Finanziato dall'Unione europea
NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA

INFN

Minimizing time-to-result: Cobrawap latest developments and applications

Cosimo Lupo
cosimo.lupo@roma1.infn.it

INFN
Istituto Nazionale di Fisica Nucleare
https://apegate.roma1.infn.it/
@APELab_INFN

© 2019 Giovanni Carrieri info@giovannicarrieri.com www.giovannicarrieri.com

Computing @ CSN5:
applications and innovations at INFN

Bari, 14-16 October 2024

Progetto EBRAINS-Italy IR00011, CUP B51E2200015006
Missione 4 - Istruzione e Ricerca, Componente 2, Azione 3.1.1

EBRAINS Italy

The software every neuroscientist dreams of…

- Standardized and generalized methods and processes, moving **from qualitative to quantitative** comparisons
- Enabling a **common language** for multi-disciplinary research, providing user-friendly software solutions and spreading innovative methods and results out of the lab
- Maximizing the *return-on-investment* (of human resources/efforts), reducing the ***time-to-journal***
- Operating in a **collaborative environment**, exploiting cutting-edge technologies and latest trends in software engineering

The software every neuroscientist dreams of…

- Standardized and generalized methods and processes, moving **from qualitative to quantitative** comparisons
- Enabling a **common language** for multi-disciplinary research, providing user-friendly software solutions and spreading innovative methods and results out of the lab
- Maximizing the *return-on-investment* (of human resources/efforts), reducing the ***time-to-journal***
- Operating in a **collaborative environment**, exploiting cutting-edge technologies and latest trends in software engineering

COBRA WAP

Cobrawap

COllaborative **BRA**in **W**ave **A**nalysis **P**ipeline

https://cobrawap.readthedocs.io

https://github.com/NeuralEnsemble/cobrawap

GitHub

neural ensemble .org

The software every neuroscientist dreams of…

- Standardized and generalized methods and processes, moving **from qualitative to quantitative** comparisons
- Enabling a **common language** for multi-disciplinary research, providing user-friendly software solutions and spreading innovative methods and results out of the lab
- Maximizing the *return-on-investment* (of human resources/efforts), reducing the ***time-to-journal***
- Operating in a **collaborative environment**, exploiting cutting-edge technologies and latest trends in software engineering

COBRA WAP

GitHub

neural ensemble .org

Cobrawap

COllaborative **BRA**in **W**ave **A**nalysis **P**ipeline

https://cobrawap.readthedocs.io

https://github.com/NeuralEnsemble/cobrawap

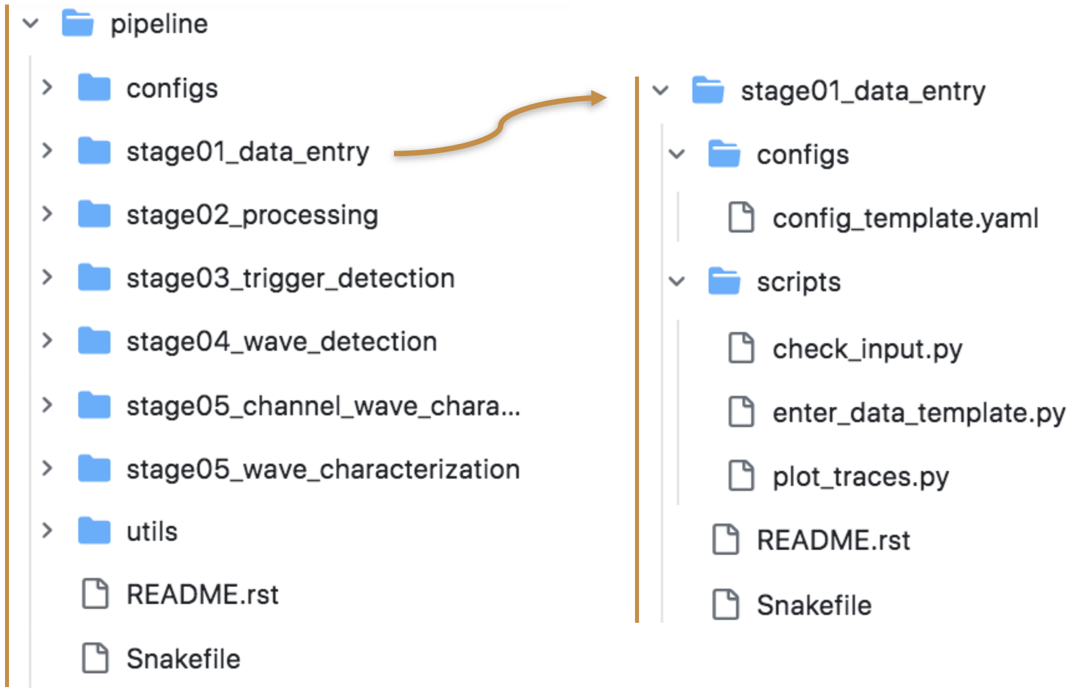
Cobrawap as a service

- Exploit remote computational resources (e.g. HPC)
- Reduce users' technical efforts for installation & execution

Cobrawap under the magnifying glass

Cell Reports Methods 4, 100681 (2024)

- originally focusing on slow-wave dynamics, now more general
- designed and developed in collaboration with Jülich Forschungszentrum
- open source, public repository (github.com/NeuralEnsemble/cobrawap)
- Python + expansion/integration of general common tools
- hierarchically built up as a sequence of (almost) fixed **stages**, each made up of customizable **blocks**



```
pipeline
  configs
  stage01_data_entry
  stage02_processing
  stage03_trigger_detection
  stage04_wave_detection
  stage05_channel_wave_chara...
  stage05_wave_characterization
  utils
  README.rst
  Snakefile
```

```
stage01_data_entry
  configs
    config_template.yaml
  scripts
    check_input.py
    enter_data_template.py
    plot_traces.py
  README.rst
  Snakefile
```

Stages

1. Data Entry → common format
2. Processing → processed data
3. Trigger Detection → transition times
4. Wave Detection → wave collection
5. Wave Characterization → wave parameters

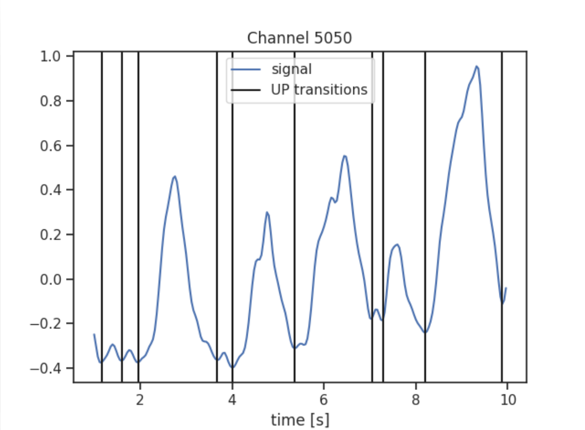
Blocks

Implement single methods and algorithms (modularity)

Cobrawap under the magnifying glass

Cell Reports Methods 4, 100681 (2024)

- originally focusing on slow-wave dynamics, now more general
- designed and developed in collaboration with Jülich Forschungszentrum
- open source, public repository (github.com/NeuralEnsemble/cobrawap)
- Python + expansion/integration of general common tools
- hierarchically built up as a sequence of (almost) fixed **stages**, each made up of customizable **blocks**



Stages

1. Data Entry → common format
2. Processing → processed data
3. Trigger Detection → transition times
4. Wave Detection → wave collection
5. Wave Characterization → wave parameters

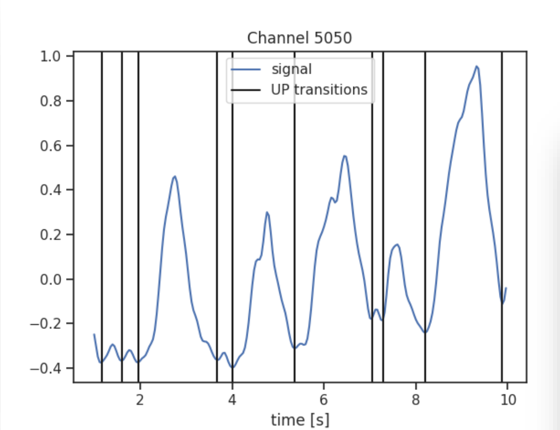
Blocks

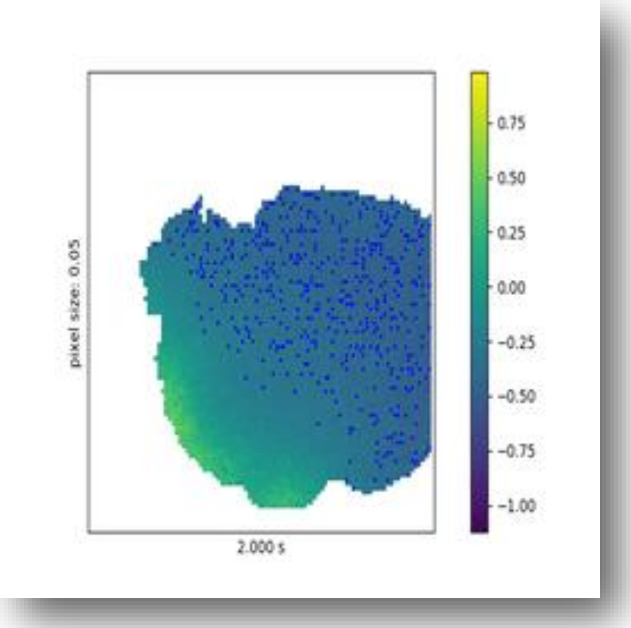
Implement single methods and algorithms (modularity)

Cobrawap under the magnifying glass

Cell Reports Methods 4, 100681 (2024)

- originally focusing on slow-wave dynamics, now more general
- designed and developed in collaboration with Jülich Forschungszentrum
- open source, public repository (github.com/NeuralEnsemble/cobrawap)
- Python + expansion/integration of general common tools
- hierarchically built up as a sequence of (almost) fixed **stages**, each made up of customizable **blocks**





Stages

1. Data Entry → common format
2. Processing → processed data
3. Trigger Detection → transition times
4. Wave Detection → wave collection
5. Wave Characterization → wave parameters

Blocks

Implement single methods and algorithms (modularity)

Comparing apples to apples…

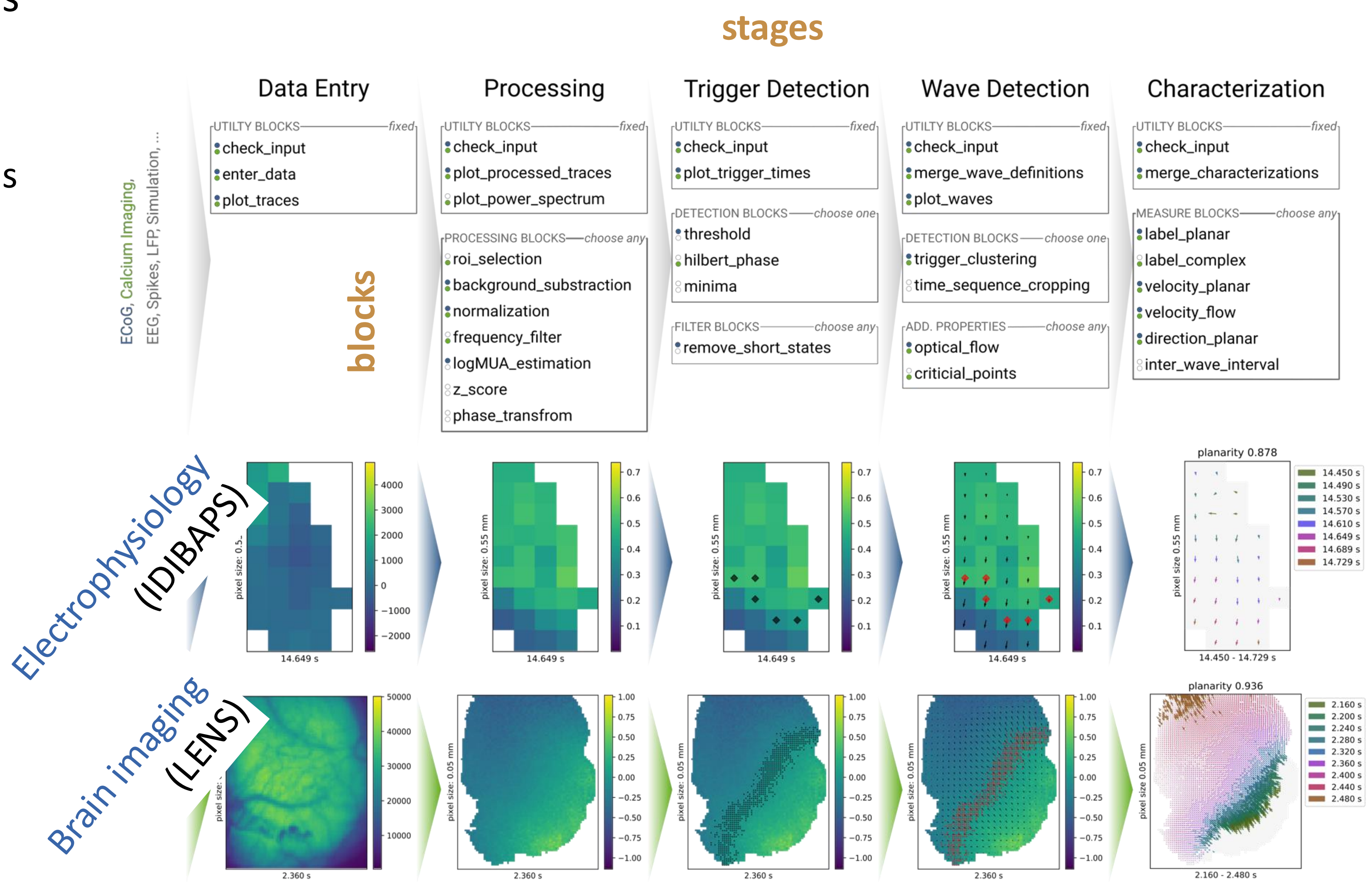
Cell Reports Methods 4, 100681 (2024)

Cobrawap allows the comparison across experimental recordings from different laboratories and techniques:

- **multimodality** and cross-domain findings
- integration and **complementarity**
- minimization of the impact of artifacts
- benchmark of methods against heterogenous data → **robustness**
- check and monitor of analysis settings integrated in the pipeline → increase confidence in findings, **reproducibility**

→ **FAIR principles**

… but it also allows for model vs experiment comparison: model **calibration** & **validation**!



stages

blocks

ECoG, Calcium Imaging, EEG, Spikes, LFP, Simulation, …

Data Entry

UTILTY BLOCKS — fixed
- check_input
- enter_data
- plot_traces

Processing

UTILTY BLOCKS — fixed
- check_input
- plot_processed_traces
- plot_power_spectrum

PROCESSING BLOCKS — choose any
- roi_selection
- background_substraction
- normalization
- frequency_filter
- logMUA_estimation
- z_score
- phase_transfrom

Trigger Detection

UTILTY BLOCKS — fixed
- check_input
- plot_trigger_times

DETECTION BLOCKS — choose one
- threshold
- hilbert_phase
- minima

FILTER BLOCKS — choose any
- remove_short_states

Wave Detection

UTILTY BLOCKS — fixed
- check_input
- merge_wave_definitions
- plot_waves

DETECTION BLOCKS — choose one
- trigger_clustering
- time_sequence_cropping

ADD. PROPERTIES — choose any
- optical_flow
- criticial_points

Characterization

UTILTY BLOCKS — fixed
- check_input
- merge_characterizations

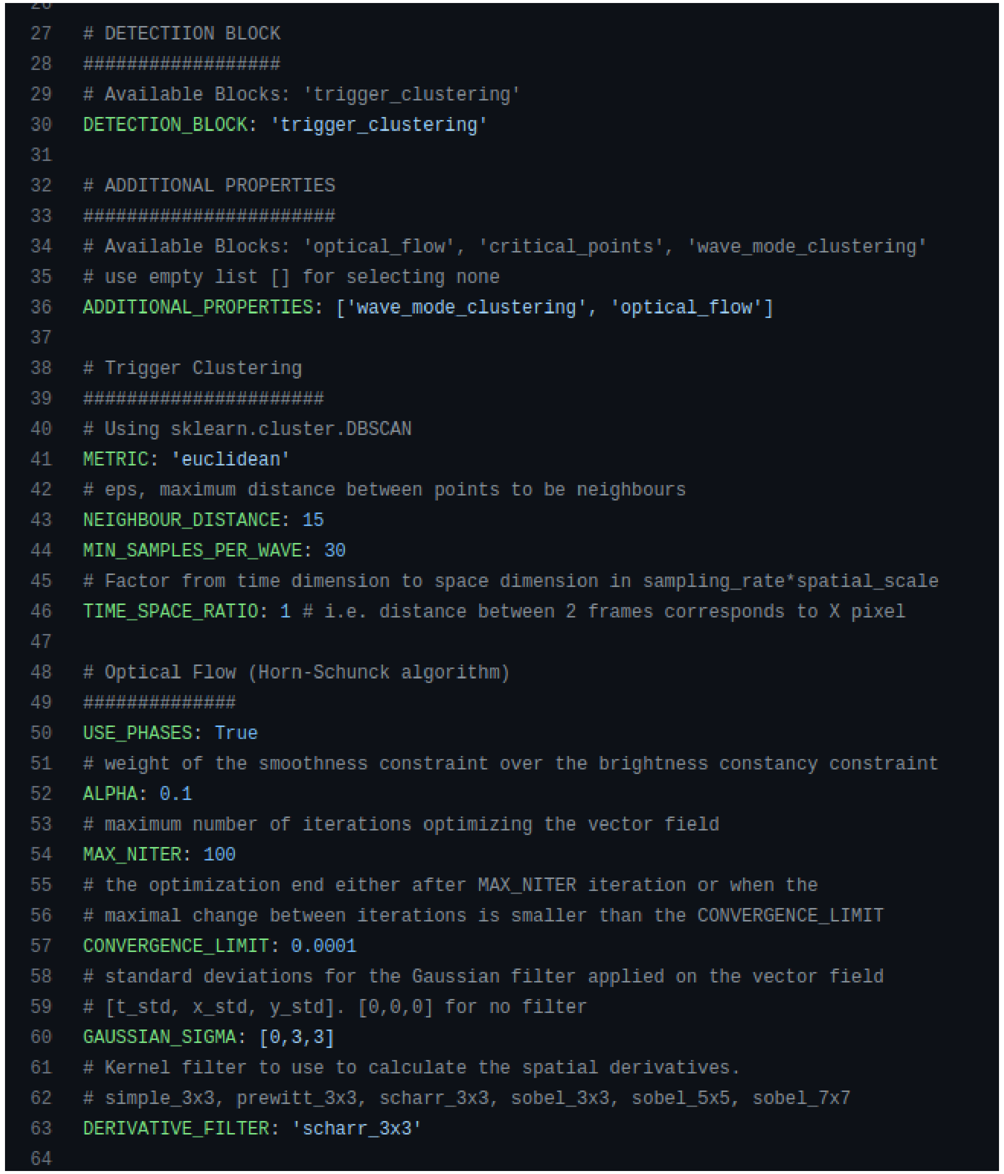
MEASURE BLOCKS — choose any
- label_planar
- label_complex
- velocity_planar
- velocity_flow
- direction_planar
- inter_wave_interval

Electrophysiology (IDIBAPS)

Brain imaging (LENS)

Pushing the limits of user-friendliness

Almost any parameter can be tuned
via a set of **human-readable config files**.



```
# DETECTIION BLOCK
##################
# Available Blocks: 'trigger_clustering'
DETECTION_BLOCK: 'trigger_clustering'

# ADDITIONAL PROPERTIES
#######################
# Available Blocks: 'optical_flow', 'critical_points', 'wave_mode_clustering'
# use empty list [] for selecting none
ADDITIONAL_PROPERTIES: ['wave_mode_clustering', 'optical_flow']

# Trigger Clustering
######################
# Using sklearn.cluster.DBSCAN
METRIC: 'euclidean'
# eps, maximum distance between points to be neighbours
NEIGHBOUR_DISTANCE: 15
MIN_SAMPLES_PER_WAVE: 30
# Factor from time dimension to space dimension in sampling_rate*spatial_scale
TIME_SPACE_RATIO: 1 # i.e. distance between 2 frames corresponds to X pixel

# Optical Flow (Horn-Schunck algorithm)
##############
USE_PHASES: True
# weight of the smoothness constraint over the brightness constancy constraint
ALPHA: 0.1
# maximum number of iterations optimizing the vector field
MAX_NITER: 100
# the optimization end either after MAX_NITER iteration or when the
# maximal change between iterations is smaller than the CONVERGENCE_LIMIT
CONVERGENCE_LIMIT: 0.0001
# standard deviations for the Gaussian filter applied on the vector field
# [t_std, x_std, y_std]. [0,0,0] for no filter
GAUSSIAN_SIGMA: [0,3,3]
# Kernel filter to use to calculate the spatial derivatives.
# simple_3x3, prewitt_3x3, scharr_3x3, sobel_3x3, sobel_5x5, sobel_7x7
DERIVATIVE_FILTER: 'scharr_3x3'

```

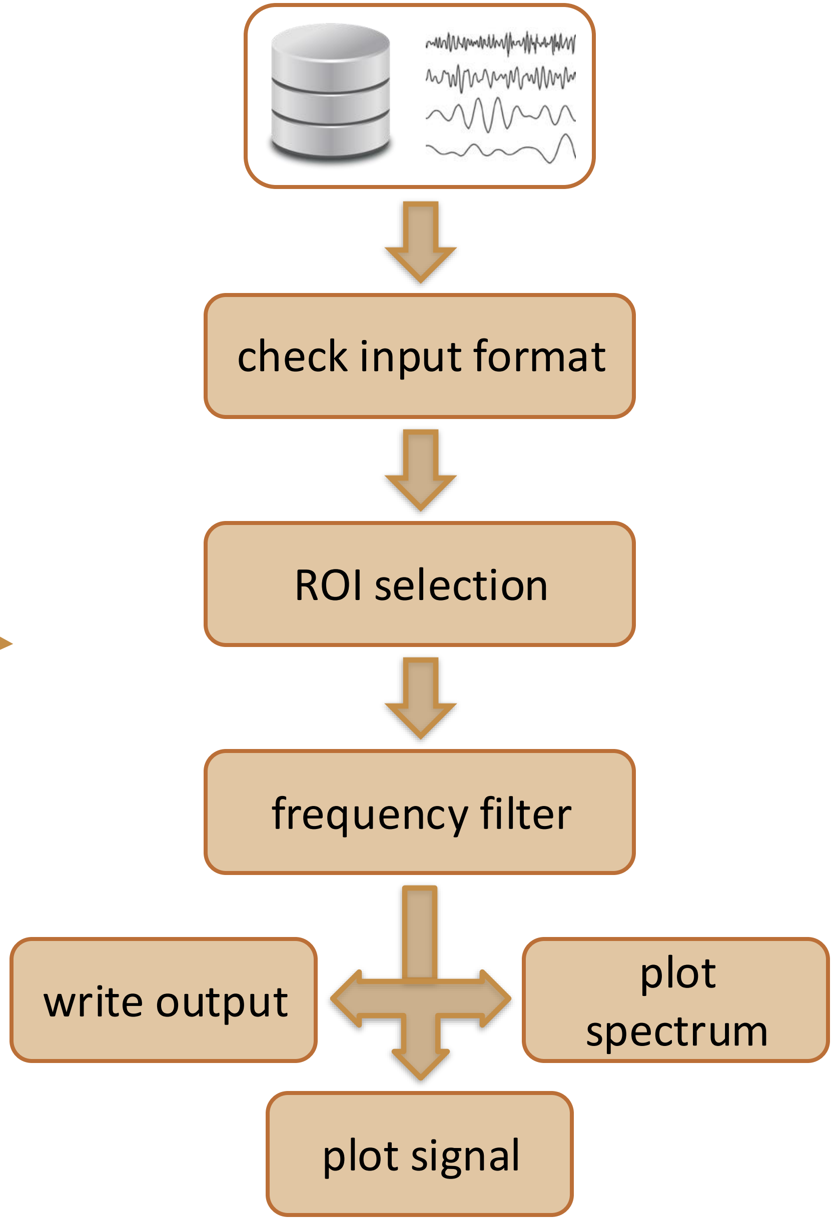
Pushing the limits of user-friendliness

Almost any parameter can be tuned
via a set of human-readable config files.

Parameters are then parsed and fed to
the pipeline via **workflow manager systems**,
that orchestrate the execution flow.

snakemake

COMMON
WORKFLOW
LANGUAGE



check input format

ROI selection

frequency filter

write output

plot spectrum

plot signal

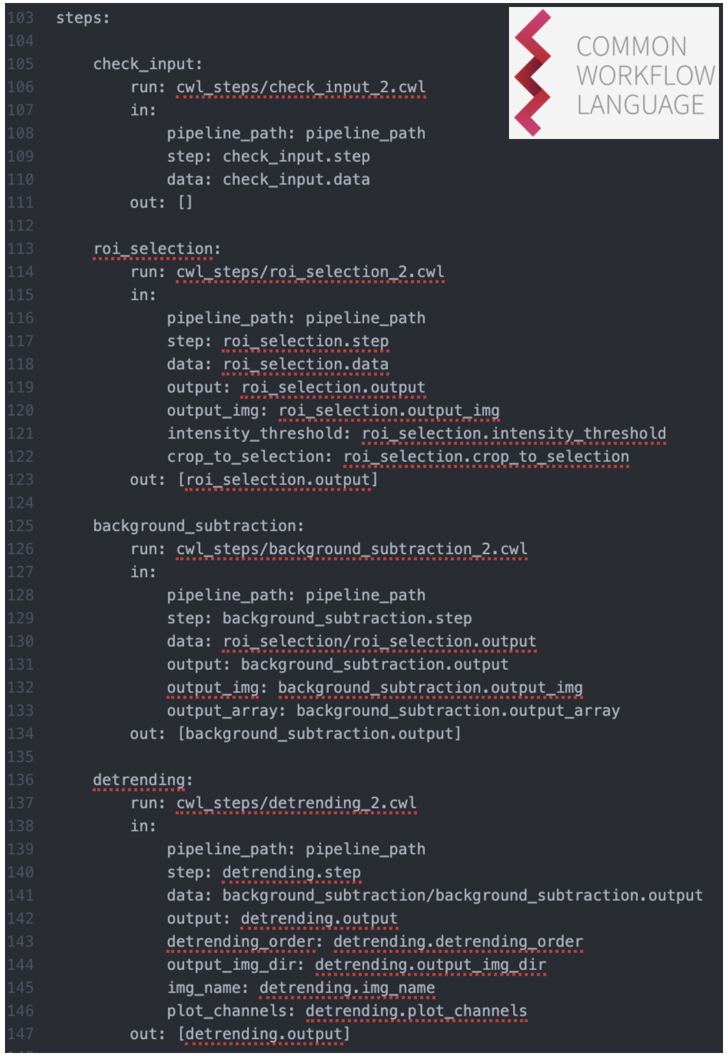
Pushing the limits of user-friendliness

Almost any parameter can be tuned
via a set of human-readable config files.

Parameters are then parsed and fed to
the pipeline via **workflow manager systems**,
that orchestrate the execution flow.

snakemake

COMMON
WORKFLOW
LANGUAGE



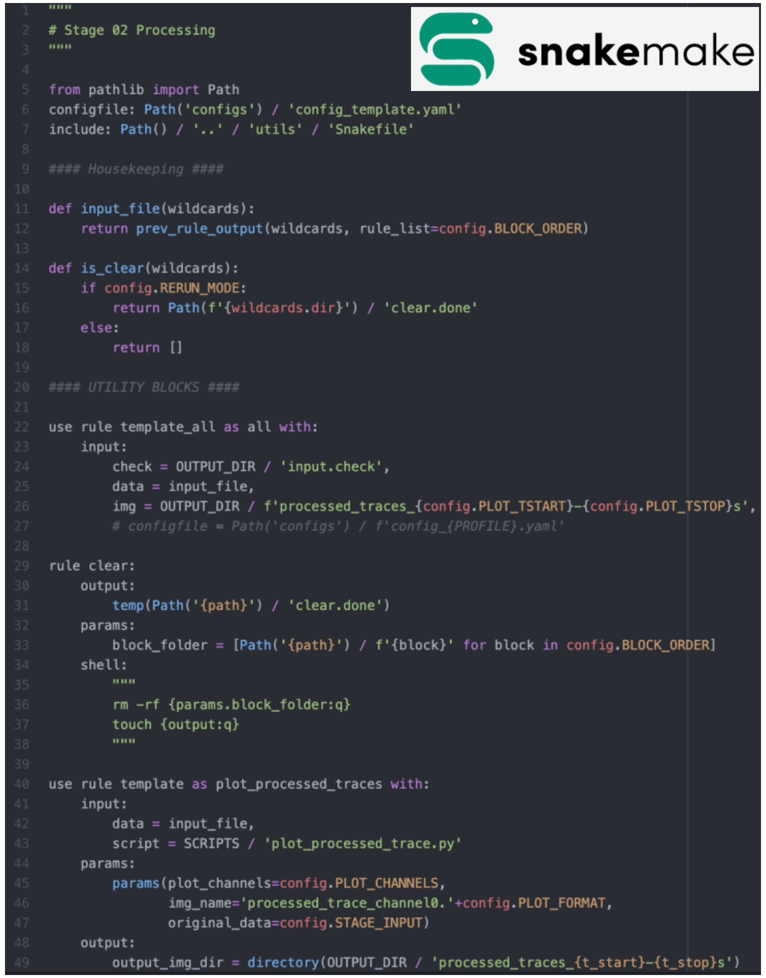
```yaml
steps:

    check_input:
        run: cwl_steps/check_input_2.cwl
        in:
            pipeline_path: pipeline_path
            step: check_input.step
            data: check_input.data
        out: []

    roi_selection:
        run: cwl_steps/roi_selection_2.cwl
        in:
            pipeline_path: pipeline_path
            step: roi_selection.step
            data: roi_selection.data
            output: roi_selection.output
            output_img: roi_selection.output_img
            intensity_threshold: roi_selection.intensity_threshold
            crop_to_selection: roi_selection.crop_to_selection
        out: [roi_selection.output]

    background_subtraction:
        run: cwl_steps/background_subtraction_2.cwl
        in:
            pipeline_path: pipeline_path
            step: background_subtraction.step
            data: roi_selection/roi_selection.output
            output: background_subtraction.output
            output_img: background_subtraction.output_img
            output_array: background_subtraction.output_array
        out: [background_subtraction.output]

    detrending:
        run: cwl_steps/detrending_2.cwl
        in:
            pipeline_path: pipeline_path
            step: detrending.step
            data: background_subtraction/background_subtraction.output
            output: detrending.output
            detrending_order: detrending.detrending_order
            output_img_dir: detrending.output_img_dir
            img_name: detrending.img_name
            plot_channels: detrending.plot_channels
        out: [detrending.output]
```



```python
"""
# Stage 02 Processing
"""

from pathlib import Path
configfile: Path('configs') / 'config_template.yaml'
include: Path() / '..' / 'utils' / 'Snakefile'

#### Housekeeping ####

def input_file(wildcards):
    return prev_rule_output(wildcards, rule_list=config.BLOCK_ORDER)

def is_clear(wildcards):
    if config.RERUN_MODE:
        return Path(f'{wildcards.dir}') / 'clear.done'
    else:
        return []

#### UTILITY BLOCKS ####

use rule template_all as all with:
    input:
        check = OUTPUT_DIR / 'input.check',
        data = input_file,
        img = OUTPUT_DIR / f'processed_traces_{config.PLOT_TSTART}-{config.PLOT_TSTOP}s',
        # configfile = Path('configs') / f'config_{PROFILE}.yaml'

rule clear:
    output:
        temp(Path('{path}') / 'clear.done')
    params:
        block_folder = [Path('{path}') / f'{block}' for block in config.BLOCK_ORDER]
    shell:
        """
        rm -rf {params.block_folder:q}
        touch {output:q}
        """

use rule template as plot_processed_traces with:
    input:
        data = input_file,
        script = SCRIPTS / 'plot_processed_trace.py'
    params:
        params(plot_channels=config.PLOT_CHANNELS,
               img_name='processed_trace_channel0.'+config.PLOT_FORMAT,
               original_data=config.STAGE_INPUT)
    output:
        output_img_dir = directory(OUTPUT_DIR / 'processed_traces_{t_start}-{t_stop}s')
```

Pushing the limits of user-friendliness

Almost any parameter can be tuned
via a set of human-readable config files.

Parameters are then parsed and fed to
the pipeline via workflow manager systems,
that orchestrate the execution flow.

Everything is transparent to the user,
being hidden by an intuitive
command-line interface (CLI)
which is pip-installable.

- Set up folder, paths and settings:

```
cobrawap init
```

- Add a specific profile for a dataset:

```
cobrawap add_profile
```

- Run the whole pipeline, or single parts of it:

```
cobrawap run

cobrawap run_stage --stage <…>

cobrawap run_block --stage <…> --block <…>
```

The Cobrawap solution

to intercept the demand for resource scalability & usability

Software delivery

GitHub
docker
CONDA

Collab & KG interaction

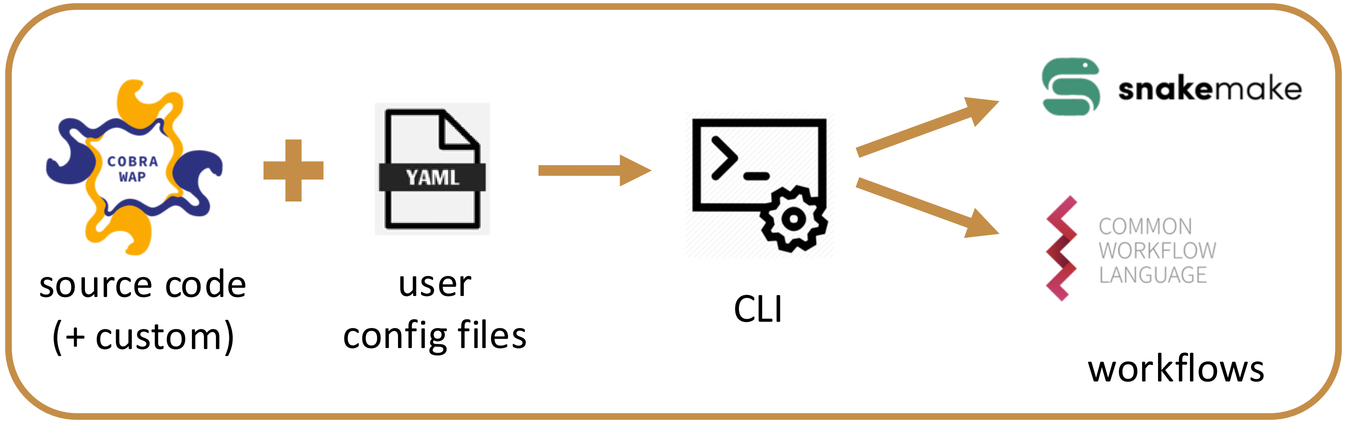
EBRAINS Collaboratory
jupyter
UNICORE

Deploy & run on HPC/cloud

FENIX RI

- JSC
- CSCS
- CINECA

Command-line interface handling workflows



The Cobrawap solution

to intercept the demand for resource scalability & usability

Software delivery

GitHub
docker
CONDA

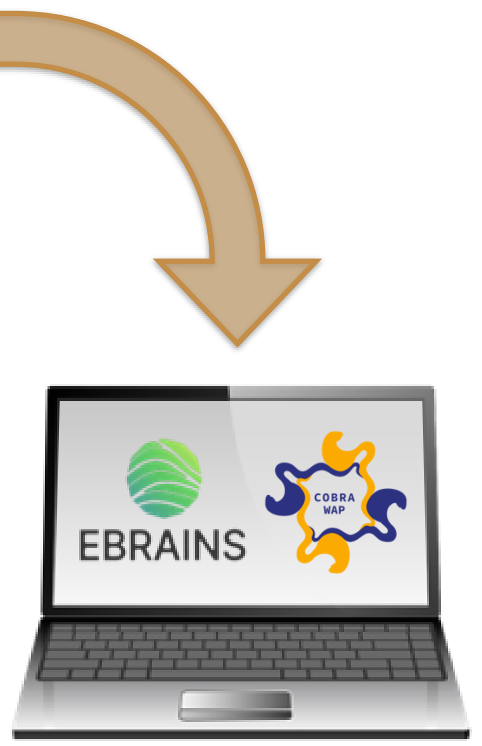
Collab & KG interaction

EBRAINS Collaboratory
jupyter
UNICORE

Deploy & run on HPC/cloud

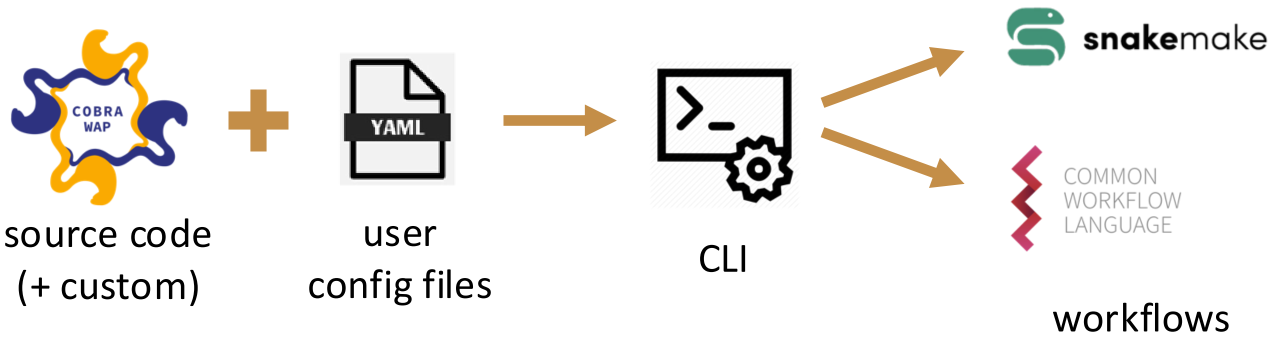
FENIX RI

- JSC
- CSCS
- CINECA



Cobrawap as a service

Command-line interface handling workflows



The Cobrawap solution

to intercept the demand for resource scalability & usability

Software delivery

GitHub, docker, CONDA

Collab & KG interaction

EBRAINS Collaboratory

jupyter

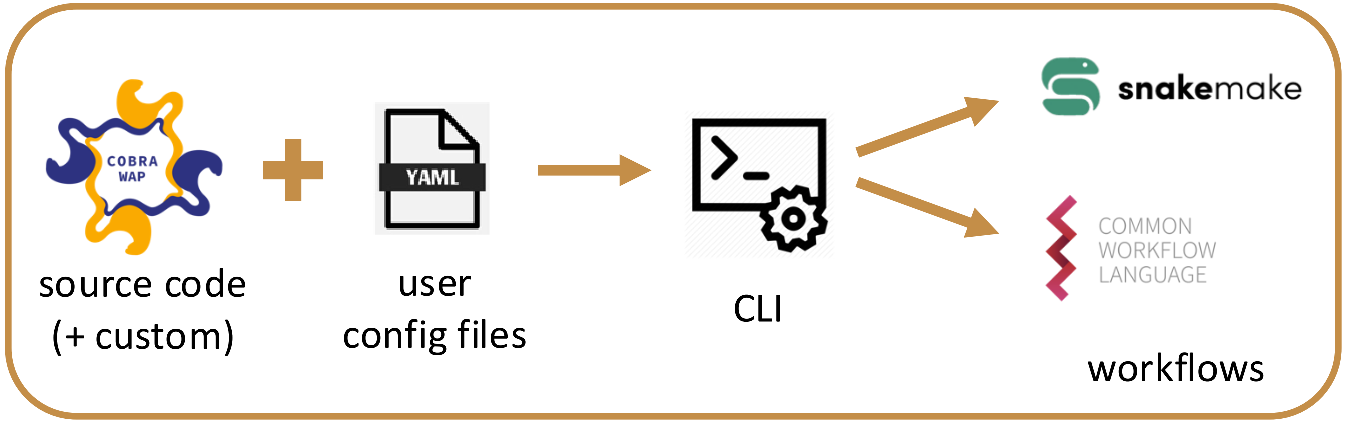
UNICORE

Deploy & run on HPC/cloud

FENIX RI

- JSC
- CSCS
- CINECA

Command-line interface handling workflows



source code (+ custom) + user config files → CLI → workflows

Cobrawap as a service

Target tasks

- Model calibration & validation
- Large-scale data analysis
- Metrics for clinical applications
- Buildout of methods & algorithms

The Cobrawap paradigm (i.e. how science and technology play together…)

Scientific Tasks (development of new blocks)

- Spontaneous vs stimulated/evoked data
- Anaesthesia vs natural sleep and wakefulness
- TVB-Human & EEG → towards human data, beyond surface recordings (3D data and models), measures of complexity
- Generalized image processing
- CBF + ECoG in mice
- BOLD-fMRI + EEG in humans

Funded until 2026 by INFN CSN5 project BRAINSTAIN

close interplay between scientific and technical tasks
→ **dialogue** and **co-design**

Funded until 2025++ by PNRR EBRAINS Italy

Technical Tasks (towards an EBRAINS service)

- Documentation & CI/CD
- Execution on HPC
- Workflow managers: Snakemake and CWL
- Parallelization and speed up
- Deployment (spack, pip, Docker)
- Input data: link with the EBRAINS-KG

The Cobrawap paradigm (i.e. how science and technology play together…)

Scientific Tasks (development of new blocks)

- Spontaneous vs stimulated/evoked data
- Anaesthesia vs natural sleep and wakefulness
- TVB-Human & EEG → towards human data, beyond surface recordings (3D data and models), measures of complexity
- Generalized image processing
- CBF + ECoG in mice
- BOLD-fMRI + EEG in humans

Funded until 2026 by INFN CSN5 project BRAINSTAIN

See in particular WP1 of BRAINSTAIN:
«Analysis pipelines, data processing, data analysis»

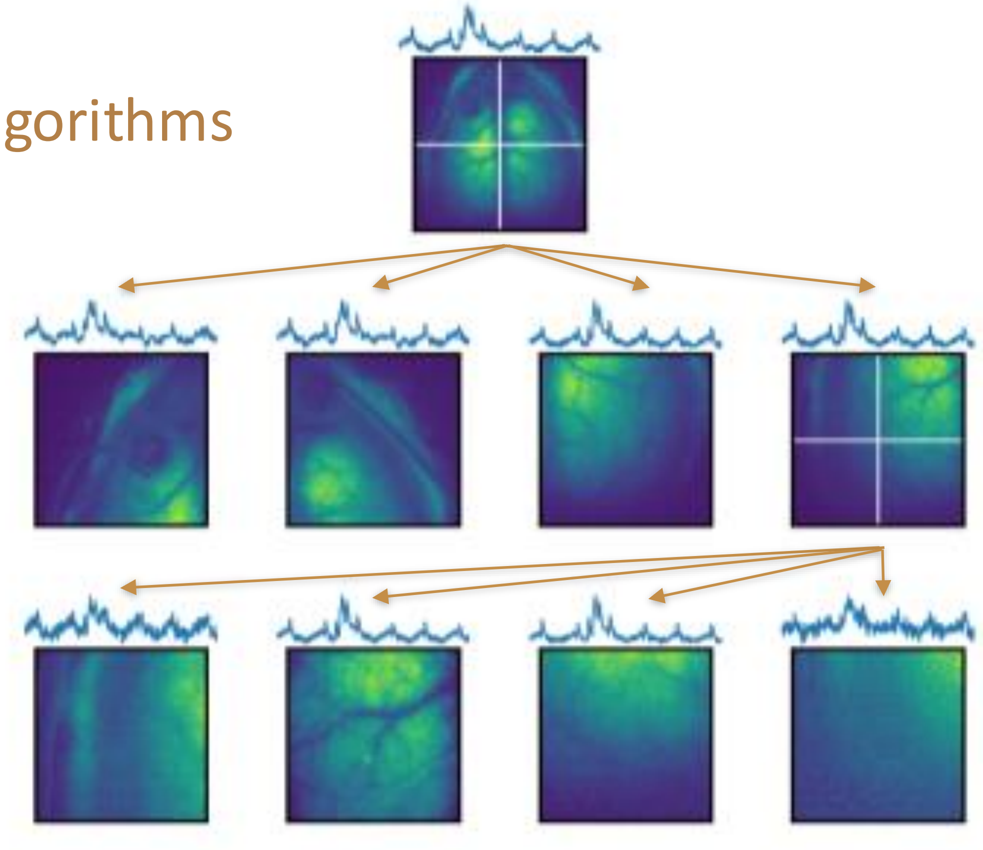
- Coordinated by Giulia De Bonis (RM1)
- Task leaders: C. Lupo (RM1), G. De Bonis (RM1), P. Oliva (CA)

Deployment (spack, pip, Docker)

- Input data: link with the EBRAINS-KG

HOS (Hierarchical Optimal Sampling)

an example of data- & question-driven development of algorithms



Dealing with high-res imaging datasets is often challenging, e.g.:

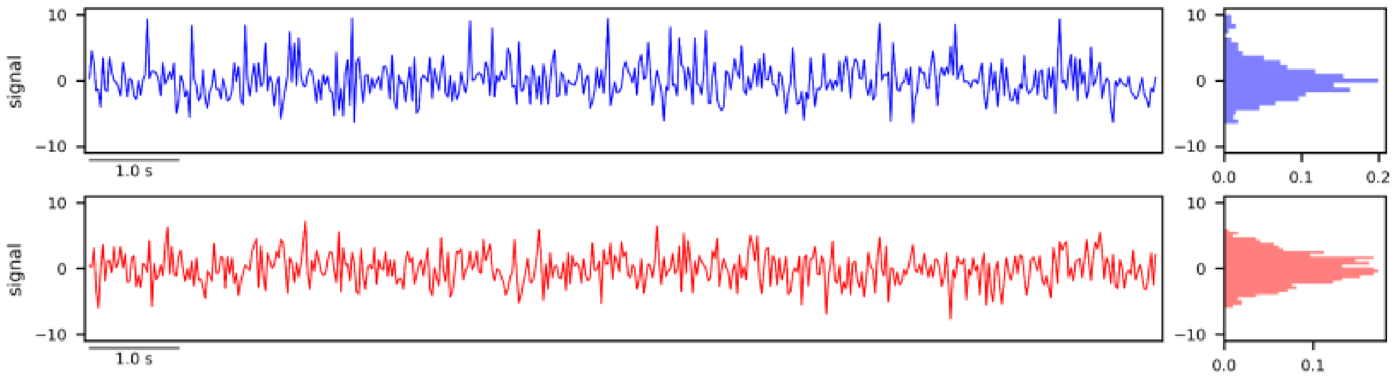
- Huge demand for storage
- Massive computational resources (CPU time, RAM, …)

… but do we need **everywhere** this super-high resolution???

large number of channels

significative percentage
of non-informative pixels or
with low signal-to-noise ratio

keep and use only the informative(1) ones,
preserve native resolution
when admittable(1) according to
a given scientific question

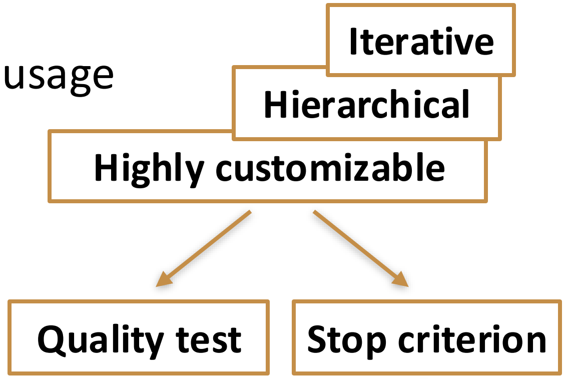


Paper in preparation

HOS (Hierarchical Optimal Sampling)

an example of data- & question-driven development of algorithms

- Can be employed to quantitatively evaluate the «goodness» of a dataset
- Allows for a «smart» (i.e. science-motivated) usage of pre-processing methods (as averages)

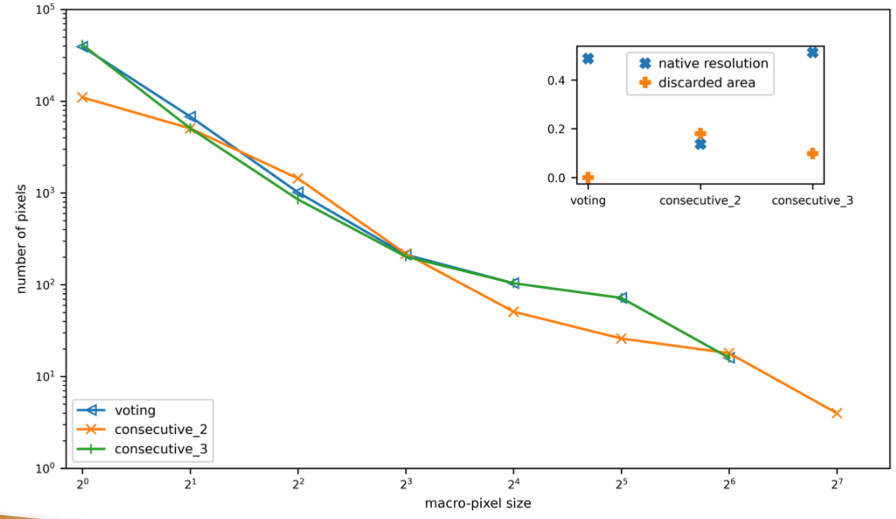

Iterative

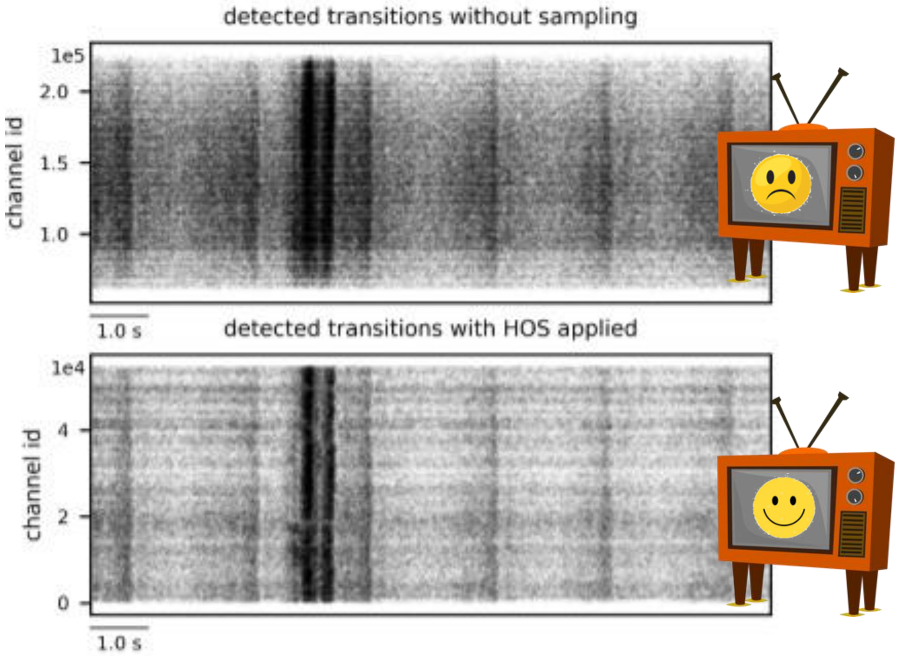
Hierarchical

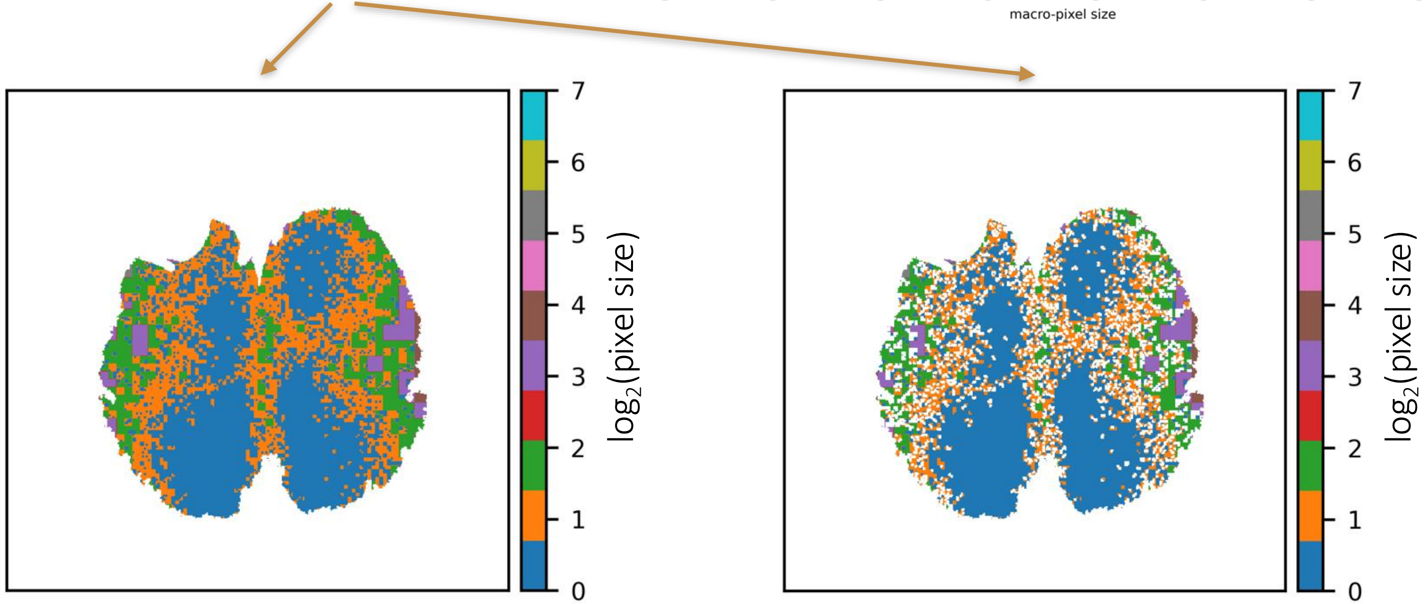
Highly customizable

Quality test

Stop criterion







Cobrawap + TheVirtualBrain: first results…

What happens when wanting to apply Cobrawap to data not arranged in a regularly-spaced 2D grid of channels, e.g. **human EEG**?

It would be necessary to go beyond the 2D representation, toward a **3D** one…

Starting point: in silico data → THEVIRTUALBRAIN.

- Open-source platform for constructing and simulating **personalised network models**
- Relies on **fully customizable** neural models and structural connectomes
- Parameters can be easily **tuned**, for better simulation results...

Cobrawap can be used as a calibration/analysis tool for TVB output!

https://www.thevirtualbrain.org

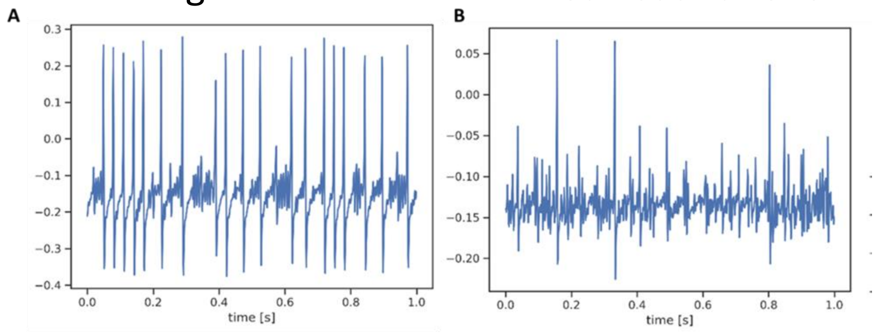
Cobrawap + TheVirtualBrain: first results…

Using TVB with Larter-Breakspear (LB) models, considering 76-node and 998-node connectomes:

- parameters can be tuned, so to retrieve in-vivo features richness
- spatio-temporal propagation of waves can be clearly seen
- different brain states during resting can be identified and classified

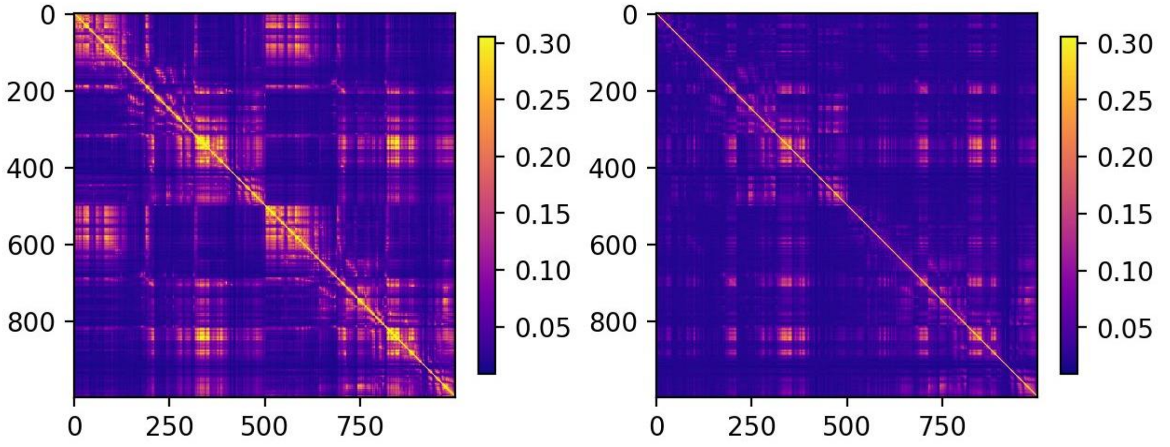
Strong oscillations

Weak oscillations

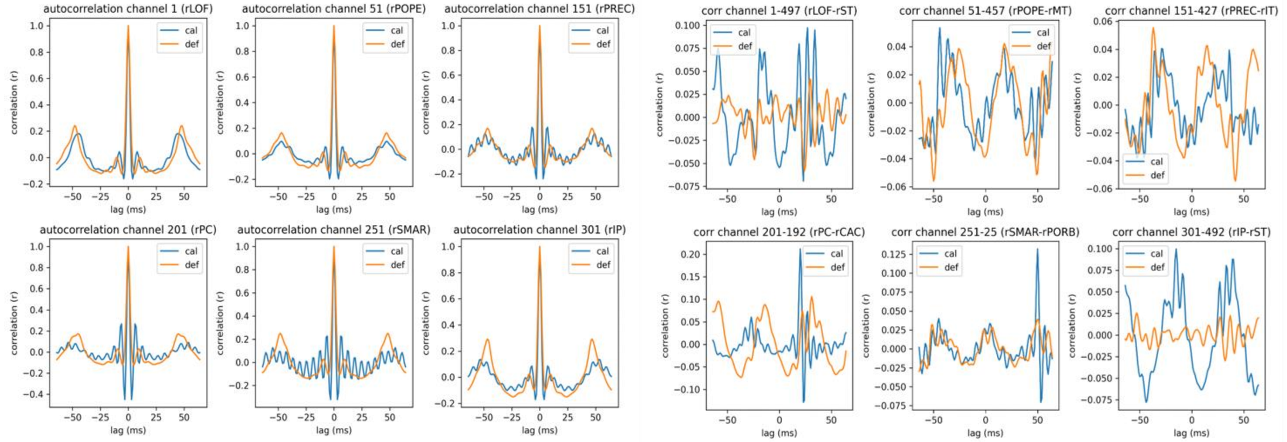


Calibrated model

Default model



Time-correlation of signals from different channels



Gaglioti et al (2024) https://doi.org/10.3390/app14020890

Cobrawap core-team

INFN
Istituto Nazionale di Fisica Nucleare

https://apegate.roma1.infn.it/
@APELab_INFN

Alessandra Cardinale
Giulia De Bonis
Cosimo Lupo
Federico Marmoreo
Pier Stanislao Paolucci

JÜLICH Forschungszentrum

Michael Denker
Sonja Grün

NYU

Robin Gutzen

cnrs

Andrew Davison

ATHENA Research & Innovation Information Technologies

Sofia Karvounari
Eleni Mathioulaki

THANK YOU!

Scientific partners

lens European Laboratory for Non-Linear Spectroscopy

IDIBAPS Institut D'Investigacions Biomèdiques August Pi i Sunyer

Universitas Studiorum Mediolanensis

Istituto Superiore di Sanità

Washington University in St. Louis