AELUMA Automatic Event Location Using a Mesh of Arrays (matlab code)

Note: this code will only work out of the box if you use data available through the DMC. Otherwise, you'll have to build your own input/output codes.

This user manual is extracted from a larger report. Once it becomes publicly available, I'll add the link.

APPENDIX E – USER MANUAL - INPUT

Appendices A through D explain how the code works. This appendix and the next serve as the user manual for the code; this appendix explains how to set up the parameter files for signal detection and source location, the next appendix describes the output files that are written.

(1) Signal detection using triads – set up **Triad_param1.m**

The code **Triad_detections(Year, JDay)**, located in /bin is run to detect signals at triads.
Year – year of interest
JDay – julian day of interest

The workings of the code are described in Appendices A and B. Before running it, information about the signal that is of interest must be supplied by the file **Triad_param1.m**, also located in /bin. The parameters to be set in Triad_param1.m are explained here, along with suggested values. The parameters deal with the type of station to be used, the network configuration, and properties of the signal under investigation.

ifDMC – if data available through the DMC data are of interest, this parameter is set to 1. Suggested value = 1 for a first pass at running the code – the user must have a good internet connection since the data are accessed online. **If this parameter is set to another value, for instance to read in data on disk, the user must supply their own code to read the data and put it in a format usable for the Triad_detections code.**
triadflag – this allows the code to be run using different parameters for a particular year and day of interest without over-writing the previous file for that date. This might allow the user, for instance, to run the code to search for seismic body waves for one run, and infrasound data for another run. Suggested value = 1 for the first time it is used for a given day and year.

The next two parameters are specific to the case where DMC data are read in. They do not need to be set if ifDMC=0.

net – this is the network code needed to read DMC data. Check the DMC pages for information on the networks available. Setting the value net ='*' searches through all stations available on the day of interest.
loc – a location code needed to read in DMC data. Suggest setting loc ='' (blank)

The next few parameters deal with data type/sampling rate for the signal of interest

chan – channel code for seismic (or infrasound) data. This may be listed as a string of codes if more than one channel type is wanted, i.e chan ='SHZ, BHZ, HHZ' could be used for high frequency seismic data
fsmin - minimum allowable data sample rate. Suggested value fsmin = 20 for high frequency seismic data
fsmax - maximum allowable data sample rate.   fsmax must be > fsmin

The next parameters deal with the geographic region of interest

ifregion – set to 0 if stations are not limited by geographic area (i.e. read data from anywhere in the world) or 1 to specify a particular geographic region= 1; Suggest setting ifregion=1
lon1, **lon2** – west/east longitudes of region of interest. lon1< lon2. For instance, to specific longitudes within contiguous United States, lat1 = -130; lon2 = -60;
lat1, **lat2** – south/north latitudes of region of interest. lat1< lat2. For instance, to specific latitudes within contiguous United States, lat1 = 25; lat2 = 50;
Note if ifregion=0, the lat and lon values are treated as dummy variables.

The next few parameters are dependent on the network geometry

rtmx – the maximum triad arm length in kilometers. This depends on the network density. The sparser the network, the higher this number will have to be. For the USArray, suggested value = about 150.
rmin - the minimum triad arm length in km. Suggested value approximately rtmax/4.
minang – minimum interior angle to define a triad (equilateral triangles are ideal). Suggest minang=20.
maxang – maximum interior angle of triad. Suggest maxang= 180-2*minang;

The next set of parameters depend on the signal of interest.

fmin – minimum signal frequency of interest
fmax – maximum frequency of interest. fmax must be > fmin. Also, fmax must be <fsmin/2 to satisfy the Nyquist criterion.
numf –number of frequency sampling bands. numf must be an integer, the simplest choice is 1. The code will compute the frequency range for each overlapping frequency band.
freqs – the code will compute these based on values of fmin, fmax, numf
flo – the code will compute these based on values of fmin, fmax, numf
fhi – the code will compute these based on values of fmin, fmax, numf
vphmin – a minimum estimate of the phase velocity in km/s. suggest vphmin approx. 2.2km/s to capture seismic waves. (vphmin approx. .23 to capture infrasound)
ifenv: =0 use waveforms for signal cross-correlation and detection

=1 use envelopes for signal cross-correlation and detection

The next few parameters are specific to the case where envelopes are being used. They do not need to be set if ifenv=0.

fsenv – sampling rate used for the envelopes. Must be set to at least 2*fmax, also must be less than fsmin. For low values, the code runs faster but the precision of the azimuth and phase velocity may be slightly reduced.
tsta, tlta – short and long time averages (in seconds) for forming envelopes. Suggest setting tsta to be >= 1/fmin. Suggest setting tlta to be 30 to 100 times tsta.

snrcut - each envelope (or waveform amplitude) must have this SNR to do further computation
twin – duration of time window (in seconds) to analyze for each detection. Choose the maximum of a) 2xtransit time across array (which depends on both rtmx and vphmin),
b) 10 cycles at the lowest frequency (depends on fmin,
c) 300 points (depends on fsmin for waveforms, on fsenv for envelopes)

The following values could be changed by the user but should not be. Change at your own risk.

tcons - consistency cutoff = ratio of |t12+t23-t31|/(|t12|+|t23|+|t31|). Strongly suggest Tcons=0.1
In the next few lines, twin (which is set above) is altered to the nearest number such that an integer number of time windows fit in a 24-hour period.
tstep – time step between successive time windows. Suggest setting to twin/4 so that there is significant overlap between windows.
makeplots – =0 to suppress plotting maps, suggest setting = 1 for a first run-through of the program.
A few more variables are set here. Leave them alone; they have to do with plotting in case makeplots=1.

Running the Triad_detections code creates a binary file named triadsRaw_jjj_yyyy_flagN.out where *jjj* and yyyy are the julian date and year, respectively, that are set when calling Triad_detections(yyyy,jjj) and N is the value of triadflag set in the Triad_param1.m file above. This file is output to the directory /data/detections/yyyy and is read in by the program that sorts detections into source locations, see Appendix E.3.

(2) *Signal detection using arrays – set up* **Array_param1.m**

The code **Array_detections(Year, JDay),** located in /bin must be run to compute signal detections at arrays. The values Year and JDay are as described in Appendix E.1.

Appendix C describes how the code works. Before running it, information about the signal of interest must be supplied by the file **Array_param1.m**, also located in /bin. Most of the parameters are as described in Appendix E.1. For completeness, they are listed here. Only new parameters are shown in red.

ifDMC - as described in Appendix E.1
arrayflag - similar to the description for triadflag in Appendix E.1, but for array output

As above, the next 2 parameters are only necessary for ifDMC = 1
    net – as described in Appendix E.1
    loc as described in Appendix E.1
chan - as described in Appendix E.1
fsmin - as described in Appendix E.1
fsmax - as described in Appendix E.1

ifregion - as described in Appendix E.1
lon1, lon2 - as described in Appendix E.1
lat1, lat2 - as described in Appendix E.1

Occasionally, there may be arrays within the area of interest that perform poorly at the date of interest, for instance due to a persistent noise source nearby (for instance, an array located near an airport). In that case, it may be desirable to exclude that array from the study by specifying an exclusion zone surrounding the array. Several zones may be specified; the latitude and longitude limits for each zone must be given. An example for 1 exclusion zone is shown here.

ifexclusion = 1; - number of exclusion zones. Suggest ifexclusion = 0.
The following values are only specified if ifexclusion is not 0. For ifexlusion=N, N latitude and longitude limits must be specified; each must be an N x 2 matrix.
    ifexlon(1,:) = [-123.46 -123.44]; - longitude limits of exclusion zone
    ifexlat(1,:) = [48.64 48.66]; - latitude limits of exclusion zone

The next few parameters are dependent on the network and are used to group the sensors into arrays.
rmax - maximum radius across array in km. Consider the signal wavelength to set this value.
namin - minimum number of stations in an array. The minimum allowable value is 3.

Parameters that depend on the signal of interest
numf - as described in Appendix E.1
fmin - as described in Appendix E.1
fmax - as described in Appendix E.1
freqs - as described in Appendix E.1
flo - as described in Appendix E.1
fhi - as described in Appendix E.1
ifenv - as described in Appendix E.1. **I suggest using ifenv = 0 for arrays**
vphmin - as described in Appendix E.1
twin - as described in Appendix E.1

Although I recommend not using the option to form envelopes of the data prior to array processing, there may be applications in which it is useful (see Gibbons etal, 2008). In that case the following variables are needed.

tlta, tsta - as described in Appendix E.1

The following values could be changed by the user but should not be. As described in Appendix E.1 above, twin is altered in the next few lines such that an integer number of time windows fit in a 24-hour period.
tstep - as described in Appendix E.1, however more overlap is required for arrays than for triads. **Suggest setting tstep to twin/4 so that there is 75% overlap between windows.**
makeplots - as described in Appendix E.1
As described in Appendix E.1, a few more variables are set here but should be left alone.

Running the Array_detections code creates a binary file named arraysRaw_jjj_yyyy_flagN.out where *jjj* and yyyy are the julian date and year, respectively, that are set when calling Array_detections(yyyy,jjj) and N is the value of triadflag set in the Array_param1.m file above. This file is output to the directory /data/detections/yyyy and is read in by the program that sorts detections into source locations, see Appendix E.4.

(3) *Event association and source locations using triad detections –set* up **triad_param2.m**

The code **Triad_locateSources(Year, JDay,hr1,hr2)**, located in /bin is run to associate the signal detections made at triads with an event and then to estimate the source parameters.
Year – year of interest
JDay – julian day of interest
hr1, hr2 – these parameters are optional. hr1>=0; 24>=hr2 >hr1. Only use these if a partial day is of interest. Generally, these parameters are left out to examine the entire day. Note that if these parameters are used, the detections will be displayed in several formats, as a function of amplitude vs. time of day, cross-correlation vs. time of day, etc. This can be helpful for debugging, or for choosing optimal parameters.

Appendix D describes how the code works. The Triad_detections code must be run before Triad_locateSources. Also, some more information must be supplied by the file **Triad_param2.m**, also located in /bin. The parameters to be set in Triad_param2.m are explained here, along with suggested values. In the first line, Triad_param1 is called as the source location code carries over some information set previously, for instance, triadflag.

The first parameters either set a region for the event search, or let the program decide on limits based on where the sensors were in the Triad_detections code.
ifregion2 – region in which to do a coarse source location search. I suggest setting ifregion2=1. If it is set to 0, the code sets limits based on the sensor locations.
wstlon, estlon – west/east longitudes of coarse source location search grid. lon1< lon2.

sthlat, nthlat – south/north latitudes of coarse source location search grid. lat1< lat2. I recommend setting a broader search region than was set for the call to Triad_detections, as sources may fall outside the network specified in Triad_param1.m. *If ifregion2=0, then* wstlon, estlon, sthlat, nthlat *are set to null values*.

The next two parameters define the coarse time and spatial search grids.
tincoarse – coarse sample start time increments, in seconds. (see y-axis of Figure D3 in Appendix D).
coarsegrid – integer number of search points per degree for the coarse spatial grid, that is, the higher the value, the finer the search grid.
I recommend setting the time increment, tincourse, to be about the time it takes for a signal to transit from one coarse grid point to another. That is, the slower the signal, the greater tincoarse.

The next few parameters define thresholds that each detection must satisfy. Low quality detections are discarded.
ampcut – minimum signal to noise ratio. I recommend setting it to a higher value than for SNR in the param1.m file. Set to 0 to ignore this threshold entirely.
xccut – minimum cross-correlation value. I recommend xccut= 0.5, possibly reducing this to 0.4 for very large, poorly correlated triad signals.
Tconscut - fractional consistency in seconds. I recommend 0.05<=Tconscut <=0.1
vphmin – minimum phase velocity associated with detection (km/s).
vphmax - maximum phase velocity associated with detection (km/s).
I suggest fairly loose constraints for these values; vphmax > vphmin. Note that vphmin is already set in Triad_param1.m, see Appendix E.1 Comment the value out if you don't want to reset it.

The next few parameters govern how signals are grouped together to form an event association. See Figure D3 in Appendix D.

celermin – minimum group velocity (km/s)
celermax – maximum group velocity (km/s)
I suggest tighter constraints than for phase velocities, so vphmax>celermax>celermin>vphmin
mincluster - minimum number of detections to define a source
Afit – the source receiver azimuths must fit to within Afit degrees. I suggest Afit = 15.

clook – set this value to 1 for careful de-bugging, i.e to look at cluster of detections for each event. I recommend clook= 0
A few more variables are set here. Leave them alone; they have to do with plotting in case clook =1.

Running the Triad_locateSources code creates ascii format output files with the names CoarseLocations_yyyy_jjj_TriadflagN.txt, StationLocs_yyyy_jjj_TriadflagN.txt and StationLocs_yyyy_jjj_TriadflagN.txt where *yyyy* and *jjj* are the year and julian date, respectively, that are set when calling Triad_ locateSources (yyyy,jjj) and N is the triadflag value set in the Triad_param1.m file. These files are output to the directory /data/events/yyyy. The output files are explained further in Appendix F.

(4) Event association and source locations using array detections – set up **array_param2.m**

The description in Appendix D of how detections are sorted into events and then into source parameter estimates applies as well to array detections.

The code **Array_locateSources(Year, JDay,hr1,hr2)**, located in /bin is run to associate the signal detections made at arrays with events and then to estimate the source parameters for each. The values Year, JDay, hr1, and hr2 are described in Appendix E.3 above. The Array_detections code must be used to detect sources before Array_locateSources is run. Also, some more information must be supplied by the file **Array_param2.m**, also located in /bin. Almost all parameters are as described in Appendix E.3 above. For completeness, all parameters are listed here; new ones are shown in red.

ifregion2 – as described in Appendix E.3
wstlon, estlon - as described in Appendix E.3
sthlat, nthlat - as described in Appendix E.3
tincoarse - as described in Appendix E.3
coarsegrid - as described in Appendix E.3
xccut - as described in Appendix E.3. I recommend xccut= 0.4 for arrays.

Tconscut - minimum fractional time misfit for arrays (see Eq. C1). **I recommend Tconscut = 0.6 for arrays, i.e. much higher than for triads.**
fdetmin – minimum threshold for F detector (see Eq. C2). I recommend fdetmin = 3.

vphmin - as described in Appendix E.3
vphmax - as described in Appendix E.3
celermin - as described in Appendix E.3
celermax - as described in Appendix E.3
mincluster - as described in Appendix E.3 (Obviously this must be <= the number of arrays)
Afit - as described in Appendix E.3
clook and remaining variables - as described in Appendix E.3

Running the Array_locateSources code creates ascii format output files with the names CoarseLocations_yyyy_jjj_ArrayflagM.txt, StationLocs_yyyy_jjj_ArrayflagM.txt and StationLocs_yyyy_jjj_ArrayflagM.txt where *yyyy* and *jjj* are the julian year and date, respectively, that are set when calling Array_ locateSources (yyyy,jjj) and M is the triadflag value set in the Array_param1.m file. These files are output to the directory /data/events/yyyy. The output files are explained further in Appendix F.

(5) *Source locations using both triads and arrays* – set up TRIADArray_param2

Finally, the **LocateSources(Year, JDay,hr1,hr2)** code, located in /bin is run to associate the signal detections made at both triads and arrays with events and then to estimate the source parameters for each. The values Year, JDay, hr1, and hr2 are as described in Appendix E.3. Both the Triad_detections and Array_detections code must be run before LocateSources. Some information must be supplied by the file **TRIADArray _param2.m**, also located in /bin. Most parameters have been described in Appendices E.1 through E.4. For completeness, all parameters are listed here; new ones are shown in red.

triadflag - as described in Appendix E.1
arrayflag - as described in Appendix E.2

ifregion2 – as described in Appendix E.3
wstlon, estlon - as described in Appendix E.3
sthlat, nthlat - as described in Appendix E.3
tincoarse - as described in Appendix E.3
coarsegrid - as described in Appendix E.3

The next few parameters are thresholds for the triad detections
triadxccut – minimum cross-correlation value for triads (see Appendix E.3 for xccut)
triadTconscut – minimum fractional consistency for triads (see Appendix E.3 for Tconscut)
ampcut - as described in Appendix E.3

The next few parameters are thresholds for the array detections
arrayxccut – minimum cross-correlation value for arrays (see Appendix E.4 for xccut)
arrayTconscut – minimum fractional time misfit for arrays (see Appendix E.4 for Tconscut)
fdetmin - as described in Appendix E.4

Afit - as described in Appendix E.3
vphmin - as described in Appendix E.3
vphmax - as described in Appendix E.3
celermin - as described in Appendix E.3
celermax - as described in Appendix E.3
mincluster - as described in Appendix E.3 (this is the number of triad + array detections)
clook and remaining variables - as described in Appendix E.3

Running the LocateSources code creates ascii format output files with the names CoarseLocations_yyyy_jjj_BothflagMN.txt, StationLocs_yyyy_jjj_BothflagMN.txt and StationLocs_yyyy_jjj_BothflagMN.txt where *yyyy* and *jjj* are the julian year and date, respectively, that are set when calling LocateSources (yyyy,jjj) . The value MN is given by 10*triadflag + arrayflag, i.e., M and N are the triadflag and arrayflag values, respectively, from the TRIADarray_param2.m file, if each is less than 10. These files are output to the directory /data/events/yyyy. The output files are explained further in Appendix F.

APPENDIX F – USER MANUAL - OUTPUT

Running the Triad_detections and Array_detections codes produce binary output files of the type triadsRaw_jjj_yyyy_flagN.out and arraysRaw_jjj_yyyy_flagN.out as described in Appendix E.1 and E.2. These files give information on the attributes of the detections (i.e. their arrival time, azimuth, amplitude, etc). These are only interim files and their formats won't be described here. An interested reader could examine the Triad_detections and Array_detections codes to see how each is formatted, or examine read_rawTriads and read_rawArrays to see how these files are read in.

The ascii files produced by the Triad_locateSources, Array_locateSources, and LocateSources sources are described here.

(1) Coarse location files

The coarse location file produced by Triad_locateSources has 11 columns for grouping of detections that are common to each event and lists rough source parameter estimates for each. The events are not listed chronologically within each day file, but instead from the events with the greatest number of detections at unique triads, to the least.

column 1 – year
column 2 – julian day
column 3 – hour
column 4 – minute
column 5 – second
column 6 – approximate source latitude
column 7 – approximate source longitude
column 8 – number of triad detections used to define the event
column 9 – number of **unique** triads to define the event (always <= than column 8)
column 10 – minimum latitude error
column 11 – minimum longitude error

The coarse location file produced by Array_locateSources is similar except that column 8 is the number of array detections used to define the event and column 9 is the number of unique arrays to define the event (sometimes a given triad or array detects a signal more than once given the time overlap between windows).

The coarse location file produced by LocateSources, which finds events based on both triad and array detections, is similar except that it has 12 columns.
Columns 1-7 are as before,
column 8 - total number of detections from both triads and arrays
column 9 – number of unique triads that define the event

column 10 – number of unique arrays that define the event
Columns 11 and 12 are the minimum latitude and longitude errors, as above.

(2) Source location files – finer scale location and origin times

The source location file produced by Triad_locateSources has 19 columns for each event. These are sorted into approximate chronological order for each day, based on initial time estimates for the coarse locations.

Column 1 – event ID (epoch time)
Column 2 – year
Column 3 – julian day
Column 4 – hour
Column 5 – minute
Column 6 – second
Column 7 –source latitude (degrees)
Column 8 –source longitude (degrees)
Column 9 – minor axes of error ellipse (in km)
Column 10 – major axes of error ellipse (in km)
Column 11 – angle of error ellipse (in degrees) with respect to North
Column 12 – mean azimuth misfit (in degrees)
Column 13 – group slowness estimate (sec/km)
Column 14 - total number of detections for the event
Column 15 – number of unique detecting triads that detected the event
Column 16 - number of stations included in detecting triads
Column 17 – range from source to nearest detecting station (in km)
Column 18 – angle subtended by all detecting stations as seen from the source (see notes below)
Column 19 – gridfit (see notes below for description)

Column 18: The angle, in degrees, subtended by all stations that detect the source, as seen from the source. This value lies between 0 to 360. For instance, if the detecting stations surround the source, the value will be close to 360 degrees. Conversely, this value will be small if all the detecting stations lie within a narrow wedge to one side of the source.

Column 19: Gridfit is a quality control value. The value is 1 if the final source location in this file is beyond the source location estimate given in the coarse location file, meaning that the final value is far from the initial estimate The value is 2 if the vslow estimate (column 13) is less than or equal to zero. The default for this value is 0. However, there are also several other ways of assessing quality of the estimated source location. The larger the area of the error ellipse (given by pi*column 9* column 10) the poorer the source location. The number of unique triads used to detect the event (column 15) is a measure of the both the source magnitude and its location accuracy. The range from the nearest detecting station (column 17) and the angle subtended by all stations that detected the event (column 18) may also be used to assess the quality of the source estimates.

The source location file produced by Array_locateSources also contains 19 columns. The only difference is that, for lines 14 through 16, the description would read 'arrays' instead of ‘triads.

The source location file produced by LocateSources has 20 columns. Columns 1 through 13 are as described above. The remainder of the columns are
Column 14 – total number of detections for the event, both triads and arrays
Column 15 – number of unique detecting triads that detected the event
Column 16 – number of unique detecting arrays that detected the event
Column 17 – - number of stations included in detecting triads and arrays
Columns 18 through 20 are as described for columns 17 through 19 above.

(3) Station Location files

All station location files have 3 columns, whether they are produced by Triad_locateSources, Array_locateSources, or LocateSources.

Column 1 – event ID (epoch time)
Column 2 – station latitude
Column 3 – station longitude

The event ID of column 1 can be matched against that of the source location files above. All station locations within each triad or each detecting array are listed.