




# NeuroAnalyzer: Julia toolbox for analyzing neurophysiological data

Adam Wysokiński <sup>1</sup>

<sup>1</sup> Medical University of Lodz, Poland

DOI: [10.21105/joss.07734](https://doi.org/10.21105/joss.07734)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

---

Editor: [Kevin M. Moerman](#)  

## Reviewers:

- [@kinleyid](#)
- [@ferchaure](#)
- [@TheCedarPrince](#)

Submitted: 29 October 2024

Published: 16 March 2025

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

NeuroAnalyzer is a collaborative, non-commercial Julia toolbox developed for researchers in psychiatry, neurology and neuroscience. It's repository is located at <https://codeberg.org/AdamWysokinski/NeuroAnalyzer.jl> and mirrored at JuliaHealth <https://github.com/JuliaHealth/NeuroAnalyzer.jl>. This software is licensed under The 2-Clause BSD License.

## Background

NeuroAnalyzer is a Julia ([Bezanson et al., 2017](#)) toolbox for analyzing neurophysiological (such as EEG, MEG, MEP and EDA) data. To my knowledge, there is no complete Julia solution for neurophysiology data analysis. There are some available packages (some are not actively developed) to perform individual tasks, such as importing EDF/BDF data ([EDF.jl](#), [EDFPlus.jl](#), [BDF.jl](#)), processing EEG data ([EEG.jl](#), abandoned), ClusterDepth multiple comparison ([ClusterDepth.jl](#)), performing linear/GAM/hierarchical/deconvolution regression on biological signals ([Unfold.jl](#)) or plotting topomaps ([TopoPlots.jl](#)). Therefore, the development of NeuroAnalyzer was started in 2022 in order to provide researchers in the field of neuroscience, psychiatry or neurology a complete, complex (but still easy to use) solution for importing, editing, processing, analyzing and visualizing neurophysiological data.

## Statement of need

There are many excellent MATLAB and Python based EEG/MEG/NIRS applications (e.g. EEGLAB ([Delorme et al., 2011](#)), Fieldtrip ([Oostenveld et al., 2011](#)), Brainstorm or MNE ([Gramfort, 2013](#))). They have been in development for many years and are well established in the scientific community. Many state-of-the-art papers were published using data prepared using these programs. However, compared with Python and MATLAB, there are many advantages of Julia, which underlie my decision to start developing such a toolbox in Julia. I believe that Julia is the future of scientific computing and scientific data analysis ([Selvaraj, 2022](#)). Major advantages of Julia are listed in Julia documentation:

- Julia is fast ([Bezanson et al., 2018](#)). In many situations Julia is considerably faster than Python (without having to use numba/cython) and MATLAB (see [Benchmarks Game](#) for example benchmarks). Moreover, Julia provides unlimited scalability. Julia programs can easily be ran on a large cluster or across distributed computers.
- Julia is open-source and free. Increasing MATLAB licensing costs are prohibitive to individual researchers and many research institutions.
- From its very beginning Julia is being focused on scientific computations ([Bezanson et al., 2014](#)). Currently only Julia, C, C++ and Fortran belong to the HPC (High Performance Computing) Petaflop Club. Julia is designed for distributed and parallel computations, making it great for distributed analyzes of large data sets.

- Most of the Julia packages are written in pure Julia. It's easier to understand and modify their code if you already know Julia.
- Julia is beautifully designed (multiple dispatches, metaprogramming capabilities, user-friendly syntax and readability, integrated packages management), making programming in Julia a pure pleasure (Pal et al., 2024). This elegant design makes Julia easy to read and write.

To sum it up, the target audience of this software are neuroscientists looking for highly performant and complete toolbox for collecting, editing and analyzing neurophysiological data.

## Three years of development

Currently NeuroAnalyzer includes functions for importing, editing, processing, visualizing, and analyzing EEG, MEP and EDA data. Preliminary functionality is also available for MEG, NIRS, ECoG, SEEG and iEEG recordings.

Various methods for modeling non-invasive brain stimulation protocols (tDCS, tACS, tRNS, tPCS, TMS, TUS, INS) are also being implemented (NeuroStim submodule). Another submodule, NeuroTester, will allow designing and running psychological studies. Certain neurophysiological data can be recorded using NeuroRecorder submodule.

NeuroAnalyzer contains a set of separate (high- and low-level) functions. Some interactive graphical user interface (GUI) functions are also available. NeuroAnalyzer functions can be combined into an analysis pipeline, i.e. a Julia script containing all steps of your analysis. This, combined with processing power of Julia language and easiness of distributing calculations across computing cluster, will make NeuroAnalyzer particularly useful for processing large amounts of neurophysiological data.

Currently NeuroAnalyzer is focused on resting-state analysis. Some ERP functions are already available, while other type of analyses will be developed in future versions. The goal is to make a powerful, expandable and flexible environment for processing and analysis of various types of neurophysiological data.

The following list of already implemented functionalities is presented below, with many more to [come in the future releases](#).

1. Load neurophysiological recordings:
  - EEG (EDF, EDF+, BDF, BDF+, GDF, Alice4, DigiTrack, BrainVision, CSV, EEGLAB, NPY, Thymatron, NCS, CNT, XDF)
  - MEG (FIFF)
  - NIRS (SNIRF, NIRS, NIRX)
  - MEP (DuoMAG)
  - body sensors: acceleration, magnetic field, angular velocity and orientation
  - electrode positions (CED, LOCS, ELC, TSV, SFP, CSD, GEO, MAT, TXT, DAT, ASC)
2. Edit:
  - edit channel data (unit, type, label)
  - edit electrode locations
  - trim (remove part of the signal)
  - resample (up/down)
  - divide into epochs (fixed and by event markers)
  - delete channels/epochs
  - auto-detect bad channels/epochs
  - interpolate channels (planar interpolation/linear regression)
3. Process:
  - reference (common/averaged/auricular/mastoid/Laplacian/custom montage)

- filter (FIR/IIR/Remez/moving average/moving median/polynomial filters), all types (HP, LP, BP, BS); with preview of filter response
  - remove power line noise
  - auto-detect and remove electrode pops
  - ICA/PCA decompose/reconstruct
  - convolution (in time and frequency domain)
  - create ERP (event-related potentials)
  - NIRS: convert raw light intensity to optical density and HbO/HbR/HbT concentrations
4. Analyze:
- signal comparison
  - stationarity
  - frequency analysis: total power, band power (absolute and relative)
  - auto- and cross- covariance and correlation (biased and unbiased)
  - time-frequency analysis: various spectrogram methods (FFT-based, short-time Fourier transform, multi-tapered periodogram, Morlet wavelet convolution, Gaussian and Hilbert transform, continuous wavelet transformation)
  - coherence and magnitude-squared coherence
  - mutual information
  - entropy, negentropy
  - envelopes (amplitude, power, spectrogram)
  - power spectrum slope
  - PLI/ISPC/ITPC
  - ERP: detect peaks, analyze amplitude and average amplitude
  - EROs (event-related oscillations): spectrogram, power spectrum
  - HRV (heart rate variability): time-domain analysis (MENN, MDNN, VNN, SDNN, RMSSD, SSSD, NN50, pNN50, NN20, pNN20)
  - MEPs: detect peaks, analyze amplitude and average amplitude
5. Plotting:
- signal (single-/multi-channel)
  - power spectrum (single-/multi-channel, 2D/3D)
  - spectrogram (single-/multi-channel)
  - topographical maps (various methods of interpolation)
  - weights at channel locations
  - weighted inter-channel connections
  - matrices (channel  $\times$  channel)
  - heatmaps
  - channel/epoch data (histogram/bar plot/dot plot/box plot/violin plot/polar plot/paired data)
  - ERPs: amplitude, topographical distribution
  - EROs: spectrogram, power spectrum
  - MEPs: amplitude

Interactive (Gtk3-based) plotting (signal, power spectrum, spectrograms, topomaps, ICA components) and editing (signal, electrode positions) are also implemented.

All computations are performed using the double-precision 64-bit floating point format. NeuroAnalyzer data is stored using standard Julia Array and can be easily exported as DataFrame. Thus, external processing of those data using Julia packages is readily available.

NeuroAnalyzer also includes NeuroRecorder, a set of functions for recording various neurophysiological signals:

1. Finger Tapping Test (FTT) – using computer keyboard or external panel attached to Raspberry Pi
2. Electrodermal Activity (EDA) = Galvanic Skin Response (GSR) – via Raspberry Pi
3. Two-point Pinch Test (TPT) – using finger-worn accelerator attached to Raspberry Pi

(in development)

4. Angular Velocity Sensors (AVS) – via Raspberry Pi (in development)

Extensive documentation and tutorials are available at <https://www.neuroanalyzer.org/docs> and <https://neuroanalyzer.org/#tutorials>, respectively.

NeuroAnalyzer functionality can be easily expanded using [plugins](#) (written in Julia).

For common tasks (importing, filtering, referencing) NeuroAnalyzer performance is  $\sim 1.7\times$  higher than MNE and  $\sim 3\times$  higher than EEGLAB. Benchmarks are available at <https://neuroanalyzer.org/benchmarks.html>.

## Research projects using the software

NeuroAnalyzer has been used in the preparation of the following publications: Wysokiński et al. (Wysokiński et al., 2023), Sochal M. et al. (Sochal, Binienda, et al., 2024), Sochal et al. (Sochal, Ditmer, et al., 2024), Datseris & Zelko (Datseris & Zelko, 2024) and Wysokiński & Pazdrak (in press).

In the Department of Old Age Psychiatry and Psychotic Disorders (Medical University of Lodz, Poland) we are using NeuroAnalyzer for analyzing EEG data to analyze EEG data from our research projects on functional (EEG-based) connectivity in schizophrenia and EEG markers of tDCS stimulation.

## References

- Bezanson, J., Chen, J., Chung, B., Karpinski, S., Shah, V. B., Vitek, J., & Zoubritzky, L. (2018). Julia: Dynamism and performance reconciled by design. *Proceedings of the ACM on Programming Languages*, 2(OOPSLA), 1–23. <https://doi.org/10.1145/3276490>
- Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2014). Julia: A fresh approach to numerical computing. *CoRR*, *abs/1411.1607*. <http://arxiv.org/abs/1411.1607>
- Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1), 65–98. <https://doi.org/10.1137/141000671>
- Datseris, G., & Zelko, J. S. (2024). Physiological signal analysis and open science using the Julia language and associated software. *Frontiers in Network Physiology*, 4, 1478280. <https://doi.org/10.3389/fnetp.2024.1478280>
- Delorme, A., Mullen, T., Kothe, C., Akalin Acar, Z., Bigdely-Shamlo, N., Vankov, A., & Makeig, S. (2011). EEGLAB, SIFT, NFT, BCILAB, and ERICA: New tools for advanced EEG processing. *Computational Intelligence and Neuroscience*, 2011, 1–12. <https://doi.org/10.1155/2011/130714>
- Gramfort, A. (2013). MEG and EEG data analysis with MNE-python. *Frontiers in Neuroscience*, 7. <https://doi.org/10.3389/fnins.2013.00267>
- Oostenveld, R., Fries, P., Maris, E., & Schoffelen, J.-M. (2011). FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data. *Computational Intelligence and Neuroscience*, 2011, 1–9. <https://doi.org/10.1155/2011/156869>
- Pal, S., Bhattacharya, M., Dash, S., Lee, S.-S., & Chakraborty, C. (2024). A next-generation dynamic programming language julia: Its features and applications in biological science. *Journal of Advanced Research*, 64, 143–154. <https://doi.org/10.1016/j.jare.2023.11.015>
- Selvaraj, N. (2022). Meet Julia: The Future of Data Science. In *Medium*. <https://towardsdatascience.com/meet-julia-the-future-of-data-science-52414b29ebb>

- Sochal, M., Binienda, A., Tarasiuk, A., Gabryelska, A., Białasiewicz, P., Ditmer, M., Turkiewicz, S., Karuga, F. F., Fichna, J., & Wysokiński, A. (2024). The Relationship between Sleep Parameters Measured by Polysomnography and Selected Neurotrophic Factors. *Journal of Clinical Medicine*, 13(3), 893–893. <https://doi.org/10.3390/jcm13030893>
- Sochal, M., Ditmer, M., Tarasiuk-Zawadzka, A., Binienda, A., Turkiewicz, S., Wysokiński, A., Karuga, F. F., Białasiewicz, P., Fichna, J., & Gabryelska, A. (2024). Circadian Rhythm Genes and Their Association with Sleep and Sleep Restriction. *International Journal of Molecular Sciences*, 25(19), 10445–10445. <https://doi.org/10.3390/ijms251910445>
- Wysokiński, A., Szczepocka, E., & Szczakowska, A. (2023). Improved cognitive performance, increased theta, alpha, beta and decreased delta powers after cognitive rehabilitation augmented with tDCS in a patient with post-COVID-19 cognitive impairment (brain-fog). *Psychiatry Research Case Reports*, 2(2), 100164–100164. <https://doi.org/10.1016/j.psycr.2023.100164>