MNRAS **000**, 1–?? (2020) Preprint 17 October 2020 Compiled using MNRAS LATEX style file v3.0

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*

Accepted XXX. Received YYY; in original form ZZZ.

SEISMIC ATTRIBUTES DOCUMENTATION

This document describes the functions in the *seismic attributes* code and provides examples of accepted usage to download waveforms from online repositories, to identify events using one or more seismometers and components, and to derive attributes of the identified signals for use in clustering analyses. The code is written in Python and imports standard packages including *numpy*, *pandas*, *matplotlib* and *seaborn*, and the specialised package *obspy*. Documentation for private functions is not included in this file.

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 seismic waveform data from online respositories and save to a local directory or external drive. The waveform data is 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 seismometer, time period and channels is present in the specified local directory before attempting to download new waveform data. The requested waveform data for a given seismometer, time period and one or more channels is output as a single *obspy* `Stream` object.

Parameters:

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

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

location : *str*
Select a single SEED location identifier; wildcards are not allowed.

channel : *str or list*
Select one or more SEED channel codes. Multiple codes are entered as a list; e.g. ['BHZ', 'HHZ'].

starttime : *UTCDateTime*
Limit results to time series samples on or after the specified start time.

endtime : *UTCDateTime*
Limit results to time series samples on or before the specified end time.

event_buffer : *float, optional*
Specify minimum duration of data to buffer before and after the requested time period; this is used to detect the full length of events that extend beyond the time period. Expected units are seconds.

waveform_name : *str or path, optional*
Specify directory to read and save waveform data. The default location is a folder named *waveforms* in the working directory.

station_name : *str or path, optional*
Specify directory to read and save station data. The default location is a folder named *stations* in the working directory.

client : *str or list, optional*

Specify 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. Missing data is downloaded by default.

Returns: **stream : *Stream***

Return 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', trigger_type='recstalta', thr_event_join=0.5, thr_travel_time=0.01, thr_coincidence_sum=-1, thr_on=5, thr_off=1, **options)

Function to trigger events from seismic waveform data using STA/LTA-type algorithms. The input waveform data is separated 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 with small gaps between triggered events removed from the event catalogue. If multiple seismometers are present the user can specify the minimum number of seismometers in 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 on or after the specified start time.

endtime : *UTCDateTime*

Limit results to time series samples on or before the specified end time.

signal_type : *str, optional*

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

trigger_type : *str, optional*

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

thr_event_join : *float, optional*

Specify the maximum duration of gaps between triggered events before those events are considered separate. Joined events are reported as a single event in the event catalogue.

thr_travel_time : *float, optional*

Specify the amount to artificially lengthen the detected events at each seismometer to handle the signal travel time between the more distant seismometers. The added duration is removed in the coincidence triggering algorithm as simultaneous detections are required at each seismometer. The event times reported in the event catalogue are for a central location in the array.

thr_coincidence_sum : *int, optional*

Specify the number of seismometers which an event must be detected for that event to be included in the event list. By default an event must be detected at every seismometer.

thr_on : *float, optional*

Threshold for switching single seismometer trigger on.

thr_off : *float, optional*

Threshold for switching single seismometer trigger off.

options

Necessary keyword arguments for the respective trigger 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 with sampling rate of trace. (e.g. sta=3, lta=10 would call the trigger with 3 and 10 seconds average, respectively).

Returns: **events : *DataFrame***

Return pandas dataframe of the events with two columns: the first named 'time' for the start time of the event; and the second named 'duration' for the length of the event. The signal type used in the event detection is included as a third column.

plot_events

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

Plot of seismic waveform data 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. Multiple days of data are included on separate pages in a .pdf document.

Parameters: **events : *DataFrame***

List of events stored as a pandas dataframe including a column named 'time' for the start time of the event and a column named 'duration' for the length of the event.

stream : *Stream*

Stream 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 on or after the specified start time.

endtime : *UTCDateTime*

Limit results to time series samples on or before the specified end time.

filename : *str or path, optional*

Specify filename and directory to save .pdf document of plots. The default filename is '[station name]_event_plot.pdf' located in the working directory.

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 events stored as a pandas dataframe including a column named 'time' for the start time of the event and a column named 'duration' for the length of the event.

stream : *Stream*

Stream 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.attribute1, user_defined_attribute, seismic_attributes.attribute4).

Returns: **attributes : *DataFrame***

Return a pandas dataframe with the attributes of each event. The first two columns named 'start_time' and 'end_time' are for the start and end time of the event; the remaining columns are for the requested attributes and are named as specified in their respective attribute function.

plot_attributes

seismic_attributes.plot_attributes(attributes, attribute_scale=None, 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 'start_time' and 'stop_time' for the start and end time of the event.

attribute_scale : *str or list*

String or list of strings specifying which (if any) attributes need to be plotted using a logarithmic scale; permanent changes to scalings are best made in the attribute functions.

filename : *str or path, optional*

Specify filename and directory to save .pdf plot. The default filename is location is the 'attribute_plot.pdf' located in the working directory.

Returns: ***None***

plot_correlations

seismic_attributes.plot_correlations(attributes, attribute_scale=None, filename=None)

Plot of the correlation matrix 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 correlation coefficient is also shown in text.

Parameters: **attributes : *DataFrame***

List of events and their attributes stored as a pandas dataframe including columns named 'start_time' and 'stop_time' for the start and end time of the event.

attribute_scale : *str or list*

String or list of strings specifying which (if any) attributes need to be plotted using a logarithmic scale; permanent changes to scalings are best made in the attribute functions.

filename : *str or path, optional*

Specify filename and directory to save .pdf plot. The default filename is location is the 'correlation_plot.pdf' located in the working directory.

Returns: ***None***

ATTRIBUTE FUNCTIONS

The *seismic attributes* code provides several examples of attribute functions that can be applied directly to waveform data. The general form of the attribute functions is also described to enable the construction of user defined functions.

attribute_1

seismic_attributes.attribute_1(component_stream, event_start, event_stop)

Function to calculate the duration of an event from the waveform data passed on from the `get_attributes` function.

Parameters: **component_stream : *Stream***

Stream 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 on or after the specified start time; the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples on or before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

Return a tuple containing the name of the attribute and its value; i.e. 'attribute_1' as a string and the duration of the event as a float.

attribute_4

seismic_attributes.attribute_4(component_stream, event_start, event_stop)

Function to calculate the ratio between ascending and descending time of an event from the waveform data passed on from the `get_attributes` function.

Parameters: **component_stream : *Stream***

Stream 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 on or after the specified start time; the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples on or before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

Return a tuple containing a the name of the attribute and its value; i.e. 'attribute_4' as a string and the ratio between ascending and descending time of the event as a float.

attribute_10_11_12

seismic_attributes.attribute_10_11_12(component_stream, event_start, event_stop)

Function to calculate the energy in the first third and the remaining part of the autocorrelation function of an event from the waveform data passed on from the `get_attributes` function.

Parameters: **component_stream : *Stream***

Stream 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 on or after the specified start time; the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples on or before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

Return a tuple containing the names of the attributes and their values; i.e. ['attribute_10', 'attribute_11', 'attribute_12'] as a list of strings and the energy in the first third and the remaining part of the autocorrelation function of the event as a list of floats.

attribute_13_14_15_17

seismic_attributes.attribute_13_14_15_17(component_stream, event_start, event_stop)

Function to calculate the energy filtered in 5 – 10 Hz, 10 – 50 Hz, 5 – 70 Hz and 1 – 10 Hz of an event from the waveform data passed on from the `get_attributes` function.

Parameters: **component_stream : *Stream***

Stream 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 on or after the specified start time; the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples on or before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

Return a tuple containing the names of the attributes and their values; i.e. ['attribute_13', 'attribute_14', 'attribute_15', 'attribute_17'] as a list of strings and the energy filtered in 5 – 10 Hz, 10 – 50 Hz, 5 – 70 Hz and 1 – 10 Hz of the event as a list of floats.

attribute_68_69_70_71

seismic_attributes.attribute_68_69_70_71(component_stream, event_start, event_stop)

Function to calculate the rectilinearity, azimuth, dip and planarity of an event from the waveform data passed on from the `get_attributes` function.

Parameters: **component_stream : *Stream***

Stream 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 on or after the specified start time; the stream is already truncated to this range in the get_attributes function.

event_stop : *UTCDateTime*

Limit results to time series samples on or before the specified end time; the stream is already truncated to this range in the get_attributes function.

Returns: **attribute : *tuple***

Return a tuple containing the names of the attributes and their values; i.e. ['attribute_68', 'attribute_69', 'attribute_70', 'attribute_71'] as a list of strings and the rectilinearity, azimuth, dip and planarity of the event as a list of floats.

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].

EXAMPLE USAGE

Download a single day of data in all three components of the broad-band channel at Ilulissat, Greenland from the start of 2018; by default data will be downloaded on the preceeding 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")
```

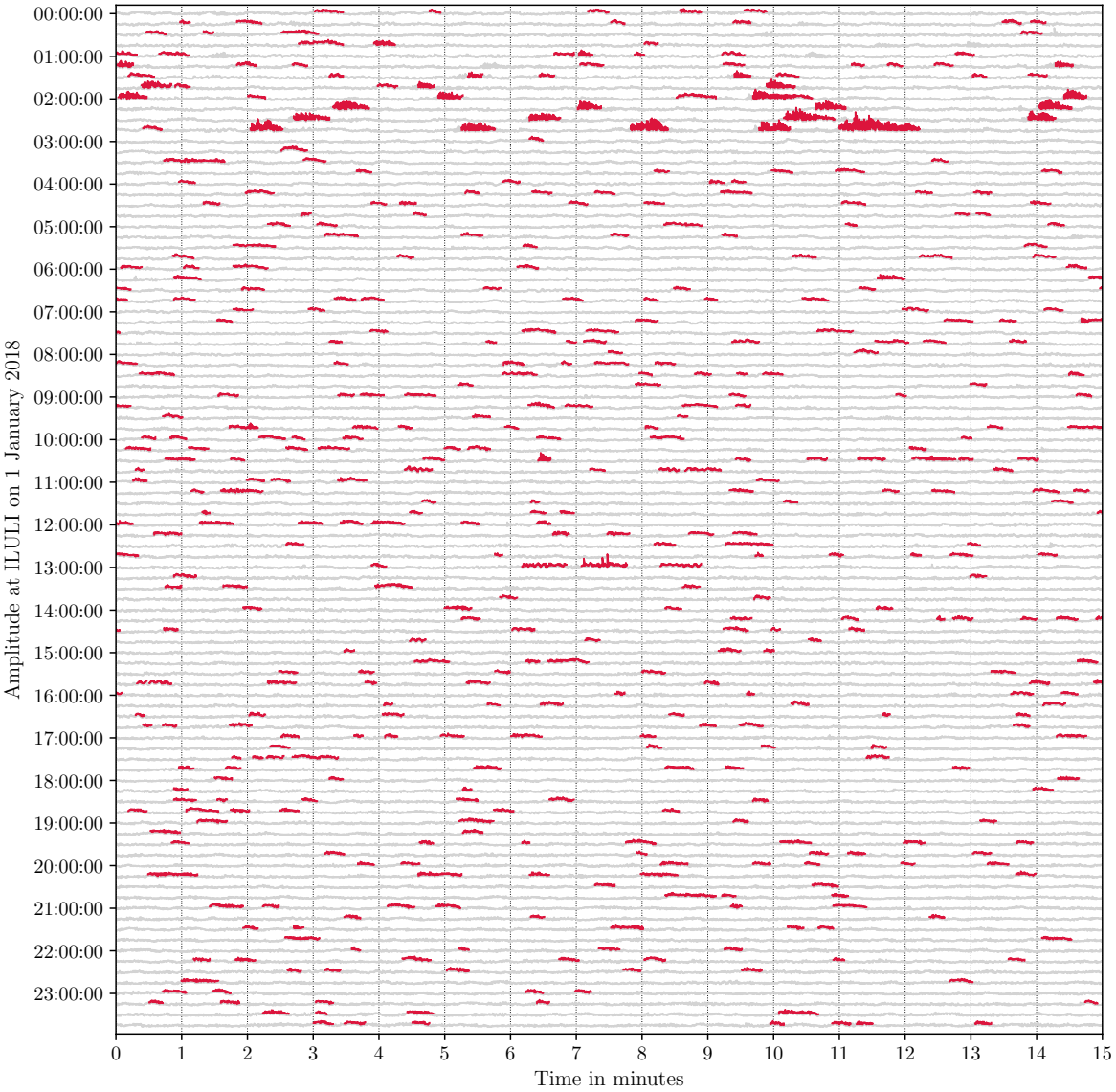

Figure 1. Events detected at Ilulissat, Greenland on 1 January 2018 using the recursive-STA/LTA algorithm.

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

Trigger events using the recursive-STA/LTA algorithm based on the eulclidean norm (amplitude) of the seismic waveform; standard values are chosen for each parameter. Events separated by less than 5 seconds are joined together. The output *pandas* `DataFrame` is written to a .csv file 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_offset=5)
>>> events.to_csv('event_catalogue.csv')
```

Plot waveform data for Ilulissat and highlight identified events. The expected output is shown in Figure 1.

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

Calculate all the pre-defined 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.

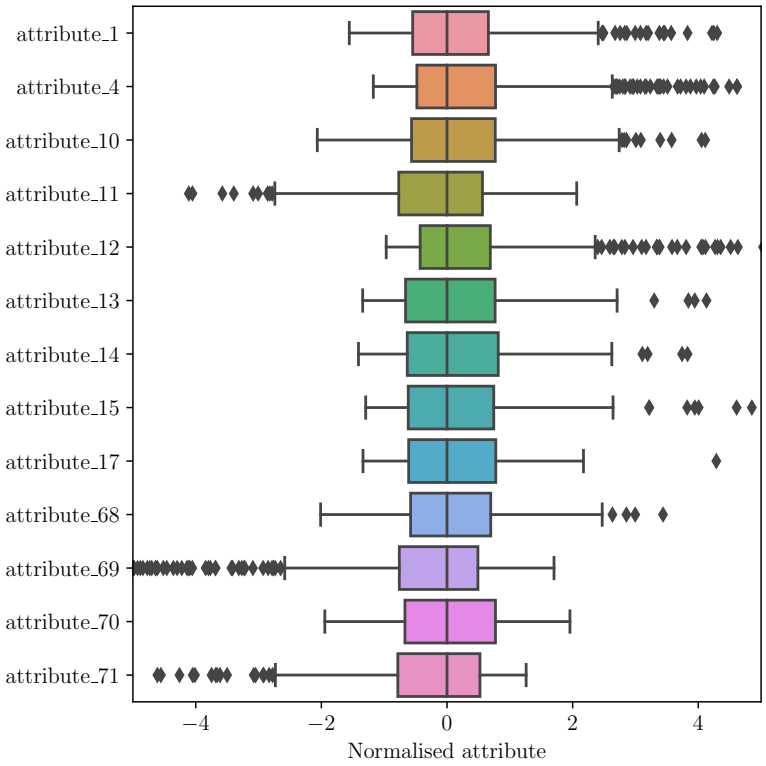

Figure 2. Box-and-whisker plot showing the distribution of attributes calculated for the events detected at Ilulissat, Greenland on 1 January 2018. Some of these attributes need a transformation applied as they do not appear normally distributed in linear space.

```
>>> attributes = sa.get_attributes(events, stream, sa.attribute_1, sa.attribute_4,
        sa.attribute_10_11_12, sa.attribute_13_14_15_17, sa.attribute_68_69_70_71)
>>> attributes.to_csv('attribute_catalogue.csv')
```

Plot 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 2.

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

Plot correlation matrix for the attributes derived for the event catalogue. The expected output is shown in Figure 3.

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

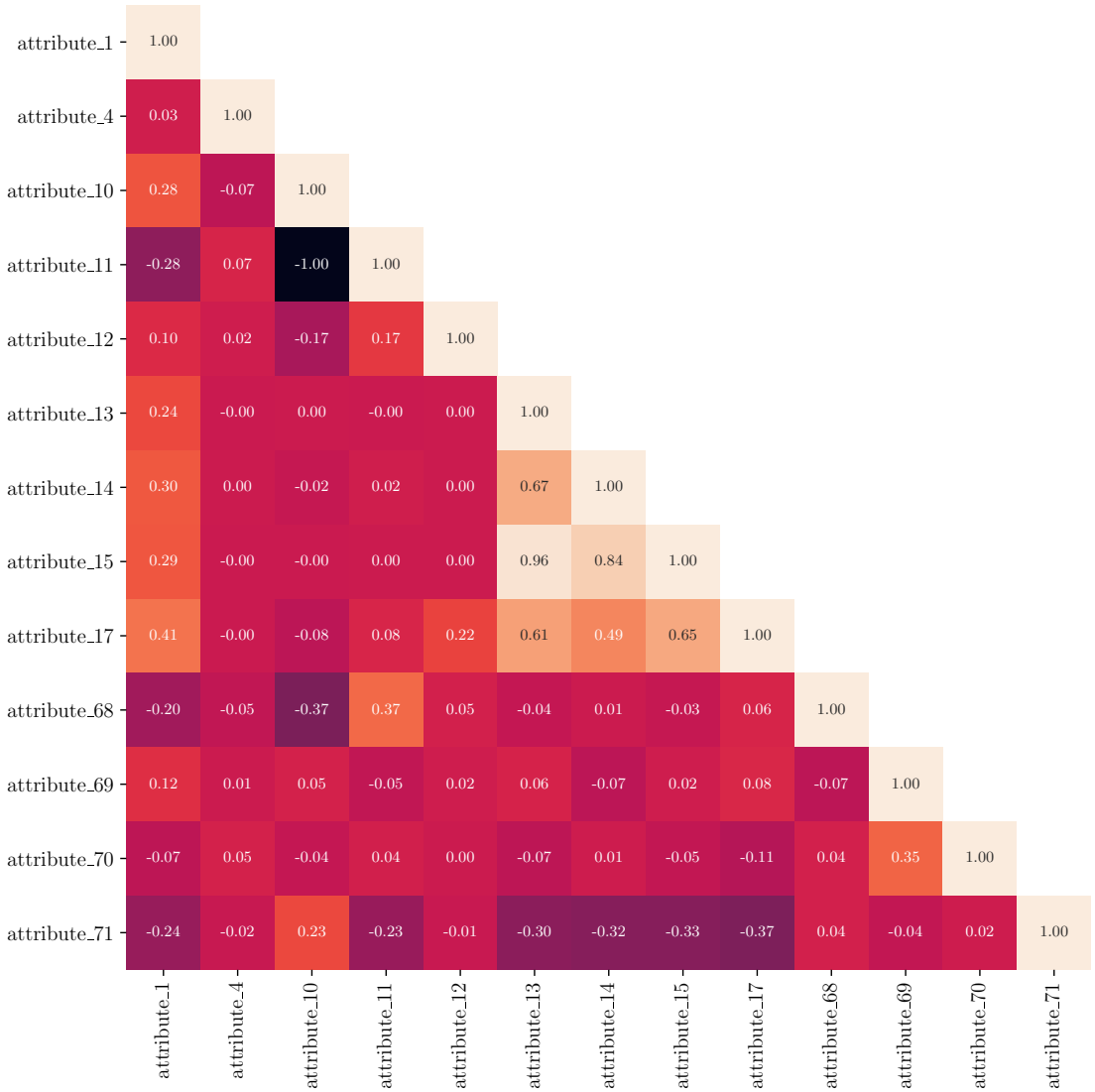

Figure 3. Correlation matrix of the attributes calculated for the events detected at Ilulissat, Greenland on 1 January 2018. The majority of attributes are not correlated, implying they reveal different information about the seismic waveforms. Attributes 10 and 11, and attributes 13 and 15 are notable exceptions.