "Error: $\n";
print OUT "FFID,ShotID,PromaxScript,ShotLogScript,dNtr,comment";

# Start of subroutine to deal with ProMAX ensemble trace headers
sub onEnsemble {
    $comment="";
    @FFID = gethdr('FFID'); #From in SEG-Y headers
    @SHOT_ID = gethdr('SHOT_ID'); #New header created with Trace Header Math (all zeros).
    print OUT "\n$FFID[0],$ShotID[$FFID[0]],"; #Print ProMAX FFID and Observer notes ShotID to debug.csv

    #Set ShotID header word.
    for ($i = 0; $i< $#SHOT_ID+1; $i++){
        $SHOT_ID[$i] = $ShotID[$FFID[0]];
    }
    puthdr('SHOT_ID',@SHOT_ID);

    #Open ShotLog file (see Appendix A) for reading
    open(IN, "<ShotLog_MANITOU-3D.txt") or die "Error: $\n";

    @SCRIPT = gethdr('SCRIPT'); #New header created with Trace Header Math (S_LINE*1000+SOU_SLOC)
    @R_LINE = gethdr('R_LINE'); #New header created with Trace Header Math (all zeros)
    @SRF_SLOC = gethdr('SRF_SLOC'); #New header created with Trace Header Math (all zeros)

    $flag=0;
    $success=0;

```

```
while (<IN>) {#loop over every line in ShotLog
  chomp();
  if (/Shot ID: (\d+)\s+Ep: (\w+)/) {#Find lines containing "Shot ID:"
    if ($ShotID[$FFID[0]] == $1) {
      if ($2 =~ /Stack/) {#and "Ep: Stack:"
        $flag = 1;
      } else {
        $flag = 0;
      }
    }
  }
  if (/Script: (\d+)/ && $flag == 1){#Find lines containing "Script:" and read the script number
    print OUT "$SCRIPT[0],$1,"; #Print ProMAX script and ShotLog script numbers to debug.csv
    if ($SCRIPT[0] == $1) {
      $flag = 1;
    } else {
      $flag = 1; #Potentially fail by setting flag = 0. Desirable?
      # Set $comment if the script numbers don't match and carry on
      $comment = $comment."Warning: Script numbers do not match! ";
    }
  }
  if (/Active Seis/ && $flag == 1){#Find lines containing "Active Seis"
    $flag = 2;
    $chan = 0;
    $r_line = 0;
    $srf_sloc = 0;
  }
  if (/Trace Count: (\d+)/ && $flag == 2) {#Find lines containing "Trace Count"
    $TraceCount = $1/3;
    print OUT "$crProMAX:NTRACES,"; #Print number of traces in the ProMAX ensemble to debug.csv
    print OUT "$TraceCount,"; #Print number of traces in the ShotLog trace count to debug.csv
    print OUT "$crProMAX:NTRACES-$TraceCount.,"; #Print the difference to debug.csv
    if ($TraceCount != $crProMAX:NTRACES) {#Set $comment of number of traces does not match
      $comment = $comment."Warning: Number of traces does not match! ";
      #exit; #Potentially fail by uncommenting this line. Desirable?
    }
  }
}
```

