

Manual of ASWMS

Ge Jin¹ and James Gaherty¹

¹Lamont Doherty Earth Observatory, Columbia University

June 12, 2014

Contents

1	Introduction	2
2	Data Preparation	2
2.1	Data Structure	3
2.2	Using SAC files	3
2.3	Using IRIS DMC Service	4
3	Parameter Adjustment	5
4	Running the Scripts	6
5	Reviewing the Result	7
5.1	Output Format	7
5.2	Result Visualization	8
5.3	Adjusting Parameters and Re-run the result	9
6	The Limitations of ASWMS	9
6.1	Array Geometry	9
6.2	Incorrect Instrument Response	10
7	Technical Details	10
7.1	run_autowinpick.m	10

7.2	gsdfmain.m	12
7.3	eikonal.eq.m	15
7.4	helmholtz.eq.m	18
7.5	stack_phv.m and stack_helm.m	18
8	Reference	19

1 Introduction

The ASWMS package is developed to measure the phase and amplitude of surface waves from the raw seismic waveforms and generate phase velocity maps via Eikonal and Helmholtz equations. The detailed theoretical development of this package can be found in Jin and Gaherty, 2014.

Section 2-5 detail how to configure the ASWMS package on your computer and adjust the parameters to fit your specific project. This setup should produce a good initial result for most teleseismic studies (e.g., arrays ranging from 100 km to continental scale, and station spacing less than 100 km). Section 7 describes additional adjustments that can be made to further customize the package so as to best suite an individual project's needs.

In this manual, the names of files and folders are written in **blue** and the names of variables are written in **orange**. All the adjustable parameters in this package are stored in the [setup_parameters.m](#), and the input data should be put in the folder [eventmat](#).

The ASWMS package is written in Matlab, and requires the following toolboxes:

Curve Fitting Toolbox (for `lsqcurvefit.m`)
Signal Processing Toolbox Statistics Toolbox
Mapping Toolbox
Statistics Toolbox

The Statistics Toolbox is not needed if users download the NaN suite and include all of its *.m files in the /matgsdf directory.

<http://www.mathworks.com/matlabcentral/fileexchange/6837-nan-suite/content/nansuite>.

2 Data Preparation

Users can either use existing SAC files and convert them to the required .mat format (Section 2.2) or use the [data_download.m](#) (included in the ASWMS package) which will download and format data using [irisFetch.m](#) (Section 2.3).

2.1 Data Structure

The input data are event-based matlab structures saved in mat files, which should be put in the folder `eventmat`. The filename for each event should follow the rule of `YYYYMMDDhhmm.comp.mat`, for example: `200901081921.LHZ.mat`, and the structure name has to be `event`. Most of the fields use the same name as sac headers, with the exception being explained below.

Below is an example of the event structure:

```
event =  
    evla: 10.2152  
    evlo: -84.2159  
    otime: 6.3399e+10  
    id: '200901081921'  
    otimestr: '08-Jan-2009 19:21:34'  
    stadata: [1x654 struct]
```

In which `otime` is the number of seconds to a certain time, which the actual time is not critical. `evla` and `evlo` describe the earthquake location.

Inside of the event structure is The field `stadata`. This is where all information concerning the waveform and megadata are stored. Below is an example of the `stadata` structure.

```
event.stadata(1) =  
    stla: 32.8920  
    stlo: -116.4223  
    stel: 1.8750  
    dist: 4.1447e+03  
    otime: 6.3399e+10  
    delta: 1  
    data: [7200x1 double]  
    cmp: 'LHZ'  
    stnm: 'MONP2'
```

The `stadata.otime` has to have the same origin time as the `event.otime`, and has to have the unit in seconds. The `stadata.dist` has the unit of km, and `delta` is the sample interval in seconds.

Rather than writing your own code to transfer your data into the matlab structure, we provide two alternative methods detailed in the following section.

2.2 Using SAC files

The script `sac2eventmat.m` can be used to transform SAC files into the appropriate eventmat files.

The input SAC files must be placed in a folder named `sacdata`. the sacdata must be organized such that every individual event has its own folder named as `YYYYMMDDhhmm`. The file

name should include the components name defined in [setup_parameters.m](#) and end with “.sac”. Here shows an example:

```
>>ls  sacdata
200801011855
200801051101
200801051144
200801070312
...
...
>>ls  sacdata/200801011855
200801011855.TA.G09A.LHZ.sac
200801011855.TA.G10A.LHZ.sac
200801011855.TA.G11A.LHZ.sac
200801011855.TA.G14A.LHZ.sac
...
...
```

The origin time of each sac file has to be the origin time of the earthquake, while the waveform starting time can be different. Each sac file should have the following information in its header:

Event information: [NZYEAR NZHOUR NZMIN NZSEC NZMSEC EVLA EVLO EVDP](#)

station information: [STLA STLO STEL KSTNM](#)

data information: [B DELTA KCMPNM](#)

An ASCII file named [eventlist](#) listing the names of all event folders must be put in the [sacdata](#) folder as well. An example of this file is:

```
>>cat  sacdata/eventlist
200801011855
200801051101
200801051144
200801070312
...
...
```

You can easily generate this file by using the shell pipe function.

2.3 Using IRIS DMC Service

The ASWMS package can directly obtain event information and download waveform data from the IRIS DMC via the matlab script [data_download.m](#). This script utilizes the DMC’s Matlab interface [irisFetch.m](#). For more details of this service please visit IRIS website:

<http://www.iris.edu/dms/nodes/dmc/software/downloads/irisFetch.m>, or Google: [irisFetch](#).

The following parameters in the [setup_parameters.m](#) should be adjusted if you want to use this service.

```
% parameters for data downloading (if using IRIS DMC)
parameters.start_time = '2009-01-07 00:00:00';
parameters.end_time = '2009-06-08 00:00:00';
parameters.is_use_timestamp = 1;
parameters.network = '_US-ALL';
parameters.minMw = 6;
parameters.maxdepth = 50;
parameters.datalength = 7200; % in second
parameters.resample_delta = 1; % in second
```

If the [parameters.end_time](#) is empty (''), then the end time of the data fetching will be 4 days before the current date. If the [is_use_timestamp](#) is 1, then after each successful run of the [data_download.m](#), the end time of this run will be saved into a mat file named [tempstamp.mat](#). Thus, the next time [data_download.m](#) is triggered, it will begin where the last run left off. This option is useful for setting up a self-updating system.

The station network can be set to '*' to fetch all the channels available in the region.

If the LHZ component is not available, the user can download the BHZ component and resample the data where the sample interval is defined by [parameters.resample_delta](#).

The downloaded data are initially stored in the folder [datacache](#) before they are further processed. If the target files already exist in this folder, the program will not redownload them so as to avoid repeat downloads and thus save running time. Most of the scripts within the ASWMS program contain this existing-data-check feature. Once the data is downloaded, the station responses are removed, and the data are reorganized and transformed into the proper event matlab structure in the folder [eventmat](#).

3 Parameter Adjustment

All the adjustable parameters are defined within the different sections of the [setup_parameters.m](#). Most of these parameters are so called “under the hood” and thus do not need to be altered for most of the teleseismic projects.

Here is a list of the key parameters that should be adjusted for each project. The initial values shown below were chosen for the USArray experiment:

```
parameters.proj_name = 'USArrayExample';
parameters.component = 'LHZ'; % determined by filenames
parameters.lalim=[25 50];
parameters.lolim=[-125 -65];
```

```
parameters.gridsize=0.3; % in degrees
```

where the `proj_name` can be any string. The `component` should be included in the sac filenames (if you are using sac as input); it is also shown in the names of the output files. It is recommended to be “LHZ” for teleseismic Rayleigh-wave projects. The `lalim`, `lolim` and `gridsize` define the bounds of the tomography grid and output files and figures. The final output grid will be defined as:

```
xnode = lalim(1):gridsize:lalim(2);  
ynode = lolim(1):gridsize:lolim(2);  
[xi yi] = ndgrid(xnode,ynode);
```

Another group of parameters that frequently need to be adjusted are the `periods` and the smoothing weight in each period (`smweight_array`). The `periods` are defined as:

```
parameters.periods = [20 25 32 40 50 60 80 100]; % in seconds
```

which are the central frequencies of the narrow-band filters. The width of the narrow-band filters around 10% of the central frequency and are defined by `min_width` and `max_width`.

And the smoothing weight is defined for each period as:

```
parameters.smweight_array = 3*[0.4 0.3 0.2 0.2 0.2 0.5 1 2];
```

It should have the same length as `parameters.periods`, with smaller value for the frequency bands with higher SNR or short wavelengths. More details on the smoothing parameters can be found in the Section 7. The example presented here shows a good ratio for the periods listed above, and you may only need to change the constant (3 in this case) for your project.

For most projects, adjusting these parameters should be able to provide you a good initial result. We will discuss other adjustable parameters in the Section 7.

4 Running the Scripts

Once the data has been prepared and the parameters have been adjusted, you can run the package and obtain an initial result. The proper sequence of the commands for a complete run of the ASWMS package is listed in `main_driver.m`. It is ready to run once you have chosen a data acquisition method and modified the script appropriately.

The sequence of these commands is listed below as well as a brief description of their function and necessary input/output. These scripts are described in details in the Section 7.

1. [sac2eventmat.m](#) OR [data_download.m](#)

Data acquisition/organization scripts - choose one or build your own.

Input: sac files/NaN Output: eventmat/*.mat

2. [cleanup_events.m](#)

Finds and deletes events within the `eventmat` folder that are too close in time such that

their surface waves may interfere with each other at the array location.

Input: eventmat/*.mat Output: NaN

3. [run_autowinpick.m](#)

Defines the window function to isolate the energy of surface waves within the records. A new field, [winpara](#), is created within the [event](#) structure. An ASCII file in the folder [winpara](#) is also generated. If you want to recalculate the window for a given event, you need to delete the relevant file in the [winpara](#) directory.

Input: eventmat/*.mat Output: winpara/* eventmat/*.mat

4. [gsdfmain.m](#)

Measures the phase delay between nearby stations via cross-correlation. This is the core program of the entire package. For each event, the script generates a structure named [eventcs](#) and saves it to the folder [CSmeasure](#).

Input: eventmat/*.mat Output: CSmeasure/*.mat

5. [eikonal_eq.m](#)

Performs the tomography inversion via Eikonal equation, based on the cross-correlation measurements. The script generates a structure named [eventphv](#), which contains the apparent phase velocity maps for each event, and save it to the folder [eikonal](#).

Input: CSmeasure/*.mat Output: eikonal/*.mat

6. [stack_phv.m](#)

Stacks all the events for which there are phase velocity measurements in the folder [eikonal](#) and outputs a structure named [avgphv](#) containing the stacked apparent phase velocity map. This is saved to the file [eikonal_stack_LHZ.mat](#). This is the final result if no amplitude correction is not applied.

Input: eikonal/*.mat Output: eikonal_stack_LHZ.mat

7. [helmholtz_eq.m](#)

Applies the amplitude correction on the apparent phase-velocity results via Helmholtz equation. The script generates a structure named [helmholtz](#), which is stored in the folder [helmholtz](#).

Input: eikonal/*.mat CSmeasure/*.mat eikonal_stack_LHZ.mat Output: helmholtz/*.mat

8. [stack_helm.m](#)

Stacks the corrected phase velocity maps from each event and generates the final result, as a structure named [avgphv](#). This is saved to the file [helmholtz_stack_LHZ.mat](#). Input: helmholtz/*.mat Output: helmholtz_stack_LHZ.mat

5 Reviewing the Result

5.1 Output Format

The final result is stored in [helmholtz_stack_LHZ.mat](#). If [helmholtz_eq.m](#) was not run/commented out in [main_driver.m](#), then the final results are in [eikonal_stack_LHZ.mat](#). Note, the

structures **avgphv** within these two files are identical, only the structure within **eikonal_stack_LHZ.mat** does not contain the amplitude correction related fields. Here we only present the structure in the **helmholtz_stack_LHZ.mat**. Each of the binary *.mat output files used for figures are also given as ASCII *.xyz files.

Below is an example of **avgphv**:

avgphv(4) =

```

    sumV: [84x201 double]
    sumV_cor: [84x201 double]
    sumweight: [84x201 double]
    GV_std: [84x201 double]
    GV_cor_std: [84x201 double]
    eventnum: [84x201 double]
    xi: [84x201 double]
    yi: [84x201 double]
    xnode: [1x84 double]
    ynode: [1x201 double]
    period: 40
    GV_cor: [84x201 double]
    GV: [84x201 double]

```

where the important fields are:

xi Latitude of the grid.

yi Longitude of the grid.

GV Phase velocity before amplitude correction.

GV_cor Phase velocity after amplitude correction.

eventnum Number of events each grid stacked.

5.2 Result Visualization

There are map plotting commands contained in the **stack_phv.m** and **stack_helm.m**, which can be activated by changing the variable **isfigure** to 1 at the beginning of these two scripts. You can also easily create your own plotting scripts based on these commands to plot your own figures. Below is a simple example of one such plotting:

```

load helmholtz_stack_LHZ.mat
ip = 4; % plot the 40s result
figure(88)
clf
ax = worldmap(lalim, lolim)
surfacem(avgphv(ip).xi, avgphv(ip).yi, avgphv(ip).GV_cor);

```



```
% set the color scale
cmap = colormap('jet');
cmap = flipud(cmap);
colormap(cmap);
% set the color range
r = 0.1; % 20 % peak to peak
meanphv = nanmean(avgphv(ip).GV_cor(:));
caxis([meanphv*(1-r) meanphv*(1+r)]);
colorbar
```

[.mat](#) output files are also available as ASCII [.xyz](#) files for users choosing to plot results themselves.

Additionally, a package is available to generate a useful summary of your results and database in HTML format. This package, named GSDF-Report, can be downloaded here:

https://github.com/jinwar/gsd_report

After setting up the path of the ASWMS package in the [setup_parameters.m](#), users may run the [main_driver.m](#) script to generate the HTML files placed in the folder [htmls](#).

5.3 Adjusting Parameters and Re-run the result

Depending on your result, you may wish to adjust some parameters like cross-correlation distance ([parameters.maxstadi](#)), smoothing weight ([parameters.smweight_array](#)) and others. However, the package is written in a way that it will pick up where it stopped during the last run (i.e. if you stopped the program in the middle of `gsdfmain.m`, the program will begin running `gsdfmain.m` on the next event without rerunning previously analyzed events). In order to generate new results, you may need to delete all the files in the [winpara](#) (for `run_autowinpick.m`), [CSmeasure](#) (for `gsdfmain.m`), [eikonal](#) (for `eikonal_eq.m`), [helmholtz](#) (for `helmholtz_eq.m`) folders, depending on the step you want to redo. The `bash` script [cleandata.csh](#) is available to help you quickly reset your project folder.

Because the `data_download.m` takes a long time to recover, it is suggested to comment it out of [main_driver.m](#) once the data download is finished.

6 The Limitations of ASWMS

6.1 Array Geometry

The major limitation of this package concerns array geometry. In general, this package is best applied on a 2D near-evenly distributed array with average station spacing less than the wavelength the highest frequency surface wave of interest.

Equally important is the need for overlapping/crossing cross-correlation paths. Users can adjust the allowable cross-correlation distance to increase ray crossing. However, if the cross-

correlation distance is more than 3-4 wavelengths, issues with cycle-skipping may occur at high frequencies.

See Figure 1 for examples of these different situations.

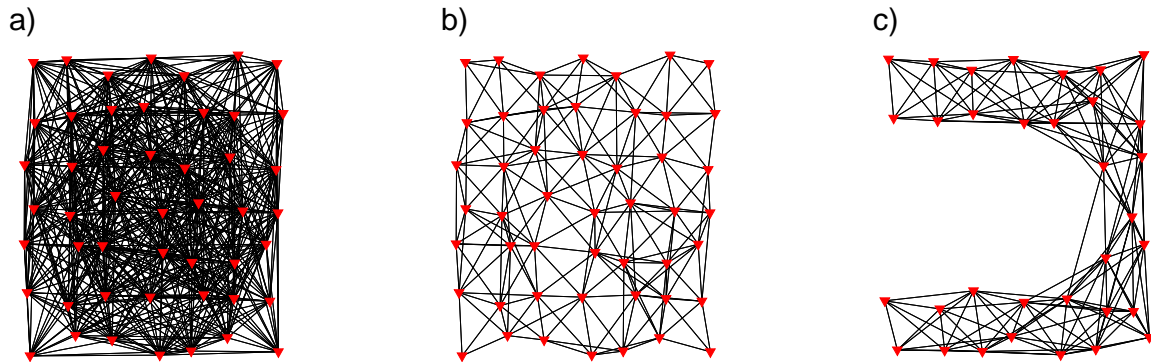


Figure 1: *Different array geometries and cross-correlation distance a) The ideal array geometry and ray density for ASWMS. b) The cross-correlation measurements are too sparse. `parameters.maxstadist` should be increased. c) Array geometry is not ideal for ASWMS.*

6.2 Incorrect Instrument Response

Many arrays contain multiple types of sensors, and their respective instrument responses stored in the IRIS DMC may not be correct. Utilization of the incorrect instrument response may generate problems in the amplitude correction step and bias the final helmholtz result. In this case, the amplitude correction is not suggested and the Eikonal result [eikonal_stack LHZ.mat](#) is suggested to be utilized as the final result.

7 Technical Details

The following section discusses the technical details in each component of the program, and provides users additional freedom to customize the package to fit their own projects.

7.1 `run_autowinpick.m`

The script [run_autowinpick.m](#) is used to define the time window for the isolation of the surface-wave energy. Figure 2 shows you an example of this function.

The scheme of this window selection is based on tracing the frequency-dependent group delay for a given surface wave within a defined group-velocity range. The location of the

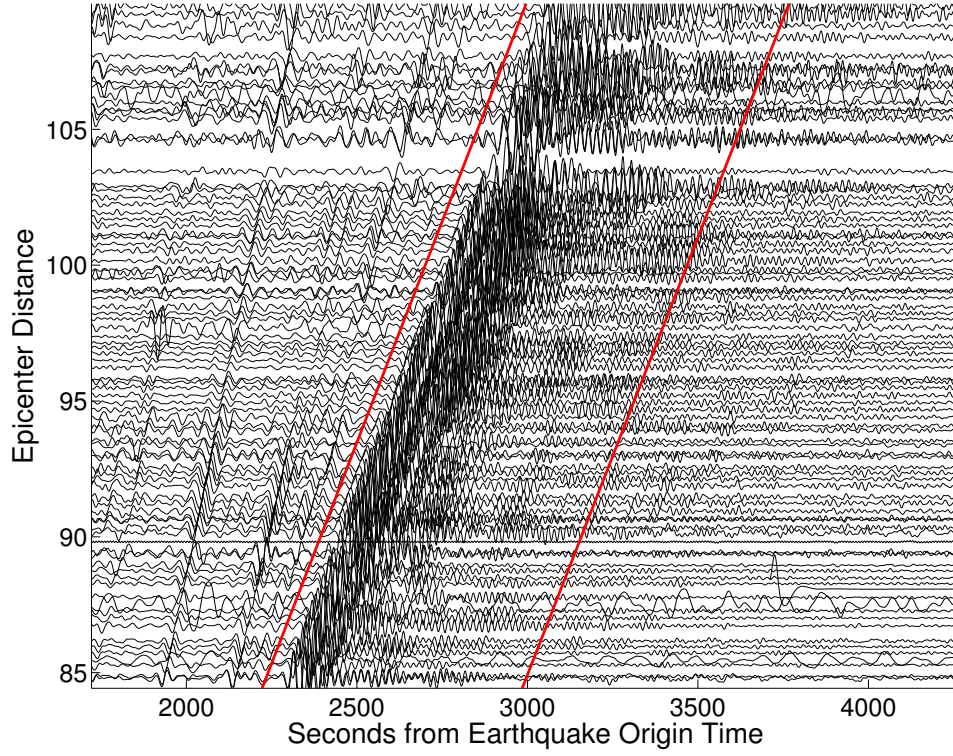


Figure 2: Window function W_S (red line) to isolate the Rayleigh wave energy

window function is linearly dependent on epicentral distance by:

$$T_1 = \frac{L}{v_1} + t_1$$

$$T_2 = \frac{L}{v_2} + t_2$$

where T_1 and T_2 are the beginning and ending time of the window, L is the epicentral distance, and v_1 , v_2 , t_1 , t_2 are the parameters estimated by the linear regression.

The following parameters control the selection of a window function:

```
parameters.min_groupv = 2;
parameters.max_groupv = 5;
parameters.cent_freq = 0.025;
parameters.largest_epidist_range = 3000;
parameters.cycle_before = 2;
parameters.cycle_after = 5;
parameters.min_dist_tol = deg2km(20);
parameters.max_dist_tol = deg2km(160);
```

min_groupv : the minimum group velocity to search for.

max_groupv : the maximum group velocity to search for.

cent_freq : the frequency band that has the best signal to noise ratio. It should in the range of frequencies defined by **parameters.periods**.

largest_epidist_range : The maximum epicentral distance range that can be processed. If it is exceeded, the program will select the epicentral distance range with the most stations and mark everything outside as bad stations. In most regional studies, you don't need to worry about this parameter. For some large arrays (e.g., USArray), the stations span a wide range of epicenter distance so the linear relation between group delay and epicentral distances break down.

cycle_before : Number of cycles that the window function should include before the group delay (the window function should include 1-2 cycles before the group delay so as to capture the peak of the envelop function).

cycle_after : the number of cycles that the window function should include after the group delay.

min_dist_tol : minimum distance between the earthquake and the center of the array. Should be large enough to allow the separation between the body wave and surface wave.

max_dist_tol : maximum distance between the earthquake and the center of the array. Should be small enough to avoid the interference between R1 and R2.

Tips: in the winpara, there is a ASCII file generated for each event. It contains 4 numbers, which are the v_1 , t_1 , v_2 , t_2 as in the equation. Something is wrong if most of your events have v_1 and v_2 being end-member values defined by the **min_groupv** and **max_groupv**.

7.2 gsdfmain.m

gsdfmain.m drives the cross-correlation measurements between the nearby stations. This is the core program of the entire package.

Figure 3 shows the waveforms from two nearby stations (90 km apart). After isolating the surface-wave energy of station 2 using window function W_S , the cross-correlation $C(t)$ is calculated between S_1 and $W_S S_2$. Then the $C(t)$ is narrow-band filtered and fit to get the phase delay between these two stations (Figure 4).

The parameters that are adjustable in this section of the package are:

```
parameters.minstadist = 5;
parameters.maxstadist = 200;
parameters.is_rm_resp = 0;
parameters.refv = 4;
parameters.refphv = ones(size(parameters.periods))*4;
parameters.min_width = 0.06;
parameters.max_width = 0.10;
parameters.wintaperlength = 30;
parameters.prefilter = [10,200];
```

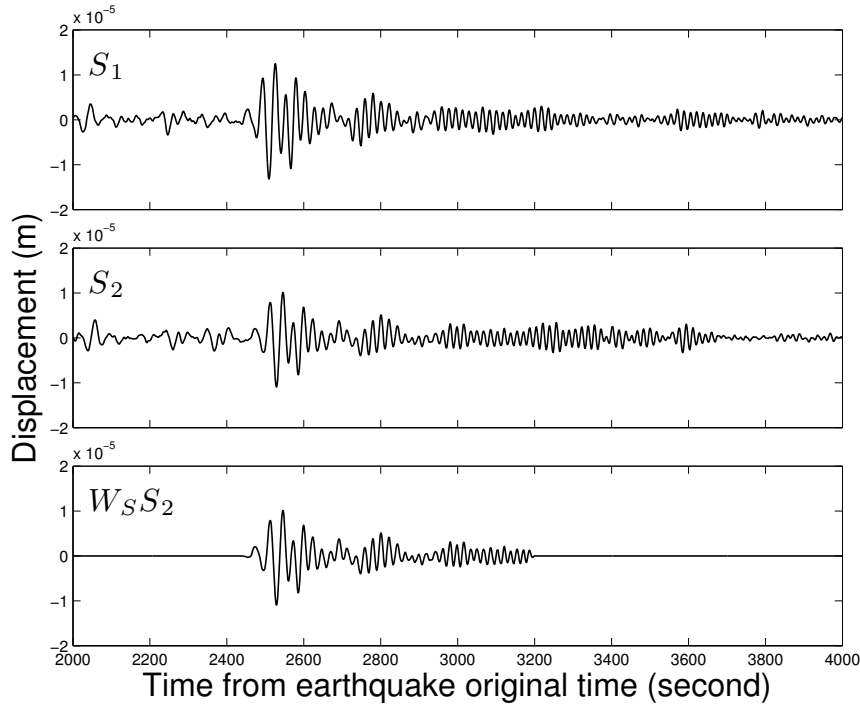


Figure 3: Waveforms from two nearby stations

```
parameters.xcor_win_halflength = 100;
parameters.Nfit = 2;
parameters.Ncircle = 5;
parameters.cohere_tol = 0.5;
parameters.tp_tol = 10;
```

minstadist : Minimum distance between two stations for the cross-correlation. Should be a small number just to avoid the station cross-correlate with itself.

maxstadist : The maximum station distance for cross-correlation. This distance should be at least twice of the average station spacing to get a good tomography result. However, should be smaller than 3-4 times the wavelength of the highest frequency surface wave.

is_rm_resp : This should be turned off (set to 0) in almost all cases.

refv : A rough estimation of the average group velocity (km/s) at the center frequency band (about 40s). It does not need to be accurate. 4 is a good number for all teleseismic projects.

refphv : A reference phase velocity (km/s) to correct for cycle-skipping. It only matters if the mean station spacing is close to the wavelength of the surface-wave wavelength. Thus, if the high frequency bands experience cycle-skipping issues, you may need to provide a more accurate estimation.

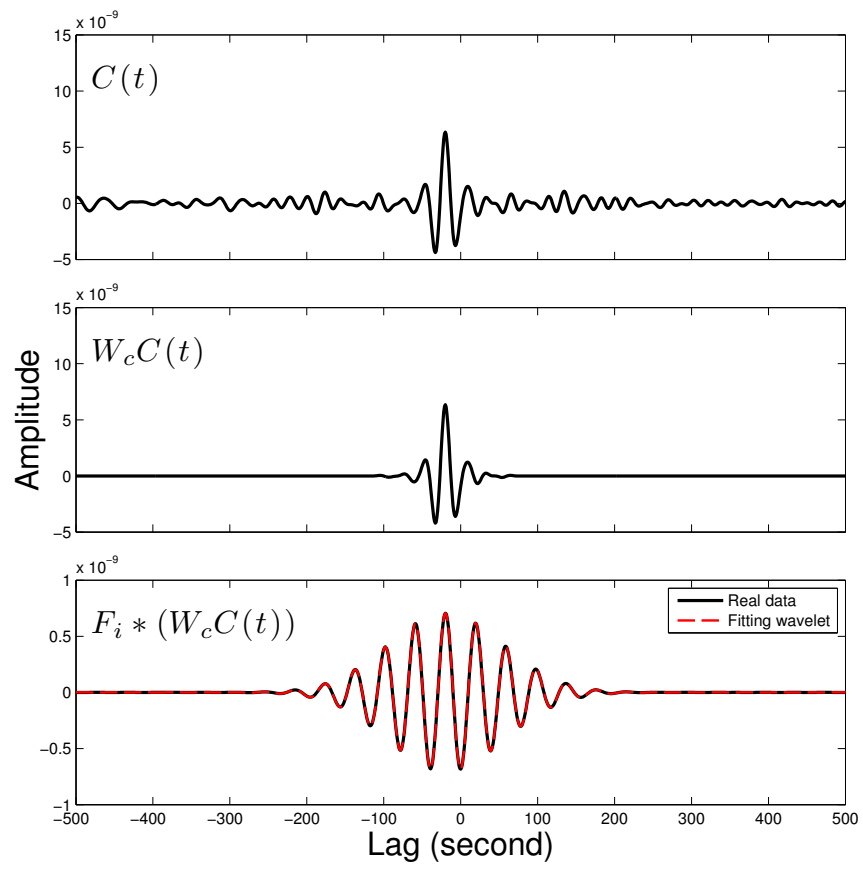


Figure 4: *Demonstration of the cross-correlation procedures*

min_width,max_width : Define the band-width of the narrow-band Gaussian filters applied on the cross-correlation waveforms. The shape of Gaussian functions for the default setup (0.06 and 0.10) is shown in Figure 5.

wintaperlength : The taper length of the window function applied on the original waveform. Because usually this window function is a few hundred seconds long, no strong (long) taper is needed. (Figure 3)

prefilter : The pre-filter applied on the original waveforms. The band-pass should be wider than your interested frequency range.

xcor_win_halflength : The length of the cross-correlation window function. Should be longer than a few periods of your lowest frequency band. (Figure 4)

Nfit : Number of from the center used to fit the five-parameter wavelet. Usually 2 or 3 is appropriate for all projects.

Ncircle : Number of cycles searched for cycle-skipping. 5 is appropriate for most of the teleseismic projects. For projects focused on very high frequencies, you may need a larger number.

cohere_tol : QC Parameter. Defines the minimum coherence that is required between station pair waveforms to pass the data QC (quality control). This represents one of the key parameters users should pay attention to. We found that for most teleseismic projects, 0.5-0.6 is appropriate.

tp_tol : QC parameter. For each event, an average phase velocity is fit for the entire array by assuming a straight ray path. The misfit of each phase delay measurement is then verified using this average phase velocity estimate. All measurements with misfits greater than the **tp_tol** are discarded as bad measurements. The number shown here is appropriate for USArray. Users may change it to a smaller value for smaller arrays. (Figure 6)

7.3 eikonal_eq.m

This script performs the slowness vector inversion based on Eikonal equation:

$$\delta\tau_p = \int_{r_i} \vec{S}(\vec{r}) \cdot d\vec{r}$$

where $\delta\tau_p$ is the phase delay measurements from [gsdfmain.m](#). The parameters utilized in this sections are:

```
parameters.smweight_array = 3*[0.4 0.3 0.2 0.2 0.2 0.5 1 2];
parameters.raydensetol=deg2km(parameters.gridsize)*2;
parameters.Tdumpweight = 0;
parameters.Rdumpweight = 0;
parameters.fiterrtol = 3;
```

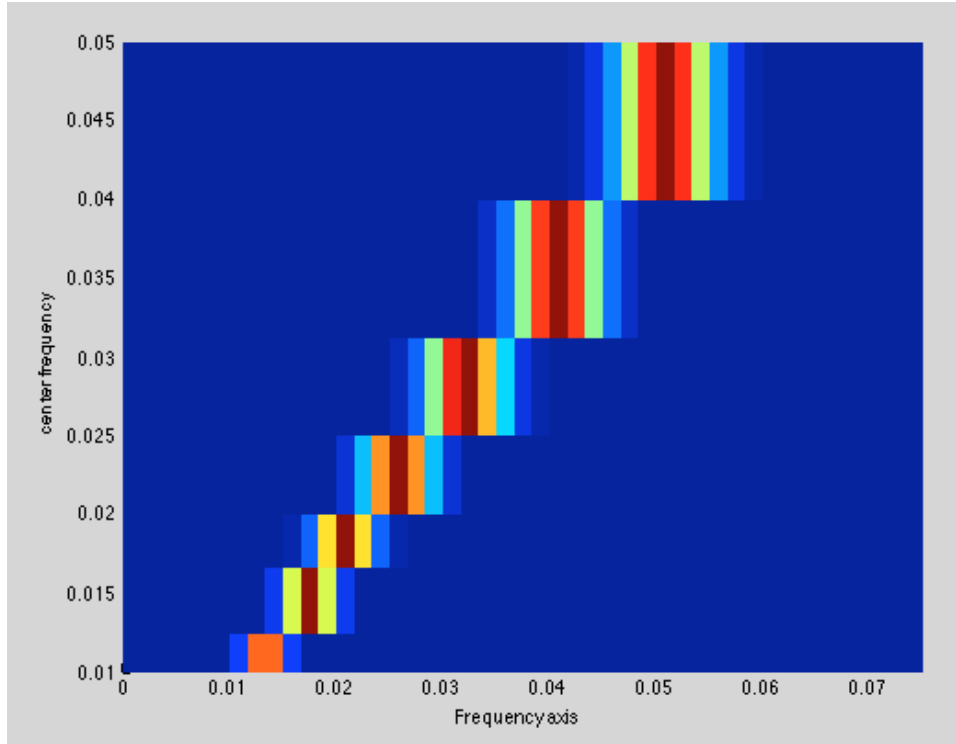


Figure 5: *The gains of Gaussian filters*

```
parameters.isRsmooth = 1;
parameters.dterrtol = 2;
parameters.inverse_err_tol = 2;
```

smweight_array : This array of parameters controls the smoothing weight of the phase velocity inversion at each period. The ratio defined here is appropriate for teleseismic projects. In most cases users only needs to change the constant before the array. This is one of the parameters that should be adjusted from project to project.

raydensetol : Controls the minimum ray density in each grid. If the ray density within a given grid is less than this value, the result of that grid will be set to NaN.

Tdumpweight : Used to force the waves to propagate along the great circle path. By setting this to a small value, the user forces the projection of the slowness vector on the tangential direction to be zero. It may be useful in some extreme conditions, otherwise it should be set to 0 to allow for ray bending.

Rdumpweight : Used to force the slowness on the radial component to be close to the value of **refphv**. Again, it may be useful in some extreme conditions, otherwise should be set to 0 in most cases.

fiterrtol : Error allowed in the wavelet fitting. It is not suggested to alter this parameter.

isRsmooth : Used to choose the application of the smoothing kernel on the NS-WE direction or on the RT direction. It is suggested to be 1 in most cases.

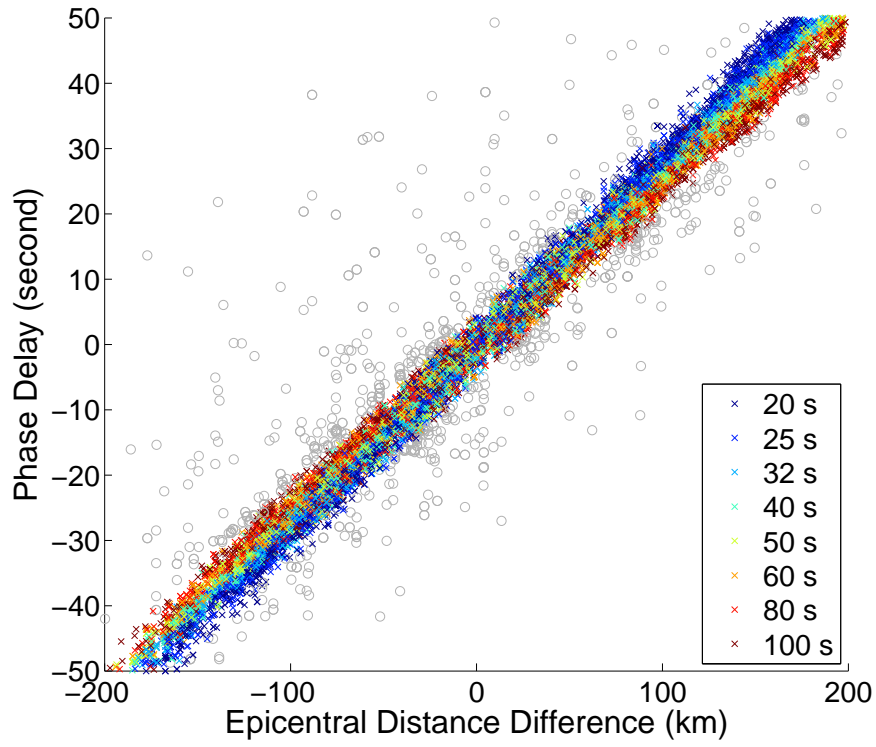


Figure 6: Relative phase delays against the epicentral distance differences for all the station pairs within 200 km for the same records shown in Fig.2. Crosses with different color represent the measurements at different frequencies, and the grey circles represent the poor quality measurements that are discarded based on the parameter *tp_tol*. A faster move-out at the lower frequencies demonstrates the average phase velocity dispersion across the array.

dterrtol : QC parameter. The slowness inversion is performed twice. After the first run, the misfits of the inversion are calculated for all the measurements, and the measurements with misfits larger than this value in seconds are discarded.

inverse_err_tol : QC parameter. Same as **dterrtol**, but with the unit of standard deviations.

7.4 helmholtz_eq.m

This script applies amplitude corrections on the apparent phase velocity from [eikonal_eq.m](#). It also reads in the [stack_phv.m](#) output so you likely need to run [stack_phv.m](#) first, as described in the [main_driver.m](#).

```
parameters.min_amp_tol = 0.1;
parameters.amp_var_tol = 2;
parameters.alpha_range = [1 1];
parameters.alpha_search_grid = 0.1;
```

min_amp_tol : QC parameter. The program calculates the median amplitude of all the stations and discards the stations with amplitude smaller than the median amplitude times this value. (Only the amplitude measurement is discarded. The phase measurements from these stations are still valid)

amp_var_tol : QC parameter. Within the range defined by **maxstadi**, the median amplitude A_m among the stations is calculated. The stations with amplitude larger than A_m times or smaller than A_m divides this value are discarded.

alpha_range, alpha_search_grid : just leave as it is.

7.5 stack_phv.m and stack_helm.m

[stack_phv.m](#) and [stack_helm.m](#) are two scripts to stack the results from Eikonal tomography ([eikonal/*](#)) and Helmholtz tomography ([helmholtz/*](#)). In the beginning of both scripts, there is a variable **isfigure** which can be changed to 1 if you want to plot the results.

The adjustable parameters here are:

```
parameters.min_csgoodratio=0.3;
parameters.min_phv_tol = 3;
parameters.max_phv_tol = 5;
parameters.is_raydense_weight = 1;
parameters.min_event_num = 10;
parameters.err_std_tol = 4;
parameters.issmoothmap = 1;
parameters.smooth_wavelength = 0.25;
```

min_csgoodratio : Discard the events with fewer good measurements than the number of bad measurements times this value.

min_phv_tol : Discard the phase velocity from an individual event within a given grid cell if it is smaller than this value.

max_phv_tol : Discard the phase velocity from an individual event within a given grid cell if it is larger than this value.

is_raydense_weight : Chooses whether or not to weight the stacking by raydensity. Users may want to try both options. For regions with large azimuthal anisotropy, it is suggested to be turned this weighting off.

min_event_num : for the final result, individual grid cells with fewer event results than this value are set to NaN.

err_std_tol : QC parameter. The stacking is performed twice. After the first stacking, the difference between the phase velocity from single event and the stacked phase velocity is calculated, and the data points with the difference larger than this number times the standard deviations are discarded before the second stack.

issmoothmap : Choose whether or not to perform one more step of smoothing for the final result. Using this additional smoothing step is helpful in most cases.

smooth_wavelength : Defines the range of the final smoothing during the last smoothing step. This value is multiplied by the average wavelength of the surface wave at each frequency of interest.

8 Reference

Jin, G., and J. B. Gaherty (2014), Surface Wave Measurement Based on Cross-correlation , Geophys. J. Int, submitted.