

Spikeometric: Spiking Neural Networks using GLMs and LNPs in Pytorch Geometric

Jakob L. Sønstebo¹ and Mikkel Elle Leppereød^{1,2}

¹ Department of Numerical Analysis and Scientific Computing, Simula Research Laboratory, Oslo, Norway ² Institute of Basic Medical Sciences, University of Oslo, Oslo, Norway

DOI: N/A

Software

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

Editor: [Open Journals](#) ↗

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: 01 January 1970

License

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

Summary

To understand the dynamics of the brain, computational neuroscientists often study smaller scale networks using simple cascade point-process models such as the Linear-Nonlinear-Poisson (LNP) model and the Generalized Linear Model (GLM) ([Gerstner et al., 2014](#); [Meyer et al., 2017](#); [Paninski, 2004](#)).

Stochastic models can give key insights into the behavior of a network on a systems level without explicitly modelling the subcellular mechanisms of each neuron. They lack some biological plausibility on the neuron level but have been shown to enjoy nice convexity properties, which can be fitted to observed spike data ([Paninski, 2004](#)). Traditionally, these are used as encoding models. For example, to study how multi-neuron systems process incoming stimuli ([Pillow et al., 2008](#)). Recently, these models have been used as generative models for inverse problems mapping activity to connectivity. As an example, ([Das & Fiete, 2020](#)) assessed bias and reconstruction errors in this setting.

This software expands on a simulator developed for testing novel reconstruction techniques using methods from the causal inference literature ([Lepperød et al., 2020](#)). It provides tunable generative models in a flexible framework based on PyTorch. The primary use case, so far, has been as a data-generator for inverse problems, but the framework can easily accommodate more complicated models for encoding applications.

Statement of need

Linear non-linear cascade models are much used in computational neuroscience and come in many flavors. Typical examples are SRM, LNP, GLM ([Gerstner, 2008](#); [Gerstner et al., 2014](#); [Meyer et al., 2017](#)). What unites these models is that they all model the spike response of a network through the same cascade-like sequence of steps. At each time step, a neuron receives input from its environment and converts it to a firing rate by a nonlinear function. This firing rate parametrizes a probability distribution from which spikes are drawn. This contrasts hard-threshold-based models, in which spikes are emitted whenever a variable representing the membrane potential exceeds a certain value. Although these models share many of the same underlying principles, no unifying framework currently permits easy implementation and direct comparison. Moreover a fast and convenient data-generation tool is currently lacking.

Implementation

The spikeometric package is a framework for simulating spiking neural networks using linear non-linear cascade models in Python. It is built on the PyTorch Geometric package and uses its powerful graph neural network modules and efficient graph representation. It is designed

to be fast, flexible, and easy to use. Moreover, it's built in a way that accommodates usage or implementation of multiple different models sharing principles from the linear non-linear cascade family of models.

The torch backend makes simulating large networks on a GPU easy, with the extra benefit of having a familiar use pattern, reducing the friction of picking up a new tool. The package relies heavily on PyTorch Geometric (Fey & Lenssen, 2019), with the networks being represented as torch_geometric Data objects and the models extending the MessagePassing base class. The PyTorch Geometric framework is a popular deep learning framework originally designed for Graph Neural Networks (GNNs), a class of neural networks for learning graph-related data (Bronstein et al., 2021). It is the perfect setting for simulating neural networks with tunable parameters, allowing us to formulate the model's equations naturally in terms of vertices and edges, and giving us access to easy automatic tuning of parameters e.g. to match a certain firing rate, provided that the nonlinearity in the model is differentiable. The tuning functionality allows for fitting arbitrary parameters and can provide a starting point for implementing encoding models.

In addition to the models, the package includes dataset classes that can generate random connectivity matrices from a distribution or load pre-constructed connectivity matrices into torch_geometric's Data objects to be passed straight to the model. These objects hold a sparse representation of our connectivity matrices and can be batched together to form isolated subgraphs of a big graph, letting us simulate many networks simultaneously.

Finally, to facilitate the common use pattern of adding an external stimulus to the simulation and recording the resulting activity, we have included various stimulation classes that can be easily added to the model and even tuned to provoke a certain response.

References

- Bronstein, M. M., Bruna, J., Cohen, T., & Velickovic, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478. <https://arxiv.org/abs/2104.13478>
- Das, A., & Fiete, I. R. (2020). Systematic errors in connectivity inferred from activity in strongly recurrent networks. *Nature Neuroscience*, 23(10), 1286–1296. <https://doi.org/10.1038/s41593-020-0699-2>
- Fey, M., & Lenssen, J. E. (2019). Fast graph representation learning with PyTorch Geometric. *ICLR Workshop on Representation Learning on Graphs and Manifolds*.
- Gerstner, W. (2008). Spike-response model. *Scholarpedia*, 3(12), 1343. <https://doi.org/10.4249/scholarpedia.1343>
- Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press. <https://doi.org/10.1017/CBO9781107447615>
- Lepperød, M. E., Stöber, T., Hafting, T., Fyhn, M., & Kording, K. P. (2020). Inferring causal connectivity from pairwise recordings and optogenetics. *bioRxiv*. <https://doi.org/8.1101/463760>
- Meyer, A. F., Williamson, R. S., Linden, J. F., & Sahani, M. (2017). Models of neuronal stimulus-response functions: Elaboration, estimation, and evaluation. *Frontiers in Systems Neuroscience*, 10. <https://doi.org/10.3389/fnsys.2016.00109>
- Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15(4), 243–262.

Pillow, J. W., Shlens, J., Paninski, L., Sher, A., Litke, A. M., Chichilnisky, E. J., & Simoncelli, E. P. (2008). Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207), 995–999. <https://doi.org/10.1038/nature07140>