Δελτίο της Ελληνικής Γεωλογικής Εταιρίας τομ. XXXVI, 2004
Πρακτικά 10ου Διεθνούς Συνεδρίου, Θεσ/νίκη Απρίλιος 2004

Bulletin of the Geological Society of Greece vol. XXXVI, 2004
Proceedings of the 10th International Congress, Thessaloniki, April 2004

A FREEWARE PACKAGE FOR THE ANALYSIS AND INTERPRETATION OF COMMON-OFFSET GROUND PROBING RADAR DATA, BASED ON GENERAL PURPOSE COMPUTING ENGINES

Tzanis A. and Kafetsis G.
Department of Geophysics and Geothermy, University of Athens, Panepistimiopoli, Zografou 157 84, Greece; atzanis@geol.uoa.gr.

ABSTRACT

The Ground Probing Radar (GPR) has become an invaluable means of exploring shallow structures for geoscientific, engineering environmental and archaeological work. At the same time, GPR analysis software is mostly proprietary and expensive. The academic freeware community has been slow to react the limited free software is usually highly focused and generally unorganized. Herein we report the beginning of an attempt to remedy this situation with a cross-platform freeware analysis and interpretation package, which can also be expandable and customizable to the requirements of a particular user with relatively little programming effort. Almost ideal platforms for the development of such a kind of software are general purpose computing engines, such as MATLAB™ and/or OCTAVE. These provide two very similar, cross-platform and complete environments for the development of advanced analysis software. We have designed a two-layered software system. The Bottom Layer comprises a set of self-contained and self-documented functions to handle, visualize, process and interpret GPR data and is expandable with addition of a user's own functions. The Top Layer organizes these functions, automating data management and streamlining the flow of work by means of a GU Interface. At the present stage of development, this GPR analysis offers a decent and in many respects advanced means of treating and interpreting common-offset data; future releases may even become competitive, if maintained and developed by collective effort along the spirit of the GNU project.

1 INTRODUCTION

The Ground Probing Radar (GPR) has become an invaluable and almost indispensable means of exploring shallow structures for geoscientific, engineering environmental and archaeological work. At the same time, GPR analysis software is mostly proprietary and usually available from GPR manufacturers or a handful of other vendors (e.g. REFLEX by Sandmeier Engineering; GRORADAR™ by Garry Olhoeft at http://www.g-p-r.com/links.htm, etc.)

There are only two exceptions. A good but quite limited freeware package provided by the USGS (Lucius and Powers, 2002), which will never work in non-Microsoft platforms and due to the particularities of its graphics drivers has even problems working in Windows XP. The second freeware package is the Radar Unix by Grandjean and Durand (1999), which is limited to Unix and Linux platforms; it does not work in Windows, OS2 or Macintosh systems unless they're augmented with the CygWin Linux emulator, and then under conditions. Furthermore, RU draws processing power from the Seismic Unix (SU) analysis system, but as it is based on an outdated version of SU, it requires extensive overhaul. Both freeware packages are written in C, while RU's graphical interface (Xforms) is written in C++. This renders both of them rather unwieldy to modification or augmentation by the average practitioner.

The academic freeware community has been slow to react on this issue and the limited freely distributed software is usually focused on very particular problems (mainly data input / output), generally unorganized and so diversely programmed, that it cannot form a consistent basis for the reliable manipulation of GPR data.

Herein we report the beginning of an attempt to remedy this situation with a freeware analysis and interpretation package, which can be truly cross-platform, as well as expandable and customizable to the needs of a particular user, with little programming effort. Although easy to say, this would be a truly demanding and ambitious undertaking, had it not been for the existence of powerful, general purpose computing engines. On this basis, the realization of such a project is feasible because a computing engine will generally provide a complete high level programming environment, inclusive of graphics, which facilitates the development of advanced software because all the complexities pertaining to low level interfacing, programming and functionality are taken care by the engines themselves. This is much easier and faster than building programs from scratch in some conventional high level language such as C, C++ or FORTRAN. A cross-platform package is also possible because computing engines are usually made for, or transportable to more than one operating systems.

2 CHOOSING THE APPROPRIATE ENGINE(S)

General purpose scientific computing engines are not many: The most commonly available are the proprietary IDL™ and MATLAB™ and the freeware OCTAVE and SCILAB. Of course, there is an abundance of other (proprietary or non-proprietary) scientific data analysis software, which is generally not suitable for our purpose because it is usually specialized (e.g. statistical analysis, curve fitting, data modelling, graphics etc.) and does not provide a sufficiently general and versatile enough programming environment for the development of integrated programs.

MATLAB and OCTAVE are, to a certain degree, complementary. They both provide the same complete high-level programming language, which is furthermore fully vectorized, but is not as demanding as C/C++ or FORTRAN because all of the low-level functionality and interfacing is taken care of by the engines themselves. Thus it is possible to create the same core software, which may run in both engines. The fact that MATLAB is proprietary software while OCTAVE is freeware can also be advantageous because the core software will work for everybody, even for those who cannot afford MATLAB.

On the other hand, MATLAB being proprietary enjoys continuous and dedicated support and development and is definitely richer in tools and more functional, in that it offers very extensive graphical and GUI capabilities and a multitude of programming shortcuts. OCTAVE offers limited GUI support unless operated together with additional freeware packages to which it may link dynamically. Such are, for instance the graphics package PLplot, which is also modelled on MATLAB and the well known Tcl-Tk, which may provide GUI support (e.g. via the Tk_Octave interface by João Cardoso. Thus, the combination OCTAVE/ PLplot / Tcl-Tk could offer a quite cumbersome to build, but totally free and quite decent substitute for MATLAB.

In short, the complementarities of MATLAB and OCTAVE allow for the development of software which, with relatively minor changes may work in all platforms and operating systems in which MATLAB can be installed (that is practically everywhere), as well as all platforms in which OCTAVE/ PLplot/ Tcl-Tk can be installed. The latter includes all flavours of UNIX and LINUX, but also the Microsoft Windows family of operating systems, provided of course they're augmented with the appropriate version of the (free) CygWin Linux emulator. Such versatility is *not* enjoyed by other computing engines.

3 DESIGN AND IMPLEMENTATION

We have designed a two-layered software system, in which the *bottom layer* comprises a suite of scripts and functions to handle, display and process the data, while the *top layer* organizes these functions, automating data management and streamlining the flow of work by means of a GU Interface. As evident from the above discussion this interface has to be different in MATLAB and OCTAVE and therefore, the package must also comprise two clones: one for MATLAB (matGPR v1.0) and one for OCTAVE (ocGPR v1.0). It is also possible to produce a basic (core) clone without GUI support; this would provide only rudimentary dialog and messaging services through the engines' command windows and its functions would have to be invoked manually, but it would be truly plat-

form independent and would work in both MATLAB and OCTAVE environments without any ado. We will now attempt a concise description of the package.

3.1 The Top Layer

In effect, the Top Layer is a long script that organizes the bottom layer functions and automates data management according to the flowchart of Figure 1. The physical appearance of the GU Interface is shown in Figure 2 (matGPR only).

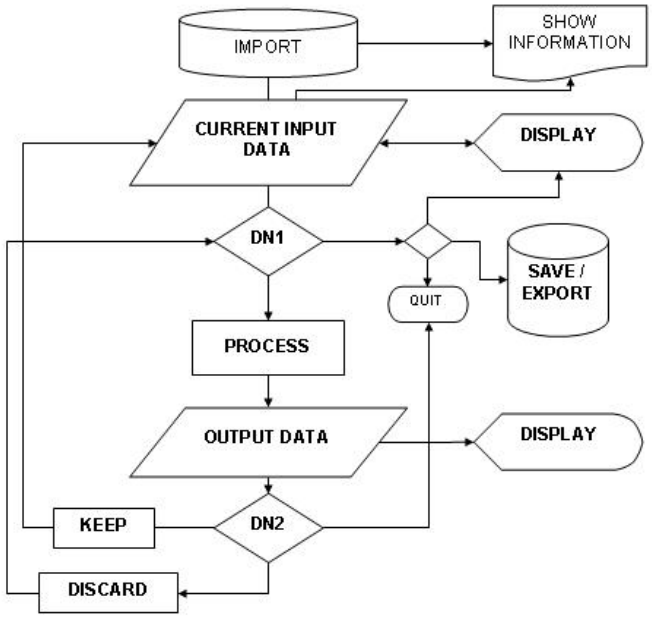

Figure 1. The flowchart of a typical GPR data analysis session with matGPR and ocGPR.

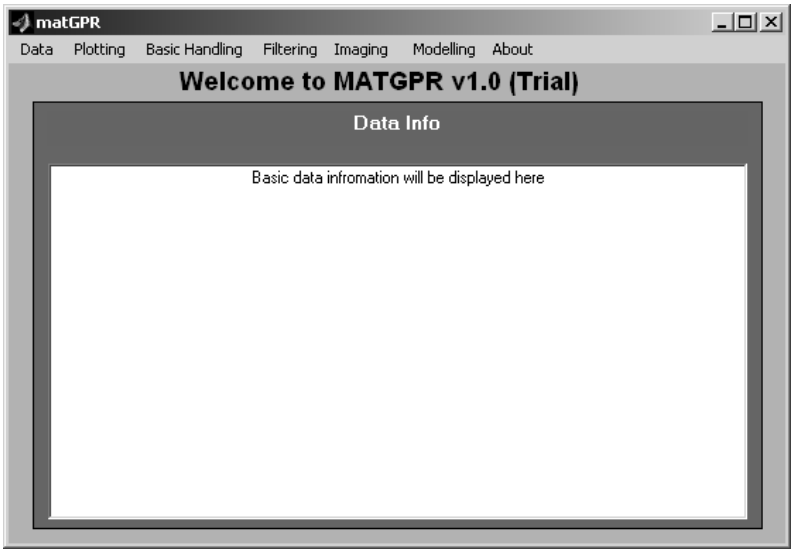

Figure 2. Physical appearance of the matPR home window, i.e. of the GUI Top Layer that realizes the flowchart of Figure 1.

The design philosophy is quite simple: work flows in a continuous cycle between the “current input data”, i.e. that data before some processing operation (step) and the “output data”, i.e. the data

resulting from this operation. The user imports, displays and inspects the current input data with the appropriate choices under the “Data” and “Plotting” menus. Then he/she decides and applies a processing step and inspects the result. If satisfied, the user will ‘keep’ the result, replacing the current input data with the output data and will cycle through the same procedure with a new processing step. If not, he/she will ignore or discard the result and cycle with another processing step. An example of this procedure is shown in Figures 3 and 4. While in Decision Node 1 (DN1), the user may save or export the current input data. Soft and hard copies of the current input and output data can be made at any time. Data management decisions are realized with the appropriate choices under the “Data” menu (Figure 2 and Appendix 1), while the ‘next processing step’ is taken from one of the tasks under the “Basic Handling”, “Filtering”, “Imaging” and “Modelling” menus (also see Figure 2 and Appendix 1). Note also that some processes under the “Plotting” menu (viewing of traces and spectra) can run parallel to the main processing cycle and automatically update themselves.

3.2 The Bottom Layer

This comprises a set of self-contained and self-documented scripts and functions to visualize process and interpret GPR data. Any function can be called individually to perform a task or part of a task. For instance, the call *[tz, z, nz, dz, ddm] = ttoz(traveltime, d, dt, v1d),* will take the two-way *traveltime*, the two dimensional (time - distance) radargram *d,* the sampling interval *dt* and the one-dimensional velocity-thickness profile *v1d*, will perform a time-to-depth conversion and will return the a traveltime-depth vector *(tz)*, a depth vector *(z),* the number of depth estimates *(nz)*, the depth sampling interval *(dz)* and the converted depth - distance radargram *ddm*, re-sampled at intervals *dz*. The invocation *help ttoz* (without arguments) will display the embedded self-documentation (help facility). The Bottom Layer is expandable with addition of a user’s own functions.

The tasks available at the present state of development are detailed in Appendix 1. The software is organized in 6 basic units: *Data (management)*, *Plotting*, *Basic Handling*, *Filtering*, *Imaging* and *Modelling* (Figure 2 and left column in Appendix 1). Each unit comprises a number of homologous task groups as shown in the middle and right columns of Appendix 1 (where they are indicated with common shading). Each task may require one or more functions to perform.

The package can accept data in the formats of the most common GPR manufacturers (GSSI and Måla geophysics and shortly, Sensors and Software (Pulse EKKO). Because the package is currently developed for common-offset surveys, only single-channel data files are acceptable. Data stored in SU and SEG-Y Revision 0.0 formats can also be imported. With reference to the latter, it should be noted that because of the outdated IBM floating point format used for storage, it has proven to be more reliable to convert the data with an external, well tested C language routine. We have implemented the program “segyread” from the SU suite, which is called by our “readsegy” function to convert the data to an interim SU format file that is subsequently imported with native routines. Once read, the data can be saved in native binary format (MAT files) for faster and easier subsequent access. Data generated at different processing stages should also better be stored in this format. For distribution or exchange, the data can be exported to SU and SEG-Y format files. For the reasons quoted above, the external SU program “segywrite” is implemented.

Data visualization options include image (colour-coded) displays with several colour maps for better discrimination of the details, wiggle-trace displays and variable-area displays. It is also possible to plot and scrutinize individual traces and trace spectra in individual figure windows.

Basic data handling includes graphical determination and adjustment of time-zero, removal of the global mean (background trace), dewow and gain manipulation. The latter includes Automatic Gain Control (AGC) functions, both in the standard form and with Gaussian tapering (for an example see Figure 3). A gain function of the form $g(t) = s*t^p$, can also be interactively applied. A graphical routine to facilitate manual design of range-gain functions is nearing completion. Additional handling facilities include resampling (increase or decrease of the recording rate) in time and in space (along the scan line), using the versatile bandlimited sinc interpolation algorithm of Smith and Gosset (1984).

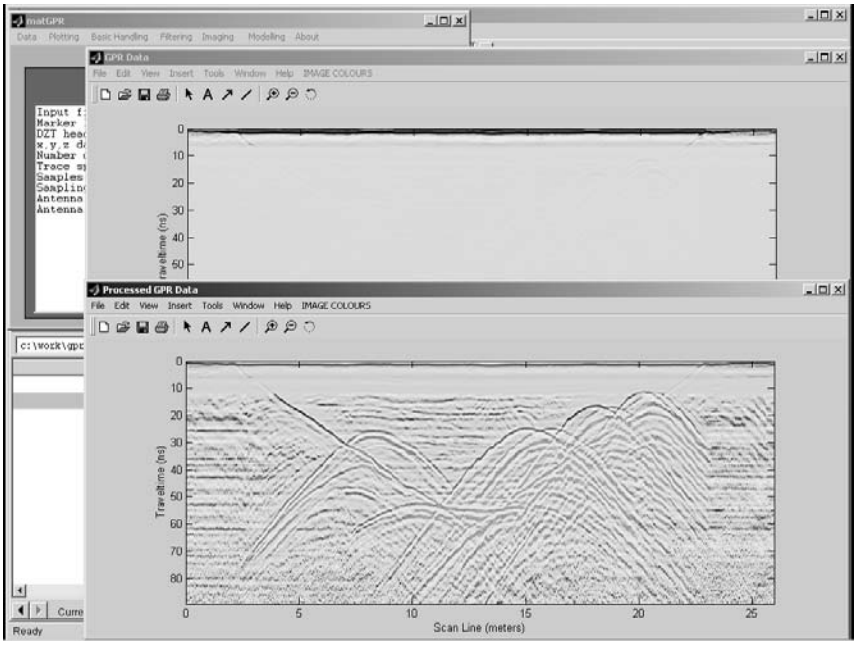

Figure 3. A snapshot of a processing session. The matGPR home window can be seen in the background (top left); the "current input data" is in the middle ground ("GPR Data") and comprises an apparently featureless section. The "output data" is in the foreground ("Processed GPR Data") and shows the same section after an AGC operation, revealing the existence of multiple diffractors buried in a trench. This data set has been distributed by Grandjean and Durand (1998).

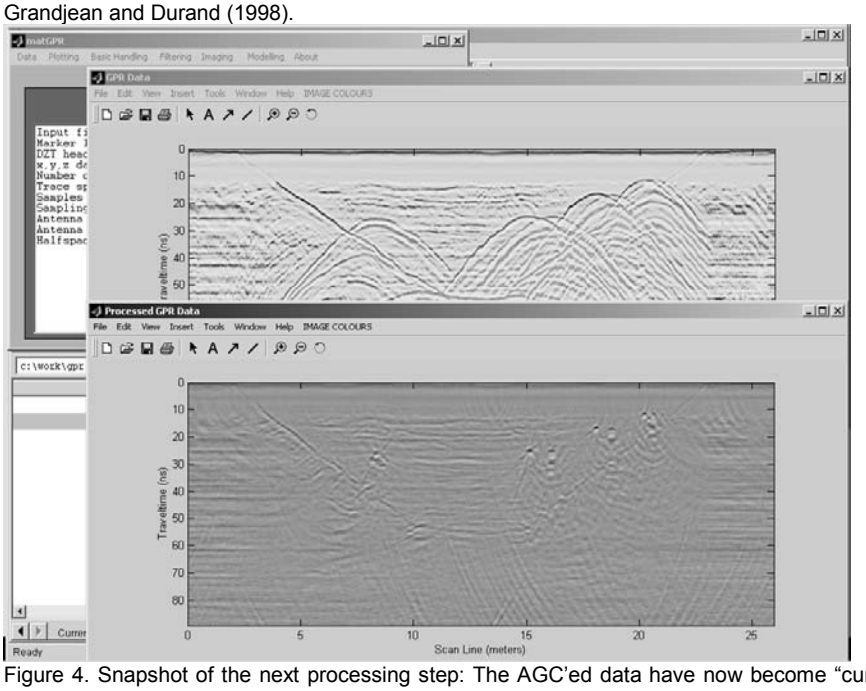

Figure 4. Snapshot of the next processing step: The AGC'ed data have now become "current input data" and the new "output data" shows the time-migrated section (Gazdag, 1978), in which it is possible to observe the outlines of the trench and the locations of the diffractors.

For GPR instruments *not* equipped with survey wheels or other automatic distance measuring and triggering device, a suite of marker interpolation routines is also provided, so as to transform data collected at equal-time spacing mode, to data at equal-distance spacing. This suite of routines

also facilitates the three-dimensional determination of trace locations, with respect to some local coordinate system. Such information is necessary for the application of static corrections prior to imaging, as well as for 3D data visualization among other things.

Smoothing facilities include mean and median spatial filtering in one and two dimensions. Because this type of filtering is computationally intensive, an external Fortran 90 program is employed to do the heavier part of the work. It should be noted that both MATLAB and OCTAVE workspaces are basically interpreters of their programming language and therefore inherently slow when sequential number crunching in nested loops is necessary and unavoidable. In this case, external compiled programs callable from within functions can offer a very fast solution to such problem. Additional spatial filtering techniques include the removal of a sliding window mean (background) trace. This will remove small horizontal features (coherent signal) from the data and may be used to expose reflections that dip at high angles, (removing for instance most reflections due to hydrogeologic sources). It is also possible to remove the sliding-window's "foreground" traces: the background trace is assigned to the centre trace in the sliding window rather than being subtracted from the data. This will remove high-angle reflections, for instance to expose the sub-horizontal hydrogeologic features (a dip filter).

The software also provides for zero-phase FIR filtering of the frequency content (applied 'vertically' to traces), and of the wavenumber content (applied 'horizontally' to – equally spaced – scan lines). All types of filters are available (low / high / band pass, band stop and notch). All filters can be designed interactively (graphically) and tested before they're applied to the data. Finally, this suite of functions is complemented with an interactive F-K filter design and implementation routine, facilitating the application of zone pass / stop filters, fan (velocity range) filters and updip/ downdip mapping in the frequency – wavenumber domain.

The package also offers a number of imaging and modelling tools to assist interpretation. These include velocity analysis by interactive fitting of diffraction front hyperbolae (assumes non-dispersive propagation). Advanced interpretation tools include static corrections, F-K migration for layered velocity structures (Stolt, 1978), Phase-shifting migration for layered velocity structures (Gazdag, 1978), Split-step F-K migration for 2-D velocity structures (Stoffa et al., 1990, Sena et al., 2003) and time-to-depth conversion (see Figure 4 for an example). Note that migration is a computationally intensive business, involving integrations with multiply-nested loops. This would cause the MATLAB and OCTAVE interpreters to run slowly and would delay the results. As before, this problem is remedied with external, compiled Fortran 90 code callable from within the migration functions to perform the heavy duty work.

Finally, interpretation can be aided with 2-D forward modelling tools. These includes a model designer and constructor, which in the MATLAB clone is fully graphical and facilitates the easy and fast creation and editing of models comprising multiple bodies of polygonal, circular or elliptical cross-section, with point-and-click operations. In the OCTAVE clone, due to the less advanced development of PLplot, this tool is a hybrid and is based on a different approach. The models can be stored in disk and can be recalled for the calculation of 2D velocity profiles (to be used for Split-step migration) as well as for modelling. The modelling technique implemented herein is in effect adjoint Split-step imaging with the algorithm of Bitri and Grandjean (1998), appropriately adapted for the requirements of our package. Here as well, the main number crunching routine is an external C or /and Fortran 90 program fashioned after the program "radar2d4" of the above authors.

3.3 Additional features

A useful side-kick is the possibility to import / export of data in SU format. This provides the capability for invoking the power of SU from within MATLAB or OCTAVE, to perform even more complex data manipulations. The invocation of SU programs can be done at the fingertip (using the system call functions of MATLAB or OCTAVE), or through more complex scripts and functions that make system calls and can work in tandem with the main package. This facility is readily available in Unix, Linux and MS Windows/CygWin systems, and even pure MS Windows systems if there will ever be a free and functional porting of SU to these platforms – the only existing effort by Hugh Winkler (SUNT) is not easily accessible or fully operable.

4 EPILOGUE

At the present stage of development matGPR and ocGPR provide a broad and functional range of tools for the analysis of common-offset GPR data. Nevertheless, they are still infant and incomplete! For instance, they do not offer deconvolution facilities, which would be a valuable addition; these are currently on the drawing board and will be implemented in the immediate future. One can also think of several other processing tools and shortcuts that may be included in (near) future releases, adding to the power of the package(s).

We consider useful to point out that at this stage, matGPR and ocGPR do not aspire to become real substitutes for commercial analysis programs, which have been developed, debugged and perfected for long. However, the do offer a very decent and in several aspects advanced means of treating common-offset data. We believe that they have the potential to become highly competitive, if maintained, developed and expanded over time with contributions from other authors, along the spirit of the GNU project. Both packages will be freely available through personal communication and over the Internet, in the immediate future.

REFERENCES

Bitri, A. and Grandjean, G., 1998. Frequency – wavenumber modelling and migration of 2D GPR data in moderately heterogeneous dispersive media, *Geophysical Prospecting*, **46**, 287-301.

Gazdag, J., 1978. Wave equation migration with the phase-shift method, *Geophysics*, **43**, 1342-1351.

Grandjean, G. and Durand, H., 1999. Radar Unix: a complete package for GPR data processing, *Computers & Geosciences* 25 141-149.

Lucius, J.E. and Powers, M.H., 2002. *GPR Data Processing Computer Software for the PC,* USGS Open-File Report 02-166.

PLplot can be sought at URL http://sourceforge.net/projects/plplot/

Sena, A.R., Stoffa, P.L. and Sen, M. K., 2003. Split Step Fourier Migration of Ground Penetrating Radar Data: 73th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1023-1026.

Smith, J. O. and Gossett, P., 1984. A flexible sampling-rate conversion method. In ICASSP-84, Volume II, pp. 19.4.1-19.4.2, New York: IEEE Press.

Stoffa, P.L., Fokkema, J.T., de Luna Freire, R.M. and Kessinger, W.P., 1990. Split-step Fourier migration, *Geophysics*, **55**, 410-421.

Stolt, R.H., 1978. Migration by Fourier Transform, *Geophysics*, **43**, 23-48.

Tk_Octave can be sought at URL http://paginas.fe.up.pt/~jcard/software/tk_octave/tk_octave.html.

APPENDIX 1

The following list presents the task groups and tasks available in matGPR v1.0 (Trial) and ocGPR v1.0 (Trial). Note that one task may require more than one function to perform and vice versa. Task groups are indicated with similar shading.

DATA	Import raw data	→	GSSI (DZT) format Måla (RD3) format Seismic Unix format SEG-Y format
	Import from native (.mat) format		
	Keep processed data Discard processed data		
	Save to native (.mat) format		
	Export data	→	Seismic Unix format SEG-Y format
PLOTTING	Select display mode	→	image (colour scale) wiggle traces variable area
	View data		

	View processed data		
	View traces View spectra		
	View processed traces View processed spectra		
	View markers		
BASIC HANDLING	Adjust signal position (time zero)		
	Remove global background trace Dewow		
	Remove DZT header gain Automatic gain control (AGC) Gaussian-taper AGC Gain function $g(t)=s*a^t$		
	Resample traces Resample scan line		
	Edit marker file	➔	use editor run edit utility
	Marker interpolate to equal spacing		
	Make x, y, z trace coordinates		
FILTERING	Spatial mean filter (1d and 2d) Spatial median filter (1d and 2d) Remove sliding window background trace Remove sliding window foreground trace		
	FIR frequency domain filter	➔	low pass high pass band pass band stop notch
	FIR K-space filter	➔	low pass high pass band pass band stop
	F-K filter		
IMAGING	Fit diffraction front hyperbola		
	Static corrections		
	Import 1D velocity model Stolt F-K migration Gazdag phase-shifting migration		
	Import 2D velocity model Split-step 2D migration		
	Time to depth conversion		
MODELLING	Design 2D model Compute 2D model (Split-step method)		