FuncLab: MATLAB receiver function analysis toolbox
Version: 1.5
Author: Kevin C. Eagar

User Manual

Contents

1. ABOUT FUNCLAB

2. DOWNLOAD AND SETUP

3. RECEIVER FUNCTION DATA DIRECTORY STRUCTURE

4. PROJECT DIRECTORY STRUCTURE AND DESCRIPTION

5. MODEL BUILDING AND SETUP (VELOCITY AND TOMOGRAPHY MODELS)

6. ADD-ON PACKAGES

APPENDIX A. LIST OF M-FILES

APPENDIX B.

APPENDIX C. DESCRIPTION OF SAC FILES

1. About FuncLab

FuncLab comprises a set of tools built within the MATLAB environment. The 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 and reflect the most recent development on Red Hat Linux 5.3 running MATLAB 7.9.0 (R2009b). Recent testing was also conducted on Mac OS X 10.6.7 running MATLAB 7.10.0 (R2010a) without 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. The names and brief description of these scripts are as follows:

1. *ignoreNaN.m* is used to handle NaN values within matrices
2. *makeColorMap.m* is used to adjust color maps in figures
3. *mmpolar.m* creates a polar plot of ray parameter vs. backazimuth

2. Download and Setup

This section will introduce 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**.

FuncLab is distributed in a tar ball and can be accessed at http://geophysics.asu.edu/funclab. Download and untar *funclab.tar* in the directory in which you would like to place the MATLAB M-files by typing:

```
tar –xvf funclab1.5.tar
```

The directory **funclab1.5** will contain the files needed to run all the functions within the FuncLab toolbox. Appendix A contains a list of all the M-files. Note that data for mapping functions (requires the Mapping Toolbox) may be downloaded from outside data sources. Links to these 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 **funclab1.5/map_data/**.

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, the user must change to the `funclab1.5` directory, and then run the `setup_funclab` script:

```
cd funclab1.5
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 Data Directory Structure

In order for this toolbox to function in a more general sense with publicly available data, certain choices were necessary as to the data formats and directory structures expected by FuncLab. 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 and is left up to the user; however, keep in mind that I developed this code using an iterative deconvolution receiver function code written by Charles Ammon at Penn State University. The link to download this code is also posted on the FuncLab website. Also posted are some generic C-shell scripts that I originally used to run the preprocessing of data in this directory structure and output filenames appropriately. We will assume that all of these steps have been completed. Note that we are currently working with the IRIS DMS (http://www.iris.edu/dms/products/) to create a data product for direct import into FuncLab. It will piggyback onto the EARS project (http://ears.iris.washington.edu/) and perform all of the preprocessing steps on the front-end.

The data directories should be structured in the follow manner:

```
RFDATA/                          (top directory to contain data –can be named anything)
   |
   |----Event_2006_319_11_14_13/   (event directory – Event_YYYY_JJJ_HH_MM_SS)
         |
         |----TA_J08A_0.6.i.eqr    (radial receiver function file – NET_STA_GAUSSIAN.i.eqr)
         |----TA_J08A_0.6.i.eqt    (transverse receiver function file - NET_STA_GAUSSIAN.i.eqt)
```

```
        |----TA_J08A_0.6.i.r          (radial seismogram file - NET_STA_GAUSSIAN.i.r)
        |----TA_J08A_0.6.i.t          (transverse seismogram file - NET_STA_GAUSSIAN.i.t)
        |----TA_J08A_0.6.i.z          (vertical seismogram file - NET_STA_GAUSSIAN.i.z)
        |----TA_K09A...
        |       ...
   |
   |----Event_...
   |       ...
```

Each file begins with the station name/code, contains the Gaussian parameter used to compute the receiver function, and ends with *i.eqr*. I have included files for both the radial and transverse receiver functions for completeness. I also include the radial, transverse, and vertical seismograms to exploit some of the plotting features within the FuncLab toolbox. All five of these files are required for a single station and single event.

Since FuncLab was originally designed for iterative time-domain receiver functions, the SAC files include the fit of the predicted waveform in the USER9 header, the Gaussian parameter in the USER0 header, and the P wave ray parameter (in sec/rad) in the USER1 header. An additional setting for pre-edited data has been added to the USER8. If USER8 is set to 1, the receiver function is marked as "turned on" and is used in the rest of the analyses. If USER8 is set to 0, the receiver function is marked as "turned off" and is discarded from analyses. If USER8 is not set, the default status is set to 1 and "turned on".

4. 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. For example, when developing this code, my work was focused on the Pacific Northwest US. I had collected data from a variety of networks and sources with seismic stations within this study area. These data were actually contained in multiple directories (e.g. TA network data from 2010 separate from ANSS network data from 2009), but were all part of the same "project" in FuncLab.

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/                              (top directory – no naming restriction)
   |
   |----RAWDATA/                        (data directory for the imported SAC files)
        |
        |----RFDATA/                    (same as RF data directory structure)
           |
           |----Event_2006_319_11_14_13/   (event directory)
                |
                |----TA_J08A_0.6.i.eqr      (radial receiver function)
```

```
                |----TA_J08A_0.6.i.eqt          (transverse receiver function)
                |----TA_J08A_0.6.i.r            (radial seismogram)
                |----TA_J08A_0.6.i.t            (transverse seismogram)
                |----TA_J08A_0.6.i.z            (vertical seismogram)
                |----TA_K09A...
                |       ...
        |----Event_...
        |       ...
|
|----Logfile.txt                                (log file)
|----Project.mat                                (project file)
|
|----CCPDATA                                    (data directory for the CCP stacks)
        |
        |----CCPData.mat                        (data file – contains rays and depth axes)
        |----Bin_1_000360_040080.mat            (stack file – Bin_BIN#_BAZ_RAYP.mat)
        |----Bin_2_000360_040080.mat
        |       ...
|----GCCPDDATA                                  (data directory for the GCCP stacks)
        |
        |----GCCPData.mat                       (data file – contains rays and depth axes)
        |----Bin_1_000360_040080.mat            (stack file)
        |----Bin_2_000360_040080.mat
        |       ...
|----HKDATA                                     (data directory for the H-κ stacks)
        |
        |----TA_J08A_000360_040080.mat          (stack file – NET_STA_BAZ_RAYP.mat)
        |----TA_K09A_000360_040080.mat
        |       ...
```

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.

Other directories will be added under **RFPROJECT/** once certain add-on packages are initiated with a project. These new directories are **CCPDATA/**, **GCCPDATA/**, and **HKDATA/**. 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 bin. 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.

5. Model Building and Setup

Velocity Models

Velocity models are an essential part of any seismic analysis. FuncLab is, at this point, only designed to handle 1D Earth models in a very simple way. This version of the package contains three commonly used 1D models: PREM[1], IASP91[2], and AK135[3]. These are contained under the **funclab1.5/VELMODELS/** directory within the text files *PREM.vel*, *IASP91.vel*, and *AK135.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 **funclab1.5/VELMODELS/** directory and with the extension *.vel*. Otherwise FuncLab will not be able to find these models.

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[4] and used for CCP stacking analysis in Eagar et al., 2010[5]. It is placed under the **funclab1.5/TOMOMODELS/** directory in the MAT-file *NWUS_P09b_1-2.mat*. 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

---

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

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

[3] 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.

[4] 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.

[5] 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.

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 **funclab1.5/TOMOMODELS/** and have the extension, *.mat*.

6. Add-on Packages

These are special code packages that are optional with FuncLab running as the backdrop. For instance, I want to perform common-conversion point stacking with some receiver function dataset to look at mantle discontinuities. I have created a CCP Add-on package that performs tasks that help me accomplish my goal and look at the data in a way that makes sense. Although this package is integrated with FuncLab, it is also separate from the FuncLab main code. This means that the core package does NOT have to be altered to change things associated with the CCP package. It means that anyone can write their own separate add-on for whatever process they want to perform.

There is a specific directory structure needed for an add-on and specific location to put it 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 all of the core FuncLab code. Place the add-on directory in the **addons** subdirectory. For example, my **ccp** directory contains all the codes for this add-on. It is placed under **addons**, which is the same directory level as the **flab** directory and *funclab.m*. Make sure all the code that you use for the add-on is in this directory. It will not recognize any directories UNDER the main one though (such as **addons > ccp > working**). As a point of organization, I prefer to name 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 my 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 that you want it to read under the "Add-ons" toolbar in the main FuncLab GUI. 'Tag' should just be some name that is unique. I use the add-on name as a prefix and the letter "m" which stands for "main". 'Enable' should ALWAYS be off. Don't worry, it does get turned on eventually, but you don't need to worry about this. The 'Callback' is followed by {@ccp_Callback} which is important. Keep this the same, except the word "ccp_Callback". This refers to a function (or m-file) by the same name. So replace "ccp" with whatever your add-on name is.

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 my case, of course, 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 your callback function. And replace "ccp(h)" with your 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 if you look at the examples provided with this manual.

APPENDIX A.

LIST OF M-FILES

setup_funclab.m
funclab.m

addons

- **ccp**
  - ccp_1D_raytracing.m
  - ccp_assign.m
  - ccp_binmap.m
  - ccp_binstack.m
  - ccp_bootstrap.m
  - ccp_Callback.m
  - ccp_corrections.m
  - ccp_ex_pp.m
  - ccp_ex_stacks.m
  - ccp_fieldnames.m
  - ccp_migration.m
  - ccp_readamps.m
  - ccp_record_fields.m
  - ccp_setup_grid.m
  - ccp_stacking.m
  - ccp.m
  - init_ccp.m
- **gccp**
  - gccp_1D_raytracing.m
  - gccp_assign.m
  - gccp_binmap.m
  - gccp_binstack.m
  - gccp_bootstrap.m
  - gccp_Callback.m
  - gccp_corrections.m
  - gccp_ex_pp.m
  - gccp_ex_volume.m
  - gccp_fieldnames.m
  - gccp_migration.m
  - gccp_readamps.m
  - gccp_record_fields.m
  - gccp_setup_grid.m
  - gccp_stacking.m
  - gccp.m
  - init_gccp.m
- **hk**
  - hk_allstack.m

hk_Callback.m
hk_ex_grids.m
hk_ex_solutions.m
hk_record_fields.m
hk_singlestack.m
hk_view_baz_image.m
hk_view_hkgrid.m
hk_view_moveout_image.m
hk_view_stacked_records.m
hk.m
init_hk.m

custom

errordlg_kce.m
movegui_kce.m
questdlg_kce.m
uigetdir_kce.m
uigetfile_kce.m

flab

fl_add_data_gui.m
fl_add_new_data.m
fl_datainfo.m
fl_datastats_gui.m
fl_default_prefs.m
fl_edit_logfile.m
fl_init_logfile.m
fl_log_gui.m
fl_pref_fieldnames.m
fl_preferences_gui.m
fl_readseismograms.m
fl_record_fields.m
fl_traceedit_gui.m
fl_view_baz_image.m
fl_view_baz.m
fl_view_data_coverage.m
fl_view_event_map.m
fl_view_moveout_image.m
fl_view_moveout.m
fl_view_record_section.m
fl_view_record.m
fl_view_stacked_records.m
fl_view_station_map.m
te_default_prefs.m
te_pref_fieldnames.m
te_preferences_gui.m
te_view_record.m

map_utilites

azimuth.m
deg2km.m
deg2rad.m
gg2gc.m
km2deg.m
latlon_from.m
rad2deg.m

utilities

colvector.m
ignoreNaN.m
makeColorMap.m
mmpolar.m
skm2srad.m
srad2skm.m

APPENDIX B.

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
  - gshhs_2.1.1:
    http://www.ngdc.noaa.gov/mgg/shorelines/data/gshhs/version2.1.1/gshhs_2.1.1.zip
- Topography
  - 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
  - 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
  - 30-arc-second (1-km) grid DEM
    GLOBE DEM:
    http://www.ngdc.noaa.gov/mgg/topo/DATATILES/elev/all10g.zip

APPENDIX C.

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*)
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*)
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.