Introduction: AutoStatsQ – Usage example

This directory contains:

- An example configuration file: `AutoStatsQ_settings.config`
- A file with seismic stations in pyrocko format: `some_stations_testset.pf`
- ...and this step by step instructions.

The two input files (`AutoStatsQ_settings.config, some_stations_testset.pf`) contain all required input to perform a test run of `AutoStatsQ`.

After downloading and installing `AutoStatsQ` (see instructions: https://github.com/gesape/AutoStatsQ), you can copy the two input files to a working directory in which you want to run and save the tests.

Please keep in mind to have some free storage place (<1 GB), because the example waveform data will be downloaded and saved in this working directory.

When you run AutoStatsQ with the proposed settings, please be aware that the **results may look slightly different** compared to the figures shown here, because the fdsn servers might not always return the same amount of data and meta data can be updated from time to time.

This tutorial is work in progress.
I am happy to receive your comments, questions, suggestions and corrections...

Table of Contents

Introduction: AutoStatsQ – Usage example....1
1. The Configuration file:....2
1.1. General settings:....2
1.2. Catalog:....2
1.3. Arrival time computation:....4
1.4. Waveform data and station meta data:....4
1.5. Preprocessing (Downsampling, Restitution, Rotation):....5
1.6. Synthetic data:....5
1.7. Quality checks:....6
1.7.1 Amplitude gain test:....6
1.7.2 PSD test:....8
1.7.3 Orientation test:....9
1.7.4 Timing test:....10
1.7.5 TeleCheck:....10
1.8. Settings for plotting of maps:....11
2. Run AutoStatsQ step-by-step:....11
2.1 Step 1: Catalog....11
2.2 Step 2: Arrival times....12
2.3 Step 3: Data & Metadata....13
2.4 & 2.5 Step 4 & 5: Preprocessing and synthetic data:....13
2.6 Step 6: Running the tests:....14

2.6.1 Step 6.1: The amplitude gain test....14
2.6.2 Step 6.2: The PSD test....17
2.6.3 Step 6.3: The orientation test....18
3. Typically observed data and metadata error gallery....20
3.1 Gains & PSDs....20
3.1.2 Large offsets of PSDs....20
3.2 Sensor orientation....21
3.2.1 Example: Inverse polarities of horizontal components....21
3.2.2 Example: Temporal change of orientation:....21
4. AutoStatsQ commands overview:....22
5. FAQs:....23
References:....23

1. The Configuration file:

The config file is used to provide all needed information to AutoStatsQ, including for example the stations which we want to check, the time range, if we want to download data or check local data and which tests to run.

In the example config file, many options are set to `false` to avoid running the entire program in a single run. After the explaination of all config file settings, you will find a section where running AutoStatsQ is described – including setting these parameters to `true` .

1.1. General settings:

Every file starts with a section for the general settings, which are the working directory path and the path to the list of stations which we want to check:

```
--- !autostatsq.config.AutoStatsQConfig
Settings:
- !autostatsq.config.GeneralSettings
  work_dir: '.'
  list_station_lists: [some_stations_testset.pf]
```

In case of this example, the working directory is the directory in which the config file is located. And the list of stations is provided in the *some_stations_testset.pf* file.

1.2. Catalog:

The next part of the config file contains the information on the earthquakes which are used to run the tests. AutoStatsQ uses subsets of deep and shallow events for the different tests which rely either on body waves or on surface waves.

To run this example, you don't have to change anything in the provided config file.
# indicates comments in the following blocks:

```
- !autostatsq.config.CatalogConfig
  search_events: false
  # search gCMT catalog for events? We want to search for events, because
      we do not have a catalog of earthquakes yet.

  use_local_catalog: false
  # If we would have a catalog, we could add the path here.

  subset_of_local_catalog: true
  # This will generate the deep and shallow subset of the catalog which we
      will download.

  use_local_subsets: false
  # If we have already subsets of some catalog downloaded, we would
      indicate "true" here and set the path in the next option
      "subset_fns".

  subset_fns: {}
  # e.g.
  # {'deep': 'catalog_deep_subset.txt',
  # 'shallow': 'catalog_shallow_subset.txt'}
  # In our example, we download a new catalog, so we keep this empty.

  # Settings for searching GCMT for teleseismic events:
  min_mag: 7.0  # should be large enough to observe in far distance
  max_mag: 8.0  # should not be too large because of point source
                  approximation in forward modeling of synthetic
                  waveforms.
  tmin_str: '2015-01-01 00:00:00'
  tmax_str: '2021-12-01 00:00:00'
      # tmin and tmax must be defined according to the run time or the
            time of interest of your stations.
  min_dist_km: 4000.0  # should be >> than network aperture
  max_dist_km: 9999999.0
  depth_options:
    deep: [25000, 600000] # [m]
    shallow: [100, 40000] # [m]


  wedges_width: 10
  # backazimuthal step for subset generation
  # adjust to get more events, especially if time range is small

  mid_point: [46.98, 10.74]
  # Give a rough estimate of the midpoint of your array or network.
  # This is optional, if the midpoint is not provided a geographic
      midpoint is calculated of all station locations is computed.
      This midpoint is used to compute the distance to the teleseismic
      events.

  ### Catalog plotting options ###
  plot_catalog_all: false
  # Plots the entire downloaded catalog on a map.

  plot_hist_wedges: false
  # Plot statistics of the catalog.

  plot_catalog_subset: false
  # Plot the deep and shallow selected subsets on maps.
```

1.3. Arrival time computation:

The next section of the config file defines the computation of the arrival times of body and surface waves. By default, the very first arriving P phase is used and the arrival time of a Rayleigh wave with v=4.0 km/s. However, both can be adjusted for example when only events in a certain distance range can be used. The P phase can be defined using the *phase_select* with pyrocko cake terminology. [Type “cake list-phase-map” into your terminal to get a list of phase names mapped to cake terminology.]
The Rayleigh wave arrival can be adjusted by modifying the v_rayleigh parameter. In case of using less distant earthquakes, it the fastest Rayleigh wave velocity may be lower than the default setting.

```
      - !autostatsq.config.ArrTConfig
        calc_first_arr_t: false
        phase_select: P|p|P(cmb)P(icb)P(icb)p(cmb)p|P(cmb)Pv(icb)p(cmb)p|
P(cmb)P<(icb)(cmb)p
        calc_est_R: false
        v_rayleigh: 4.0
```

1.4. Waveform data and station meta data:

The next step contains the settings to download data and metadata – or to use local data. In case of our example, we download data and station meta data from the fdsn server of BGR:

```
      - !autostatsq.config.MetaDataDownloadConfig
        download_data: false
        # set to true to download data
        download_metadata: false
        # set to true to download metadata

        local_data: []
        # Lists of paths to include local data.
        local_metadata: []
        # Lists of paths to include local station meta data.
        local_waveforms_only: false
        # Set to true to use only local waveforms, do not download data.
        sds_structure: false
        # Set to true if you have local waveform data which is saved in sds
            structure directories.

        use_downmeta: true
        # Use the downloaded metadata, can just stay true at all times.

        channels_download: HH*
        # Define channels to download. * is wildcard.
        all_channels: false
        # Or indicate to download all channels of a station (not recommended,
            can be time consuming).
        sites: [bgr]
        # Define fdsn server from which data is downloaded. Available are:
            geofon, iris, orfeus, bgr, geonet, knmi, ncedc, scedc, usgs, bgr,
            koeri, ethz, icgc, ipgp, ingv, isc, lmu, noa, resif, usp
        token: {}
```

```
# If data is restricted, you can add a token here, the format is:
    {'geofon': '/path/to/token'}

dt_start: 0.1
# Start of download before origin time in hours.
dt_end: 1.5
# End of download after origin time in hours.
# The time window should be long enough to contain body and surface wave
    phases and to allow bandpass filtering in low frequency ranges.
```

1.5. Preprocessing (Downsampling, Restitution, Rotation):

In order to run the tests, AutoStatsQ needs restituted data, which means the instrument response of the seismometer is removed from the waveform signal. In addition, the traces are rotated into Z,R,T coordinate system, taking into account the channel information of the response file. Finally, the waveform data is downsampled, in oder to reduce computation time. If a test relying on the comparison of synthetic and recorded data is run, then the downsampled rate needs to be the same as the sampling rate of the Green's function database (see step 6).

```
- !autostatsq.config.RestDownRotConfig
  rest_data: false
  # set to true to start restitution
  freqlim: [0.005, 0.01, 0.2, 0.25]
  # frequency filter for restitution; first two entries give lower corner
      frequency, last two entries are the upper corner frequencies
  rotate_data: false
  # set to true to rotate to ZRT system
  deltat_down: 2.0
  # Dampling rate after downsampling, in [s].
```

1.6. Synthetic data:

The next section of the config file defines if and how synthetic data should be computed. The computation is based on the pyrocko precalculated Green's function databases, see also https://pyrocko.org/docs/current/apps/fomosto/. Default is to use the global_2s store, which is can be downloaded using pyrocko's `fomosto: fomosto download kinherd global_2s` (https://greens-mill.pyrocko.org/).

Synthetic data is used for the PSD test, where spectra of synthetic and recorded data are observed. In addition, synthetic data can be used in the gain test, for a comparison of P wave amplitudes. However, the amplitude gain test can also be run relative to a reference station. **Synthetic data only needs to be computed if one of the aforementioned tests shall be performed. Otherwise, the Green's function database does not need to be downloaded.**

In this example, you can decide if you want to run the PSD test. If so, please download the GF database. This might take some time depending on your internet connection. It is also possible to skip this step (and then also skip the PSD test).

```
- !autostatsq.config.SynthDataConfig
  make_syn_data: false
  # start computation of synthetic waveform data
```

```
engine_path:  path/to/gf_stores
# path to GF store directory
store_id:  global_2s
# Name of the store which you want to used.
```

1.7. Quality checks:

AutoStatsQ currently contains three well tested quality checks and two which are still in development. In this example, we will only run the three stable checks on P wave amplitude gains, PSDs and sensor orientations.

For methodological descriptions, explanations and discussions, please refer to Petersen et al., 2019. Here we only explain how to run the tests step by step.

1.7.1 Amplitude gain test:

The first test section contains the settings for testing the amplitude gains. Several methods are implemented. Most often, maximum P wave amplitudes are compared either to a reference station, or to the amplitudes of synthetic data. **The used phase is defined by the phase settings used to compute the arrival time (see 3. Arrival time computation).**
In case of both, the comparison to a reference station and to synthetic data, the median result of all azimuthally distributed events should be considered to avoid bias from single noisy events or travel-path effects. Smaller amplitude differences can result from site-effects, while larger amplitude errors are more likely to result from instrument or meta data errors.

```
- !autostatsq.config.GainfactorsConfig
  calc_gainfactors: false
  # start running the gain test

  gain_factor_method:
  - reference_nsl
  - [GR, BFO]
  # Sets the gain factor method. In the case of this example, the P
  # amplitudes are compared to the reference station GR.BFO. See below
  # this config section for all methods & recommendations.

  fband:
    corner_hp: 0.01
    corner_lp: 0.2
    order: 4
  # Settings for the bandpass filter, filtering is performed before #
  # cutting the time windows.

  wdw_st_arr: 5
  # Start of time window before phase arrival time.
  wdw_sp_arr: 120
  # End of time window after phase arrival time
  taper_xfrac: 0.25
  # taper of time window: fade in and fade out as fraction between 0. and
  # 1. of pyrocko.trace CosFader – can remain at default.

  snr_thresh: 1.0
  # SNR threshold, sometimes a bit buggy.
```

```
debug_mode: false
# Start an interactive debugging mode. This will open the used waveform
# data with time window and filter settings as selected above. Helpful
# to debug in case of suspicious results – but should be turned off to
# run entire test, because it slows down the process significantly.

phase_select: first(P|p|P(cmb)P(icb)P(icb)p(cmb)p|P(cmb)Pv(icb)p(cmb)p|P(cmb)P<(icb)(cmb)p)
# selected phase for the test, default is first arriving P phase.

components: [Z]
# component(s) to run the test on

plot_median_gain_on_map: false
# plot the gain test result on a map
plot_allgains: false
# plot single gain test results for each station & event
# (simple x-y-plot)
```

Gain check methods:

(a) relative to a reference station: ← **We use this method in the example.**

```
gain_factor_method:
- reference_nsl
- [NET, STAT]
```

→ Should only be used if the distance of the network stations to the reference stations is small compared to the event distance. We are using a teleseismic catalog with a large minimum distance, therefore it is safe to assume that when neglecting site effects, the amplitudes of an event should be similar at the different stations.

→ In case of a short operation time period the number of large, teleseismic events might not be sufficient to run this test. Then the amplitude gain test should rather be based on closer, moderate events and be compared to synthetic amplitudes.

→ Can also be used to study site effects on stations.

(b) compared to synthetic data:

```
gain_factor_method: ['syn_comp']
```

→ Compares the amplitudes of each event at all stations to the amplitudes of the synthetic waveforms.
→ Can also be used when using closer, moderate earthquakes.

(c) compared to median of all events:

```
gain_factor_method: ['median_all_avail']
```

→ Compares the amplitudes of each event at all stations to median of all stations.

1.7.2 PSD test:

Power spectral density plots (PSDs) are computed for synthetic and observed data of all (deep) events and stations. PSDs of synthetic and observed waveforms are calculated for all station–component–event combinations in long time windows (>30 min). For each pair of synthetic and observed PSDs, the ratio of both PSDs is calculated. Subsequently, the median power ratio over all events is computed for each frequency band of each station and component.

Frequency ranges with a stable PSD ratio are determined by line fits to a given number of frequency ratio points (default *n_poly: 25*). Successive lines having a slope below a given threshold (default *norm_factor: 10*) set up the recommended frequency ranges. We recommend testing different values for these parameters. Higher thresholds and fitting more frequency ratio points can be used to search for large misfits due to erroneous amplitude gains or extremely noisy environments, as well as to verify the instrument corner frequency.

```
- !autostatsq.config.PSDConfig
  calc_psd: false
  # Start running the PSD check

  ### Complete time window:
  dt_start: 60
  # Start of time window in [s] before first arrival time.
  dt_end: 120
  # End of time window in [s] before arrival of Rayleigh waves.


  ### Increments of time window to compute PSD:
  tinc: 600
  # Time increment (window shift time) to compute PSD.
  tpad: 200
  # Padding time appended on either side of the data windows.


  ### Line fitting parameters to determine frequency ranges with flat
  ### ratios between spectra of synthetic and observed data:
  n_poly: 25
  # number of points for line fit (see text above)
  norm_factor: 50
  # Line slope threshold (see text above)
  f_ign: 0.02
  # Frequency bandwidth to ignore in case of outliers/ instabilities of
  # ratio between observed and synthetic spectras.
  only_first: true
  # Output only the first, lowest frequency range with a stable ratio
  # between observed and synthetic data?


  ### Plotting:
  plot_psds: false
  # Plot PSDs of synthetic and observed data for each event and station.
  plot_ratio_extra: false
  # Plot ratio of syn and obs PSDs for single events – only needed for
  #  debugging.
  plot_m_rat: false
  # Plot median ratio between syn and obs PSDs over frequency – only
needed for    #  debugging
  plot_flat_ranges: false
  # Output plot showing median ratio of PSDs and the frequency range(s)
  # for which this ratio is stable.
```

```
        plt_neigh_ranges: false
        # remove, outdated
```

1.7.3 Orientation test:

The next part of the config files contains the settings for the orientation test.

Rayleigh waves have a 90° phase shift between the vertical and the radial component. Waveforms of a set of shallow, teleseismic events are analyzed by rotating the radial traces in steps of 1°. For each step, the cross-correlation of the Hilbert-transformed Z component and the rotated R component is calculated. Correctly oriented sensors would result in a maximum cross-correlation value at 0° rotation.
For example, a N component pointing towards N20E would result in a correction angle of -20°. The median of the correction angles of all events are used are the correction angle for one station. We recommend using the median instead of the mean of all events as the correction angle for a station, because the results of single events in one azimuthal direction may be biased by crustal structures. See Petersen et al. 2019 for more information.

```
      - !autostatsq.config.OrientConfig
        orient_rayl: false
        # Start orientation test

        bandpass: [3, 0.01, 0.05]
        # Bandpass filter [steepness, lower corner frequency, higher corner
        # frequency], should be chosen to emphasize Rayleigh waves.

        start_before_ev: 30
        stop_after_ev: 600
        # Settings of time window, start before and end after theoretical
        # Rayleigh wave arrival.

        ccmin: 0.80
        # Threshold of cross-correlation value. If maximum cc value of an event
        # at a station is below this threshold, then this event will be excluded
        # from the analysis. Should be at least 0.8, but even higher values
        # provide more stable results.

        plot_heatmap: false
        # Plot heatmap showing cc value over correction angle for each station
        # and event. Usually the next, distribution plot is easier to read.
        plot_distr: false
        # Plot max. cross-correlation value over correction angle for each
        # station and event.
        plot_orient_map_fromfile: false
        # Plot map with horizontal sensor orientation as arrows.
        plot_angles_vs_events: false
        # Plot correction angle vs. event time for each station.
        plot_angles_vs_baz: false
        # Plot correction angle vs. backazimuth to event for each station.


        debug_mode: false
        # run test in debug mode, will open waveform data in used time window
        # and with filter in interactive waveform browser snuffler. Only for
        # debugging, otherwise slow.
```

1.7.4 Timing test:

The small implemented check for timing errors is based on the cross-correlation between the recorded traces and the synthetic vertical traces:
(1) First, for each event and station the two (syn. + obs.) traces are correlated to obtain the time shift for which the correlation is highest.
(2) In a second step the median time shift of each event over all stations is determined and the time shift values at the single stations are corrected for this median value. This is done to avoid errors from wrong origin times in the catalog, to take into account large deviations between origin time and centroid time, and to consider large path effects of the teleseismic test events which effect all stations in a similar manner.

The test is run in low frequency ranges (e.g. 0.01-0.10 Hz) and using synthetics computed from a global GFDB with a sampling of 2s. Therefore this test can only be applied to detect large timing errors in the order of several seconds. This error range is only useful to check prior to other seismological applications which use a similar frequency range as e.g. MT inversions.

```
- !autostatsq.config.TimingConfig
  timing_test: false
  # Start timing test.

  bandpass: [3, 0.01, 0.1]
  # bandpass filter for timing test,
  # [steepness, lower corner frequency, higher corner frequency]

  time_wdw: [firstP, 1200]
  # Time window – from first P phase until 1200 s after that.

  cc_thresh: 0.6
  # Waveform cross-correlation threshold to consider event at a station.
  search_locations: false
  # search for stations with available waveform data for this test –
  # default is false, then the test tries to use all stations.
  RENAME this!!!

  debug_mode: false
  # Interactive debugging mode.
```

1.7.5 TeleCheck:

This is a new, not well checked interactive test, which will not be covered in this manual. Simply leave tele_check: false.

```
- !autostatsq.config.TeleCheckConfig
  tele_check: false
```

1.8. Settings for plotting of maps:

In the last section of the config file, the maps to show the results of the orientation and gain tests can be adjusted:

```
- !autostatsq.config.maps
  map_size: [30.0, 30.0]
  pl_opt: ['automatic']  # [50, 8.3, 1000000, 'split']
  # [Center of map latitude, longitude, radius, gmt color scale name] or
  # [‘automatic’] to determine from data.
  pl_topo: false
  # If set to true, topographic data from ETOPO or SRTM will be downloaded
  # and used. This may slow down the plotting.
  outformat: png
  # Output format of the map (png or pdf).
```

2. Run AutoStatsQ step-by-step:

2.1 Step 1: Catalog

We will now start with a very first run of AutoStatsQ. As a first step, we will only download the **catalog**. Therefore we set

```
search_events: true
```

and to also have a look at the output plots:

```
  plot_catalog_all: true
  plot_hist_wedges: true
  plot_catalog_subset: true
```

`We then run AutoStatsQ in the terminal:`

```
autostatsq --config AutoStatsQ_settings.config --run -l INFO
```

The **`-l info`** defines the log level, the amount of terminal output while running AutoStatsQ.

Running this command, a new directory “*results*” is generated in your work directory. This directory contains a subdirectory “*catalog*”, in which the catalogs and catalog plots are stored.

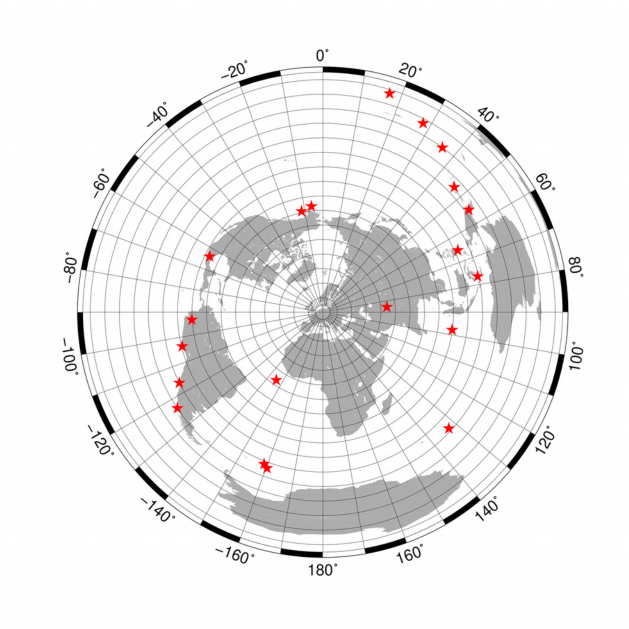

Fig. 1: Resulting catalog of azimuthally distributed deep events.

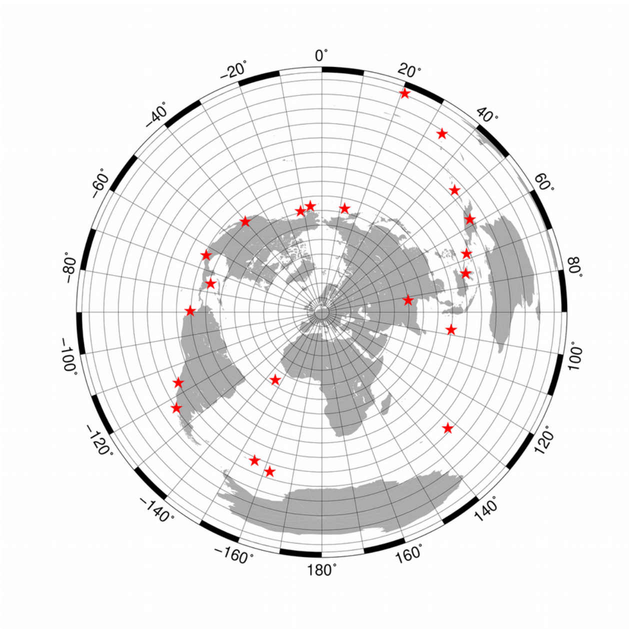

Fig. 2: Resulting catalog of azimuthally distributed shallow events.

The two catalog files for the deep and shallow subsets can now be used to move on to the next steps of AutoStatsQ. We therefore include them into the config file:

```
subset_fns: {'deep': 'results/catalog/catalog_Mgr7.0_deep.txt',
             'shallow': 'results/catalog/catalog_Mgr7.0_shallow.txt'}
use_local_subsets: true
```

In addition, we set:

```
search_events: false
plot_catalog_all: false
plot_hist_wedges: false
plot_catalog_subset: false
```

2.2 Step 2: Arrival times

We now compute the **arrival times** of P and Rayleigh waves by setting:

```
calc_first_arr_t: true
calc_est_R: true
```

..and we rerun AutoStatsQ:

```
autostatsq --config AutoStatsQ_settings.config --run -l INFO
```

The calculation of arrival times may take few minutes. The arrival times are saved as numpy arrays in the /ttt directory. Next we set:

```
calc_first_arr_t: false
calc_est_R: false
```

…so that we do not rerun the calculation.

2.3 Step 3: Data & Metadata

And we continue with the data and metadata section. In the example, we want to download both, waveform data and station meta data from the fdsn server of BGR. Therefore we set to true:

```
download_data: true
download_metadata: true
```

Then we re-run AutoStatsQ. The download of data can take few minutes to few hours, depending on the amount of data, the internet connection and also the responding fdsn server. Sometimes, due to errors servers return not any data…

```
autostatsq --config AutoStatsQ_settings.config --run -l INFO
```

We set back **`download_data: false`** and **`download_metadata: false`**.

2.4 & 2.5 Step 4 & 5: Preprocessing and synthetic data:

Now we <u>preprocess the data and compute the synthetic data</u> as a last step before we actually start running the tests:

```
rest_data: true
rotate_data: true

make_syn_data: true
engine_path:  path/to/gf_stores
```

The engine path should point to the directory where you have stored the downloaded GF database.

After running AutoStatsQ again (**`autostatsq --config AutoStatsQ_settings.config --run -l INFO)`**, three new directories exist in your working directory:

- `/rest`: Contains the restituted data (instrument response removed) – still with original sampling rate and component orientation.
- `/rrd`: Contains the restituted and downsampled data, rotated to ZRT.
- `/synthetics`: Contains the computed synthetic data, sampling frequency as in `/rrd` directory.

After checking that these new directories exist, we reset the above parameters in the config file to `false`:

```
rest_data: false
rotate_data: false

make_syn_data: false
```

→ → → **Now, all data is prepared to run the quality checks.**

2.6 Step 6: Running the tests:

Each test section (gain, PSD, orient, timing) has one parameter in the config file which indicates that the test should be run. These are:

- **`calc_gainfactors`**
- **`calc_psd`**
- **`orient_rayl`**
- **`timing_test`**

By setting them to **`true`**, the test will be started.

2.6.1 Step 6.1: The amplitude gain test

We will run one test after another, starting with the amplitude gain check. We will run the test to check the vertical component amplitudes relative to the reference station GR.BFO. To start the test, we set the according parameters to true:

```
calc_gainfactors: true
plot_median_gain_on_map: true
plot_allgains: true
```

...and rerun AutoStatsQ:

```
autostatsq --config AutoStatsQ_settings.config --run -l INFO
```

The results are stored in the directory **`/results/gains/`**. Two output files are generated:

- `gains_median_and_meanZ.txt`: A file with the median, mean, standard deviation and number of used events for each station. Can be opened in any editor.
- `gains_all_eventsZ.txt`: A comma spread (csv) table containing the single results for each station and event. Best to open with a spreadsheet application.

Looking at the median gain rations of the different stations, we see that the amplitudes are in the same order of magnitude as BFO at most stations.

The results are also shown in two figures:

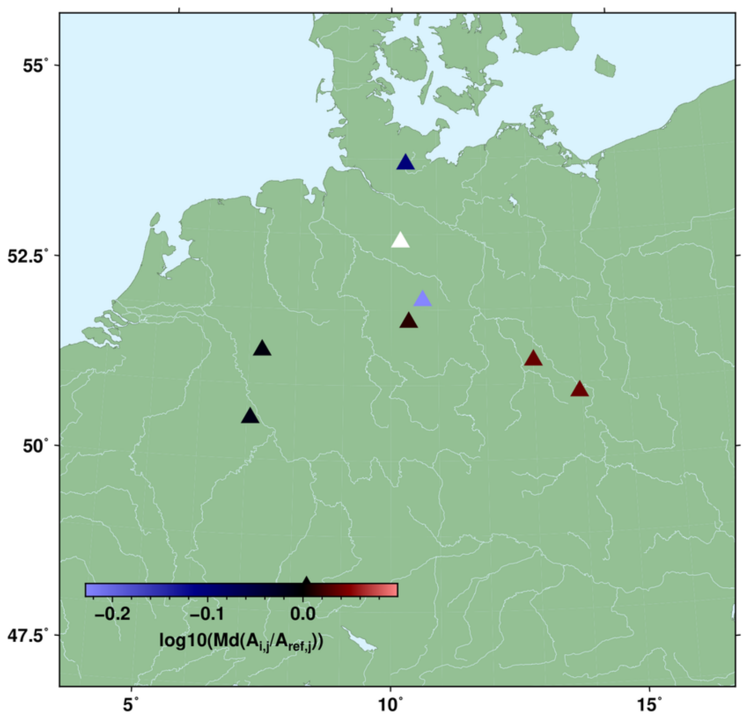

Fig. 3: Map showing the median amplitude ratio of all events relative to reference station BFO.

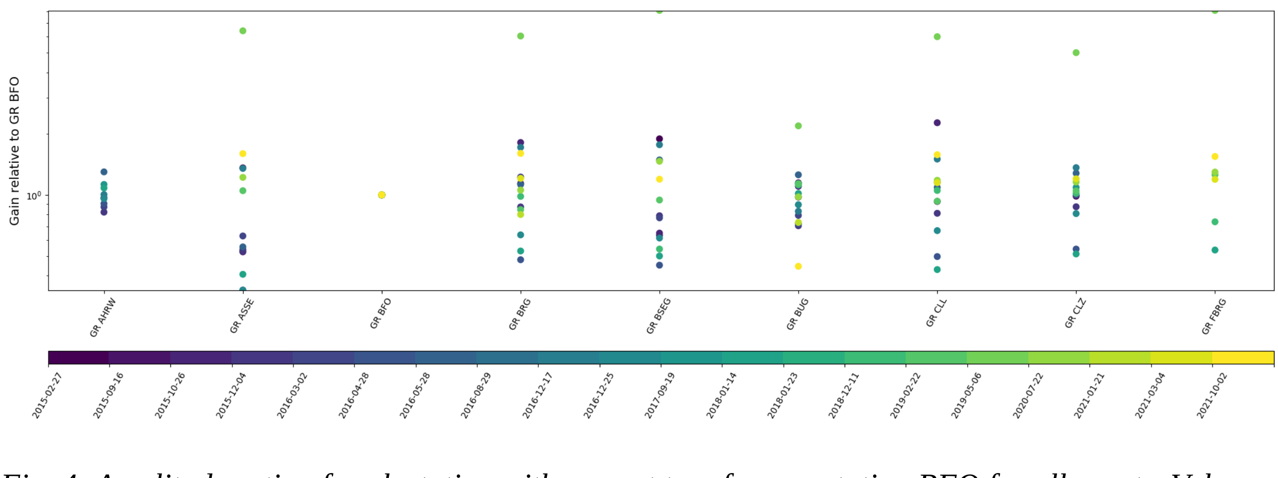

Fig. 4: Amplitude ratio of each station with respect to reference station BFO for all events. Values around 10°=1 indicate very similar and therefore non-suspicious values.

2.6.2 Step 6.2: The PSD test

To start the test, we set the gain parameters back to false and the orientation test parameters to true:

```
calc_gainfactors: false
plot_median_gain_on_map: false
plot_allgains: false

calc_psd: true
plot_psds: true
plot_flat_ranges: true
```

...and rerun AutoStatsQ:

`autostatsq --config AutoStatsQ_settings.config --run -l INFO`

AutoStatsQ now computes amplitude spectra for the observed and synthetic waveforms of all events and stations. Then, for each station-event-pair, the ratio of observed and synthetic spectra is computed in overlapping frequency bands. A perfect match of synthetics and observed data would result in a ratio of 1 for all frequency bands. Large deviations between observed and synthetic data can be attributed to various reasons:

- Errors in the transfer function of the seismometer → These can lead to amplitude offsets.
- Below the lower corner frequency of the instrument, the amplitudes of the obs. Spectra are generally lower than of the synthetic data.
- Similar to the P phase gain test, site effects from the subsurface at the station can lead to frequency-dependent deviations of the amplitudes and hence the spectra.

The results of the PSD check are saved in the directory **`/results/freq`**. This directory contains:

- `*NET*.*STA*__*comp*.png`: Synth. and observed spectra of all events for a visual check.
- `*NET*.*STA*__*comp*_flatrange*n_poly*_*norm_factor*.png`: Plot showing the median PSD ratio and the frequency range for which this ratio is considered to be "flat" (stable).
- `psd_flat_ratio_linefit2.yaml`: File containing the frequency ranges of each station for which a stable ratio between observed and synthetic data is found and the median ratio itself.

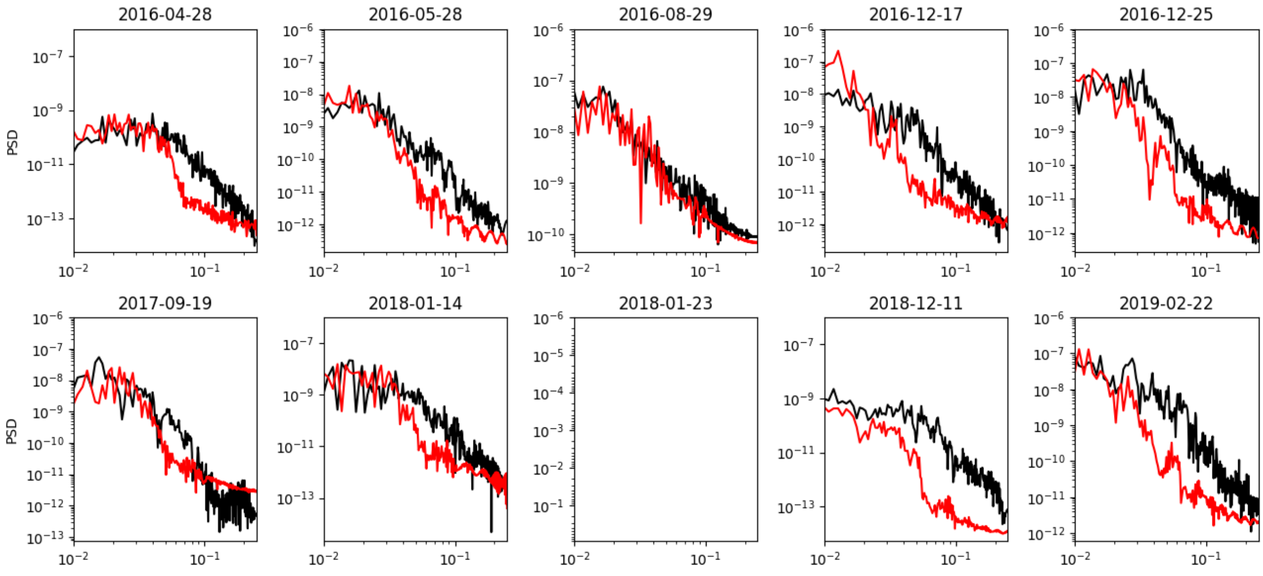

Fig. 5: Examples of synthetic (red) and observed (black) PSDs of station GR.CLZ.

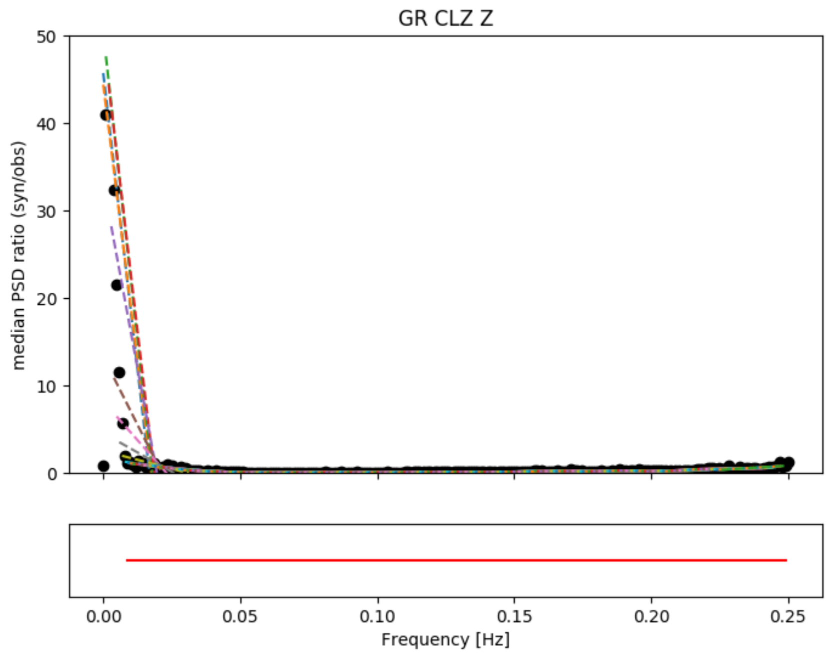

Fig. 6: (upper) Median PSD ratio for different frequency bands and line fits through the median values. The slope of the line fits are evaluated to determine the "flat" frequency range with a stable PSD ratio (lower plot).

2.6.3 Step 6.3: The orientation test

To start the test, we set the PSD parameters back to false and the orientation test parameters to true:

```
orient_rayl: true
plot_distr: true
plot_orient_map_fromfile: true
plot_angles_vs_events: true
```

...and rerun AutoStatsQ:

```
autostatsq --config AutoStatsQ_settings.config --run -l INFO
```

The results of the tests are stored in the directory /results/orient, which contains different files:

- AllCorrectionAngles.yaml: Correction angles of all events obtained for every station.
- CorrectionAngles.yaml: Median, mean, standard deviation and number of events used for each station.
- *NET*_*STA*__distr.png: One figure for each station with subplots for each event showing the cross-correlation coefficient over correction angle for each event. All events with waveform data shown, even if below cross-correlation threshold.
- *NET*_*STA*_overtime.png: Correction angle for each event over event time. Can help to identify temporal changes e.g. due to a misorientation after swop of instruments.

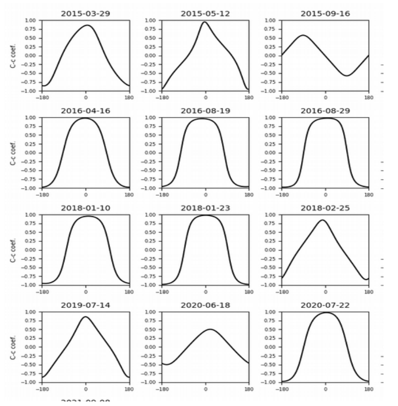

Fig. 7: Example results of GR.CLL of 15 earthquakes. Cross-correlation coefficient of Z component and Hilbert transformed R component over applied correction angle. A maximum at 0° indicates a station with perfectly oriented horizontal components.*

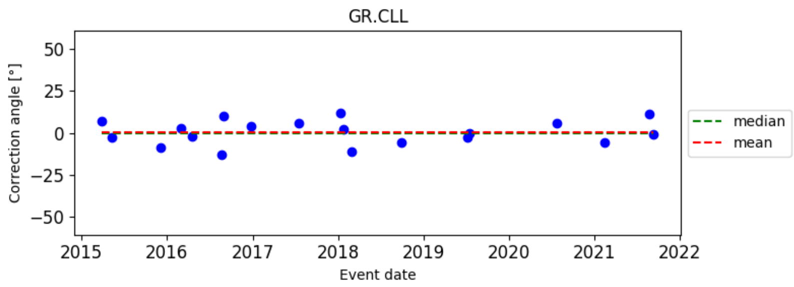

Fig. 8: Obtained correction angle for each event, station GR.CLL.We observe some scatter, but no temporal trend.

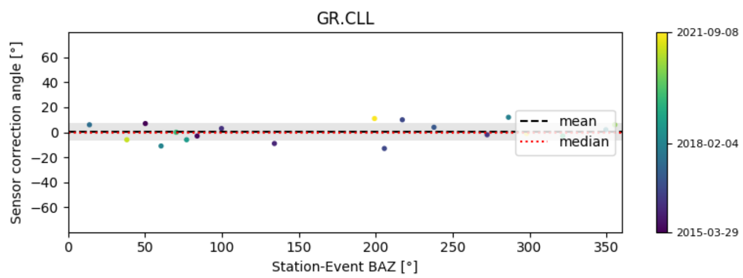

Fig. 9: Obtained correction angle for each event, sorted by back azimuth to station. We observe some scatter, but no clear directional trend. Grey area indicates the standard deviation around the mean (black line). Median as dotted red line.

The three result plots show a median and mean correction angle around 0° with some scatter. Scatter of the single event results can have several reasons:

- Variations in the noise level → We have chosen a cross-correlation threshold of 0.8. By increasing the threshold or manually checking the waveforms, it is possible to exclude the events for which a station has a decreased similarity of Hilbert transformed R component and Z component.

- A too narrow frequency band can result in wide plateaus of correction angles with high cross-correlation values. If the frequency band is dominated by very low frequencies only, the resolution abilities can be limited.
  It can help to test different frequency bands, e.g. 0.01-0.05 Hz, 0.01-0.03 Hz, 0.02-0.05 Hz.

- The obtained angle for events from a certain back azimuth direction can be influenced by large scale subsurface features. In our case, the third plot does not show such a trend.

- The second plot, showing correction angle over event time, can help to understand if there is a jump over time, as for example due to maintenance work in the field including a correction of sensor orientation.

3. Typically observed data and metadata error gallery

3.1 Gains & PSDs

3.1.2 Large offsets of PSDs

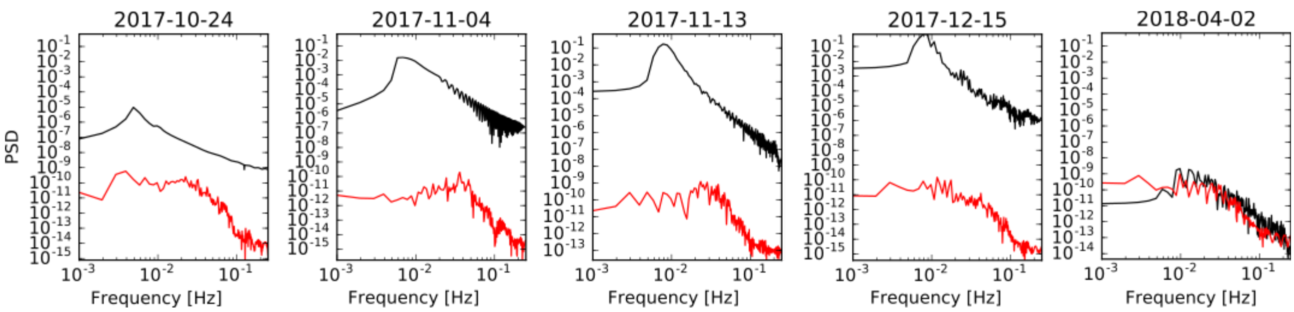

Fig. 10: Example of significant differences between the recorded and synthetic frequency spectra before the replacement of the acquisition system in March 2018. From Petersen et al., 2019.

3.2 Sensor orientation

3.2.1 Example: Inverse polarities of horizontal components

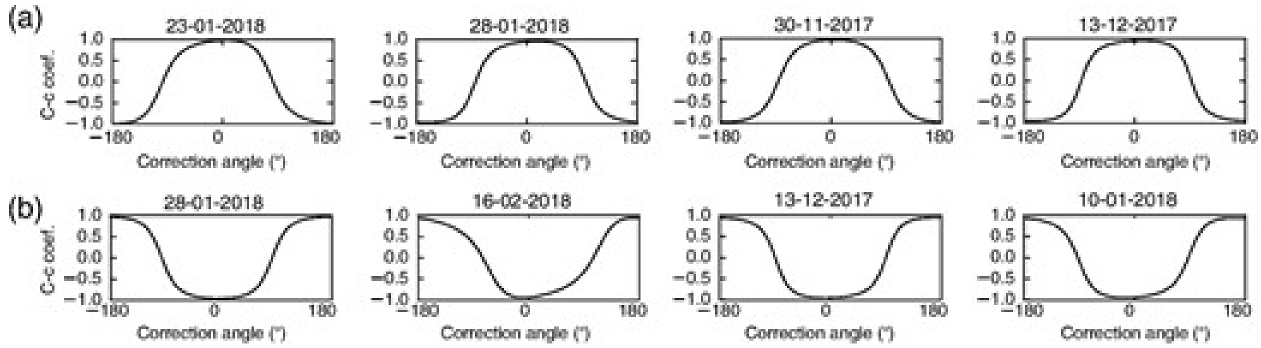

Fig. 11: (a) A correctly oriented sensor results in maximum cross-correlation values for a correction angle of 0°. (b) If a station has inverse polarities of the horizontal components, the resulting correction angle will be 180°. From Petersen et al., 2019.

3.2.2 Example: Temporal change of orientation:

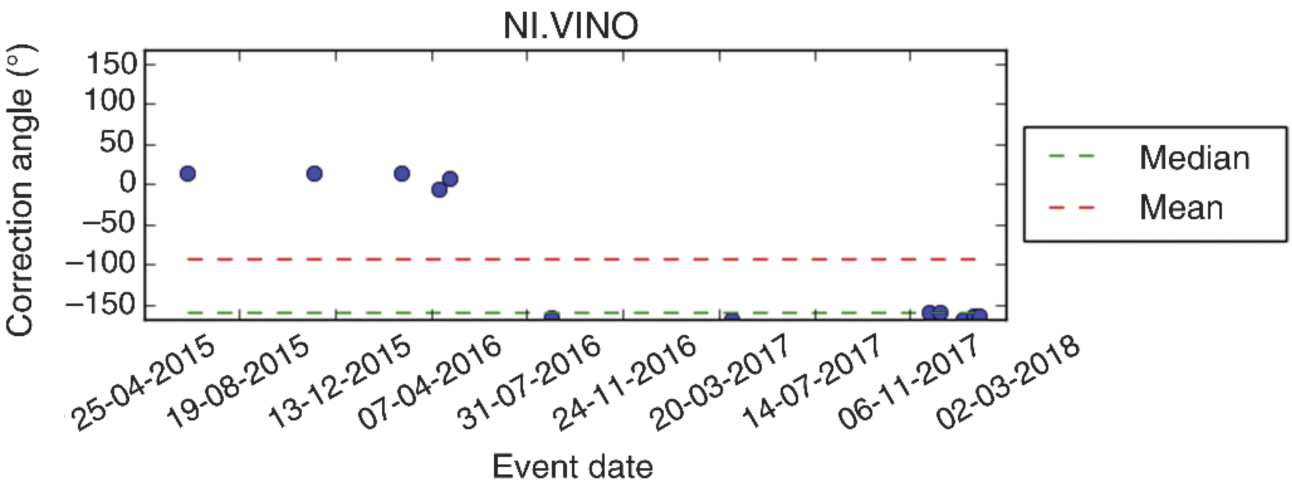

Fig. 12: Correction angle over time showing a temporal change of orientation. Such changes can be related to sudden flips of polarity or e.g. to maintenance work. From Petersen et al., 2019.

3.2.3 Example: Evaluation of misorientations from result map

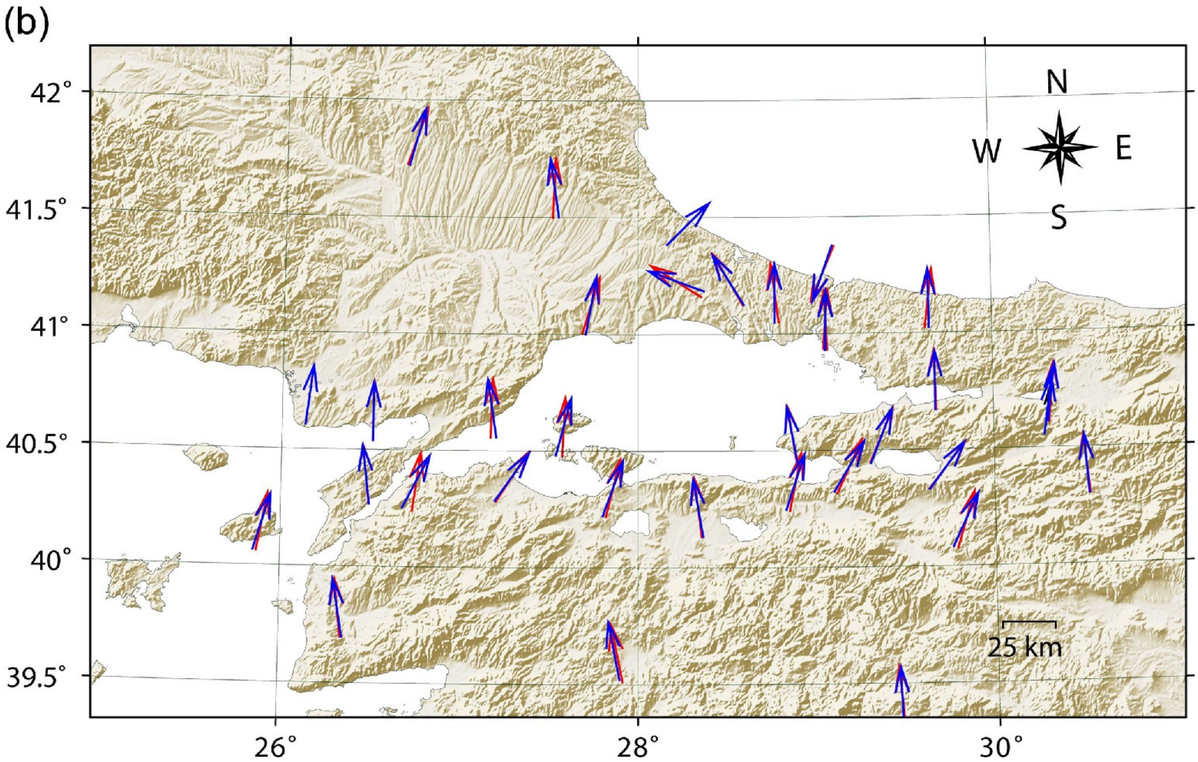

Fig. 13: Orientation of North components as obtained from AutoStatsQ (red) and a P polarization methods (blue). Arrows pointing not in North direction indicate misorientations. From Büyükakpınar et al., 2021.

4. AutoStatsQ commands overview:

- show basic commands/ help:

```
autostatsq -h
```

- generate an example config file:

```
autostatsq --generate_config
```

- run AutoStatsQ

```
autostatsq --config name_of_config_file --run
```

- To get detailed error/ info logging, use the -l option:

```
autostatsq --config name_of_config_file --run -l INFO
```

- Helpful for debugging: Forward terminal output into a logging file by using the option `--logoutput FILENAME`.

- To generate a html report after running AutoStatsQ use the option:

```
autostatsq --config name_of_config_file --report
```

The report is saved in a directory result_report and the file index_report.html can be opened for example with firefox.

5. FAQs:

My stations did only run for a few months so I do not find many teleseismic events. What do I do now?

- Set wedges_width to 1°
- smaller magnitudes, include closer events

References:

Büyükakpınar, P., Aktar, M., Maria Petersen, G., & Köseoğlu, A. (2021). Orientations of broadband stations of the KOERI seismic network (Turkey) from two independent methods: P-and Rayleigh-wave polarization. *Seismological Society of America*, *92*(3), 1512-1521.

Petersen, G. M., Cesca, S., Kriegerowski, M., & AlpArray Working Group. (2019). Automated quality control for large seismic networks: Implementation and application to the AlpArray seismic network. *Seismological Research Letters*, *90*(3), 1177-1190.