December 14th 2022

OpenViBE: an open source BCI software suite

Concluding Remarks – OpenViBE & more…

Arthur Desbois, Marie-Constance Corsi

ARAMIS team, Paris Brain Institute

PART 3 – Concluding remarks & perspectives

3.1 – OpenViBE & more…

Prototyping, Designing, Tutorials…

- As seen in Part 2, with OpenViBE you can easily prototype and design BCI protocols and experiments

- Lots of **scenario examples and templates** are already available in the install!

```
<openvibe-3.3.0-64bit>\share\openvibe\scenarios\bci-examples
```

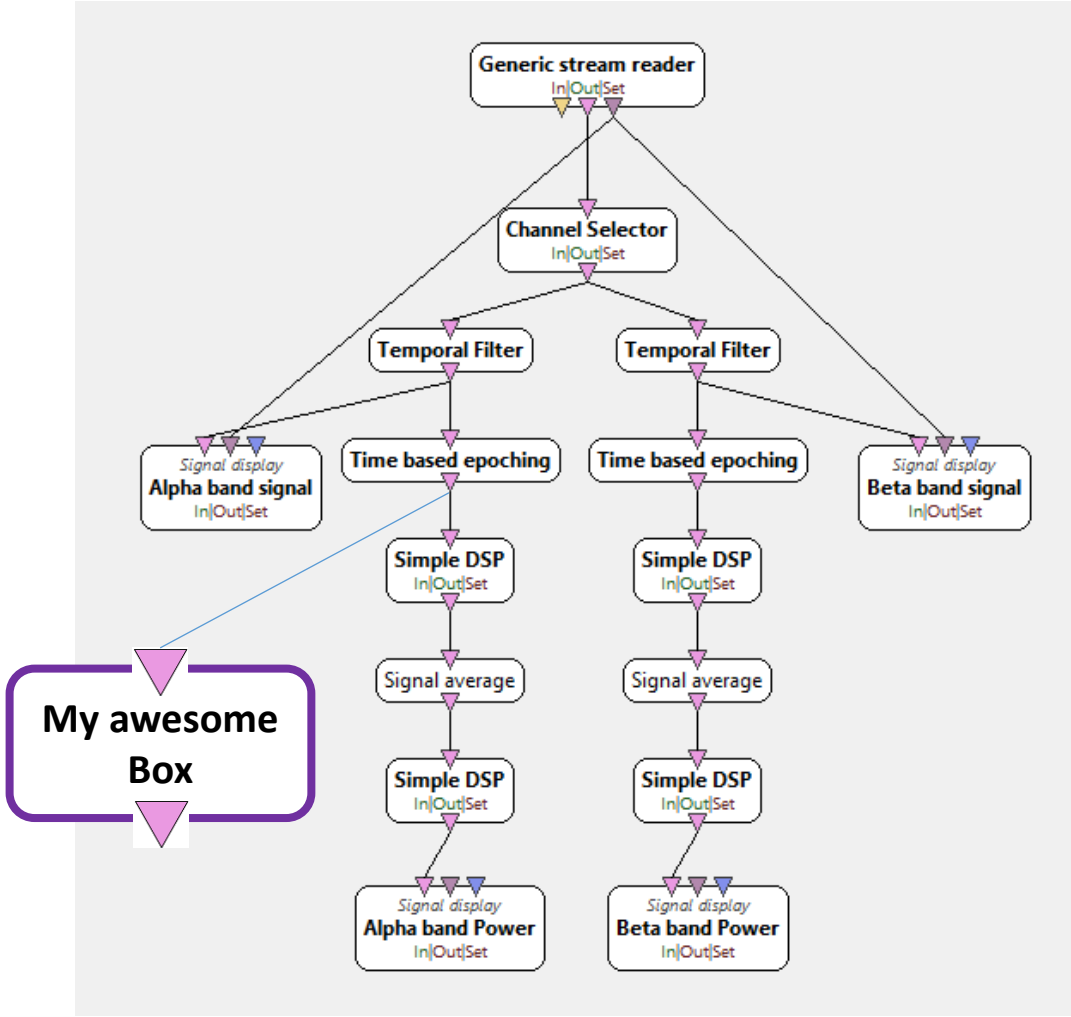
- Want to use a particular box? Tutorial scenarios are there for you:

```
<openvibe-3.3.0-64bit>\share\openvibe\scenarios\box-tutorials
```

- Check the general documentation for a great amount of info:
  http://openvibe.inria.fr/documentation-index/

Box Development

- So, you want to develop a **new processing box**?



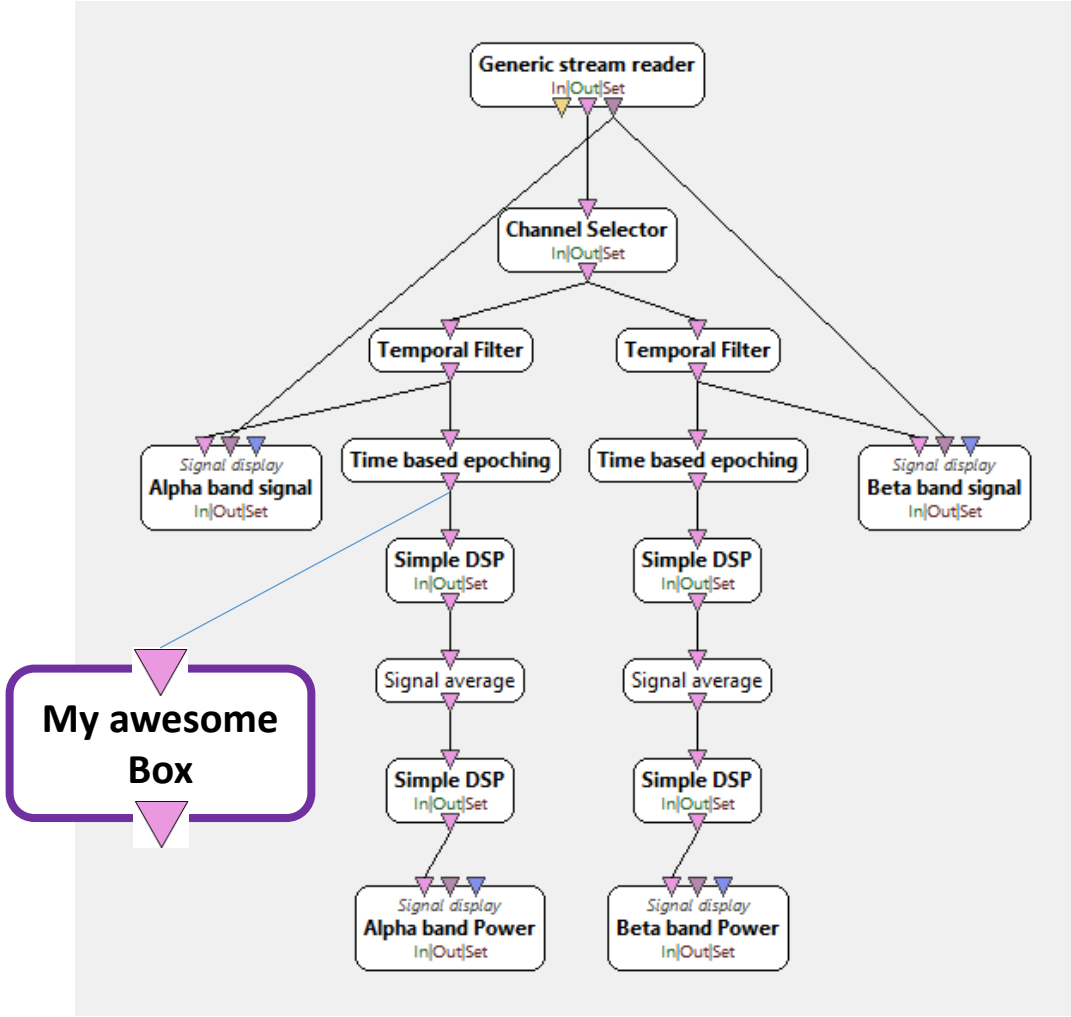
Box Development

- So, you want to develop a **new processing box**?

- **First step**:

  check if an existing box has what you need…

  … or if you can do what you want using a combination of existing boxes.



Box Development

- So, you want to develop a **new processing box**?

- **First step**:

  check if an existing box has what you need…

  … or if you can do what you want using a combination of existing boxes.

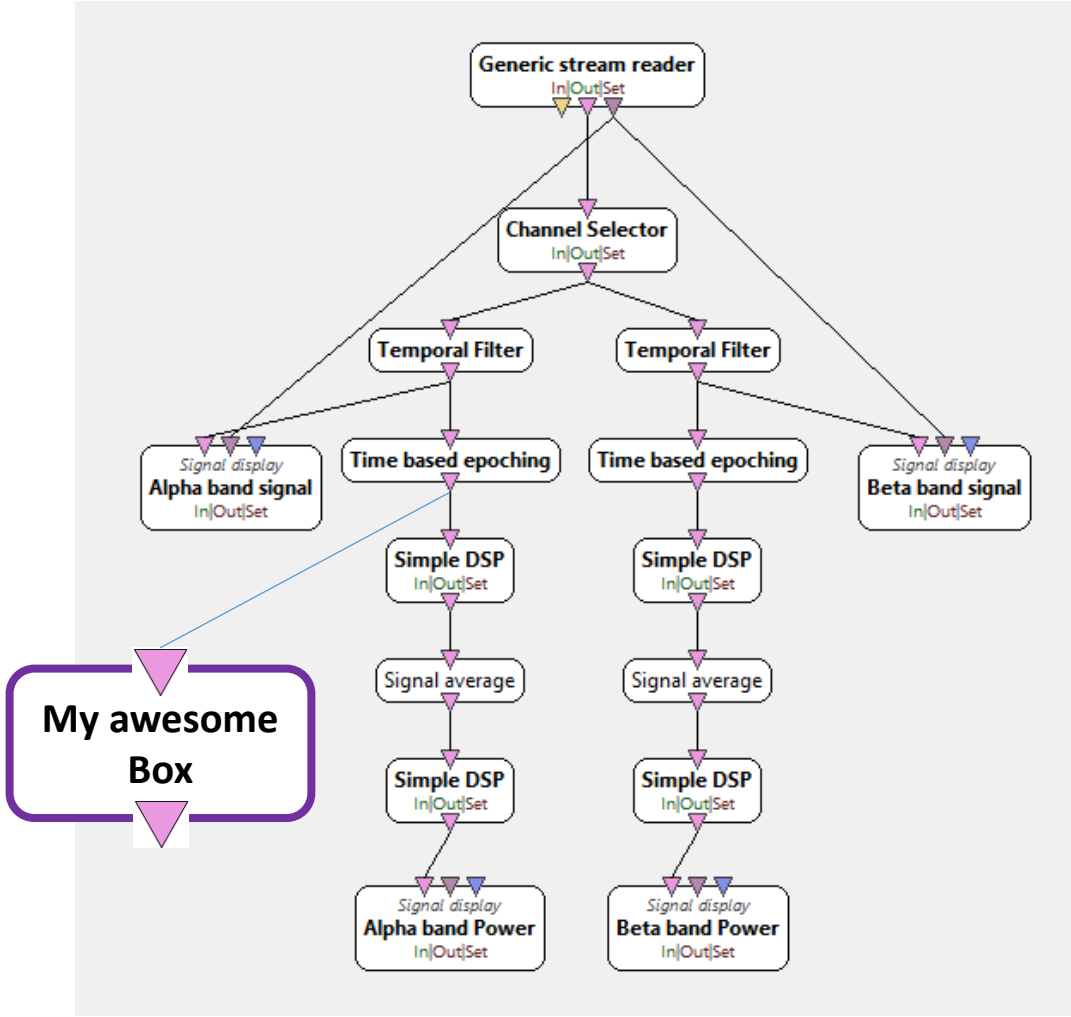
- **If not - then**:

  Do you want a quick & flexible prototype?

  → box calling **Python/MATLAB scripts**

  … or a fine-tuned optimized algorithm?

  → **C++ Box & Algorithm classes**



Box Development - Python/MATLAB

- Using Python/MATLAB scripts in OpenViBE scenarios

Use cases:

- Need for a quick proof-of-concept (e.g. signal processing)
- Don't want/need to code in C++
- Python/MATLAB implementation is already perfect
- Need specific libraries (numpy, scikit-learn…)

➔ http://openvibe.inria.fr/tutorial-using-matlab-with-openvibe/

➔ http://openvibe.inria.fr/tutorial-using-python-with-openvibe/

- Great Python tutorial: (courtesy of MENSIA)

➔ http://openvibe.inria.fr/openvibe/wp-content/uploads/2016/06/Quick-prototyping-in-OpenViBE-with-Python.pdf

Box Development - C++

- Developing C++ OpenViBE boxes

Use cases:

- Need speed!
- Complete integration with OpenViBE, contribution to the open-source project

➔ http://openvibe.inria.fr/build-instructions/

- 2016 Tutorial:

➔ http://openvibe.inria.fr/openvibe/wp-content/uploads/2016/06/jl_hacking_boxes_2016.pdf

Box Development - C++

- ## Skeleton generator

  Simplest, fastest, go-to solution for beginners…

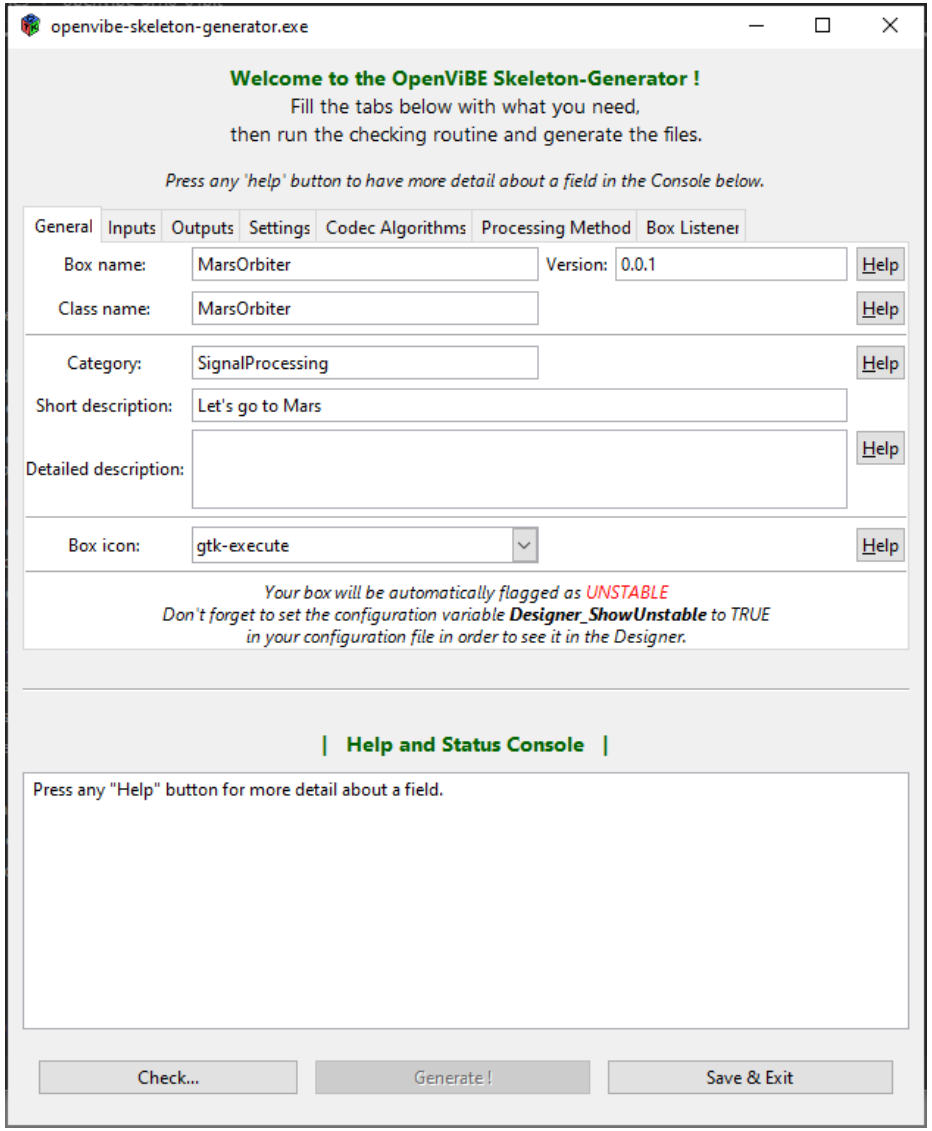
  `openvibe-skeleton-generator.cmd`

- GUI helping with creating the bare minimum a box needs, with given inputs/outputs, parameters, etc.

  ➔ All the “OpenViBE glue” is here!

  You “just” need to add your specific code.

http://openvibe.inria.fr/tutorial-1-implementing-a-signal-processing-box/



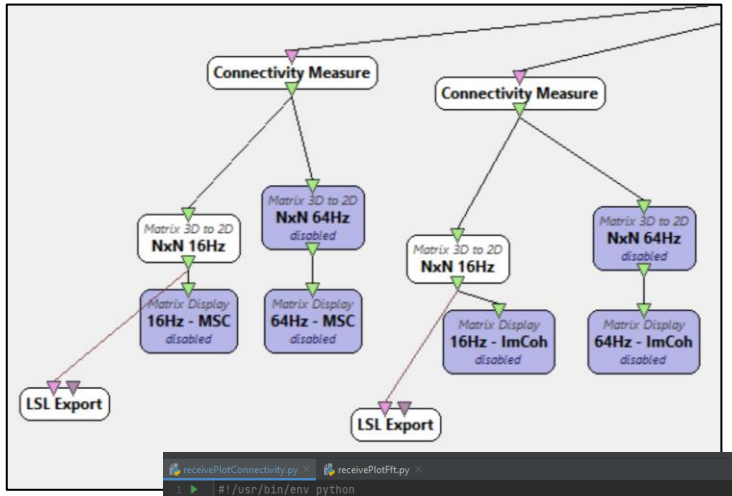
Tip: take inspiration from existing boxes…!

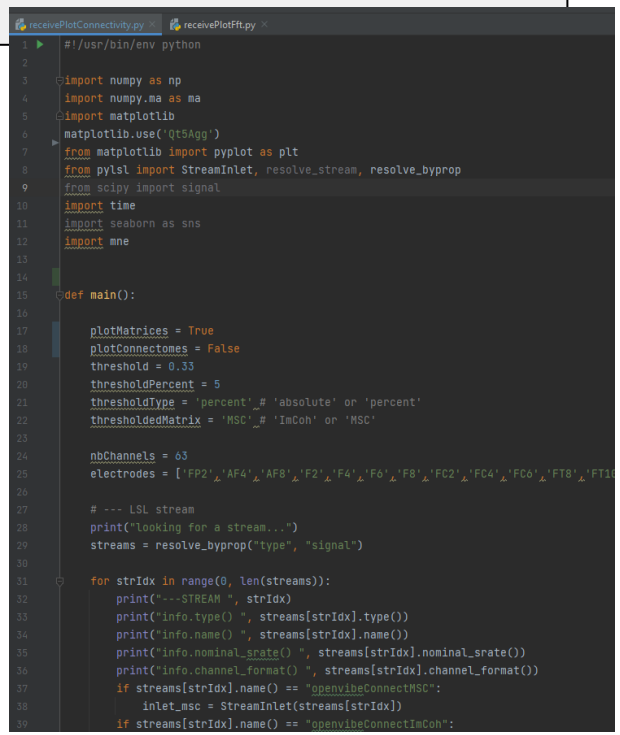
Interfacing with external software

- Examples: interface w/ virtual reality products, video games, external data visualization toolboxes…
- Various available solutions to stream data/events between OpenViBE and external apps.
  - **VRPN**
  - **TCP/IP**
  - **LSL** (Lab Streaming Layer)
  - **Python/MATLAB boxes**
- Demo using LSL to visualize Connectivity/Adjacency Matrices using an external Python script
- Code and example scenarios:

  ➔https://github.com/AsteroidShrub/openVibe-Lsl-Demo

Interfacing with external software



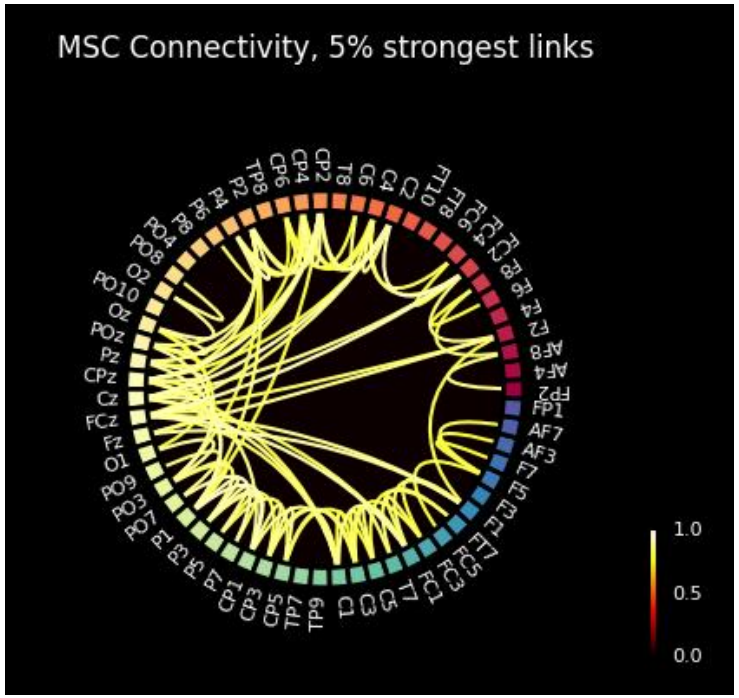


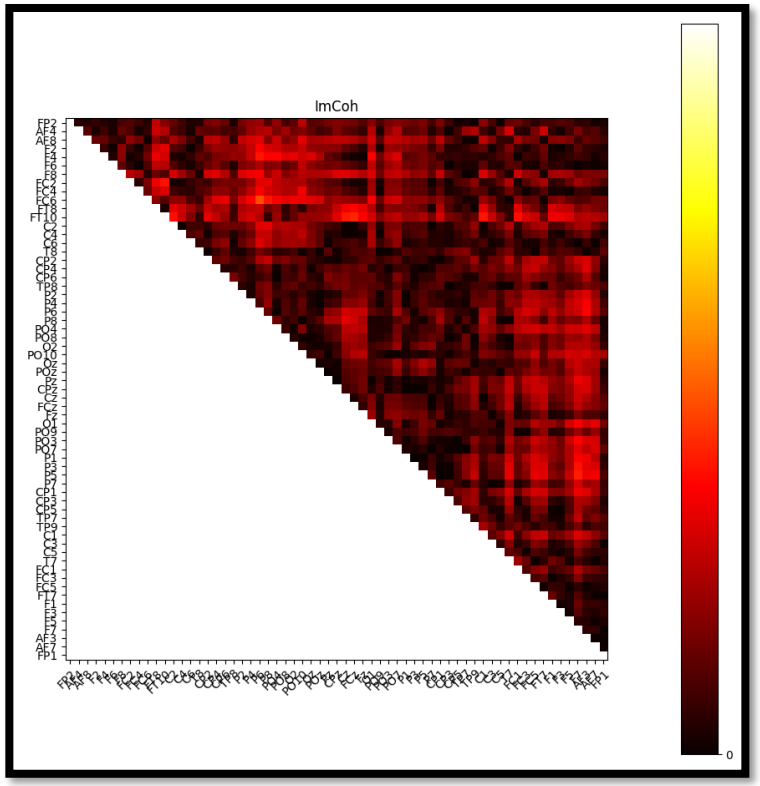
- OpenViBE “LSL Export” Box

Connectivity Measurement Box

- + Python scripting (using pylsl)

  ➔ external matrices analysis/plotting



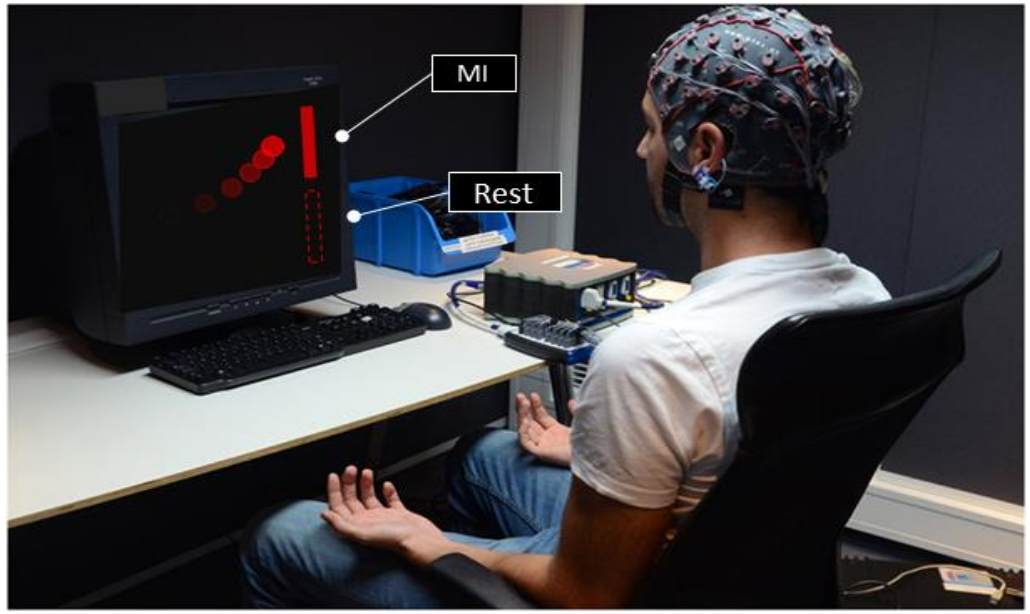


PART 3 – Concluding remarks & perspectives

3.2 – HappyFeat

HappyFeat – BCI in a clinical setting

- **Clinical setting = time constraints**
  - Setting up the EEG sensors
  - BCI/MI experiments can be long and strenuous



HappyFeat – BCI in a clinical setting

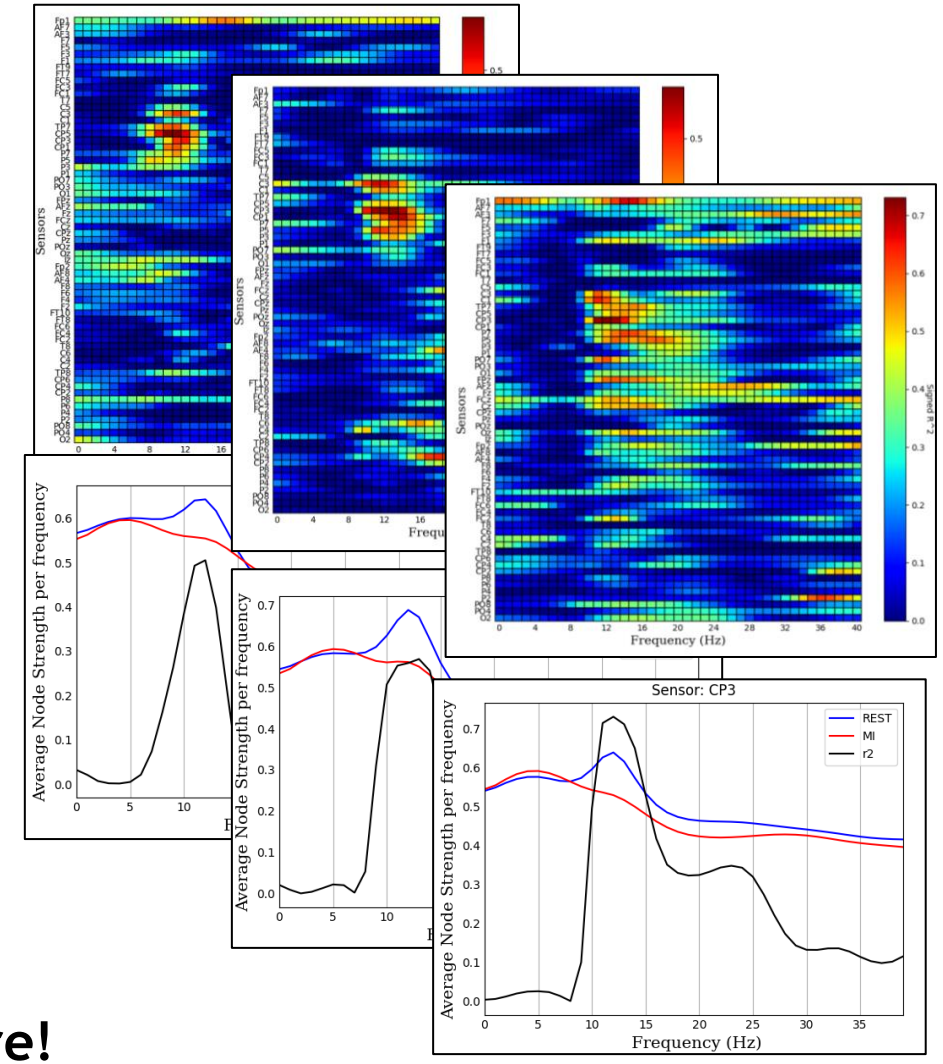
- **Features of interest (FOI)**
  - Finding adequate FOIs is a **difficult** and **crucial step**
  - After acquisition of MI trials EEG signals, an **analysis phase** is needed to select best FOIs.

➔ Usually involving other scientific softwares (i.e. MATLAB)

  - If the analysis phase is too long, a lot can change in the meantime:
    - EEG sensors impedances
    - Subject's brain behaviors
    - Subject's fatigue & motivation drop

➔ Signal characteristics & Features may differ a lot between Acq/Training phase and Online phase…

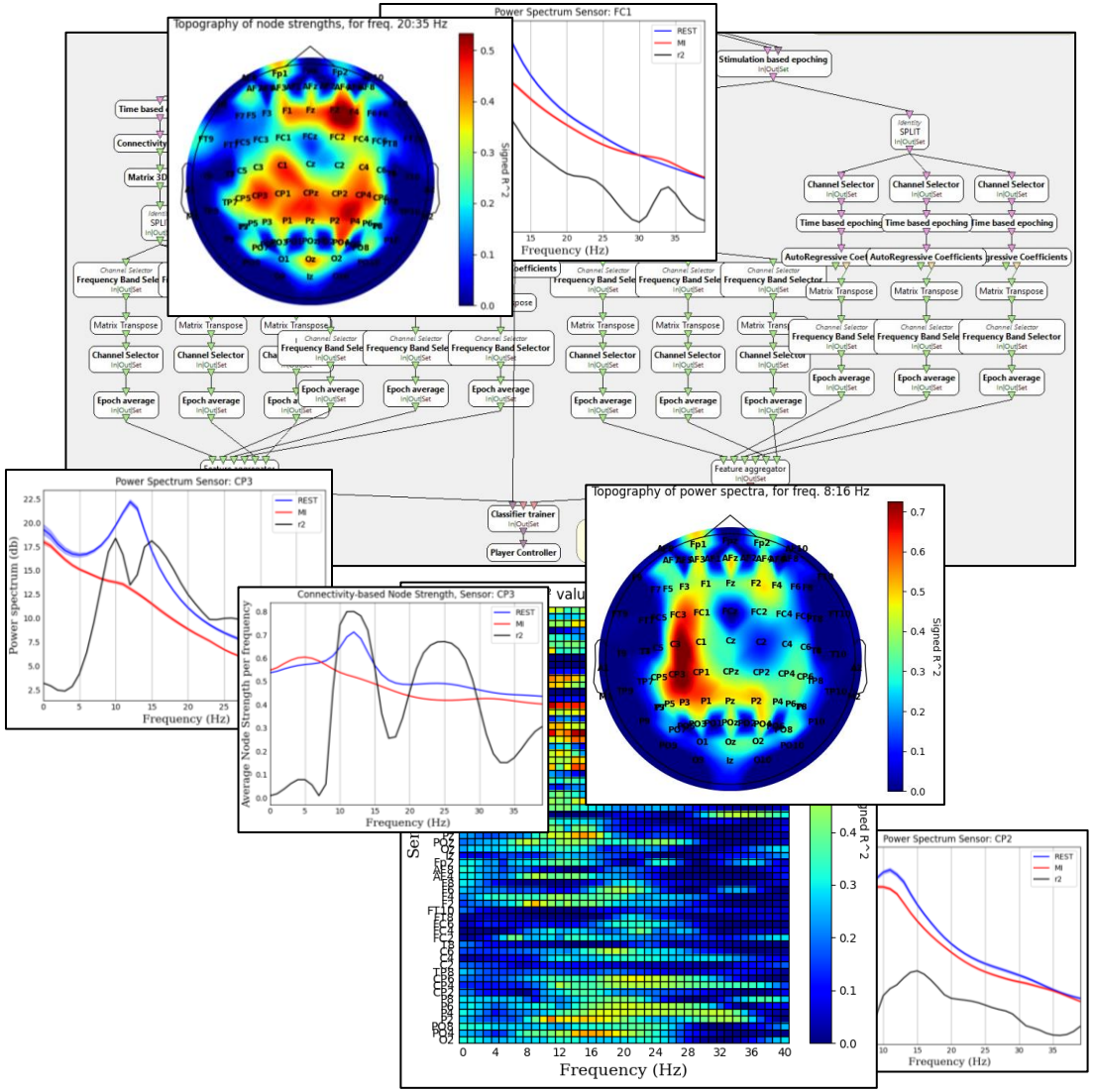
➔ **Your trained classifier may not be relevant anymore!**



HappyFeat - BCI in a clinical setting

- **Features of interest (FOI)**
  - The "**analysis phase**" involves a lot of manipulations:
    - Setting up "feature extraction" scenarios in OpenViBE…
    - Selecting FOIs through visualization tools…
    - Setting up & running training scenarios in OpenViBE…
    - … and maybe going again through those steps multiple times until "correct" features have been selected, or to account for inter-run variability

➔ Tedious, error-prone, hard to achieve in a limited time



("Please send help", Desbois 2022)

HappyFeat - Main Concepts

Python-based framework for facilitating MI pipelines

Main focus:

making Feature Selection

& Training phases

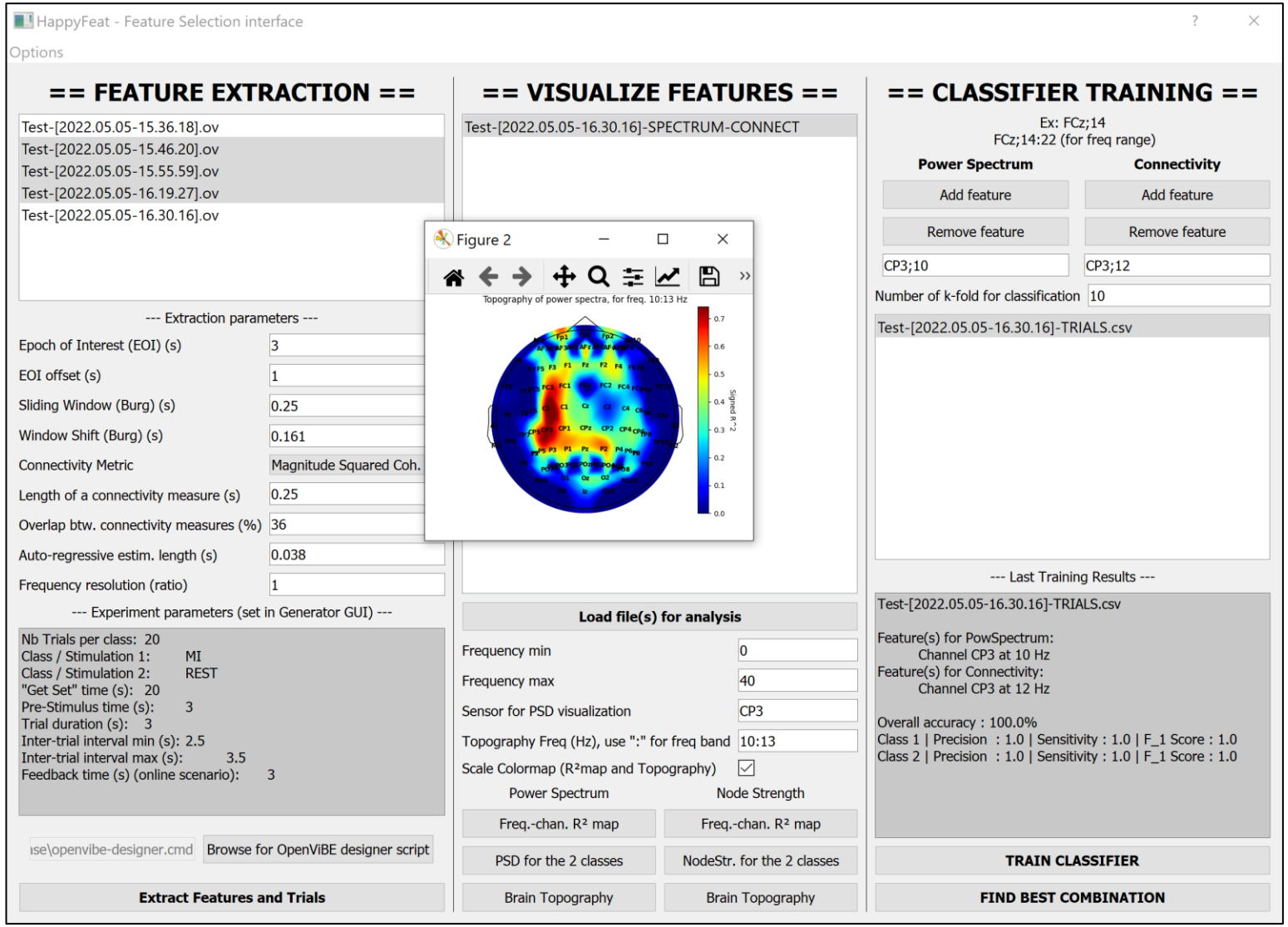
easy & fast.

Analyze,

select your Features

& train your classifier

in less than 5 minutes!



HappyFeat – Main Concepts

- **Multiple pipelines available**
  - Available features for classification: Spectral Power, Connectivity-based network metrics
  - ... It's also possible to train the classifier using a fusion of different types of features.
- **Feature extraction**
  - Easy access to experiment & signal processing parameters.
  - Using either **pre-recorded signals** or **live signals, on-the-fly during acquisition phase.**
- **Visual Analysis for feature selection**
  - R^2 maps, PSD comparison across trials, time/freq. ERD/ERS analysis, brain topography…
- **Classifier training**
  - **Run as many training attempts as needed**, using different features, in only a few clicks.
  - Concatenate trials from multiple recorded sessions
  - OpenViBE scenarios are **updated and launched on-the-fly**.
  - Automatically generates/updates the "online classification" scenario.

HappyFeat – Main Concepts

- **Key feats & mechanisms:**
  - **Clean, risk-free environment**

    ➔ avoid unnecessary & error-prone manipulations.
  - **Trial-and-error oriented workflow**

    ➔ all steps can be repeated quickly & as many times as needed
  - **Unified "dashboard" GUI**
  - **OpenViBE** used in the background, as a processing engine.

    ➔ no scenario edition/manipulation necessary: everything is **automated**!

A few technical details…

Full list of dependencies:

- **Python 3.9**
  - shutils>=0.1.0
  - mne>=0.23.0
  - numpy>=1.21.1
  - pandas>=1.3.1
  - PyQt5>=5.15.6
  - statsmodels>=0.13.1
  - scipy>=1.7.1
  - spectrum>=0.8.0
  - matplotlib>=3.4.2
- **OpenViBE v3.4.0**

HappyFeat – Main Concepts

- Already available online, work-in-progress version:
  https://github.com/Inria-NERV/happyFeat

- To be continued…
  - **More pipeline flexibility**
  - **More network metrics based on connectivity**
  - **Associated visualization tools**
  - **Choose from different classification algorithms**
  - **Workspace/session manager** to save/load session settings

  - First official release early 2023 – stay tuned!

Part3 – Q&A

BCI Motor Imagery with OpenViBE in X-Men: First Class

Thanks for your attention! Any questions?