

# NeuroCAPs: A Python Package for Performing Co-Activation Patterns Analyses on Resting-State and Task-Based fMRI Data

Donisha Smith<sup>1</sup>

<sup>1</sup> Department of Psychology, Florida International University

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

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

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

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

## Summary

Co-Activation Patterns (CAPs) is a dynamic functional connectivity technique that clusters similar spatial distributions of brain activity. To make this analytical technique more accessible to neuroimaging researchers, NeuroCAPs ([D. Smith, 2025](#)), an open source Python package, was developed. This package performs end-to-end CAPs analyses on preprocessed resting-state or task-based functional magnetic resonance imaging (fMRI) data, and is most optimized for data preprocessed with fMRIPrep, a robust preprocessing pipeline designed to minimize manual user input and enhance reproducibility ([Esteban et al., 2019](#)).

## Background

Numerous fMRI studies employ static functional connectivity (sFC) techniques to analyze correlative activity within and between brain regions. However, these approaches operate under the assumption that functional connectivity patterns, which change within seconds ([Jiang et al., 2022](#)), remain stationary throughout the entire data acquisition period ([Hutchison et al., 2013](#)).

Unlike sFC approaches, dynamic functional connectivity (dFC) methods enable the analysis of dynamic functional states, which are characterized by consistent, replicable, and distinct periods of time-varying brain connectivity patterns ([Rabany et al., 2019](#)). Among these techniques, CAPs analysis aggregates similar spatial distributions of brain activity using clustering techniques, such as the k-means algorithm, to capture the dynamic nature of brain activity ([Liu et al., 2018](#)).

## Statement of Need

Currently, performing an end-to-end CAPs analysis presents challenges due to the numerous steps required. Researchers must:

1. clean timeseries data through nuisance regression and censor frames with high framewise displacement (excessive head motion).
2. perform spatial dimensionality reduction.
3. concatenate timeseries data across multiple subjects for k-means clustering.
4. select an optimal cluster size using heuristics such as the elbow or silhouette methods.
5. implement various visualization techniques to enhance interpretability of the results.

While other excellent CAPs toolboxes exist, they are often implemented in proprietary languages such as MATLAB (which is the case for TbCAPs ([Bolton et al., 2020](#))), lack comprehensive end-to-end analytical pipelines for both resting-state and task-based fMRI data with temporal

dynamic metrics and visualization capabilities (such as `capcalc` (Frederick & Drucker, 2022)), or are comprehensive, but generalized toolboxes for evaluating and comparing different dFC methods (such as `pydFC` (Torabi et al., 2024)). NeuroCAPs addresses these limitations by providing an accessible Python package specifically for performing end-to-end CAPs analyses, from post-processing of fMRI data to creation of temporal metrics for downstream statistical analyses and visualizations to facilitate interpretations. However, many of NeuroCAPs' post-processing functionalities assumes that fMRI data is organized in a Brain Imaging Data Structure (BIDS) compliant directory and is most optimized for data preprocessed with fMRIPrep (Esteban et al., 2019) or preprocessing pipelines that generate similar outputs (e.g. NiBabies (Goncalves et al., 2025)).

## Modules

NeuroCAPs consists of four modules, with core functionality primarily distributed between two main modules (`neurocaps.extraction` and `neurocaps.analysis`) that handle the entire workflow, from post-processing to temporal metric computation and visualization capabilities.

### `neurocaps.exceptions`

This module contains two custom exceptions: `BIDSQueryError` and `NoElbowDetected`. `BIDSQueryError` supports NeuroCAPs' integration with PyBIDS (Yarkoni et al., 2019), a tool for querying BIDS-compliant directories, providing user support guidance when subject IDs cannot be detected in BIDS directories. `NoElbowDetected` offers solutions when the elbow method (implemented via `KneeLocator` from `Kneed` (Arvai, 2023)) fails to identify the optimal cluster size for k-means.

### `neurocaps.extraction`

This module contains the `TimeseriesExtractor` class, which:

- leverages `Nilearn`'s ([contributors, n.d.](#)) `NiftiLabelsMasker` to perform nuisance regression on resting-state and task-based fMRI data and use lateralized brain parcellations (such as the Schaefer (Schaefer et al., 2018), Automated Anatomical Labeling (Tzourio-Mazoyer et al., 2002), and Human Connectome Project extended (Huang et al., 2022) parcellations) for spatial dimensionality reduction.
- censors high-motion volumes using fMRIPrep-derived framewise displacement values and stores the extracted timeseries information in a dictionary mapping subject IDs to run IDs and their associated timeseries.
- reports quality control information for framewise displacement.
- saves extracted timeseries data as a pickle file.
- visualizes timeseries data for a specific subject's run.

### `neurocaps.analysis`

This module contains the `CAP` class, which:

- allows group-specific analyses or analyses on all subjects.
- performs k-means clustering (from `Scikit-learn` (Pedregosa et al., 2011)) for CAP identification, while supporting a single cluster size or a range of clusters in combination with various cluster selection methods to determine the optimal cluster size.
- computes various subject-level temporal dynamics metrics for downstream statistical analyses.
- enables conversion of CAPs to NIFTI statistical maps.
- provides diverse visualization options, using `Matplotlib` (Hunter, 2007), `Seaborn` (Waskom, 2021), `Plotly` (Inc., n.d.), and `Surfplot` (Gale et al., 2021), to facilitate scientific interpretations.

Additionally, the module provides standalone functions for:

- 85     ▪ changing the data type (Harris et al., 2020) and performing additional standardization
- 86     of timeseries data produced by TimeseriesExtractor.
- 87     ▪ merging multiple timeseries data across different dictionaries produced by
- 88     TimeseriesExtractor by identifying similar subjects and concatenating their
- 89     data, which facilitates analyses to identify CAPs across sessions or tasks.
- 90     ▪ creating averaged transition probability matrices from subject-level transition probabilities.

## 91 neurocaps.typing

92 This module provides custom type definitions compatible with static type checkers, enabling  
93 proper construction of dictionary structures for parcellations or timeseries data when manual  
94 creation is necessary.

## 95 Example Application

96 NeuroCAPs was originally developed (and later refined for broader use) to facilitate the analysis  
97 in D. D. Smith et al. (2025), which has been submitted for peer review by the same author.  
98 In this manuscript, NeuroCAPs was used to extract timeseries data, cluster and identify CAPs  
99 using the elbow method, compute temporal metrics for downstream statistical analyses, and  
100 produce visualizations for CAPs (i.e., heatmap, surface plots, correlation matrix, and radar  
101 plots).

## 102 Acknowledgements

103 Funding provided by the Dissertation Year Fellowship (DYF) Program at Florida International  
104 University (FIU), assisted in further refinement of NeuroCAPs.

## 105 References

- 106 Arvai, K. (2023). *Kneed*. Zenodo. <https://doi.org/10.5281/ZENODO.8127224>
- 107 Bolton, T. A. W., Tuleasca, C., Wotruba, D., Rey, G., Dhanis, H., Gauthier, B., Delavari,  
108 F., Morgenroth, E., Gaviria, J., Blondiaux, E., Smigielski, L., & Van De Ville, D. (2020).  
109 TbCAPs: A toolbox for co-activation pattern analysis. *NeuroImage*, 211, 116621. <https://doi.org/10.1016/j.neuroimage.2020.116621>
- 110 contributors, N. (n.d.). *nilearn*. <https://doi.org/10.5281/zenodo.8397156>
- 111 Esteban, O., Markiewicz, C. J., Blair, R. W., Moodie, C. A., Isik, A. I., Erramuzpe, A., Kent,  
112 J. D., Goncalves, M., DuPre, E., Snyder, M., Oya, H., Ghosh, S. S., Wright, J., Durnez,  
113 J., Poldrack, R. A., & Gorgolewski, K. J. (2019). fMRIprep: A robust preprocessing  
114 pipeline for functional MRI. *Nature Methods*, 16(1), 111–116. <https://doi.org/10.1038/s41592-018-0235-4>
- 115 Frederick, B. D., & Drucker, D. M. (2022). *Bbfrederick/capcalc: Version 1.2.2.2 - 8/30/22*  
116 *deployment bug fix*. Zenodo. <https://doi.org/10.5281/ZENODO.7035806>
- 117 Gale, D. J., Vos de Wael, R., Benkarim, O., & Bernhardt, B. (2021). *Surfplot: Publication-*  
118 *ready brain surface figures*. Zenodo. <https://doi.org/10.5281/ZENODO.5567926>
- 119 Goncalves, M., Markiewicz, C. J., Esteban, O., Feczko, E., Poldrack, R. A., & Fair, D. A.  
120 (2025). *NiBabies: A robust preprocessing pipeline for infant functional MRI*. Zenodo.  
121 <https://doi.org/10.5281/ZENODO.14811979>
- 122 Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D.,  
123 Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk,  
124 M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., ... Oliphant,

- 127 T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- 128
- 129 Huang, C.-C., Rolls, E. T., Feng, J., & Lin, C.-P. (2022). An extended human connectome  
130 project multimodal parcellation atlas of the human cortex and subcortical areas. *Brain*  
131 *Structure and Function*, 227(3), 763–778. <https://doi.org/10.1007/s00429-021-02421-6>
- 132 Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science &*  
133 *Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- 134 Hutchison, R. M., Womelsdorf, T., Allen, E. A., Bandettini, P. A., Calhoun, V. D., Corbetta,  
135 M., Della Penna, S., Duyn, J. H., Glover, G. H., Gonzalez-Castillo, J., Handwerker, D. A.,  
136 Keilholz, S., Kiviniemi, V., Leopold, D. A., De Pasquale, F., Sporns, O., Walter, M., &  
137 Chang, C. (2013). Dynamic functional connectivity: Promise, issues, and interpretations.  
138 *NeuroImage*, 80, 360–378. <https://doi.org/10.1016/j.neuroimage.2013.05.079>
- 139 Inc., P. T. (n.d.). *Chart title*. <https://plotly.com/python/radar-chart/>; Plotly Technologies  
140 Inc.
- 141 Jiang, F., Jin, H., Gao, Y., Xie, X., Cummings, J., Raj, A., & Nagarajan, S. (2022). Time-  
142 varying dynamic network model for dynamic resting state functional connectivity in fMRI  
143 and MEG imaging. *NeuroImage*, 254, 119131. [https://doi.org/10.1016/j.neuroimage.2022.](https://doi.org/10.1016/j.neuroimage.2022.119131)  
144 [119131](https://doi.org/10.1016/j.neuroimage.2022.119131)
- 145 Liu, X., Zhang, N., Chang, C., & Duyn, J. H. (2018). Co-activation patterns in resting-state  
146 fMRI signals. *NeuroImage*, 180, 485–494. [https://doi.org/10.1016/j.neuroimage.2018.01.](https://doi.org/10.1016/j.neuroimage.2018.01.041)  
147 [041](https://doi.org/10.1016/j.neuroimage.2018.01.041)
- 148 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M.,  
149 Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D.,  
150 Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python.  
151 *Journal of Machine Learning Research*, 12, 2825–2830.
- 152 Rabany, L., Brocke, S., Calhoun, V. D., Pittman, B., Corbera, S., Wexler, B. E., Bell, M. D.,  
153 Pelphrey, K., Pearlson, G. D., & Assaf, M. (2019). Dynamic functional connectivity in  
154 schizophrenia and autism spectrum disorder: Convergence, divergence and classification.  
155 *NeuroImage: Clinical*, 24, 101966. <https://doi.org/10.1016/j.nicl.2019.101966>
- 156 Schaefer, A., Kong, R., Gordon, E. M., Laumann, T. O., Zuo, X.-N., Holmes, A. J., Eickhoff,  
157 S. B., & Yeo, B. T. T. (2018). Local-global parcellation of the human cerebral cortex  
158 from intrinsic functional connectivity MRI. *Cerebral Cortex*, 28(9), 3095–3114. <https://doi.org/10.1093/cercor/bhx179>
- 159
- 160 Smith, D. (2025). *NeuroCAPs: A python package for performing co-activation patterns*  
161 *analyses on resting-state and task-based fMRI data: archive*. Zenodo. [https://doi.org/10.](https://doi.org/10.5281/zenodo.15200608)  
162 [5281/zenodo.15200608](https://doi.org/10.5281/zenodo.15200608)
- 163 Smith, D. D., Bartley, J. E., Peraza, J. A., Bottenhorn, K. L., Nomi, J. S., Uddin, L. Q., Riedel,  
164 M. C., Salo, T., Laird, R. W., Pruden, S. M., Sutherland, M. T., Brewe, E., & Laird, A. R.  
165 (2025). *Dynamic reconfiguration of brain coactivation states associated with active and*  
166 *lecture-based learning of university physics*. <https://doi.org/10.1101/2025.02.22.639361>
- 167 Torabi, M., Mitsis, G. D., & Poline, J.-B. (2024). On the variability of dynamic functional  
168 connectivity assessment methods. *GigaScience*, 13, giae009. [https://doi.org/10.1093/](https://doi.org/10.1093/gigascience/giae009)  
169 [gigascience/giae009](https://doi.org/10.1093/gigascience/giae009)
- 170 Tzourio-Mazoyer, N., Landeau, B., Papathanassiou, D., Crivello, F., Etard, O., Delcroix,  
171 N., Mazoyer, B., & Joliot, M. (2002). Automated anatomical labeling of activations in  
172 SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain.  
173 *NeuroImage*, 15(1), 273–289. <https://doi.org/10.1006/nimg.2001.0978>
- 174 Waskom, M. L. (2021). Seaborn: Statistical data visualization. *Journal of Open Source*

175      *Software*, 6(60), 3021. <https://doi.org/10.21105/joss.03021>

176      Yarkoni, T., Markiewicz, C., De La Vega, A., Gorgolewski, K., Salo, T., Halchenko, Y.,  
177      McNamara, Q., DeStasio, K., Poline, J.-B., Petrov, D., Hayot-Sasson, V., Nielson, D.,  
178      Carlin, J., Kiar, G., Whitaker, K., DuPre, E., Wagner, A., Tirrell, L., Jas, M., ... Blair, R.  
179      (2019). PyBIDS: Python tools for BIDS datasets. *Journal of Open Source Software*, 4(40),  
180      1294. <https://doi.org/10.21105/joss.01294>

DRAFT