Geol 483.3

Lab project #4

Analysis of a 2D refraction dataset from 2021 Geophysics field school

In this lab, you will analyze first-arrival travel times from one of the lines recorded during the Geophysics field school in 2021. This was a small 3-D survey (about 100 m long) along the median of University Drive in Saskatoon using a fixed geophone spread using the 96-channel system we built three years ago. Coordinates were measured at a grid of survey points by GPS and interpolated to determine the locations of each geophone and source point.

I performed the initial editing of the dataset. The complete dataset (also including two other lines) can be found in directory `/data/morozov/Riverbank_2021_Refr/` on Linux computer named `sura (sura.usask.ca)`. This computer is located on the right side of Geology room 135, and it is also available remotely. For this lab, you will need to use the SEGY (seismic data) file `data_line02.sgy` from this directory. A copy of this file is here and some Matlab/Octave codes are included in this archive file. I will update this file as we progress with the lab project.

Refraction data analysis will consist of several steps:

1) Loading the SEGY file into a commercial software package called TomoPlus by GeoTomo. This software is designed for analysis and inversion of near-surface seismic records. In this software, you will display the seismic line, examine its parameters, evaluate data quality, and try various types of filtering and display.
2) Picking first arrivals in TomoPlus and exporting them into ASCII tables for Matlab.
3) Plotting the travel times in several forms in Matlab;
4) Inverting the first-arrival travel times using Matlab programs which we will develop during this lab. Methods of this inversion are outlined in section "Methods" below.

Because of limited time remaining in this term, you may not achieve the complete inversion but you will still obtain intermediate results which will be useful for deriving a model of the shallow subsurface beneath the University Drive.

Prior to starting this lab, you will need to set up your Linux account as on any of the Linux computers in room 135. The procedure is explained under link "Linux setup for labs" on the web site. You will only need to do this once. After this setup is complete, you should be able to login onto any of our machines and use it to run GeoTomo and other programs. Computer `sura.usask.ca` can be used remotely from on and off campus after performing the appropriate VPN setup.

During and after this setup, you will need to learn about basic Unix shell commands:

- `pwd` (print work directory),
- `cd` (change directory),
- `ls` (list file names),
- `mkdir` (create directory),
- `cp` (copy files), `mv` (move or rename files), `rm` (delete files)

- `man` (view manual about any command, with many options),
- `scp` (secure copy of files or directories from any machine over the network)
- `more` or less (display contents of text files),

and other.

Methods

Inversion of first-arrival travel times consist in finding a model of the subsurface which would predict travel times for head waves close to those picked from the dataset. Below, I define the various components of this inversion.

Model

We will use a layered model of the subsurface in a delay time (sometimes also called "time term") form. In this form, head wave delay times are used instead of the depths of the refracting boundaries. Between the boundaries, model velocities are constant vertically and smoothly variable horizontally. Thus, the *n*-th boundary ($n = 1..N_b$) is described by two smooth functions of the horizontal coordinate *x*:

$$\text{delay time } \delta t_n(x) \text{ and slowness below the refractor } p_n(x). \qquad (1)$$

The refractor depths are related to delay times as

$$z_n = z_{n-1} + \frac{V_{n-1}}{\cos\theta_n}\left(\delta t_n - \sum_{k=1}^{n-1}\frac{\sqrt{1-(p_nV_k)^2}}{V_k}\right), \qquad (2)$$

where $V_n(x) = 1/p_n(x)$ is the velocity *above* the refractor, and $\theta_n = \arcsin(p_nV_{n-1})$ is the critical angle for the refractor (this formula is simply an inverse of the expression for the delay time for a stack of layers we saw in class). These depths should be measured relative to some smooth datum surface. As the datum, we will select a smooth line below the minimum surface elevation.

The uppermost near-surface layer ($n = 0$) also contains model parameters given in eq. (1). For this layer, function $\delta t_0(x)$ has the meaning of delay time within a very thin near-surface layer, and $p_0(x)$ give the variations of the direct-wave velocity.

For numerical inversion, the continuous functions and $\delta t_n(x)$ and $p_n(x)$ need to be discretized. They can be discretized by selecting several control points (these points may be different for the depth-related functions $\delta t_n(x)$ and for velocity-related functions. Between these points, the values of functions will be determined by interpolation. Interpolation of discrete points can be represented by summations:

$$\delta t_n(x) = \sum_{i=1}^{N_{\delta t}}\varphi_i(x)\delta t_{ni}, \quad p_n(x) = \sum_{i=1}^{N_p}\varphi_i(x)p_{ni}, \qquad (3)$$

where δt_{ni} and p_{ni} are the model parameters at the discrete points, and $\varphi_i(x)$ is a "sawtooth"-shape basis function centered on the *i*th control point x_i. The numbers of control points $N_{\delta t}$ and N_p can be different. Usually, N_p (number of points at which the layer velocities are defined) is small (2 to 5), although in the midpoint method described below, N_p can be large. The number $N_{\delta t}$ controls the detail of depth variation of the layers, and this number would usually be larger than N_p.

Thus, matrices δt_{ni} and p_{ni} contain all parameters of the model we will need to invert for. If these parameters are known, the model can be plotted and all travel times can be predicted.

Travel time prediction

The complete predicted travel-time model that we will use for matching the observed travel times is

$$t_n^{\text{pred}}(S,R) = t_S^{\text{rec}} + t_{Sn}^{\text{elev}} + t_{Rn}^{\text{elev}} + t_0(x_S) + t_0(x_R) + t_n^{\text{model}}(S,R) + t_S + t_R, \tag{4}$$

where S denotes the source, R denotes the receiver, and n is branch of the wave (direct or refracted on the nth boundary). In eq. (4):

1) t_S^{rec} is the reciprocal-time correction applied to each source (explained in the next subsection);
2) Elevation-related terms t_{Sn}^{elev} and t_{Rn}^{elev} representing additional delay times of the source and receiver locations due to their elevations relative to the datum: $t_n^{\text{elev}} = \frac{\sqrt{1-(p_n V_0)^2}}{V_0}(z - z_{\text{datum}})$.
3) Zero-offset time terms $t_0(x_S)$ and $t_0(x_R)$ due to a possible very low velocity, thin near-surface layers. These terms are "surface consistent", which means that they relate to the surface locations x only and are equal for source and receiver located at the same point.
4) Term $t_n^{\text{model}}(S,R)$ is the travel time predicted by a layered subsurface model, <u>with the source and receiver located on the datum</u>;
5) The last terms t_S and t_R account for small source- and receiver related travel-time variations which are not accounted for by the 2-D model. These terms are also described in the next subsections.

For a given layer number $n \geq 0$, the travel time $t_n^{\text{model}}(S,R)$ from a source S to receiver R located <u>on the datum surface</u> is predicted by the delay-time relation:

$$t_n^{\text{model}}(S,R) = \delta t_n(x_S) + \delta t_n(x_R) + \int_{x_S}^{x_R} p_n(x)\,dx. \tag{4}$$

With $n = 0$, this equation gives the direct-wave travel times and with $n > 0$ – head wave travel times. Using eq. (3), the integral in this expression is transformed into a sum:

$$t_n(S,R) = \sum_{i=1}^{N_{\delta t}} g_i(S,R)\,\delta t_{ni} + \sum_{i=1}^{N_p} p_{ni} f_i(S,R), \tag{5}$$

where $g_i(S,R) = \varphi_i(x_S) + \varphi_i(x_R)$ and $f_i(S,R) = \int_{x_S}^{x_R} \varphi_i(x)\,dx$ is an integral of the ith basis function along the source-receiver path. These integrals are easily calculated analytically using the known picewise-linear functions $\varphi_i(x)$, and so eq. (5) represents a matrix product and summation which can be easily evaluated in Matlab.

Inversion

Inversion of the observed travel times consists in finding the subsurface model and the additional terms in eq. (4) so that $t_n^{\mathrm{pred}}(S,R) \approx t^{\mathrm{obs}}(S,R)$ in the least-squares sense. This inversion can be done in the order of terms shown in eq. (4), as described below.

Correction of source times

First, you will **invert the mismatches of all reciprocal times for the source time errors**. For each source S_1, consider all other sources S_2 such that each of them has the travel times picked in the vicinity of the other source. Any velocity structure has the reciprocity property $t^{\mathrm{model}}(S_1,S_2) = t^{\mathrm{model}}(S_2,S_1)$, and therefore the difference of these reciprocal travel times equals

$$t_{S_1}^{\mathrm{rec}} - t_{S_2}^{\mathrm{rec}} = \Delta t^{\mathrm{reciprocal}}(S_1,S_2), \qquad (6)$$

where $\Delta t^{\mathrm{reciprocal}}(S_1,S_2) = t^{\mathrm{obs}}(S_1,S_2) - t^{\mathrm{obs}}(S_2,S_1)$ is the difference between the observed travel times for the two shots. This is a linear inverse problem for t_S^{rec}, which can be solved by the least-squares method.

When inverting eq. (6), you will notice that the inverse is nonunique because this equation allows adding an arbitrary constant to all t_S^{rec}. This problem is easily corrected by adding an additional constraint to the system of equations (6). The constraint can be setting $t_S^{\mathrm{rec}} = 0$ for one shot or requiring that the average of all t_S^{rec} equals zero: $\sum_S t_S^{\mathrm{rec}} = 0$.

Ideally, $\Delta t^{\mathrm{reciprocal}}(S_1,S_2)$ should equal zero and therefore all $t_S^{\mathrm{rec}} = 0$. However, as $\Delta t^{\mathrm{reciprocal}}(S_1,S_2) \neq 0$ in the real data, inversion of eq. (6) gives the source times t_S^{rec} correcting for this error. These terms should then be subtracted from the data:

$$t_{\substack{\mathrm{corrected}\\ \mathrm{reciprocity}}}^{\mathrm{obs}}(S,R) = t^{\mathrm{obs}}(S,R) - t_S^{\mathrm{rec}}, \qquad (7)$$

giving corrected input data for further inversion, which are free of reciprocal travel-time mismatches.

Inversion for subsurface model

The next two steps of inversion consist in obtaining the subsurface model (parameters $t_0(x)$ and δt_{ni} and p_{ni} in the preceding section). For this, it is useful to view the first-arrival travel times as samples of a continuous "time field" (TTF) function $T(x_S, x_R)$ of continuously variable source and receiver coordinates x_S and x_R. The corrected observed picks (eq. (7)) represent sampling of this function at the available source and receiver pairs. For the subsequent plotting, data analysis and inversion, it is convenient to grid this function on a regular grid of midpoint coordinates $x_{mp} = \frac{x_S + x_R}{2}$ and (signed) source-receiver distances $d = x_R - x_S$. This gridding can be easily performed by using the Delaunay triangulation using function `griddata` in Matlab, and the result will be a matrix $T\left(d, x_{mp}\right)$ with columns representing the common-midpoint travel times and rows representing the common-offset travel times.

To obtain a subsurface model, you will need to **set the number of layers and obtain $t_0(x)$ and slownesses p_{ni}** first. Consider the common-midpoint travel time profiles $T\left(d, x_{mp}\right)$ at each of the selected control points $x_{mp} = x_i$ *i*. By plotting this *T* as a function of offset *d*, a first-arrival travel-time dependence can be recognized and interpreted. By identifying several linear segments and crossover points, crossover points, the number of layers can be determined at the given location x_i. Slopes of these segments give the slownesses within these layers (p_{ni} in eq. (5)), and the intercept of the first segment (at *d* = 0) gives the time $t_0\left(x_{mp}\right)$ in eq. (4).

Once parameters $t_0\left(x_{mp}\right)$ and p_{ni} are estimated, the elevation-related terms t_{Sn}^{elev} and t_{Rn}^{elev} in eq. (4) can be calculated and subtracted from the data. This subtraction should reduce the scatter of the travel times due to elevation variations and improve the identification of slownesses p_{ni} Therefore, the evaluation of the elevation of t_{Sn}^{elev} and t_{Rn}^{elev} and estimation of p_{ni} should be iterated a couple times until these values become consistent.

During the estimation of the number of layers (refracting boundaries) N_b and slownesses p_{ni}, the crossover distances for the refractions will also be determined. These distances can be used for partitioning of travel-time dataset into segments corresponding to direct waves (value of $n = 0$) and head waves from different boundaries ($n \geq 1$).

Using the identified layer slownesses, the delay times for the refracting boundaries can be obtained by correcting the observed times for all of the above effects and solving the linear inverse problem in eq. (5):

$$t_{\text{corr}}^{\text{obs}}\left(S,R\right) \approx \sum_{i=1}^{N_{\delta t}} g_i\left(S,R\right)\delta t_{ni} , \tag{8}$$

where the corrected data are

$$\underset{n}{t_{\text{corr}}^{\text{obs}}}\left(S,R\right) = t^{\text{obs}}\left(S,R\right) - \left[t_S^{\text{rec}} + t_{Sn}^{\text{elev}} + t_{Rn}^{\text{elev}} + t_0\left(x_S\right) + t_0\left(x_R\right) + \sum_{i=1}^{N} p_{ni} f_i\left(S,R\right)\right]. \tag{9}$$

Equation (8) is also an overdetermined linear inverse problem for unknowns δt_{ni}, which is readily amenable to the least-squares inverse.

Residual travel-time terms

After all "surface-consistent" model-related terms in the right-hand side of eq. (4) are inverted for, the "residual" terms t_S and t_R terms can be obtained. These terms are also obtained from a linear inverse problem:

$$t_{\text{error}}\left(S,R\right) \approx t_S + t_R , \tag{10}$$

where $t_{\text{error}} = t_{\text{corr}}^{\text{obs}} - t_{\text{model}}^{\text{obs}}$ is the total error of the travel-time prediction by the final model. Equation (10) can also be solved by the least-squares inversion. However, as its forward model (right-hand side) is very simple, it can be easily solved even in a better approximation. Let us use the median (statistical) inverse:

$$t_S = median\left[t_{\text{error}}\left(S,R \right)\Big|_S \right], \text{ and } \quad t_R = median\left[t_{\text{error}}\left(S,R \right)\Big|_R \right], \qquad (11)$$

where the notation $median\left[t\big|_{S \text{ or } R} \right]$ means evaluation of the median of all values *t* over all travel-time picks for the given source *S*, or for the given receiver *R*.

Assignments

1) **Create a work directory** under path /data/ on `sura`. Use 'cd /data/' and then `mkdir` followed by your username. Then '`cd`' to that directory. In the following, <u>place all files and work only in this directory</u>. You can create any subdirectories or files in there. In particular, when you start using GeoTomo, place its project into this directory.

2) **Start GeoTomo (TomoPlus) programs** by typing `geotomo` in a Unix shell. In GeoTomo, create a project and load file `data_line02.sgy` in it.

   Familiarize yourself with SEGY headers in the file, display geometry of the data.

   Create seismic displays and evaluate the quality of data records. There are a number of poorly recorded records, and channels 1 and 93-96 were disconnected. This data quality is normal. You need to ignore the poor records or exclude them by marking as bad or 'killed' records.

   The different shots in the records will be identified by different "field file identifiers" (FFID), which are called (I think) Shot ID in GeoTomo. The individual trace records are identified by their channel numbers used in the recording instrument.

   Identify the first arrivals in the records.

3) Select display form (time range, frequency filtering, AGC, wiggle-trace or variable-intensity color style) and **perform picking of the first-arrival travel times**. This will have to be done manually, but you can also explore the available automatic picking options.

   Try picking the different FFIDs consistently, i.e, pick the zero crossings from the first negative trace swing to the large positive amplitude peak. This may not always be possible to do, so only pick the records where such identification can be made. In GeoTomo, there should be options for automatic snapping of the manual pick to the nearest zero crossing, and also for picking groups of adjacent traces.

   To help ensuring picking consistency for different FFIDs, try switching between the shot display (usually the default) and common-midpoint and common-receiver display modes. In these displays, traces from different FFIDs are shown side by side, and it is therefore easy to see whether the same wave is being picked on them.

4) After the picks (or maybe a representative sample) are completed, **display them in GeoTomo** and evaluate the travel-time patterns. **Identify the direct waves, refractions** (head waves), and roughly estimate the cross-over distances.

5) **Export the first-arrival picks into an ASCII format** and prepare them for further processing in Matlab or GNU Octave.

   The preparation may require commenting out (by using symbols '`%`') certain header lines so that the resulting files can be loaded using function `load()` in Matlab. Alternatively, the pick files can be edited to include names of variables and represent parts of a Matlab program.

The following tasks are performed in Matlab using codes provided in this archive file. These programs are work in progress, and so please exercise patience and try understanding what is being done.

The different steps of processing are split into two scripts:

- `lab4_pass1.m` performing loading the data from GeoTomo file, creating geometry, correcting a couple errors found in the data, and performing the reciprocal time analysis and inversion;
- `lab4_pass2.m` performing semi-interactive analysis of the travel time and obtaining estimates of the near-surface velocities and delay times (at present, velocities in the deeper layers are unavailable.)

Execute `lab4_pass1.m` , look and generally understand the resulting plots (not very well labeled, sorry), then execute `lab4_pass2.m` . This script will produce Figures 5 and 6 with travel times from selected sources and midpoints. Looking at these plots, you will need to adjust the values of columns in matrix `model.crossover` to represent the offsets at which you estimate the crossover distances at the various locations within the profile.

Then, repeat the travel-time plots at the end of `lab4_pass2.m` with different selections of midpoints and make more adjustments in `model.crossover`.

After the picking of crossover distances is finished, re-run `lab4_pass1.m` and `lab4_pass2.m` for the cross-over picks to take effect. In the following steps, use plots produced by these scripts.

6) **Plot the first-arrival times from all shots in Matlab**, in the form similar to the T-X displays in GeoTomo. This is done by Figure 4 produced by `plot_ttimes()` in `lab4_pass1.m.` You can also call this function directly.

7) In Matlab, **extract all reciprocal-time mismatches and evaluate the source time corrections** as described in and after eq. (6). This result is shown in Figure 7 from `lab4_pass1.m.`

   Check whether these corrections are anomalously high for some shots, and whether these shots may need to be repicked.

   Apply these corrections to the picked travel times by using eq. (7). This result is shown in Figure 7 produced by `lab4_pass1.m.`

8) T**ransform the reciprocity-corrected picked times into a continuous gridded travel-time field** (TTF) in the plane of (*midpoint_coordinate*, *source_receiver_distance*). This can be done separately, by calling function `ttf()`. Use `help ttf` to see the description of this function. This function will take the desired offset increment and maximum offset and return three matrices: the TTF, its mapping into the direct-wave and head-wave branches, and the grid of offsets values used.

   Using the se outputs of `ttf()`, **display the gridded travel-time field surface** in one or several ways: in color using `imagesc()`, or contour it using `contour()`, or display in 3D using `plot3()`.

To clearer see the different branches of the observed TTF, it should be convenient to calculate and plot the "reduced" TTF by using relation:

$$T_r^{\text{obs}}\left(d, x_{mp}\right) = T^{\text{obs}}\left(d, x_{mp}\right) - p_r d \,, \tag{12}$$

where p_r is the reduction slowness (and $V_r = 1/p_r$ is called the reduction velocity). **Try several values of p_r.** If p_r is selected close to the value of p for some boundary, the refraction from this boundary would appear as near-horizontal on the graph of $T_r^{\text{obs}}\left(d\right)$, and it would look as a zone of a constant color contrasting with other areas in a color plot of $T_r^{\text{obs}}\left(d\right)$.

Inspect the results. Try identifying the zero-offset times t_0 (at the axis d = 0 of the plots), crossover distances, and the direct-wave and (maybe several) head-wave branches of the travel-time field. These branches will be seen as areas of areas of near-constant values of $T_r^{\text{obs}}\left(d, x_{mp}\right)$ in the reduced-TTF plots, when p_r is selected close to the actual moveout of the refractor.

9) <u>**We will likely stop at this point in this term!**</u>

   Plot the intercept values and report the slownesses of the uppermost layer. This is done in Figure 2 (produced by function `plot_model()` in both scripts).

   Compare the delay times to the surface elevation (in the same Figure).

10) **Define several control points (midpoints) for the evaluation of layer slownesses** and extract from $T\left(d, x_{mp}\right)$ offset-dependent, common-midpoint travel time curves.

    Plot the common-midpoint travel-time curves. Identify the travel-time branches, **measure t_0** (intercept of the direct-wave branch) and **intercepts and slownesses p_{ni}** for direct wave and head wave branches. From these intercepts and slownesses, **calculate the cross-over distances** for each of the head wave branches.

11) Define another (or use the same) grid of control points for delay times δt_{ni}. Using the inverted slownesses p_{ni} at the slowness control points, **solve eqs. (8) for delay times at the control points**.

    Make sure to apply the zero-offset and elevation corrections, and iterations described in "Methods".

    Transform the inverted δt_{ni} into boundary depths using eq. (2).

12) **Plot and discuss the model**.

Hand in:

Codes, plots, and report in electronic format.