Book K

Image to Data Trace Files

By
Robert W. Lankston
May 1, 2023

In spring of 2012, I developed a largely manual workflow for converting the images of the redisplayed seismic sections from the 1970 Flathead Lake seismic survey into seismic traces. Perhaps quasi-seismic traces would be a better term. The objective was to have the data in a trace form such that they could be interpreted in a seismic workstation. Of course, given that the redisplayed sections effectively contain sign bit information only, amplitude information in the recovered traces is largely suggestive of what might have actually been recorded. However, when displayed by a workstation or in a processing environment such as Colorado School of Mines' Seismic Unix (SU), the traces maintain the geologic character seen in the paper sections. The sections for Lines A, B, C, D, E, and F that are converted from the redisplayed sections exhibit an amazing similarity to the respective sections plotted from the data digitized from the analog tape that was discovered in the United States Geological Survey (USGS) library at Woods Hole, MA, in 2006.

In addition to structural and stratigraphic interpretation of the data in a workstation environment, the data in trace form might be suitable for processing to remove bubble pulse effects or multiples. Aspects of the image to seismic traces process are illustrated in the PowerPoint slides in Book L.

The process of converting the images to seismic traces is, conceptually, straightforward, but it has complicating nuances. The starting point is an assertion that a column of black and white pixels in the scanned image is a sign bit seismic trace. I have no information that either the field recorder or the recorder used in the post-acquisition processing generated a variable density display in which the shade of gray represented an amplitude range.

With the assertion that the image of the redisplayed seismic section is simply black dots on white paper, then, all that one has to do is convert the sequence of black and white pixels in a column extracted from the image to signed amplitudes, e.g., +1 for a black pixel and -1 for a white pixel. The series of 1's and -1's from a column of pixels becomes a sign bit seismic trace.

I chose a convention here, i.e., the first nuance. The image is not truly black dots on white paper. In fact, the 2006 scans of the redisplayed sections are coded in full RGB color. Therefore, the color images had to reduced to grayscale images. This is easily done in image processing software. The pixel color values in a grayscale image range from 0 to 255, i.e., black to white, respectively. I set a value of 160 to be the dividing point between black and white. Black and dark gray pixel values below 160 (out of 255) were set to +1 in the output seismic trace. White and light gray pixel values greater than 160 were set to -1. I adopted this convention assuming that the dots burned into the recording paper were positive amplitudes analogous to the shaded positive amplitude lobes in a variable area seismic display. This sign convention also maintains the black on white character of the original reprocessed sections.

Another nuance on the conceptually straightforward process is that the initial scans of the redisplayed sections had timing lines that were in the range of -1.5^{o} to +1.5^{o} off of horizontal, almost not worth worrying about for studying the images themselves but significant for the image to data conversion process. In the initial, manual processing, I used the open-source image editor named gimp to convert to grayscale, set the threshold between white and black, and to rotate the image to make the timing lines horizontal.

The next conceptual step is to generate traces that have a wiggly appearance. Many schemes can be developed to convert the +1's and -1's to a wiggly form. The simplest is probably to apply a common trapezoid-shaped band pass filter. A Ricker wavelet or a zero-phase wavelet could be also used for the filtering.

The series of +1's and -1's from each column in an image could be conditioned in a variety of ways, e.g., perhaps a 0 should be inserted instead of a -1 where the signal transitions from +1 down to -1 or from -1 up to 1. The logic for this is that the black dots in the image are showing only the positive values. Zero crossings are rendered white just like the negative portions of the waveforms. In another conditioning scheme, one might assert that the number of contiguous +1's is proportional to the peak amplitude of that lobe and replace the series of +1's with a weighted, single spike in the middle of the contiguous series. This could be applied to the -1's , also. Then, the spike train would be filtered as in generating a synthetic seismogram from a series of reflection coefficients.

Another nuance is that the dots for an actual seismic trace are not constrained to one column or a set of a few adjacent columns of pixels in the image. Because the recording paper moved continuously under the recording stylus, each trace is actually along a diagonal zone on the paper and, consequently, in the scan of the redisplayed section (Figure 1). I also show an example of this skew in the PowerPoint slides in Book L. As if the skew were not enough, the apparent diagonal path of the stylus across the paper appears to have a waviness in its path. This waviness is not systematic and is probably impossible to correct for. While I do not know the cause of the waviness superimposed on the generally linear skew, my guess is that it is the result of small distortions in the recording paper resulting from humidity fluctuations and handling through the years. Correcting for skew should be done before the sign values are assigned, but it was not done in the initial manual processing.

At some point, I realized that the manual operations were too slow and had too much opportunity for subjectivity from line to line. By the fall of 2012, I began using Python scripts for the various conditioning tasks, i.e., thresholding, image rotation for timing line flattening, and so forth. Robb Lankston, a computer science and mathematics student at the University of Montana at that time, designed the basic Python Imaging Library (PIL) scripts, which I augmented with system calls to SU modules. One script returned the best rotation angle to a precision of 0.1^{o}. While the precision may not have been better than the manual approach in gimp, the determination was quantifiable and definitely less tedious. This rotation analysis is illustrated in the Book L PowerPoint slides. The extraction of the 1's and -1's was in another script.

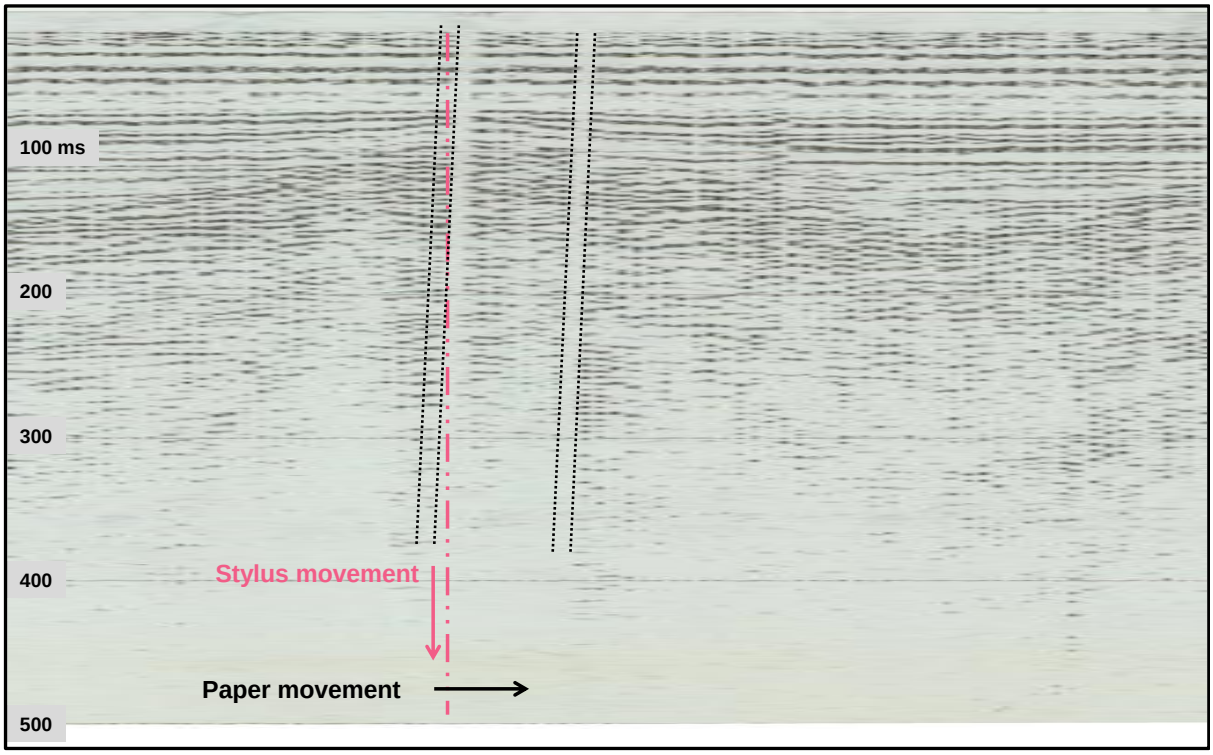

Figure 1. Example of Trace Skew in Redisplayed Section. The stylus is constrained to move in a track from the top of the section to the bottom (pink line). At the same time, the paper moves steadily as indicated. Therefore, the dots align along a diagonal pattern in the section. The amount of the skew in the original section is approximately 0.10 in from top to bottom, i.e., the skew is not noticeable. However, if the objective in the dots to data process is to recover black dots from just one trace at a time, the image must be shifted such that the parallel diagonal lines in this image would be vertical like the pink line indicating the stylus movement. This image shows a feature that I have not really considered, i.e., how to recognize missed stylus scans across the recording paper. Some missing scans can be related to missed traces during field recording. What those white bands in this example do show, though, is how wide one trace is in the redisplayed medium.

SU functions were called from another Python script to build SU format trace files, i.e., the a2b and suaddhead operations discussed in Book I, and to apply the band pass filter. The corner frequencies of the trapezoid operator were 30, 50, 250, 300 Hz. System calls from the Python script to SU modules were used to mix the filtered traces (laterally) in order to smooth over the white space between the columns of dots in the actual image. The white space is not the dead trace situation in Figure 1. In general, the white space between traces is just one or two pixels wide. The trace mixing was followed by decrementing the traces such that the spacing between traces was similar to what was actually recorded in the field (approximately 10 m). SU functionality was used again to resample the traces in time for 1 ms sampling intervals. The 300 dpi pixel spacing in the images yielded approximately 5.8 pixels per millisecond. The lateral mixing was an easy step to implement. However, like the skew issue, I realized early that a better scheme should be applied to defining each individual trace.

I used the open source geographical information system (GIS) Qgis to georegister the Silverman et al. (1971) (Prahl) map with a web basemap, and I recovered the UTM coordinates of the end points and turning points for all of the lines. I then interpolated the x-y coordinates for the individual traces from the coordinates of end and turning points. The trace headers were updated with the x-y coordinates, assuming zero offset between source and receiver as discussed in Book I, using another Python script with SU calls. This final script generated both SU and SEG-Y format files, and the SEG-Y files are in this book of the collection. These files should be easily imported to a seismic interpretation workstation. The coordinates are NAD83 UTM Zone 11N (EPSG:3741) in units of meters.

For Lines K, Q3, S, and T, I processed both versions of the redisplayed images. In the case of Line Q, I spliced together the three segments, Q1, Q2, and Q3 to form a single line as is drawn on the Silverman et al. (1970) track line map. Concatenating the Line Q1, Q2, and Q3 segments as suggested by the Prahl map is probably an oversimplification, and a user of the Line Q SEG-Y file might want to compare the respective scanned images to the corresponding segments of the Line Q SEG-Y data. Similarly, I combined the coordinates for Lines M and MW to form one Line M in the archive.

Finally, because of its length, the redisplayed section for Line I was scanned in 2006 into two files, i.e., I1 and I2. I processed these two files through the image to data conversion and then spliced the resulting SU files together using SU windowing and concatenation tools prior to conversion to SEG-Y format. Therefore, the archive has only one trace file for Line I. While the scans of the two Line I sections have a slight overlap, the SU windowing eliminated that overlap.

Line G presents an interesting case. In my early work, I casually assumed that a one-to-one correspondence existed between the field sections and the redisplayed sections. In initially recovering the line endpoints from the Prahl map, I set the Line G endpoints at the eastern- and western-most points of Line G as drawn on the map. In reviewing the Line G redisplayed section in 2023, I realized that the redisplay of Line G started on the eastern end of the line but ended on the west in the vicinity of Cedar Island. I have no idea as to why the redisplayed section does not cover the entire range that was recorded. I guessed at where the western end of the Line G redisplay was. That guess could be off by 100 m or more. However, my guess is a better location than what I initially used as the west end of the line. Therefore, after a 10-year hiatus, I went back into my GIS application, my data files, and my bash and Python scripts and recalculated the trace positions for Line G. The new Line G SEG-Y file is now in the archive. If you have used the Line G SEG-Y file in a workstation prior to May 2023, you might find that the new version complements your interpretations of faults on the east side of the lake and other features better.

What makes the Line G case more curious is that Wold (1982) showed only the western portion of the line, i.e., generally the segment from Cedar Island to the western shore of the lake. That is the segment of Line G that is not in the redisplayed seismic section in this archive. The image in the redisplayed Line G in this collection shows clear reflections and the deep sediment trough near the eastern shore of the lake. One might surmise that Wold had access only to that western portion of the line. Perhaps more to the point, Wold clearly had some information on the line locations and the directions the respective lines were shot.

In the 2012-13 era of the dot-to-data work, I modified the parameters in my scripts as necessary to convert a few of the scans of sections from the Kogan (1980) survey. Kogan mentioned that his data were recorded in variable density, but I used the simple threshold method described

above for the conversion. One line that I converted is displayed in three dimensions with lines from the 1970 survey in the PowerPoint slide show in Book L.

In Book A, I mentioned that I was a mentor to a team of computer science students who accepted my proposed project as their senior capstone focus. The first thing that I wanted the team to develop was a good skew removal method and then a technique to recover the 1's and -1's from individual traces, i.e., to have traces that were essentially one to one equivalents to the traces that would have been recorded on magnetic tape. The result of the technique that the students developed would be tested against signals from the USGS magnetic tape for lines such as Line E or Line F.

Unfortunately, the students made only nominal progress with my ideas. Their capstone course focused on how to manage an application development effort and included aspects of design, negotiations with prospective users, and various reporting. Actual code development was secondary.

As I launched into the 2023 maintenance of this archive, I decided to do one more proof-of-concept. My long-simmering idea for determining the proper skew parameters was first to define a window in the image. I used a window and not an entire scan of a redisplayed line assuming that the skew was constant across the entire image and to reduce computation time. Then, in that window of the image, I calculated the sum of the black dots in each column of pixels.

I kept a table of trace column versus sum for each skew value that was tested. Initially, my skew parameter was a simple lateral shift in units of pixels to shift laterally per image row down through the image (window). This worked, of course. However, Python has more sophisticated image shifting functions, and I soon adopted those. I looped through a range of skew parameters updating the tabulated values and plotting the sums versus trace position in the window. Figure 2 shows a snippet of a typical result.

With the optimum skew value defined, the next step was to apply that skew and capture the column locations for the respective peaks. The output sign bit traces, then, would be the +1 and -1 pixels in the columns defined by the respective peaks. I decided to capture a bin of pixel values to the left and right of the columns defined by the peaks. For a bin of three columns, for example, I would return a +1 if two of the three columns had black pixels. I would return a -1 if none of the columns had a black pixel or if only one black pixel was encountered.

While I did not test the following idea, it probably merits a try. As I have noted in other contexts, I have no information that the recorder used in the redisplay generated a variable density display. However, on the chance that the display is variable density, instead of using the scheme mentioned above for returning 1's and -1's, one could take the sum of those three pixels in the bin and assign an amplitude to that. Of course, that brings up the question of whether the four values, i.e., 0, 1, 2, or 3 in the bin example above, represent all positive amplitudes from the original signal or whether they represent the full negative and positive range. Obviously, one can ask a lot of questions and try many different algorithms.

With the 1's and -1's recovered from the explicit trace positions, I went forward with the rest of the processing as described above, i.e., convert from an array of pixel values to quasi-seismic traces in SU, add header values to the traces, filter the traces, and so forth. Every step after generating the sign bit traces from the redisplayed image requires setting parameters. For the most part, these are subjective settings.

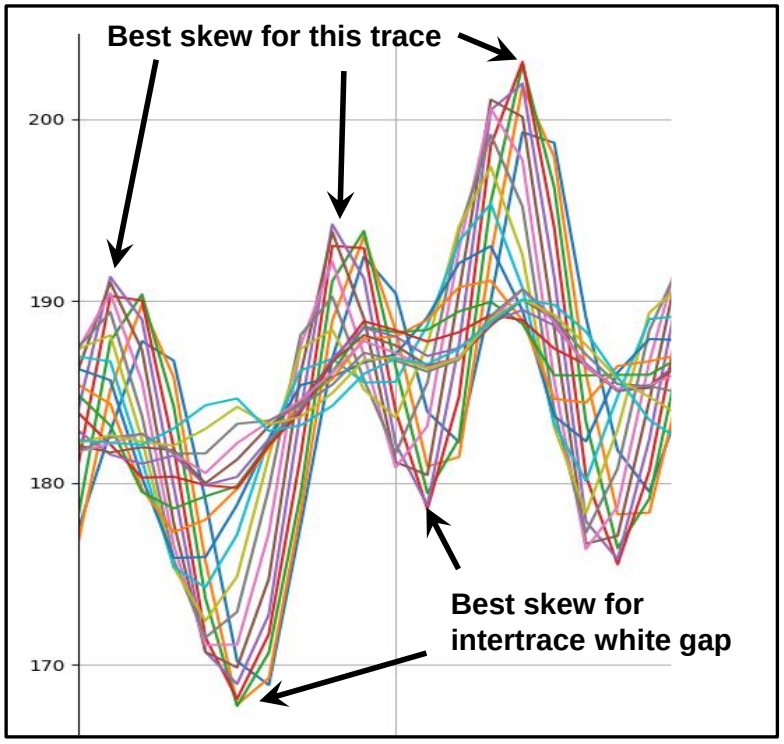

Figure 2. Skew Adjustment Evaluation. Each curve represents the sum of the black pixels down a column in the analysis window. The horizontal axis is trace position. The vertical axis is the sum. This segment shows just three trace positions from a window that had 200 or 300 trace positions. The peaks indicated a trace position while the troughs indicated the white space between actual traces. In this short segment, a red and a green curve seemed to always indicate the peak or the trough. Obviously, I had to make a call on which skew value to use for the next stage of processing. These calculations were based on a 300 dpi image. Had I used the original 600 dpi image, perhaps the curves would have suggested a skew value between the red and green curves here.

Filtering or convolving with a wavelet is a subjective step. One factor that affects the result of the filtering or convolution is the zero level of the input sign bit trace. I have not experimented with this. One way to change the zero level would be to change the +1's, which I have described, to +1.2, for example, and change the -1's to -0.8. This would say that the original signal had stronger peaks than it had deep troughs. One might test this idea first on the traces recovered from the USGS tape.

The characteristics of the filter will affect the result. Then, after the filtering, the display parameters will affect the appearance of the section. In Figure 3, I show the result of the image-to-data process applied to the western half of Line C. I used a trapezoidal filter with the high end rolling off from 200 to 400 Hz. Spectral analysis of the traces digitized from the USGS tape might suggest passing even higher frequencies. I adjusted amplitude clipping parameters such that the resulting view for Figure 3 has similar gray tones as the redisplayed section.

The result in Figure 3 is based on traces extracted from the redisplayed section using the skew removal that I describe above. Whether this result is significantly better than the result for the 2012-14 episode of generating traces, I cannot say. I do know, however, that each trace in the output is much more closely related to the original traces than those that were extracted after the lateral mixing of the original round of image-to-data processing.

As of this writing, I have not processed all of the lines through the workflow that included the skew adjustment. If I do that, I am inclined to post only the +1/-1 traces. Then, the user can experiment with all of the settings that go into making a final section.

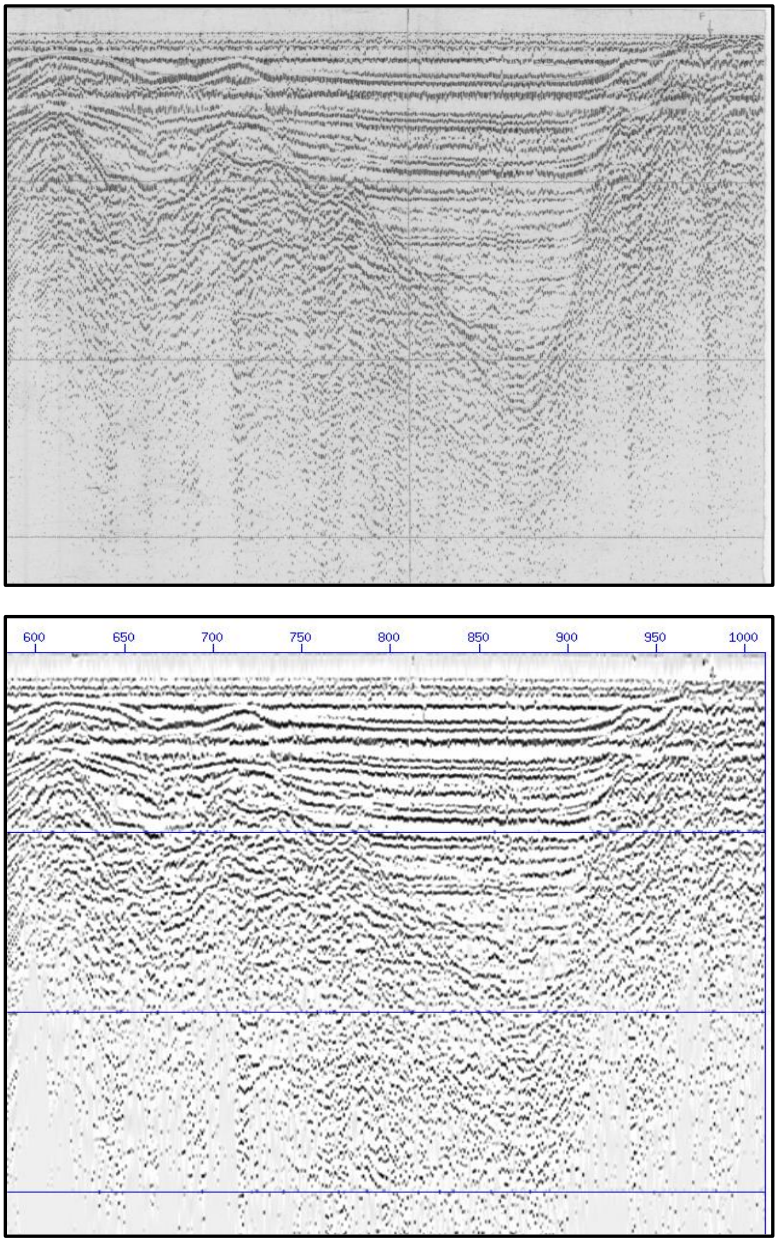

Figure 3. Comparison of Redisplayed and Image-to-Data Section for Part of Line C. Upper panel is the redisplayed section. More details are in the text. Timing lines are at 100 ms intervals. The west end of Line C is on the right in the images. The numbers along the top of the image-to-data section are trace number along the line. The image-to-data process yielded 1011 traces. 1020 traces were recovered from the USGS tape for Line C during the 2008 work.

References Cited

Kogan, J., 1980, A seismic sub-bottom profiling study of recent sedimentation in Flathead Lake, Montana: UM master's thesis. (URL: http://scholarworks.umt.edu/etd/7735/)

Silverman, A. J., Pevear, D. R., and Prahl, S. R., 1971, Bathymetry of Flathead Lake, Montana: unpublished. (URL: http://scholarworks.umt.edu/cgi/viewcontent.cgi?filename=2&article=1015&context=flathead&type=additional)

Wold, R. J., 1982, Reflection seismic study of Flathead Lake, Montana, USGS Miscellaneous Field Studies Map MF-1433: US Geological Survey. (URL: https://pubs.usgs.gov/mf/1433/plate-1.pdf)

Citing this Narrative

For "books", i.e., the ScholarWorks term for a folder or a directory, ScholarWorks will display a Recommended Citation for the entire book. Some of the books (directories) in this collection have what ScholarWorks calls "supplemental material" (files). ScholarWorks does not suggest a citation for the individual files, and you may have occasions when you want a reader to be able to find exactly what you are talking about in your own work.

Therefore, I am suggesting a citation for this narrative. My suggested citation is in a little different form than the ScholarWorks form, but that is not critical. Every medium has its preferred format, and the format of my own citations in these narratives varies. The important components of a digital citation are the author(s), the title of the work, the year of creation, the name of the collection in which the work appears, and the URL of the work. The following meets those specifications.

Lankston, R. W., 2023, "K. Image to Data Trace Files– narrative by Robert W. Lankston" in *1970 Flathead Lake Seismic Survey*. URL: https://scholarworks.umt.edu/cgi/viewcontent.cgi?filename=0&article=1009&context=flathead&type=additional

You can find the URL for my narrative by moving your mouse to the Download button next to the narrative title in the list of files for the book. Right click the mouse and, in the pop-up options box, choose to copy the link address. Then, the address is in your clipboard, and you can drop the text string of the address into your work.

I would suggest the same citation format and the same technique for capturing the URL of the supplemental files that form the heart of the content of this book.

Feedback

The standard ScholarWorks format limits how/when an author's email address is displayed. With the author convention that I use, my email address is not displayed for books such as Book D. Actually, my email address is not even in the book metadata. If you need to contact me regarding the content, a broken URL, and so forth, you can use:

rwlankston@gmail.com

Last Update: 6/2/2023 11:28 AM