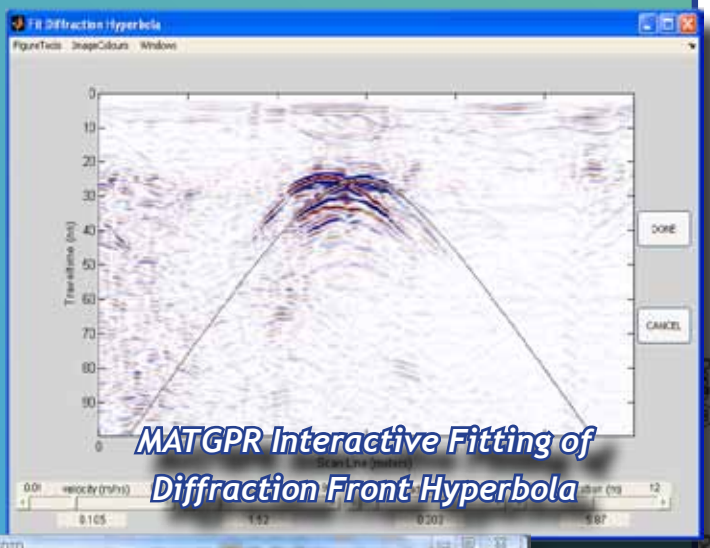
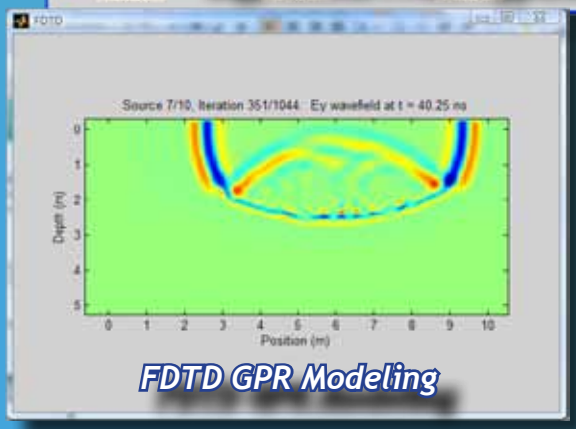
FastTIMES
April 2010
Volume 15, Number 1
www.eegs.org

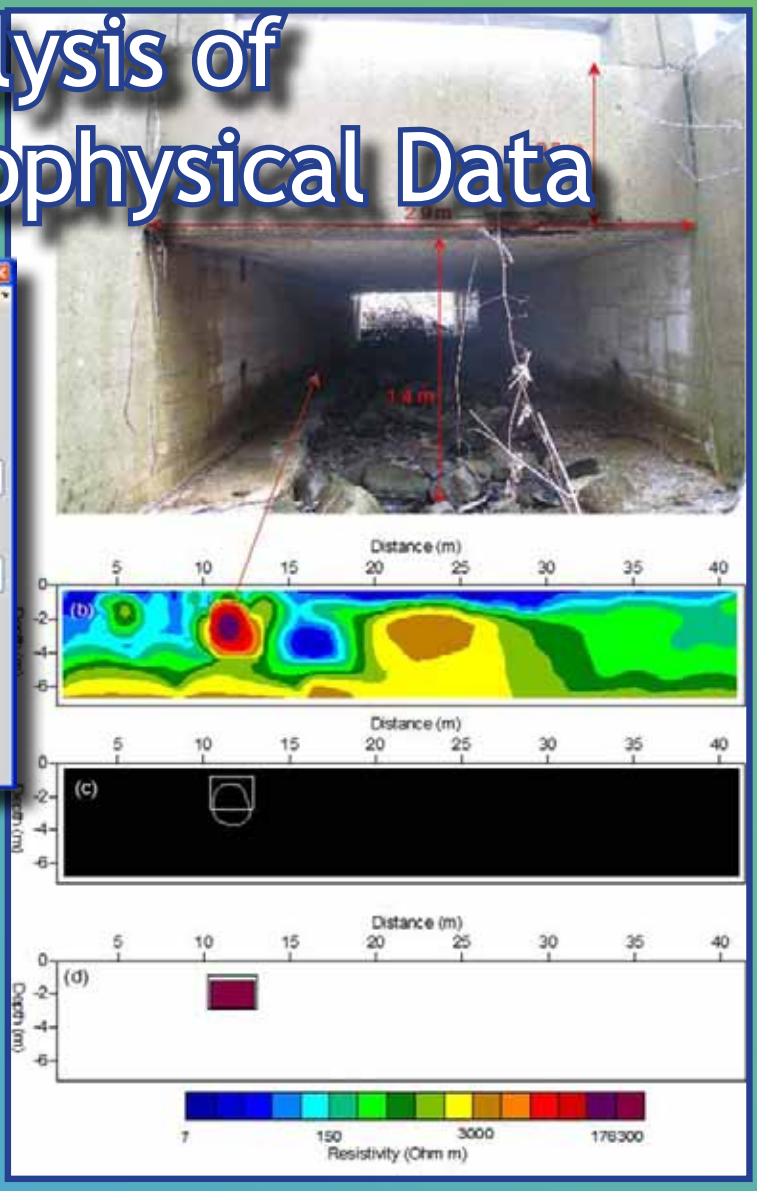
Software for Analysis of Near-Surface Geophysical Data



MATGPR Interactive Fitting of Diffraction Front Hyperbola



FDTD GPR Modeling



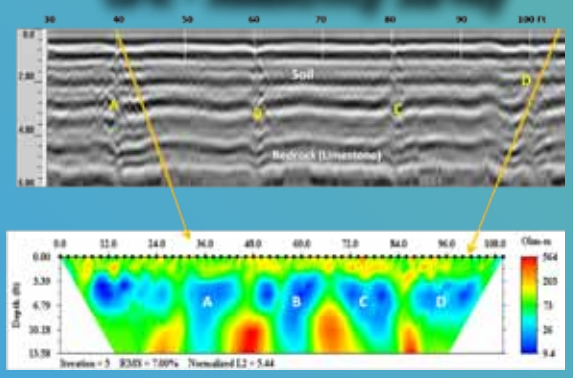
Neural Network Target Reconstruction

Also in this issue . . .

- New Products
- State-of-the-Art in Multi-Dimensional Electromagnetics
- Conference on Sinkholes & Karst
- Follow FastTIMES on Twitter

. . . and more!

GPR - Resistivity Survey



FastTIMES *welcomes short articles on applications of geophysics to the near surface in many disciplines, including engineering and environmental problems, geology, soil science, hydrology, archaeology, and astronomy. In the articles that follow, the authors present software developed to process and interpret near-surface geophysical data.*

MATGPR Release 2: A freeware MATLAB® package for the analysis & interpretation of common & single offset GPR data

Andreas Tzanis, Department of Geophysics, National and Kapodistrian University of Athens, Panepistimiopoli, Zografou 15784, Greece (atzanis@geol.uoa.gr)

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. There are only three exceptions. A good but quite limited freeware package provided by the USGS (Lucius and Powers, 2002), which will not work in non-Microsoft platforms and due to the particularities of its graphics drivers has even problems working in Windows XP. Another freeware is the Radar Unix by Grandjean and Durand (1999), which is limited to Unix and Linux platforms; it does not work in Windows, OS(2) or Mackintosh systems unless they're augmented with the CygWin Linux emulator, and then under severe restrictions. 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/ C++ and are rather unwieldy to modification or augmentation by the average practitioner. The most recent arrival on the scene is the openGPR project of Matthias Schuh, Tuebingen, Germany. This can be found in the URL http://opengpr.sourceforge.net/?openGPR and is a rather impressive piece or work. Unfortunately, it works only in Linux OS and is heavily dependent on SU and a host of other support programs. It can be used by people who know their way about with computing.

With the references above aside, the academic free software community has been slow to react and the limited free 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. matGPR marks an effort to create a GPR analysis and interpretation package that can be truly cross-platform, as well as expandable and customizable with little programming effort. This is an ambiguous project, albeit feasible because MATLAB provides an all-inclusive high level programming environment that facilitates the development of advanced software, much easier and faster than building programs with conventional high level languages such as C, C++ or FORTRAN.

The appearance of MATLAB and its imitators made a big impact in the scientific community. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. To do this, MATLAB also introduced a new vector-oriented programming language, an interactive environment, built-in graphics and strict runtime error checking. These features offered many advantages and boosted productivity. MATLAB has since evolved to embed a large collection of state-of-the-art numerical tools, high quality graphics, object-oriented extensions, a built-in interactive

debugger, web services and a host of other facilities. Of course, C and FORTRAN have also evolved, with some editions also acquiring high-level (visual) application development environments. However, neither offers as many facilities as MATLAB does, graphics for instance being a major issue. When it comes to effectiveness, what really matters is the efficiency of the entire process of realizing a new scientific idea, or creating an analysis and interpretation procedure. Arguably, MATLAB is overall much more efficient in this respect, facilitating the rapid prototyping of new algorithms or applications.

This paper comprises a presentation of matGPR Release 2 from a practical (user's) point of view. Programming and technical details will be given sparingly and the science behind the analytical methods will be limited to the bare essential; references, however, will be provided whenever possible, in the interest of completeness. The distribution bundle of matGPR comes complete with documentation: the "matGPR Manual and Technical Reference" and its HTML equivalent (on-line help). These documents amply explain the details of programming methods, analytical methods and execution procedures. In consequence, any description given herein will be complete but succinct: for additional information, the interested reader will be (implicitly or explicitly) referred to the "matGPR Documentation" whenever necessary.

General Features and Organization

matGPR is a two-layered software system, in which the bottom layer comprises a suite of functions to handle, display, inspect and process the data, while the top layer organizes these functions, automating data management and streamlining the flow of work.

The top layer and backbone of the program is realized in the form of the matGPR GUI (Figure 1). Data management decisions are made with the appropriate choices under the *Data* menu. Visualization

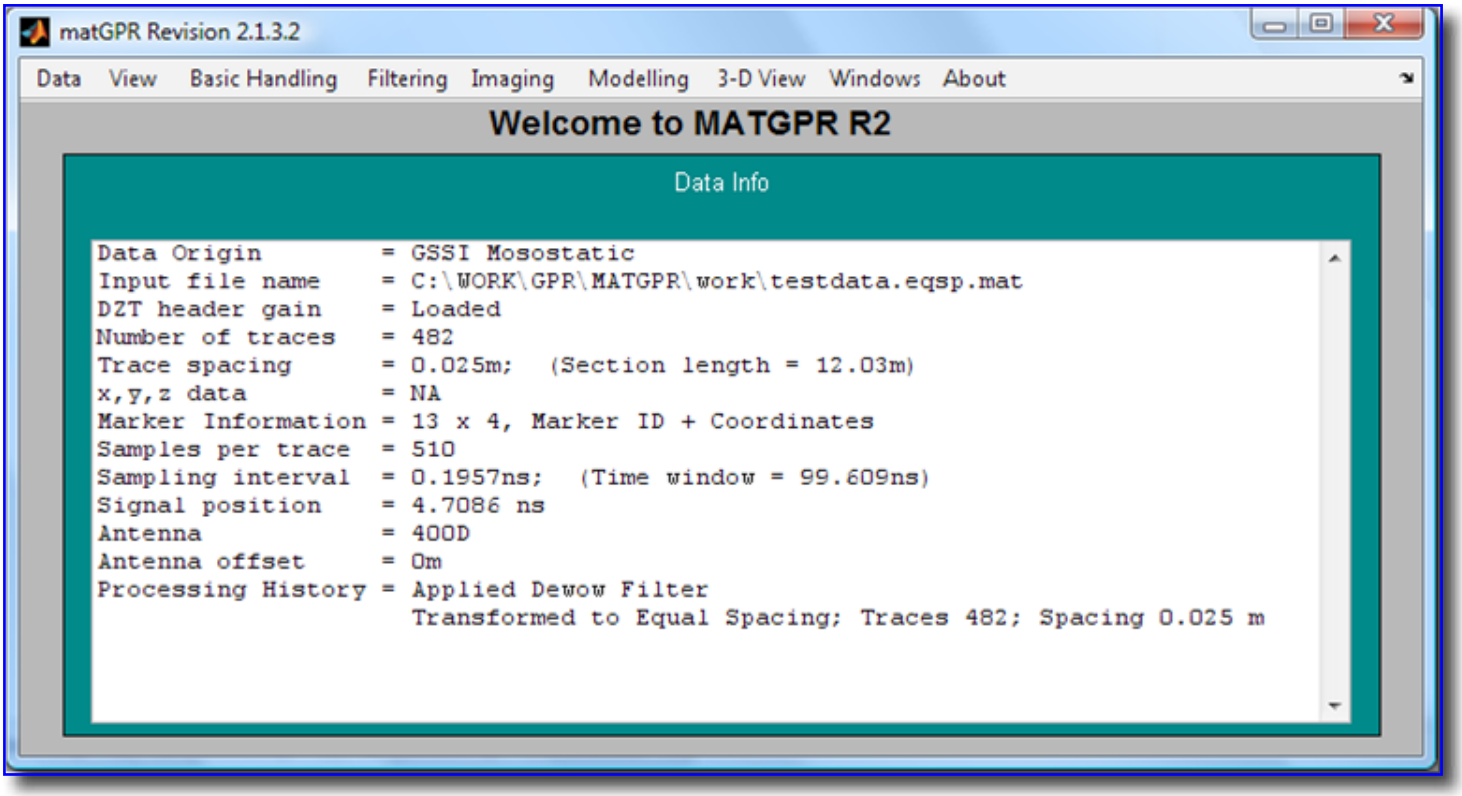


Figure 1. The matGPR GUI complete with information about the data being handled and its processing history.

and inspection is facilitated by utilities offered in the *View* menu, while processing and interpretation utilities can be found in the *Basic Handling, Filtering, Imaging* and *Modeling* menus. The *3D-View* menu provides a suite of functions to assemble, manipulate and visualize three-dimensional GPR data volumes and the *Windows* menu offers a fast navigation and window-switching service *exclusive* to matGPR. Finally, the *About* menu includes the *on-line help* and updater services.

The operation of the program is quite simple and succinctly outlined in Figure 2: 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. One may import, display and inspect the Input Data with the appropriate choices under the *Data* and *View* menus. Then one decides and applies a processing step and displays/ inspects the result, i.e. the Output Data. If satisfied, one keeps (holds) the result, replacing the Input Data with the Output Data and repeats the cycle with a new

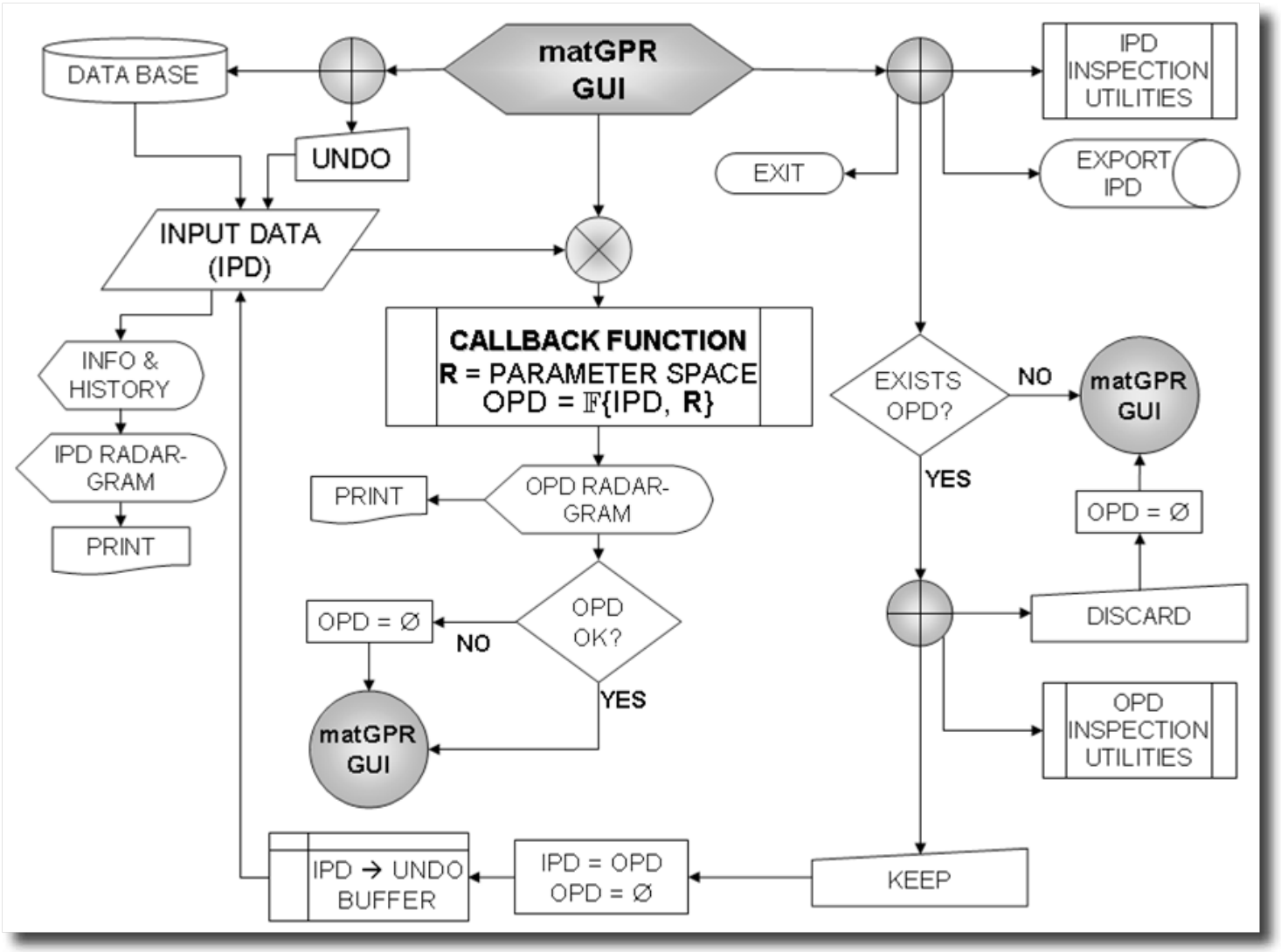


Figure 2. The flowchart of a data analysis session with matGPR.

processing step. If not, one may discard the result and cycle with another processing step. A *multi-level* undo/ restore utility is available at any time at which control is returned to the matGPR GUI. The Input Data may also be saved/ exported via the matGPR GUI. Soft and hard copies of the Input and Output

Data (and any figure produced during a session), can be made at any time using the *FigureTools* menu of the figure windows (see below).

The Input Data is held in a data structure named IPD (for Input Data) and the Output Data in an identical data structure named OPD (for Output Data). In addition to the radargrams (data) the IPD/ OPD structures hold detailed spatial, temporal and archiving information, including the processing history. Some of this information is projected in the matGPR GUI (Figure 2). Details can be found in the matGPR Documentation.

The IPD and OPD structures are accessible through MATLAB's base workspace and can be used or manipulated independently. For example, their fields can be input/ output to the user's own, or third party processing functions, while the results will still be available to matGPR for further manipulation. Essential information about the Input Data is displayed in the matGPR GUI and is updated after each and every processing step.

matGPR is a modular program With the exception of the management and direct visualization utilities each module performs a self-contained operation and comprises the Callback function of the corresponding menu item and its accessory I/O or analysis function. In Figure 2, the Callback function is displayed as a subprogram which includes (calls) the analysis function $\mathbb{F}\{\text{IPD},\mathbf{R}\}$, where **R** is the set of parameters required for the execution of $\mathbb{F}\{\bullet\}$. Although it may appear redundant to have a Callback function execute an analysis function instead of calling it directly, this scheme facilitates flow control operations such as consistency checking and error trapping, as well as data management. It also allows the I/O and analysis functions to be used independently (e.g. from the MATLAB Command Window), or without the restrictions imposed by the matGPR data structures (e.g. with different data or in an altogether different program).

Now, consider that M-code (MATLAB's language) executes rather slowly, on account of being interpreted and not compiled. This may be a serious shortcoming when it comes to computationally intensive tasks. However, MATLAB offers three solutions:

1. It makes an honest effort to use an efficient programming style: for many applications, vectorized and well-written M-code is as fast as compiled code, to the point of being competitive.

2. It can integrate fast, compiled C or FORTRAN code with M-code by means of the MEX-file concept. The FORTRAN-MEX sources are included in the distribution bundle, so that the MEX-file may be (re)built if its functionality is somehow lost.

3. It provides a set of alternative (computationally intensive) functions that drive external, stand-alone FORTRAN executables to do the number crunching.

Details about which processes use MEX-files or alternative functions and how to implement them can be found in the matGPR Documentation.

The modular architecture of matGPR allows one to expand the program with new analysis modules easily and without having to know many details about the program's infrastructure. At any rate, familiarity

with the MATLAB language and graphical object manipulations in particular, is necessary. The matGPR documentation provides detailed instructions of how to expand the program.

It is possible to *undo* a number of processing steps in order to restore the data to a previous state. A GUI displays the history of the data and the user may specify any previous state to which it will be restored (Figure 3). The number of processing steps that may be undone is adjustable; the default is 4, which seems to be a good compromise for Windows XP OS with 2GB of core memory. The previous processing steps stored in the Undo Buffer can be removed to reclaim memory.

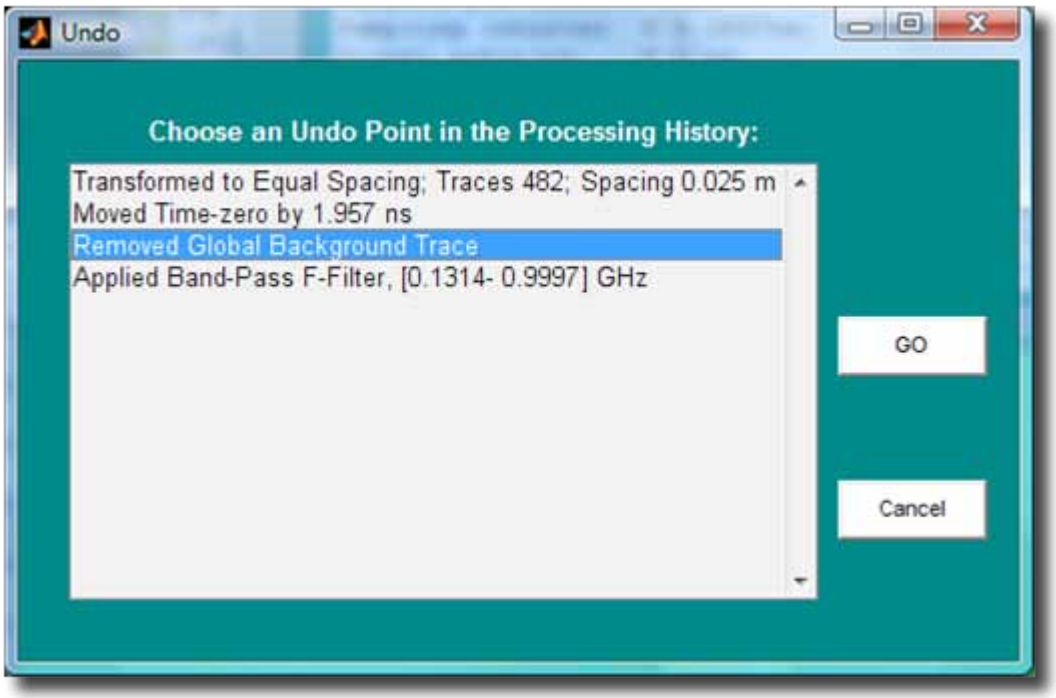


Figure 3. The *Undo* GUI. On pressing GO, the IPD will be restored to the highlighted state of its processing history.

Data Formats and I/O operations

In matGPR, the term *"raw data"* defines data sets that exist in the GPR manufacturer's native storage format and *prior* to importing /converting to the matGPR data structure format. At present, matGPR can import raw data written in the formats of the GPR manufacturers GSSI (RADAN/ DZT), Måla Geophysics (RAMAC/ RD3), Sensors and Software (PULSE EKKO/ DT1) and ZOND (SEG-Y). *Only single-channel* data files are acceptable, inasmuch as matGPR is currently being offered for zero- or single-offset surveys. With respect to bistatic Pulse Ekko (DT1) data, only data recorded in "reflection" mode (single-offset) are fully supported. Multiple-offset data can be imported and displayed, but cannot be processed save for rudimentary editing and filtering.

In addition to the above, matGPR uses the SEG-Y Revision 0 and 1 Data Exchange Formats to import/ export data: it can import data written in both Revision 0 and Revision 1 styles but will export data in Revision 1 style only. These formats are widespread in exploration geophysics and have been adopted as the preferred method of archiving or exchanging data. This is also the reason why matGPR does not export to proprietary file formats (RADAN, RAMAC, etc.). The format used by CWP's *Seismic Unix* package – the SU format – is also supported (see http://www.cwp.mines.edu/cwpcodes/); this is a simplified version of the SEG-Y Rev.0 format. Additional and extensive information about the implementation of the SEG-Y and SU standards exist in the matGPR Do-cumentation.

Finally, matGPR implements two specific binary files formats and also implements MATLAB's standard MAT-file format for interim I/O operations. The matGPR-specific formats are identified by the extensions

.mgp and *.m3d* and are considerably more compact, packing the same information in less than half of the space required by MAT-files. The MGP-file and MAT-file formats are used to store the IPD structure (MGP is the default). The M3D-format is used for storing large 3-D data volumes. The structures of the MGP- and M3D-files are dully described the matGPR Documentation.

Once read, the raw data is automatically saved in MGP-file or MAT-file binary format. The save destination folder is the parent directory of the raw data file. By default, *all* subsequent output operations concerning the derivate (processed) data sets will be directed to the parent directory of the raw data file, which thus becomes the home directory of the data analysis project. The *home directory* can change only by importing data from a different directory.

Consecutive or broken GPR sections can be joined (*concatenated*) into a single data set, provided that all data sets are in MGP or MAT format and have the same number of samples per trace, the same sampling rate and the same trace spacing. The operation also applies to unequally spaced data taken with instruments without survey wheels, or prior to marker interpolation. Marker trace information is preserved during concatenation. Finally, depth migration marks the end-point of a data processing line. At present, there's practically nothing more that can be done with matGPR that would be theoretically correct and would extract meaningful information. Accordingly, depth migrated data can only be saved to a binary file, for future use (e.g. concatenation or generation of 3-D data volumes for interpretation).

Data Visualization

The visualization of radargrams includes image (color-coded) displays with an assortment of color schemes, wiggle-trace displays and variable-area displays, as well as combined image/wiggle or image/ variable-area presentations. The mode of display is selectable through the Settings sub-menu. The Input Data is displayed in the *GPR Data* figure. The *Processed GPR Data* figure displays the Output Data; it is created whenever a processing step is taken and destroyed when the Output Data is held or discarded, or another processing step is taken. Figure 4 shows the Input Data in "image" display mode; examples of alternative display modes can be found in the matGPR Documentation.

matGPR provides a dedicated menu to change and manipulate the colors of "images", the *ImageColours*. The available color schemes include some of MATLAB's native "colormaps" like *grey, bone, jet* and *hot*. In addition, there are three custom color schemes dubbed *Blue – Red* (varying smoothly from navy blue to white and from white to brick red), *Brown – Black* (varying from brown to white and from white to black) and *Red – Black* (varying from brick red to white and from white to black). These are useful for emphasizing the higher amplitude reflections while masking low amplitude noise. The default colormap is jet (rainbow). It is also possible to manipulate the color schemes by adjusting the *Colour Saturation* (contrast) via a slider GUI, increasing/ decreasing brightness, reversing color order and interchanging color indices.

Another feature of matGPR's visualization complement is the *FigureTools* menu, which provides utilities for handling graphical objects. The MATLAB Figure Menu contains a large collection of utilities organized in tool bars; some of those may be useful in rare circumstances and some may actually not apply in the context of matGPR, for instance lighting a 2-D image. *FigureTools* provides a concise collection of the Figure Menu's tools and utilities that every user would like to have readily available while doing away with the rest. However, if one must access some feature of the Figure Menu (for

instance advanced annotation utilities), it is always possible to recover it from within *FigureTools*. The collection of graphical object handling tools provides for: Color scheme editing specific to the current figure (as opposed to the global color manipulation utilities available through *ImageColors*, zooming, panning, data inspection, exportation of the current figure in any of the formats supported by MATLAB, printing, and editing the current figure/ axes and their child objects.

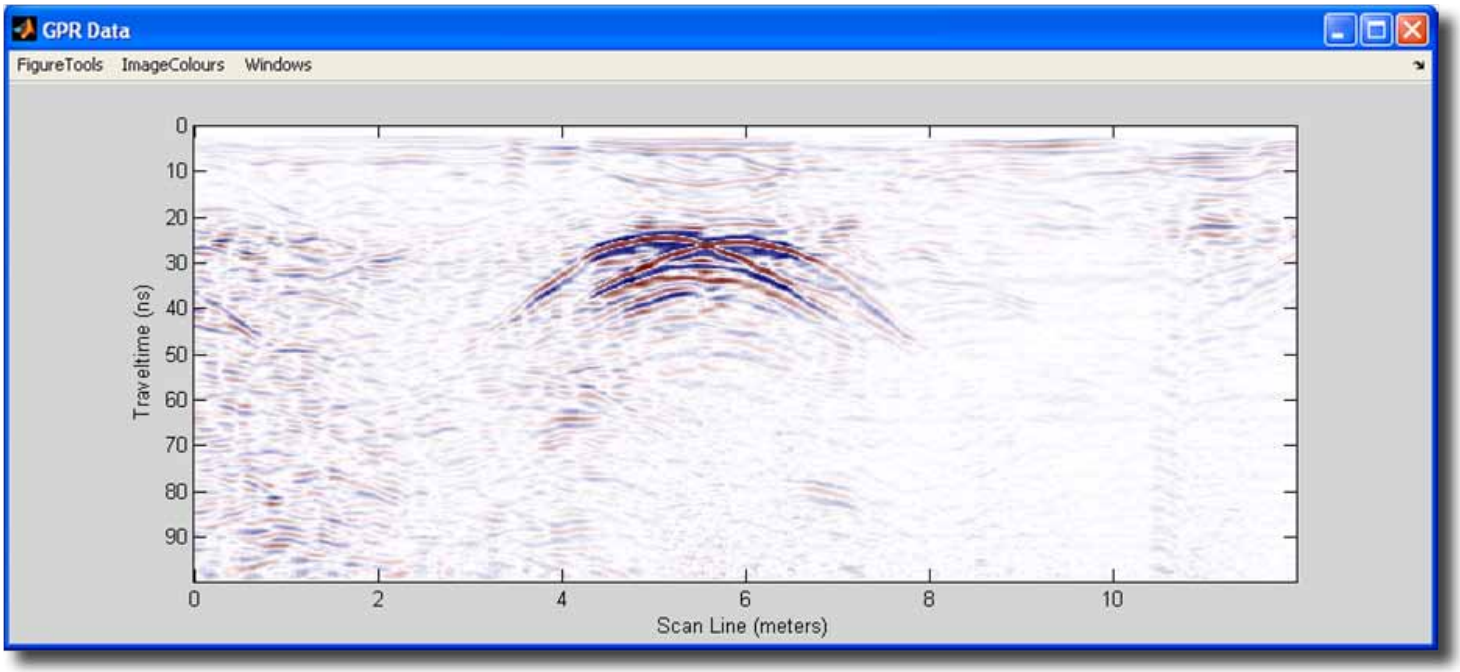


Figure 4. The GPR Data figure presenting the IPD in image display mode (Blue-Red color scheme). The OPD is displayed by an iden-tical figure.

Additional visualization tools include utilities to scrutinize the data. The Trace Viewer and the *Spectra Viewer* respectively facilitate inspection of the traces and Fourier amplitude spectra of the Input/ Output Data. A particular characteristic of matGPR is the *Attenuation Characteristics* viewer. The program computes the instantaneous power of all traces in the IPD or OPD data, hence the median and the mean power attenuation functions (median and mean instantaneous power). It also determines best fitting models for power-law and exponential attenuation based on the median attenuation function. Finally, it displays the attenuation functions and the best fitting attenuation models as per Figure 5. The result of this operation allows insight into the attenuation characteristics of the GPR signal, hence the properties of the medium; it also facilitates gain

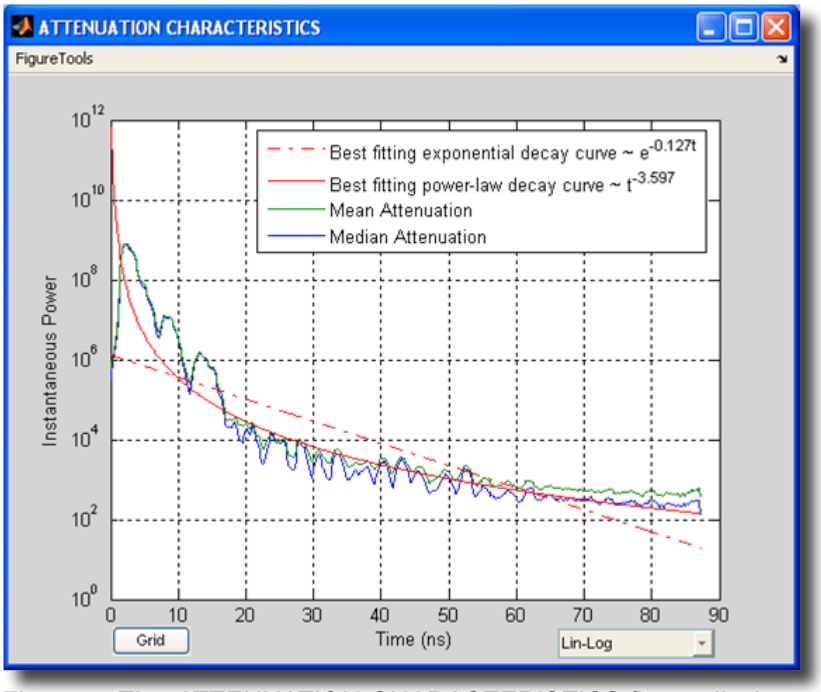


Figure 5. The *ATTENUATION CHARACTERISTICS* figure displays the median and mean power attenuation functions and the best fitting power-law and exponential decay models.

manipulations. Finally, it is possible to display the instantaneous attributes of the radargrams in the form of “images”. The instantaneous amplitude of the input sequence is the amplitude of the analytic signal. For GPR data it measures the reflectivity strength, reducing the appearance of random signal. Moreover, one may view the instantaneous phase angle of the Input Data, the unwrapped phase angle of the analytic signal and the instantaneous frequency (i.e. the time rate of change of the instantaneous phase angle).

Editing and Basic Processing

Basic data handling includes basic radargram editing (including resampling), gain manipulation, and, for GPR instruments not equipped with survey wheels, transformation of data collected at equal-time spacing mode to data at equal-distance spacing. The latter suite of functions also facilitates the three-dimensional positioning of traces with respect to some local co-ordinate system.

Trace Editing Utilities

Time-zero (or *Signal Position*) adjustment allows control of the instant of the surface reflection, i.e. the time when the radar pulse enters the subsurface (time zero). The ‘blank’ data between the beginning of a trace and time-zero is usually stored as part of the data. The post-acquisition determination and adjustment of time-zero amounts to a static correction and matGPR facilitates the operation via a modification of the *Trace Viewer* utility. It is also common for the later parts of the traces to contain useless noise occupying valuable memory; matGPR allows one to reduce the size of the radargram along the temporal dimension by discarding late-time arrivals (*Time-Window Trimming*). In combination with the *Signal Position* utility, this is also useful for extract-ing a certain part of the data for some specific purpose. Along the same vein, the *Edit Scan Axis* utility reduces the size of the radargram by discarding groups of traces at the beginning or at the end of the section’s scan axis (scanline). It is also used to extract groups of traces or even large parts of the section to a separate data set; a GUI facilitates the operation, as detailed in the matGPR Documentation.

Very often, the acquisition protocols oversample the data in both the temporal and spatial dimensions, sometimes by an excessive amount. In other cases, the sampling rate should be increased, for instance to unify data sets collected with different instruments and acquisition protocols. Since the original signal is always assumed to be band-limited to half the sampling rate, (otherwise aliasing would occur), Shannon’s sampling theorem says that the signal can be exactly and uniquely reconstructed for all time from its samples, by *band-limited interpolation*. The flexible and effective public-domain algorithm of Smith and Gosset (1984) takes care of this problem and allows sampling rate conversion by *any* rational factor.

GPR field systems frequently generate errant (bad) traces or groups of traces, for instance due to the antenna(e) coupling with local ground features or bouncing on small obstacles. It is possible to eliminate such traces using the *Remove Bad Traces* utility. The bad traces can be picked by pointing and clicking and are subsequently re-placed with interpolants computed from their near neighbors. Because this function uses interpolation, it works best for *isolated* bad traces, or small clusters of

traces; it is *not* recommended for removing extended groups of bad traces, as it will probably yield unreliable interpolants.

General Conditioning Utilities

These include removing the dc component (arithmetic mean) from each trace in the radargram (de-meaning), eliminating wow with zero-phase high pass filtering at exactly 2% of the Nyquist, and *Trace Equalization*. The latter makes the sum of the absolute values of all samples in a *reference* trace the *same* for all traces; a GUI allows selection of the reference, which can be the first trace, the mean trace, the median trace, any trace specified by the user or, finally, any positive number (base value).

Signal Amplification

matGPR offers four different data amplification methods, each with its own merits. *Automatic Gain Control* (AGC) is a process by which gain is automatically adjusted in a specified manner, as a function of a specified parameter, such as signal level. In matGPR, the time-varying signal level is measured by the RMS amplitude computed over a sliding time-window. By scaling the amplitude of the data at the centre of the window with respect to the RMS amplitude of the window, the process ensures that ranges of low amplitudes are emphasized with respect to ranges with high amplitudes. Two variants of AGC are provided: The *Standard AGC* where the window is a boxcar function and the *Gaussian-tapered AGC* where the boxcar is weighted (tapered) with a Gaussian bell function, whose breadth depends on noise level (input parameter). This emphasizes the contribution of the data around the centre of the window, thus producing a more focused result.

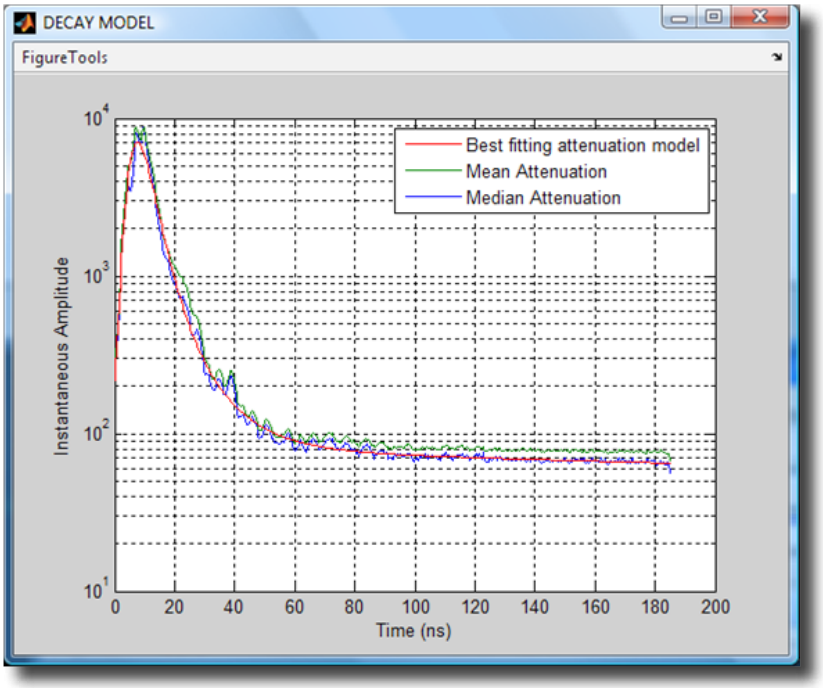


Figure 6. The result of the Inverse Amplitude Decay analysis.

The *Inverse Amplitude Decay* process applies an empirical gain function, which exactly compensates the mean or median amplitude attenuation observed in a GPR section. First the analytic signal for all traces is computed, whence the median and the mean instantaneous amplitude attenuation function is derived (Figure 6). The best fitting median and mean attenuation models are then computed using an exponential spectral function of the form $A(t) = c_1.\exp(-a_1 t) + c_2.\exp(-a_2 t) + \ldots + c_N.\exp(-a_3 t)$ with N linear parameters and N non-linear parameters. By means of a floating menu, the user may experiment with the *order N* of the fitting function and explore the merits of using the median or mean amplitude decay curve to obtain an optimal gain function. The gain function is the normalized inverse of the preferred amplitude decay model, $g(t) = [\, A(t) / \max\{A(t)\} \,]^{-1}$.

The *Inverse Power Decay* process applies a gain function of the form $g(t) = scale \times t^{power}$ and may also be used for the so-called spherical divergence correction. The user is asked to supply the power. Prior

to this, the program computes and displays the attenuation characteristics of the input data and a best fitting power-law attenuation model as per Figure 6. Then, it suggests a *power* which is the *exponent of the best-fitting model less 1*. This is usually a very good solution, but experimentation may provide a result more suitable for the data. The scale is computed automatically.

Experience shows that for zero-mean and time-zero adjusted data, the suggested *power* is around 2 and the observed attenuation is around 3. From a practical point of view, *power* ~ 3 applies little gain, or even attenuates the early (near surface) arrivals, while applying too much gain and emphasizing the late arrivals. On the other hand, *power* ~ 2 apportions higher gain at the early parts of the time window and lower gain at the late parts, providing for a more balanced eventual distribution of amplitudes. Claerbout (1996, pp. 222-223) gives a theoretical justification for using *power* = 2 in seismic data. He shows that the basic geometry of energy spreading predicts a single power of time for the spherical divergence correction and argues that an additional power arises from a simple absorption calculation: This justification has not been demonstrated in the case of GPR data; it is a theoretical exercise for the future, but nevertheless worth noting!

Marker Interpolation and Trace Positioning

For GPR instruments without survey wheels or other automatic triggering devices, matGPR provides a suite of marker interpolation utilities, so as to transform data collected in equal-time spacing mode, to data at equal-distance spacing. Key to this operation is the marker information matrix (MIM) stored in the IPD structure. On importing raw, unequally spaced data with Marker Traces, the MIM will comprise a single column vector with the sequential numbers of the Marker Traces. To be useful for interpolation, the MIM must be augmented to a 4 column matrix, so as to include the x-, y- and z-coordinates of the Marker Traces; this is done with utilities suitable for either regularly or irregularly spaced Marker Traces (details can be found in the matGPR Documentation). The MIM may also be exported/ imported separately to/from an ASCII disk file.

Once the MIM is prepared, (either in the IPD or on disk), it is possible to transform data collected at equal-time spacing mode, to data at equal-distance spacing using the *Interpolate to Equal Spacing* option of the *Basic Handling* menu. The interpolation routine uses the piecewise cubic Hermite polynomials method. The MIM is also the basis for generating the x, y and z coordinates of the radargram traces, thus including topographic information into the data set. Prerequisite is that the Input Data has equally-spaced traces. This is also done with interpolation and comprises the *Make X Y Z* utility of the *Basic Handling* menu. The output x, y and z coordinates are stored in the IPD structure.

Filtering and Advanced Processing

matGPR offers a variety of smoothing and processing facilities which can be generally classified in the broader categories of spatial filters, frequency and wavenumber filters and deconvolution filters. There's also a specialized Radon domain filter (τ-p filter). These utilities are collected under the *Filtering* menu of he matGPR GUI and will be briefly presented below.

Spatial and Low Dimensional Subspace Approximation Filters

The most rudimentary noise suppression utilities available in matGPR are *Mean* and *Median Smoothing Filters* in one and two dimensions. The filters are applied by sliding a given filter window over a 2-D

data wall. The data value corresponding to the central element of the window is substituted by the mean or median of the window data. Zero padding is applied; therefore, the edges of the output data are expected to be distorted. For the same reason, the size of the smoothing window should be kept reasonably small.

A second suite of classical and simple spatial filters, suitable for crude separation of data components with different spatial characteristics, involves *foreground/ background trace manipulation* and comprises:

- *Global Background Removal*. The global background trace is actually the mean trace determined by adding all traces together and dividing by the number of traces. This is a stacking process that reduces randomly varying signal (e.g. reflections from the subsurface) and enhances coherent signal, for instance horizontal banding (system noise). Caution is required when handling data with strong natural horizontal reflectors.

- *Horizontal Event Suppression*. The mean (background) trace of a sliding window is subtracted from the data in the window. This will eliminate small horizontal features (coherent event signal) and may be used to ex-pose reflections (events) that dip at high angles.

- *Dipping Event Suppression*. Conversely to the above, this removes the 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 (a dip filter), for instance to expose the sub-horizontal hydrogeologic features more clearly.

The Karhunen–Loeve (KL) transform (Karhunen, 1946; Loeve, 1955; Fukunaga, 1990), is a preferred method for approximating a set of vectors or images by a low dimensional subspace. In matGPR the approximating subspace is defined in terms of the first (largest) *N* singular values and associated singular vectors of the Input Data. The KL transform of the data is re-synthesized from the *N* largest singular values and vectors. The appropriate value of *N* should be determined by experiment. On output, matGPR displays the reconstructed model of the data (e.g. Figure 7 top), and the residuals after subtracting the model from the input data (Figure 7 bottom). When *N* is reasonably large (e.g. 3-10% of the shortest data dimension), the model is a smooth version of the input data, exhibiting enhanced lateral coherence of the GPR events. When *N* is very short, the model comprises only the most powerful principal components of the input data. In Figure 7

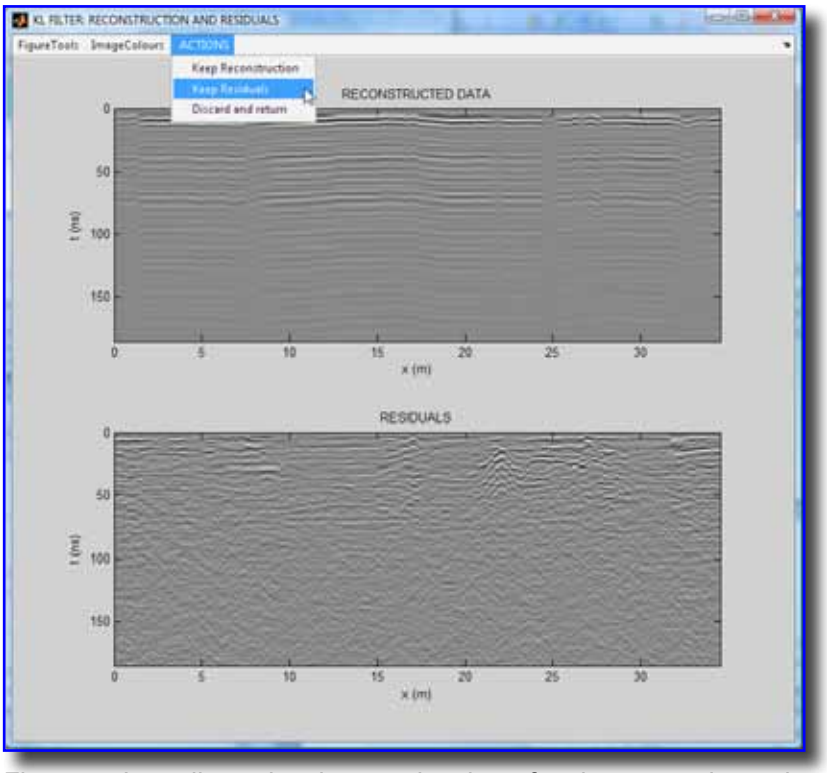


Figure 7. Low dimensional approximation of a data set using only the first 2 principal components. The top panel displays the reconstructed model. The bottom panel shows the residual of subtracting the model from the data.

N=2; the model contains only *strong* horizontal reflections and ringing and the residuals represent a version of the data free of powerful and unwanted interference.

Frequency and Wavenumber Filters

matGPR provides a suite of frequency domain (F), wavenumber domain (K) filters and frequency-wavenumber domain (F-K) filters. The F- and K- filters are identical, the only difference being in the sense they're applied (time-wise vs. scan axis-wise). The simple strategy employed in matGPR is to construct a very long FIR wavelet (at least 75% of the data length) in order to achieve very narrow transition bands, while eliminating Gibbs effects and ripple structure. The filter is applied in the frequency (wavenumber) domain and in a forward and a backward sense, so as to preserve phase information. All types of filters (low/ high pass, band pass and band stop) are available. There are three ways to design the filters:

1. *Graphically, on a test trace or scanline:* matGPR prompts the user to pick a test trace or scanline and displays its power spectrum as per Figure 8 (blue line). The cut-off frequencies are picked by pointing and clicking. The spectrum of the filter is then displayed as per Figure 8 (red line), together with the spectrum of the filtered data (green). The user may then apply or discard the filter and repeat the process.

2. *Graphically, on the mean trace:* The filter is designed based on the spectrum of the average trace or scanline computed from the Input Data, following the same procedure as above.

3. *By direct input* of the cut-off frequencies in an appropriate dialog box.

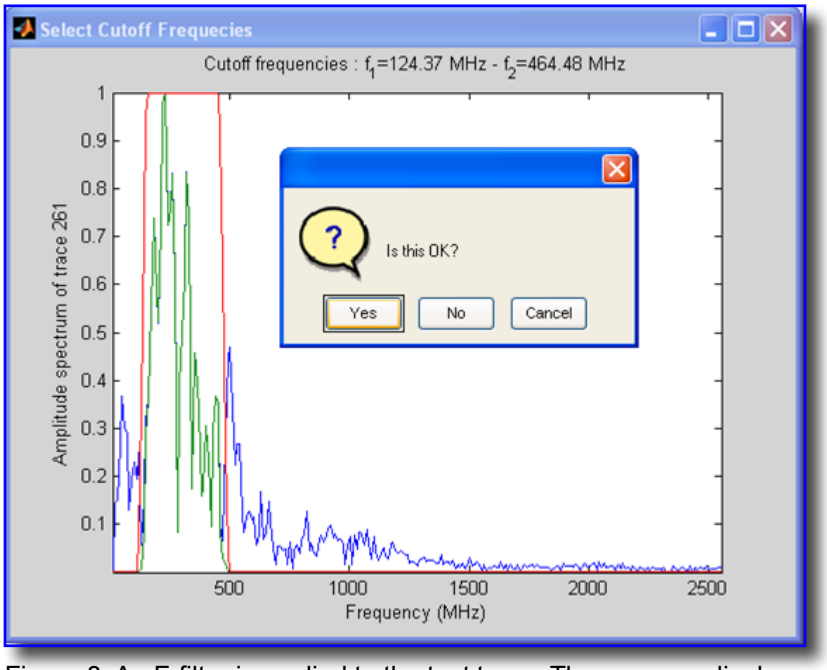


Figure 8. An F-filter is applied to the test trace. The program displays the result and awaits instructions.

The *F-K filters* may be zone-pass or zone-stop, fan-filters (velocity range pass/ stop) and up-dipping/ down-dipping event separators. The desired option is selected via a floating menu.

Zone-pass/ stop filters work like band-pass/ stop filters. The user is required to define a polygonal area (*zone*). Zone-pass operators retain the F-K spectral contributions *in* and *on* the sides of the zone; zone stop operators work the opposite way. There are two methods to define the zone, also selectable via a floating menu:

- Import the coordinates of the zone from an ASCII disk file (as specified in the matGPR Documentation).

- Set the coordinates graphically, on-screen: matGPR displays the log-amplitude of the F-K spectrum as per Figure 9 and the vertices of the polygonal zone are set by left-clicking at the desired (F, K)

position. To facilitate the process, the cursor coordinates are displayed at the bottom of the figure, the vertices are marked and the enclosed area is outlined with a transparent rubber band. Incorrect clicks can be undone by middle-clicking the mouse. Right-clicking completes the design and applies the filter.

Velocity range pass / stop provides for fan filtering in the F-K domain. The user must define the [upper, lower] negative and [lower, upper] positive apparent velocity limits in an appropriate dialog box. For *Velocity Range Pass*, the spectral contributions *inside* the fan will be retained and vice versa for *Velocity Range Stop*. The positive and negative velocity bounds do not need to be symmetric, but then caution is necessary because aliases may be introduced.

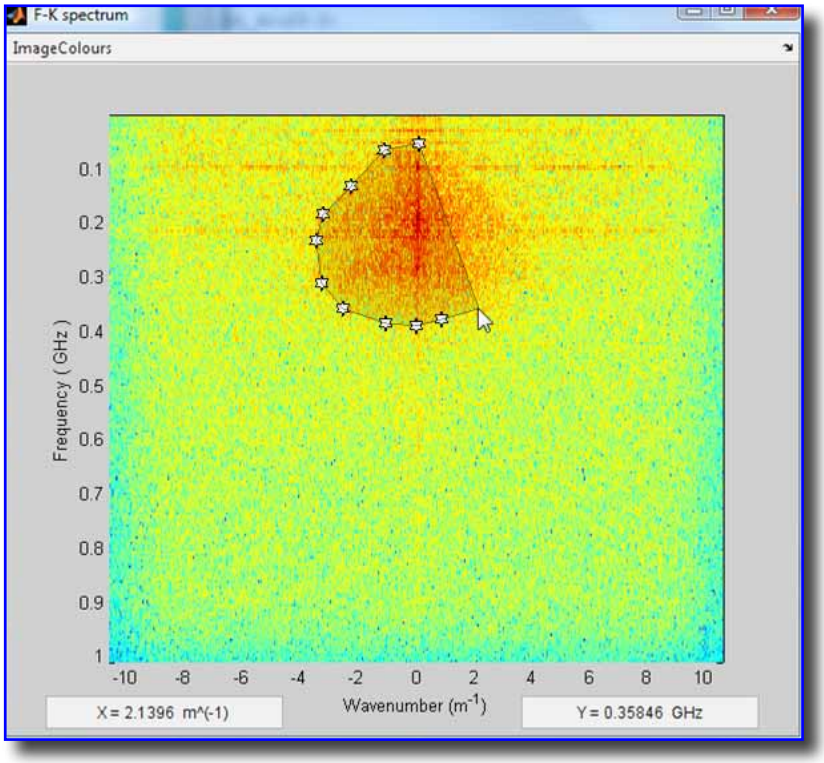


Figure 9. F-K filtering: Graphical assignment of a polygonal zone. This F-K spectrum belongs to the data of Figure 23.

Up/ Down-dipping event separation. The F–K transform has the property that reflections with positive slope (up-dipping) map into the positive-K half, while reflections with negative slope (down-dipping) map into the negative-K half. Therefore, by muting an entire half-domain, it is possible to reject reflections dipping in one direction or the other: the result is a section with features having unique dip directions.

Deconvolution Filters

F-X Deconvolution is a method of random noise cancelation in the frequency-space domain, first proposed by Canales (1984) and Gulunay (1986). The data is transformed into the F-X domain. For each frequency, Wiener filtering with unity prediction in *space* is used to model the data and the model is re-mapped onto the T-X domain. Credits for the F-X deconvolution routine go to M.D. Sacchi, University of Alberta; only minor changes have been effected for compliance with matGPR. For additional information see http://www-geo.phys.u-alberta.ca/saig/SeismicLab.

Predictive deconvolution is a very well known exploration seismology process, frequently used for the suppression of multiples or reverberations.

Sparse-spike deconvolution attempts to find an Earth reflectivity series r consistent with the observed data series y in the sense $y = f * r$ and exhibiting the *minimum* possible number of reflectors; f is a given source wavelet. By design, sparse deconvolution emphasizes the larger amplitudes and more important events in the observed data, while suppressing small amplitude or spurious events and noise. matGPR implements the procedure described by Sacchi (1997). On output, matGPR displays both the *model* $\hat{y} = f * r$ in which the more significant events have been retained and the recovered reflectivity series r. The user may then decide whether to use the model, or the reflectivity as outcome

of the operation. Credits for the sparse deconvolution routine go to M.D. Sacchi, University of Alberta, with minor changes effected for compliance with matGPR.

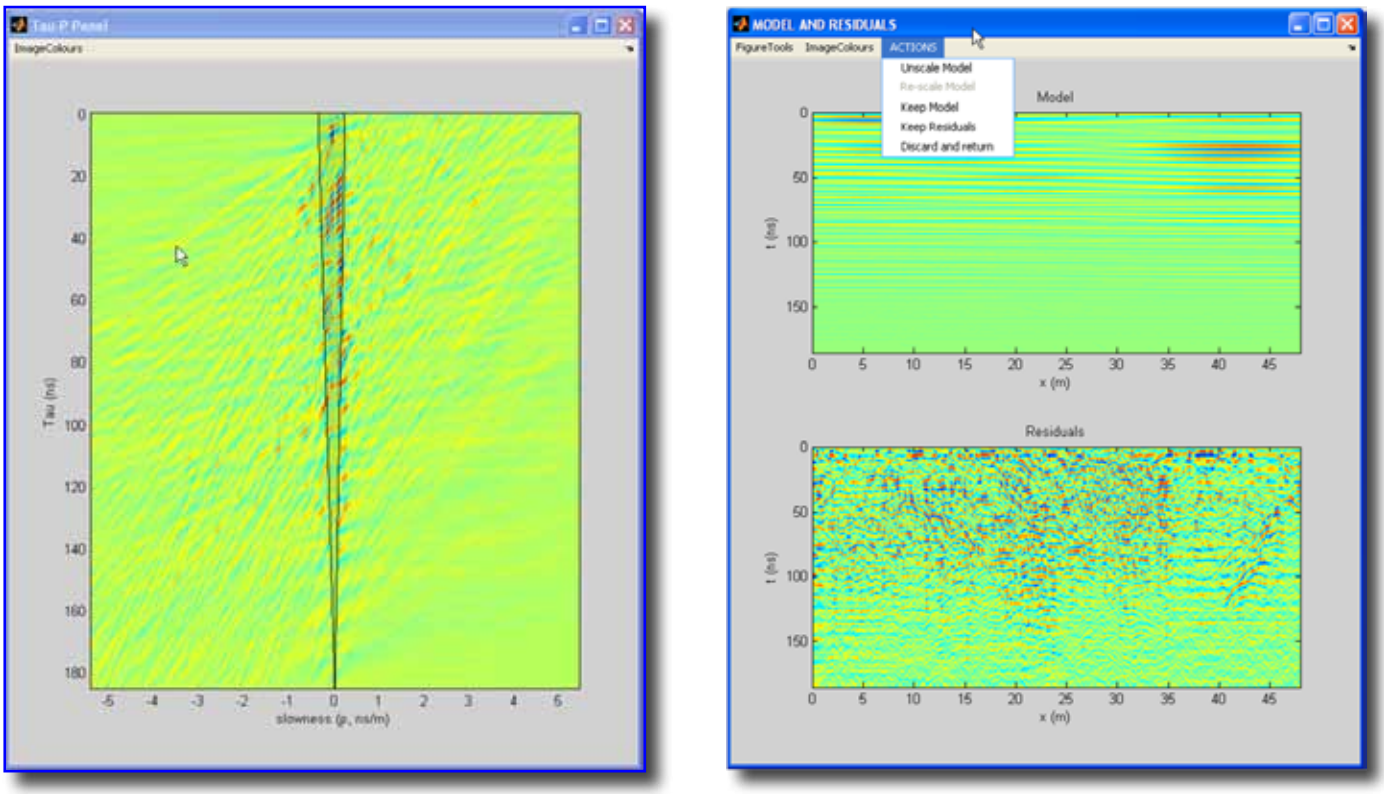


Figure 10. **Left**: The τ-*p* panel of the data shown in Figure 23. The shaded area represents the pass-zone containing the main contributions of the horizontal banding effects. **Right**: A model of the ringing computed by inverse transforming the τ-*p* elements within the pass zone (top) and the residual of subtracting the model from the data (bottom) 23.

Tau-p Filter

This is a simple application of Radon domain filtering. The radargram is transformed to the τ-*p* domain by integration in the T-X domain. A model of the data (noise) is extracted by muting τ-*p* domain contributions outside (inside) a pass (stop) polygonal zone and inverse transforming to the T-X domain also by integration. The resulting model (residuals) is taken to comprise the filtered data section. There are only two options, Zone-pass or Zone-stop filtering. As with F-K filters, there are two methods to define the zone: import the predefined coordinates from an ASCII disk file, or set the coordinates on-screen. Figure 10 shows the application of τ-*p* filtering to the data of Figure 23, in order to remove ringing effects (see §10 for details). In the left panel, the ringing is modeled in the τ-*p* domain by muting all contributions outside the shaded pass-zone. Upon inverse τ-*p* transformation, matGPR displays the model, (in this case the ringing effect), and the residuals (Figure 10 Right). By means of the ACTIONS menu, the user may decide whether to keep the model, or the residuals as the outcome of the τ-*p* filtering process. The residuals are more clearly seen in Figure 24 and are practically free of ringing.

Velocity Analysis and Imaging

matGPR offers a collection of imaging (migration) and modeling tools to assist in the interpretation. All these exercises require velocity models, which are constructed with utilities presented below.

Notably, some migration routines incorporate frequency dependence of the phase velocity due to the frequency dependence of the dielectric constant, (resistivity and magnetic permeability are assumed to be constant). The calculation of fre-quency dependence follows the analysis of Bano (1996). matGPR also provides a utility to manually calculate phase velocities, namely the *Simple Velocity Calculator* under the *Imaging* menu. It is probably worth noting that the one- and two-dimensional velocity models are stored in corresponding fields of a dedicated data struc-ture (VS), which is directly accessible via the MATLAB Workspace. The composition of VS is dully explained in the matGPR Documentation.

Velocity Analysis and Static Corrections

In matGPR, at present, *"velocity analysis"* implies the manual fitting of diffraction front hyperbolae based on the premise of non-dispersive propagation in a uniform halfspace and targets comprising point diffractors, or finite sized objects with quasi-circular cross section. The *Fit Diffraction Hyperbola* procedure (Figure 11) displays the Input Data in image display mode; parameters involved in the fitting process are the halfspace *Velocity*, the *Radius* of the target and the coordinates of its centre (*Depth* and *Location* along the scan axis). The parameters can be changed interactively, a) using the slider UI controls shown at the bottom of Figure 11, b) by clicking the *middle* mouse button at the *apex* of a diffraction hyperbola and turning the scroll-wheel to *automatically* adjust velocity and depth, or, c) using both methods above.

The calculated hyperbola is plotted on the data and the quality of fitting is determined by the eye, (depends on the experience of the user). Clicking on the DONE button causes the current value of the halfspace velocity to be assigned to the appropriate field of the velocity data structure, whence it is accessible by the imaging functions.

Static (topographic) corrections may be applied with the *Static Correction* option under the *Imaging* menu. The routine initializes a GUI, in which the necessary parameters are specified (ground velocity, sense of the correction (up or down) and elevation datum). The elevation data necessary for the corrections are provided through the IPD and should have been prepared with the *Make X Y Z* procedure (see §5).

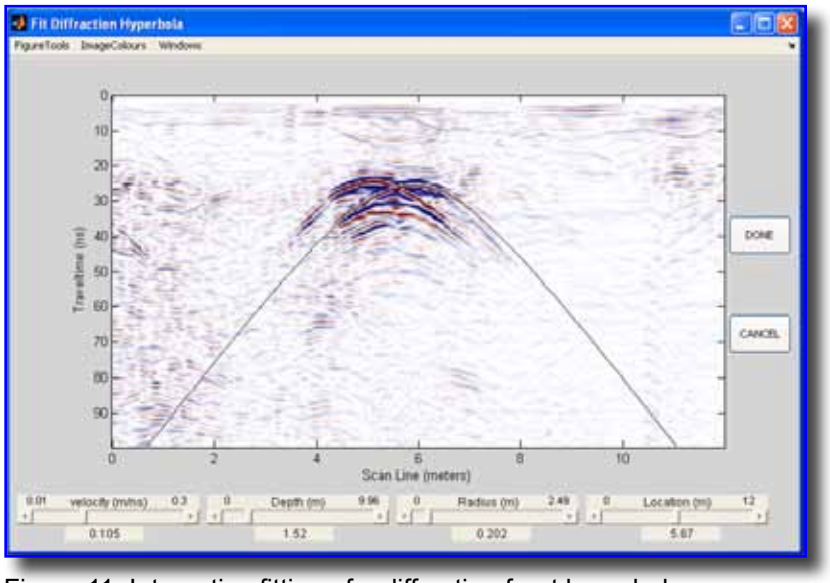


Figure 11. Interactive fitting of a diffraction front hyperbola.

1-D Time and Depth Migration

The layered velocity model necessary for the one-dimensional migrations may either be imported from a disk file (format specified in the matGPR Documentation), or supplied via an appropriate dialog box (the Get 1-D Velocity Model utility under the Imaging menu). The available methods are:

- Stolt's (1978) F-K migration for uniform or layered velocity structures (option *1-D FK Migration*). The routine uses Stolt stretching to account for non-uniform (layered) velocity structures and comprises

a very compact and fully vectorized code, fast enough to outperform compiled implementations of Stolt migration.

- Gazdag's (1978) phase-shifting migration for uniform or layered velocity structures (option *1-D Phase-shift Migration*).

- One-dimensional depth migration (option *Time-to-Depth Conversion*). The algorithm implemented in matGPR follows a rather complex procedure. First, it computes a time vs. depth function *t(z)* from the velocity vs. depth function *v(t)* and inverts *t(z)* to *z(t)* by inverse linear interpolation. Next, it remaps *z(t)* to depth *z* and, based on this function remaps a trace *y(t)* to a trace *y(z)*.

2-D Depth Migration

The two-dimensional velocity model necessary for the two-dimensional depth migrations may either be read from a disk file, or generated by the *Get 2-D Velocity Model* utility after importing a synthetic structural model previously prepared by the *Model Builder* (see §8.1). The available 2-D migration options are:

- Gazdag and Sguazzerro's (1984), Phase-shift Plus Interpolation (PSPI) migration for zero-offset data, with lateral velocity variation. The PSPI migration routine uses only the non-dispersive phase velocity term because, at present, there's no accurate method to interpolate the dispersive terms in the sense required by the Gazdag and Sguazzero algorithm. Thus, PSPI migration should be used when there's some confidence that ignoring frequency dependence wouldn't cause any harm.

- The 2-D Split-step Migration algorithm by Stoffa et al. (1990), for Earth structures with lateral velocity variation, augmented to account for the frequency dependence of the phase velocity (dispersion). The default for matGPR is to include dispersion. The Split-step migration routine can be used independently of matGPR, in which case dispersion may be excluded by user intervention.

Modeling

matGPR includes two modeling methods: the fast, adjoint split-step approach of Bitri and Grandjean (1998) and the slow but precise Finite Difference method of Irving and Knight (2006). The modeling suite is complemented with the *Model Builder*, a GUI utility enabling construction of 2-D velocity models for the depth migration methods discussed above and the forward modeling methods presented below.

Creating Models

Figure 12 shows the menus of the *Model Builder*. The *Data* menu provides I/O functions and the *Actions* menu the tools by which to build and edit a model.

The model comprises an ensemble of objects with polygonal or circular cross-sections. The first object to enter the list is a uniform *background* with dimensions equal to the size of the model. The *Builder* will also

Figure 12. *Model Builder* menus.

enquire its properties and antenna frequency, calculate the phase velocity and ask for confirmation. An affirmative response prepares the *Builder* for new input.

There are three ways to enter a polygonal object, depending on the *Input Mode*. a) Graphically, where the vertices of the polygonal object are set by pointing and left-clicking at the desired coordinates. To facilitate design, the cursor coordinates are displayed at the bottom of the figure, the vertices are marked and the enclosed area is outlined with a translucent rubber band (Figure 13 left). If a vertex is incorrectly set, it can be erased by clicking the *middle* button. Right-clicking terminates the procedure. b) Through dialog boxes and, c) from ASCII disk files. When all the coordinates have been set, the *Builder* will paint the object and will inquire its properties and tag (which is then printed at the centroid of the object). After a new polygonal object is inserted, the *Builder* searches and snaps together the vertices of all object found to lie within 1% of each other.

Similarly, there are two ways to enter a circular object: Through dialog boxes by typing the coordinates of the centre and the length of its radius, and graphically by pointing and left-clicking first at the centre and then at a distance equal to the radius. The *Builder* then calculates a polygonal approximation to the circumference paints it on the model and enquires for its properties and tag.

The objects can be overlapping and the order by which they are introduced in the structural model determines the way they will appear in the velocity model (or if they will appear at all). The program adopts a front to back hierarchy: foreground objects will appear whole and background objects will appear partially if masked by foreground objects. Figure 13 (right) shows a model comprising a “trench” dug in the background, with a water pipe at (7, 2.5) meters, and a metal pipe at (13, 2) meters, both parallel to the strike of the trench.

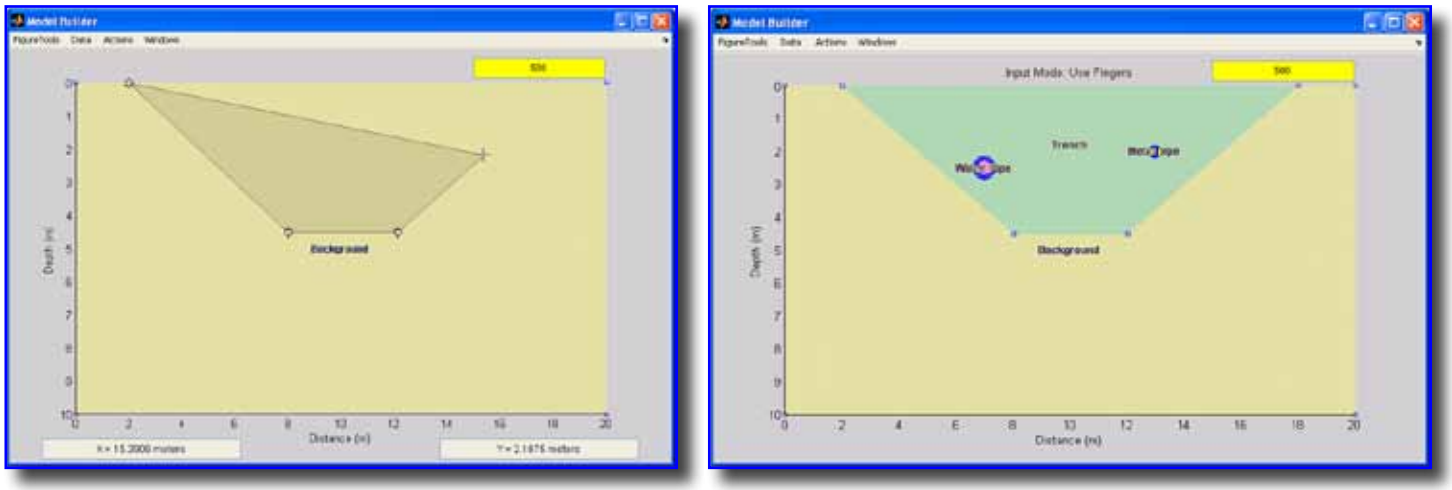


Figure 13. **Left:** Graphical insertion of a “trench” on top the structure. **Right:** Final model of the trench and two pipes.

Editing Models

The *Model Builder* allows extensive editing and modification of the model. The properties (dielectric constant, resistivity and magnetic permeability) of any object can be changed through *Review EM Properties* option of the Actions menu. The *Builder* projects a drop menu in which the object can be selected by its tag, (Figure 14), and then inquires the new properties through dialog boxes. Any object

can be removed from the model also by selecting its tag in a similar drop menu (the *Undo Shape* option).

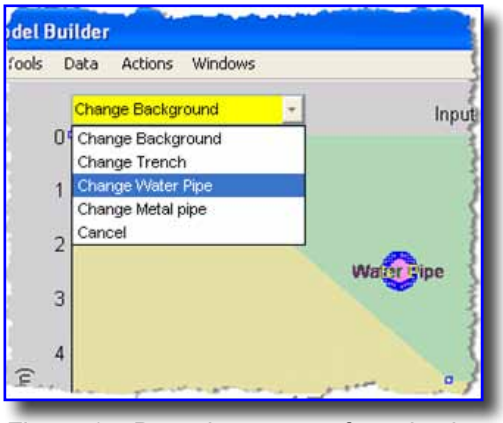


Figure 14. Drop down menu for selecting objects to edit their EM properties.

The geometry of polygonal objects can be changed by relocating (moving) or removing vertices and by inserting new vertices. This can be done either graphically, by pointing and clicking, or through dialogue boxes, depending on the *Input Mode*. Every time an object is modified, the *Builder* snaps together all vertices found to within 1% of each other. Modifications made in the geometry of an object may be reversed using the *Undo Changes* option. The model may be recolored at any time (the *Builder* assigns colors to objects *randomly* and sometimes the resulting scheme is dysfunctional). Finally, the *Actions* menu provides a set of functions to show or hide the vertex markers, show or hide tags and interchange tags with the corresponding EM properties.

The *Builder* saves the model for later access by itself, by the 2-D migration routines and by the Split-step and FDTD modeling programs. The format of the *model file* is free ASCII and is dully explained in the matGPR Documentation. It is simple enough and may be accessed/ modified independently, by any ASCII editor.

Split-Step 2-D Modeling

Synthetic GPR sections can be generated with the method of Bitri and Grandjean (1998). This is a fast and relatively dirty approach, lacking in detail (does not model secondary effects like multiples), and possibly prone to artifacts if applied indiscriminately. Conversely, it is efficient enough to effectively supplement the interpretation. The program will first ask for a model file prepared by the *Model Builder*. Based on the range of velocities it calculates from the model, it will compute and suggest a low, medium and high resolution grid for the synthetic radargram. It will also offer the choice of customizing the size of the synthetic radargram for particular requirements (e.g. simulation of observed data).

Experience shows that the high resolution synthetic is not necessarily the best choice, as it always comes at the expense of computing time. It does, however, exhibit the least amount of artifacts and aliasing. The low resolu-tion synthetic is computed quickly and is often acceptable, but depending on the model, it occasionally exhibits artifacts and aliasing. The medium resolution synthetic is often a good compromise. It should be noted that the dimension of the traveltime axis often tends to be overestimated and may be safely reduced through customization.

Finite Difference – Time Domain 2-D Modeling

Synthetic GPR sections may also be created with the TM-mode, Finite-Difference Time-domain method of Irving and Knight (2006). The program imports a model prepared by the *Model Builder* and constructs and displays an initial discretization of the model, as per Figure 15 (left). Based on this discretization, it computes and suggests *maximum permissible* values for trace spacing (*dx*) and depth spacing (*dz*) and requests confirmation/ modification of the parameters. The term “maximum permissible” refers to the values below which, aliasing effects should not appear! Based of the final values of *dx* and *dz* the final model grids are built. matGPR also requests the initial location of the source and trace (source) spacing

along the scan axis. Finally, it suggests values for the sampling interval and length of the traveltime vector and requests confirmation.

matGPR offers the option to view the simulated E_y wavefield propagating through the model in real time. It is also possible to make AVI movies of the simulation. If real time viewing is selected, the progress is shown as per Figure 15 (right). Otherwise the progress is shown in a wait bar which is continuously updated with time-to-completion information.

The FDTD solution of the forward problem can be *quite slow*, especially when large models and high frequencies are involved. In the latter case, the size of the problem may overflow the memory allocatable by MATLAB. Normal sized problems may take several hours, even days to complete and it is advisable to run trial simulations before committing the system to heavy number crunching. The size of the model is very important in this respect and it is always recommended to scale it down with increasing frequency.

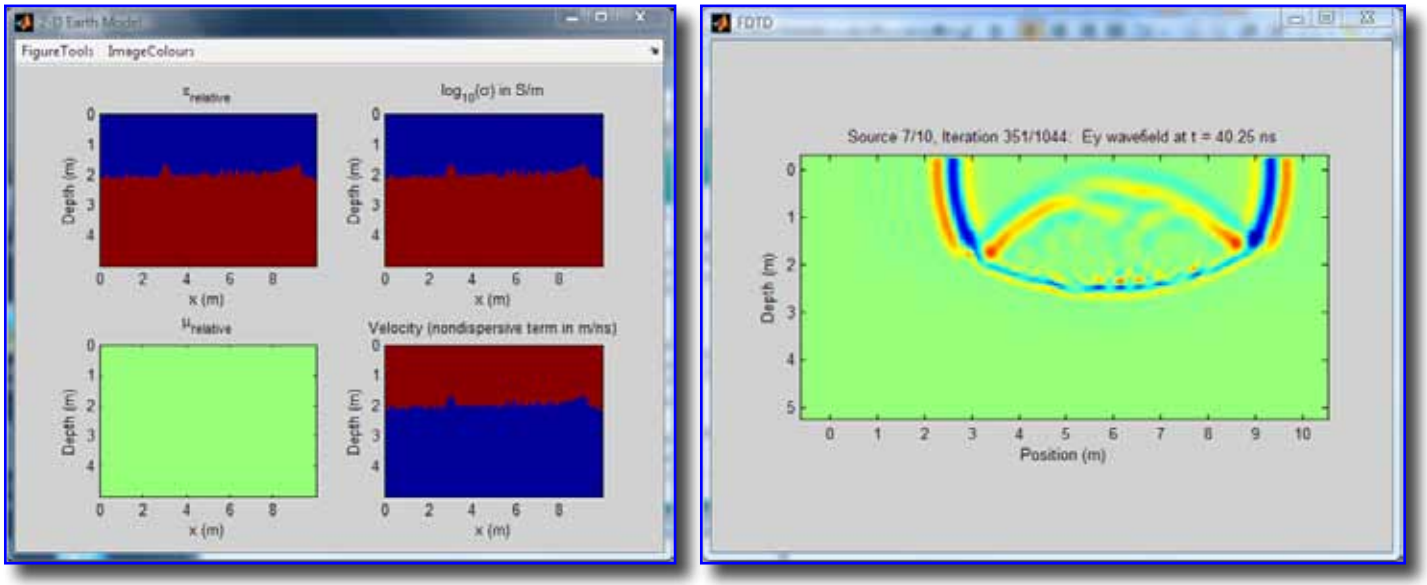


Figure 15. **Left:** Discretized material properties of the synthetic structural model (in this case an anomalous interface); dielectric constant (top-left), conductivity (top-right) and magnetic permeability (bottom-left); the non-dispersive term of the phase velocity is shown at the bottom-right panel. **Right:** The propagation of the Ey wavefield in the model can be displayed in real time.

Three-Dimensional Viewing

The presentation of matGPR will be completed with an overview of the 3-D visualization utilities, presently offered in the form of isometric surface and 3-D slice projections. These are only a first step toward assembling a more comprehensive set of 3-D visualization tools to facilitate the interpretation of complex data sets.

At present, 3-D data volumes may be generated from parallel 2-D radargrams; matGPR assumes that the data has been collected in a local co-ordinate system with the scan axis (longitudinal direction) parallel to the x-axis and perpendicular to the y-axis. The vertical axis can be either time or depth. The input profiles need not have the same lengths, or trace spacings, or traveltime vectors, or sampling rates, or depth vectors, or depth spacings; matGPR will homogenize the data with interpolation. Moreover,

the profiles need not be equally spaced along the y-axis. The only acceptable data formats are the matGPR-specific MGP-files and MAT-files. It is imperative that the x- and y- coordinates of all traces have been assigned prior to the generation of the 3-D volume (see the *Make XYZ* utility of the *Basic Handling* menu in §5). A dedicated GUI helps to assemble the necessary data files.

If one or more 2-D radargrams have different sampling rates or trace spacings, or depth spacings, they are all automatically resampled with respect to the median of the corresponding parameter. Finally, if one or more ra-dargrams have different traveltime or depth vectors, or scan-axis lengths, the user is asked to decide whether to trim the longer data sets to the length of the shortest, or to zero-pad the shorter data sets to the length of the longest. The 3-D Volumes can be exported to an M3D-file and (re) imported to continue an interpretation ses-sion.

Isosurface Displays

An *"isosurface display"* comprises the presentation of 3-D data in the form of orthographic projections of (iso-metric) surfaces with equal signal amplitudes, or less rigorously stated surfaces of equal reflectivity. A typical example is shown in Figure 17. The attributes of the display are controlled by the GUI shown in Figure 16, the *Iso-Surface Display Controls*, which enable the specification of: a) The signal amplitude to be displayed, via the *Iso-Surface Value* slider and editable box. b) The viewing angle (*Viewer Position* panel) – free rotation is also possible by activating the *Free Rotate* checkbox. c) The axes aspect ratio (*Aspect Ratio* panel). The default is *x:y:z* = 1:1:10 and may be changed either by means of the respective editable boxes, or gradually using the *increase* (▲) or *decrease* (▼) push buttons. d) The direction of lighting (*Light Position* panel) and, e) the color of the iso-surface with the *ReColor* push button.

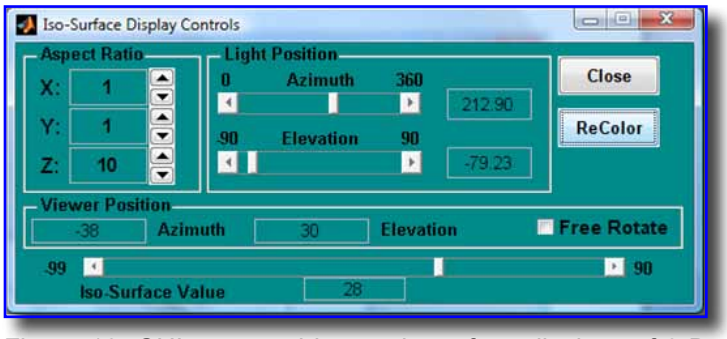


Figure 16. GUI to control isometric surface displays of 3-D GPR data volumes.

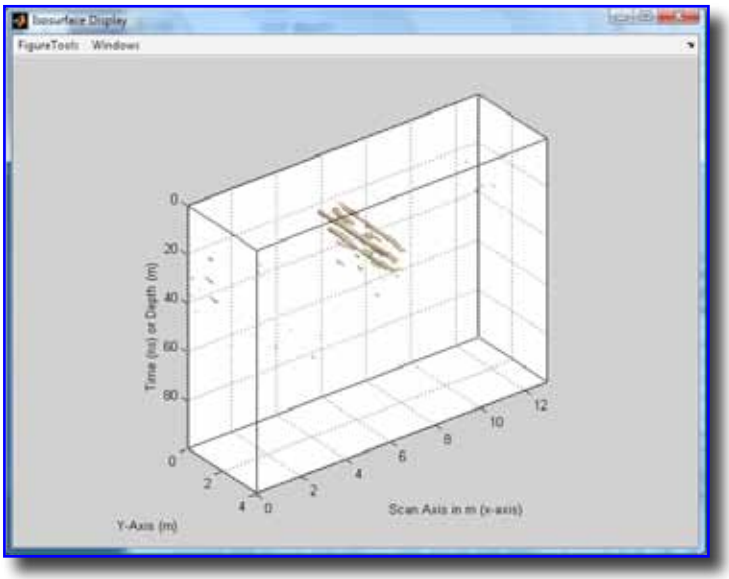


Figure 17. Equal signal amplitude display: Pipes buried at depths of 1.4–1.8 meters.

3-D Slice Displays

A *"Slice Display"* is an orthographic projection of the radargrams in the form of sections parallel to the scan-axis (x-axis), as in Figure 19. The slices are initially opaque. The display is controlled by the GUI of Figure 18 (*3-D Display Controls*), which is organized in panels as follows:

The *Y-slices* panel determines the number of slices displayed and their transparency. The number of slices is controlled via a slider and/or an editable box, both labeled *Display Y-slices*, according to their position with re-spect to the y-axis. By default, all Y-slices, i.e. all the profiles comprising the

data volume are shown on startup, arranged from last to first. Translucence is enabled/ disabled via the *Translucence On/Off* box. The trans-parency may be changed via the *increase* (▲) or *decrease* (▼) push buttons. The method by which to introduce translucence in the display is credited to Conroy and Radzevicius (2003).

The Slice Display can be augmented with addition of X- and Z-slices, one of each at a time. The position of these slices can be controlled with the sliders and editable boxes of the *X, Z slices* panel. X- and Z- slices are not displayed on start-up, but may be introduced later, either by moving the respective sliders to the appropriate position, or by typing their position in meters, in the respective editable boxes. Figure 20 shows the same data as per Figure 19, but with increased transparency, an X-slice at 23m along the scan axis (where a buried wall was detected), and a Z-slice at 38ns along the traveltime axis, where there are intense reflections from a buried inter-face.

Color Saturation (contrast) can be varied with an appropriate slider. In combination with translucence, increas-ing the contrast allows many features to stand out clearly. The aspect ratio of the axes can be controlled as above, while the viewing angle may be changed with the *Azimuth* and *Elevation* sliders and editable boxes (*Viewer Position* panel).

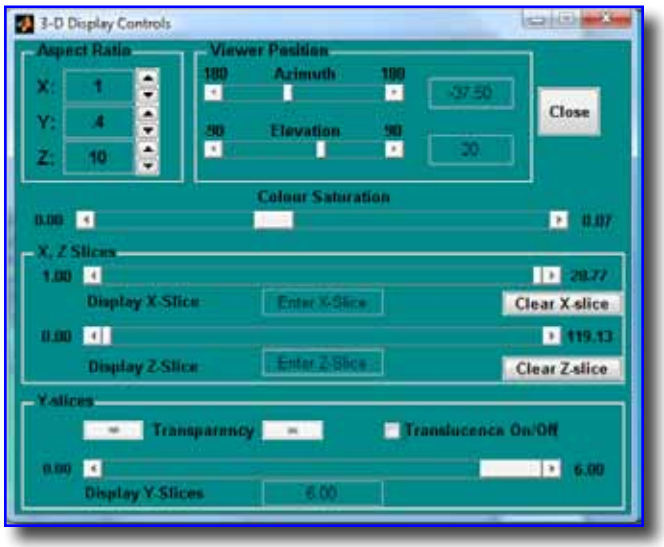


Figure 18. GUI to control 3-D slice presentations of GPR data volumes.

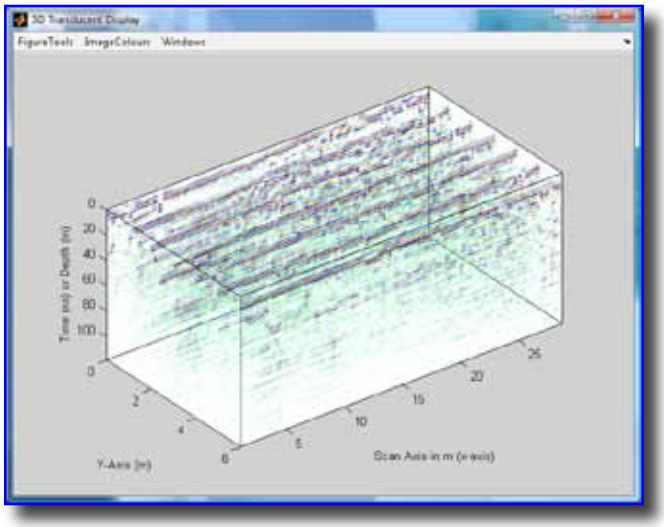


Figure 19. 3-D Slice Display with translucence.

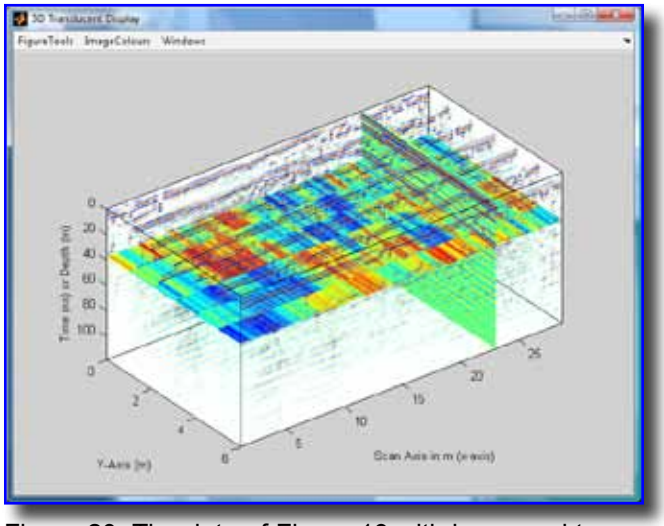


Figure 20. The data of Figure 19 with increased trans-parency, an X-slice and a Y-slice.

An Example

As should already be apparent, matGPR particularly emphasizes on analysis and tries to include advanced and effective processing and interpretation methods to handle normal, as well as complicated data sets. In order to assert this point, and because no statement or description is good without examples, this presentation concludes with the analysis of a not-so-usual radargram.

The data was collected on a ridge (Mt Ktenias, altitude ~1600m, NE Peloponnesus, Greece), where high-rising wind-powered electricity generators were to be erected, as part of a geotechnical survey to investigate the conditions and consistency of the foundation

ground (e.g. to detect voids). The geological setting is shown in Figure 21. The ground comprises dipping, thin-plated limestone with intercalations of argillaceous material, intensely fragmented and karstified (see insets in Figure 21). As evident in the photograph, many cracks and voids are filled with lateritic soil with a considerable argillaceous component.

Figure 21. The geological setting of the Ktenias ridge. The inset pictures show the type of karstic voids and cracks abundant in the subsurface. Photos by courtesy of Mr Pavlos Sotiropoulos, Terramentor EEIG.

The raw data was collected with a Måla GPR system and 250MHz antenna. Figure 22 shows the example section after time-zero adjustment, global background removal and AGC with a 20ns Gaussian-tapered window. The later parts of the data traces, (after approx. 80 ns), are infested with apparently random noise which is amplified by the AGC and masks other useful reflections. In addition, is apparent that the data suffer from a considerable degree of ringing, presumably due moist, near surface thin argillaceous layer(s).

The random noise cannot be treated with spatial smoothing filters effectively, because the medium contains many distributed targets whose reflections will be smeared in the process. Figure 23 is the data after F-K, zone-pass filtering; the F-K filter was preferred because as is apparent in the F-K spectrum, there are strong, frequency-local, laterally propagating noise components, presumably due

to arrivals from dipping or sub-vertical scattering surfaces (e.g. cracks); it is more efficient to handle such noise simultaneously in the F-K domain, that separately in the F- or K- domains.

Elimination of the random noise shows that the ringing is actually quite strong, certainly stronger than what it appeared to be, and that at places interferes with reflections from cracks and voids. Elsewhere, it appears to be absent, especially at places with a high density of reflection events, presumably due to the disturbance/ destruc-tion of the conductive layer(s) by the formation of cracks and voids. The uncertainty introduced by the interfe-rence significantly complicates interpretation.

In this case, the ringing shown in Figure 23 can be modeled by separating its τ-*p* domain contributions, as discussed in section *Tau-p Filter* above and Figure 10. The residual of this exercise are shown in Figure 24 and is apparently free of the ringing effect. Notably, the same result could be obtained with Karhunen-Loeve transformation; in this approach, a representative model of the ringing can be reconstructed using the first two principal components (N=2).

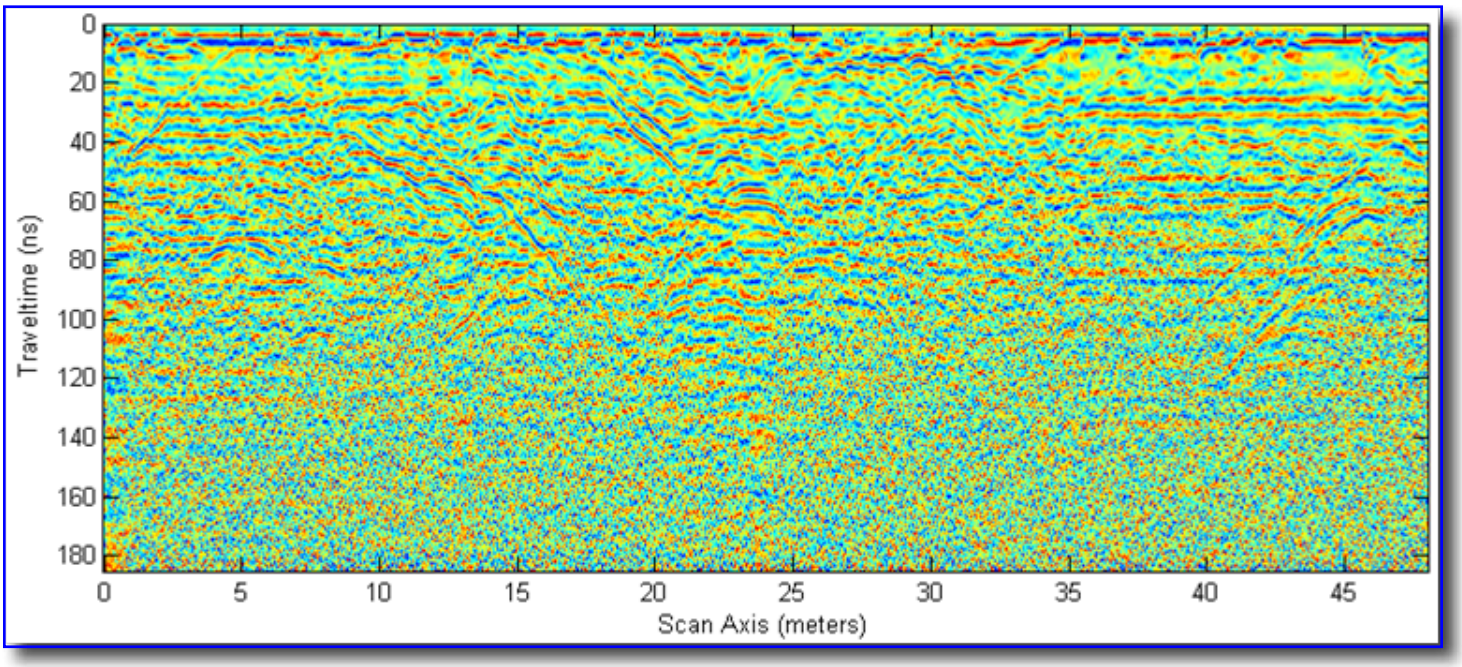


Figure 22. A GPR section collected above fragmented and karstified limestone at the location shown in Figure 21, after preprocessing (time-zero adjustment, background removal and amplification).

There are several diffraction hyperbolae in the section, to allow velocity analysis at different locations and depths; these can be seen more clearly in gray scale but are not included herein for brevity. The fitting process is not shown also for brevity, but the results indicate a small lateral velocity variation from ~0.085 m/ns to ~0.095 m/ns. It is not exactly clear why this should be, but it may be related to an inactive fault, which in Figure 21 can be seen just behind the radar operators, dipping from left to right between meters 22 and 32 of the section and bringing together strata with slightly different lithological characteristics. The karst and voids appear to have developed in association with this fault.

The resulting 2-D velocity model is simple enough, as to not justify the space required for its presentation. On the other hand, the application of split-step depth migration with this velocity model produces the result of Figure 25, in which the intricate structure of the foundation ground comes clearly in focus. It is almost straightforward to observe subvertical faults and fractures possibly filled with conductive

(e.g. lateritic) material and at places intersecting, (e.g. around coordinates x=22.5, z=3.5), point-like reflections corresponding to small targets like (filled or empty) voids and a dipping interface (the bedding is clearly observable in Figure 21) approximately between coordinates (x=5, z=1) and (x=30, z=4.5).

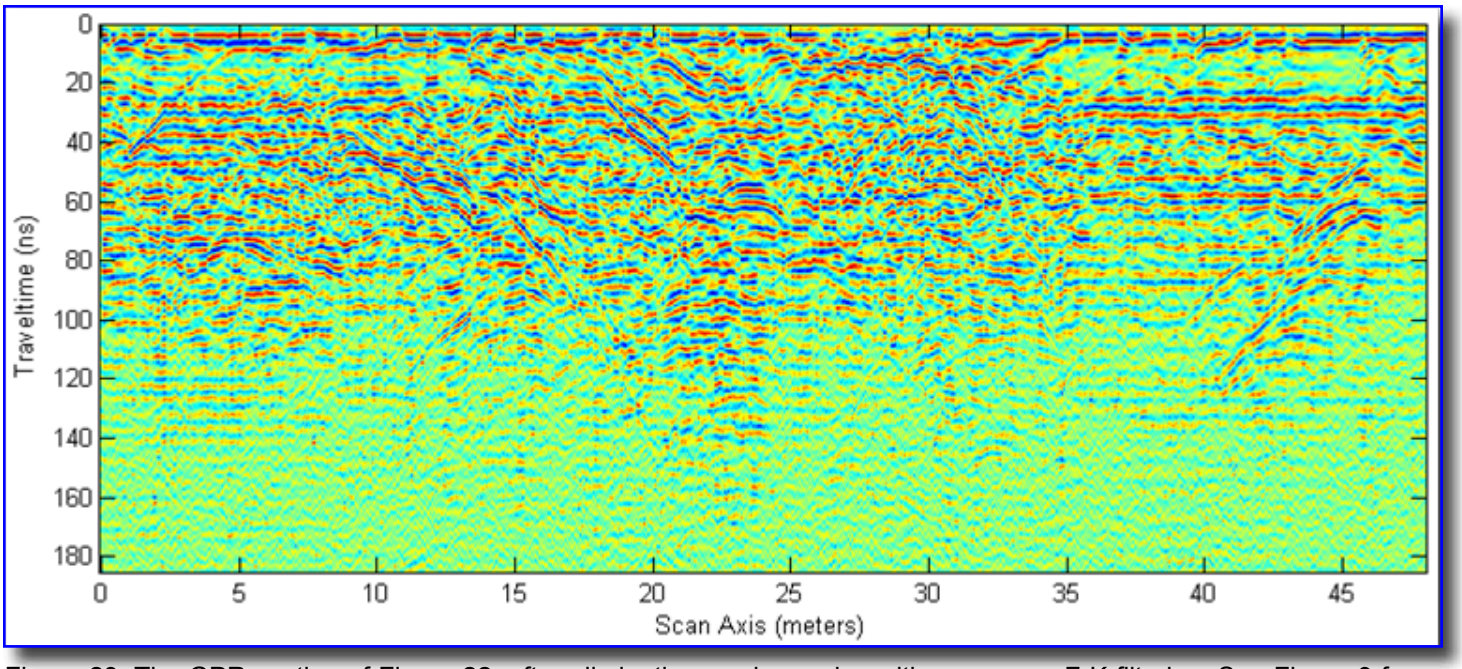


Figure 23. The GPR section of Figure 22, after eliminating random noise with zone-pass F-K filtering. See Figure 9 for a map of the F-K spectrum and the shape of the pass zone.

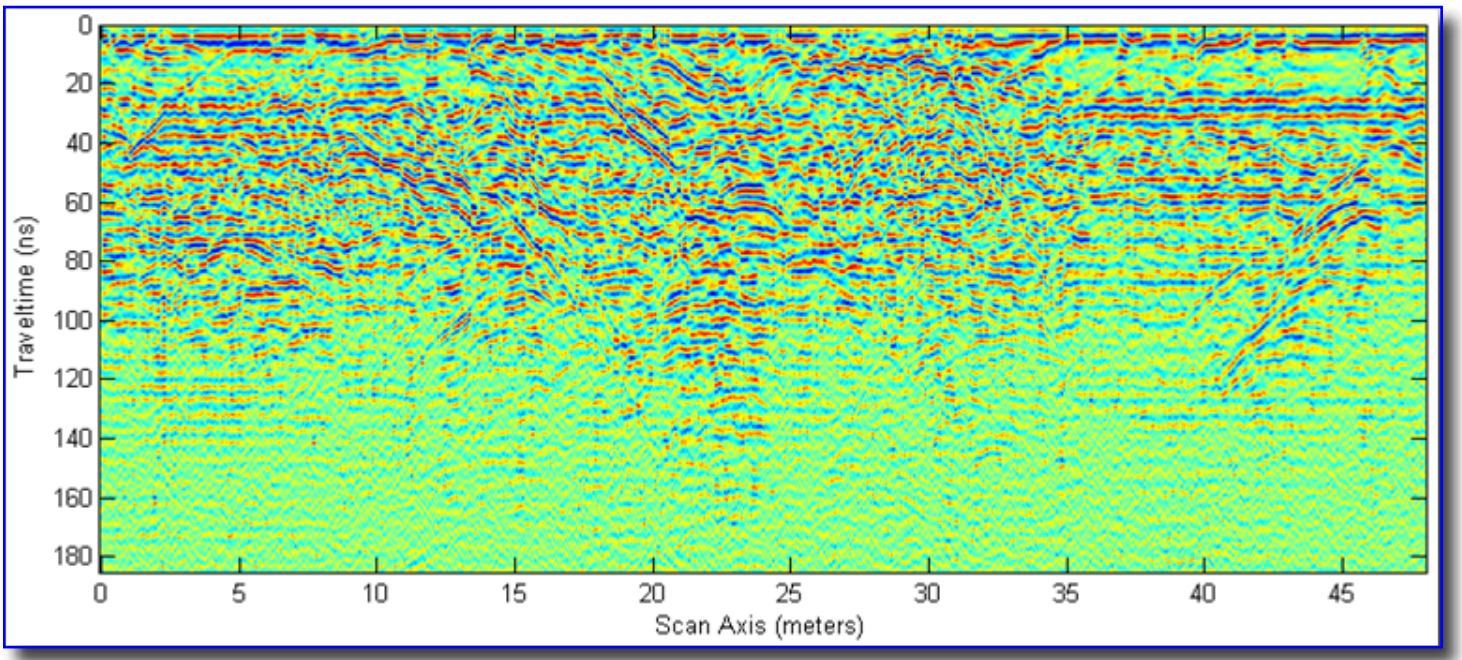


Figure 23. The GPR section of Figure 22, after eliminating random noise with zone-pass F-K filtering. See Figure 9 for a map of the F-K spectrum and the shape of the pass zone.

Epilogue

As demonstrated in the hitherto description, matGPR puts emphasis on data analysis and provides a relatively broad and functional range of analytical methods and procedures for the analysis of zero

and single-offset GPR data. Its toolboxes enable the processing of "normal" and, more importantly, complicated and difficult data sets. For the same reason it can be used for education, albeit at a relatively advanced level (in Athens, we use a stand-alone educational version of Release 1). Moreover, special provisions were made to include in-house solutions for such algorithms and analysis techniques that are necessary but not supplied with the MATLAB core and can only be found in toolboxes. Thus, matGPR can be used by all those in possession of a basic MATLAB license.

On the downside, there are limits to backward compatibility, in as much as MATLAB evolves rapidly and continuously. matGPR is programmed to detect the MATLAB version and does not invoke functions

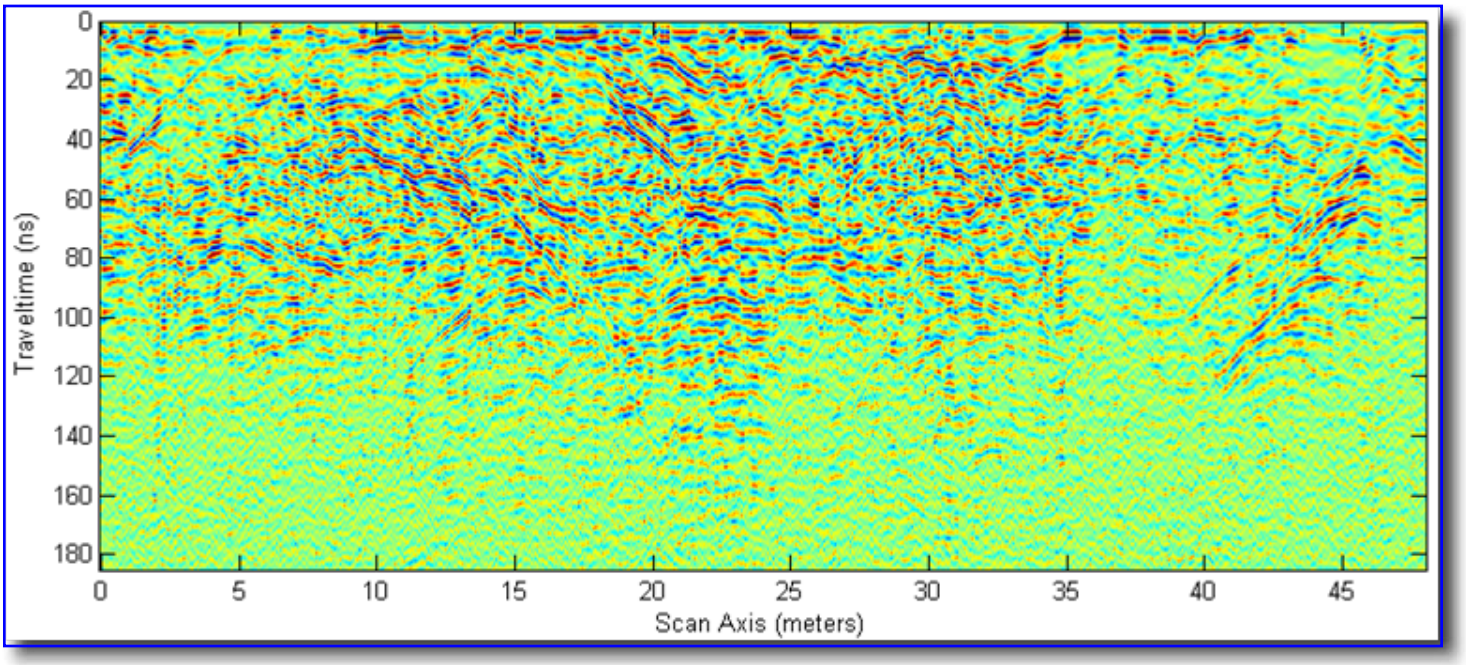


Figure 24. The GPR section of Figure 23 after removal of random noise with τ-p filtering. See Figure 10 for a map of the τ-p domain and the shape of the pass zone.

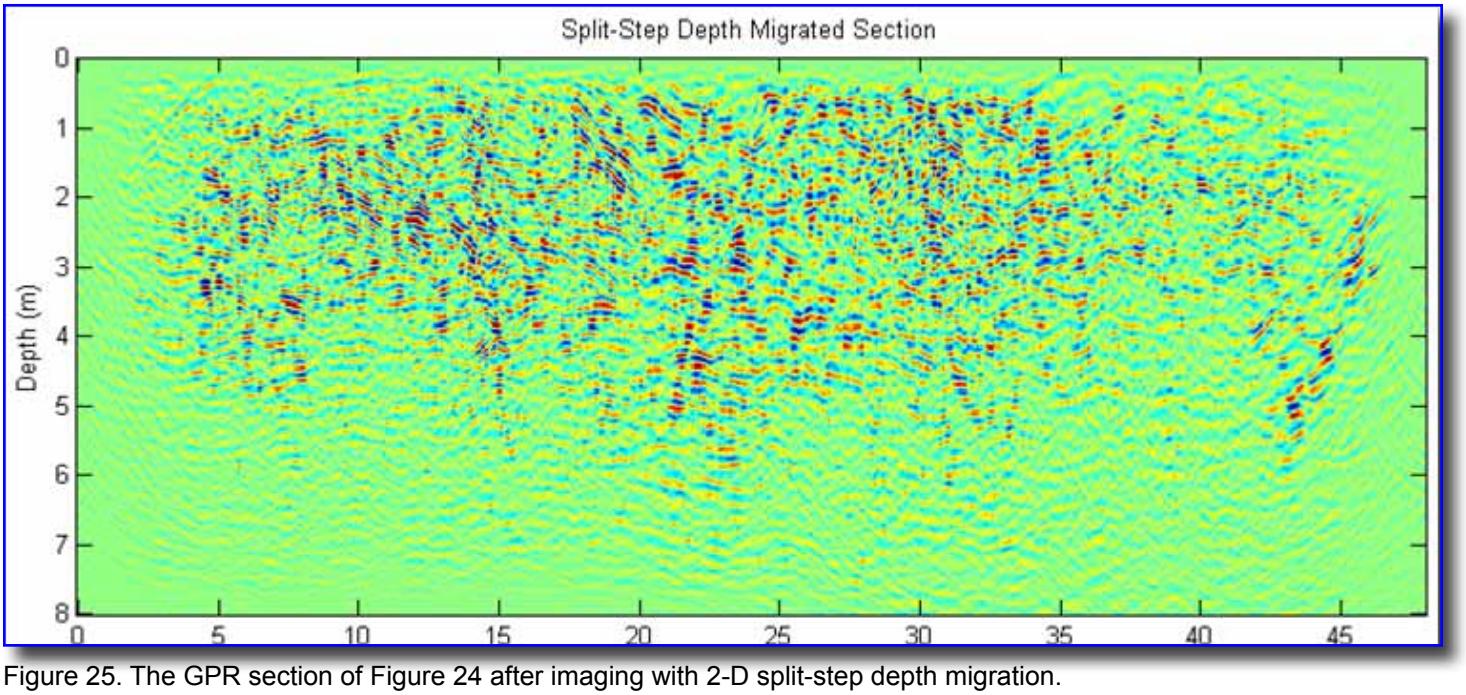


Figure 25. The GPR section of Figure 24 after imaging with 2-D split-step depth migration.

and procedures from a specific version, but executes alternative in-house solutions. The point of no return, however, has to do with the implementation of data constructs and procedures like *structures* and *cell arrays*, *function handle callbacks* etc. Thus, MATLAB V4 is *altogether out* of the picture. Also note that matGPR has never been tested with MATLAB V5.x and is not expected to work to anyone's satisfaction. It will cooperate with MATLAB V6.5, but probably not seamlessly. It should, however, cooperate seamlessly with MATLAB V7.1 – 7.3 and is fully functional with V7.4 and above.

At the present stage of development matGPR has a rather broad scope of application and is also versatile in that it is expandable and customizable to the needs of its users, provided of course they have MATLAB programming skills. Nevertheless, one can think a multitude of analysis tools that can be included in future releases. The author hopes that matGPR has the potential to grow and expand over time, welcoming contributions from other researchers in the spirit of the GNU project. At present, matGPR can be obtained from *http://users.uoa.gr/~atzanis/matgpr/matgpr.html*.

Acknowledgments

The author acknowledges limited support by the "Kapodistrias Programme" of the Research Secretariat, National and Kapodistrian University of Athens. Limited support was also provided by the European Social Fund and Greek National Resources through program EPEAEK II – PYTHAGORAS II, "Support for Research Groups in Universities". The example data from Ktenias Ridge and the photographs in Figure 21 are courtesy of Mr Pavlos Sotiropoulos, Terramentor EEIG, Athens, Greece (*http://www.terramentor.com*).

References

Bano, M., 1996. Constant dielectric losses of ground-penetrating radar waves, Geophysical Journal Int., 124, 279-288.

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.

Canales, L. L., 1984, Random noise reduction: 54th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session:S10.1.

Claerbout, J., 1996, Imaging the Earth's Interior, Network Distribution Version, pp. 50-57 and 222-223; http://sepwww.stanford.edu/sep/prof/index.html.

Conroy, J. P. and Radzevicius, S. J., 2003. Compact MATLAB code for displaying 3D GPR data with translu-cence, Computers and Geosciences, 29/5, 679-681.

Crochiere, R and Rabiner, L. R., 1983. Multirate Digital Signal Processing, Englewood Cliffs, NJ: Prentice-Hall, Inc.

Fukunaga, K., 1990, "Introduction to Statistical Recognition", Academic Press.

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

Gazdag, J. and Sguazzero, P., 1984. Migration of seismic data by phase-shift plus interpolation, Geophysics, 49, 124-131.

Grandjean, G. and Durand, H., 1999. Radar Unix: a complete package for GPR data processing,

Computers & Geosciences, 25 141-149.

Gulunay, N., 1986, FX deconvolution and complex Wiener prediction filter: 56th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, Session:POS2.10.

Irving, J. and Knight, R., 2006. Numerical modeling of ground-penetrating radar in 2-D using MATLAB, Computers & Geosciences, 32, 1247–1258.

Karhunen, K., 1946, “Zur Spektraltheorie Stochastischer Prozesse”, Ann. Acad. Sci. Fennicae, 372.

Loeve, M.M., 1955, “Probability Theory”, Princeton, N.J.: VanNostrand.

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

Sacchi, M.D., 1997. Re-weighting strategies in seismic deconvolution, Geophysical Journal International, 129, 651-656.

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 “Proceedings of the International Conference on Acoustics, Speech, and Signal Processing” ICASSP-84, Volume II, pp. 19.4.1-19.4.2, New York: IEEE Press. Also, see expanded tutorial and associated free software at the Digital Audio Resampling Home Page: http://www-ccrma.stanford.edu/~jos/resample/

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.