REFRACT3D – a computer program for 3D refraction data processing

Jan Valenta

23rd January 2008

This manual describes the REFRACT3D program, version 0.2.

The REFRACT3D is a computer program, developed for processing of three dimensional refraction data sets. The program is written in Pascal programming language and the GUI (graphical user interface) was developed using the Lazarus[1].

The current version of REFRACT3D is a development version. Currently only two layer media is supported (layer over the half-space). The system of equations is either solved directly by the program or send to Octave script[2] for solving (the latter option is available only for UNIX-like systems and currently is tested on Linux). When utilising the Octave script option, it is necessary to use Octave version 2.9.6 or newer. The Octave script is used because Octave handle large matrices more efficiently than the REFRACT3D built-in routines. Microsoft Windows are not capable of running Octave scripts directly so the only option here is to use the built-in routines.

The program is used for computing layered models for a 3D refraction data sets. However, currently is limited to a model of a layer over the half-space. It uses modified time-term method, described in Valenta and Dohnal (2007) and in more details in the author's PhD thesis (Valenta, 2007). Seismic parameters of the subsurface (normal thicknesses of layers and layer velocities) are computed in a regular rectangular grid with arbitrary grid spacing. The inversion process is iterative with small number of iterations. Input and output files are column ASCII files separated by tabs or spaces. Input is a file with x and y coordinates of sources and receivers and with first arrival data classified according to the boundary along which the head wave is travelling. Output is a file with x and y coordinates of individual grid points with velocities for all layers in the dataset, standard deviations of these velocities, thicknesses of all layers and their standard deviations.

1 REFRACT3D – User's manual

To start REFRACT3D type the `refract3d` on the command-line or start it from some graphical commander. In case you want to utilise the Octave script, move to the directory where the REFRACT3D is installed. Make sure that also the Octave script `ComputeSVD.m` is present. Check, if the path to the Octave executable on the first line of the `ComputeSVD.m` script is correct. Otherwise, you will have to modify it (Fig. 1).

The REFRACT3D has a simple graphical user interface (GUI). When it is started, the initial window looks similarly to that one in Figure 2. The input file is loaded through the File → Open menu. If the file is read successfully, you will receive a confirmation message on the program's status bar.

[1] Lazarus is an open-source development environment designed to be similar to the Delphi from Borland. It is available for various operating systems and can be downloaded from the Lazarus WWW pages at `http://lazarus.freepascal.org`

[2] Octave is a Matlab-like environment with many ready to use functions for numerical computations. It is available for wide range of operating systems on the `http://www.octave.org`

```
#! /usr/bin/octave -qf
##
## The first line of this file should point to the Octave executable.
## If you have Octave binary in another directory than /usr/bin,
## modify, please, the path. If the "/usr/bin/octave" does not work,
## you may want to try "/usr/local/bin/octave".
##
##
... cut to save space
```

Figure 1: Beginning of the `ComputeSVD.m` script. The first line in this script must point to the Octave executable.

The structure of the input file is strict and does not allow any comments. Numbers are delimited by tabs or spaces. The file consists of blocks, each related to particular source (see Fig. 3). Each block begins with a line related to the source. It contains x and y coordinates of the source (floating point numbers), followed by the number of first arrivals for current source (integer) and a dummy number 0 to keep the four-columns structure. The source line is followed by lines with first arrivals. They consists of x and y coordinate of the receiver (floating point numbers), first arrival time (floating point number) and number of layer on top of which the head wave is travelling (integer). The uppermost layer has number 1 and it means that the wave travelling through this layer is a direct wave. The velocity of the uppermost layer is computed from this travel time. Seismic wave refracted on the first interface has a layer number 2 and so on. This block is repeated for every source in the dataset.

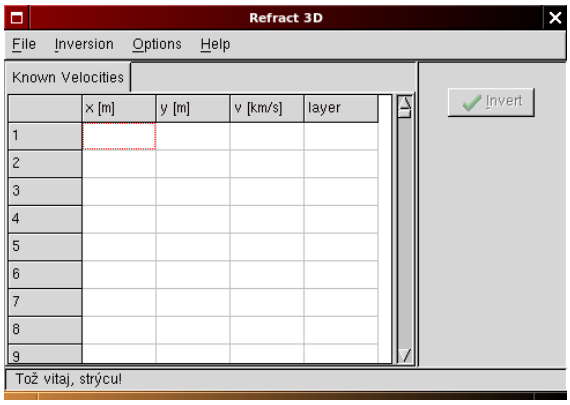

Figure 2: Initial window of REFRACT3D.

The layer velocities are considered to be constant in a vertical direction in every individual layer. This is implied by the simple layered model used. In a real world, the head waves (considered here) are rather refracted waves in the gradient media. In this sense the velocities computed should be called boundary velocities.

The data in the output file (see Fig. 4) are written to columns. Number of columns depends on the number of layers in processed dataset. The first line in the file is header, labelling individual columns. The first two columns are x and y coordinates of the current grid point (they are labelled `x` and `y` in the header). The third column contains resolved velocities of a direct wave labelled `v0`. Number of following columns depends on the number of layers included. Velocities of individual layers are labelled `v1`, `v2`, etc. Corresponding standard deviations are labelled `std_v1`, `std_v2` and so on. *Please note, that the standard deviations are deviations of slownesses (velocity reciprocals).* Thicknesses of individual layers are marked `d0`, `d1`, `...`, where `d0` is the normal thickness of the uppermost layer. Standard deviations of thicknesses are marked as `std_d0` and so on until the half-space is reached.

The "known velocities" table in the main window of the program can be used for supplying already known velocities from another type of survey. These velocities then enter the inversion process. However, they are not kept fixed throughout the inversion process. This is an intention, because velocities resolved by different methods are not equal. If the user really wants to keep the velocities fixed, then he must enter them by Octave commands in the `ComputeSVD.m` script. Then he would probably also want to change uncertainties for these values in the covariance matrix. On one hand this is very inconvenient for the user, because dealing directly with the source code may not be trivial. On the other hand it gives more flexibility in specifying probabilities of user supplied values.

Some useful inversion parameters can be specified using the Options → Configure menu (Fig. 5). The "Smooth Velocities" checkbox can be used, if it is desired to smooth velocities

```
11 56 12 0                  source line – x-coord, y-coord, number of first arrivals, dummy
19 76 18.062195 2           receiver line – x-coord, y-coord, first arrival time, layer number
19 72 17.973307 2           receiver line
19 68 16.782198 2           receiver line
19 64 16.693308 2           receiver line
19 60 15.377755 2           receiver line
19 56 15.431088 2           receiver line
15 56 11.893315 1           receiver line
15 60 13.066648 2           receiver line
15 64 14.435534 2           receiver line
15 68 14.524423 2           receiver line
15 72 15.004422 2           receiver line
15 76 15.57331 2            receiver line
11 60 24 0                  source line
27 76 20.266636 2           receiver line
27 72 20.319969 2           receiver line
... cut to save space
```

Figure 3: Example of input file for REFRACT3D.

```
x  y  v0    v1    std_v1 d0    std_d0
11 55 0.336 2.358 0.061  2.017 0.135
11 59 0.266 2.582 0.032  2.321 0.080
11 63 0.265 4.300 0.025  2.375 0.075
11 67 0.316 4.336 0.025  2.570 0.089
11 71 0.340 3.044 0.028  2.391 0.096
... cut to save space
```

Figure 4: Example of output file for the two layer media from REFRACT3D.

of the direct wave (velocities in the uppermost layer). Smoothing is done by averaging in moving window, the width of the averaging window is 3 samples. It is possible to specify number of smoothing passes.

The "Grid Spacing" field defines the size of the inversion cells in metres. The cells may have an arbitrary size. The cell size is dependent on the source and geophone separation and on the ray coverage. You can also determine the optimal cell size for each particular dataset by trial and error method. If the cell size is too large, then the resulting depths and velocities lack details. Too small cell size will produce ill determined system of equations and no reasonable results should be expected. The rule of thumb is that the cell size should be comparable with the source and receiver step of the survey. Finally, the "Use Octave Script" checkbox selects between the built-in routines and the Octave script for solving the equations.

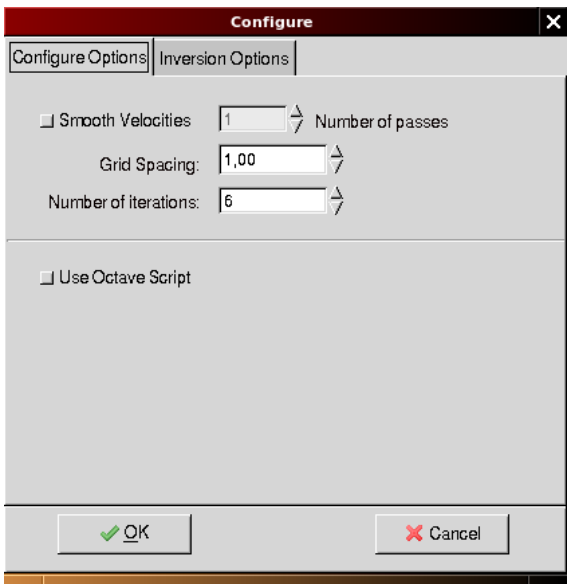

Figure 5: Configuration window of REFRACT3D.

When all of these parameters are set correctly, it is possible to run the inversion by clicking on the "Invert" button. The program sorts all the travel times

and finds which sources are identical with
receivers. This is a valuable piece of information for the inversion process reducing ambiguities in the receiver and source depths. According to the equation of the head wave a change in the depth beneath the receiver h_r can be compensated with a change in the depth beneath the source h_s. This affects all source-receiver pairs in the data set. However, when the depth beneath the particular receiver is set to be equal to the depth beneath the particular source, this ambiguity is resolved. It is a good idea to plan the field survey layout with respect to this limitation.

When the iteration process is finished, you can save resolved depths, velocities and appropriate standard deviations using the File → Save menu entry.

2 REFRACT3D – how does it work?

The user manual finished with pressing the "Invert" button. Now, we can look under the hood and describe the technical details.

When the corresponding sources and receivers are found, the lengths of rays in individual cells are computed and the matrix of coefficients $\mathbf{A}$ is assembled. Next the vector of right-hand sides $\mathbf{b}$, the data $\mathbf{C}_D$ and model $\mathbf{C}_M$ covariance matrices and the vector of *a priori* information $\mathbf{m}_{pr}$ (starting model) are created. Finally, the solution is found solving the equation (Tarantola, 2005):

$$\mathbf{x} = \mathbf{m_{pr}} + \left(\mathbf{A}^t\mathbf{C_D^{-1}}\mathbf{A} + \mathbf{C_M^{-1}}\right)^{-1}\mathbf{A}^t\mathbf{C_D^{-1}}\left(\mathbf{t} - \mathbf{A}\mathbf{m_{pr}}\right), \tag{1}$$

This equation is solved either by the built-in routines or by the Octave script, both following the same algorithm. The matrix inversion is done by means of the singular value decomposition (SVD) and the Moore-Penrose pseudoinverse (generalised inverse, pseudoinverse) technique.

In case, when user selects the Octave script usage, the matrix $\mathbf{A}$ and vector of right-hand sides $\mathbf{b}$ are written to the files. The files are called `MatA.csv` and `VectB.csv` and are located in the same directory as the REFRACT3D. Both of these files are read by the `ComputeSVD.m` script. The only role of the `ComputeSVD.m` script is to solve the system of equations and to compute uncertainties of individual unknowns.

The singular value decomposition (SVD) is a well known technique of solving systems of linear equations, suitable when the system is ill determined. The SVD algorithm was described e.g. by the Press *et al.* in their book (Press *et al.* 1994). The singular value decomposition of a matrix $\mathbf{A}$ is:

$$\mathbf{A} = \mathbf{U}\mathbf{W}\mathbf{V}^T \tag{2}$$

and the inverse of a matrix $\mathbf{A}$ is

$$\mathbf{A}^{-1} = \mathbf{U}\mathbf{W}^{-1}\mathbf{V}^T, \tag{3}$$

where $\mathbf{W}^{-1}$ is an inverse matrix to $\mathbf{W}$:

$$\mathbf{W}^{-1} = \mathrm{diag}\left(\frac{1}{w_i}\right). \tag{4}$$

The pseudoinverse is closely related to the SVD and the pseudoinverse matrix can be considered analogous to the inverse matrix. Pseudoinverse of a matrix $\mathbf{W}$ is a matrix $\mathbf{W}^{\mathrm{I}}$:

$$\mathbf{W}^{\mathrm{I}} = \mathrm{diag}\left(p_i\right), \tag{5}$$

where

$$\begin{array}{ll} p_i = \frac{1}{w_i} & \text{for } w_i \neq 0, \\ p_i = 0 & \text{for } w_i = 0 \end{array}$$

and a matrix $\mathbf{A}^+$ is the pseudoinverse of a matrix A:

$$\mathbf{A}^+ = \mathbf{U}\mathbf{W}^{\mathrm{I}}\mathbf{V}^T. \tag{6}$$

The script solves the equations and writes computed depths, velocities and their standard deviations to files `VectX.csv`, `VectX_std.csv` and `VectXX.csv`. `VectXX.csv` file is used as an *a priori* information for next iteration. Therefore before running the inversion make sure that no file with this name is present in the REFRACT3D directory. The files `VectX.csv` and `VectXX.csv` are identical, the latter one is used only for supplying *a priori* information for successive iterations, but may also be used as a start-up point when additional iteration is desired, or for supplying more complex *a priori* information for first iteration. The file is an ASCII file with one column of floating-point numbers. These are depths of every cell of all refractors and slownesses in all cells of every refractor.

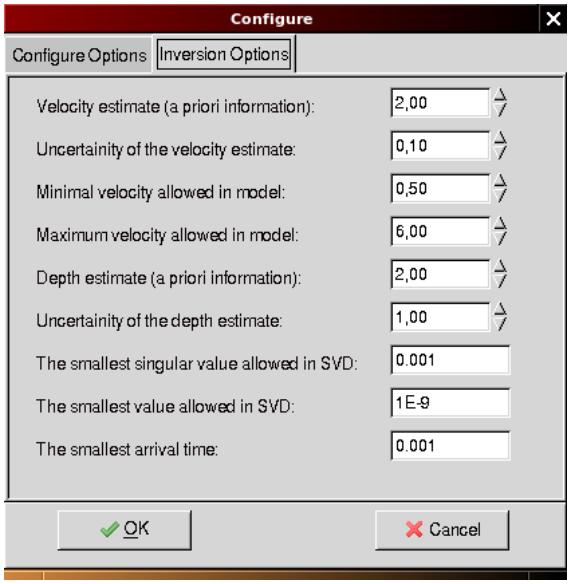

Figure 6: Inversion options window of REFRACT3D.

`VectX.csv` and `VectX_std.csv` contains resolved depths and slownesses, whilst the latter one also includes appropriate standard deviations. These files are read by the REFRACT3D, the nonsense resolved values (e.g. the negative ones) are deleted and replaced with the default ones and new matrix of coefficients is computed. This process is repeated until the desired number of iterations is reached. The RMS (root mean squared error) of travel times is written on the status bar and to the standard output. (Exactly the same algorithm is followed when using the built-in routines.)

The inversion process needs an *a priori* information about the geological media in order to stabilise the inversion process. However it is not much sensitive to slightly wrong values of *a priori* information. In a case of a good ray coverage, any values which make sense should be sufficient. If the user would like to change the default values, it can be done using the Options → Configure → menu entry (Fig. 6), or, when using the Octave script, by editing appropriate lines in the `ComputeSVD.m` script. The lines defining values of important parameters are on the beginning of the file and are well commented, so defining of new values should be easily done also by the non-programmers (see Fig. 7).

3 References

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P., 1994. *Numerical Recipes in FORTRAN: The Art of Scientific Computing.* Cambridge University Press, second edition.

Tarantola, A., 2005. *Inverse Problem Theory.* Society for Industrial and Applied Mathematics, Philadelphia.

Valenta, J. and Dohnal, J., 2007. 3D seismic travel time surveying – a comparison of the time-term method and tomography (an example from an archaeological site). *Journal of Applied Geophysics*, 63, 46–58.

Valenta, J., 2007. New approaches in high-resolution shallow seismic prospection. PhD thesis.

```
vel_prior    = 2;     # default (initial) value of velocities (a priori
                      # information)
vel_uncert   = 0.1;   # standard deviation of initial value of velocities
depth_prior  = 2;     # default (initial) value of depths (a priori
                      # information)
depth_uncert = 1;     # standard deviation of initial value of depths
data_uncert  = 0.1;   # standard deviation of travel time errors
min_vel      = 1.5;   # minimal velocity allowed in model (if smaller number
                      # than min_vel is encountered, then the value
                      # of vel_prior is used)
max_vel      = 6;     # maximal velocity allowed in model
min_depth    = 0.2;   # minimal depth allowed in model
max_depth    = 5;     # maximal depth allowed in model
tol          = 0.001; # the smallest singular value allowed in SVD
```

Figure 7: Default values of user-adjustable parameters in `ComputeSVD.m` script