The Journal of Open Source Software

MultiSim: A Python Toolbox for Simulating Datasets with time resolved multivariate effects

Alex Lepauvre [1,2¶], **Qian Chu** [1,3,4,5], **Lucia Melloni** [1,6,7], **and Peter Zeidman** [8]

1 Neural Circuits, Consciousness and Cognition Research Group, Max Planck Institute for Empirical Aesthetics, Frankfurt am Main, Germany **2** Donders Institute for Brain, Cognition and Behaviour, Radboud University Nijmegen, Nijmegen, The Netherlands **3** Max Planck – University of Toronto Centre for Neural Science and Technology **4** Krembil Brain Institute, Toronto Western Hospital, University Health Network, Toronto, ON, Canada **5** Institute of Biomedical Engineering, University of Toronto, Toronto, ON, Canada **6** Department of Neurology, New York University Grossman School of Medicine, New York, NY, USA **7** Predictive Brain Department, Research Center One Health Ruhr, University Alliance Ruhr, Faculty of Psychology, Ruhr University Bochum, Bochum, Germany **8** Wellcome Centre for Human Neuroimaging, Institute of Neurology, University College London, London, UK ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

In MEG/EEG research, validating analysis pipelines is hampered by the lack of ground-truth neural signals in real data. SimMEG fills this gap by generating realistic, time-locked multivariate effects of known magnitude that you can inject into simulated sensor data. You can then run any pipeline—e.g. decoding, sensor-level statistics, or source estimation—against these datasets to benchmark sensitivity and specificity.

Key benefits include:

- Testing whether your pipeline reliably detects effects of a chosen size.
- Providing demonstrable, reproducible benchmarks for reviewers or collaborators.
- Offering a controlled teaching environment for newcomers.

Below, we describe the rationale (Statement of needs), and the data-generation method (Methods), a hands-on example (Results), and potential extensions (Discussion).

Statement of needs

Multivariate pattern analysis (MVPA) is now routine in cognitive neuroscience for probing how the brain represents information (Haxby et al., 2001; Haynes & Rees, 2006; Kriegeskorte et al., 2008; Poldrack et al., 2009; Ritchie et al., 2019). Applied to high-temporal-resolution electrophysiology signals such as electro and magneto-encephalography (EEG and MEG respectively), decoding techniques reveal the millisecond-by-millisecond unfolding of mental representations (Cichy et al., 2014; Consortium et al., 2025; King et al., 2016; King & Dehaene, 2014; Kok et al., 2017). Strikingly, despite the ubiquity of MVPA techniques, to our knowledge, no method exists to test the sensitivity and specificity of decoding analysis pipelines, nor to estimate, before data collection, how many trials and how many participants are required to detect an effect of a given size.

MultiSim addresses this gap by letting investigators simulate time-resolved multi-channel signals with paramaters matching that of their recording setups, and specify multivariate effects with known timing, spatialization and strength, while controlling channel covariance, sensory noise and between subjects variability (see (**?**) for a minimal working example and (**?**) for a visuallization of the pipeline). Our algorithm produces multi-subject data sets in which ground truth effects are known. By running their pipeline on these data, researchers obtain a direct read-out of its true-positive rate (can it recover the injected effects?) and false-positive rate (does it raise alarms when nothing is present). In addition, our simulator can be used to perform computational power analysis, to determine the number of trials and subjects, by iterating over these parameters.

```
{code} python :label: minimal_example :caption: Creating a TensorMesh using
SimPEG import numpy as np from meeg_simulator import simulate_data X = pd.DataFrame(np.r
1), columns=["face-object"]) # 100 trials, 1 experimental condition t_win =
np.array([[0.2, 0.5]])  # Effect between 200-500 ms effects = [     {"condition":
'face-object',       "windows": [0.1, 0.3],       "effect_size": 0.5     } ]
sims = Simulator(    X, noise_std=0.1, n_channels=64, n_subjects=20,    tmin=-
0.2, tmax=0.8, sfreq=250,    t_win=t_win, effects=effects   ) sim.summary()  #
Should return 20 subjects
```

This toolbox promotes best-practice MVPA by giving researchers a tailored benchmark for their specific experimental designs, a testbed for developing new decoding methods, and a principled way to check that planned studies are properly powered—ultimately enabling more reliable and efficient investigations of brain function.

"'{figure} ./figure1.png label: figure-1

Figure 1. Overview of the simulation and decoding framework **A**. General data parameters for the simulation. Left: `n_channels` corresponds to the number of channels in the montage, with 2 experimental conditions. Middle: `X` is the design matrix (each column is an experimental condition and each row a trial). Right: `ch_cov` is the channel by channel covariance matrix of the data to be simulated. **B**. Minimal simulation example with effects for each experimental condition with large effect size. Left: `effects` is a dictionary that specifies the `"condition"`, time window (`"windows"`) and effect size (`"effect_size"`) of each effect to simulate. The example specifies an effect of category from 0.1 and 0.2 s with an effect size of 4, and an effect of the attention condition from 0.4 to 0.5 with an effect size of 4. Middle: example of the activation of a single channel. Right: resulting decoding accuracy (using a Support vector machine classifier). *C*. Example of simulated effects with a an added gamma kernel to simulate effects with biologically plausible temporal dynamcs. Left: `effects` similar to that of B but with effect size of 0.5 for each condition. Middle: gamma `kernel`, specifying the temporal dynamics of the multivariate effect. Right: Resulting decoding accuracy. **D**. example of simulated data with cross temporal generalization of the category effect. Left: `effects` dictionary specifies two different time windows for the effect of category as a list. Middle: resulting decoding accuracy. Right: Temporal generalization of the decoding.

~~~~
# Code Quality and Documentation

SimMEG is hosted on GitHub. Examples and API documentation are available on the platform
mathematical_details.html). The package is available on Linux, macOS and Windows for Pyt
It can be installed with pip install simMEG. To ensure high code quality, all implementa

# Acknowledgements

# References
```{bibliography}
~~~~

The Journal of Open Source Software

Supplementary

Cichy, R. M., Pantazis, D., & Oliva, A. (2014). Resolving human object recognition in space and time. *Nature Neuroscience*, *17*(3), 455–462.

Consortium, C., Ferrante, O., Gorska-Klimowska, U., Henin, S., Hirschhorn, R., Khalaf, A., Lepauvre, A., Liu, L., Richter, D., Vidal, Y., & others. (2025). Adversarial testing of global neuronal workspace and integrated information theories of consciousness. *Nature*, 1–10.

Haxby, J. V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, *293*(5539), 2425–2430.

Haynes, J.-D., & Rees, G. (2006). Decoding mental states from brain activity in humans. *Nature Reviews Neuroscience*, *7*(7), 523–534.

King, J.-R., & Dehaene, S. (2014). Characterizing the dynamics of mental representations: The temporal generalization method. *Trends in Cognitive Sciences*, *18*(4), 203–210.

King, J.-R., Pescetelli, N., & Dehaene, S. (2016). Brain mechanisms underlying the brief maintenance of seen and unseen sensory information. *Neuron*, *92*(5), 1122–1134.

Kok, P., Mostert, P., & De Lange, F. P. (2017). Prior expectations induce prestimulus sensory templates. *Proceedings of the National Academy of Sciences*, *114*(39), 10473–10478.

Kriegeskorte, N., Mur, M., & Bandettini, P. A. (2008). Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, *2*, 249.

Poldrack, R. A., Halchenko, Y. O., & Hanson, S. J. (2009). Decoding the large-scale structure of brain function by classifying mental states across individuals. *Psychological Science*, *20*(11), 1364–1372.

Ritchie, J. B., Kaplan, D. M., & Klein, C. (2019). Decoding the brain: Neural representation and the limits of multivariate pattern analysis in cognitive neuroscience. *The British Journal for the Philosophy of Science*.