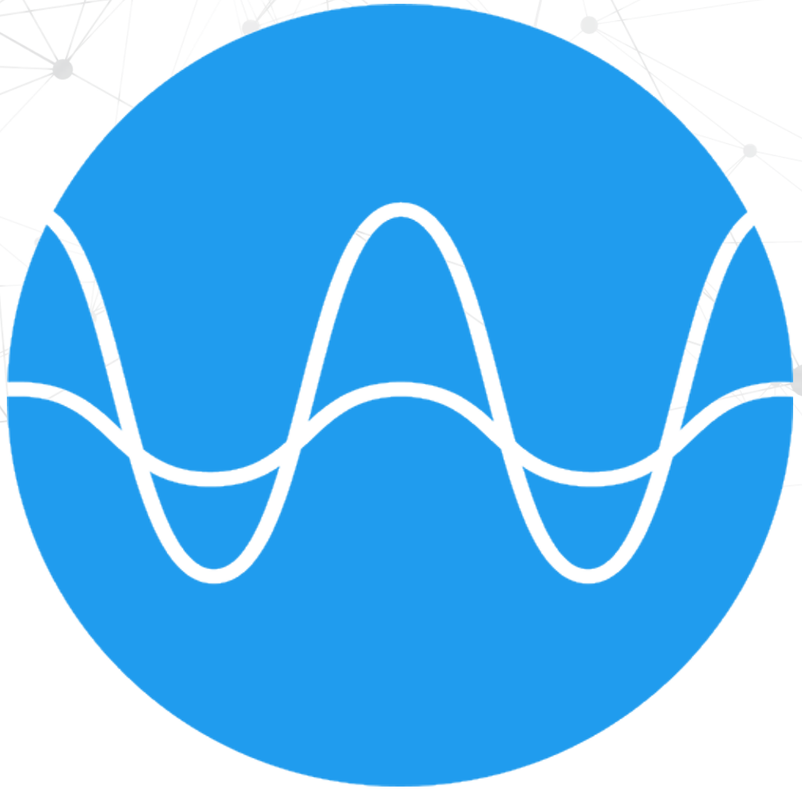
TIMEFLUX

RIEMANNIAN GEOMETRY FOR ONLINE BCI

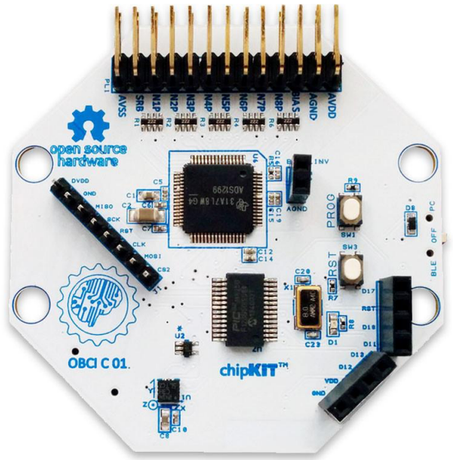


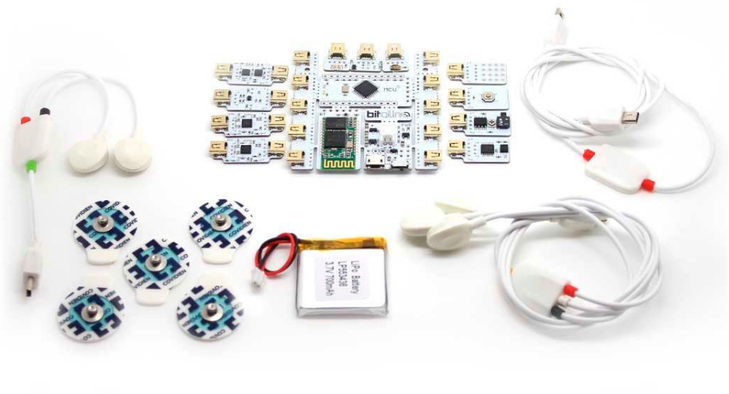
Overview

Use cases

- Data and event **acquisition** from multiple sources
- **Stimulus** presentation
- Bio-feedback
- **Brain-Computer Interfaces**
- Interactive installations
- And more :)

It works with your devices





In a nutshell

- Fits well within the **Python datascience** ecosystem
- Permissive **MIT license**: commercial use authorized
- Works both **offline** and **online**
- Quick **prototyping**

Easy to learn, use, and extend

- **Familiar concepts**: graphs, nodes, edges
- Relies on **industry standards**: Pandas, Xarray, Scikit-Learn, Lab Streaming Layer
- **Descriptive pipelines**: simple YAML syntax, no coding required
- **Custom nodes**: standard Python classes

Batteries included

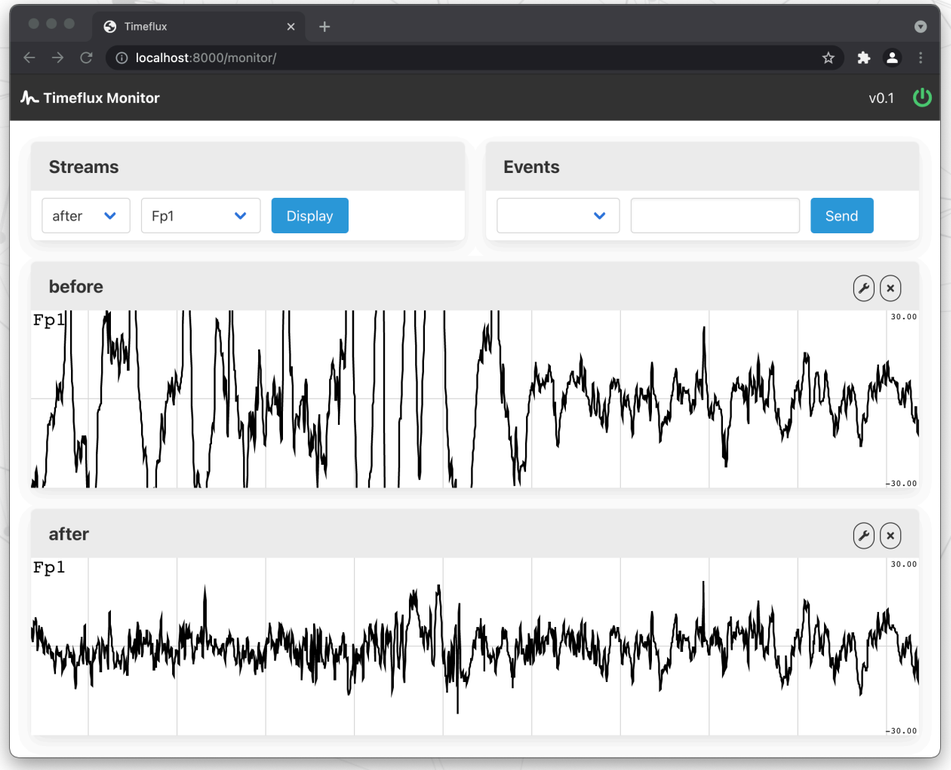
- **Networking**: Pub/Sub, Lab Streaming Layer, OSC, WebSocket
- **Recording and replay**: HDF5 file format
- **Digital Signal Processing**
- **Machine Learning**
- **User interface**: monitoring, web apps

Batteries included

- Multidimensional **matrix manipulation**: queries, transformations, expressions, epoching, windowing
- Native **device drivers**
- Sub-millisecond **synchronization**
- **Debugging** tools
- Pre and post **hooks**

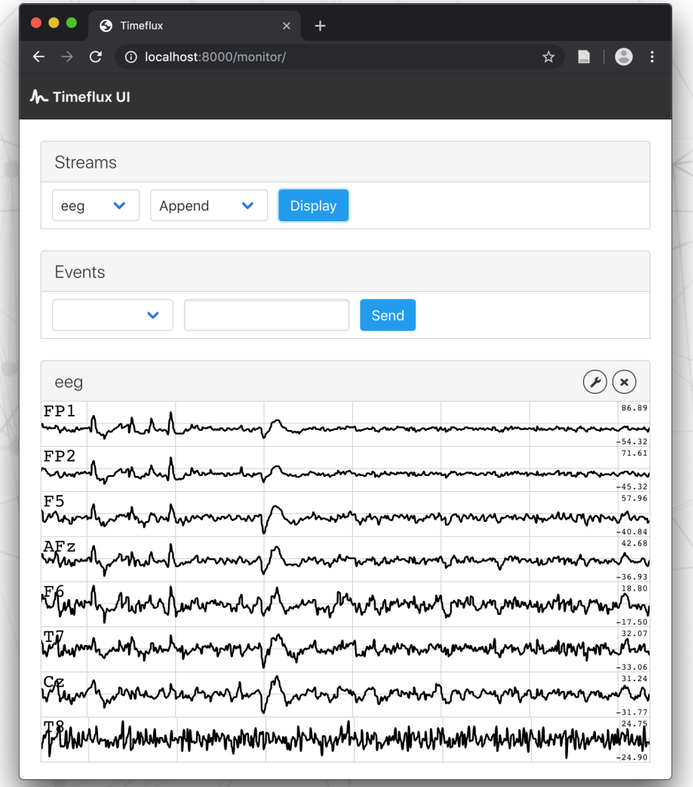
Timeflux ❤️ Riemannian Geometry

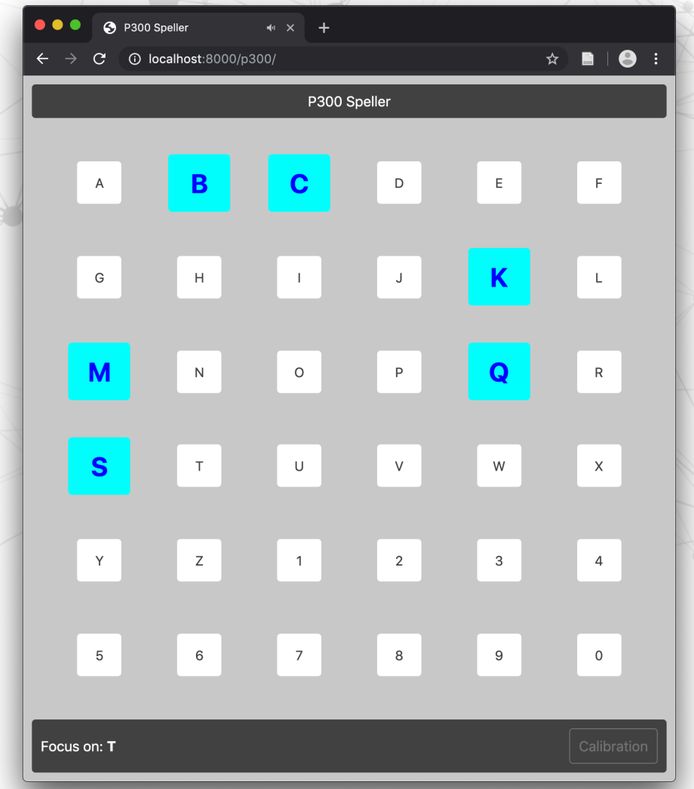
- Full **pyRiemann** integration
- Real-time **denoising** based on rASR [1]

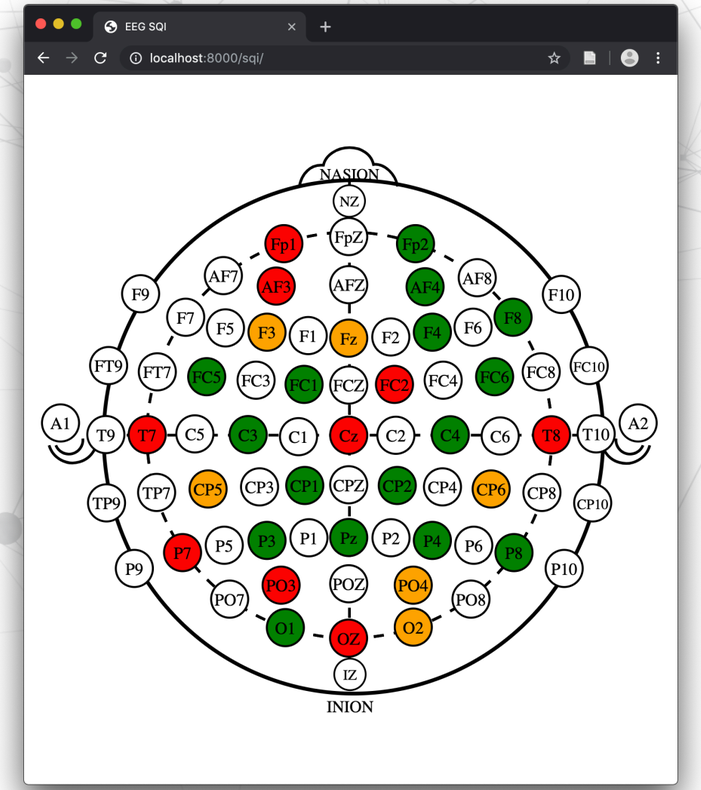


[1] Blum, Sarah, et al. “A Riemannian Modification of Artifact Subspace Reconstruction for EEG Artifact Handling.”

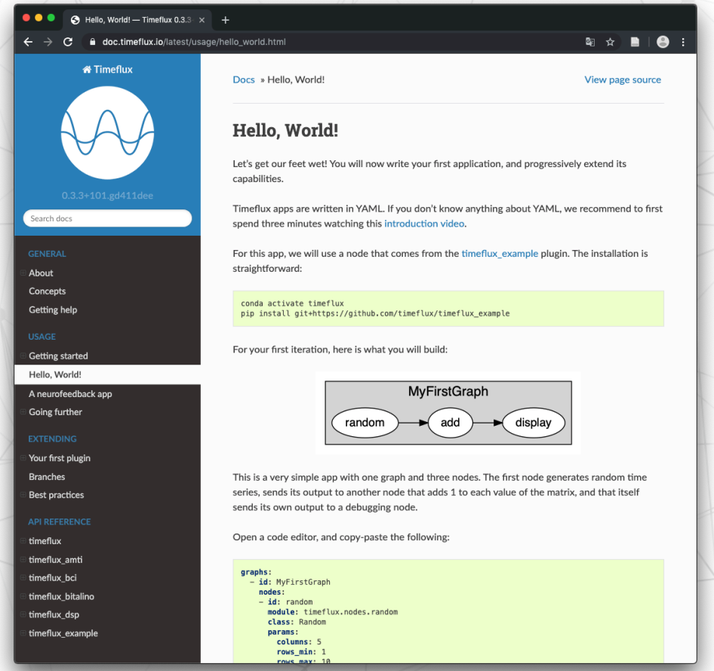
It happens in the browser

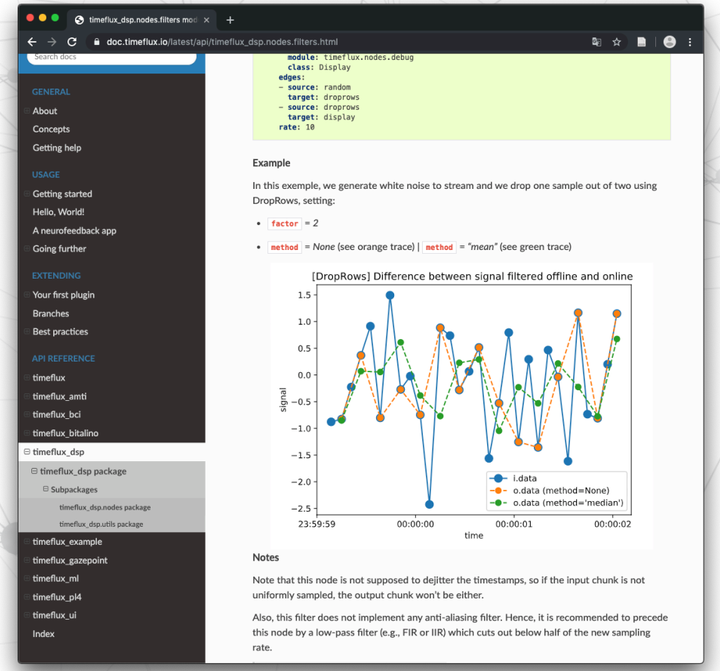






Documentation

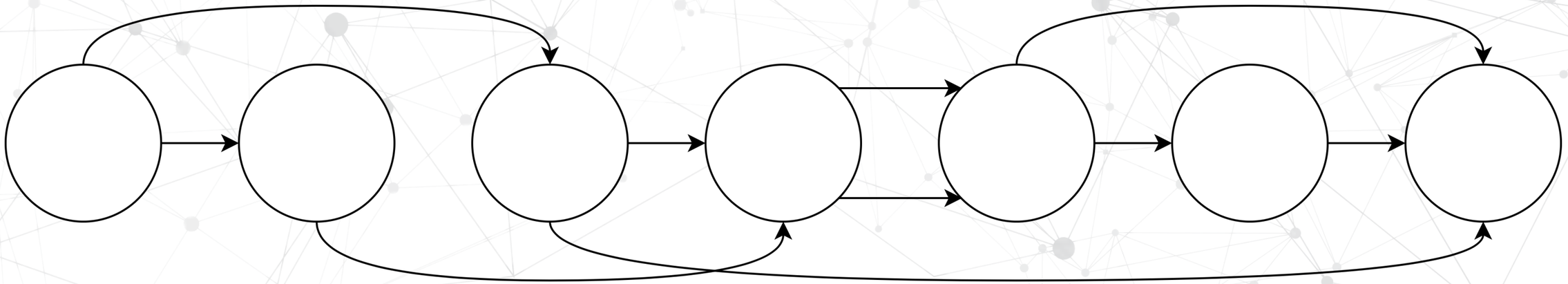




Pipelines

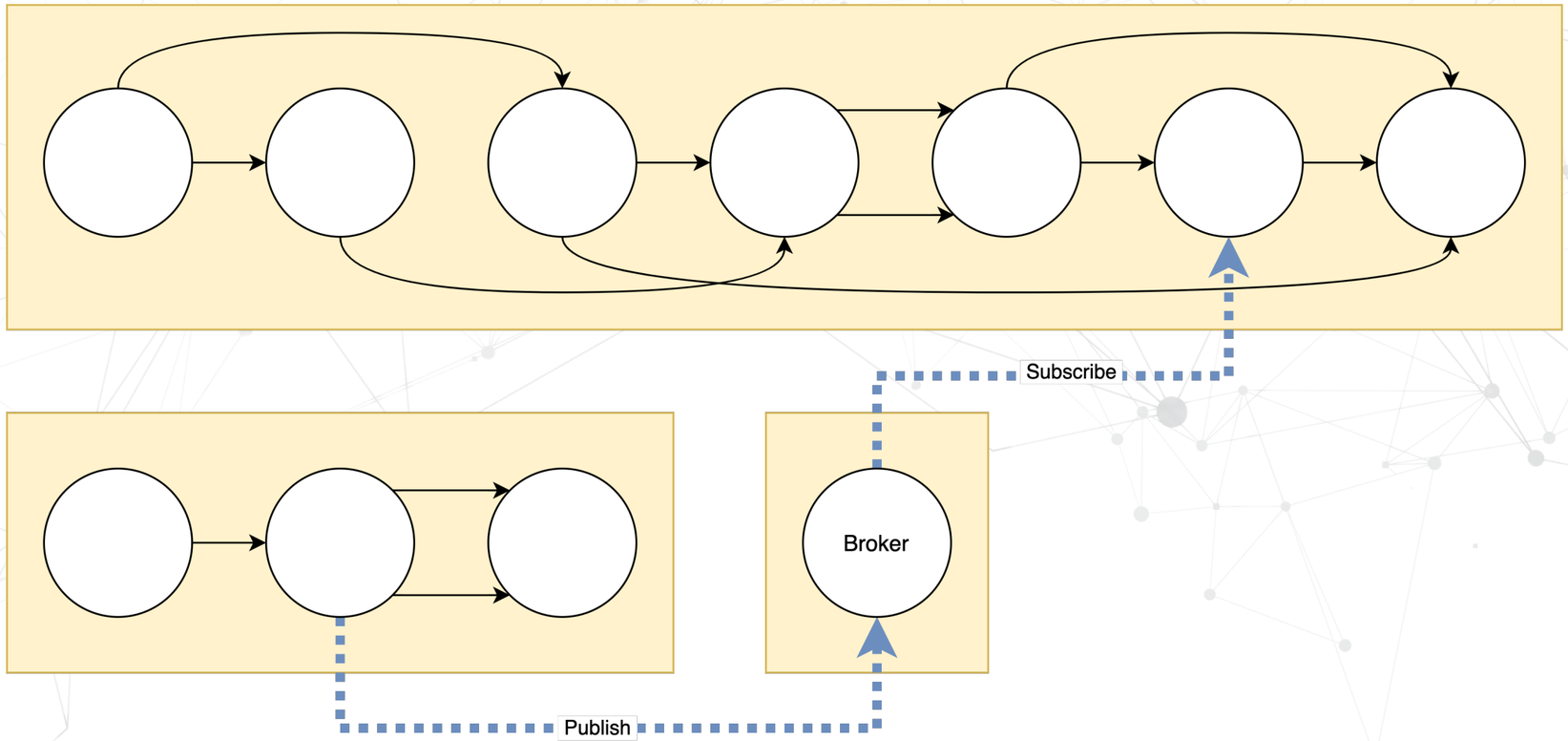
Directed Acyclic Graph (DAG)

A set of **nodes** connected by **edges**, where information **flows** in a given direction, **without any loop**.



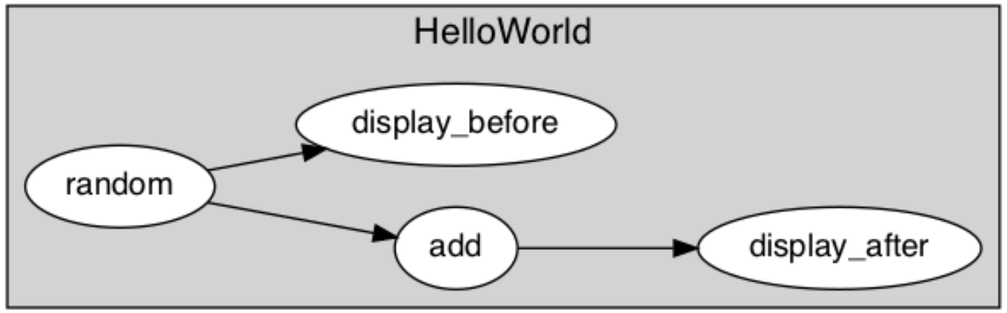
Multiple DAGs

Pub/Sub allows asynchronous loops without breaking anything.



helloworld.yaml

```yaml
graphs:
  - id: HelloWorld
    nodes:
    - id: random
      module: timeflux.nodes.random
      class: Random
    - id: add
      module: timeflux_example.nodes.arithmetic
      class: Add
      params:
        value: 1
    - id: display_before
      module: timeflux.nodes.debug
      class: Display
    - id: display_after
      module: timeflux.nodes.debug
      class: Display
    edges:
    - source: random
      target: add
    - source: random
      target: display_before
    - source: add
      target: display_after
    rate: 1
```



Run it!

```
timeflux -d helloworld.yaml
```

Interfaces

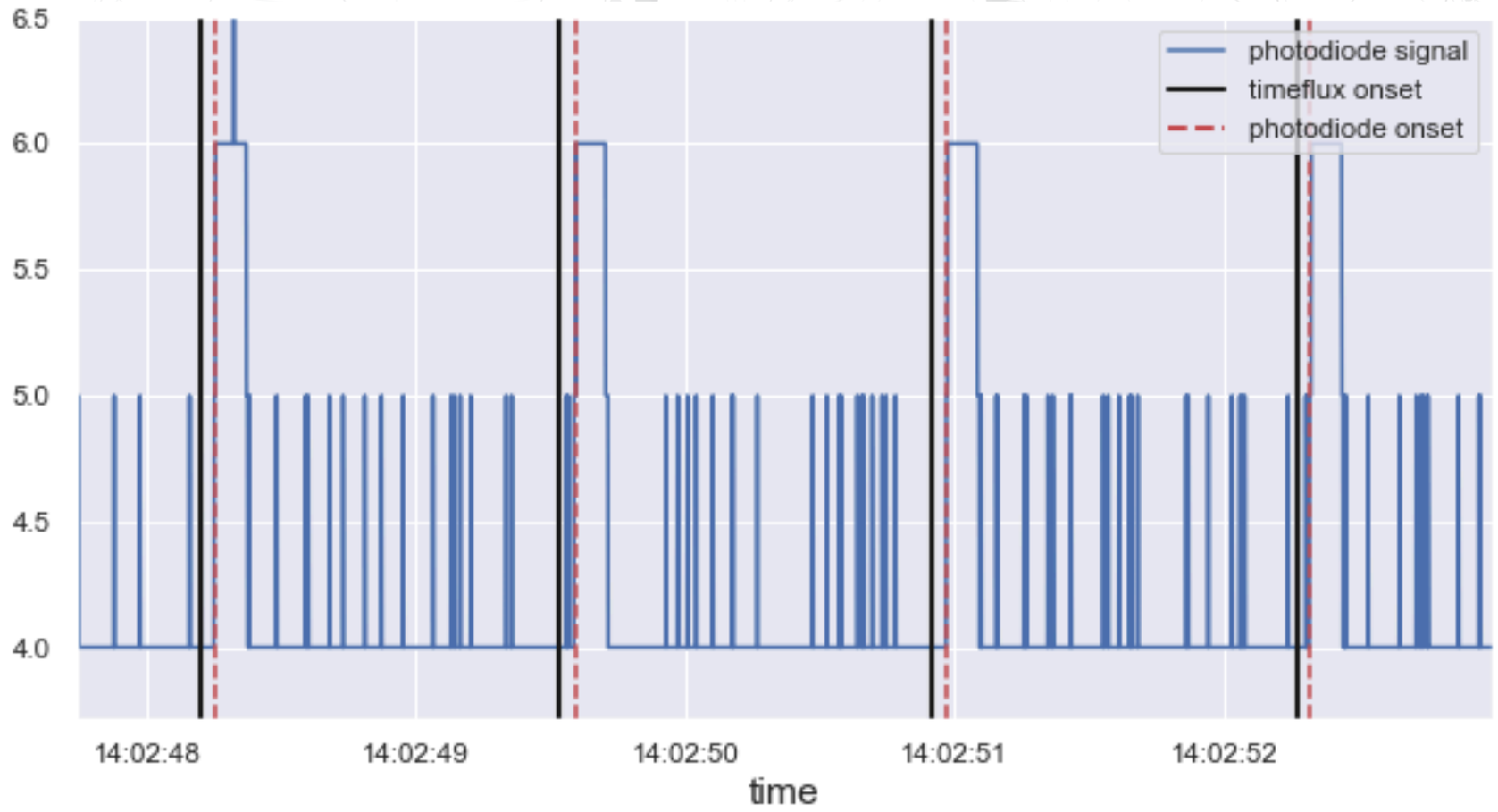
Timeflux.js

- A **JavaScript** API for:
  - Building **user interfaces** available from a browser
  - Receiving and sending data **streams** and **events**
  - Delivering **precisely scheduled** stimuli: suitable for SSVEP and ERP research

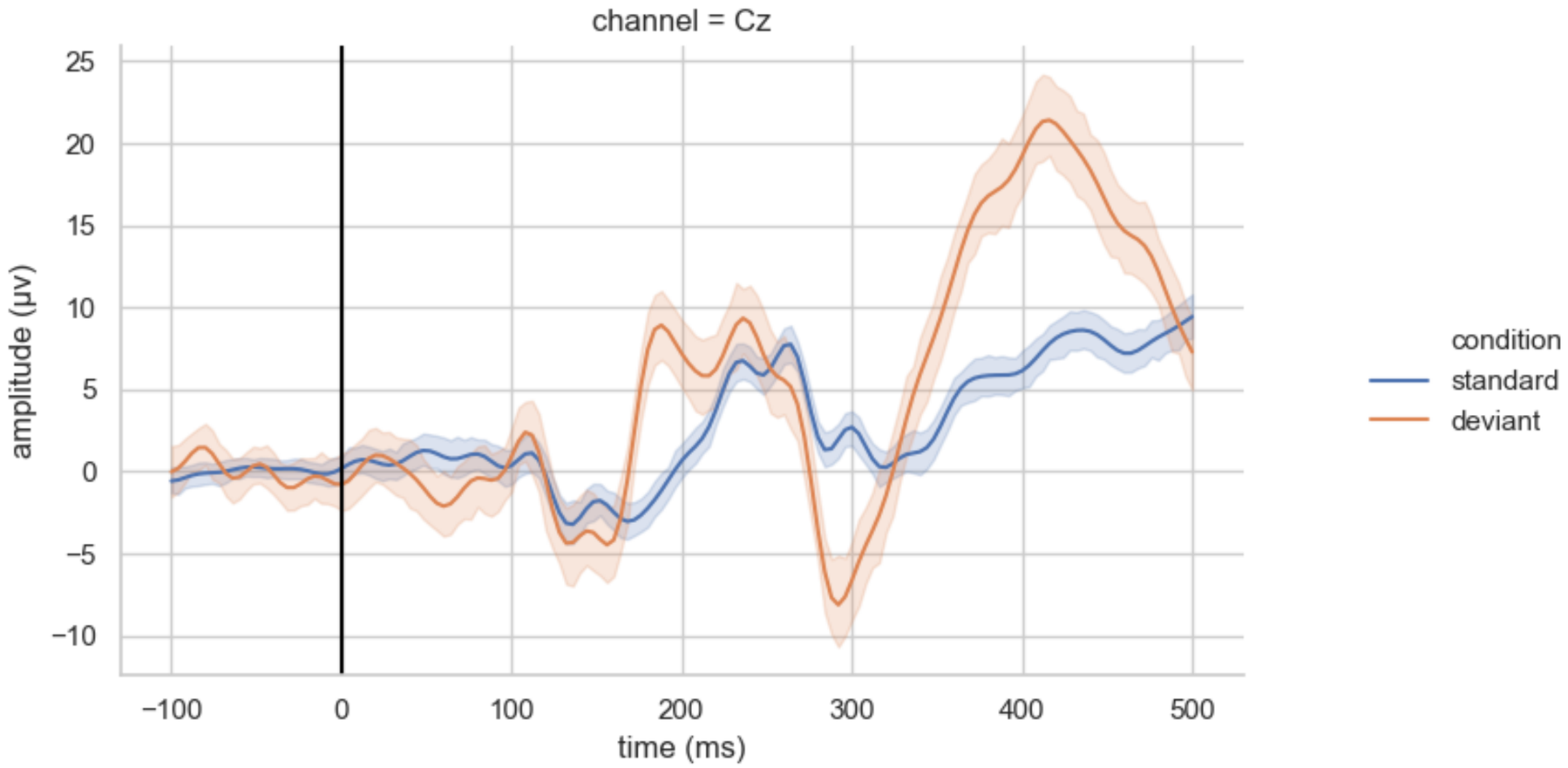
Stimulus presentation

- **Perfect timing** in a browser is **hard**
- But Timeflux.js makes it **easy**!
- Schedule **repeating** stimuli or **one-time** tasks
- Know **exactly** when the stimulus has been displayed
- Well tested in **Chrome**. Other browsers *may* have issues

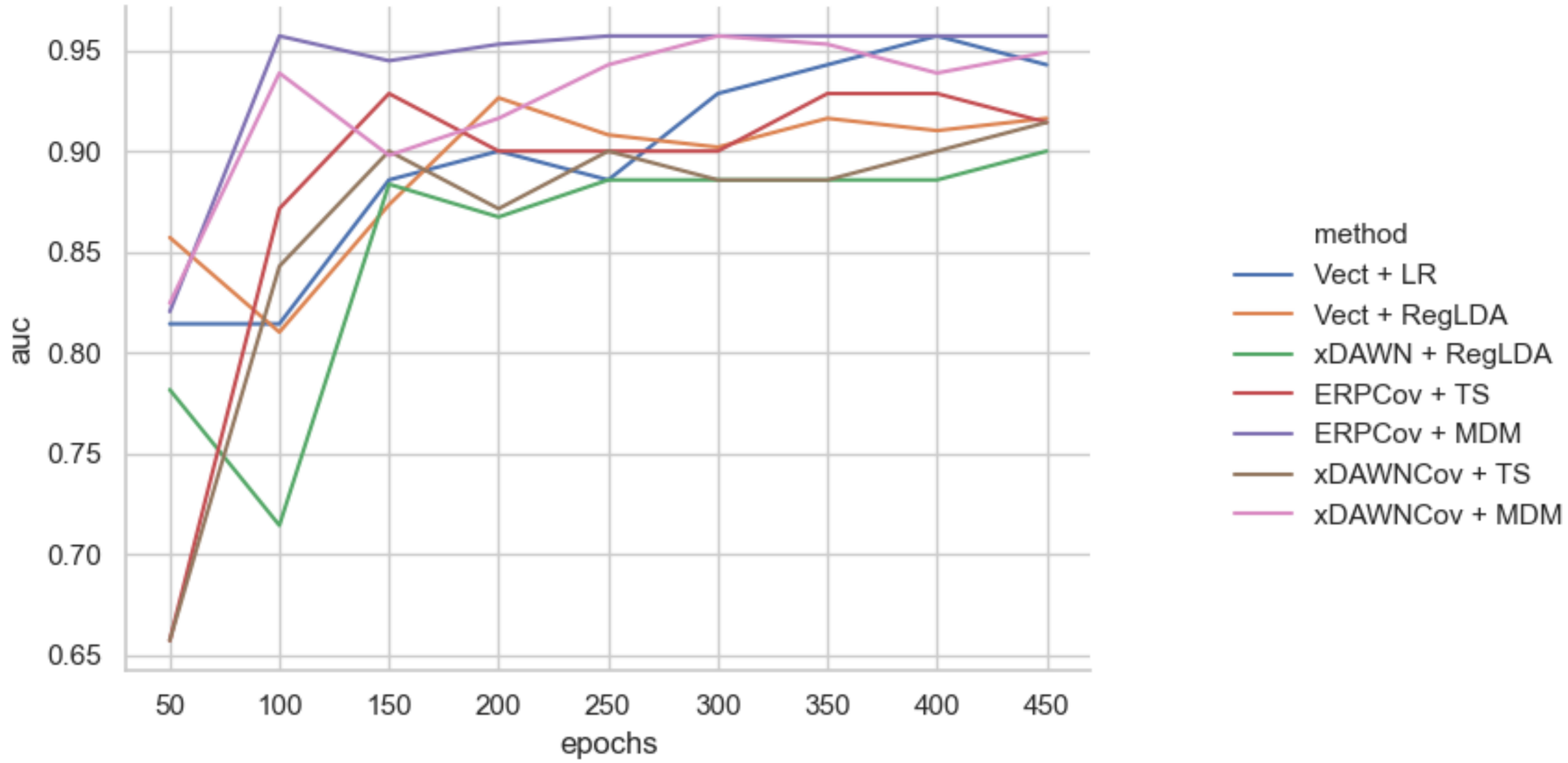
Validation: software event VS photodiode



Validation: evoked potentials



Validation: single-trial ERP classification



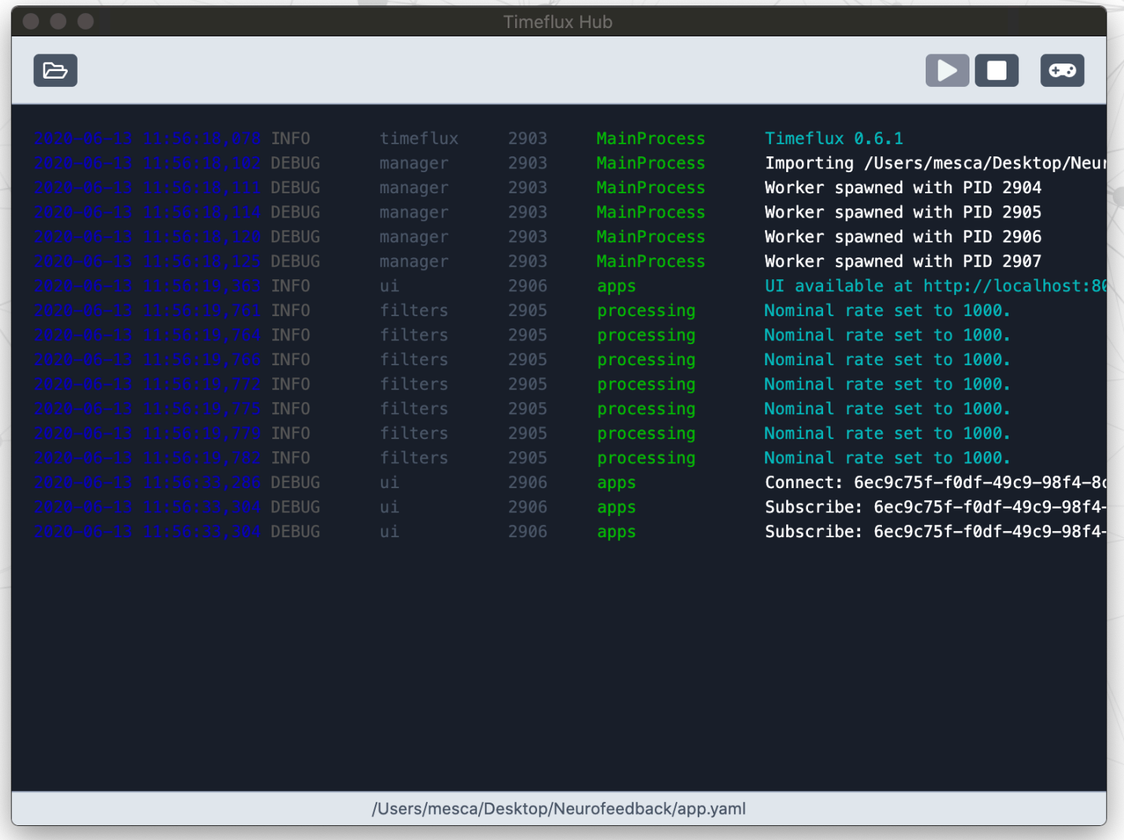
Demo!

Coming soon

Plug-and-play pipelines

- Standard **BCI paradigms** (EEG):
  - SSVEP
  - CVEP
  - P300
  - Motor Imagery
- **Neurofeedback** (EEG)
- **Cardiac coherence** (ECG, PPG)
- **Gesture detection** (EMG)

Timeflux Hub



Conclusion

Getting help

- **Website**: https://timeflux.io
- **Documentation**: https://doc.timeflux.io
- **Bugs**: https://github.com/timeflux
- **Slack**: https://timeflux.io/slack
- **Email**: contact@timeflux.io

Thanks!