NeoViewer: Facilitating reuse of electrophysiology data through browser-based interactive visualization

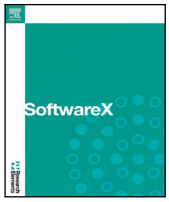
Onur Ates, Shailesh Appukuttan, Hélissande Fragnaud, Corentin Fragnaud, Andrew P. Davison

SoftwareX 26 (2024) 101710

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx



Original software publication

NeoViewer: Facilitating reuse of electrophysiology data through browser-based interactive visualization

Onur Ates, Shailesh Appukuttan, Hélissande Fragnaud, Corentin Fragnaud, Andrew P. Davison *

Université Paris-Saclay, CNRS, Institut des Neurosciences Paris-Saclay, Saclay, 91400, France

ARTICLE INFO

Dataset link: https://github.com/neuralensemble/neo-viewer

Keywords:
Neurophysiology data
Visualization
FAIR
Web browser
File formats

ABSTRACT

Interactive visualization of scientific data in a web browser can be a powerful tool, particularly in a context of open data sharing following the FAIR principles. It obviates the need to install visualization software locally, and allows scientists to explore shared datasets of potential interest before further analysis or download. NeoViewer is an open-source tool for visualizing and exploring neurophysiology datasets stored in a wide variety of file formats. It enables neuroscientists to view recordings of membrane potential, local-field potentials, spike trains, and other neurophysiological signals within their web browser. The tool consists of a web API, easily deployed as a Docker container, and a Javascript component, available for the ReactJS and AngularJS frameworks, that can be embedded in any web page.

Code metadata

Current code version	GitHub tag: v2.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-24-00007
Permanent link to Reproducible Capsule	–
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	Python (FastAPI), JavaScript (AngularJS, ReactJS)
Compilation requirements, operating environments & dependencies	Docker
If available Link to developer documentation/manual	https://neo-viewer.brainsimulation.eu/
Support email for questions	support@ebrains.eu

1. Motivation and significance

Recent years have seen an increased and concerted effort towards the sharing and dissemination of scientific resources. This has culminated in principles such as FAIR (Findable, Accessible, Interoperable, Reusable) that help set guidelines for effectively sharing scientific data [1]. The availability of large amounts of scientific data leads researchers into a new set of challenges. Scientists searching for datasets to use in their research are required to identify if a given dataset meets their requirements. This process can be significantly aided if these shared datasets have been curated in detail prior to sharing [2], although this is typically not the case.

It is therefore imperative that researchers have a way to quickly examine the available data, to help shortlist datasets for use in their study. The first step towards this is generally to visualize and explore the data. Unfortunately, it is very common that the various data files exist in very different file formats, and often cannot be handled by the same tool/software. Furthermore, it is cumbersome to have to download each data file (which can at times be very large) from its repository and find a suitable tool (or develop your own custom script) to load and visualize the data.

The open-source NeoViewer tool presented here aims to address this problem. It is a web-based visualization tool that makes it easy to interactively visualize neurophysiology data within the web browser. Web-page authors may allow end users to specify the URL of the target data file in a web form, and the visualizer will plot the contained data, or they may predefine the URLs of the data files that should be shown.

* Corresponding author.
E-mail address: andrew.davison@cnrs.fr (Andrew P. Davison).

https://doi.org/10.1016/j.softx.2024.101710
Received 4 January 2024; Received in revised form 14 March 2024; Accepted 22 March 2024
Available online 8 April 2024

There is also a trend towards more interactive presentation of scientific data, with live/interactive/executable papers and other web-based documents appearing as a complement or alternative to traditional static scientific papers. NeoViewer was developed to perfectly fit into this ecosystem, so that individual scientists can showcase their own data on their websites, and developers can easily integrate the NeoViewer visualizer component into their own web-based tools and services. Data repositories, which host and help disseminate the numerous scientific data files, could employ this visualizer within their existing websites to offer a richer user experience, and help improve scientific productivity.

2. Software description

2.1. Software architecture

NeoViewer has two components: a web server providing a REST API written using the FastAPI framework and a web component written in Javascript. The REST API reads neurophysiology data files and makes the content available in JSON format. The web component, with implementations in two of the most popular Javascript frameworks, AngularJS and ReactJS, reads data from the API and displays it in the web browser using the open source Plotly library.

NeoViewer builds on the Neo software [3], which has support for reading a wide variety of neurophysiology file formats, including proprietary formats such as AlphaOmega, Plexon and NeuroExplorer, and open formats such as Neurodata Without Borders [4]. Neo loads this data into a common object model with the aim of increasing the interoperability of neurophysiology software tools and thus facilitating sharing of data between different projects.

Neo defines various types of neural activity data as illustrated in Fig. 1. NeoViewer currently supports the following subset of these:

- **Block**: outer container for an entire recording session.
- **Segment**: inner container for data objects recorded at the same time, e.g. in response to a given stimulus.
- **SpikeTrain**: an array of action potentials (spikes) emitted over a period of time.
- **AnalogSignals**: continuous signal data with a fixed sampling interval (e.g. membrane potentials).
- **IrregularlySampledSignal**: analog signal with a varying sampling interval (e.g. recordings from simulations using a variable-time-step integration method).

Fig. 2 shows screen captures of visualizations of some of these data types. Support for the remaining Neo types, such as 'Events' (e.g. triggers during behavioral experiments), 'Epochs' (e.g. the time during which a stimulus is presented), and 'ImageSequence' (for data produced by functional microscopy modalities such as calcium imaging or voltage-sensitive dye imaging) is planned.

2.2. Software functionalities

NeoViewer can be integrated in an HTML document by anyone with some experience of modern web development. A single HTML page can contain multiple instances of the NeoViewer. The 'source' is the minimal attribute that needs to be set for each instance of the NeoViewer, indicating the URL of the data file to be loaded. After the data is loaded, it is possible to choose different Blocks and Segments, via a drop down menu. In the segments you can select AnalogSignals (including IrregularlySampledSignal) or SpikeTrains, whichever is available in the file. As an example, Segments might include data for a stimulus and the corresponding response of the neuron (e.g. membrane potential) as two separate AnalogSignal plots.

A more detailed view of the data can be achieved by zooming with mouse gestures. There is an option to show all the signals from a given segment on the same axes, if the units and sampling rates match. It is also possible to show all the signals of a given type across all segments on the same axes. SpikeTrain recordings are represented as a raster plot, with the spikes from each neuron on a separate line and with different colors. Removing and replacing individual traces in a plot takes a single mouse click. Any plot can be downloaded as a PNG image file.

```
import Visualizer from 'neural-activity-visualizer-react'
...
let source_url = "<<file_path>>"
<Visualizer source={source_url} />
```

Listing 1: Basic Usage of NeoViewer using ReactJS

```
import Visualizer from 'neural-activity-visualizer-react'
...
<Visualizer
        source = "<<file_path>>"
        width = {750}
        height = {500}
        downSampleFactor = {5}
        segmentId = {15}
        signalId = {1}
/>
```

Listing 2: Specifying additional attributes for NeoViewer using ReactJS

Experiments often produce large volumes of data which in turn can affect the performance of the visualization. NeoViewer therefore provides an optional 'down-sample factor' attribute which is used to increase the sampling period of the plot to reduce the data loading time. Caching and the use of lazy loading (where individual data arrays are only read on demand) are also used to provide faster loading times. The file format is inferred from the file extension where possible, but since many data recording tools produce output files with the same file extensions (e.g. '.dat'), a web page author can explicitly specify the file type using an HTML tag attribute. The ReactJS version of the component offers some additional attributes to assist in further customization. These are described in the online documentation, along with a live example to help explore them.

2.3. Using the software

The REST API can be deployed in a standalone web server (Docker image available from docker-registry.ebrains.eu/neuralactivity/neo-viewer:prod). A demonstration server is available at https://neo-viewer.brainsimulation.eu, hosted by the EBRAINS digital research infrastructure. The ReactJS version of the web component is available from the npm registry; the AngularJS version can be obtained from GitHub. All source code is available on GitHub at https://github.com/NeuralEnsemble/neo-viewer.

The neural-activity-visualizer-react app can be installed via npm as follows:

```
npm install --save neural-activity-visualizer-react
```

Listing 1 shows simple source code for basic usage of the NeoViewer to visualize a data file.

Listing 2 illustrates how additional attributes can be specified for customizing NeoViewer. In this example, we have specified the required dimensions of the visualizer, requested for the data to be down-sampled by a factor of 5, and indicated that signal 1 of segment 15 should be displayed.

The documentation for the ReactJS documentation provides a listing of all the additional attributes, and the interactive demo also helps

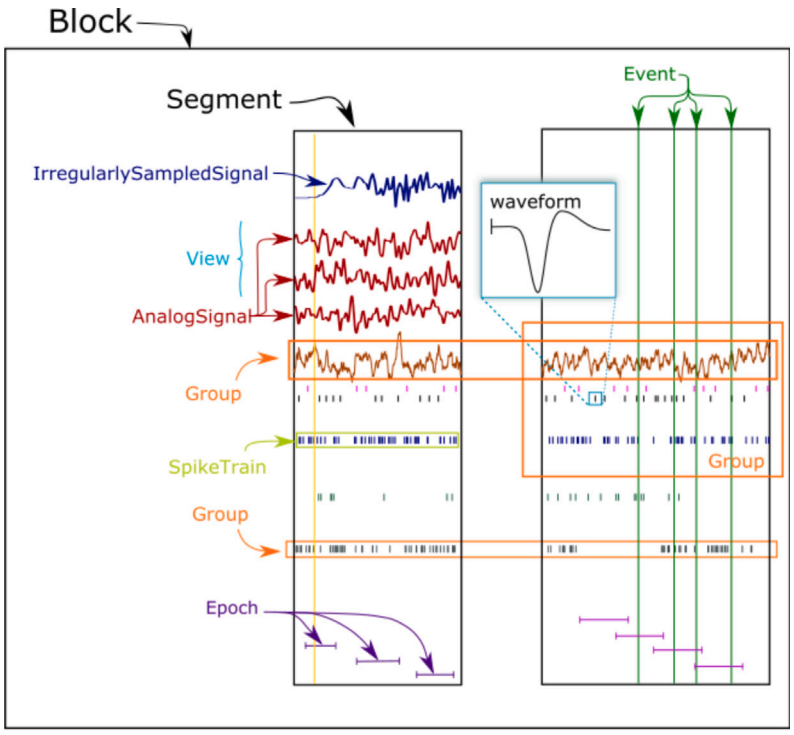


Fig. 1. Illustration of different types of data objects handled by Neo.

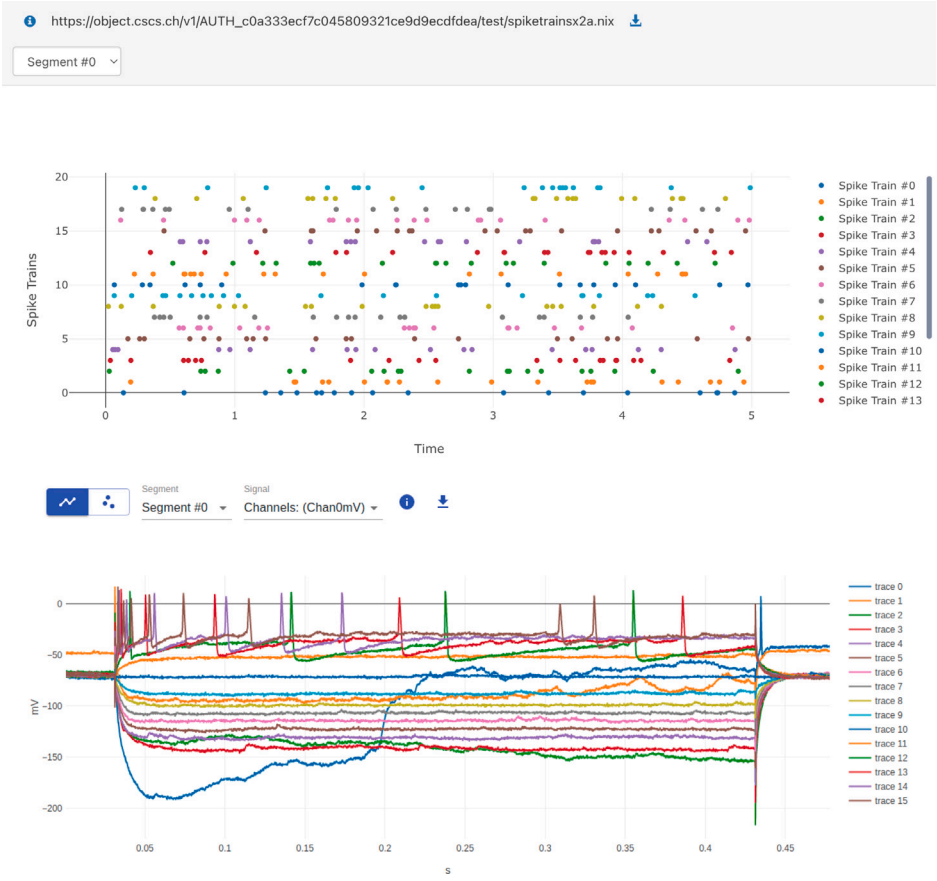


Fig. 2. Screenshots of data visualization in Neo Viewer. (top) All the spike trains in a given segment being plotted using the AngularJS implementation of the component. (bottom) A plot of a given analog signal channel across all segments using the ReactJS component.

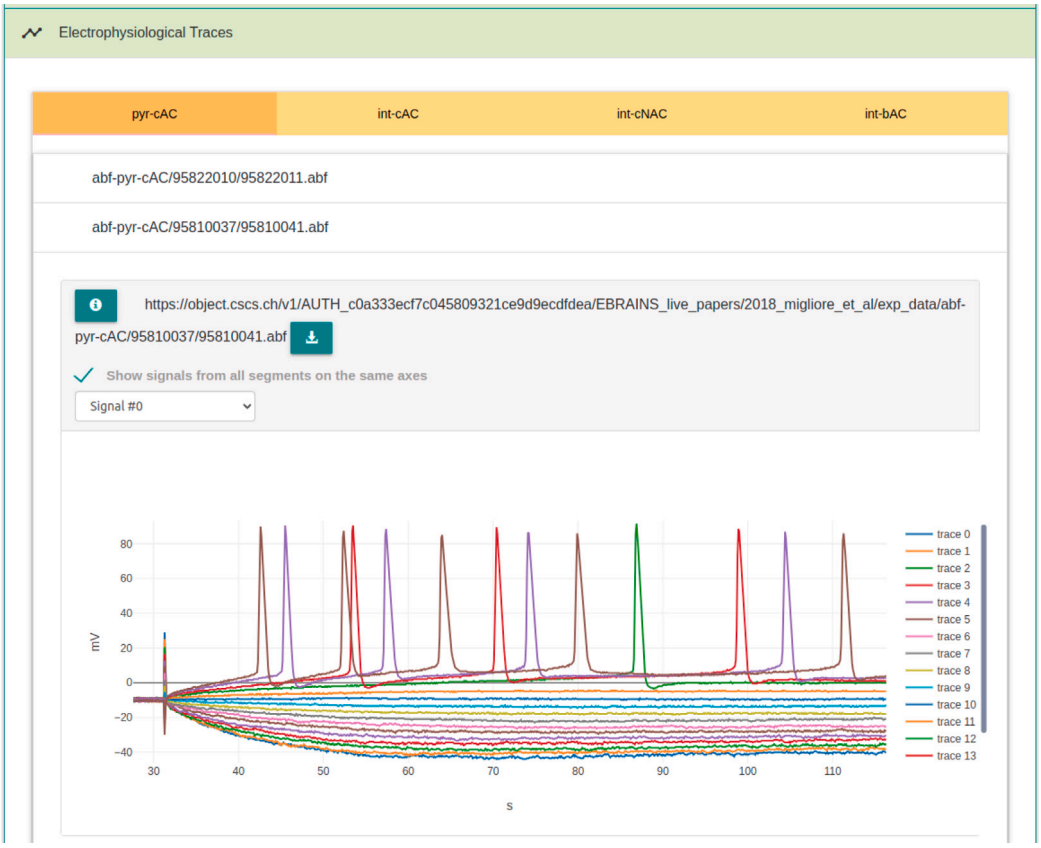


Fig. 3. Example usage of NeoViewer inside EBRAINS Live Papers. Clicking on a file name opens a viewer for that file; multiple viewers can be open at once.

auto-generate the relevant source code. The documentation can be viewed at https://neo-viewer.brainsimulation.eu/react/.

Similarly, examples of source code and usage of the AngularJS component is available on its dedicated documentation page at https://neo-viewer.brainsimulation.eu/angularjs.

To use NeoViewer with files stored locally, for example because the files contain sensitive data that cannot be shared online, a minimal webserver can be run from within a terminal to serve files only to the computer's internal network. Such a webserver is part of the Python standard library, and as such is available by default on most modern operating systems. To launch the server, run:

```
python3 -m http.server
```

and then the data file URL to be provided to NeoViewer would be of the form http://localhost:8000/path/to/my/data/file.

3. Illustrative examples

The EBRAINS live paper platform extensively uses NeoViewer to visualize experimental data from published studies [5]. This allows viewers to readily examine the data from within the webpage, without having to download these files and load them in compatible visualization tools.

As an example, let us take a look at the Live Paper accompanying Migliore et al. [6]. This can be accessed at https://doi.org/10.25493/EF9C-ZKU.

The study of Migliore et al. involved the development of biophysically realistic models of hippocampal neurons. The experimental data was extracted from several electrophysiological recordings. The authors shared these data in their Live Paper, and the NeoViewer enables them to provide visualization from within the web pages. Fig. 3 illustrates an example of one of the data files being visualized from within the live paper. Each data file is linked to an individual instance of NeoViewer, thereby enabling simultaneous visualization of multiple files. For each file, the visualizer allows users to specify the segments and signals of interest, or, alternatively, to show the specified signal from all segments at once. The plot allows users to interact with the data via options such as zoom, pan, scale and allows reading values. Entire data files can be saved in source format, or downloaded as images.

4. Impact

NeoViewer substantially simplifies the work of researchers in visualizing and exploring various forms of electrophysiological data stored in different file formats. Sharing of data is a fundamental pillar of scientific research, which is disrupted by lack of interoperability wherein data in certain file types can only be accessed by specific software tools. Moreover, electrophysiological data is stored often in large files which makes their loading, plotting and manipulation computationally intensive.

This is where NeoViewer fills an evident gap. NeoViewer is designed to provide a simple, fast and smooth user experience to researchers. It achieves this in several ways. First and foremost, NeoViewer is built on top of the widely used file standardization tool called 'Neo' [3], which largely solves the interoperability issues and enables handling of data stored in different file formats. The performance-related issues were overcome through the following approaches: (i) After evaluating several popular plotting libraries, Plotly [7] was chosen as it provides stability, good performance, is lightweight and offers useful features such as zooming, selecting/disabling signals and downloading the plots

as images. (ii) We implemented a lazy loading feature, which enables loading only a smaller subset of the data that is immediately required, while delaying the loading of remaining data. This greatly helps improve performance and reduce bandwidth requirements. (iii) NeoViewer provides the feature to optionally increase the sampling period of individual plots by specifying a *downsample* factor. For example, a *downsample* factor of 2 (default value is 1; i.e. no down sampling) would skip every other data point, and thereby produce a plot with only half the original data points.

To the best of our knowledge NeoViewer is the only web-based and standalone open-source visualization tool for electrophysiological data, built as a library that can be easily integrated into other tools and workflows. Other related web based tools in neuroscience, such as Geppetto [8] (with OSB extension) employed in Open Source Brain [9], Allen Brain Atlas-Driven Visualizations [10] and NEST Desktop [11] are capable tools but tightly coupled to the services they were developed for, whereas NeoViewer can be easily embedded in any web-based project.

Web-based visualization makes it possible to skip the whole process of downloading data to a local computer and creating a software environment to load and visualize it. In its absence, a common approach, for example using the Python programming language, would be to create a suitable software environment locally by first identifying the relevant libraries (packages) and installing them, loading the data and then using a visualization library like Matplotlib [12] to create graphs. This process can be especially time, space and energy consuming when there is a repository of many data files to be dealt with. Furthermore this kind of approach requires familiarity with software development, which reduces the accessibility of the data for researchers and students lacking experience in programming. With a web-based tool like NeoViewer, all that is needed is a URL pointing to the file.

The web-based architecture also makes it simple to integrate the visualizer as a sub-component of other applications. It can be a highly useful part in interactive research where the ongoing process of work must be displayed graphically. A scientist sharing the current state of their research or multiple scientists working in collaboration can greatly benefit from its modular and web-based approach. This is already demonstrated in the interactive EBRAINS Live Papers, where it provides much needed interactivity in presenting scientific data.

5. Conclusions

While much scientific software is concerned with data acquisition, generation or analysis, software to facilitate other phases of the research lifecycle, such as data sharing and reuse, can also greatly accelerate research progress. NeoViewer aims to address the needs of neuroscientists working with data extracted from electrophysiology experiments. It copes with the incompatibility issues of different file formats by using the Neo API which supports a large number of file types encountered in the field of neuroscience. With its web based architecture it provides a lightweight and portable visualization tool, thereby obviating the need to locally download data files and install a suitable software environment before visualizing and exploring datasets. In summary, NeoViewer is a fast and efficient plotting library for interactive visualization of electrophysiological data with support for a wide range of file formats, thereby contributing to interoperability in neuroscience.

CRediT authorship contribution statement

Onur Ates: Writing – review & editing, Writing – original draft, Software. **Shailesh Appukuttan:** Writing – review & editing, Writing – original draft, Software. **Hélissande Fragnaud:** Software. **Corentin Fragnaud:** Software. **Andrew P. Davison:** Writing – review & editing, Supervision, Software, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Source code is available at https://github.com/neuralensemble/neo-viewer.

Acknowledgments

This open source software code was developed in part or in whole in the Human Brain Project, funded from the European Union's Horizon 2020 Framework Programme for Research and Innovation under Specific Grant Agreements No. 720270, No. 785907 and No. 945539 (Human Brain Project SGA1, SGA2 and SGA3).

References

[1] Poline J-B, Kennedy DN, Sommer FT, Ascoli GA, Van Essen DC, Ferguson AR, et al. Is neuroscience FAIR? A call for collaborative standardisation of neuroscience data. Neuroinformatics 2022;1–6.

[2] Arguillas F, Christian T-M, Gooch M, Honeyman T, Peer L, Wg C-F. 10 Things for Curating Reproducible and FAIR Research. Zenodo; 2022, http://dx.doi.org/10.15497/RDA00074.

[3] Garcia S, Guarino D, Jaillet F, Jennings TR, Pröpper R, Rautenberg PL, et al. Neo: an object model for handling electrophysiology data in multiple formats. Front Neuroinform 2014;8:10.

[4] Teeters JL, Godfrey K, Young R, Dang C, Friedsam C, Wark B, et al. Neurodata Without Borders: creating a common data format for neurophysiology. Neuron 2015;88(4):629–34.

[5] Appukuttan S, Bologna LL, Schürmann F, Migliore M, Davison AP. EBRAINS Live Papers-interactive resource sheets for computational studies in neuroscience. Neuroinformatics 2022;1–13.

[6] Migliore R, Lupascu CA, Bologna LL, Romani A, Courcol J-D, Antonel S, et al. The physiological variability of channel density in hippocampal CA1 pyramidal cells and interneurons explored using a unified data-driven modeling workflow. PLoS Comput Biol 2018;14(9):e1006423.

[7] Plotly Technologies Inc. Collaborative data science. Montreal, QC: Plotly Technologies Inc.; 2015, URL https://plot.ly.

[8] Cantarelli M, Marin B, Quintana A, Earnshaw M, Court R, Gleeson P, et al. Geppetto: A reusable modular open platform for exploring neuroscience data and models. Philos Trans R Soc B 2018;373(1758):20170380.

[9] Gleeson P, Cantarelli M, Marin B, Quintana A, Earnshaw M, Sadeh S, et al. Open Source Brain: A collaborative resource for visualizing, analyzing, simulating, and developing standardized models of neurons and circuits. Neuron 2019;103(3):395–411.

[10] Zaldivar A, Krichmar JL. Allen Brain Atlas-driven visualizations: A web-based gene expression energy visualization tool. Front Neuroinform 2014;8:51.

[11] Spreizer S, Senk J, Rotter S, Diesmann M, Weyers B. NEST Desktop, an educational application for neuroscience. eNeuro 2021;8(6). http://dx.doi.org/10.1523/ENEURO.0274-21.2021.

[12] Hunter JD. Matplotlib: A 2D graphics environment. Comput. Sci. Eng. 2007;9(3):90–5. http://dx.doi.org/10.1109/MCSE.2007.55.