Supplement to Journal of Open Research Software

An ObsPy library for event detection and seismic attribute calculation: preparing waveforms for automated analysis

Ross J. Turner[1]*, Rebecca B. Latto[1], Anya M. Reading[1,2]
[1] *School of Natural Sciences, University of Tasmania, Private Bag 37, Hobart, 7001, Australia*
[2] *Institute for Marine and Antarctic Studies, University of Tasmania, Private Bag 129, Hobart, TAS 7001, Australia*

18 January 2021

SEISMIC ATTRIBUTES DOCUMENTATION

This document describes the functions in the *seismic attributes* library and provides examples of accepted usage to download waveforms from online repositories, to identify events using seismic records from one or more stations (and/or components), and to derive attributes of the identified signals for use in clustering or other semi-automated analyses. The code is written in Python 2/3 and imports standard packages including *numpy*, *pandas*, *matplotlib* and *seaborn*, and the specialised package for observational seismology, *obspy*. Documentation for private functions may be accessed through the comments in the code itself. Refer to package documentation (e.g. for *obspy*) also, which is mostly not repeated here.

get_waveforms

seismic_attributes.get_waveforms(network, station, location, channel, starttime, endtime, event_buffer=3600, waveform_name='waveforms', station_name='stations', client=['IRIS', 'LMU', 'GFZ'], download=True)

Function to download waveform data from an online seismic repository and save to a local directory or external drive. The waveform data are split into files containing a single day of data to enable fast recall and storage across multiple drives. The function checks if data for the requested station, channels (components) and time period are present in the specified local directory before attempting to download new waveform data. The requested waveform data are output as a single *obspy* `Stream` object.

Parameters:

network : *str*
A single SEED or data center defined network code; wildcards are not allowed.

station : *str*
A single SEED station code; wildcards are not allowed.

location : *str*
A single SEED location identifier (often blank or '0'); wildcards are not allowed.

channel : *str or list*
One or more SEED channel codes; multiple codes are entered as a list; e.g. ['BHZ', 'HHZ']; wildcards are not allowed.

starttime : *UTCDateTime*
Start time series on the specified start time (or after that time in the case of a data gap).

endtime : *UTCDateTime*
End time series (one sample) before the specified end time.

event_buffer : *float, optional*
Minimum duration of data to buffer before and after the requested time period; expected units are seconds. This is used to capture the full length of events that extend beyond the time period. The default value is 3600 seconds.

waveform_name : *str or path, optional*
Path to directory to read (check for) and write waveform data (an existing file of the same name will not be overwritten). The default location is a directory named *waveforms* in the working directory.

station_name : *str or path, optional*

Path to directory to read (check for) and write station data (location coordinates, elevation etc, as provided by data repository). The default location is a directory named *stations* in the working directory.

client : *str or list, optional*

One or more clients to use to download the requested waveform data if it does not already exist in the specified local directory. Multiple clients are entered as a list; e.g. ['IRIS', 'GFZ']. By default IRIS, LMU and GFZ are queried.

download : *bool, optional*

Specify whether days with missing waveform data are to be downloaded from client; e.g. True or False, alternatively 1 or 0. Missing data are downloaded by default.

Returns: **stream : *Stream***

A stream object with one or more traces for each component at the requested seismometer.

get_events

seismic_attributes.get_events(stream, starttime, endtime, signal_type='amplitude', station_name='stations', trigger_type='recstalta', avg_wave_speed=2, thr_event_join=0.5, thr_coincidence_sum=-1, thr_on=5, thr_off=1, **options)

Function to detect events from seismic waveform data using STA/LTA-type algorithms. The input waveforms are denoted by seismometer and channel (e.g. 'BH?', 'HH?', 'LH?'); the signal from seismometers with multiple components (i.e. 'Z', 'N', 'E') are combined into a single waveform using the Euclidean norm. The triggering algorithm is applied to the resulting amplitude or energy waveform. Small gaps between triggered events are removed before the combined event details are written to the (reference) event catalogue. If multiple seismometers are present the user can specify the minimum number of seismometers on which an event must be detected for that event to be included in the catalogue.

Parameters: **stream : *Stream***

Stream containing waveform data for each component from one or more seismometers.

starttime : *UTCDateTime*

Limit results to time series samples starting on the specified start time (or after that time in the case of a data gap).

endtime : *UTCDateTime*

Limit results to time series ending (one sample) before the specified end time.

signal_type : *str, optional*

Apply the event detection algorithm to the 'amplitude' (i.e. absolute value) or 'energy' (i.e. amplitude-squared) waveform. The event detection algorithm is applied to the amplitude waveform by default.

station_name : *str or path, optional*

Path to directory to read station data, specifically the GPS coordinates. The default location is a directory named *stations* in the working directory.

trigger_type : *str, optional*

The trigger algorithm to be applied (e.g. 'recstalta'). See e.g. obspy.core.trace.Trace.trigger() for further details. The recursive STA/LTA algorithm is applied by default.

avg_wave_speed : *float, optional*

The speed at which seismic waves propagate through the local medium, contributing to a delay between detections in elements of a seismic array. The distance between the closest set of n (defined by the `thr_coincidence_sum` parameter) seismometers defines a critical distance the seismic waves must cover for coincidence triggering to detect events simultaneously at n seismometers. The default value for the average wave speed is 2 km/s.

thr_event_join : *float, optional*

The maximum duration of gaps between triggered events before those events are considered separate. Joined events are reported as a single event in the (reference) event catalogue. The maximum gap duration is assumed to be 0.5 seconds by default.

thr_coincidence_sum : *int, optional*

The number of seismometers, n, on which an event must be detected for that event to be included in the (reference) event catalogue. By default an event must be detected at every seismometer.

thr_on : *float, optional*

Threshold for switching single seismometer trigger on. The default value is a threshold of 5.

thr_off : *float, optional*

Threshold for switching single seismometer trigger off. The default value is a threshold of 1.

options

Necessary keyword arguments for the respective trigger algorithm that will be passed on. For example 'sta' and 'lta' for any STA/LTA variant (e.g. sta=3, lta=10). Arguments 'sta' and 'lta' (seconds) will be mapped to 'nsta' and 'nlta' (samples) by multiplying by the sampling rate of trace (e.g. sta=3, lta=10 would call the trigger algorithm with 3 and 10 seconds average, respectively).

Returns: **events : *DataFrame***

A pandas dataframe containing the events in the form of a (reference) event catalogue with eight columns including the reference start time and duration of the event based on coincidence triggering. The format of the event catalogue is detailed in the Event and Trace Catalogues section of this documentation.

traces : *DataFrame*

A pandas dataframe containing the trace (metadata) with eight columns including the start time and duration of the triggers for each trace based on single station triggering. The format of the trace (metadata) catalogue is detailed in the Event and Trace Catalogues section of the documentation.

plot_events

seismic_attributes.plot_events(events, stream, starttime, endtime, filename=None)

Generates a plot of the seismic waveform used in event detection from first seismometer included in the *obspy* `Stream` with identified events highlighted. The signal is plotted as either an amplitude or energy waveform based on the signal type used in the event triggering algorithm in the `get_events` function. Note that the purpose of this plot is to gain an overview of the events detected (see also following function, if a manual visual check on a given event is desired). Multiple days of data are presented on separate pages in the output .pdf document.

Parameters: **events : *DataFrame***

List of events or traces stored as a pandas dataframe including as a minimum a column named 'time' (or 'ref_time') for the start time of the event and a column named 'duration' (or 'ref_duration') for the length of the event.

stream : *Stream*

Stream object containing waveform data for each component from one or more seismometers. The waveform for the first seismometer in the stream is plotted using the signal type (amplitude or energy) specified in the event detection function.

starttime : *UTCDateTime*

Limit results to time series samples starting on the specified start time (or after that time in the case of a data gap).

endtime : *UTCDateTime*

Limit results to time series samples ending (one sample) before the specified end time.

filename : *str or path, optional*

Path to directory to write the .pdf document of plots. The default filename/path is '[station name]_event_plot.pdf' located in the working directory.

Returns: ***None***

plot_waveforms

seismic_attributes.plot_waveforms(events, event_id, start_buffer=10, end_buffer=30, filename=None, waveform_name='waveforms', station_name='stations', client=['IRIS', 'LMU', 'GFZ'], download=True)

Generates a plot of waveforms at all seismometers with a detection of a given event in the trace (metadata) catalogue. The purpose of this function is to enable a manual, visual check on a given event. The waveforms are separated into panels by the recording channel; only seismometers with an identical set of channels to that of the first seismometer (in order of input with a detection) are included on the plot. The function searches for the required waveform data in the specified local directory before attempting to download new waveform data.

Parameters: **events : *DataFrame***

List of traces stored as a pandas dataframe including as a minimum a column named 'time' for the start time of the event and a column named 'duration' for the length of the event. The input must be either the trace (metadata) catalogue or trace attribute catalogue, not the (reference) event catalogue.

event_id : *str*

Unique time-based identifier for an event of the form 'yyyymmddThhmmssZ', , for the year 'yyyy', month 'mm', day of month 'dd', hour 'hh', minute 'mm' and second 'ss'. These are found in the (reference) event and trace (metadata) catalogues.

start_buffer : *float, optional*

Number of seconds of waveform data to display before the identified start time of event at each seismometer (in trace catalogue). The default value is 10 seconds.

end_buffer : *float, optional*

Number of seconds of waveform data to display after the identified end time of event at each seismometer (in trace catalogue). The default value is 30 seconds.

filename : *str or path, optional*

Path to directory to write the .pdf document of plots. The default filename/path is '[event_id]_plot.pdf' located in the working directory.

waveform_name : *str or path, optional*

Path to directory to read (check for) and write waveform data (an existing file of the same name will not be overwritten). The default location is a directory named *waveforms* in the working directory.

station_name : *str or path, optional*

Path to directory to read (check for) and write station data (location coordinates, elevation etc, as provided by data repository). The default location is a directory named *stations* in the working directory.

client : *str or list, optional*

One or more clients to use to download the requested waveform data if it does not already exist in the specified local directory. Multiple clients are entered as a list; e.g. ['IRIS', 'GFZ'].

download : *bool, optional*

Specify whether days with missing waveform data are to be downloaded from client; e.g. True or False, alternatively 1 or 0. Missing data are downloaded by default.

Returns: ***None***

get_attributes

seismic_attributes.get_attributes(events, stream, *attributes)

Function to calculate characteristics of seismic waveform data at times of identified seismic events. The event catalogue output by the `get_events` function is used to extract the waveform data at the relevant times from an input *obspy* `Stream` object. Pre-defined or user-defined attribute functions are applied to the waveform data from each seismometer and channel (e.g. 'BH?', 'HH?', 'LH?') over the duration of each event. The attribute function specifies the calculation of one or more characteristics based on the component waveform. The value for each requested attribute is tabulated for the event catalogue. If multiple seismometers are present, the average value of each attribute is returned for a given event.

Parameters: **events : *DataFrame***

List of (reference) events or traces (metadata) stored as a pandas dataframe including at minimum a column named 'time' (or 'ref_time') for the start time of the event and a column named 'duration' (or 'ref_duration') for the length of the event.

stream : *Stream*

Stream object containing waveform data for each component from one or more seismometers.

attributes

Pre-defined or user defined attribute functions to apply to the waveform data. Each function is included separately and comma separated; e.g. get_attributes(..., seismic_attributes.waveform_attributes, user_defined_attribute, seismic_attributes.polarity_attributes).

Returns: **attributes : *DataFrame***

Return a pandas dataframe with the attributes of each event or trace. The dataframe includes columns for the start time and duration of the event in addition to event data from the event or trace catalogue. The remaining columns are for the requested attributes and are named as specified in their respective attribute function. The format of the attribute catalogue is detailed in the Event and Trace Catalogue section of the documentation.

plot_attributes

seismic_attributes.plot_attributes(attributes, filename=None)

Box-and-whisker plot of the distribution of values for each attribute derived for the event catalogue using the `get_attributes` function.

Parameters: **attributes : *DataFrame***

List of events and their attributes stored as a pandas dataframe including columns named 'time' (or 'ref_time') and 'duration' (or 'ref_duration') for the start time and duration of the event.

filename : *str or path, optional*

Specify path to directory and filename to write the .pdf plot. The default filename/path is 'attribute_plot.pdf' located in the working directory.

Returns: ***None***

plot_correlations

seismic_attributes.plot_correlations(attributes, filename=None, plot_type='matrix')

Correlation matrix, or alternatively a corner plot, of attributes derived for the event catalogue using the `get_attributes` function. The correlation matrix is a lower triangular matrix coloured by the value of the correlation coefficient; the numerical value of the Pearson correlation coefficient is also shown in text. The corner plot is also a lower triangular matrix with scatter plots of the correlation between each attribute and bar charts of the distribution of each attribute along the diagonal.

Parameters: **attributes : *DataFrame***

List of events and their attributes stored as a pandas dataframe including columns named 'time' (or 'ref_time') and 'duration' (or 'ref_duration') for the start time and duration of the event.

filename : *str or path, optional*

Specify path to directory and filename to write the .pdf plot. The default filename/path is 'correlation_plot.pdf' located in the working directory.

plot_type : *str, optional*

Specify whether correlations are to be plotted as a correlation matrix ('matrix') or corner (or pair) plot ('corner'). By default a correlation matrix is plotted.

Returns: ***None***

ATTRIBUTE FUNCTIONS

The *seismic attributes* code provides three bundles of attribute functions that can be applied directly to waveform data. These attribute functions implement the waveform, spectral and polarity attributes defined by Provost et al. (2017); we do not include a few attributes which are network specific or may lack stability in wider application. The attributes included in the three bundles of attribute functions are listed under the function definitions. Private functions exist for subsets of the waveform, spectral and polarity attributes and are documented in the code.

waveform_attributes

seismic_attributes.waveform_attributes(component_stream, event_start, event_stop)

Function to calculate the waveform attributes of an event from the waveform data passed on from the `get_attributes` function. Attributes 1 through 22 are returned as log-scaled (base 10) values.

Parameters: **component_stream : *Stream***

Stream object containing waveform data for each component from a single seismometer between the start and end time of the event.

event_start : *UTCDateTime*

Limit results to time series samples starting on the specified start time (or after that time in the case of a data gap); the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples ending (one sample) before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

A tuple containing the name of the attributes and their values; i.e. ['attribute_1', ..., 'attribute_22'] as a list of strings and the values for the waveform attributes of the event as a list of floats. Attributes 1 through 22 are returned as log-scaled (base 10) values.

Number	Description
1	Duration
2	Ratio of the mean over the maximum of the envelop signal
3	Ratio of the median over the maximum of the envelop signal
4	Ratio between ascending and descending time
5	Kurtosis of the raw signal (peakness of the signal)
6	Kurtosis of the envelope
7	Skewness of the raw signal
8	Skewness of the envelope
10	Energy in the first third of the autocorrelation function
11	Energy in the remaining part of the autocorrelation function
12	Ratio of 11 and 10
13–17	Energy of the signal filtered in 5–10 Hz, 10–50 Hz, 5–70 Hz, 50–100 Hz, and 5–100 Hz
18–22	Kurtosis of the signal in 5–10 Hz, 10–50 Hz, 5–70 Hz, 50–100 Hz, and 5–100 Hz frequency range

spectral_attributes

seismic_attributes.spectral_attributes(component_stream, event_start, event_stop)

Function to calculate the spectral attributes of an event from the waveform data passed on from the `get_attributes` function. Attributes 24 through 37 are returned as log-scaled (base 10) values.

Parameters: **component_stream : *Stream***

Stream object containing waveform data for each component from a single seismometer between the start and end time of the event.

event_start : *UTCDateTime*

Limit results to time series samples starting on the specified start time (or after that time in the case of a data gap); the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples ending (one sample) before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

A tuple containing the name of the attributes and their values; i.e. ['attribute_24', ..., 'attribute_40'] as a list of strings and the values for the spectral attributes of the event as a list of floats. Attributes 24 through 37 are returned as log-scaled (base 10) values.

Number	Description
24	Mean of the discrete Fourier transform (DFT)
25	Max of the DFT
26	Frequency at the maximum
27	Central frequency of the 1st quartile
28	Central frequency of the 2nd quartile
29	Median of the normalized DFT
30	Variance of the normalized DFT
34–37	Energy in DFT for $[0, \frac{1}{4}]$Nyf, $[\frac{1}{4}, \frac{1}{2}]$Nyf, $[\frac{1}{2}, \frac{3}{4}]$Nyf, $[\frac{3}{4}, 1]$Nyf
38	'Spectral centroid' (as defined by Provost et al.)
39	Gyration radius
40	Spectral centroid width

polarity_attributes

seismic_attributes.polarity_attributes(component_stream, event_start, event_stop)

Function to calculate the polarity attributes of an event from the waveform data passed on from the `get_attributes` function. This function is only applicable for seismometers with a three component signal (i.e. 'Z', 'N' and 'E' components).

Parameters: **component_stream : *Stream***

Stream object containing waveform data for each component from a single seismometer between the start and end time of the event.

event_start : *UTCDateTime*

Limit results to time series samples starting on the specified start time (or after that time in the case of a data gap); the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples ending (one sample) before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

A tuple containing the name of the attributes and their values; i.e. ['attribute_68', ..., 'attribute_71'] as a list of strings and the values for the polarity attributes of the event as a list of floats.

Number	Description
68	Rectilinearity
69	Azimuth
70	Dip
71	Planarity

The form of user-defined attribute functions must use the same syntax as these three bundled attribute functions. The components of the signal included in the `component_stream` may be combined into a single trace using the Euclidean norm with the `group_components` function below. The private attribute functions included in the code provide a good guide for implementing new attribute functions.

group_components

seismic_attributes.group_components(stream, signal_type='amplitude')

Function to calculate the Euclidean norm of the waveform amplitude for seismometers with multiple component measurements. The normal can be returned as an absolute value amplitude waveform or an energy waveform.

Parameters: **stream : *Stream***

Stream containing waveform data for each component from one or more seismometers between the start and end time of the event.

signal_type : *str, optional*

Specify whether components are grouped as an 'amplitude' (i.e. absolute value) or 'energy' (i.e. amplitude-squared) waveform. The components are grouped as an amplitude waveform by default.

Returns: **stream_list : *list***

Return a list of streams for each seismometer with the data representing the amplitude or energy of the waveform measured by taking the normal of the signal from each component. The first stream is accessed as group_components(...)[0] and the trace of that stream as group_components(...)[0][0].

EVENT AND TRACE CATALOGUES

Event catalogue

The (reference) event catalogue is stored as a *pandas* `DataFrame` and is the first of two outputs from the `get_events` function. The columns included in the event catalogue are detailed in the following table:

Name	Type	Description	Units
event_id	*str*	Unique time-based identifier for the event of the form 'yyyymmddThhmmssZ', for the year 'yyyy', month 'mm', day of month 'dd', hour 'hh', minute 'mm' and second 'ss'. The reference time discussed below is used for this identifier.	–
stations	*list*	Station codes of seismometers with a detection of the event for at least part of its duration. If coincidence triggering is used only events with n (i.e. `thr_coincidence_sum`) simultaneous detections are included in the catalogue.	–
network_time	*float*	Travel time of a seismic wave propagating between the n closest seismometers that detect the event. It is assumed that each seismometer has a high likelihood of detecting an event when calculating the expectation value. If $n = 1$ the travel time is zero.	seconds
ref_time	*UTCDateTime*	Start time of event based on coincidence triggering algorithm. This reference time corresponds to the first instance when N seismometers have simultaneous detections of the event. The reference time preceeds this start time by half the network time for the n closest seismometers, and the stop time forward by the same amount, to maintain the correct duration for the actual event (i.e. not shortened due to triggers mis-aligned in time).	–
ref_duration	*float*	Duration of event between reference start and stop time; i.e. the time period when at least n seismometers have simultaneous detections of the event. The reference duration is greater than this coincidence time period by the network time for the n closest seismometers to maintain the actual event duration (i.e. not shortened due to triggers mis-aligned in time). Small gaps between time periods with n detections of up to length `thr_event_join` are ignored.	seconds
ref_amplitude	*float*	Peak amplitude of the seismic event between the reference start and stop time. In seismic arrays, the reference amplitude is taken as the average of the peak amplitude at the set of n seismometers with the largest amplitudes.	(amplitude)
ref_energy	*float*	Energy (i.e. integral of the amplitude squared with respect to time) of the seismic event between the reference start and stop time. In seismic arrays, the reference energy is taken as the average of the energy at the set of n seismometers with the largest energies.	$(\text{amplitude})^2 \cdot$ seconds
signal_type	*str*	Record of transformation applied to the raw waveform data used in the event triggering algorithm; i.e. the Euclidean norm of multiple component signals is stored as either an 'amplitude' (i.e. absolute value) or 'energy' (i.e. amplitude-squared) waveform.	–

Trace catalogue

The trace (metadata) catalogue is stored as a *pandas* `DataFrame` and is the second of two outputs from the `get_events` function. All seismometers and channels in a seismic array are included in a single output file. The columns included in the trace catalogue are detailed in the following table:

Name	Type	Description	Units
event_id	*str*	Unique time-based identifier matching the identifier for an event in the event catalogue.	–
station	*str*	Station code of the seismometer used to record this trace with a detection of the event.	–
components	*list*	Channel codes for each waveform component included in the Euclidean norm for this trace.	–
time	*UTCDateTime*	Start time of the trigger for this trace based on the single station triggering algorithm. The trigger must at least partially overlap the time period covered by the reference start and stop times of the event. If multiple triggers are present in this region the start time is taken as the start time of the first partially overlapping trigger. The stop time is similarly taken as the stop time of the last partially overlapping trigger.	–
duration	*float*	Duration of the trigger for this trace based on the single station triggering algorithm; i.e. duration between the start and stop time of any triggers partially overlapping the reference start and stop times of the event.	seconds
amplitude	*float*	Peak amplitude of the trace between the trigger start and stop time.	(amplitude)
energy	*float*	Energy (i.e. integral of the amplitude squared with respect to time) of the trace between the trigger start and stop times.	$(\text{amplitude})^2 \cdot$ seconds
signal_type	*str*	Record of transformation applied to the raw waveform data used in the event triggering algorithm; i.e. the Euclidean norm of multiple component signals is stored as either an 'amplitude' (i.e. absolute value) or 'energy' (i.e. amplitude-squared) waveform.	–

Attribute catalogue

The attribute catalogue is stored as a *pandas* `DataFrame` and is the sole output from the `get_attributes` function. The format of the output file is dependent on whether the (reference) event or trace (metadata) catalogue is passed into the `get_attributes` function as an input. The columns included in the attribute catalogue are detailed in the following tables for the event and trace catalogues as inputs respectively:

Event catalogue attributes

Name	Type	Description	Units
event_id	*str*	Unique time-based identifier matching the identifier for an event in the event catalogue.	–
stations	*list*	as for event catalogue	–
network_time	*float*	as for event catalogue	seconds
ref_time	*UTCDateTime*	as for event catalogue	–
ref_duration	*float*	as for event catalogue	seconds
attribute_xx	*list*	Output of the `attribute_xx` attribute function when applied to the trace at each seismometer between the reference start and stop time. The output values for the seismometers are stored in a list. The average value of this list is calculated in some of our plotting functions, however the user should make an informed decision based on the physical meaning of the attribute; e.g. angles should not be averaged as they wrap from 360 around to 0 degrees.	(units)

Trace catalgoue attributes

Name	Type	Description	Units
event_id	*str*	Unique time-based identifier matching the identifier for an event in the event catalogue.	–
station	*str*	as for trace catalogue	–
components	*list*	as for trace catalogue	–
time	*UTCDateTime*	as for trace catalogue	–
duration	*float*	as for trace catalogue	seconds
attribute_xx	*float*	Output of the `attribute_xx` attribute function when applied to the trace between the trigger start and stop time.	(units)

EXAMPLE USAGE

Example code to produce figures shown in this documentation

Download a single day of data in all three components of the broad-band channel at Ilulissat, Greenland (Network, DK; Station code, ILULI) from the start of 2018; by default data will be downloaded on the preceding and subsequent day to provide a one hour buffer for event detection.

```
>>> import seismic_attributes as sa
>>> t1 = sa.UTCDateTime("2018-01-01T00:00:00.0Z")
>>> t2 = sa.UTCDateTime("2018-01-02T00:00:00.0Z")
>>> stream = sa.get_waveforms('DK', 'ILULI', '', ['BHE', 'BHN', 'BHZ'], t1, t2)
```

Detect events using the recursive STA/LTA algorithm based on the Euclidean norm (amplitude) of the seismic waveform; standard values are chosen for each parameter. Events separated by less than 5 seconds are joined together. Additional seismometers are included by passing in an *obspy* `Stream` object with waveforms from multiple seismometers; e.g. `stream = stream1 + stream2 + stream3`. The output *pandas* `DataFrame` for the event and trace catalogues are written to .csv files for later use.

```
>>> events = sa.get_events(stream, t1, t2, trigger_type='recstalta', sta=1, lta=1000,
        thr_on=3, thr_off=1, thr_event_join=5)
>>> events[0].to_csv('event_catalogue.csv')
>>> events[1].to_csv('trace_catalogue.csv')
```

Plot (Euclidean norm) waveform data for Ilulissat and highlight identified events. The event catalogue is used by default if the `tuple` containing both catalogues is passed as an input. The expected output is shown in Figure 1; the plot will be saved as 'ILULI_event_plot.pdf' in the working directory.

```
>>> sa.plot_events(events, stream, t1, t2)
```

Plot the waveform data as the three components of the signal for the event at 02:54:46 UTC on 1 January 2018. The trace catalogue is used by default if the `tuple` containing both catalogues is passed as an input. Waveform data are plotted from 30 seconds prior to the start of the event up to 60 seconds after the end of the record. The expected output is shown in Figure 2; the plot will be saved as '20180101T025446Z_plot.pdf' in the working directory.

```
>>> sa.plot_waveforms(events, '20180101T025446Z', start_buffer=30, end_buffer=60)
```

Calculate all the spectral and polarity attributes in the *seismic attributes* library for each event included in the event catalogue. The output *pandas* `DataFrame` is written to a .csv file for later use.

```
>>> attributes = sa.get_attributes(events, stream, sa.spectral_attributes,
        sa.polarity_attributes)
>>> attributes.to_csv('attribute_catalogue.csv')
```

Plot a box-and-whisker diagram of distribution of values for each requested attribute over the events in the catalogue. The expected output is shown in Figure 3; the plot will be saved as 'attribute_plot.pdf' in the working directory.

```
>>> sa.plot_attributes(attributes)
```

Plot the correlation matrix for the attributes derived for the event catalogue. The optional `plot_type` input is set to 'corner' to produce a corner plot. The expected output is shown in Figure 4; the plot will be saved as 'correlation_matrix.pdf' in the working directory.

```
>>> sa.plot_correlations(attributes)
```

Example code to produce figures shown in Turner et al. (2021, in review)

The corner (or pair) plot in Figure 3 (of publication above) uses the attribute catalogue for the events detected at Ilulissat, Greenland on 1 January 2018 (as above), though only for the spectral attributes. The `get_attributes` function therefore needs to be run again (after the above code) followed by the additional line of code to produce the plot. That is,

```
>>> attributes = sa.get_attributes(events, stream, sa.spectral_attributes)
>>> sa.plot_correlations(attributes, plot_type='corner')
```

The waveform plot in Figure 2 (of publication above) uses the trace (metadata) catalogue for four seismometers on the Whillans Ice Stream in Antarctica from 13:35:37 on 16 December 2010. This dataset needs to be downloaded and the event catalogue created before the plot can be produced. We require a minimum of three of the four seismometers to have simultaneous detections for an event to be identified (`thr_coincidence_trigger` = 3). The full code is shown below:

```
>>> import seismic_attributes as sa
>>> t1 = sa.UTCDateTime("2010-12-16T01:00:00.0Z")
>>> t2 = sa.UTCDateTime("2010-12-18T0:00:00.0Z")
>>> stream1 = sa.get_waveforms('2C', 'BB01', '', ['HHE', 'HHN', 'HHZ'], t1, t2)
>>> stream3 = sa.get_waveforms('2C', 'BB03', '', ['HHE', 'HHN', 'HHZ'], t1, t2)
>>> stream4 = sa.get_waveforms('2C', 'BB04', '', ['HHE', 'HHN', 'HHZ'], t1, t2)
>>> stream6 = sa.get_waveforms('2C', 'BB06', '', ['HHE', 'HHN', 'HHZ'], t1, t2)
```

```
>>> events = sa.get_events(stream1+stream3+stream4+stream6, t1, t2,
        trigger_type='multistalta', sta=0.03, lta=100, delta_sta=18, delta_lta=56,
        epsilon=10, avg_wave_speed=2, thr_event_join=10, thr_coincidence_sum=3)
>>> events[1].to_csv('whillians_traces.csv')
```


```
>>> events = sa.get_events(stream1+stream3+stream4+stream6, t1, t2,
        trigger_type='multistalta', sta=0.03, lta=100, delta_sta=18, delta_lta=56,
        epsilon=10, avg_wave_speed=2, thr_event_join=10, thr_coincidence_sum=3)
>>> events[1].to_csv('whillians_traces.csv')
```


```
>>> sa.plot_waveforms(events, '20101216T133748Z')
```

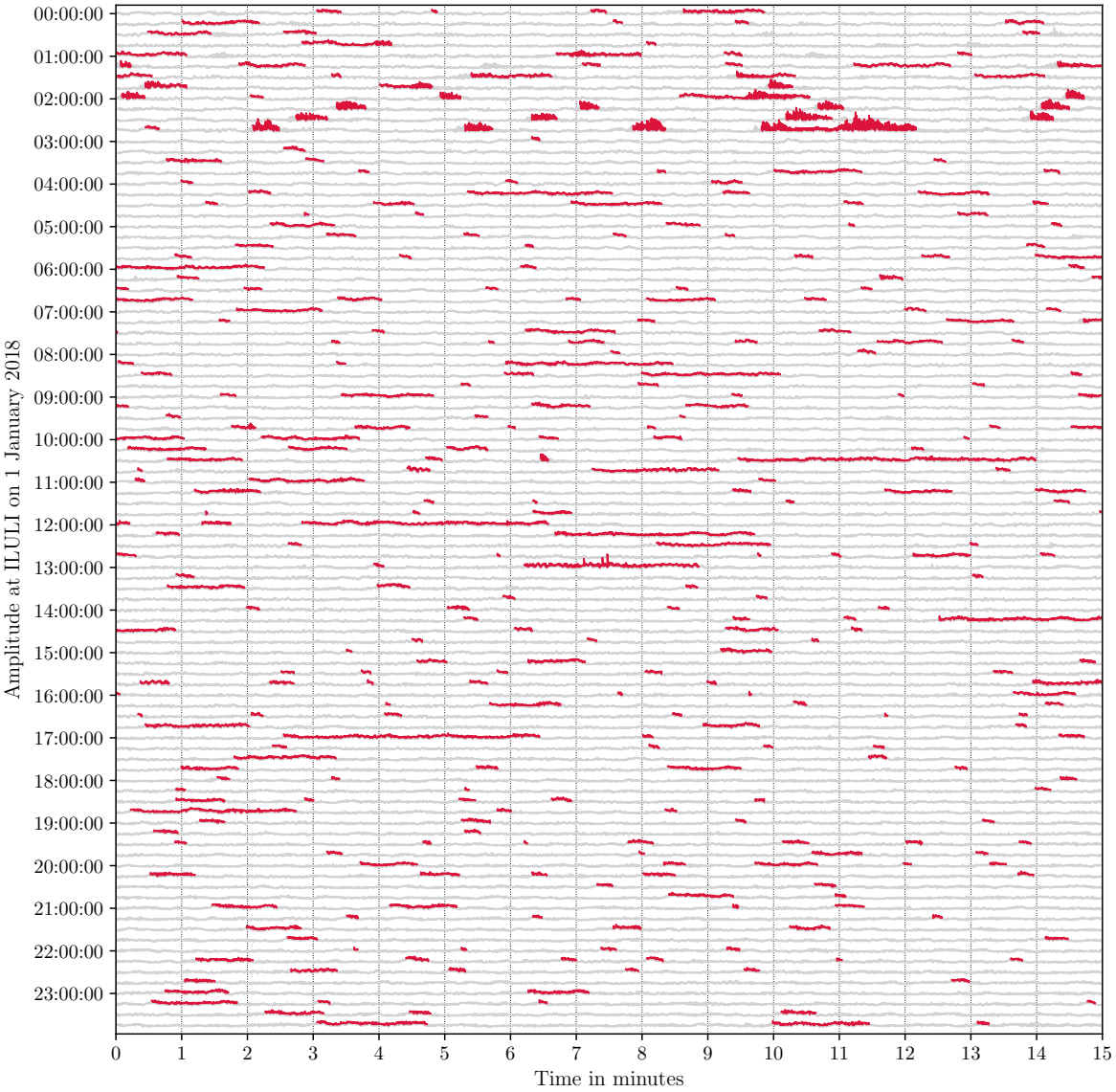

Figure 1. Events detected at Ilulissat, Greenland on 1 January 2018 using the recursive STA/LTA algorithm, shown on the (Euclidean norm) amplitude.

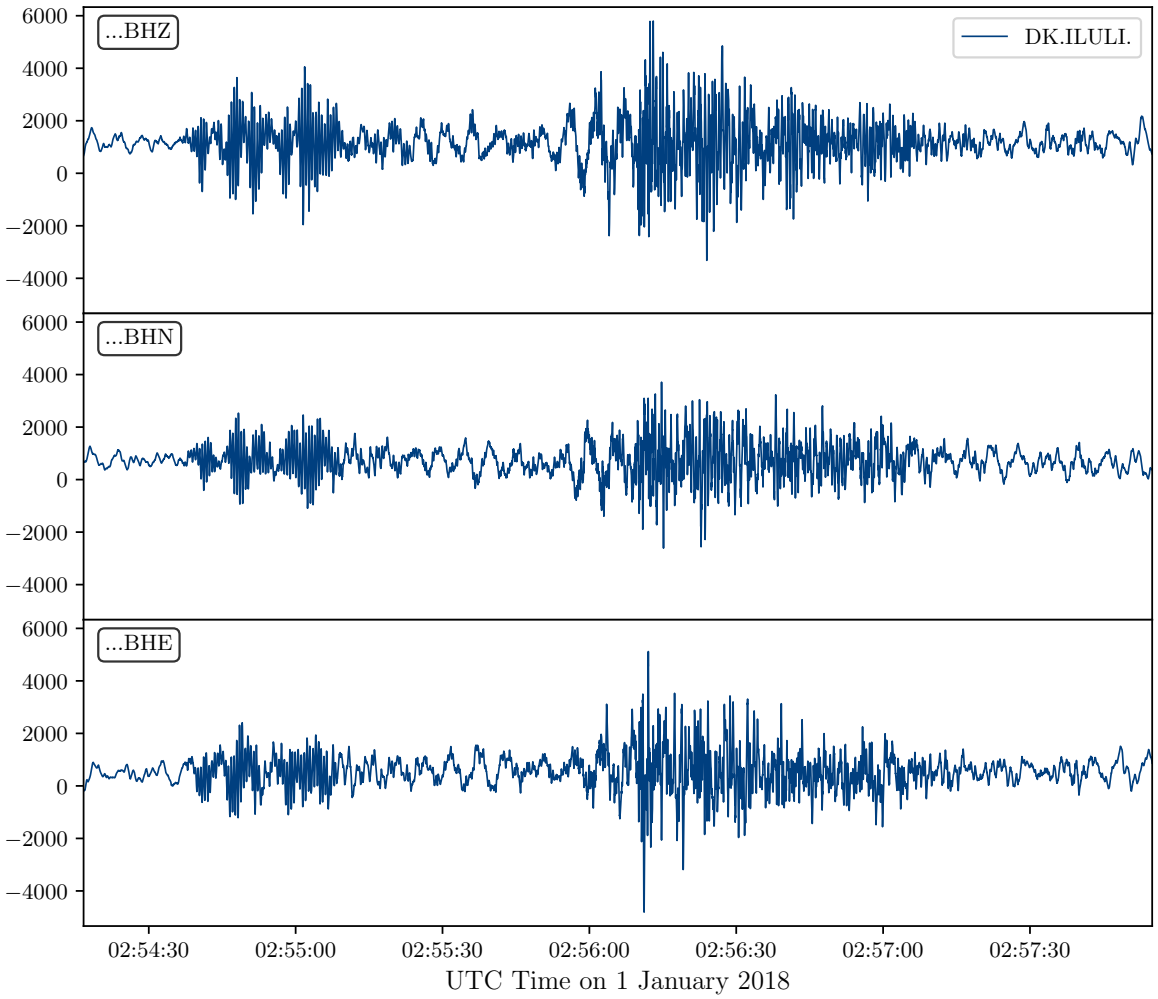

Figure 2. Three component waveform data for the event detected at Ilulissat, Greenland from 02:54:46 UTC on 1 January 2018 using the recursive STA/LTA algorithm.

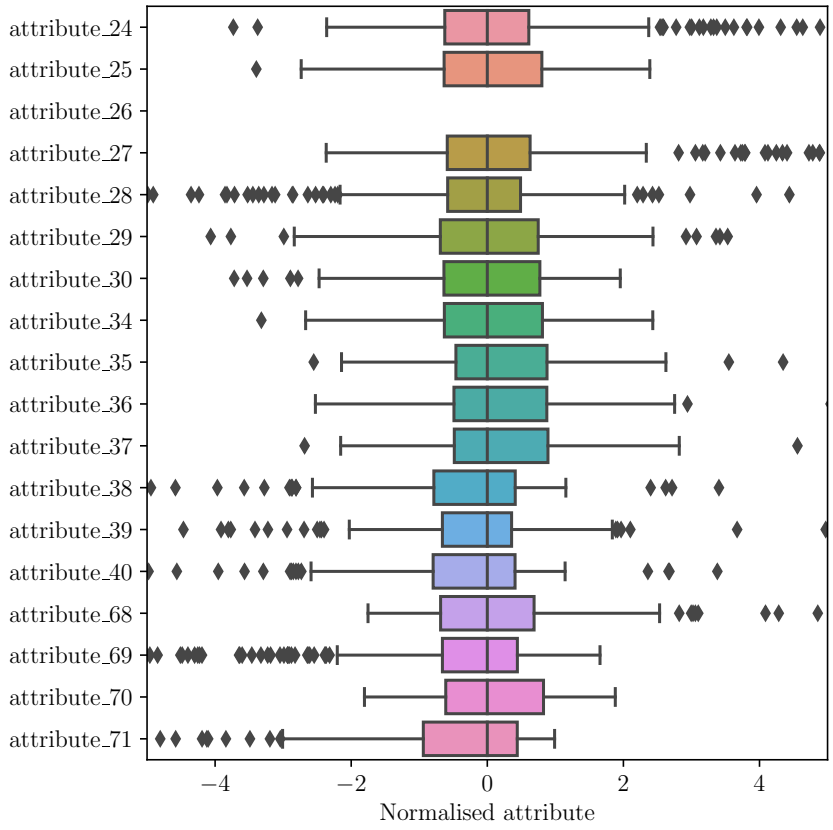

Figure 3. Box-and-whisker plot showing the distribution of attributes calculated for the events detected at Ilulissat, Greenland on 1 January 2018. These attributes are all approximately normally distributed, with the possible exception of the polarity attributes. Attribute 26 has no valid data as it considers a frequency range to close to the sampling rate of the waveform data. Transformations may need to be applied to user-defined attributes to ensure they are normally distributed for use in machine learning applications.

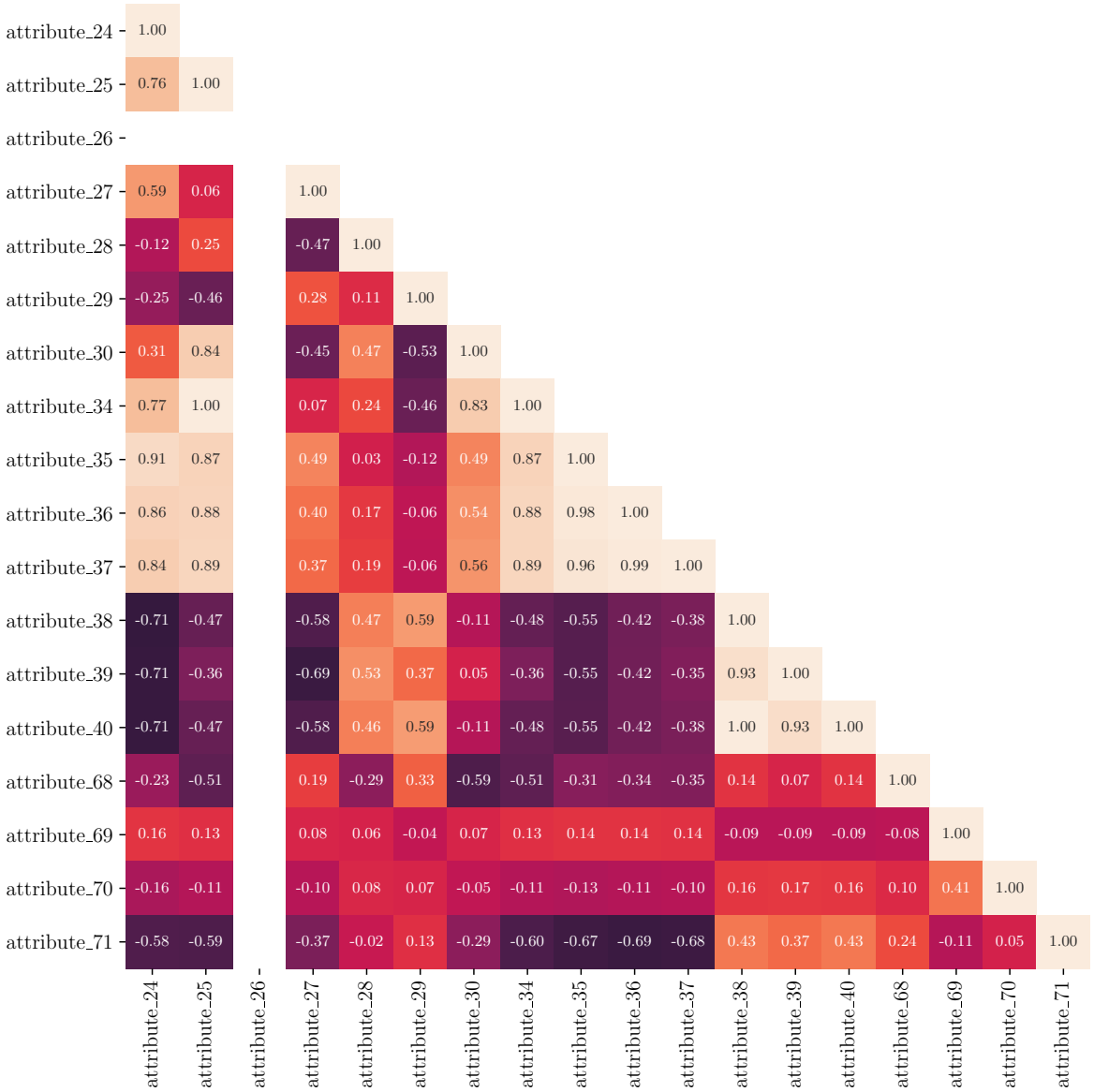

Figure 4. Correlation matrix of the spectral and polarity attributes calculated for the events detected at Ilulissat, Greenland on 1 January 2018. The majority of attributes are not correlated, implying they reveal different information about the seismic waveforms. However, there is a strong correlation between attributes 35, 36 and 37 for the energy in the discrete Fourier transform in various bands above one-quarter of the Nyquist frequency. Attribute 26 has no valid data as it considers a frequency range too close to the sampling rate of the waveform data.