B3AMpy: Beamforming Toolbox for three-component ambient seismic noise

Authors: Claudia Finger (1)
Katrin Löer (2)
Affiliations: 1: Fraunhofer IEG, Fraunhofer Institution for Energy Infrastructures and Geothermal Systems, Bochum
2: Department of Geoscience & Engineering, Delft University of Technology
Contact: claudia.finger@ieg.fraunhofer.de

This is an archived version of version 0.1 with small updates from 26.10.2023. The latest version can be found at: https://github.com/cl-finger/B3Ampy
A Matlab Version of this code can be found at https://github.com/katrinloer/B3AM

List of related publications:

Finger, C. and Löer, K. (2023), Depth of sudden velocity increases from multi-mode Rayleigh waves derived with three-component ambient noise beamforming, EGU General Assembly 2023, Vienna, Austria, 24–28 Apr 2023, EGU23-12396, https://doi.org/10.5194/egusphere-egu23-12396.

Löer, K., Finger, C., Obiri, E., and Kennedy, H. (2023). A comprehensive beamforming toolbox to characterise surface and body waves in three-component ambient noise wavefields, EGU General Assembly 2023, Vienna, Austria, 24–28 Apr 2023, EGU23-5670, https://doi.org/10.5194/egusphere-egu23-5670.

Löer, K., Toledo, T., Norini, G., Zhang, X., Curtis, A., and Saenger, E.H. (2020), Imaging the Deep Structures of Los Humeros Geothermal Field, Mexico, Using Three-Component Seismic Noise Beamforming, Seismological Research Letters, 91 (6): 3269–3277.

Löer, K., Riahi, N., and Saenger, E. H. (2018), Three-component ambient noise beamforming in the Parkfield area, Geophysical Journal International, Volume 213, Issue 3, Pages 1478–1491, https://doi.org/10.1093/gji/ggy058

Riahi, N., and Saenger, E. H. (2014), Rayleigh and Love wave anisotropy in Southern California using seismic noise, Geophys. Res. Lett., 41, 363– 369, doi:10.1002/2013GL058518.

Riahi, N., Bokelmann, G., Sala, P., and Saenger, E. H. (2013), Time-lapse analysis of ambient surface wave anisotropy: A three-component array study above an underground gas storage, J. Geophys. Res. Solid Earth, 118, 5339– 5351, doi:10.1002/jgrb.50375.

0) What you need

- Python 3.8.1
- Python libraries (version I used)
  - numpy (1.19.1)
  - matplotlib (3.3.2)
  - scipy (1.5.2)
  - glob
  - math
  - datetime
  - time
  - multiprocessing
- your 3C waveforms in .npy format (Example of conversion from mseed is provided in prepare_input/)
- list of station coordinates in cartesian coordinates in Meters from center of array (Example of converting GPS to UTM is provided in prepare_input)

1) Prepare your variables

- modify params.py
  - provide your paths and filenames to your 3C waveform data (.npy) and station file (.txt). Note: You should be able to put paths with / or \ depending on your operating system.
  - fill in required information line by line
  - modify all desired resolution parameters manually or leave on default

- B3Ampy will cut waveforms into small time windows and will analyse discrete frequencies. All input data found in file will be analysed. If you want to process your data in smaller time chunks (e.g. a day at a time or an hour at a time), provide only reduced input. Note that the current version of B3Ampy is consistent in itself and will overwrite any existing files in paths.
- B3Ampy assumes that you have done general quality checks beforehand and excluded data from unwanted stations. Make sure that three components of data are available for all stations.

2) Test your parameters and resolution

- run B3Am_prep.py
- B3Am_prep.py will import your desired resolutions and your station file as defined in params.py
- array response function will be calculated and plotted together with wavenumber and frequency limits (default or your custom values)
- check if everything is as you want it and change values in params.py until you are satisfied before proceeding

3) Run the beamformer

- run B3Am_main.py
- if you opted to run the beamformer in parallel, you can start it on your local machine in parallel or submit it to a queueing system on a HPC cluster (probably needs a job file suited for the queueing system, not included here)
- B3AM performs the four major steps successively:

i) pre-processing (can include downsampling, spectral whitening, one-bit normalisation)
ii) short-time fast Fourier transformation
iii) frequency-wavenumber analysis (beamforming)
iv) identification of extrema (maxima) in the beam responses

- The script provides output in the command line documenting it's progress. It is recommended to pipe this output to a file when running on cluster.
- Script may take some time to compute. The test dataset included in this version comprises of 2 min of 3C synthetic waveform data recorded on 100 receivers and takes about 5 min to compute in serial on my local laptop (intel i7 Quad Core 1.2 GHz)
- Outputs of B3Am_main.py are

  i) Figures of beampower response and picked extrema for each time window and frequency (if option is selected in params.py) sorted in subfolders named with frequency

  ii) List files named results_fXX.txt for each analysed frequency containing parameters of each picked extrema. Saved to outdir provided in params.py

iii) .npy files containing resolutions for each parameter saved in outdir. This enables plotting at same resolutions.

4) Analyse and plot results

- run B3Am_post.py
- params.py is read to find path where results have been saved
- results (list files) are read
- .npy files containing resolutions
- A number of plots are created
    - wavefield composition in percentage summed over all frequencies
    - wavefield composition relative and absolute as histogram and line plot
    - statistics (counts) of parameters for each wave type
    - histograms of parameters and picks for each wave type
- Picked parameters (plus uncertainties) are saved as list files named [wavetype]_picks.txt for future use (e.g. in inversion schemes)
- Note that wavenumbers are always horizontal wavenumbers. For surface waves, horizontal velocity (phase velocity) can be calculated as v = f/k. For body waves, the incidence angle needs to be considered to convert wavenumber to velocity, which is not implemented yet.