A PRACTICAL GUIDE (2.0)

Receiver Function Analysis using Multitaper Spectral Correlation Estimates

Relevant background reading: Park and Levin (2000) and Park and Levin (2016a, 2016b)

This guide is developed by current and former members of Rutgers Seismology group (Xiaoran Chen, Yiran Li, James Bourke), with contributions from Zhenxin Xie, and encouragement from Prof. Vadim Levin (Rutgers) and code's author Prof. Jeffrey Park (Yale). It is based on the previous guide to J. Park's codes by (then) members of Rutgers Seismology group Alex Nikulin and Ben Marshall.

Instructions described in this manual are for

1) downloading and using SAC files fetched by PyWEED,
2) performing initial quality control using Seisgram2K,
3) running basic receiver function analysis,
4) running receiver function analysis with harmonic decomposition

ESSENTIAL NOTE 1: This guide will explain *how* to perform data analysis, but not *why* this analysis is done, that part is a responsibility of the user.

ESSENTIAL NOTE 2: To get a better idea of what the goals of receiver function analysis are – read the background papers and references therein. To understand what the codes and scripts do to get the analysis done – read them.

Required software

SAC – Seismic Analysis Code, available from IRIS (http://ds.iris.edu/ds/nodes/dmc/forms/sac/)
GMT – Generic Mapping Tools, command line graphic package maintained by the University of Hawaii (https://www.generic-mapping-tools.org/); Scripts used by this distribution rely on GMT version 5.
PyWEED – GUI application for data retrieval from IRIS DMC; built on ObsPy (https://ds.iris.edu/ds/nodes/dmc/software/downloads/pyweed/)
Any other methods of data retrieval are fine, as long as we ensure that required fields in the SAC header are turned on or filled before the data analysis.
Seisgram2K – an easy-to-use, platform-independent, Java software package for interactive visualization and analysis of earthquake seismograms. SeisGram2K runs and reads data files locally and over the Internet(http://alomax.free.fr/seisgram/SeisGram2K.html)

FORTRAN compilers and necessary libraries – Codes described here are developed and compiled on Mac OS X operating system, with Developer Tools libraries, and **gfortran** compiler from the *High Performance Computing for Mac OS X* site (*http://hpc.sourceforge.net/index.php*)
Supporting scripts are developed in ***csh*** environment on a Mac OS X workstation. These should work on any UNIX/Linux implementation however do check for variants in shell-script dialect.

Codes included in this distribution

MTC Receiver Functions code:

rfmigrate_21.f – Computes RFs for BAZ and EPI gathers, applies moveout corrections ("migration" according to J. Park) and shifts target depth. **If a target depth of 0 is adopted, then normal receiver functions for BAZ and EPI gathers will be computed.**

Harmonic Expansion code:

rfmig_mcboot_21.f – computes directional expansions of combined R + T (shifted) components, producing "anisotropy/dip" and "unmodelled" harmonic stacks; moves the target depths. Note that the program expects the vertical, radial and transverse components of seismic data to form a right-handed coordinate system. A key feature of this system is that (transverse) = (vertical) x (radial), that is, the transverse unit vector is the vector cross-product of the vertical and radial unit vectors. Some seismic processing codes, e.g., ObsPy, define a "transverse" component with opposite polarity, forming a left-handed relationship (vertical) x (radial) = -(transverse).

NOTE: These codes are written in FORTRAN and require compilation by **gfortran** using the following command:

gfortran xxxx.f -o xxxx (choose an output name)

The compiled codes included in this distribution, ***rfmigrate_21*** and ***rfmig_mcboot_21***, expect the path to TauP files in the source codes is consistent with where those files are stored in the file system. Alternatively, if you wish to keep the TauP files in a separate directory (e.g. with the rest of the codes and scripts), you may also compile the source codes yourself based on your local set-up. To make changes, simply search for the string ".dat" in the source code and insert the path after "file=". The search should yield 3 hits (taup_P.dat, taup_PP.dat and taup_PKP.dat).

After compilation, update the $PATH variable of your shell to include the RECFUNK folder so the compiled codes can be easily accessed from the command-line interface.

Service scripts:

rename_pyweed.csh – renames the long file names default to SAC files retrieved through PyWeed
make-rotate-pyweed.csh – creates SAC macro to rotate the 3 components to RTZ coordinate system
make-pick-macro-pyweed.csh – creates SAC macro for picking P-arrivals
make-recpick-input.csh – creates a list of selected SAC files (in_recpick) that ***rfmigrate_21*** will process
bexp_ad.plot.txt – a GMT script to plot harmonic decomposition results
plotbaz-short – a GMT script to plot the back azimuthal spreads of the computed receiver functions
plotepi – a GMT script to plot the epicentral spreads of the computed receiver functions

plotepi_baz – a GMT script to plot the epicentral spreads of the computed receiver functions with customized back azimuthal ranges
procedure_C_epi – generates migrated receiver function epicentral gathers with customized back azimuthal ranges
make-*-RF.C.txt – a family of scripts to automate multiple receiver function analyses with different frequency contents
make-*-RF.C.epi.txt – a family of scripts to automate multiple receiver function analyses with different frequency contents but with customized back azimuthal ranges

SAC header viewing utility **sachdr** is helpful. It comes from a software package developed by PASSCAL (https://www.passcal.nmt.edu). It is written in C. A folder is included with this distribution

1) DATA RETRIEVAL VIA PYWEED

All procedures below anticipate that processing is done one site at a time. Any way of data retrieval is acceptable, though it is important to ensure that some fields in the SAC metadata are filled to avoid problems in the later processing steps.

Key criteria for data necessary to compute tele-seismic receiver functions with ***rfmigrate _21*** and ***rfmig_mcboot_21*** codes are:

1) 3 component records of compressional waves (P, PKP, PKIKP) are from earthquakes at distances 20–180 degrees from the receiver, in SAC format. Ideally, time windows must include the target wave, at least 100 s of record after it, but better longer, and an equal amount of time *before* it.

2) Information about locations of the source (earthquake) and the receiver. These have to be inserted into the headers of the SAC files before most of the preprocessing steps.

PyWEED provides an intuitive GUI for selecting sites and events. The data are most easily retrieved on a station-by-station basis. Once the parameters for station and event query are set on the right and the left panel, hit "Get Stations" and "Get Events" buttons in the middle to gather lists of station and events that fulfill the requirement. To retrieve the waveforms, highlight by clicking on the appropriate channels and events in the list. "Get Waveforms" button becomes active once the events and channels are highlighted; click on it, and a new window will pop up to show the fetched traces.

The fetching could take up some time, depending on the amount of data you are requesting. The top left panel of the fetching window allows you to adjust the length of the displayed/downloaded traces. A quick initial quality control can be done based on trace view. Check for

a) Visibility of records
b) Noise and gaps in data
c) Missing components

d) Consistency of start and end time, as well as the length

Etc.

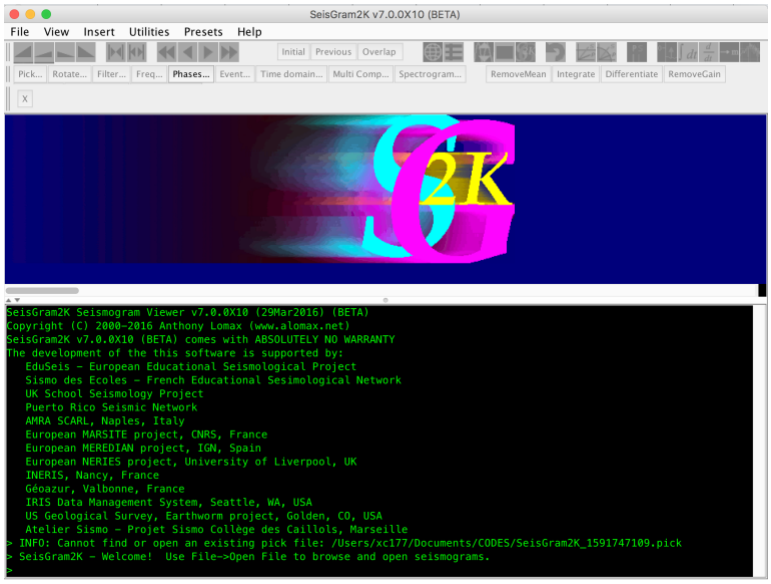
If you decide to disregard a trace for any of the above reasons, remove the check mark from the "Keep" box on the left to omit it from the download list. Once the quality control is completed, download the traces through the options in the top right panel of the fetching window.

Depending on the amount of data, there are alternative options for quality control. One is described below.

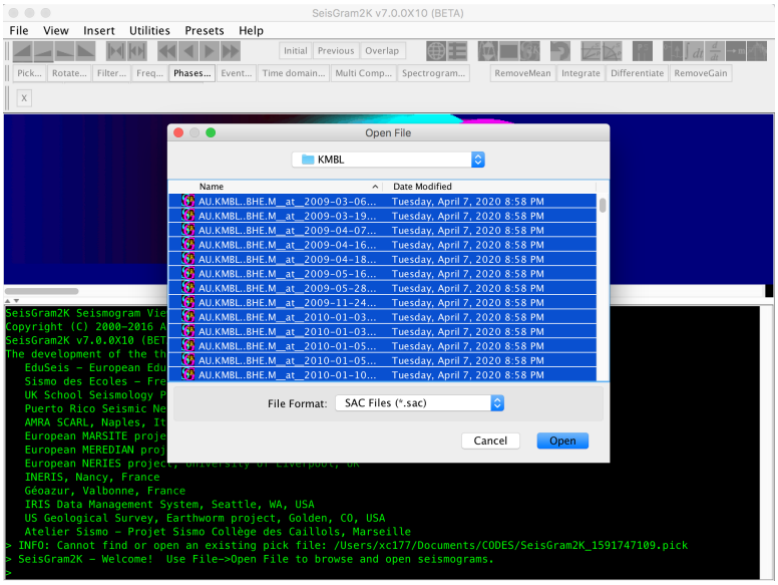
2) INITIAL QUALITY CONTROL WITH SEISGRAM2K

Use ***Seisgram2k*** to examine the dataset as the first round of quality control especially when large volumes of data are downloaded and the analyst does not have enough time to examine them before downloading.

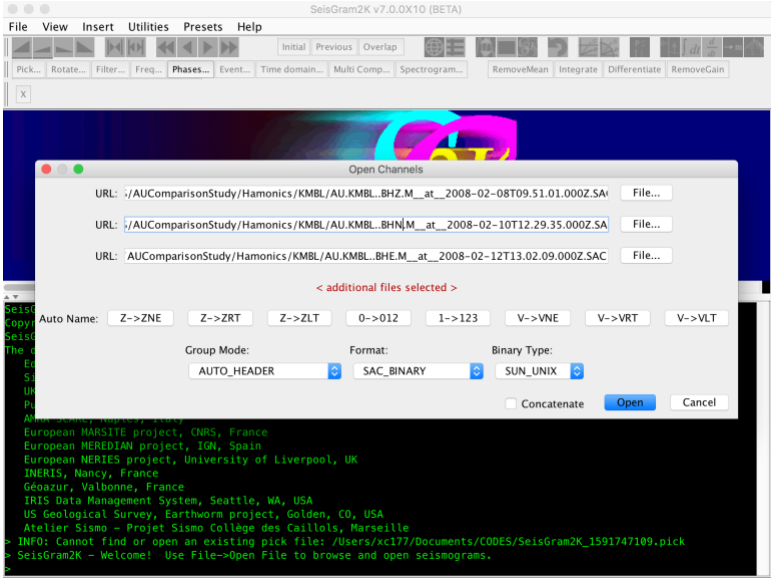
Seisgram2k is a java software package with a GUI as below:



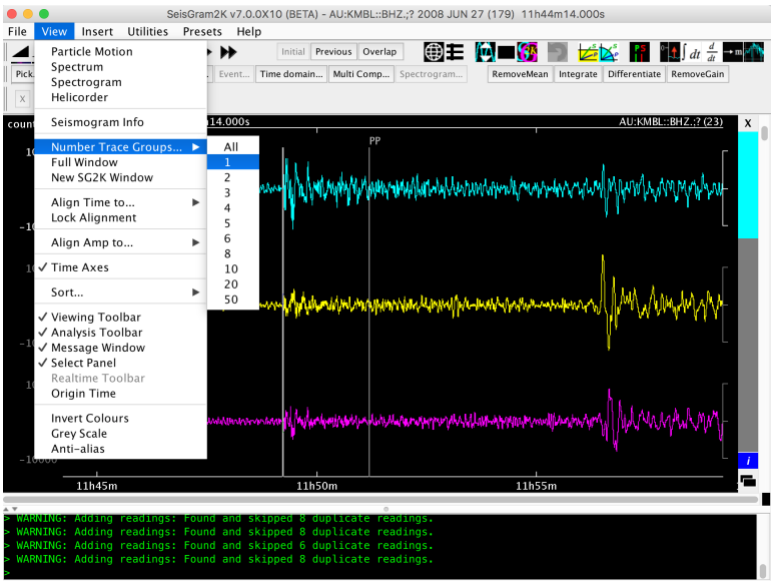
1. Start ***Seisgram2k*** by double clicking the icon or typing to search.
2. Click 'File', then select 'Open File...' and then a window will pop up to let you choose the directory where the data are saved. Choose the corresponding directory and make sure to select all the *.sac* files as shown in the screenshot. Click 'Open'.



3. Manually revise the components to be *?HZ*, *?HN* and *?HE* as shown below and click ‘Open’.



4. Change the view option as shown below:



5. Now it is the time to analyze the records. Different options of preliminary data processing tools are provided in the ribbon including filtering, picking phases, demean, detrend… If you would like to keep the record, just drag the bar on the right to move to the next one, otherwise click the 'x' next to the bar and this will automatically exit out the traces you do not want. Standards are the same as the quality control procedure in SAC.
6. After going through all the records, save all picked data in a new folder as below (a little trick here: you need to make sure that you make a new folder first before saving the records in it):

7. Now in the new folder, all the traces should have 3 components and same durations.

3) PRE-PROCESSING

A repeat NOTE: following procedures assume work is done on one site at a time. It helps to split data into site-specific folders.

3a) Renaming SAC files

Important: By default, the SAC files downloaded through ***PyWEED*** have empty fields for BAZ (back azimuth) and GCARC (epicentral distance) in the header. This will cause problems later for rotating the horizontal components into the Radial and Transverse coordinates. As long as the event location (EVLA and EVLO) and station location (STLA and STLO) are filled, SAC can calculate BAZ and GCARC; this is enabled by turning on the LCALDA option in the header. To do so, in the terminal:

Navigate to the directory storing all SAC files, start SAC, read files into the working memory, and type:

SAC> chnhdr LCALDA TRUE
SAC> lh LCALDA (to check if the field changed to TRUE)
SAC> write over
SAC> lh BAZ GCARC (to make sure the fields are calculated)

PyWEED also writes ridiculously long file names for the SAC files; we therefore recommend renaming them according to a more sensible format: YYYYMMDDHHMM.C.STNM.SAC, where YYYY is year, MM is month, DD is day, HH is hour, and MM is minutes. C stands for channel (E, N, Z) and STNM stands for station name. Run the renaming script by typing in the command line:

csh rename_pyweed.csh STNM

STNM stands for the station name that the script takes as the input. This will generate a new script called **do-rename**, which consists of renaming commands for all of the SAC files. Examine that script in a text editor to check for any obvious syntax or naming errors. Running the **do-rename** is what actually performs the renaming. Type in the command line:

csh do-rename
ls (to check if the rename was successfully done)

Now the file names should look much neater.

3b) Rotating components to RTZ coordinate system

The pre-processing consists of 4 steps:

1) rotation to RTZ (R – radial, T – transverse, and Z – vertical) coordinate system
2) first-order demean and detrend or the traces
3) applying high-pass filter at 50 s period
4) generating a macro for picking P wave arrivals in SAC

Steps 1, 2, and 3, are done simultaneously in the script called **make-rotate-pyweed.csh.** To run, simply type in the command line:

csh make-rotate-pyweed.csh STNM

STNM again stands for the input station name. It generates a file called **rotate.macro**, which is a macro to be run in SAC. It also generates a file called **list_of_files,** which will be used in step 4.

The **rotate.macro** file should have a set of lines like the following for each event:

r 201309210150.E.STNM.SAC 201309210150.N.STNM.SAC (reading horizontal components into SAC memory)
rtrend (SAC command for removing linear trend)
rmean (SAC command for removing mean)
highpass corner 0.02 npoles 4 (highpass filter at 50 s period)
rot to gcp reversed (rotating to RTZ coordinates; "reversed" is a choice that makes a right-handed RTZ system)
w 201309210150.STNM.r 201309210150.STNM.t (writing the radial and transverse components)
r 201309210150.Z.STNM.SAC (reading the vertical component into SAC memory)
rtrend (SAC command for removing linear trend)
rmean (SAC command for removing mean)
highpass corner 0.02 npoles 4 (highpass filter at 50 s period)
w 201309210150.STNM.z (writing the vertical component)

The rotated components are generated as separate SAC files ending with *.[rtz]*. To run the macro, type in the command line:

sac (launch SAC)
SAC> macro rotate.macro

This will send a lot of output to the screen. If trouble arises – SAC sends error messages to the screen and quits. It is helpful to check the last output on the screen to see where the macro crashed.

A common error would be uneven data length; depending on the way the traces are cut in the data retrieval software, we often see component length mismatching by a single sample point (you can check this by using **lh npts** command in SAC). To fix this problem, identify the event that caused the macro to crash (the last entry before the error message) and search for the corresponding set of lines in the macro, and insert the bolded in the following:

cut b n xxxxx
r 201309210150.E.STNM.SAC 201309210150.N.STNM.SAC
rtrend
Rmean
...
highpass corner 0.02 npoles 4
w 201309210150.STNM.z

cut off

xxxxx here should be the shortest sample length among the three channels. Inserting these would instruct SAC to only read up to that sample point for the event. Should you decide to disregard the event completely (e.g. because the difference is too large), simply remove the whole set of lines relevant to that event. For some reason, repeatedly running rotate.macro can progressively slow down the machine. In such case, simply quit and relaunch SAC and get back to running the macro.

Once the rotated files are successfully generated, move onto generating macro for picking P arrivals using the script **make-pick-macro-pyweed.csh**. To run, type in the command line:

csh make-pick-macro-pyweed.csh list_of_files

List_of_files is again the file produced in the previous step. This script will generate a macro called **list_of_files.sac.macro.** Each line consists of SAC commands to display the traces by events and allow you to interact with the window to place picks at the times of P arrivals. It may be worth renaming it to **STNM.list_of_files.sac.macro** or something similar to keep track of what file belongs to which station.

3c) Picking P Arrivals

By picking arrivals, you place markers in the SAC header for the ***rfmigrate_21*** or ***rfmig_mcboot_21*** code to find the times to begin analysis. To launch the graphical interface for picking, type in the command line:

sac
SAC> qdp off; macro list_of_files.sac.macro (***qdp off*** makes waveforms look nicer)

This macro employs a ***plotpk*** interactive picking function in SAC. It is invoked for each triplet of traces, allowing user to inspect the timeseries and to decide where to place a time mark that will serve as the start of the analysis time window. Being familiar with this function before running the macro helps.
FORTRAN codes will treat the values in the record before the time mark as "noise", after the mark - as "data".

Identify P arrivals and place the cursor to where you wish to place a marker. Then press "t" and either 1, 2, or 3 to assign correct phases; the marker should appear on the screen. The **rfmigrate_21** expects T1 marker to be P, T2 marker to be PKP, and T3 marker to be PP. Proceed to the next record by pressing Q. If you wish to disregard a particular event, also simply press Q to move onto the next record; this would allow extra quality control.

If you make a mistake and place a wrong mark into a file, simply jot down the event and revisit later. First, remove the wrong marker in SAC with the following:

sac
SAC> r xxxxxxxxxxxxx*.? (xxxxxx corresponds to the event)
SAC> chnhdr T1 undef (removes the marker)

SAC> write over

Then copy the line of command in **list_of_files.macro** that corresponds to the particular event and run in SAC. The graphical interface should appear again for you to re-pick using the same above procedure. Go through all of the records.

4) COMPUTE RECEIVER FUNCTION

First, a final input file for the receiver function code needs to be prepared. The input file is hardwired to be named "in_recpick" and lists all SAC files that the code will use to compute the receiver functions time series. To generate the list, we use the script named **make-recpick-input**, which requires you to have **sachdr** code and added to the shell path. To run, type in the terminal:

csh make-recpick-input list_of_files

The script will check headers and generate **in_recpick** based on the time markers in the SAC header. It will ignore files without T marks in the header.

If you use this, ensure that a word "stop" is present as the very last line at the end of the **in_recpick** file.

In addition to **in_recpick** file, the ***rfmigrate_21*** and ***rfmig_mcboot_21*** codes also require an input of velocity model in the format of Jeffrey Park's codes (named, for example, SITE-model, where SITE is the station name; this name has to be provided to the codes at input). Moreover, in the same folder where all the data are saved, make sure you put 'SITE-model' and .dat files (taup_P.dat, taup_PP.dat, taup_PKP.dat) in the same directory, otherwise you will get an error.

P_0.05_ani_tilted_vs_horizontal_axes <-- alphanumeric string title of the model
3 <-- number of layers above the halfspace
0 0 <-- symmetry-axis orientation in layer #1 theta,phi, theta angle from vertical, phi CW from N
20000 6000 -0.0 0.00 3450 -0.0 2650 <-- depth of lower interface of layer #1 (meters), Vp (m/s), B, C, Vs (m/s), E, density (kg/m^3)
45 45 <-- symmetry-axis orientation in layer #2 theta,phi, theta angle from vertical, phi CW from N
30000 6300 -0.05 0.00 3550 0.00 2800 <-- depth of lower interface of layer #2 (meters), Vp (m/s), B, C, Vs (m/s), E, density (kg/m^3)
90 90 <-- symmetry-axis orientation in layer #3 theta,phi, theta angle from vertical, phi CW from N
40000 6600 -0.05 0.00 3650 0.0 2900 <-- depth of lower interface of layer #3 (meters), Vp (m/s), B, C, Vs (m/s), E, density (kg/m^3)
0. 0. <-- symmetry-axis orientation in halfspace theta,phi, theta angle from vertical, phi CW from N
72000 8300 0.0 0.00 4500 0.0 3200 <-- "depth" of halfspace (NOT USED, so any number is OK), Vp (m/s), B, C, Vs (m/s), E, density (kg/m^3)

Once you have both inputs, you are ready to run the receiver functions analysis. Run the compiled ***rfmigrate_21*** code by simply typing in the command line:

rfmigrate_21

given that the path to the compiled code is known to the system. If successfully executed, the code will prompt you to enter necessary parameters for computing the receiver functions. For each prompt, type your answer and press enter. It is advisable to try this at least once to see how the code behaves. WARNING – the code sends lots of output to the command window, this is NORMAL. In routine use described below this output is captured into a file.

At each run the receiver function code ***rfmigrate_21*** generates three sets of files, namely: two back-azimuthal sweeps of receiver functions called ***outr_baz.grid, outt_baz.grid*** (for R and T component, respectively) without move-out correction, two back-azimuthal sweeps ***oumt_baz.grid, oumr_baz.grid*** with move-out correction, and two epicentral sweeps ***oumt_epi.grid, oumr_epi.grid*** with move-out correction. All output files are ASCII tables formatted for GMT plotting. To plot the outputs, run the GMT scripts **plotbaz-short** for back azimuthal spreads, and **plotepi** for epicentral spreads. You will need a PostScript viewer (Preview on a Mac works, as does Acrobat Reader) to see the plotted results. The time '0' s in the plotted results correspond to the selected migration target depth. The time series is shifted if target depth of greater than 0 km is adopted. In this guide, we choose 50 km as the target depth and hardwired it in the output file names, you can choose other depths and modify the file names correspondingly.

More often than not, the user wishes to run multiple receiver function analysis with slightly different parameters (e.g. the maximum frequency in the filter applied to produced receiver functions). To automate multiple analysis using **rfmigrate_21**, the user can use **make-mo-RF.C.txt**, which runs multiple receiver analysis as well as the GMT plotting scripts. You can manually edit these parameters to suit your needs according to the annotations below.

```
#Input - 1:name of the site 2:fmax
#name of the model
echo $1-model >! rfmigrate_run
#depth to which migration will be done (use 0 for normal gathers)
echo 50 >> rfmigrate_run
#fmax
echo $2 >> rfmigrate_run
#time intervals in SAC header
echo 1 >> rfmigrate_run
#name of pick file
echo in_recpick >> rfmigrate_run
#duration of data windows
echo 80 >> rfmigrate_run
```

#min number of traces in the averaging bin
echo 2 >> rfmigrate_run
#rotate to LQT? -1 = YES, and change the value
echo -1 >> rfmigrate_run
echo 7.5 >> rfmigrate_run
#change step in baz sweep
echo 1 >> rfmigrate_run
echo 360 0 15 >> rfmigrate_run
#epi sweep global
echo 0 360 >> rfmigrate_run
#change the spacing
echo 1 >> rfmigrate_run
echo 0 180 10 >> rfmigrate_run

rfmigrate_21 < rfmigrate_run >! rfmigrate_LOG

After finishing running the code, we will obtain three pairs of different grid files and rename them according to the information of the site and fmax.

#input for plotting the back azimuthal sweeps after migrating to 50 km without move-out correction
mv outr_baz.grid $1_$2_50km_baz.L.rgrid
mv outt_baz.grid $1_$2_50km_baz.L.tgrid

#input for plotting the back azimuthal sweeps after migrating to 50 km with move-out correction
mv oumr_baz.grid $1_$2_50km_baz.mo.L.rgrid
mv oumt_baz.grid $1_$2_50km_baz.mo.L.tgrid

#input for plotting the epicentral sweeps after migrating to 50 km with move-out correction
mv oumr_epi.grid $1_$2_epi_50km_360.mo.L.rgrid
mv oumt_epi.grid $1_$2_epi_50km_360.mo.L.tgrid

Finally, we can use the script '**plotbaz-short**' to plot the back azimuthal gathers. Notice that we can adjust the starting and ending back azimuths in the script so that we can have a better consistency with the RFs patterns according to the back azimuths. For instance: we choose 270° as the upper boundary of the back azimuths and anything larger than that will be negative.

```
awk '$2 < 270 {print} $2 > 270 && $2 != 360 {print $1, $2-360,$3}' $1 > $1.EW

awk '$2 < 270 {print} $2 > 270 && $2 != 360 {print $1, $2-360,$3}' $2 > $2.EW

#awk '$2 < 90 {print} $2 > 90 && $2 != 360 {print $1, $2-360,$3}' $1 > $1.EW

#awk '$2 < 90 {print} $2 > 90 && $2 != 360 {print $1, $2-360,$3}' $2 > $2.EW
```

After modification, do not forget to change settings of the frame of the plot and the range of the vertical green lines as below:

```
#set BOX = -R-8/25/-280/100
set BOX = -R-8/25/-100/280
gmt psxy $FRAME $BOX -W2p,red -O -K <<END >> RFwig-baz.ps
0 -100
0 280
END
```

plotbaz-short $1_$2_50km_baz.L.rgrid $1_$2_50km_baz.L.tgrid 0.1 $1 0.05
mv RFwig-baz.ps $1_$2_50km_baz.L.ps
plotbaz-short $1_$2_50km_baz.mo.L.rgrid $1_$2_50km_baz.mo.L.tgrid 0.1 $1 0.05
mv RFwig-baz.ps $1_$2_50km_baz.mo.L.ps
plotepi $1_$2_epi_50km_360.mo.L.rgrid $1_$2_epi_50km_360.mo.L.tgrid 0.1 0.05
mv RFwig-epi.ps $1_$2_50km_epi_360.mo.L.ps

This section of command lines has been commented out in '*make-mo-RF.C.txt*', and can be uncommented based on the user's needs.

5) HARMONIC DECOMPOSITION

rfmig_mcboot_21.f – Below is the script with comments that explain the procedure to run harmonic decomposition analysis. The detailed script is called '*make-harmonic-RF.C.txt*'. All inputs in this step are the same as the ***rfmigrate_21*** except minor differences as below to generate the inputs for ***rfmig_mcboot_21***. The code has a capability to use bootstrapping for an empirical assessment of results' stability. If it is engaged, and the data set is large, the code runs *forever*. This functionality is NOT covered by this present manual. Contact Jeffrey Park for guidance.

#first run
#do not bootstrap
echo 0 >! harmonic_run
#model to use for stack
echo $1-model >> harmonic_run
#sampling rate
echo 100 >> harmonic_run
#inverse-variance weight
echo 1 >> harmonic_run
#depth to which migration will be done (make sure the same depth is adopted as in rfmigrate_21)
echo 50 >> harmonic_run

#fmax
echo $2 >> harmonic_run
#time intervals in SAC header
echo 1 >> harmonic_run
#name of pick file
echo in_recpick >> harmonic_run
#duration of data windows
echo 80 >> harmonic_run
#rotate to LQT? -1 = YES and change the speed
echo -1 >> harmonic_run
#a velocity of 7.5 km/s should be kept here or do not change the speed there; this is to make sure that there is no energy on the Q component after rotation; details can be found in Park and Levin (2016b).
echo 7.5 >> harmonic_run

rfmig_mcboot_21 < harmonic_run >! harmonic_LOG

The output files at this step are simple:

mv outr_cexp.grid $1_$2_50km_cexp.L.rgrid
mv outt_cexp.grid $1_$2_50km_cexp.L.tgrid

The file names are confusing since they are actually not the grid files of radial and transverse components, instead they are the five time-varying coefficients of the harmonic terms for both 'modelled part' and 'unmodelled part' (Park & Levin, 2016a, 2016b).

Harmonic stacks can be plotted out using the script '**bexp_ad.plot.txt**'.

6) Customized epicentral sweeps:

This step is not necessary but is helpful when the analyst would like to check the converted phases in detail. In this step, customized back azimuthal ranges will be included and used to generate the normal RFs and apply migration.

Inside the script, it assumes you create two txt files *sitelist* and *fmaxlist* beforehand. Inside *sitelist*, you are expected to put the name of the folder (typically the site name) where you keep all your preprocessed records for each row (A sample for each file can be found in the package). Inside *fmaxlist*, each frequency is listed in a row. After that, you can go inside the script and replace 'cat sitelist' by 'cat xx/xx/xx/sitelist' and 'cat fmaxlist' by 'cat xx/xx/xx/fmaxlist', 'xx/xx/xx' here stands for the absolute paths where those files are stored on your work station. Always make sure the path is set correctly and you have all the necessary files. The spacing of back azimuths can be adjusted inside the script by modifying 'set dbaz=x', 'x' represents the spacing you would like to set. Then you can run the script '**procedure_C_epi**' as below:

procedure_C_epi $baz1 $baz2

For instance, if the user would only like to examine the EPI gathers from 100° to 210°, the user can simply run '***procedure_C_epi 100 210***' here.

Make sure $baz1 < $baz2 and if both of them are negative (i.e. measured west of N), you should consider adding 360° to the original back azimuth.

All the input files are the same as previous steps except for the beginning ($3) and ending ($4) back azimuths (marked as **bold**).

In ***rfmigrate_21***:

#first run
#name of the model
echo $1-model >! rfmigrate_run
#depth to which migration will be done
echo 50 >> rfmigrate_run
#fmax
echo $2 >> rfmigrate_run
#time intervals in SAC header
echo 1 >> rfmigrate_run
#name of pick file
echo in_recpick >> rfmigrate_run
#duration of data windows
echo 80 >> rfmigrate_run
#min number of traces
echo 2 >> rfmigrate_run
#rotate to LQT? -1 = YES, and change the value
echo -1 >> rfmigrate_run
echo 7.5 >> rfmigrate_run
#change step in baz sweep
echo 1 >> rfmigrate_run
echo 360 0 15 >> rfmigrate_run
#epi sweep custom
echo $3 $4 >> rfmigrate_run
#change the spacing
echo 1 >> rfmigrate_run
echo 0 180 $5 >> rfmigrate_run

REFERENCES:

Park, & Levin. (2016a). Anisotropic shear zones revealed by backazimuthal harmonics of teleseismic receiver functions. *Geophysical Journal International, 207*(2), 1216-1243. http://dx.doi.org/10.1093/gji/ggw323

Park, & Levin. (2016b). Statistics and frequency-domain moveout for multiple-taper receiver functions. *Geophysical Journal International, 207*(1), 512-527. https://doi.org/10.1093/gji/ggw291

Park, & Levin, V. (2000). Receiver Functions from Multiple-Taper Spectral Correlation Estimates. *Bulletin of the Seismological Society of America, 90*(6), 1507-1520. http://dx.doi.org/10.1785/0119990122