NeoViewer: Visualising Electrophysiology Data

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

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

Abstract

One of the biggest challenges while working with neurophysiology data is the incompatibility of file formats when moving the data from one tool or platform to another. Many data recording tools produce proprietary file formats which are often not compatible with the analysis or visualization tools at hand. NeoViewer is an open-source and free web-based 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, often the first step in understanding a dataset before proceeding to a more in-depth analysis. Also, developers can easily integrate the visualizer within their own web-based tools.

Keywords: neurophysiology data, visualization, data formats

Required Metadata

Current code version

Nr.	Code metadata description	
C1	Current code version	GitHub tag: v2.0
C2	Permanent link to code/repository used for this code version	https://github.com/NeuralEnsemble/neo-viewer
C3	Code Ocean compute capsule	-
C4	Legal Code License	MIT License
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python (Django), JavaScript (Angular, ReactJS)
C7	Compilation requirements, operating environments & dependencies	npm
C8	If available Link to developer documentation/manual	https://neo-viewer.brainsimulation.eu/
C9	Support email for questions	support@ebrains.eu

Table 1: Code metadata

The permanent link to code/repository or the zip archive should include the following requirements:

README.txt and LICENSE.txt.

Source code in a src/ directory, not the root of the repository.

Tag corresponding with the version of the software that is reviewed.

Documentation in the repository in a docs/ directory, and/or READMEs, as appropriate.

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]. Various tools and services have been developed to help assist with this, such as the EBRAINS Live Papers [2] and OpenNeuro [3]. 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 could be significantly aided if these shared datasets were curated in detail prior to sharing ([4]), but that is typically not the case.

It is therefore imperative that researchers have a way to to quickly examine the available data, to help shortlist 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 their repository, find a suitable tool (or develop your own custom script) to load and visualize each data file.

The free and 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 neurophysiological data from within the web browser. Users can simply specify the URL of the target data file, and the visualizer will plot the contained data.

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. The NeoViewer was developed to perfectly fit into this ecosystem, whereby 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 Django 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 [5], 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 [6]. 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; see Fig. 2
- **AnalogSignals**: continuous signal data with a fixed sampling interval (e.g. membrane potentials); see Fig. 3
- **IrregularlySampledSignal**: analog signal with a varying sampling interval (e.g. recordings from simulations using a variable-time-step integration method)

Support for other types, such as 'Events' (e.g. triggers during behavioural experiments) and 'Epochs' (e.g. the time during which a stimulus is presented) is planned. Another upcoming feature will be to support visualising the newly implemented 'ImageSequence' type which is used for data produced by functional microscopy such as calcium imaging or voltage-sensitive dye imaging.

2.2. Software Functionalities

NeoViewer can be easily integrated in an HTML document. 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 in the same 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 colours. Removing and replacing individual traces in a plot takes a single mouse click. Any plot can be downloaded as a PNG image file.

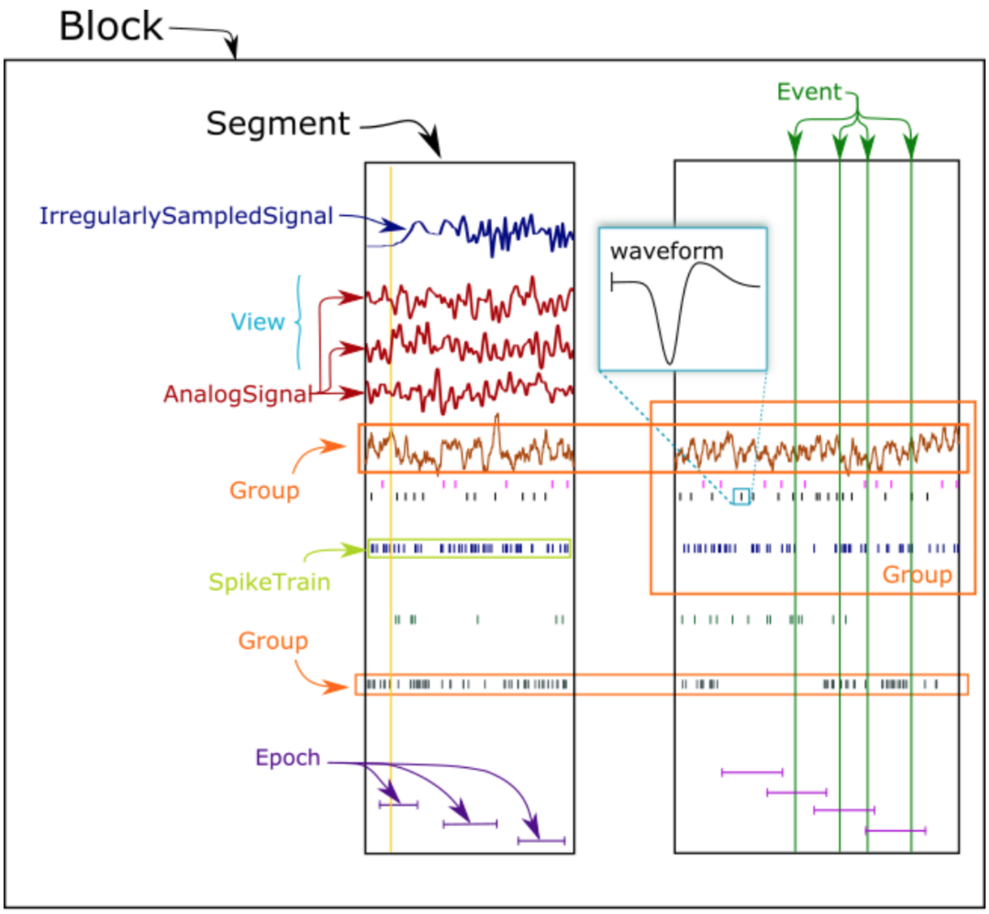

Figure 1: Illustration of different types of data objects handled by Neo.

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 improve 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.

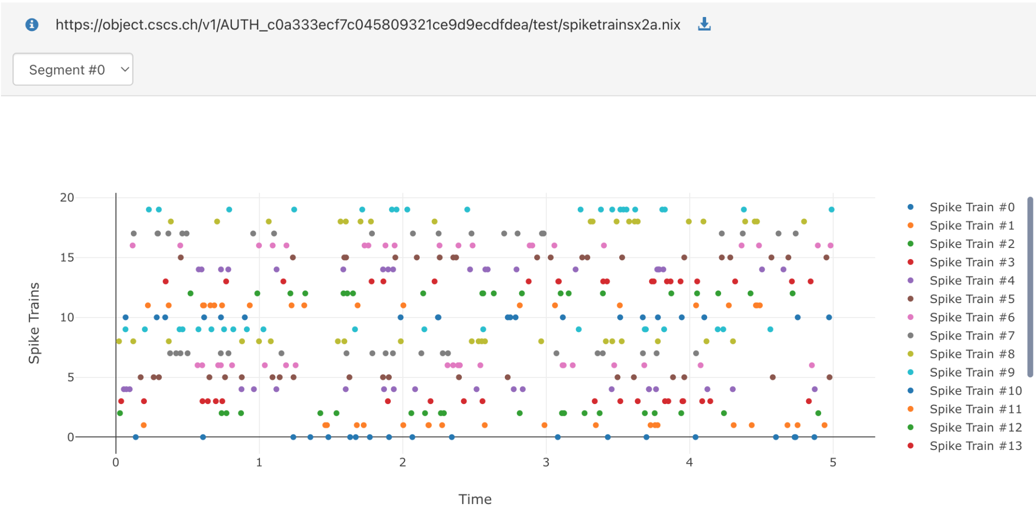

Figure 2: Screenshot of all analog signals in a segment being plotted simultaneously using the ReactJS implementation of the tool.

2.3. Sample code snippets analysis

The REST API can be deployed in a standalone web server (Docker image available) or added to an existing Django project. 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.

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

Listing 1: Basic Usage of NeoViewer using ReactJS

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 2 of the 15th segment should be displayed.

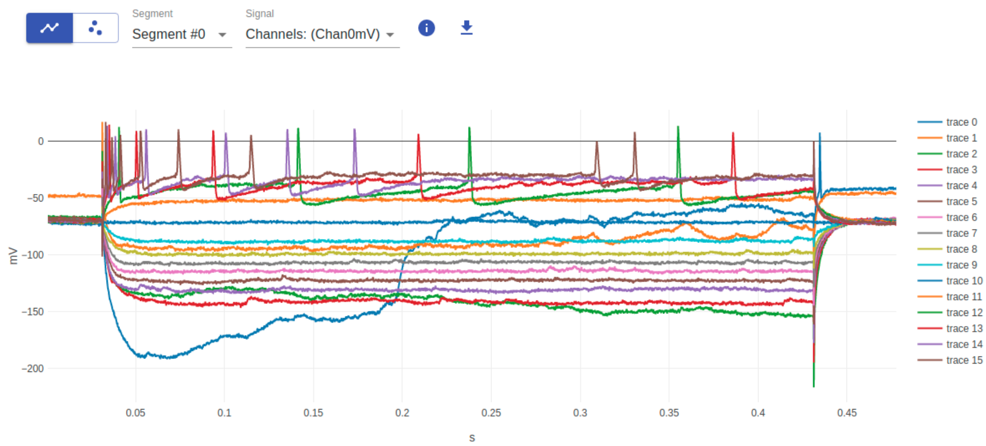

Figure 3: Screenshot of all analog signals in a segment being plotted simultaneously using the ReactJS implementation of the tool.

```
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

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

https://neo-viewer.brainsimulation.eu/angularjs

3. Illustrative Examples

The EBRAINS live paper platform extensively uses NeoViewer to visualize experimental data from published studies [2]. 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 corresponding to Migliore et al. [7]. This can be accessed at the following URL:

https://live-papers.brainsimulation.eu/#2018-migliore-et-al

Their study involved the development of biophysically realistic models of hippocampal neurons. The experimental data was extracted from several electrophysiological recordings. The authors shared these data on their Live Paper, and the NeoViewer enables them to provide visualization from within the web pages. Fig. 4 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 void. NeoViewer is designed to provide a simple, fast and smooth user experience to researchers. It achieves this in manifold ways. First and foremost, NeoViewer is built on top of the widely used file standardisation tool called 'Neo' [5], 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 [8] 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 economize on resources. (iii) NeoViewer provides the feature to optionally increase the sampling period of individual plots

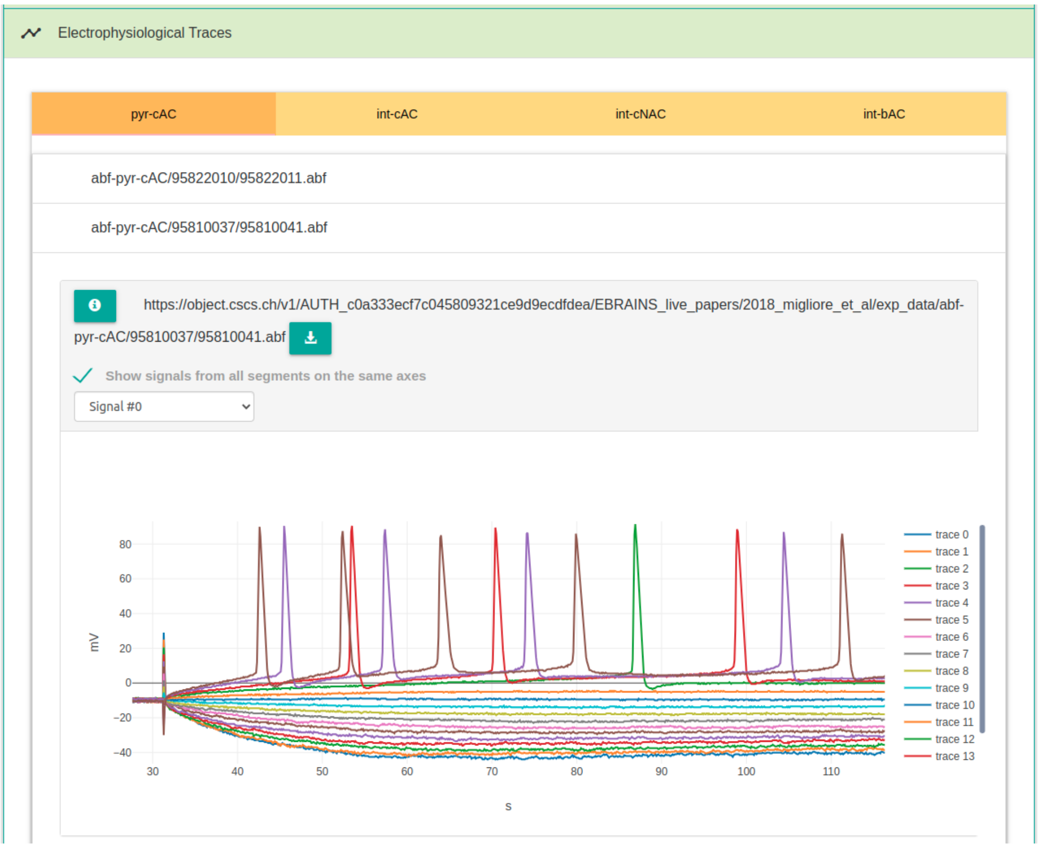

Figure 4: Example usage of NeoViewer inside EBRAINS Live Papers.

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 visualisation 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[9] (with OSB extension) employed in Open Source Brain [10], Allen Brain Atlas-Driven Visualizations [11] and NEST Desktop [12] are competent tools but tightly coupled to the services they were developed for, whereas NeoViewer can be easily embedded in any web-based project.

Web-based visualisation makes it possible to skip the whole process of downloading data to a local computer and creating a necessary environment to run and visualise it. In its absence, a common approach, for example, using Python programming language would be to create a proper software environment locally by first identifying the relevant libraries (packages) and installing them, loading the data and then use a visualisation tool like Matplotlib [13] to create various 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 makes it also simple to integrate and be 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 integrates effortlessly into these *live* publications and provides much needed interactivity in presenting scientific data.

NeoViewer is developed under the Human Brain Project and hosted with the EBRAINS infrastructure. It is currently primarily used by scientists within the project's scope who wish to share scientific data. It is a good choice for neuroscientists who provide and use web services (e.g. EBRAINS). Its lightweight and portable nature makes it a good choice for any scientist working with electrophysiology data.

5. Conclusions

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 visualisation tool, thereby allowing to skip the process of locally downloading data files and generating their plots.

In summary, NeoViewer is a fast and easy-to-use plotting library for interactive visualization of electrophysiological data with support for a wide range of file formats, thereby contributing to interoperability in neuroscience.

6. Conflict of Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

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] J.-B. Poline, D. N. Kennedy, F. T. Sommer, G. A. Ascoli, D. C. Van Essen, A. R. Ferguson, J. S. Grethe, M. J. Hawrylycz, P. M. Thompson, R. A. Poldrack, et al., Is neuroscience fair? a call for collaborative standardisation of neuroscience data, Neuroinformatics (2022) 1–6.

[2] S. Appukuttan, L. L. Bologna, F. Schürmann, M. Migliore, A. P. Davison, Ebrains live papers-interactive resource sheets for computational studies in neuroscience, Neuroinformatics (2022) 1–13.

[3] C. J. Markiewicz, K. J. Gorgolewski, F. Feingold, R. Blair, Y. O. Halchenko, E. Miller, N. Hardcastle, J. Wexler, O. Esteban, M. Goncavles, et al., The openneuro resource for sharing of neuroscience data, Elife 10 (2021) e71774.

[4] F. Arguillas, T.-M. Christian, M. Gooch, T. Honeyman, L. Peer, C.-F. WG, 10 Things for Curating Reproducible and FAIR Research (Jun. 2022). `doi:10.15497/RDA00074`.
URL https://doi.org/10.15497/RDA00074

[5] S. Garcia, D. Guarino, F. Jaillet, T. R. Jennings, R. Pröpper, P. L. Rautenberg, C. Rodgers, A. Sobolev, T. Wachtler, P. Yger, et al., Neo: an object model for handling electrophysiology data in multiple formats, Frontiers in neuroinformatics 8 (2014) 10.

[6] J. L. Teeters, K. Godfrey, R. Young, C. Dang, C. Friedsam, B. Wark, H. Asari, S. Peron, N. Li, A. Peyrache, et al., Neurodata without borders: creating a common data format for neurophysiology, Neuron 88 (4) (2015) 629–634.

[7] R. Migliore, C. A. Lupascu, L. L. Bologna, A. Romani, J.-D. Courcol, S. Antonel, W. A. Van Geit, A. M. Thomson, A. Mercer, S. Lange, et al., The physiological variability of channel density in hippocampal ca1 pyramidal cells and interneurons explored using a unified data-driven modeling workflow, PLoS computational biology 14 (9) (2018) e1006423.

[8] P. T. Inc., Collaborative data science (2015).
URL https://plot.ly

[9] M. Cantarelli, B. Marin, A. Quintana, M. Earnshaw, R. Court, P. Gleeson, S. Dura-Bernal, R. A. Silver, G. Idili, Geppetto: a reusable modular open platform for exploring neuroscience data and models, Philosophical Transactions of the Royal Society B: Biological Sciences 373 (1758) (2018) 20170380.

[10] P. Gleeson, M. Cantarelli, B. Marin, A. Quintana, M. Earnshaw, S. Sadeh, E. Piasini, J. Birgiolas, R. C. Cannon, N. A. Cayco-Gajic, et al., Open source brain: a collaborative resource for visualizing, analyzing, simulating, and developing standardized models of neurons and circuits, Neuron 103 (3) (2019) 395–411.

[11] A. Zaldivar, J. L. Krichmar, Allen brain atlas-driven visualizations: a web-based gene expression energy visualization tool, Frontiers in neuroinformatics 8 (2014) 51.

[12] S. Spreizer, J. Senk, S. Rotter, M. Diesmann, B. Weyers, Nest desktop, an educational application for neuroscience, eNeuro 8 (6) (2021). arXiv:https://www.eneuro.org/content/8/6/ENEURO.0274-21.2021.full.pdf, doi:10.1523/ENEURO.0274-21.2021.
URL https://www.eneuro.org/content/8/6/ENEURO.0274-21.2021

[13] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science Engineering 9 (3) (2007) 90–95. doi:10.1109/MCSE.2007.55.

Please add the reference to the software repository if DOI for software is available.

Current executable software version

Ancillary data table required for sub version of the executable software: (x.1, x.2 etc.) kindly replace examples in right column with the correct information about your executables, and leave the left column as it is.

Nr.	(Executable) software metadata description	Please fill in this column
S1	Current software version	v2.0
S2	Permanent link to executables of this version	https://github.com/NeuralEnsemble/neo-viewer/releases/tag/2.0
S3	Legal Software License	MIT License
S4	Computing platforms/Operating Systems	web based
S5	Installation requirements & dependencies	Node.js
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	https://neo-viewer.brainsimulation.eu/
S7	Support email for questions	support@ebrains.eu

Table 2: Software metadata (optional)