Pluvianus: CaImAn Result Browser Graphical Interface

Authors

1. KATONA, Gergely \*
2. DÁVID, András
3. SLÉZIA, Andrea
4. KASZÁS, Attila

Author Roles and Affiliations

HUN-REN RCNS, Budapest, Hungary

Abstract

Pluvianus is a graphical user interface designed for the visual exploration and quality control of calcium imaging results produced by CaImAn’s CNMF and OnACID algorithms. Developed entirely in Python and structured as a stand-alone Qt 6 desktop application, Pluvianus provides researchers with synchronized spatial and temporal views, intuitive visual comparison tools, and interactive component acceptance. It imposes no constraints on the underlying computational workflow or data organization, enhancing usability and integration flexibility. Publicly distributed on GitHub under the MIT license, Pluvianus supports seamless adoption and community-driven development, making it broadly usable for neuroscientists working with calcium imaging data across diverse experimental paradigms.

Keywords

CaImAn; GUI; neuroscience; calcium imaging; two-photon microscopy;

Introduction

Two-photon calcium imaging has become a workhorse technique for monitoring neuronal activity in the living brain at cellular resolution, producing terabyte-scale datasets (Bae et al., 2025) that follow tens of thousands of neurons over time spans from milliseconds to months. Although a decade of methodological innovation has aimed to extract meaningful cellular activity signals from these imaging data, the reliable extraction, curation, and validation of neuronal traces remains a major bottleneck in routine laboratories (Cai et al., 2022).

Early efforts such as SIMA introduced the first open-source toolkit for motion correction and region-of-interest segmentation in these dynamic fluorescence recordings (Kaifosh et al., 2014), but were quickly surpassed by newer algorithms that excelled in the Neurofinder (<http://neurofinder.codeneuro.org> ) challenge. Still, the search for better performing, more reliable and widely applicable solutions is ongoing (Mi et al., 2025).

Two mature implementations now dominate everyday practice: Suite2p, which utilizes SVD decomposition (Pachitariu et al., 2016), and CaImAn, built around constrained non-negative matrix factorization (CNMF) that enable demixing of overlapping neuronal sources (Giovannucci et al., 2019). Both algorithms reliably achieve human-level accuracy on standard two-photon imaging datasets, yet bibliometric analyses show CaImAn surpassing 400 citations, suggesting broader adoption in the community.

Deep-learning methods span a spectrum—from spatiotemporal 3D CNNs (STNeuroNet;(Soltanian-Zadeh et al., 2019)), through shallow U-Nets optimised for speed (SUNS; (Bao et al., 2021)) and semi-supervised U-Net pipelines that slash the number of required ground-truth labels (Baker & Gong, 2024), to online engines such as CITE-On that couple segmentation with real-time trace extraction (Sità et al., 2022). Despite impressive benchmark scores, none has yet become routine: for some, high computational demands or limited reproducibility pose barriers; for others, the lack of annotated training data or ongoing software support limits practical adoption. These practical constraints have kept CNMF-based pipelines the de-facto standard in most experimental laboratories.

Recognizing the steep learning curve of CaImAn, a graphical user interface (GUI) would be highly beneficial. However, CaImAn’s existing GUI is considered experimental and remains limited by its implementation within the Jupyter notebook framework (<https://caiman.readthedocs.io/en/latest/GUI.html>). Several groups have wrapped the library in convenience GUIs tailored to specific experimental paradigms or analysis workflows—e.g. Mesmerize for workflow organization (Kolar et al., 2021), EZcalcium for simple rodent sensory cortex recordings (Cantu et al., 2020), and CalciumZero for iPSC-derived brain organoids (He et al., 2025). While these front-ends lower entry barriers, their tight coupling to niche paradigms compromises CaImAn’s hallmark versatility; uptake has therefore been modest. Workflow integrators such as NeuroWRAP embed CaImAn and Suite2p into containerized pipelines but focus on batch reproducibility rather than interactive quality control (Bowen et al., 2023).

Pluvianus was developed to visualize the output of CaImAn, providing users with intuitive insight into what the analyzed data actually looks like—without sacrificing the package’s flexibility valued by expert users. It serves both as a tool for power users to fine-tune and validate algorithmic performance, and as a natural second step for newcomers after running the CaImAn demo notebooks—offering a visual, code-free way to explore results and build understanding before moving on to custom analyses. Unlike previous wrapping attempts that embed custom pre-processing steps or enforce rigid folder hierarchies, Pluvianus deliberately focuses solely on post-hoc inspection and validation. It imposes no constraints on the computational engine or the user’s file organization, which remain entirely untouched.

**Implementation and architecture**

The software package is implemented entirely in Python with minimal additional dependencies beyond the core CaImAn suite. It provides a standalone, Qt 6-based desktop application designed specifically for visual quality control, seamlessly opening native \*.hdf5 result files generated by CaImAn’s CNMF or OnACID algorithms.

The GUI (Figure 1) combines synchronized spatial and temporal views with an interactive scatter-plot of component metrics, allowing users to click-select outliers, adjust component acceptance thresholds, or toggle component acceptance individually. The two spatial panels can cycle through the raw movie, CNMF reconstruction, residuals, component footprints, and summary images (mean, median, correlation), while temporal panels may overlay raw fluorescence, temporal components, ΔF/F traces, and residuals for instant comparison. Currently the software handles only 2D movies.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 1

Main GUI of Pluvianus, showing temporal and spatial panels, and scatter plot of component evaluation metric, all synchronized to the selected component and timepoint.

The Compute menu provides algorithms (including several that invoke CaImAn functions) for calculating missing component evaluations, ΔF/F estimations, or – for comparison – the raw fluorescent traces. After inspecting the results, the modified threshold levels for component evaluation and classification state of the components can be saved back to the .hdf5 file for subsequent pipeline steps. The final activity traces can also be saved to pynapple .npy files.

Example Use Cases

Here we present several use-case scenarios that can be performed with Pluvianus.

Verifying signal-to-noise separation

Load the .hdf5 file of your CaImAn result and its data array (.mmap file). Compute both the original fluorescence traces and the ΔF/F traces in the Compute menu if not previously computed. Display the raw and ΔF/F traces side-by-side on the temporal widget. While these curves are not directly comparable in highly complex or overlapping scenarios, in most cases, this view allows users to assess whether the algorithm has properly subtracted baseline fluctuations. Additionally, by displaying each component with identical y-axis scaling, users can more readily determine which components stand out from the noise, and which represent only overfitting of the noise. At cases of interest, you can drag the time axis to the particular period of activity, and you can inspect both the component’s and the surrounding regions’ activity in the spatial widget. To suppress noise, you may average the data both spatially and temporally.

Inspecting component’s highest activity

Check the “Auto” option in both the temporal and spatial widgets to automatically zoom in on a newly selected component, enhancing the visibility of its features. This makes the time axis centered on the period of maximal activity for the given component. By loading the raw data into the spatial widget, one can directly inspect whether the component’s footprint matches the expected active morphology of a cell—confirming, for example, whether the fluorescence event is truly localized to that region.

Assessing completeness of component extraction

Computing the three maximum residual images allows you to assess whether CaImAn’s algorithm performed correctly, and if calculation parameters were optimal. You may display two of these three images at the same time in the spatial widgets to compare:

1. Maximum of residuals after background subtraction (MaxResNone, Y- BG): This image highlights areas of highest activity after background subtraction. Robust activity patches should be delineated as good components.
2. Maximum of residuals using good components (MaxResGood, Y – BG – RCM,good): In this image, all "good" components have been subtracted. Ideally the previously identified robust components should no longer be visible. This step allows users to confirm that strong, well-separated components have been fully removed and to examine cases of overlap or incomplete subtraction. It also provides insight into how much activity remains associated with components previously labelled as "bad".
3. Maximum of residuals using all components (MaxResAll, Y – BG – RCM,all): Here, all components—regardless of their classification—have been subtracted. At this stage, ideally little or no structured activity should remain. If substantial activity persists, especially if remaining activity patches exceed the typical strength of rejected components, further adjustment of the CaImAn parameters (e.g., increasing the number of components (K), overlap between neighboring patches (stride)) is necessary.

If a local maximum in the residual image needs further analysis, its temporal coordinate (index) can be identified with the cursor hovering over the feature. Around this timepoint, the user can compare raw data, reconstructed activity, and the residual movie, optionally displaying two of these views side by side in the spatial widgets.

Component Evaluation Using Thresholds and Manual Review

Users can review the acceptance criteria for individual components. By rotating the scatter plot and selecting borderline cases, thresholds can be refined to optimize classification. Upon changing the thresholds, the good/bad classification can be recalculated with the Evaluate button. For manual correction, users can iterate through components in various orders (e.g. from the highest SNR to the lowest SNR) and adjust classification manually using keyboard shortcuts (b, g, up and down arrows). The modified, user-validated classifications can be written back to the CaImAn .hdf5 file to support downstream analysis within the user’s pipeline.

**Quality control**

Pluvianus has been manually tested through interactive use of its graphical user interface. Functional correctness was verified by visually inspecting CaImAn's official demo datasets, as well as through extensive practical use on our own in-house dataset—a mouse visual cortex orientation-tuning map database analyzed using CaImAn. No automated unit or integration tests have been implemented at this stage. Detailed step-by-step instructions for installation, launching Pluvianus, and loading the result of CaImAn demo workflows are provided in the project's README file on GitHub, enabling users to quickly verify correct installation and operation.

**Availability**

***Operating system***

Windows, Linux and Mac

***Programming language***

Python 3.10

***Additional system requirements***

See requirements for CaImAn: <https://caiman.readthedocs.io/en/latest/Installation.html>

***Dependencies***

Environment with CaImAn set up, see: <https://caiman.readthedocs.io/en/latest/Installation.html>

Additionally, the PySide6, pyqtgraph, pynapple packages.

***List of contributors***

Gergely Katona, researcher at HUN-REN RCNS

András Dávid, student at BME

***Software location:***

***Archive***

***Name:*** The name of the archive

***Persistent identifier:*** e.g. DOI, handle, PURL, etc.

***Licence:*** MIT License

***Publisher:*** Name of the person who deposited the software

***Version published:***The version number of the software archived

***Date published:*** dd/mm/yy

**Code repository:** GitHub

***Name:*** Pluvianus

***Identifier:*** <https://github.com/katonage/pluvianus>

***Licence:*** MIT License

***Date published:*** dd/mm/yy

***Language***

English

**Reuse potential**

CaImAn is designed to analyze calcium imaging datasets. Pluvianus complements CaImAn’s scalable source-extraction algorithms with a modern, researcher-friendly visualization GUI, without imposing constraints on how analyses are performed or how files are organized, thus preserving CaImAn's exceptional adaptability. This design ensures compatibility with all existing CaImAn-based applications, enabling Pluvianus to be easily integrated wherever researchers require an interactive workflow or the ability to inspect intermediate results during pipeline development. Within neuroscience, any laboratory already utilizing CaImAn’s CNMF or OnACID pipelines can seamlessly add Pluvianus to their existing conda environment and immediately browse native \*.hdf5 output files, independent of the specific imaging modality or experimental setup. Pluvianus is publicly distributed on GitHub under the permissive MIT license, allowing researchers to freely reuse, modify, and extend its functionality to fully leverage CaImAn's extensive feature set. Community contributions from interested researchers are anticipated and welcomed via GitHub issues, discussions following standard GitHub workflows.

**Acknowledgements**

**Funding statement**

**Competing interests**

The authors declare that they have no competing interests.

**References**

Bae, J. A., Baptiste, M., Baptiste, M. R., Bishop, C. A., Bodor, A. L., Brittain, D., Brooks, V., Buchanan, J., Bumbarger, D. J., Castro, M. A., Celii, B., Cobos, E., Collman, F., da Costa, N. M., Danskin, B., Dorkenwald, S., Elabbady, L., Fahey, P. G., Fliss, T., … Zhang, C. (2025). Functional connectomics spanning multiple areas of mouse visual cortex. Nature, 640(8058), 435–447. <https://doi.org/10.1038/s41586-025-08790-w>

Baker, C. M., & Gong, Y. (2024). A Semi-supervised Pipeline for Accurate Neuron Segmentation with Fewer Ground Truth Labels. Eneuro, 11(2), ENEURO.0352-23.2024. <https://doi.org/10.1523/ENEURO.0352-23.2024>

Bao, Y., Soltanian-Zadeh, S., Farsiu, S., & Gong, Y. (2021). Segmentation of neurons from fluorescence calcium recordings beyond real time. Nature Machine Intelligence, 3(7), 590–600. <https://doi.org/10.1038/s42256-021-00342-x>

Bowen, Z., Magnusson, G., Diep, M., Ayyangar, U., Smirnov, A., Kanold, P. O., & Losert, W. (2023). NeuroWRAP: integrating, validating, and sharing neurodata analysis workflows. Frontiers in Neuroinformatics, 17. <https://doi.org/10.3389/fninf.2023.1082111>

Cai, Y., Wu, J., & Dai, Q. (2022). Review on data analysis methods for mesoscale neural imaging in vivo. Neurophotonics, 9(04). <https://doi.org/10.1117/1.NPh.9.4.041407>

Cantu, D. A., Wang, B., Gongwer, M. W., He, C. X., Goel, A., Suresh, A., Kourdougli, N., Arroyo, E. D., Zeiger, W., & Portera-Cailliau, C. (2020). EZcalcium: Open-Source Toolbox for Analysis of Calcium Imaging Data. Frontiers in Neural Circuits, 14. <https://doi.org/10.3389/fncir.2020.00025>

Giovannucci, A., Friedrich, J., Gunn, P., Kalfon, J., Brown, B. L., Koay, S. A., Taxidis, J., Najafi, F., Gauthier, J. L., Zhou, P., Khakh, B. S., Tank, D. W., Chklovskii, D. B., & Pnevmatikakis, E. A. (2019). CaImAn an open source tool for scalable calcium imaging data analysis. ELife, 8. <https://doi.org/10.7554/eLife.38173>

He, X., Wang, Y., Gao, Y., Wang, X., Sun, Z., Zhu, H., Leong, K. W., & Xu, B. (2025). CalciumZero: a toolbox for fluorescence calcium imaging on iPSC derived brain organoids. Brain Informatics, 12(1), 2. <https://doi.org/10.1186/s40708-024-00248-5>

Kaifosh, P., Zaremba, J. D., Danielson, N. B., & Losonczy, A. (2014). SIMA: Python software for analysis of dynamic fluorescence imaging data. Frontiers in Neuroinformatics, 8. <https://doi.org/10.3389/fninf.2014.00080>

Kolar, K., Dondorp, D., Zwiggelaar, J. C., Høyer, J., & Chatzigeorgiou, M. (2021). Mesmerize is a dynamically adaptable user-friendly analysis platform for 2D and 3D calcium imaging data. Nature Communications, 12(1), 6569. <https://doi.org/10.1038/s41467-021-26550-y>

Mi, X., Chen, A. B.-Y., Duarte, D., Carey, E., Taylor, C. R., Braaker, P. N., Bright, M., Almeida, R. G., Lim, J.-X., Ruetten, V. M. S., Wang, Y., Wang, M., Zhang, W., Zheng, W., Reitman, M. E., Huang, Y., Wang, X., Li, L., Deng, H., … Yu, G. (2025). Fast, accurate, and versatile data analysis platform for the quantification of molecular spatiotemporal signals. Cell, 188(10), 2794-2809.e21. <https://doi.org/10.1016/j.cell.2025.03.012>

Pachitariu, M., Stringer, C., Dipoppa, M., Schröder, S., Rossi, L. F., Dalgleish, H., Carandini, M., & Harris, K. D. (2016). Suite2p: beyond 10,000 neurons with standard two-photon microscopy. <https://doi.org/10.1101/061507>

Sità, L., Brondi, M., Lagomarsino de Leon Roig, P., Curreli, S., Panniello, M., Vecchia, D., & Fellin, T. (2022). A deep-learning approach for online cell identification and trace extraction in functional two-photon calcium imaging. Nature Communications, 13(1), 1529. <https://doi.org/10.1038/s41467-022-29180-0>

Soltanian-Zadeh, S., Sahingur, K., Blau, S., Gong, Y., & Farsiu, S. (2019). Fast and robust active neuron segmentation in two-photon calcium imaging using spatiotemporal deep learning. Proceedings of the National Academy of Sciences, 116(17), 8554–8563. <https://doi.org/10.1073/pnas.1812995116>