FuncLab: MATLAB receiver function analysis toolbox Version: 1.8.0

Author: Robert W. Porritt

User Manual

Email: rwporritt@gmail.com

Contents

1. ABOUT FUNCLAB

2. DOWNLOAD AND SETUP

3. RECEIVER FUNCTION OVERVIEW, PREPARATION, AND DATA DIRECTORY STRUCTURE

    3.1 RECEIVER FUNCTION OVERVIEW

    3.2 RECEIVER FUNCTION FORMAT AND PREPARATION

    3.3 RECEIVER FUNCTION RAW DATA DIRECTORY STRUCTURE

4. GRAPHICAL USER INTERFACE

    4.1 STARTING A NEW PROJECT FROM EXISTING RF DATA

    4.2 STARTING A NEW PROJECT FROM irisFetch.m

    4.3 COMPUTE NEW RFS FROM WAVEFORM DATA

    4.4 LOADING AN EXISTING PROJECT

    4.5 PROJECT.MAT FILE AND FUNCLAB VARIABLES

    4.6 MAIN FUNCLAB GUI

    4.7 TRACE EDITING

    4.8 VIEWING DATA (PLOTS AND VISUALIZATIONS)

        4.8.1 VIEWING SEISMOGRAMS

4.8.2 VIEWING STATION/EVENT SUMMARY PLOTS

4.8.3 VIEWING CROSS SECTIONS

4.8.4 VIEWING STATION/EVENT MAPS

4.9 COMPUTE MENU

4.10 SUBSETS MENU

4.11 EXPORT MENU

4.12 CITATIONS MENU

5. PROJECT DIRECTORY STRUCTURE AND DESCRIPTION

6. MODEL BUILDING AND SETUP

6.1 VELOCITY MODELS

6.2 TOMOGRAPHY MODELS

7. ADD-ON PACKAGES

8. REFERENCES

APPENDIX A. WEBSITE REFERENCES

APPENDIX B. DESCRIPTION OF SAC FILES

Updated: July 17, 2017

1. About FuncLab

FuncLab comprises a set of tools built within the MATLAB environment to analyze receiver functions (*Eagar and Fouch*, 2011). Receiver functions are seismic time series computed by deconvolving the radial from the vertical seismogram to isolate the Earth's radial *S* wave response. While *P*-to-*S* conversions arrive later relative to the *P* wave, *S*-to-*P* conversions arrive earlier relative to the *S* wave. Both mode conversions can be utilized in receiver functions for different purposes. In this first release of FuncLab, tools are available for *P*-to-*S* receiver function analysis. We note that *S*-to-*P* receiver function analysis could be incorporated via a community-developed add-on.

These tools were developed on the Mac OS X platform beginning in 2006 with MATLAB 7.1.0.21 (R14) student version. Most of the developed tools were created to be compatible with this version, but were not tested to be backwards compatible with earlier versions of MATLAB. Figure dimensions may also vary from platform to platform, but reflect development on Red Hat Linux 5.3 running MATLAB 7.9.0 (R2009b). Testing has also been conducted on Mac OS X 10.6.7 running MATLAB 7.10.0 (R2010a) without problems. The most recent development has been on Mac OS X 10.12.5 and MATLAB 2017a. Filesystem differences in Windows platforms may cause some problems.

There are some tools that are used within the toolkit that were not originally developed with FuncLab, but are borrowed from free open-source forums on the web. They are:

1. *ignoreNaN.m*: used to handle NaN values within matrices

2. *makeColorMap.m*: used to adjust color maps in figures

3. *mmpolar.m*: creates a polar plot of ray parameter vs. backazimuth

Additionally, several more expansive toolboxes have been incorporated including:

1. irisFetch.m and IRIS-WS.jar: used to query and download data via webservices
2. cptcmap: used to convert GMT formatted .cpt files into MATLAB format
3. m_map: used as an alternative to the Mapping Toolbox
4. matTaup: calculates predicted traveltimes and teleseismic ray paths.
5. processRFmatlab: provides routines for calculation of receiver functions, depth mapping, and miscellaneous signal processing.

2. Download and Setup

This section introduces the step-by-step instructions for downloading and setting up the M-files in the FuncLab package for your MATLAB session. For the purposes of this manual, UNIX and MATLAB command line inputs are typed in courier font, file names and MATLAB variables are in *italics*, and directory names are **bolded**.

The FuncLab website, which includes the source code, a quick start guide, this manual, sample datasets, and links to needed external files is located at http://robporritt.wordpress.com/software. Download the latest release and untar it in the directory in which you would like to place the MATLAB M-files. For example:

```
unzip funclab1.8.0.zip
```

The directory **funclab1.8.0** contains the files needed to run all the functions within the FuncLab toolbox. Appendix A contains a list of all the M-files. Links to extra map data and other relevant websites are provided in Appendix B of this manual and on the FuncLab website. These data include DEM or topography such as ETOPO1, ETOPO2v2c, and GLOBE DEM, as well as GSHHS coastline data. I recommend placing these under **funclab1.8.0/map_data/**. Alternatively a package including topography and coastline data is available from http://robporritt.wordpress.com/software.

In order to use the FuncLab functions, the user will need to add the paths to the files in the MATLAB session. To perform this task, from MATLAB, the user must change to the main funclab directory, and then run the setup_funclab script:

```
cd funclab1.8.0
setup_funclab
```

The setup_funclab script automatically adds all the directories to the search paths in MATLAB. In addition, this script checks for the Mapping and Signal Processing Toolboxes on the machine. If these are installed, certain functions will be available for use. However, if they are not installed, some functions may not be available. If the user has administrative rights on the machine running FuncLab, they may choose to insert the full pathname of the setup_funclab script to the startup.m file, located under the MATLAB application directory. See MATLAB documentation for startup.m for specifics (see http://www.mathworks.com/help/techdoc/ref/startup.html). This will then enable setup_funclab to run automatically at the start of MATLAB. FuncLab is now ready to be used on your machine.

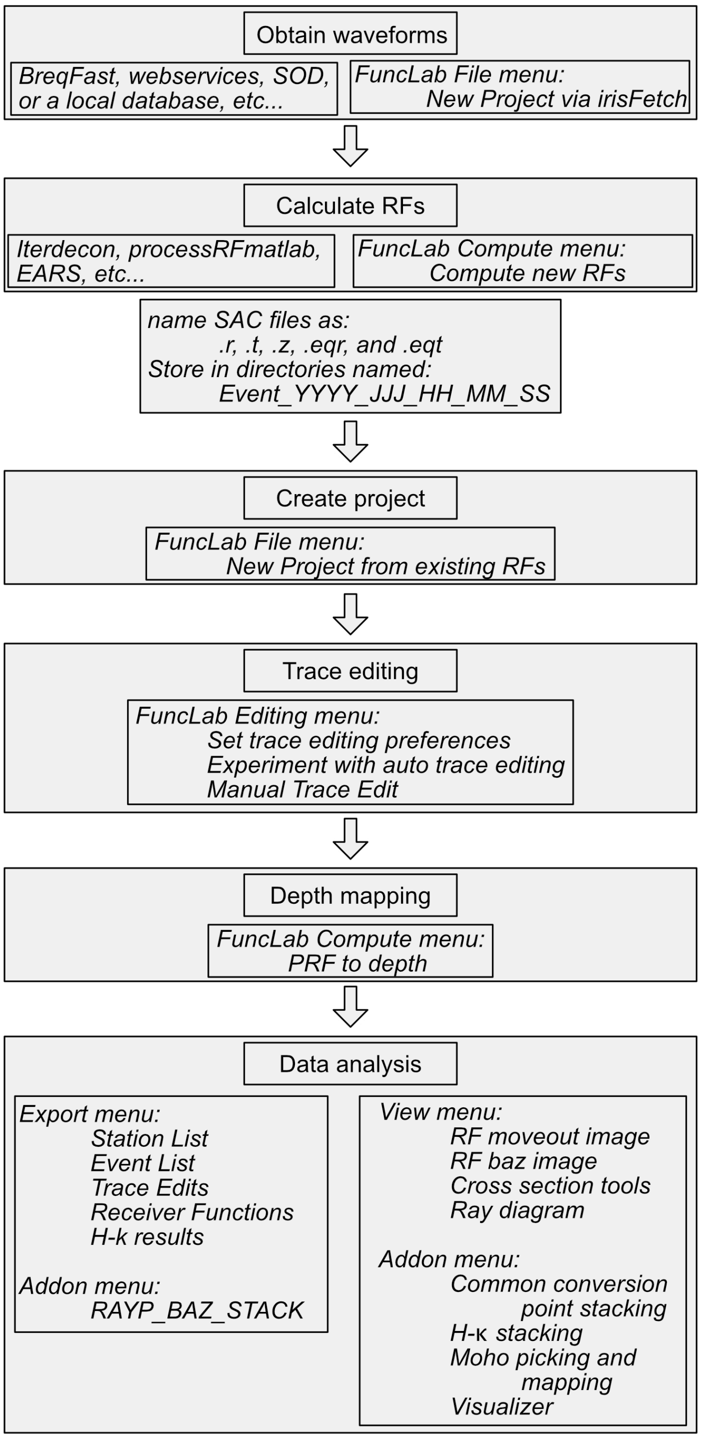
3. Receiver Function Overview, Preparation, and Raw Data Directory Structure

3.1 Receiver Function Overview

P-to-*S* converted waves (*Pds*) that arrive directly after the *P* wave provide information about velocity and density discontinuities in the Earth below seismic stations. These phases may be investigated directly on the horizontal components if the recorded earthquakes have simple impulsive source-time functions. However, many earthquake source wavelets are complicated and generally unknown. Since the *Pds* arrivals lie within the coda of the *P* wave and have amplitudes an order of magnitude smaller, stacking many records together to enhance the signal-to-noise ratio is necessary. Both *Langston* (1979) and *Vinnik* (1977) recognized the need for an approach to compare converted phases from events with different source signals and developed similar methods for isolating the *SV* from the *P* wave signal and removing the influence of the source signal from the seismograms. The receiver function therefore represents the Earth's impulse response, containing time and amplitude information that is sensitive to the local earth structure beneath a station. We refer the reader to *Langston* (1979) and *Vinnik* (1977) for further details on method, theory, and development.

There are several choices in source normalization procedures to isolate the *SV* energy, including the choices of coordinate system and deconvolution method. Regarding the choice of the coordinate system, *Langston* (1979) rotated the horizontal component seismograms along the free surface to point in the radial and tangential directions from the backazimuth of the earthquake (ZRT coordinate system). *Vinnik* (1977) performed a more complete rotation to the LQR coordinate system, where the radial component includes the raypath of the incident wave in the radial direction assuming a slowness vector from a 1-D Earth model. *Bostock* (1996) improved on the method of *Vinnik* (1977) by maximizing the *SV* wave energy using slowness as a function of time from the radial and vertical components' covariance matrix over a given time window. Although these successively improve upon each other, it is still difficult to achieve full *P-SV* wavefield separation and all methods represent an approximation. For the purposes of most imaging, however, this approximation is acceptable, as the signal of the *Pds* is visible with any of these rotation choices.

The second choice in source normalization involves the deconvolution method of the source from the seismograms. Deconvolution can either be performed in the frequency domain or the time domain. The most common method is done in the frequency domain, but requires water-level stabilization to eliminate spectral holes that result from division by zero in the frequency domain (*Clayton and Wiggins*, 1976). Although the algorithm is computationally fast, it has the disadvantage of requiring the user to manually search for the optimal water-level while trying to minimize the loss of spectral information. Side-lobes on *Pds* arrivals are also an adverse consequence of spectral leakage from this regularization method. An alternative frequency-domain method is the multi-taper method, which reduces the loss of spectral information, but requires greater computation (*Park and Levin*, 2000). Time-domain methods include matrix inversions (e.g., *Abers et al.*, 1995; *Gurrola et al.*, 1995; *Sheehan et al.*, 1995), which also require regularization in the form of damping parameters, and the iterative deconvolution method of *Ligorria and Ammon* (1999), which is more appropriately characterized as a predictive convolution that produces truly causal signals with no loss of spectral information.



3.2 Receiver Function Format and Preparation

In order for FuncLab to function in a more general sense with publicly available data, certain choices were necessary regarding data formats and directory structures. The current version is only capable of importing seismic data that is in the SAC format (http://www.iris.edu/software/sac/manual.html). It can read and handle SAC files that are either little endian or big endian byte-order automatically. It also requires a specific directory structure for the seismic data import and the creation of a new FuncLab project (next section). This directory structure originated from the output of publicly available data download using the Standing Order for Data (SOD) program (http://www.seis.sc.edu/sod/).

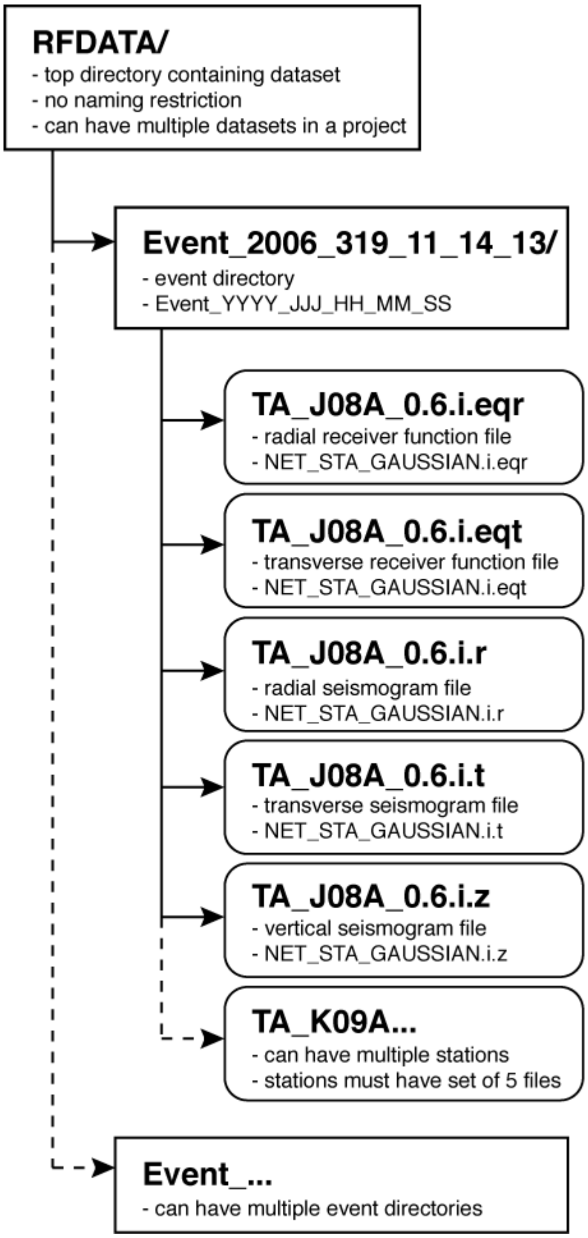
The choice of pre-processing methods to obtain receiver functions is independent of FuncLab, but through the integration of irisFetch.m and the processRFmatlab toolbox, FuncLab is able to retrieve seismic waveforms with webservices and prepare properly formatted receiver function data, load raw three component SAC waveforms from disk and calculate new receiver functions, or create a new project from existing receiver functions.

In general, we suggest deconvolution with an iterative deconvolution receiver function code written by Charles Ammon at Penn State University (http://eqseis.geosc.psu.edu/~cammon/HTML/RftnDocs/rftn01.html). FuncLab is thus written assuming that the receiver function output will be similar to that for this iterative deconvolution package. The link to download this code is also posted on the FuncLab website. Also posted are some generic C-shell scripts used to run the preprocessing of data in this directory structure and output filenames appropriately to help ensure that receiver functions are produced in a way in which FuncLab can read them. We also note that the EarthScope Automated Receiver Survey (EARS) data product (http://ears.iris.washington.edu/) should be able to provide receiver functions that can be used in FuncLab. For a more complete description of SAC file formats and header descriptions, see Appendix C.

There are some common pitfalls when preparing data for usage in FuncLab. The compute new RFs tool allows for various standard data formats, but assumes there is only one location code per station and that channels are named ?HE, ?HN, and ?HZ where the "?" represents the sample rate indicator of the channel code, usually L, B, or H. This naming convention may need to be changed as more stations are replacing the E and N indicators with 2 and 1. It also requires exactly 3 waveforms per station and therefore data should be merged before compute new RFs is called.

3.3 Receiver Function Raw Data Directory Structure

The raw data directories should be structured in the follow manner:



Receiver function files begin with the station name/code, contains the Gaussian parameter used to compute the receiver function, and ends with *i.eqr* (radial) or *i.eqt* (transverse). Files for both the radial and transverse receiver functions must be included for completeness. Also, it is helpful to include the radial, transverse, and vertical seismograms to exploit some of the seismogram viewing features within the FuncLab toolbox. All five files are required for a single event-station pair. In the case of a dataset downloaded with the irisFetch interface, additional files containing the LQT (ray centric) rotated data are also included as .p and .sv and data file names contain more parameters to further describe the data.

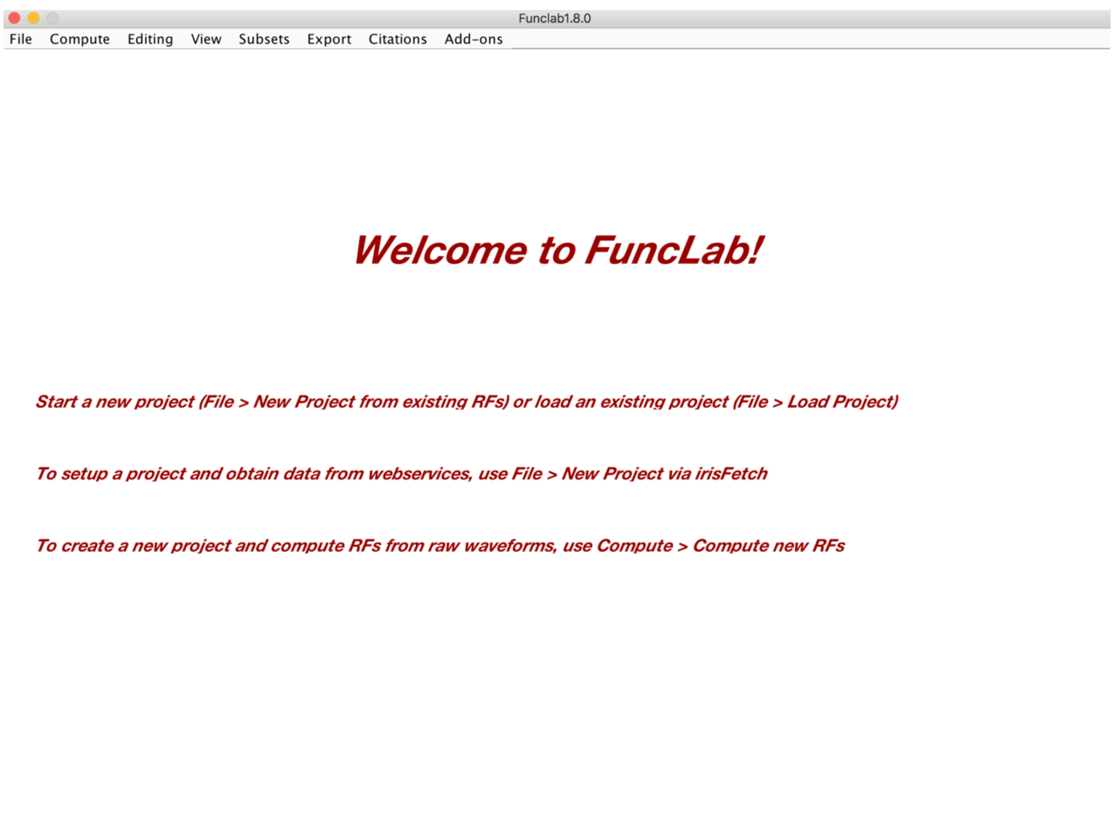
4. Graphical User Interface

To begin, the user invokes the main FuncLab GUI using the *funclab* command in MATLAB. From this interface, the user may start a new project or load an existing one.

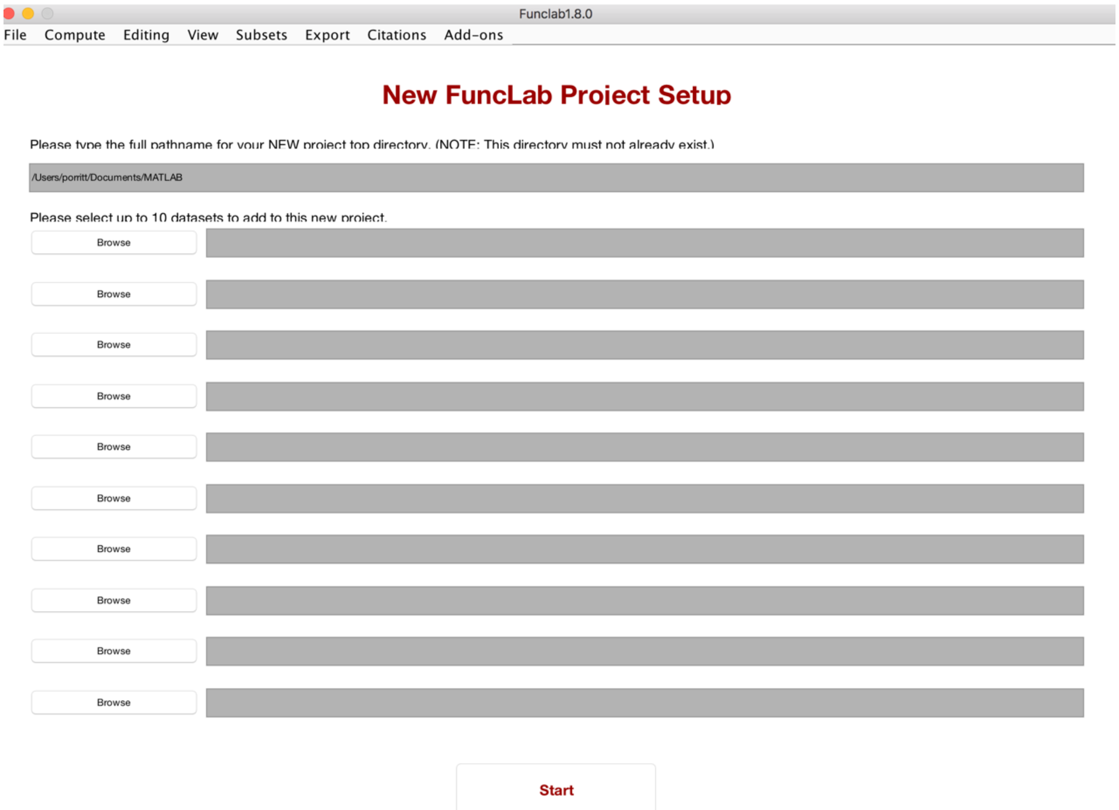
4.1 Starting a new project from existing RF data

To begin by starting a new project, where the data has already been pre-processed outside of FuncLab, select *File→New Project* from the top menu.

The "New FuncLab Project Setup" interface will appear. From here, the user will first determine the directory for the new project. This directory must not exist, as FuncLab will create and populate it with the necessary files and sub-directories for the project. Up to 10 separate data directories, formatted in the required structure as discussed above, can be imported at the initiation of the project. The user may either enter these directories

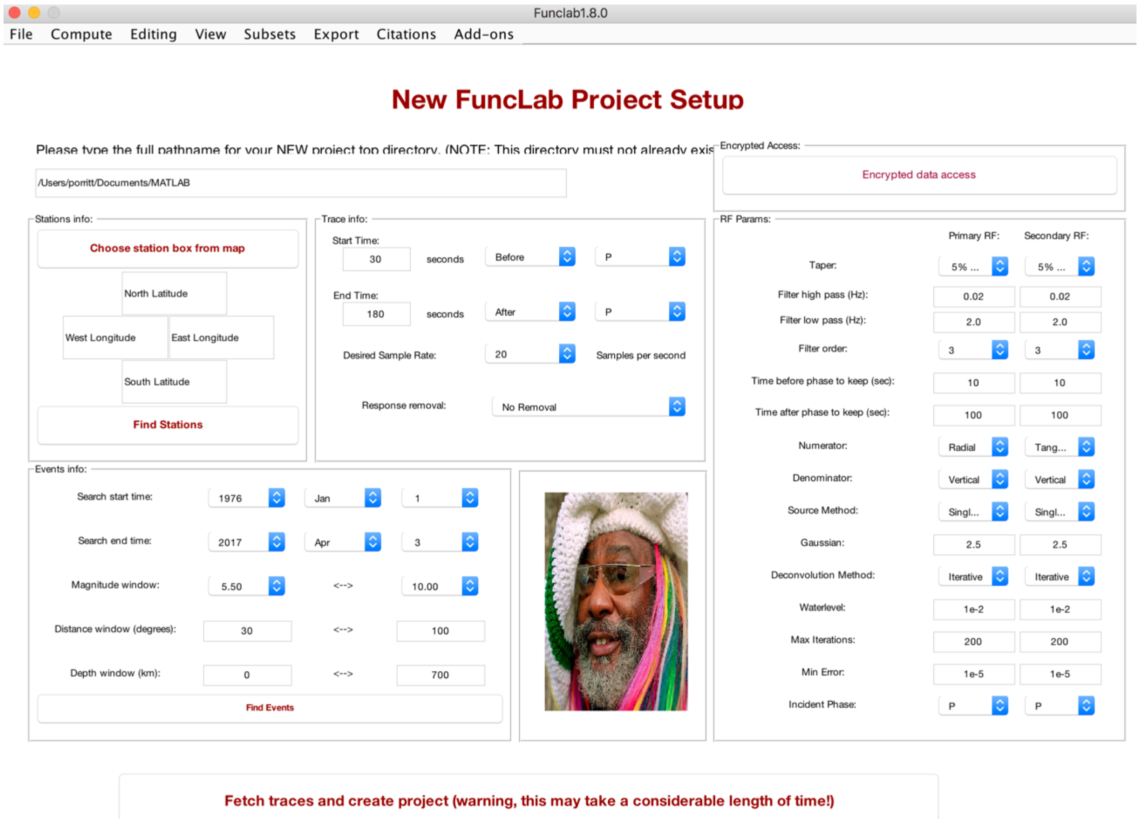


manually in the specified boxes, or browse for these directories on the computer. After entering as many data directories as the user wishes to import for this project (up to 10, but more can be added once the project is created), the user hits the *Start* button. FuncLab, at this point will create the new project directory with some sub-directories, copy the data directories into the project directory and populate the new *Project.mat* file that will handle most of the metadata for interaction with the GUI.



4.2 Starting a new project from irisFetch.m

Since the initial release of FuncLab, there has been a major revolution in the distribution of seismic data through webservices, which allow programmatic pulling of waveform data from servers. This significantly shortens the time in data preparation compared to most previous email-based request tools that often take researchers days to email a server, wait for the server to assemble the data, and then pull the data. In addition to waveform data, webservices can retrieve event, station, and response metadata directly within Matlab. Therefore, the new FuncLab contains an interface to design a project by selecting stations, events, and receiver function parameters and then it pulls the data. The stations info box allows the user to set geographic limits on the stations to request and then sends a request for available stations to IRIS webservices. The list of found stations is then presented in a table allowing the user to select or deselect stations. In the events info box, the user can set the time, magnitude, distance and depth ranges as filtering criterion and FuncLab will call irisFetch.m to retrieve the desired earthquakes information. The returned event data are presented in a table for further selection. The trace info box allows setting the length of the waveform relative to the earthquake origin time or a specific phase arrival calculated with matTaup. Finally, the RF Params box sets parameters used by processRFmatlab in calculating receiver functions.

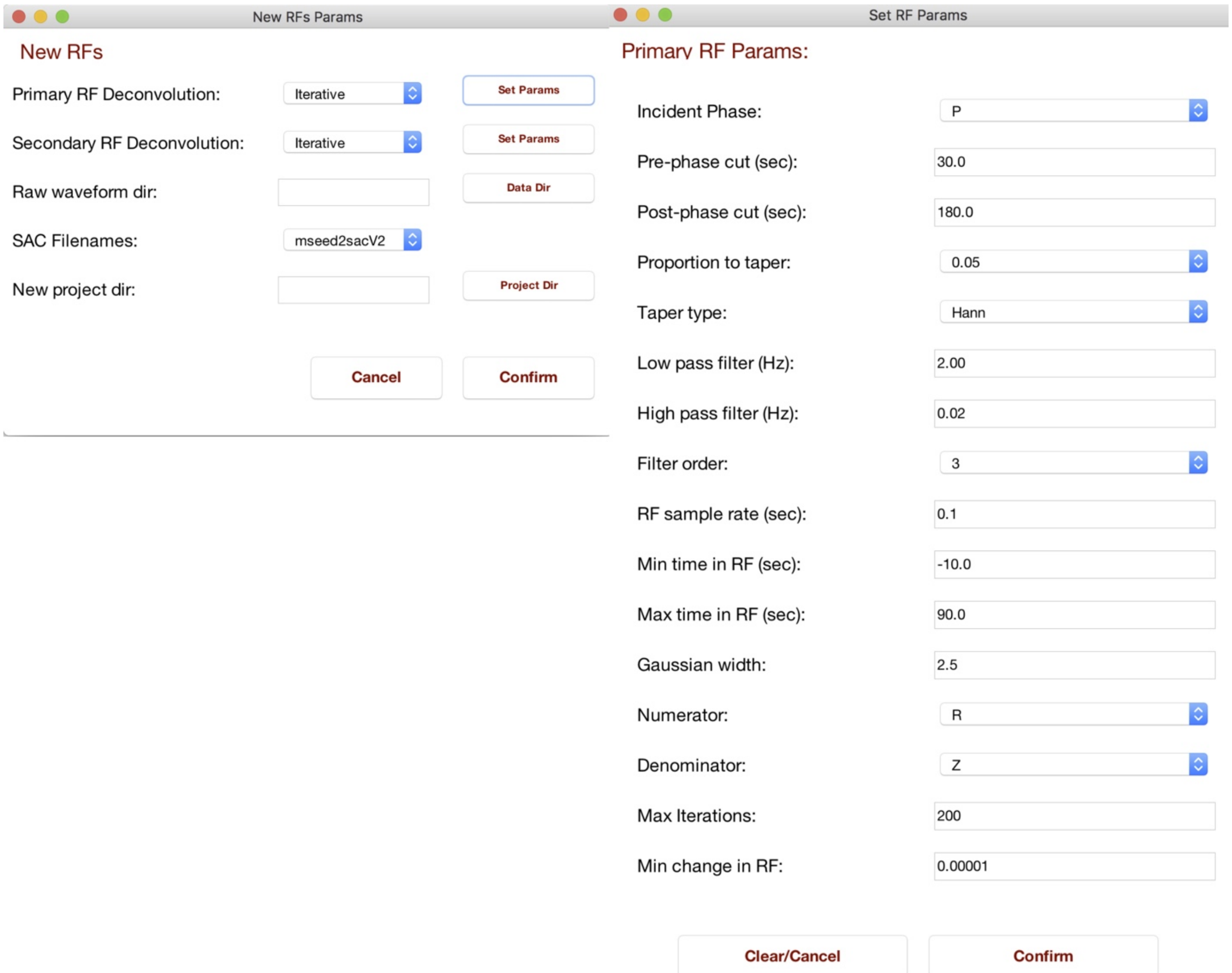


Data fetching interface. Top box gives an editable element to put in a new directory for the project. Station, event, and trace info boxes determine the dataset to pull. RF params determines the parameters in receiver function calculations after data download. Encrypted data access pushbutton provides an interface to allow access to encrypted data. Image of George Clinton fits with the theme of "Func".

Data is downloaded into a new directory called **RAWTRACES/** within the project directory. When the download is completed, the **RAWTRACES/** directory is copied into **RAWDATA/** to fit the usual FuncLab format.

4.3 Compute new RFs from waveform data

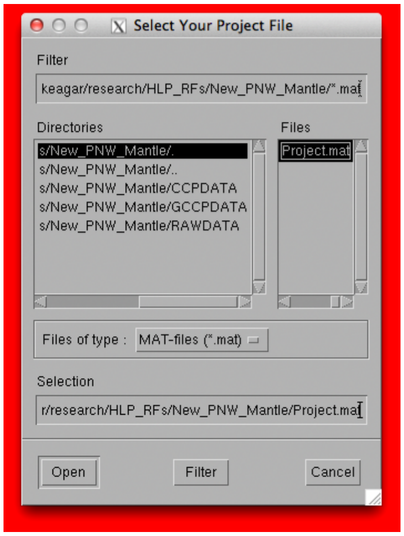
The new compute menu provides GUI front ends for processRFmatlab. The compute new RFs menu prompts for primary and secondary RFs (alternatively labeled radial and transverse) and allows the user to set the type of deconvolution, choose the directory with existing waveforms in a standard file naming format convention, and then a new directory into which the computed RFs will be placed and a new project will be formed.

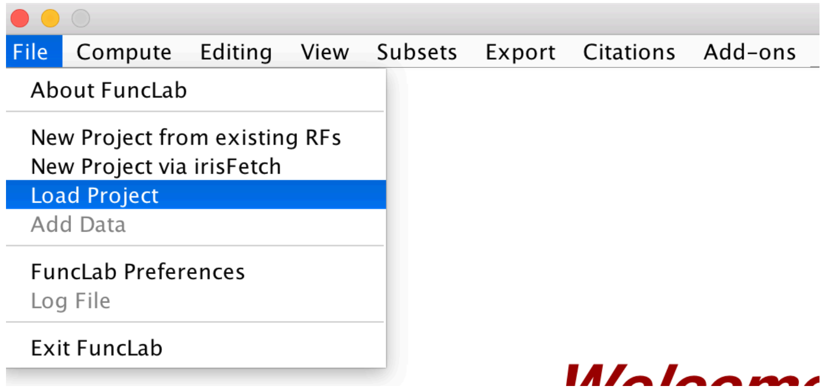


Compute new RF interface. The smaller window on the left opens first and pull down menus allow changing the deconvolution type and sac filename formats. Upon hitting the Set Params button, the window on the right opens allowing the user to set a variety of deconvolution specific parameters.

4.4 Loading an Existing Project

If you are working from an existing FuncLab project, you can load the data into FuncLab by select *File→Load Project* from the top menu.





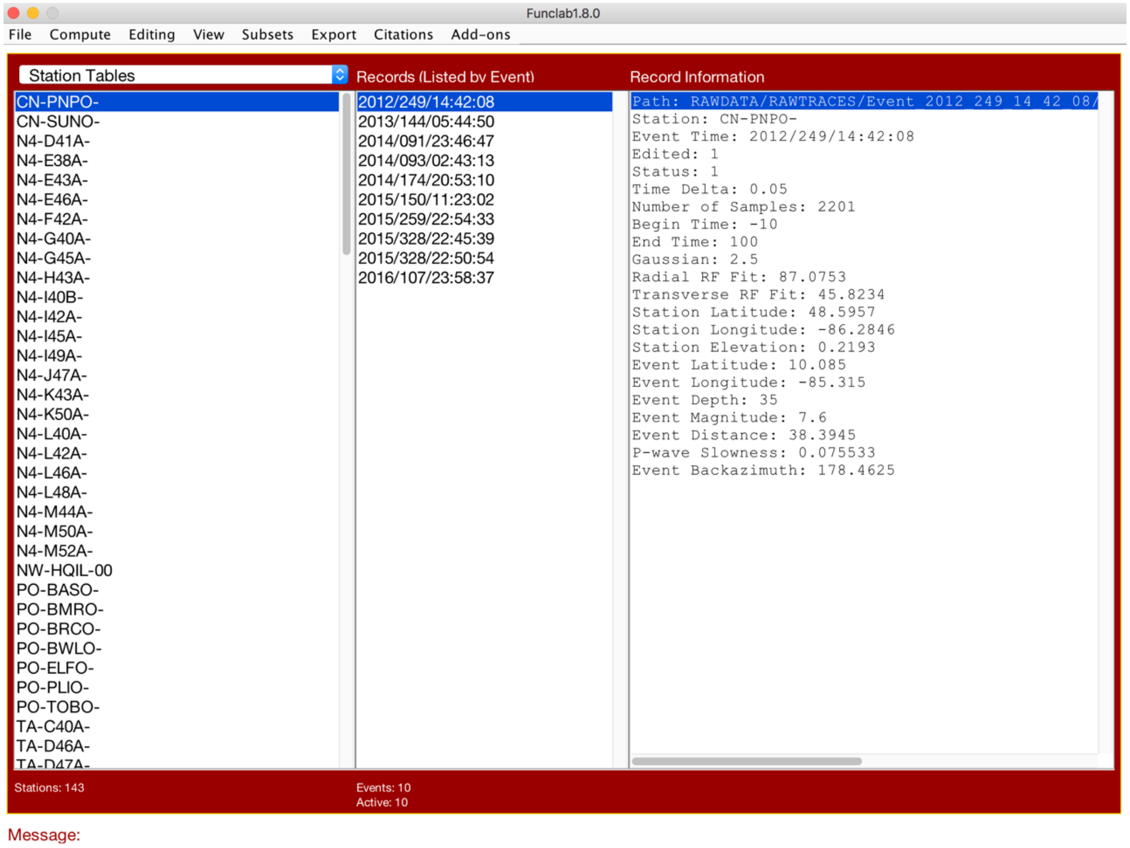
Select the *Project.mat* file that is contained one level into the existing Project directory and select *Open*. The main FuncLab GUI will open with your project and you are ready to begin.

4.5 Project.mat File and FuncLab Variables

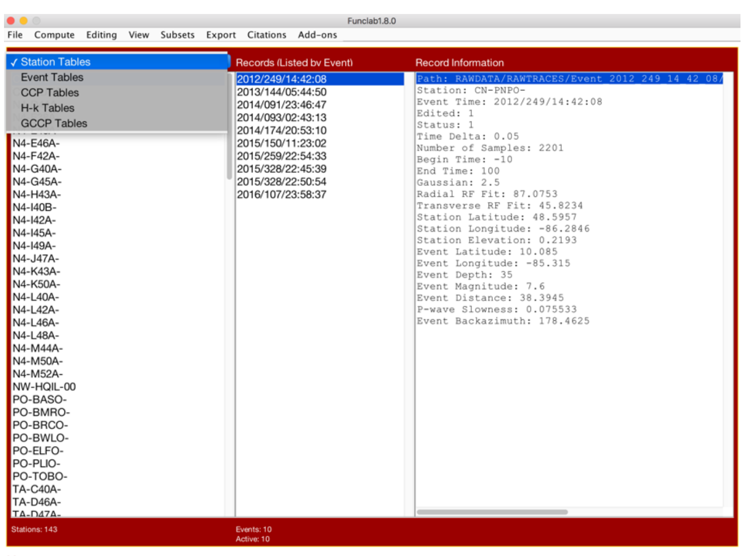
FuncLab saves the metadata for each station-earthquake pair of waveform files in the *Project.mat* MAT-file. Metadata for the records are organized into a cell array variable called *RecordMetadataStrings* and a double array variable called *RecordMetadataDoubles*, where each column contains different information such as station name, event origin time, waveform sample rate, etc.

4.6 Main FuncLab GUI

The main FuncLab GUI allows the user to interact with the entire dataset. The GUI includes eight top menus; *File*, *Compute*, *Editing*, *View*, *Subsets, Export, Citations,* and *Add-ons*. A drop-down menu in the top-left defines the manner in which receiver function metadata is organized for visualization in the GUI (we refer to these as tables, although this is not a formal relational database system). Records common to a particular station are viewed in "Station Tables". The stations are then listed and selectable in the left-hand panel. Below this panel is listed the total number of tables displayed. The records within the "Station Table" are listed by the event origin time in the center panel. The total number of records and number of active records are listed below the panel. Information or metadata about the selected record in the center panel are then listed in the right-hand panel. These include station and event information, time series information, and the path of the SAC files for the record.



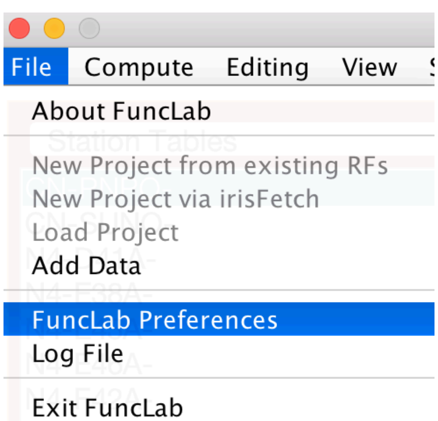
The user can also view the records organized by event origin by selecting the “Event Tables” in the drop-down menu. Other options in the drop-down menu become available when using add-on analysis tools discussed below.

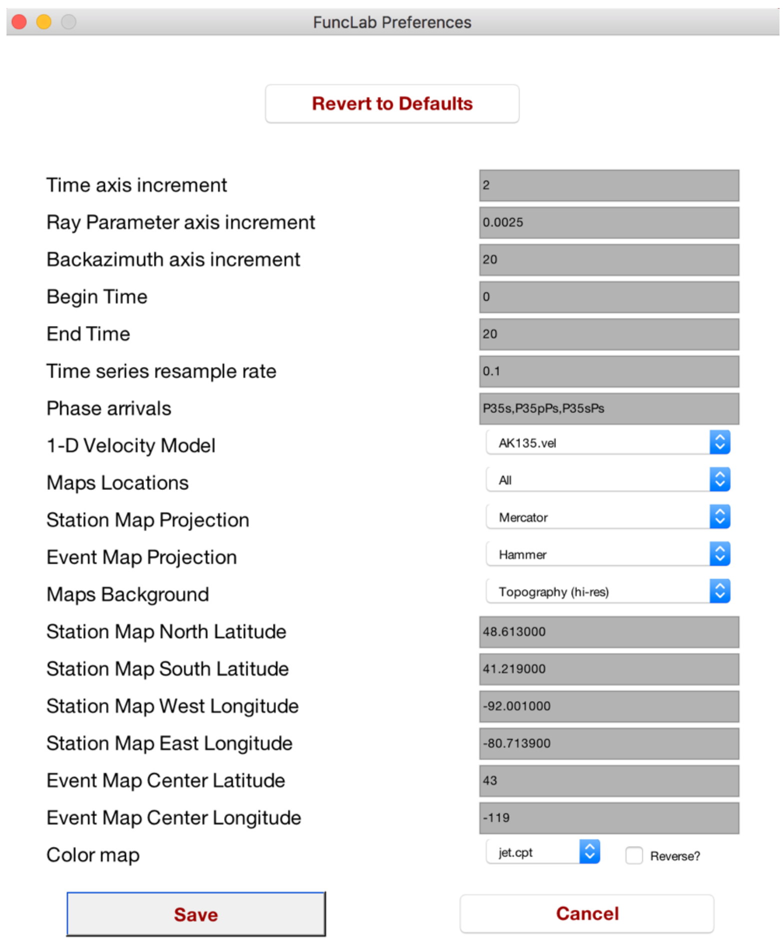


The user can also view the records organized by event origin by selecting the “Event Tables” in the drop-down menu. Other options in the drop-down menu become available when using add-on analysis tools discussed below.

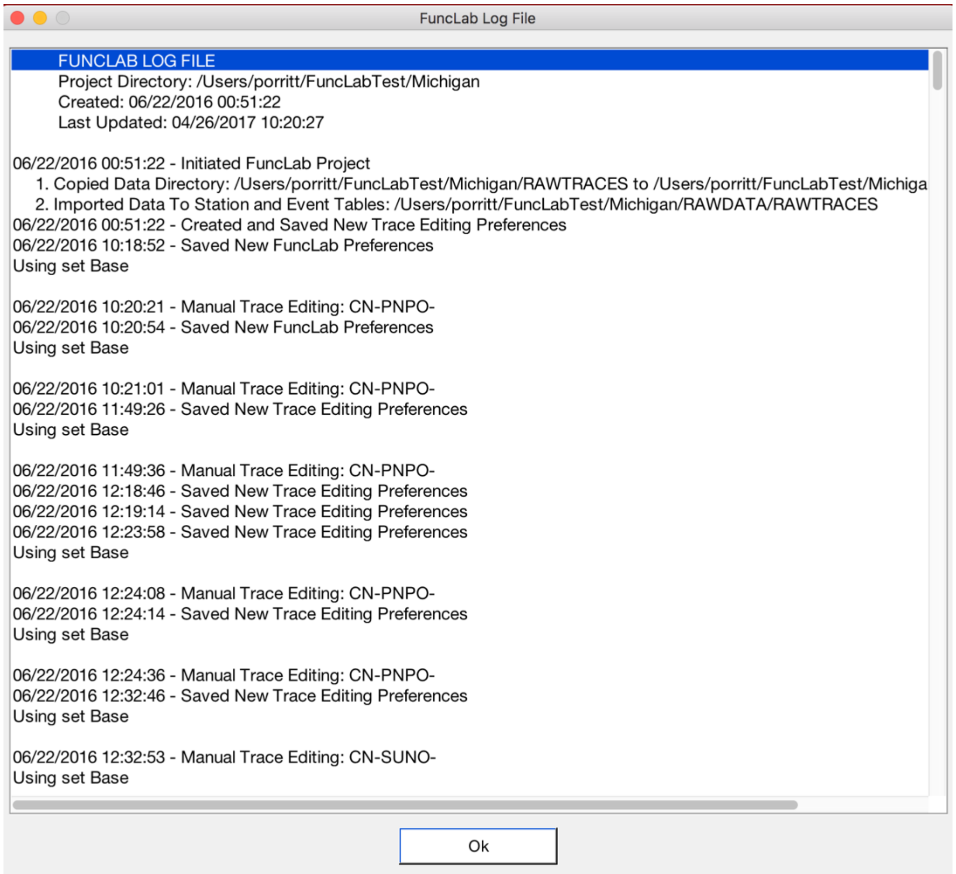
The *File* menu contains options that control FuncLab and projects as a whole, such as starting or loading projects, adding data, and changing preferences. New data may be added to an existing project by selecting *File→Add Data* from the top menu.

FuncLab preferences control certain aspects of all the plotting functions, such as time axis parameters, backazimuth and ray parameter axis parameters, and mapping parameters. These are accessed through the *File→FuncLab Preferences* menu item. Adjustments of map boundaries and axis settings from the defaults are likely required depending on the imaging target of the receiver function study.



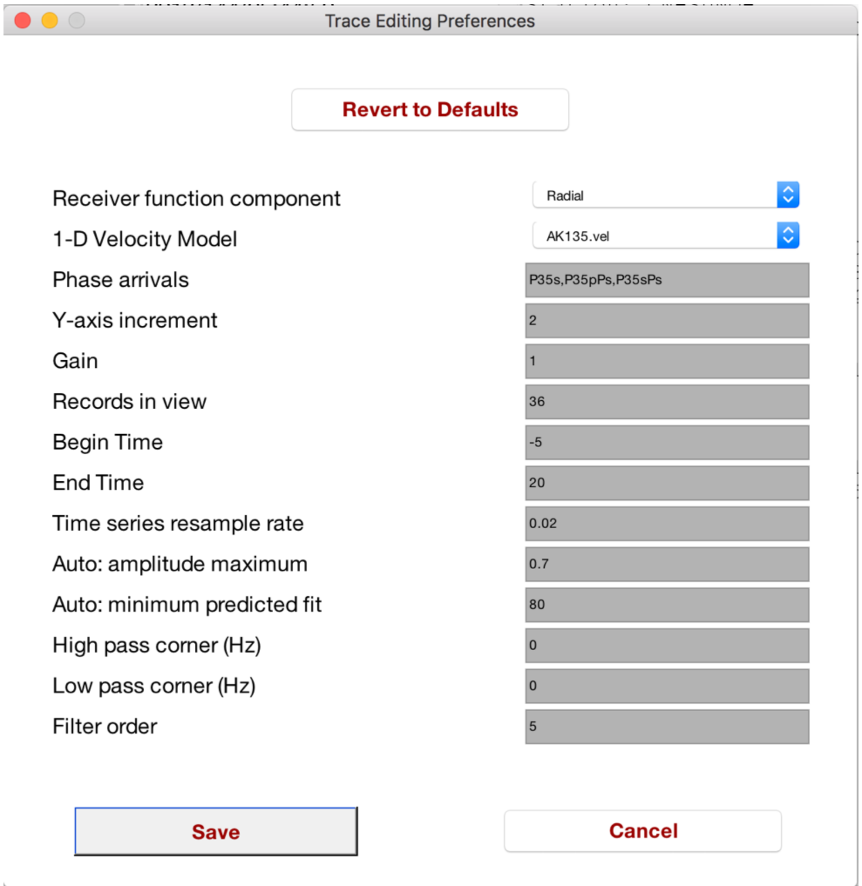


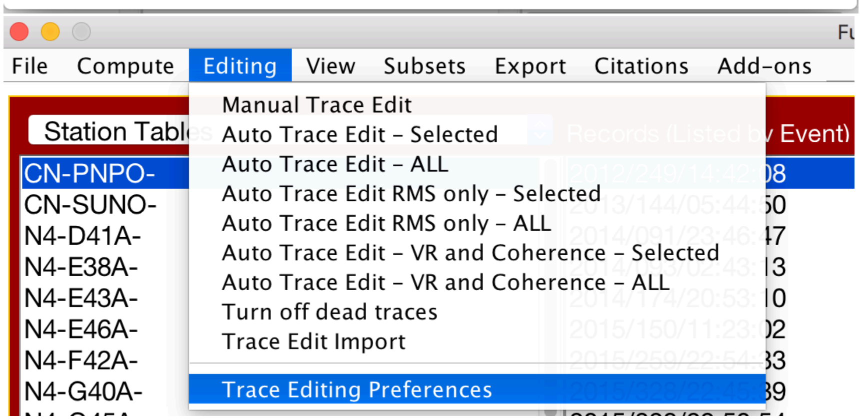
A log file keeps information about each process performed on the dataset. The log file can be viewed from the main FuncLab GUI using the menu *File→Log File*. This is a text file that is initiated when the project is first created. Dates and times of processes are logged, as well as the name of the process and certain preferences set to run the process. This is not editable in the GUI and is intended for user guidance.



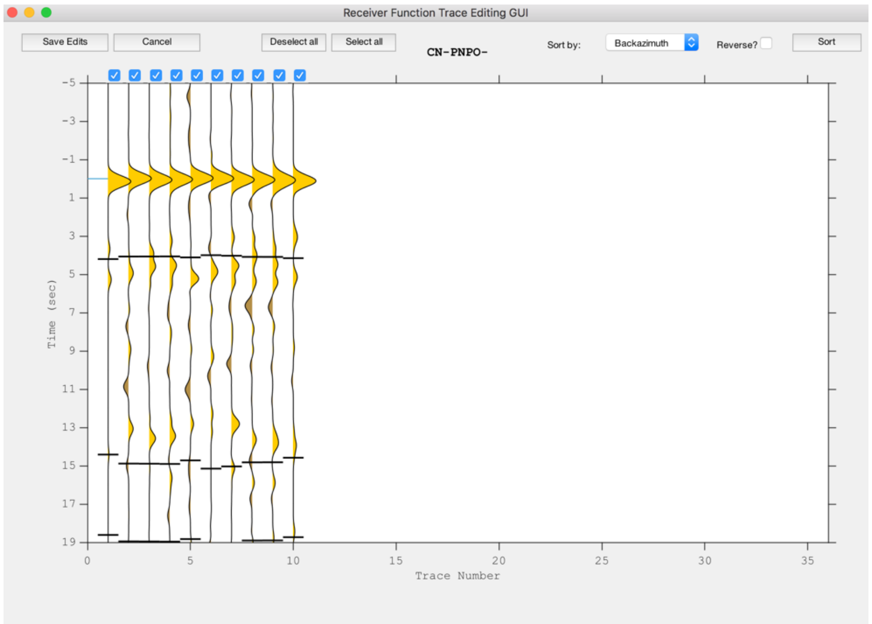
4.7 Trace Editing

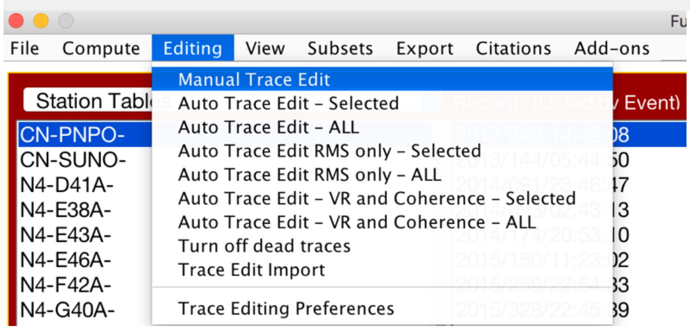
From the main FuncLab GUI, the Trace Editor can be accessed through the *Editing* menu, which contains options for manual and automated trace editing. The first step that should be performed is setting the trace editing preferences. These can be accessed through *Editing→Trace Editing Preferences*. These control aspects of the GUI display such as receiver function component (radial or transverse), time axis parameters, gain, number of receiver functions in the view, and sample rate for resampling. Other preferences, such as the maximum amplitude and the minimum receiver function fit, can be specified as control parameters in automated trace editing.



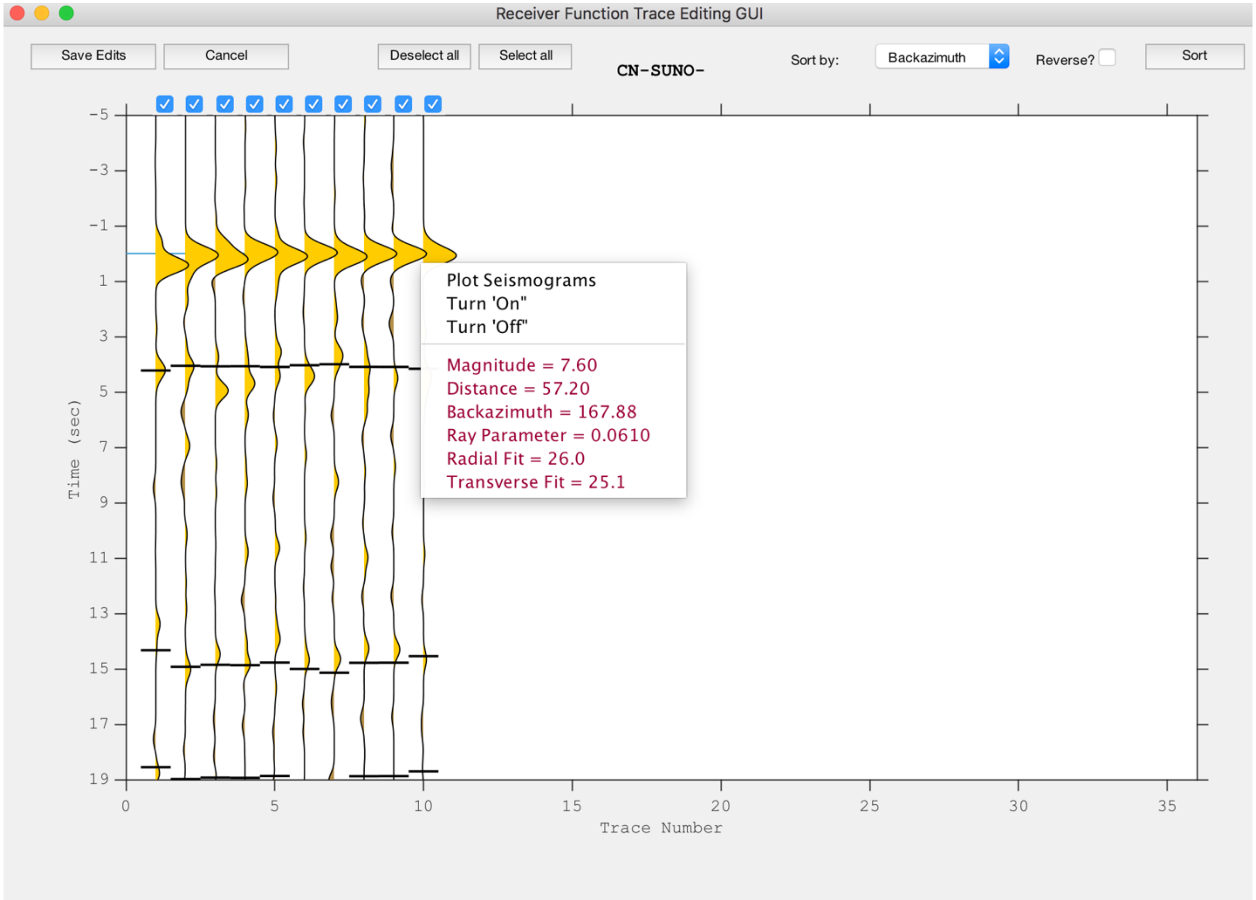


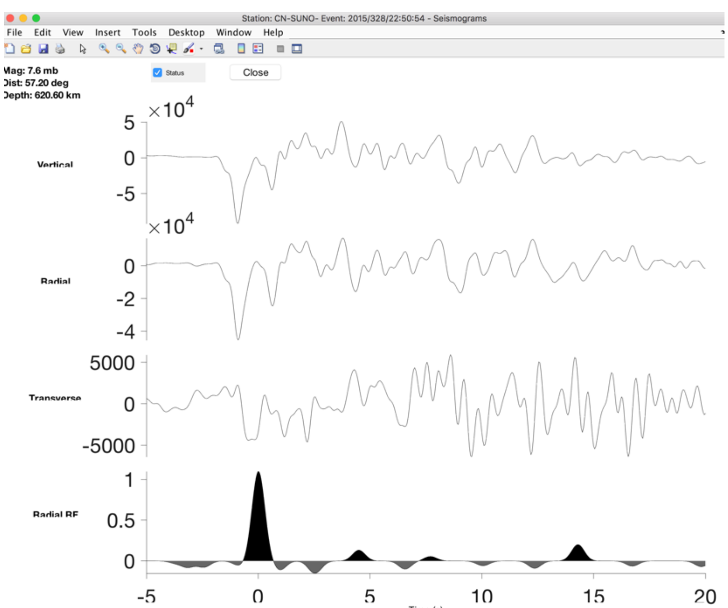
Opening the *Manual Trace Edit* GUI will read in the receiver function SAC files of the records in the selected table in the left-hand panel. The Trace Editing GUI displays the receiver functions as wiggle traces with time increasing down the y-axis and each record one unit wide across the x-axis. The scrollbar at the bottom allows the user to scan through records in the table to display them in the window. The record status is changed in the "Status" column of *RecordMetadataDoubles* and saved to *Project.mat* at the end of the trace editing session. Each record has a checkbox above it, which is used to select the status of the record (check means "on" and no check means "off"). The ability to change a record's status allows the user to specify good records that will be used in future analyses (i.e., stacking and visualization processes) and ignore bad records. Records with status “off” remain part of the dataset and are not deleted from the project. The color of the wiggle traces is another indicator of the record’s status. New data that have not yet been evaluated through the Trace Editor are colored red, whereas data that has been looked at in the GUI before are colored gold. All records with status “off” are shaded gray. Buttons along the top allow turning all waveforms on/off or sorting by any of the record metadata.





Furthermore, right-clicking the waveform brings up a menu to start another GUI for visualizing the seismograms and radial receiver function, change the record status, and displays event and record information useful in determining the appropriate status of the record. The record status is changed in the “Status” column of *RecordMetadataDoubles* and saved to *Project.mat* when the *Save Edits* button is selected.





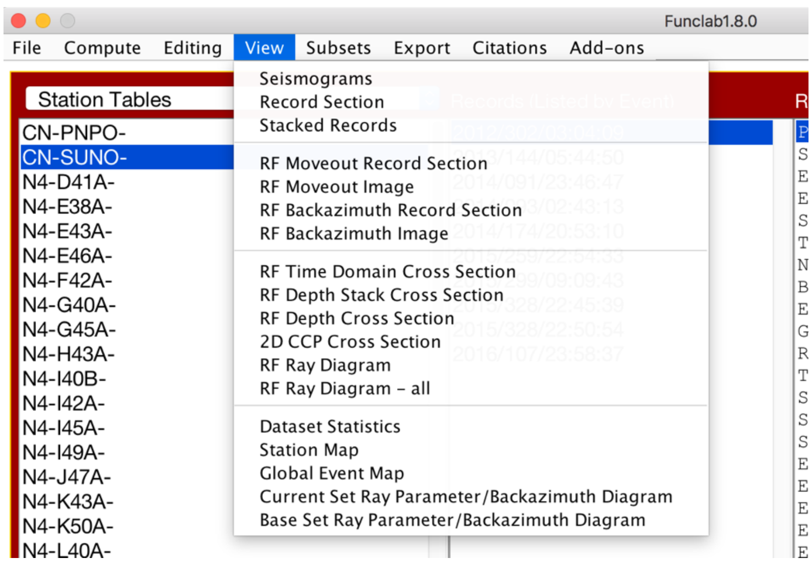
Automated trace editing can be performed on a selected table (*Editing→Auto Trace Edit – Selected*) or the entire dataset (*Editing→Auto Trace Edit – ALL*). The second criteria is a minimum fit or measure of uncertainty in the receiver function, expected to be set in the USER9

SAC header. As of version 1.8.0, additional auto options have been included such as using the RMS only or a variance reduction and coherence metric for all RFs at one station.

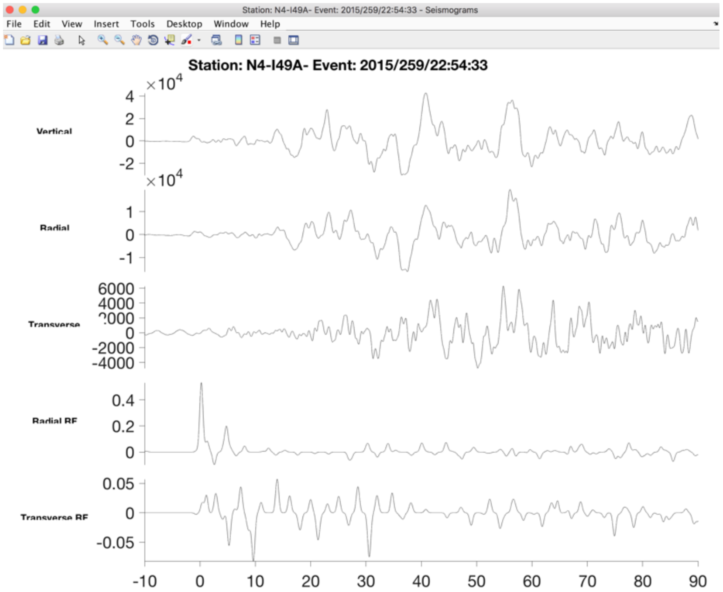
4.8 Viewing Data (Plots and Visualizations)

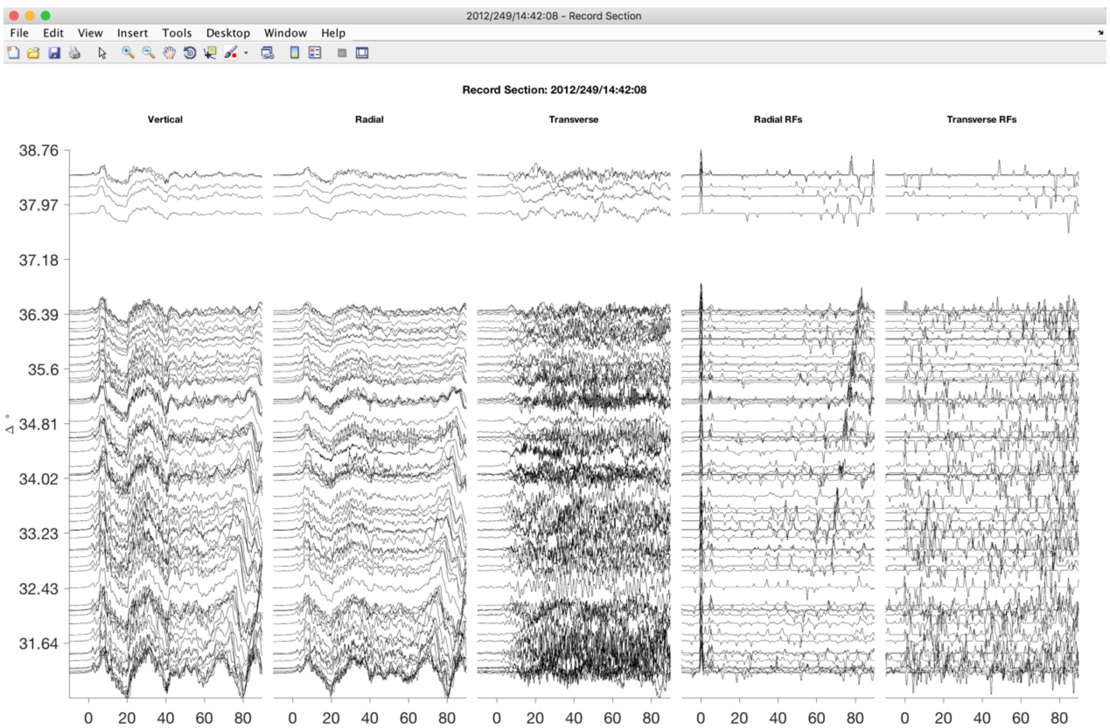
A number of data plotting functions are located under the *View* menu in the main FuncLab GUI and are divided into four sections. The top section displays plots related to the selected record in the center panel. Options in the middle section display plots of active records in the selected table in the left-hand panel. The third section provides options to view the RF data projected in geographic coordinates. The bottom section includes visualizations that involve active and inactive records within the dataset.

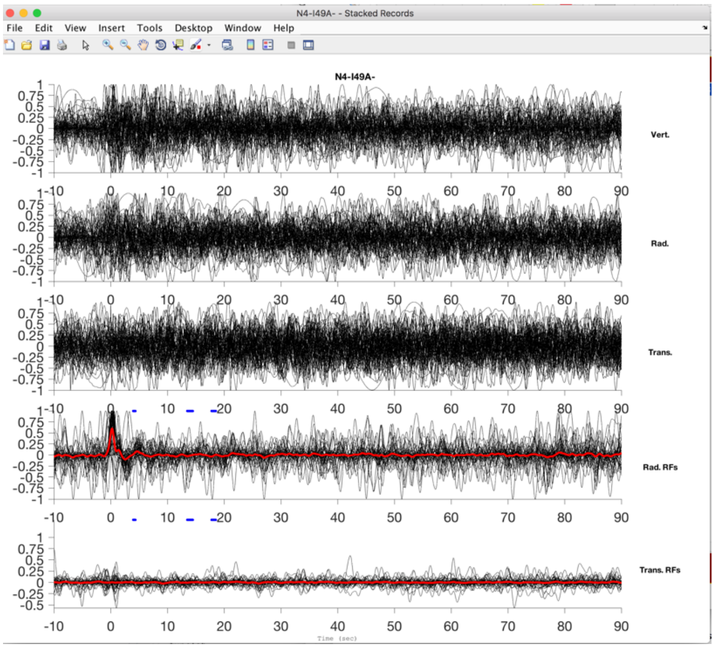
4.8.1 Viewing seismograms



The *Seismograms*, *Record Section,* and *Stacked Records* items plot the vertical, radial, and transverse seismograms and radial and transverse receiver functions. Note that the record section plot is more effective when switching to “Event Tables” and highlighting an event, rather than a particular station. A quick method of QC and trace editing your data may be to look at these plots by event, then by right clicking the event table, you can select either “Turn On” or “Turn Off” to edit all records in that event.


Station: N4-I49A- Event: 2015/259/22:54:33 - Seismograms
File Edit View Insert Tools Desktop Window Help
Station: N4-I49A- Event: 2015/259/22:54:33
Vertical
Radial
Transverse
Radial RF
Transverse RF
Time (s)

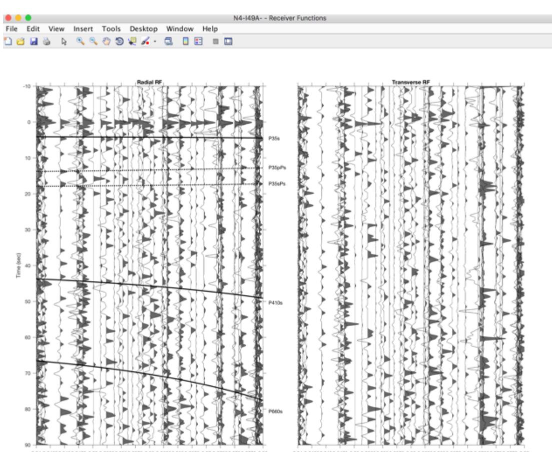

2012/249/14:42:08 - Record Section
File Edit View Insert Tools Desktop Window Help
Record Section: 2012/249/14:42:08
Vertical
Radial
Transverse
Radial RFs
Transverse RFs
Time (sec)

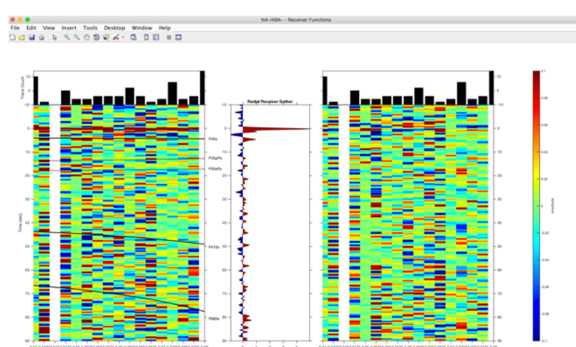


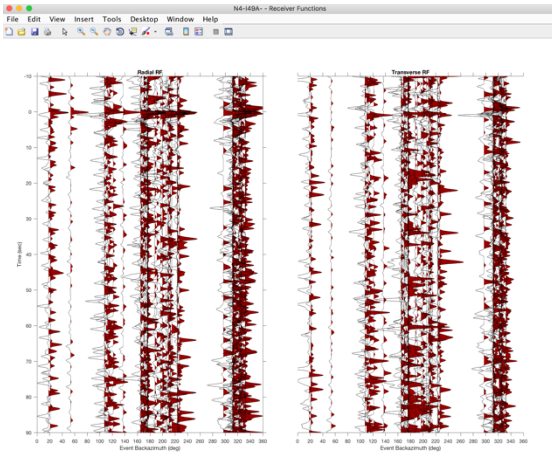
4.8.2 Viewing station/event summary plots

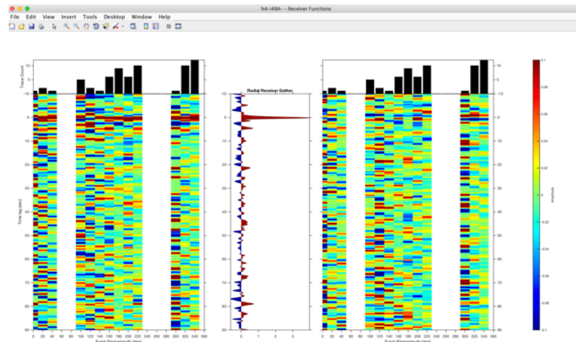
The *RF Moveout Record Section* item plots receiver function wiggle traces of active records vs. ray parameter, and the *RF Moveout Image* plot the vertical, radial, and transverse seismograms and radial and transverse receiver functions. The *RF Moveout Record Section* item plots receiver function wiggle traces of active records vs. ray parameter, and the *RF Moveout Image* stacks records into ray parameter bins and displays the amplitudes by a color scale. In the Image plots, a central 1D stack of all the bins is displayed.

RF Backazimuth Record Section and *RF Backazimuth Image* are similar except that records are displayed vs. backazimuth.



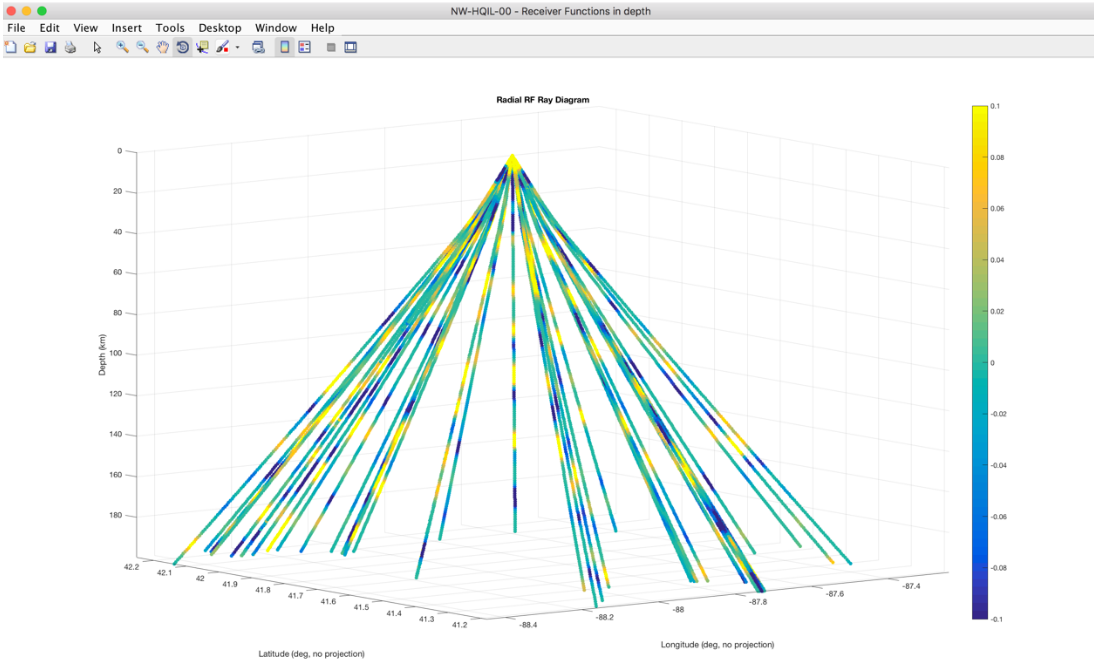




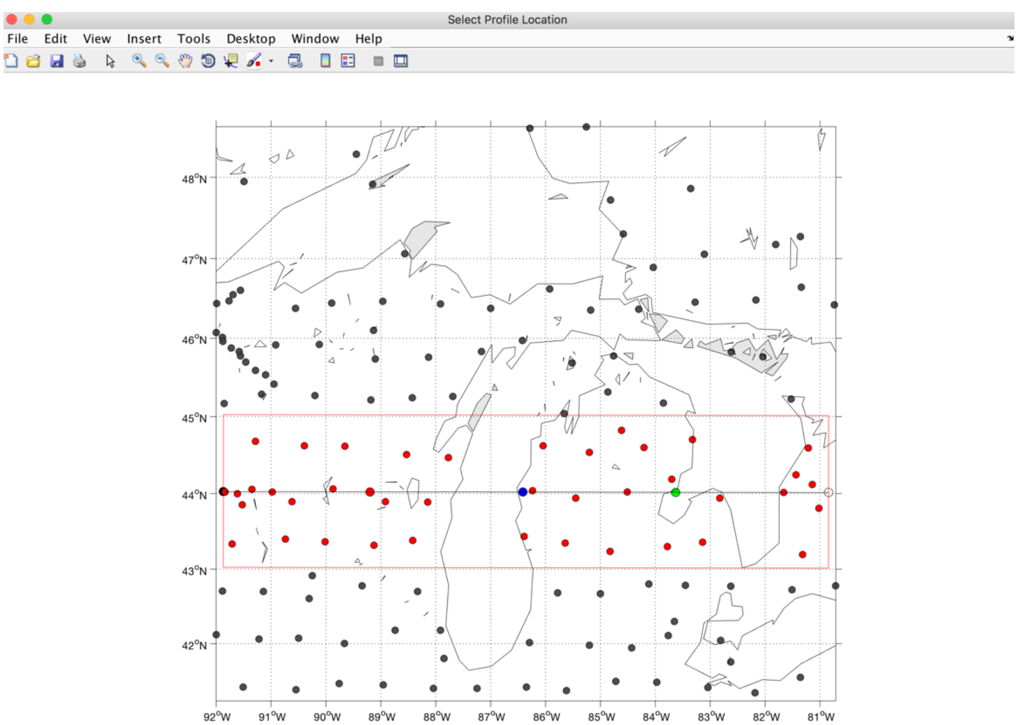


4.8.3 Viewing cross sections

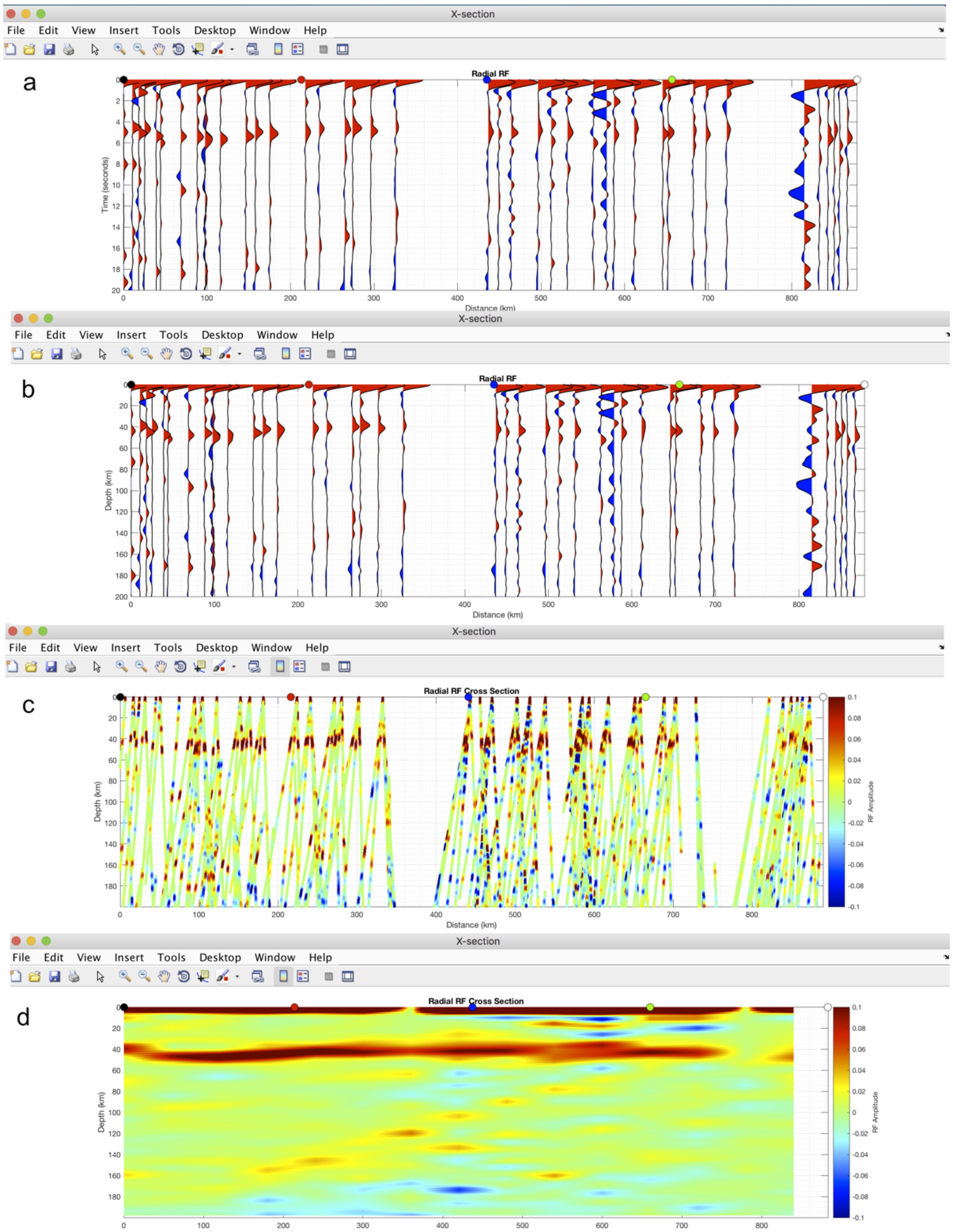
As of major release 1.8.0, FuncLab includes four new cross-section tools and a 3D ray diagram tool. The latter can be run on the current table or all waveforms and presents a 3D scatter plot of the converted wave amplitudes at their geographic locations after time to depth migration. Running this module for all waveforms is slow for medium to large datasets. The four cross-section tools provide relatively fast algorithms for estimating structure along an arbitrary linear transect. They operate in similar fashion by first opening a station map with m_map and then prompting the user to select start and end points visually. Once the start and end points are selected, a new prompt is created asking for parameters such as box width, waveform gain, and color limits. The RF Time Domain Cross-section tool uses the 1D velocity model selected in FuncLab Preferences to correct RFs for each station in the selected box for move-out, stacks them in depth, and then re-maps the stacked RF to the time domain based on the same 1D velocity model. The cross-section is then plotted as color filled wiggles vertically under each station along the profile. RF Depth Stack Cross-section is similar, but avoids the final step of re-mapping the stacked RF to the time domain. If the user has already mapped the RF to depth with the compute menu, then the RF Depth Cross-section is available to project the depth mapped RFs onto a plane. This tool presents the RFs as a series of pine-tree like traces colored by amplitude and located based on their projection onto the plane. The result is similar the 2D CCP diagram, but without stacking into bins. Finally, the 2D CCP Cross-section tool performs on-the-fly CCP stacking along the profile. This will result in a smoothed picture of subsurface structure, but patterns that may have been missed in other tools may be highlighted.



Example ray diagram for station HQIL. Rays are mapped with a 3D velocity model and projected into a Cartesian space. Colors show the amplitude of the radial component RF.



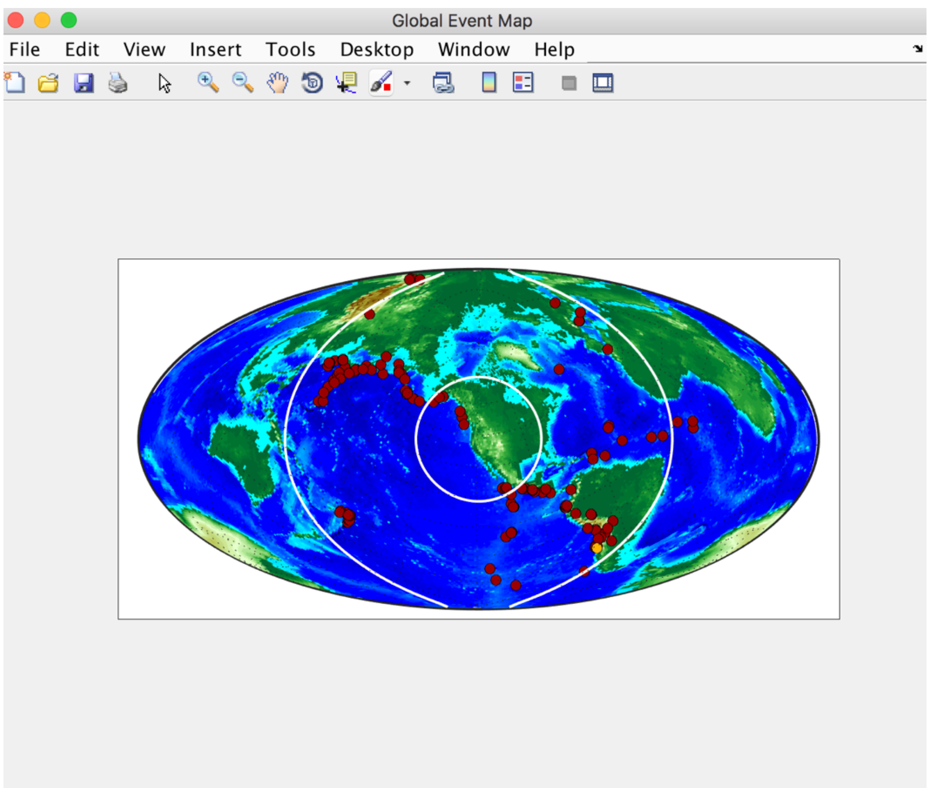
View of a station map after a 2D RF cross-section has been selected with a width of 2 degrees. Black dots show stations not included in the cross-section, red dots show stations included in the cross-section, and the larger dots with multi-colors are mirrored at the top of the profile in the resulting cross-section figures.

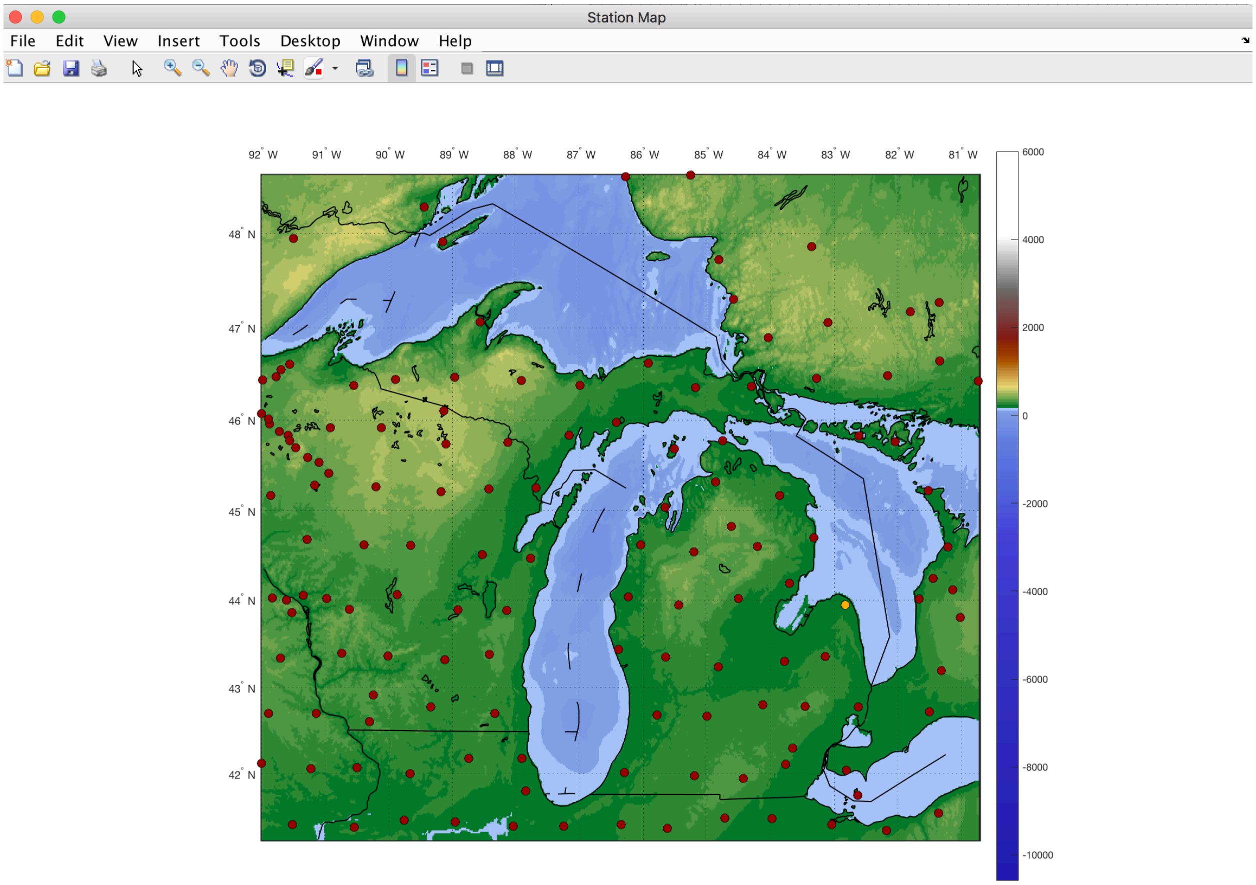


Examples of view RF cross-sections along the line above. Section (a) is the time domain RF stack. Traces are plotted at the projection of the station onto the cross-section line. RFs are migrated to depth before stacking to account for moveout. Section (b) is similar to (a), but the vertical access is in depth after assuming a 1D velocity model. Section (c) is the geographically mapped RFs projected onto the cross-section plane. No stacking is performed for this image. Section (d) is the 2D on-the-fly CCP stack.

4.8.4 Viewing station/event maps

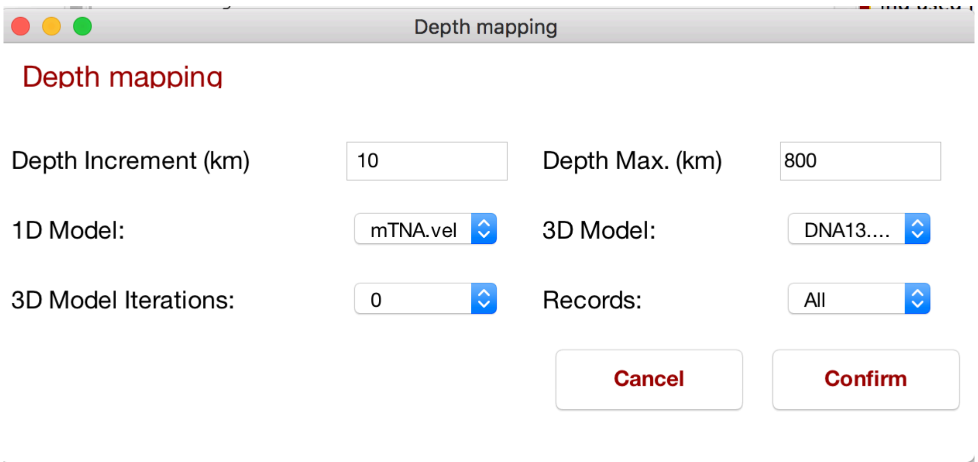
The *Dataset Statistics* item is not a true plot or visualization, but displays a textbox summarizing various statistics, such as the number of total active records, about the dataset. Another data coverage visualization is the *Ray Parameter/Backazimuth Diagram* which plots these metadata on a polar plot. Finally, two map options, a *Station Map* and a *Global Event Map*, are available. The available projections vary with whether or not the user has the Mapping Toolbox installed. In these maps, the active stations/events are plotted in red, the inactive stations/events are plotted in grey, and the currently selected station/event is in gold.





4.9 Compute menu

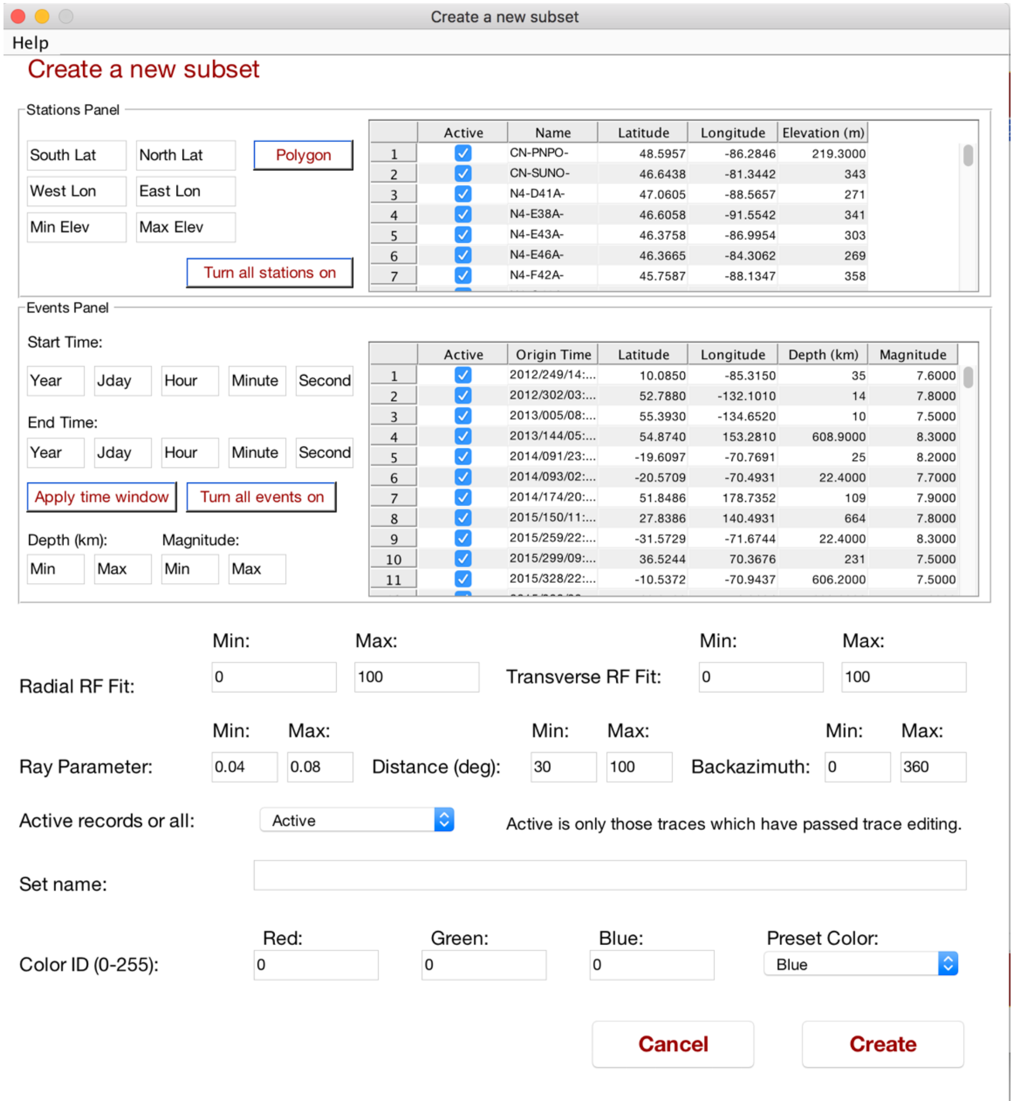
The compute menu was added for major release 1.8.0. This menu allows creating a new project from three component sac files, a call to matTaup to calculate the incident P wave ray parameters, and map the PRFs (P wave receiver function as opposed to SRF for S wave receiver functions) to depth. A similar function to calculate arbitrary incidence is not yet available, nor is an option to map S to P receiver functions to depth. The depth mapping routine allows the user to set a maximum depth and depth increment, choose a 1D velocity model, and optionally iterate through a 3D velocity model. Depth mapping is first done through the 1D model and then, if 3D model iterations are selected, a new 1D model is found for each RF based on ray tracing through the 3D model.



Once the depth mapping is completed, the View RF Ray Diagram and View RF Ray Diagram – all become active under the view menu. This also writes the depth mapped receiver functions to disk under the **DepthTraces/** folder into a file *DepthTracesRayInfo.mat* and saves the parameters as *DepthTracesParams.mat*. The *DepthTracesRayInfo.mat* file can then be used in the CCPV2 Add-on interface.

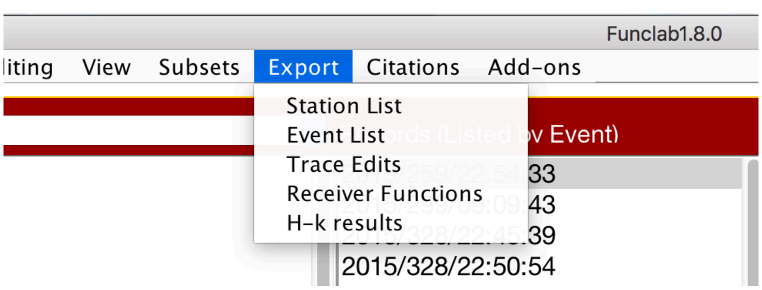
4.10 Subsets menu

The subsets menu is designed to allow a user to create a primary, large area project and then carve out smaller, more targeted projects. The Create New Subset tool has tables for selecting stations and events similar to the Fetching interface. It also contains RF parameter subsetting tools such as minimum fit, ray parameter window, back-azimuth window, or state of trace-editing. Also included are naming for the set and the option to choose an accent color to help quickly identify which set is active. When invoked, the subsetting tool only works on the metadata so that it is relatively fast and does not greatly inflate the size of the overall project. Once a subset has been created, the tools to delete a subset and choose a subset become active.



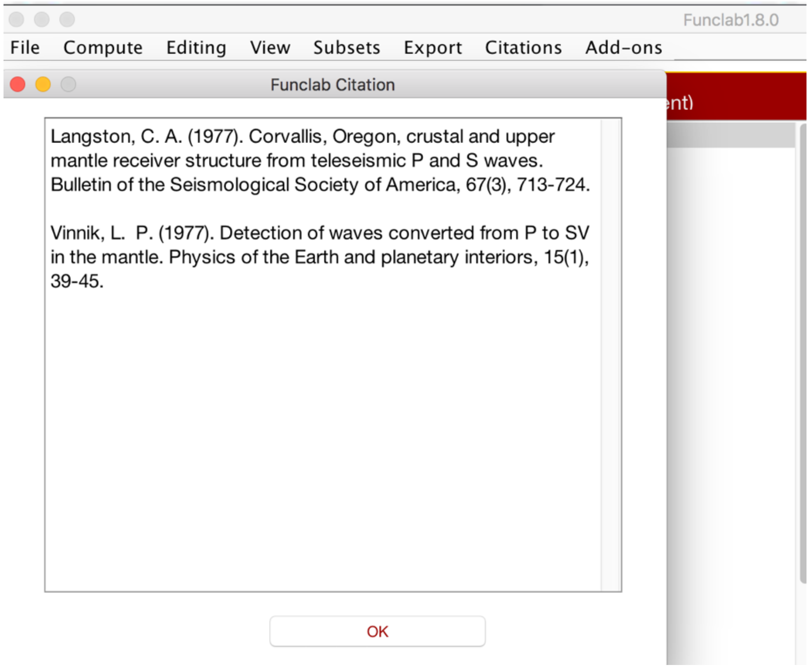
4.11 Export Menu

The Export menu allows the user to write data from the project file into simple text format files.



4.12 Citation Menu

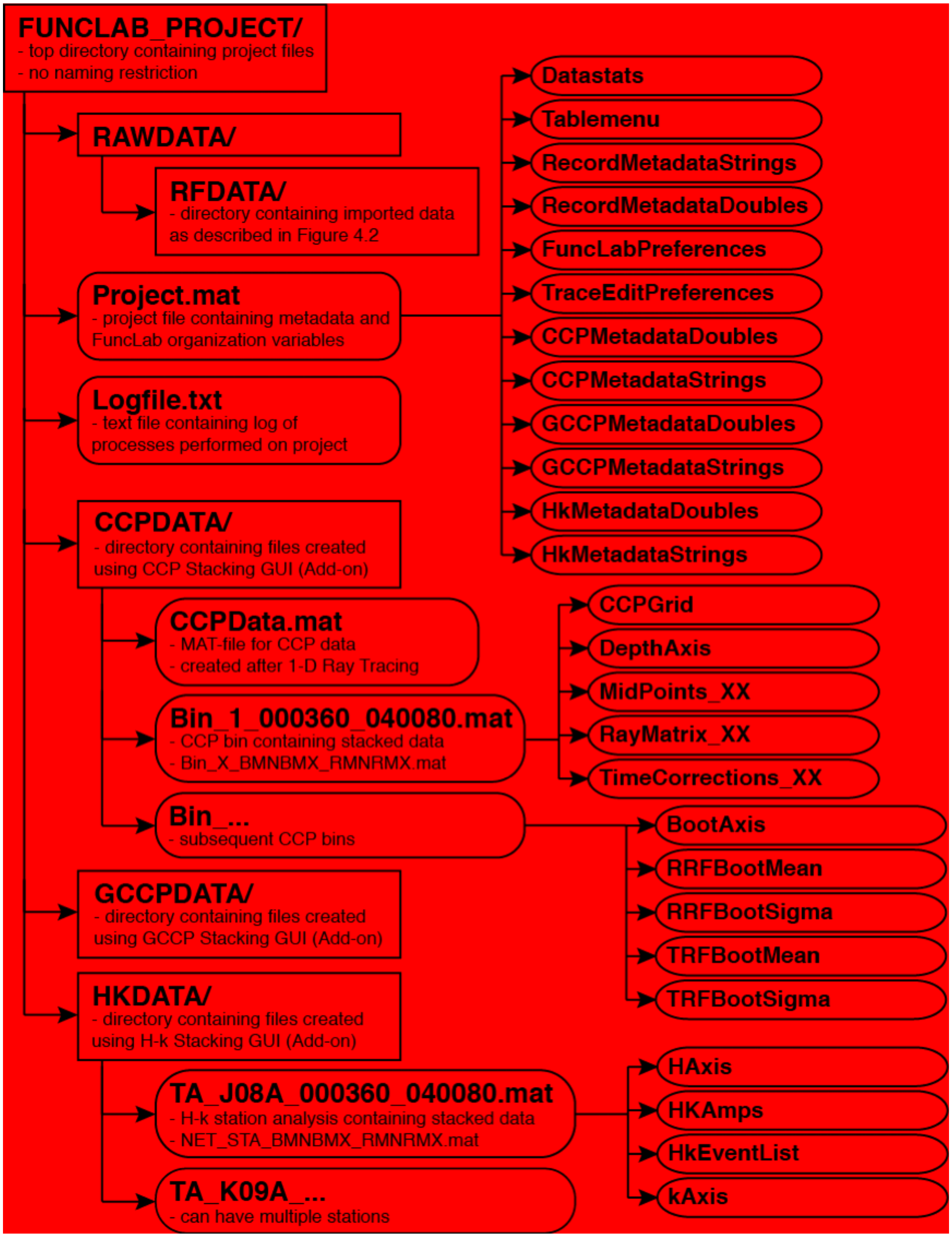
The Citation menu returns text boxes with some references that users may find useful.



More citations may be added later. Users are encouraged add their own as well.

5. Project Directory Structure and Description

A FuncLab "project" defines an entire dataset to be analyzed. It can contain multiple data directories as defined in section 3. A project directory contains all the raw data used by FuncLab, as well as files and directories it creates when a user initiates a new project. A new project is created within the FuncLab GUI by importing at least one data directory of the structure defined in section 3. The following shows the structure of the automatically generated project directory:



RFPROJECT (this is a generic name given for the purposes of illustration) is the top directory name, which the user specifies at the onset of a new project. The **RAWDATA** directory contains all the data directories that a user imports into the project. Notice that these are of the same structure as described in the previous section. The *Logfile.txt* is a text file that is used to store a log of all the processes run within FuncLab on a specific project. It contains a description of the process, parameters used, and time stamps. This is generated automatically and can be useful when retrieving a record of the different runs for each process. The *Project.mat* file is a MATLAB formatted data file that contains all the MATLAB variables used to reference the raw seismic data and metadata within a project. It is essentially the key "database" of the project. There are also variables contained in *Project.mat* that are used to communicate with the different GUIs of FuncLab and keep them talking to one another. When certain new figures are made, image files are saved in a new directory, **FIGURES/**. When the Compute->PRF to depth menu item is completed, a new directory called **DepthTraces/** is created and a .mat file is created to store all the depth mapped traces.

Other directories will be added under **RFPROJECT/** once certain add-on packages are initiated with a project. These new directories are **CCPDATA/**, **GCCPDATA/, HKDATA/, CCPV2/, and RAYP_BAZ_STACK/**. Each of these contains MAT-files storing variables created in MATLAB specific for the types of analyses performed. For example, the **CCPDATA/** directory contains *CCPData.mat* which stores variables that define the CCP grid and ray tracing information for each receiver function used in the stacks. It also contains a separate MAT-file for each CCP bin (e.g. *Bin_1_000360_040080.mat*) that stores variables defining the CCP stack for that. This is similar for **GCCPDATA/**. The **HKDATA/** directory contains a separate MAT-file for each *H*-κ stack (e.g. *TA_F10A_000360_040080.mat*). The first and second set of numbers in the bin and *H*-κ stack files refers to the backazimuthal range and ray parameter range in the stacking, respectively. **CCPV2/** is designed to be more simple common conversion point stacking routine and contains only two output .mat files. Both are defined by the user via standard save file dialog boxes and one contains the metadata about the CCP while the other contains the three-dimensional gridded volume. **RAYP_BAZ_STACK/** contains stacked mean and standard deviation SAC files for each back azimuth and ray parameter defined in the stacking interface.

6. Model Building and Setup

6.1. Velocity Models

FuncLab is currently only designed to handle 1D Earth models in a very simple way. This version of the package contains five commonly used 1D models including: PREM (*Dziewonski and Anderson*, 1981), IASP91 (*Kennett and Engdahl*, 1991), AK135 (*Kennet et al.*, 1995), VJMA2001 (The Japanese Meteorlogical Agency), and mTNA. These are contained under the **funclab1.8.0/VELMODELS/** directory within the text files *PREM.vel*, *IASP91.vel*, *AK135.vel*, *VJMA2001.vel, and mTNA.vel* respectively. Each contains three columns, where column 1 is depth, column 2 is Vp, and column 3 is Vs. Each row represents a new depth shell beginning at the depth defined in column 1.

Other models can easily be built to use in FuncLab. In order for FuncLab to recognize these, however, it is necessary to place the new velocity models in text files under the **funclab1.8.0/VELMODELS/** directory and with the extension *.vel*.

6.2. Tomography Models

Certain analysis packages, such as CCP and GCCP stacking, can utilize seismic tomography models for timing corrections with receiver function time-to-depth conversions or migrations. This version of FuncLab is distributed with a *P*-wave tomography model of the Pacific Northwest published in *West et al.*, 2009 and used for CCP stacking analysis in *Eagar et al.*, 2010. It is placed under the **funclab1.8.0/TOMOMODELS/** directory in the MAT-file *NWUS_P09b_1-2.mat*. Additionally, the GyPSuM model (Simmons et al., 2010) and the DNA13 (Porritt et al., 2014) are included. This file stores 6 variables used to define the model: *Depths*, *Lons*, *Lats*, *ModelP*, *ModelS*, and *TomoType*. The first 5 are 3x3 matrices, where the first, second, and third dimensions define x,y, and z (longitudes, latitudes, and depth) of the 3D volume. The *ModelP* and *ModelS* variables define the *P*- and *S*-wave perturbations or velocities, respectively at each point in the volume. *TomoType* either defines the model as a “perturbation” model or a “velocity” model to handle differencing the volume from the 1D velocity model used in the ray tracing.

As with velocity models, other tomography models can be built to be used in FuncLab. These new models should be place under the **funclab1.8.0/TOMOMODELS/** and have the extension, *.mat*.

7. Add-on Packages

These are special code packages that are optional with FuncLab running as the base. For instance, common-conversion point (CCP) stacking of receiver functions to look at mantle discontinuities is a common approach. Similarly, *H*-κ stacking of receiver functions for examining Moho depth and crustal Vp/Vs is another common analysis. FuncLab includes add-on packages for both of these analyses. Although these add-on packages are integrated with FuncLab, it is also separate from the FuncLab main code. This means that a core add-on package does NOT have to be altered to change things associated with the CCP package. Here we describe how to develop separate add-on packages for additional receiver function processing.

There is a specific directory structure needed for an add-on as well as a specific location relative to the FuncLab directories. In *funclab.m*, the path of the *funclab* command is searched for. That path should be the same path that the **flab** directory is housed. This is the directory that contains the entire core FuncLab code. Place the add- on directory in the **addons** subdirectory. For example, the **ccp** directory contains all the codes for the CCP add-on. All add-on code must be in a single subdirectory. FuncLab will not recognize any subdirectories beneath the main one (such as **addons > ccp > working**). As a point of organization, FuncLab includes m-files with the prefix of the add-on directory (e.g. *ccp_assign.m*).

The next component is an m-file under this add-on directory called *init_{add- on}.m*. Replace {add-on} with the name of the add-on. In the CCP add-on case, it would be called *init_ccp.m*. This file need only contain 1 line that looks like this:

```
uimenu(MAddons,'Label','Common-Conversion Point
Stacking','Tag','ccp_m','Enable','off','Callback',{@ccp_Callback});
```

The string after ‘Label’ is the name to be listed under the “Add-ons” toolbar in the main FuncLab GUI. ‘Tag’ should just be a unique name. In FuncLab, the add-on name is a prefix and the letter “m” which stands for “main”. ‘Enable’ should ALWAYS be off. ‘Enable’ is turned on eventually, but must be initialized as “off”. The ‘Callback’ is followed by {@*{add-on}*_Callback} and is important. This refers to a function (or m-file) by the same name.

This brings us to the third required component: a callback function that transfers the FuncLab GUI handles to your add-on GUI. This is essential to connecting the core package with the add-on. This callback function must be named *{add-on}_Callback.m* and must be in the add-on directory. As before, replace {add-on} with its proper name. In the CCP example, it is called *ccp_Callback.m*. It must contain 4 lines of code as follows:

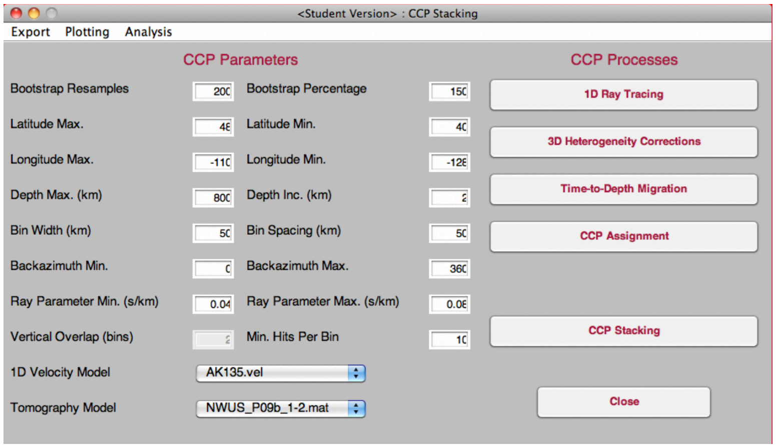
```
function ccp_Callback(cbo,eventdata,handles)

h = guidata(gcbf);

Project = get(h.project_t,'string');

ccp(h)
```

Replace “ccp_Callback” with the name of the new callback function, and replace “ccp(h)” with the m-file that begins the add-on GUI. The input “h” is the FuncLab GUI handles that MUST pass to the Add-on GUI for things to work properly.

The final step is to create this m-file that calls the Add-on GUI. It can have any structure. The only difference is that when you want to use handles that are associated with the main FuncLab GUI, you must refer to them in the input structure array. This is clearer using the examples provided with this manual.

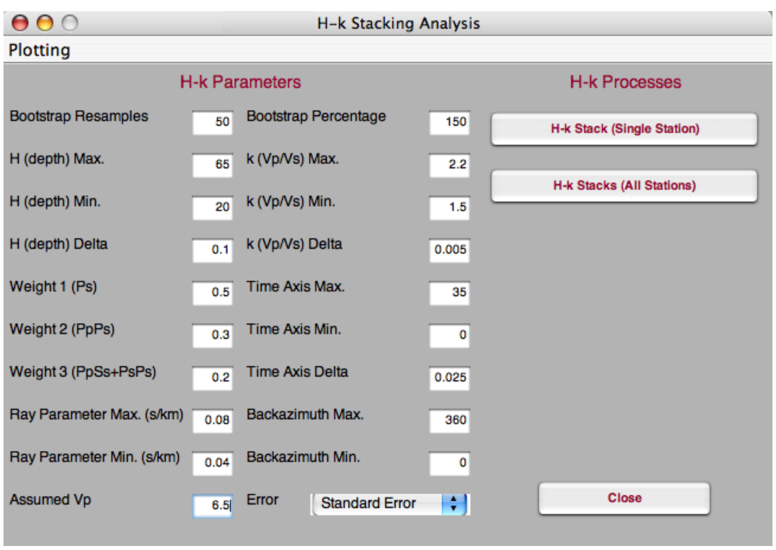
The CCP GUI contains editable parameters on the left-hand side and functional pushbuttons for data processing on the right-hand side. 1-D ray tracing creates the *RayMatrix_XX* and *MidPoints_XX* matrices that store the ray coordinates at each depth increment and half-depth increment, respectively. The *XX* is included in the matrix name to account for the possibility that more than one matrix may need to be generated, since each matrix is allowed to store up to 25,000 records to save memory in MATLAB. Both variables are stored in the *CCPData.mat* MAT-file under the created subdirectory in the project directory, *CCP*. The correction for 3-D heterogeneity is an optional step and requires the user to choose a tomography model, which can be defined or imported by the user as explained in the user manual. The *MidPoint_XX* matrix will be populated with timing corrections as described in *Eagar et al.* [2010] and are stored in the *TimeCorrections_XX* matrix, also saved to *CCPData.mat*. Time-to-depth conversion saves receiver function amplitudes in *RayMatrix_XX* that are now associated with each ray coordinate and depth increment. Each CCP bin saved in the *CCPGrid* matrix is assigned the amplitudes associated with it at each depth increment. CCP stacking results in the creation of two new arrays to store the CCP metadata, *CCPMetadataStrings* and *CCPMetadataDoubles*, which are saved to the *Project.mat* file. The stacking function also adds a row to the *Tablemenu* variable in the *Project.mat* file to control the drop- down menu in the main FuncLab GUI and allow the user to view records of CCP bins in the same manner as receiver function records.

The GCCP GUI is quite similar to the CCP GUI with only a few minor differences, such as the choice of the Gaussian width parameter and weights, which are



stored in the *RayMatrix_XX* variable. In all other variables and functions, *CCP* is replaced with *GCCP*. M-files for this add-on are found in the subdirectory *addons/gccp*.

Algorithms for *H*-κ stacking are included in m-files found in the subdirectory *addons/hk*. To perform a single station stack, the user selects the "*H*-κ Stack (Single Station)" button. All active records from the station selected from the station tables in the main FuncLab GUI will be stacked based on the parameters selected on the left-hand side of the *H*-κ Stacking Analysis GUI. Stacking one or all stations will create the *HkMetadataStrings* and *HkMetadataDoubles* arrays to store metadata for each *H*-κ stack that will be saved to the *Project.mat* file. Several visualizations to aid the user in the analysis are included in the package.



8. References

Abers, G. A., X. X. Hu, and L. R. Sykes (1995), Source scaling of earthquakes in the Shumagin region, Alaska: time-domain inversion of regional waveforms, *Geophys. J. Int.*, *123*, 41-58, doi: 10.1111/j.1365-1246X.1995.tb06660.x.

Bostock, M. G. (1996), Ps conversions from the upper mantle transition zone beneath the Canadian landmass, *J. Geophys. Res.*, *101*, 8393-8402.

Clayton, R. W., and R. A. Wiggins (1976), Source shape estimation and deconvolution of teleseismic body waves, *Geophys, J. R. Astron. Soc.*, *47*, 151-177.

Dziewonski, A.M., and D.L. Anderson (1981), Preliminary reference Earth model, *Phys. Earth. Planet. Int., 25*, 297-356.

Gurrola, H., G. E. Baker, and J. B. Minster (1995), Simultaneous time-domain deconvolution with application to the computation of receiver functions, *Geophys. J. Int.*, *120*, 537-543, doi: 10.1111/j.1365-1246X.1995.tb01837.x.

Eagar, K. C. and M. J. Fouch (2011), FuncLab: A MATLAB interactive toolbox for handling receiver function datasets, *Seis. Res. Lett.* in press.

Eagar, K.C., M.J. Fouch, and D.E. James (2010), Receiver function imaging of upper mantle complexity beneath the Pacific Northwest, United States, *Earth Planet. Sci. Lett., 297*, 141-153.

Kennett, B.L.N., and E.R. Engdahl (1991), Traveltimes for global earthquake location and phase identification, *Geophys. J. Int., 105*, 429-465.

Kennett, B.L.N., Engdahl, E.R., and R. Buland (1995), Constraints on seismic velocities in the Earth from travel times, *Geophys. J. Int., 122*, 108-124.

Langston, C. A. (1979), Structure under Mount Rainier, Washington, inferred from teleseismic body waves, *J. Geophys. Res.*, *84*, 4749-4762.

Ligorria, J. P., and C. J. Ammon (1999), Iterative deconvolution and receiver-function estimation, *Bull. Seism. Soc. Am.*, *89*, 1395-1400.

Park, J., and V. Levin (2000), Receiver functions from multiple-taper spectral correlation estimates, *Bull. Seism. Soc. Am.*, *90*, 1507-1520.

Porritt, R. W., R.M. Allen, and F. F. Pollitz. 2014. “Seismic imaging east of the Rocky Mountains with USArray.” Earth Planet. Sci. Lett. doi:10.1016/j.epsl.2013.10.034.

Sheehan, A. F., G. A. Abers, C. H. Jones, and A. L. Lerner-Lam (1995), Crustal thickness variations across the Colorado Rocky Mountains from teleseismic receiver functions, *J. Geophys. Res.*, *100*, 20,391-20,404.

Simmons, N.A., A.M. Forte, L. Boschi, and S.P. Grand. 2010. "GyPSuM: A joint tomographic model of mantle density and seismic wave speeds." J. Geophys. Res. 115:B12310.

Vinnik, L. P. (1977), Detection of waves converted from P to SV in the mantle, *Phys. Earth. Planet. Int.*, *15*, 39-45.

West, J. D., M. J. Fouch, J. B. Roth, and L. T. Elkins-Tanton (2009), Vertical Mantle Flow Associated with a Lithospheric Drip Beneath the Great Basin, *Nature Geoscience, 2*, 439-444.

APPENDIX A.

WEBSITE REFERENCES

Seismic Analysis Code (SAC): http://www.iris.edu/software/sac/manual.html Standing Order for Data (SOD): http://www.seis.sc.edu/sod/

Receiver Function Codes (Charles Ammon, Penn State):

http://eqseis.geosc.psu.edu/~cammon/HTML/RftnDocs/thecodes01.html Generic Mapping Tools (GMT): http://www.soest.hawaii.edu/gmt/ Earthscope Automated Receiver Survey (EARS): http://ears.iris.washington.edu/

MATLAB Mapping Data: • Coastlines

o gshhs_2.1.1: http://www.ngdc.noaa.gov/mgg/shorelines/data/gshhs/version2.1.1/gshhs_2.1.1.zip

• Topography o 1-arc-minute DEM

ETOPO1_ice_c_i2 (Ice Surface):

http://www.ngdc.noaa.gov/mgg/global/relief/ETOPO1/data/ice_surface/cell_registered/binary/etopo1_ice_c_i2.zip ETOPO1_bed_c_i2 (Bedrock Surface): http://www.ngdc.noaa.gov/mgg/global/relief/ETOPO1/data/bedrock/cell_registered/binary/etopo1_bed_c_i2.zip

o 2-arc-minute DEM ETOPO2v2c_i2_MSB: http://www.ngdc.noaa.gov/mgg/global/relief/ETOPO2/ETOPO2v2-2006/ETOPO2v2c/raw_binary/ETOPO2v2c_i2_MSB.zip

o 30-arc-second (1-km) grid DEM GLOBE DEM:
http://www.ngdc.noaa.gov/mgg/topo/DATATILES/elev/all10g.zip

APPENDIX B.

DESCRIPTION OF SAC FILES

SAC files for the receiver functions and seismograms may be produced in a variety of ways, but regardless of the method of preprocessing, some important formatting standards must be followed. This appendix describes the format and headers needed in each of the files.

FuncLab imports 5 separate SAC files for a given calculated receiver function. These include the vertical seismogram, radial seismogram, transverse seismogram, radial receiver function, and transverse receiver function. It is preferred that all 5 files are time synched to the zero time of the receiver functions. This makes plots in FuncLab make a lot more sense and prevents issues when interpreting the time axis.

Naming of the SAC files is an important first step. All SAC files have a naming convention as follows:

STA_2.5.i.z,

where "STA" is the station name, "2.5" represents the Gaussian lowpass filter (or any type of lowpass filter used), "i" means that we used "iterative deconvolution" to calculate the receiver function, and "z" is the seismogram component of motion. The station name and the extension are the most important items of the name. The extensions define the type of file as follows:

.z = vertical seismogram .r = radial seismogram .t = transverse seismogram .eqr = radial receiver function .eqt = transverse receiver function.

The SAC header definitions and descriptions for the seismograms are below:

NPTS = 4401 (*number of points in seismogram*)

B = -1.000000e+01 (*beginning time of seismogram*)

E = 1.000000e+02 (*end time of seismogram*)

IFTYPE = TIME SERIES FILE (*defines the type of file*)

LEVEN = TRUE (*defines the sampling of the seismogram as "even" – this is a MUST*)

DELTA = 2.500000e-02 (*sample rate of the seismogram*)

IDEP = UNKNOWN (*type of dependent variable – not necessary for our purposes although we assume it is a velocity seismogram*)

DEPMIN = -1.740033e+02 (*minimum amplitude in seismogram*)

DEPMAX = 1.473583e+02 (*maximum amplitude in seismogram*)

DEPMEN = 8.144172e-01 (*mean amplitude in seismogram*)

OMARKER = -721.03 (*number of points in the time series*)

T1MARKER = 0 (P) (*arrival time and marker of predicted P wave – this needs to be set as your zero time for the seismograms*)

KZDATE = AUG 14 (227), 2008 (*origin date of earthquake*)

KZTIME = 00:18:41.200 (*origin time of earthquake*)

IZTYPE = BEGIN TIME (*defines the reference time for this file – needs to be set to BEGIN TIME*)

KSTNM = G08A (*station code*)

CMPAZ = 0.000000e+00 (*component of motion azimuth – needs to be 0 for vertical, BAZ-180 for radial and radial RFs, and BAZ-90 for transverse and transverse RFs*)

CMPINC = 0.000000e+00 (*component of motion inclination – needs to be 0 for vertical and 90 for radial or transverse, including RFs*)

STLA = 4.529040e+01 (*station latitude*) STLO = -1.189595e+02

(*station longitude*) STEL = 1.318000e+03

(*station elevation in meters*) STDP = 0.000000e+00

(*station depth in meters*) EVLA = 1.640000e+01

(*earthquake latitude*) EVLO = 1.469000e+02 (*earthquake longitude*) EVEL = 0.000000e+00

(*earthquake elevation in meters*) EVDP = 5.300000e+04

(*earthquake depth in meters*) DIST = 9.048033e+03

(*distance in kilometers from station to earthquake*)

AZ = 4.540545e+01 (*azimuth from earthquake to station*)

BAZ = 2.844766e+02 (*backazimuth from station to earthquake*)

GCARC = 8.138271e+01 (*distance in degrees from station to earthquake*)

LOVROK = TRUE (*set to TRUE to overwrite the file*)

USER0 = 2.500000e+00 (*Gaussian parameter used to compute iterative deconvolution RF – see description of process below*)

USER1 = 3.028038e+02 (*predicted P wave ray parameter in sec/rad*)

USER8 = 0.000000e+00 (*predefines the trace edit status – this is OPTIONAL, but set to 0 if it is bad or 1 if it is good and you want to define it as an active RF*)

USER9 = 6.000000e+01 (*fit or match of the predicted waveform from iterative deconvolution – see description of process below*)

NVHDR = 6 (*header version number – current version is 6*)

LPSPOL = TRUE (*set to TRUE if component has positive polarity*)

LCALDA = TRUE (*set to TRUE if DIST, AZ, BAZ, and GCARC are calculated from station/earthquake coordinates*)

KCMPNM = BHZ (*component code – should be set to BHZ for vertical, BHR for radial and radial RFs, and BHT for transverse and transverse RFs*)

KNETWK = TA (*network code*)

MAG = 5.500000e+00 (*earthquake body wave magnitude*)

One note of caution: the CMPINC specifications in the description assume that the seismograms have been rotated in the direction of the earthquake along the plane of the surface. Rotations into true radial and transverse along the back-projected ray path will have a different inclination.

The iterative deconvolution code computes a receiver function by first performing a cross-correlation on the vertical and radial seismogram. Then, based on the lag time, convolves a spike train that represents that lag time with a Gaussian function. The Gaussian parameter determines the width of the Gaussian wavelet and essentially acts as a low-pass filter for the receiver function. After the first pulse is generated on the predicted receiver function, the receiver function is convolved with the vertical seismogram to get a predicted radial seismogram. The difference between the predicted

radial and the data is then computed to get a residual. As long as the residual is greater than a given threshold, the process continues. The first pulse is stripped and the next pulse is generated, again by the process of cross-correlation, convolution with a Gaussian wavelet to produce a predicted receiver function, and convolution of the receiver function with the vertical seismogram. More details can be found in *Ligorria and Ammon*, 1999.