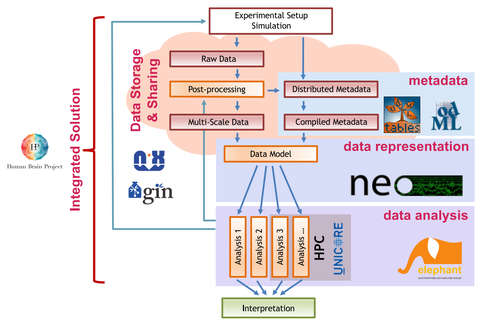
Designing reproducible analysis workflows for experimental and simulated activity using Elephant

Danylo Ulianych[1], Robin Gutzen[1], Julia Sprenger[1], Elena Pastorelli[2], Giulia De Bonis[2], Pier Stanislao Paolucci[2], Andrew Davison[3], Sonja Grün[1,4], Michael Denker[1]

[1] Inst. of Neuroscience and Medicine (INM-6), Inst. for Adv. Simulation (IAS-6), JARA Inst. Brain Structure-Function Relationships (INM-10), Jülich Research Centre, Germany
[2] National Institute for Nuclear Physics (INFN), Rome, Italy
[3] Unité de Neurosciences, Information et Complexité, Neuroinformatics group, CNRS FRE 3693, Gif-sur-Yvette, France
[4] Theoretical Systems Neurobiology, RWTH Aachen University, Germany

Introduction

The need for **reproducible research** is a topic of intense discussion in the neurosciences. In the context of **data analysis**, we develop the Electrophysiology Analysis Toolkit (Elephant, [1]) as a central resource to provide tested and validated reference implementations of common analysis methods for activity data. However, reproducibility also requires such tools to be embedded in **collaborative, holistic workflows** [2] providing **clear, traceable** analysis steps from data acquisition to publication.

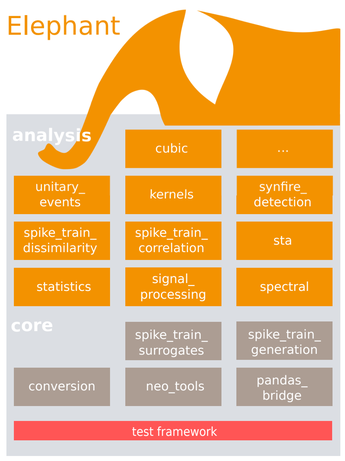


Here, we showcase how Elephant is integrated into an analysis workflow running on the the Collaboratory, reproducing work in [3]. The workflow consists of complementary open-source tools and services [4] for: metadata management (odML and odMLtables, [5,6,7]), data query (HBP Knowledgegraph, [8]), data versioning (gin, [9]), data storage (nix, [10]), data handling (Neo, [11]). Finally, we outline how these building blocks, combined with generic tools, can be assembled into formalized workflows to support reproducible research, e.g. the validation of network simulations with NetworkUnit.

Analysis using Elephant

Elephant is a community-centered, open-source software package that provides components for the analysis of multi-scale electrophysiological data (e.g., spike trains, local field potentials) from experiments and neuronal simulations, focusing on:

- methods for the analysis of parallel recordings
- correlative features of brain dynamics
- bridging different scales of observation



Summary

The presented analysis workflow...

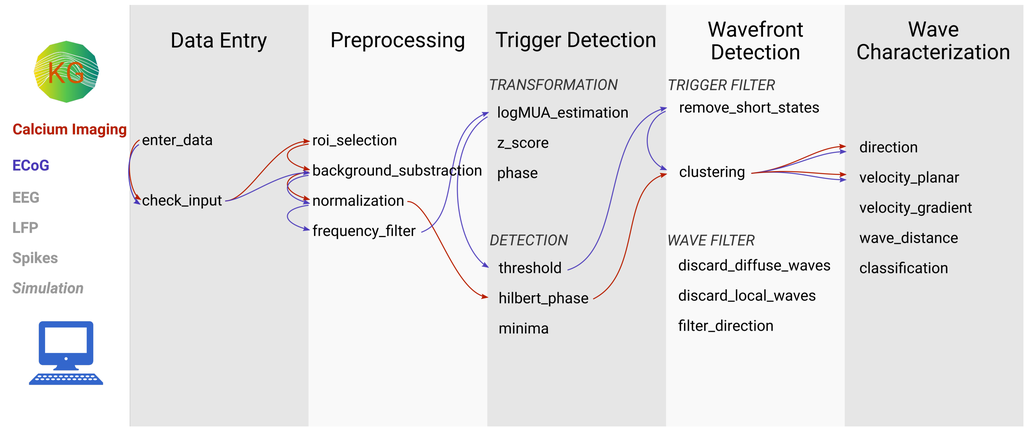
- ... **combines** several public, community-centered software tools to achieve a reproducible analysis.
- ... is re-usable and suitable for **collaborative work** between laboratories by use of the HBP Collaboratory.
- ... provides a comprehensible data flow across scales **independent of the data format** using the Neo library.
- ... leads the way towards the implementation of **future analysis workflows** based on the Elephant library.

Find further resouces:

http://python-elephant.org
https://github.com/NeuralEnsemble/elephant

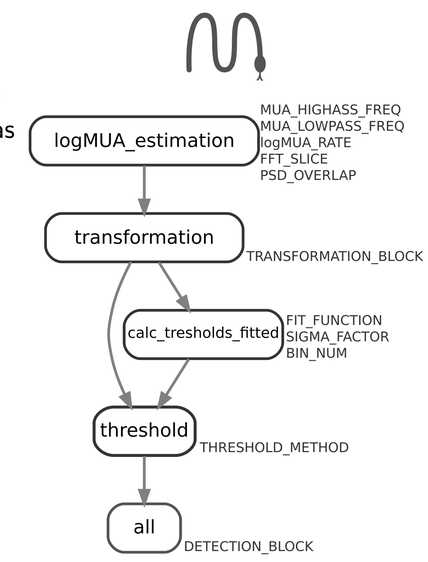
Multi-scale analysis workflow on the Collaboratory

The Collaboratory infrastructure of the Human Brain Project hosts the overall workflow of the project, required tools, central data management, data search, HPC access, and the ability for interactive work.
The implementation employs a variety of existing tools to describe the workflow in a **generalized** and **reusable** form. It is structured as a hierarchical arrangement of **modular** elements. The top level is a *pipeline*, which is composed of sequential *stages*. Each *stage* is a collection of modular analysis/processing *blocks*.



- The execution instructions of blocks and stages are defined with Snakemake.
- Parameter settings and the selection of *blocks* is separated in config files.
- The interfaces between *stages*/*blocks* are standardized (Neo, pandas).
- The *blocks* use standard implementations of algorithms (e.g from Elephant).
- Fairgraph enables fetching data and storing (intermediate) analysis results as well as analysis activity (provenance) with the KnowledgeGraph.

→ **reproducible & comparable analysis/validation/benchmarking**




```
rule frequency_filter:
    input:
        data = input_file,
        script = 'scripts/frequency_filter.py',
        config = 'config.yaml'
    output:
        data = os.path.join(output_path, 'frequency_filter.nix'),
    shell:
        """
        python {input.script} --data "{input.data}" \
                              --output "{output.data}" \
                              --highpass_freq {config['HIGHPASS_FREQ']} \
                              --lowpass_freq {config['LOWPASS_FREQ']} \
                              --order {config['FILTER_ORDER']} \
                              --filter_function {config['FILTER_FUNCTION']}
        """
```

elephant signal_processing.butter

Validation with NetworkUnit

- **Validation** is the process of establishing confidence in a model by quantitatively testing whether its prediction accuracy is within an acceptable agreement to its system of interest.
- **Network-level validation** evaluates the model simulation on the level of the network activity as opposed to the complementary approach of validating on a single-cell level.
- **Model-to-model validation** compares models (or their implementations) for consistency, cross-validation, simulator evaluation, or quantification of model developments. [12]

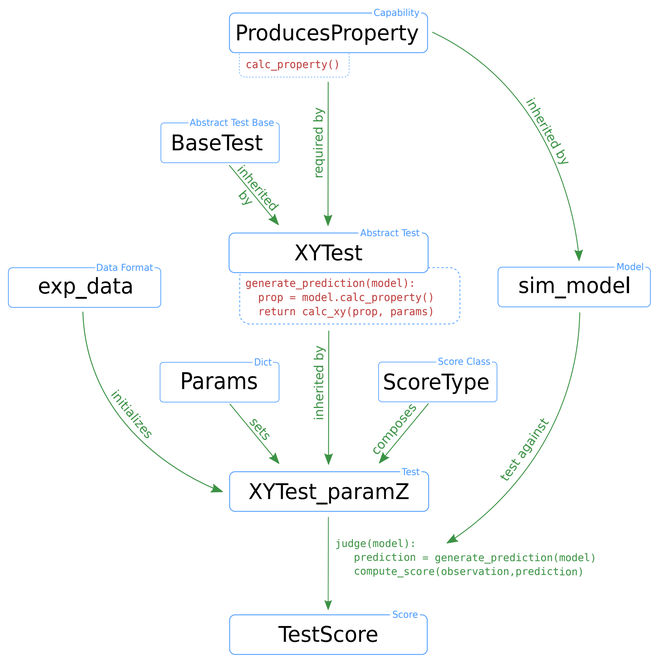
The Python module NetworkUnit [13] is based on SciUnit [14] and Elephant, and provides a formalized framework along with a battery of standardized tests for network-level validation.

- Models are matched to appropriate tests via 'capabilities'.
- New tests can be easily derived from a range of base tests.
- Tests can be adapted to also compare multiple models.
- Test scores are annotated with their provenance.

Standardization → Reproducibilty

Modularization → Versatility

Formalization → Understandability



References

[1] http://python-elephant.org
[2] Badia, R., et al. "INCF Program on Standards for data sharing: new perspectives on workflows and data management for the analysis of electrophysiological data." Techn. Report, International Neuroinformatics Coordination Facility (INCF). 2015.
[3] Denker, M, et al. (2018) LFP beta amplitude is linked to mesoscopic spatio-temporal phase patterns. Scientific Reports 8.1: 5200. DOI:10.1038/s41598-018-22990-7
[4] Denker M. and Grün S (2016). Designing Workflows for the Reproducible Analysis of Electrophysiological Data. In Brain-Inspired Computing, K. Amunts, L. Grandinetti, T. Lippert, and N. Petkov, eds. (Cham: Springer International Publishing), pp. 5872
[5] Grewe, J. et al. (2011). A bottom-up approach to data annotation in neurophysiology. Frontiers in Neuroinformatics, 5, p.16.
[6] Zehl, L. et al. (2016). Handling metadata in a neurophysiology laboratory. Frontiers in Neuroinformatics, 10, p.26.
[7] https://github.com/INM-6/python-odmltables
[8] https://www.humanbrainproject.eu/en/explore-the-brain/search
[9] https://web.gin.g-node.org
[10] https://github.com/G-Node/nix/wiki
[11] Garcia S, et al., (2014). Neo: an object model for handling electrophysiology data in multiple formats. Frontiers in Neuroinformatics, 8, 10. DOI:10.3389/fninf.2014.0001
[12] Gutzen et al. (sub.) "Reproducible neural network simulations: statistical methods for model validation on the level of network activity data"
[13] https://github.com/INM-6/NetworkUnit
[14] https://github.com/scidash/sciunit

This research has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 785907 (Human Brain Project SGA2).