

Machine learning without human supervision on neuroscience data

Alexandre Gramfort
alexandre.gramfort@inria.fr
<http://alexandre.gramfort.net>



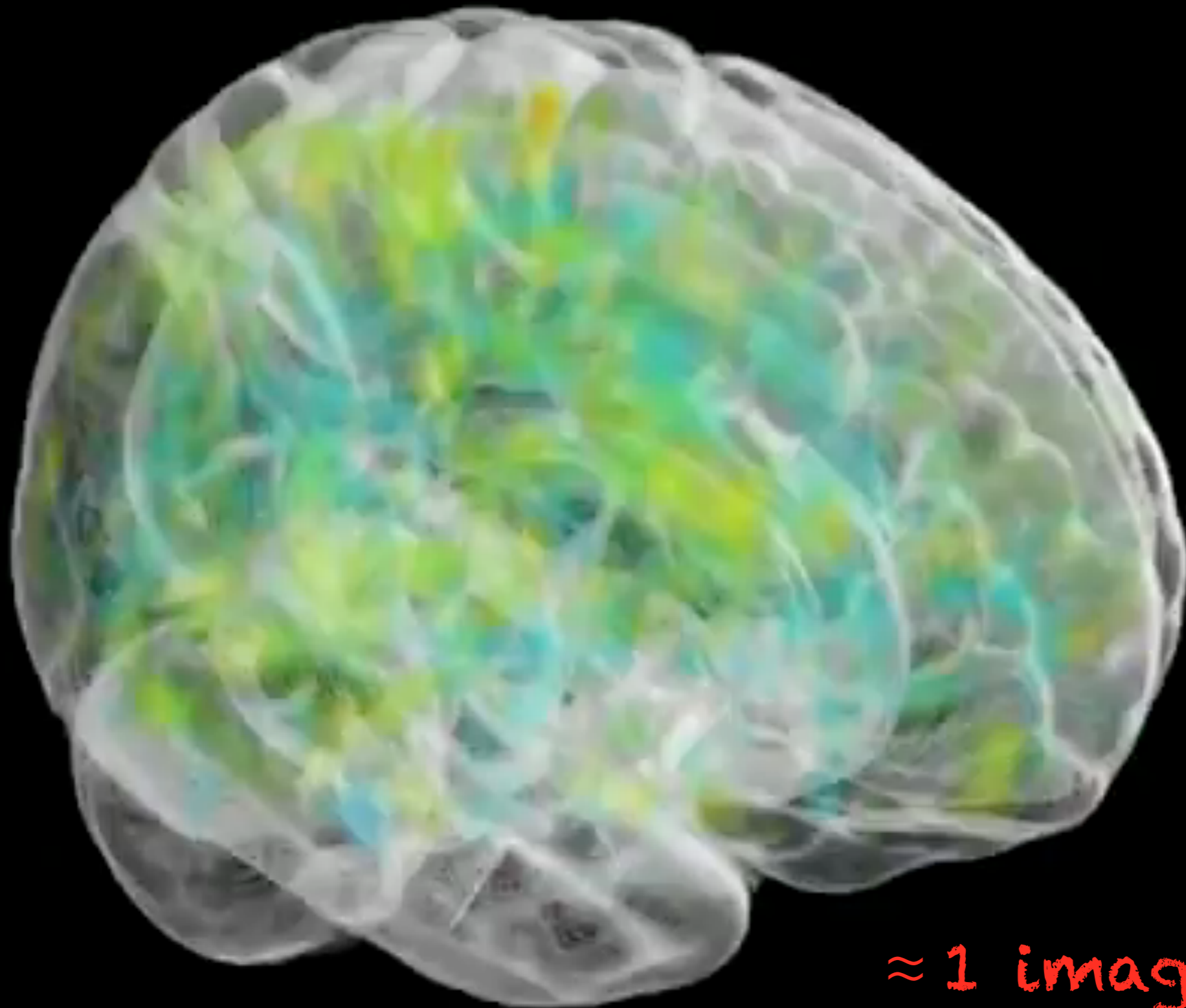
université
PARIS-SACLAY



Nantes - June 2022



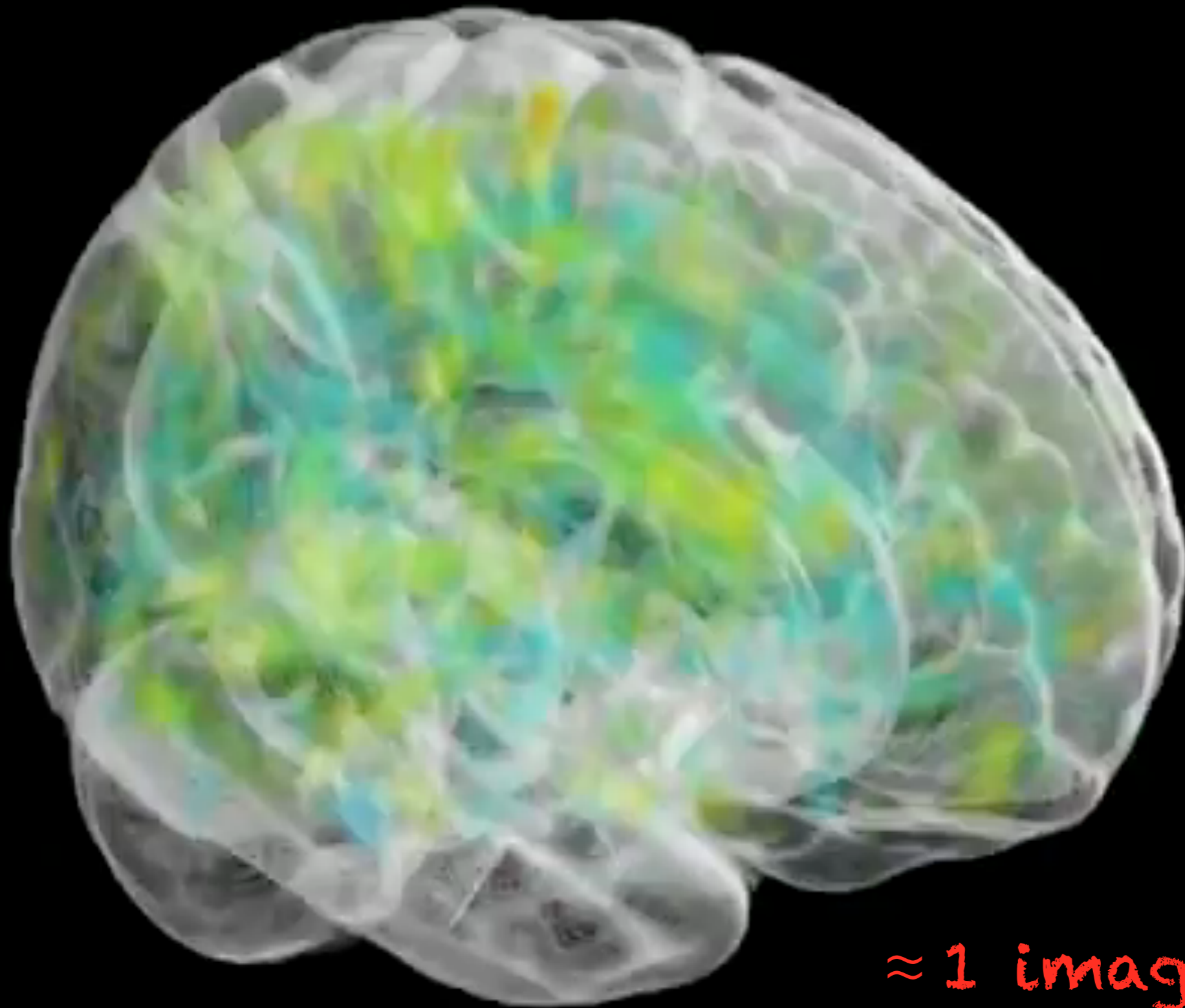
Functional MRI



$\approx 1 \text{ image} / 2s$

<http://www.youtube.com/watch?v=uhCF-zlk0jY>

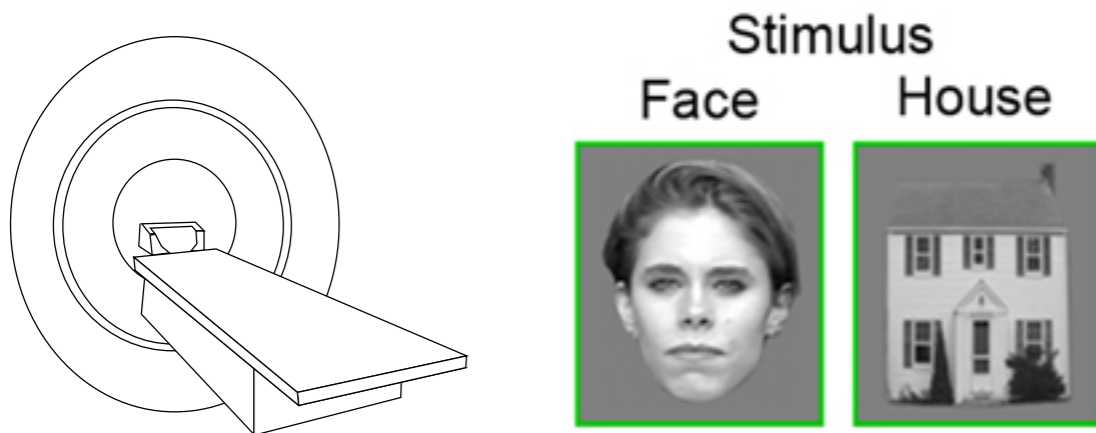
Functional MRI



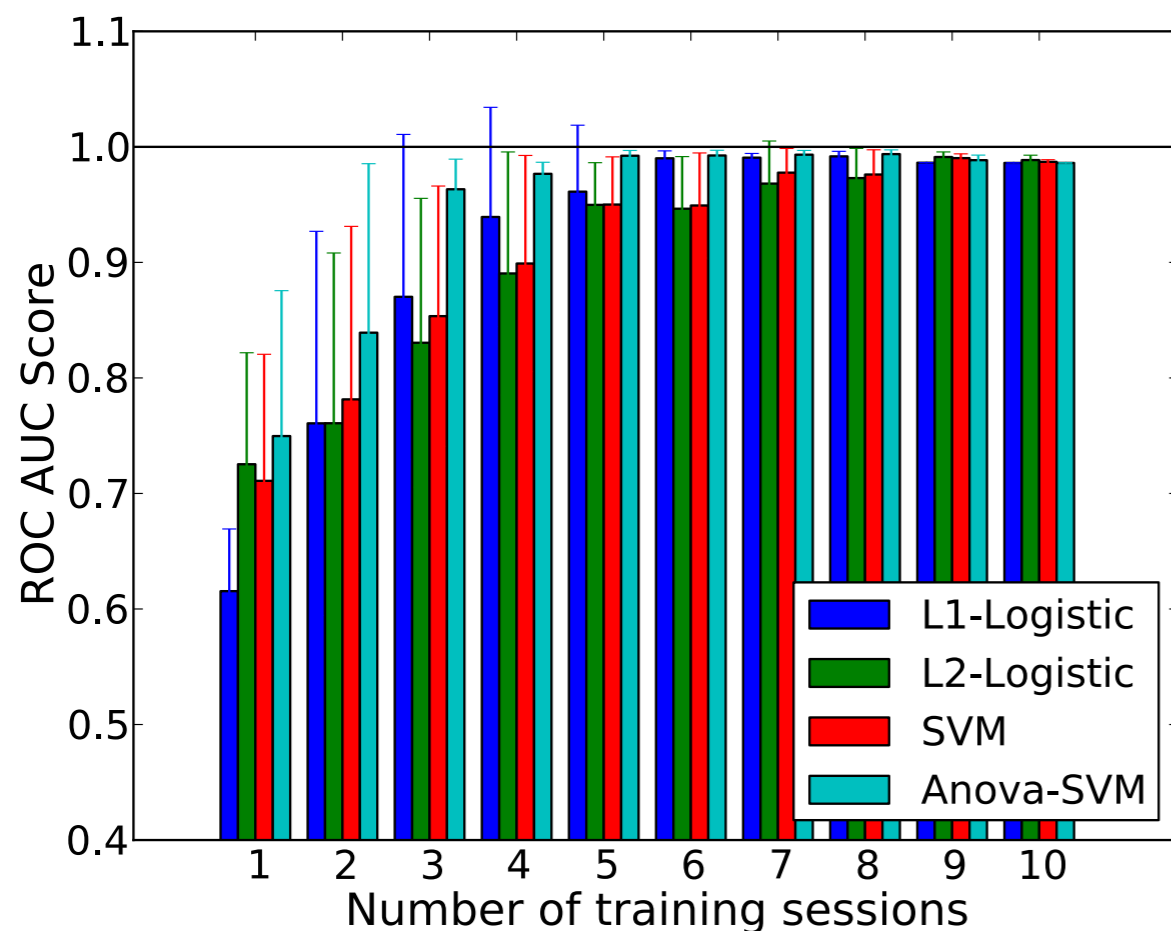
$\approx 1 \text{ image} / 2s$

<http://www.youtube.com/watch?v=uhCF-zlk0jY>

Supervised Learning on fMRI: Face vs. House



Binary classification results



[Gramfort et al. 2011]

- The more data the better
- Almost no noise

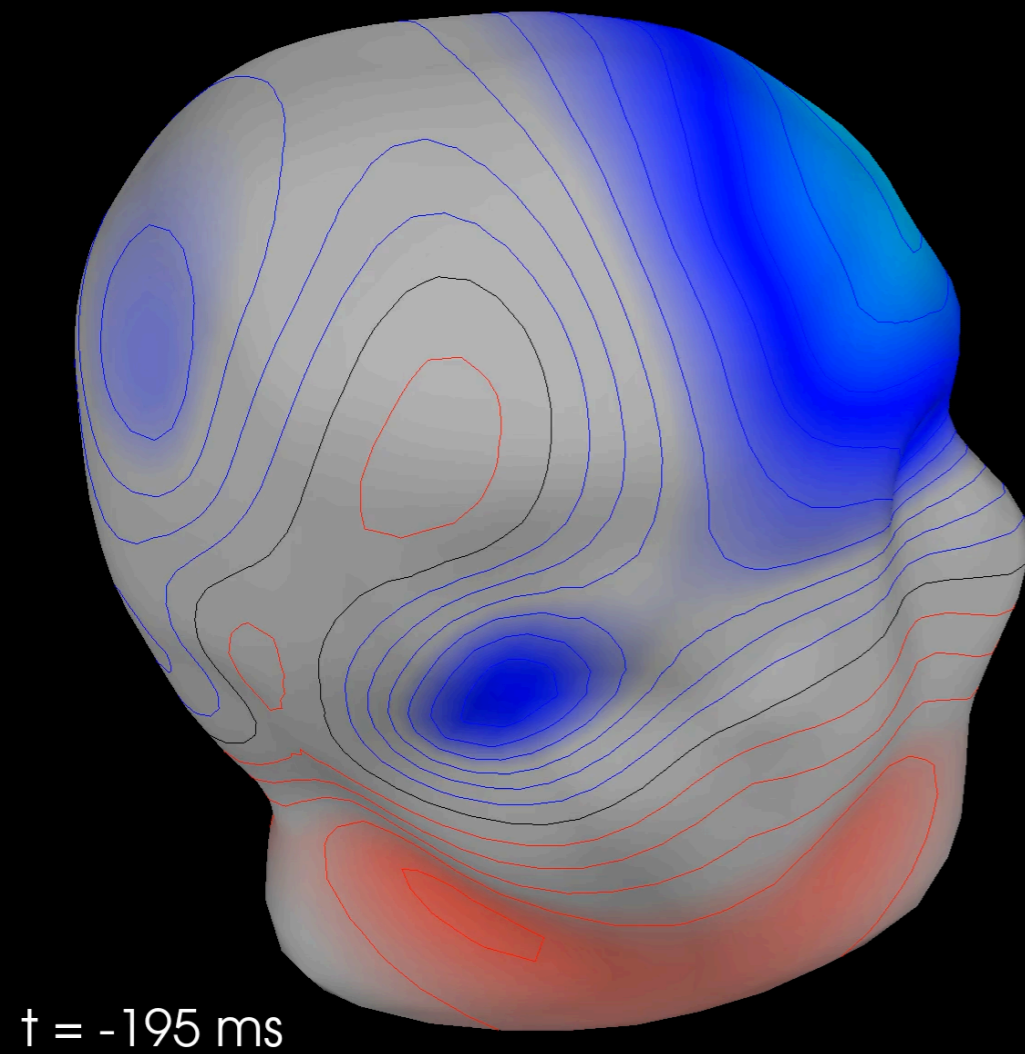
It's often much harder:

- Smaller effect sizes
- Intersubject variability
- Device variability etc.

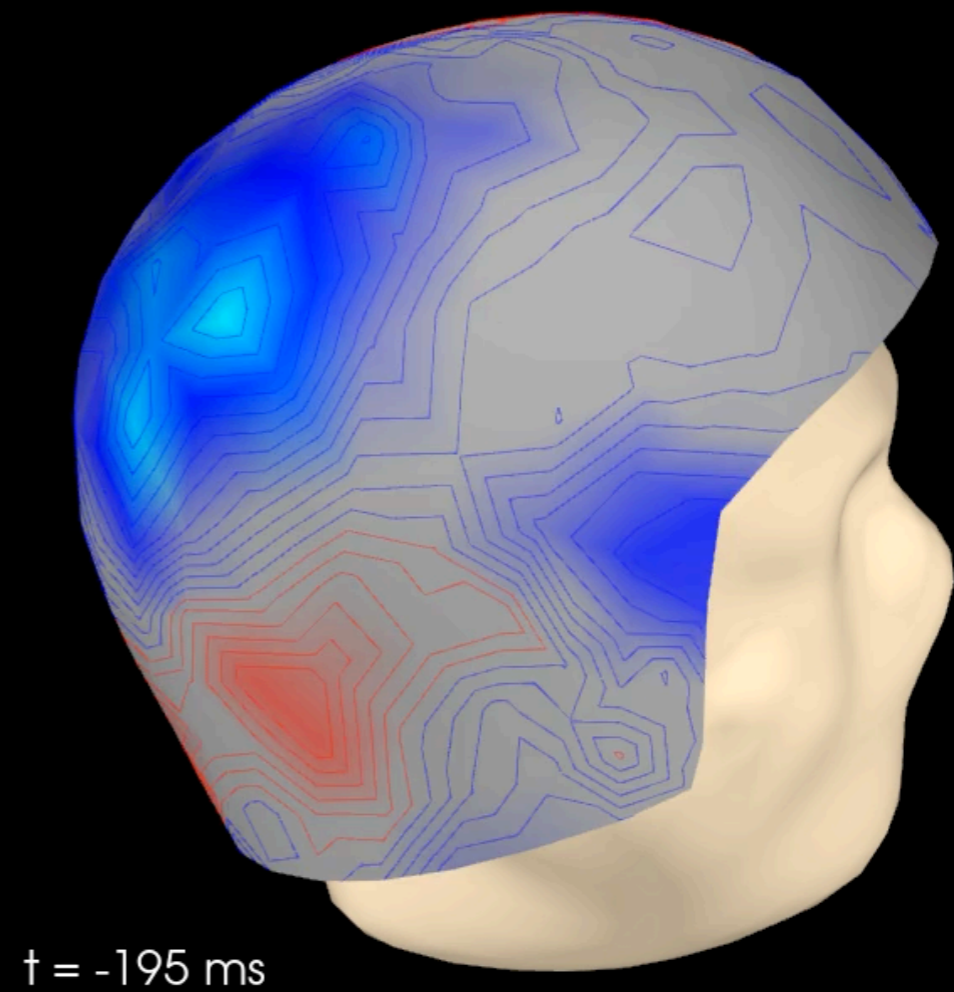


*[Pedregosa, Varoquaux,
Gramfort et al. JMLR 2011]*

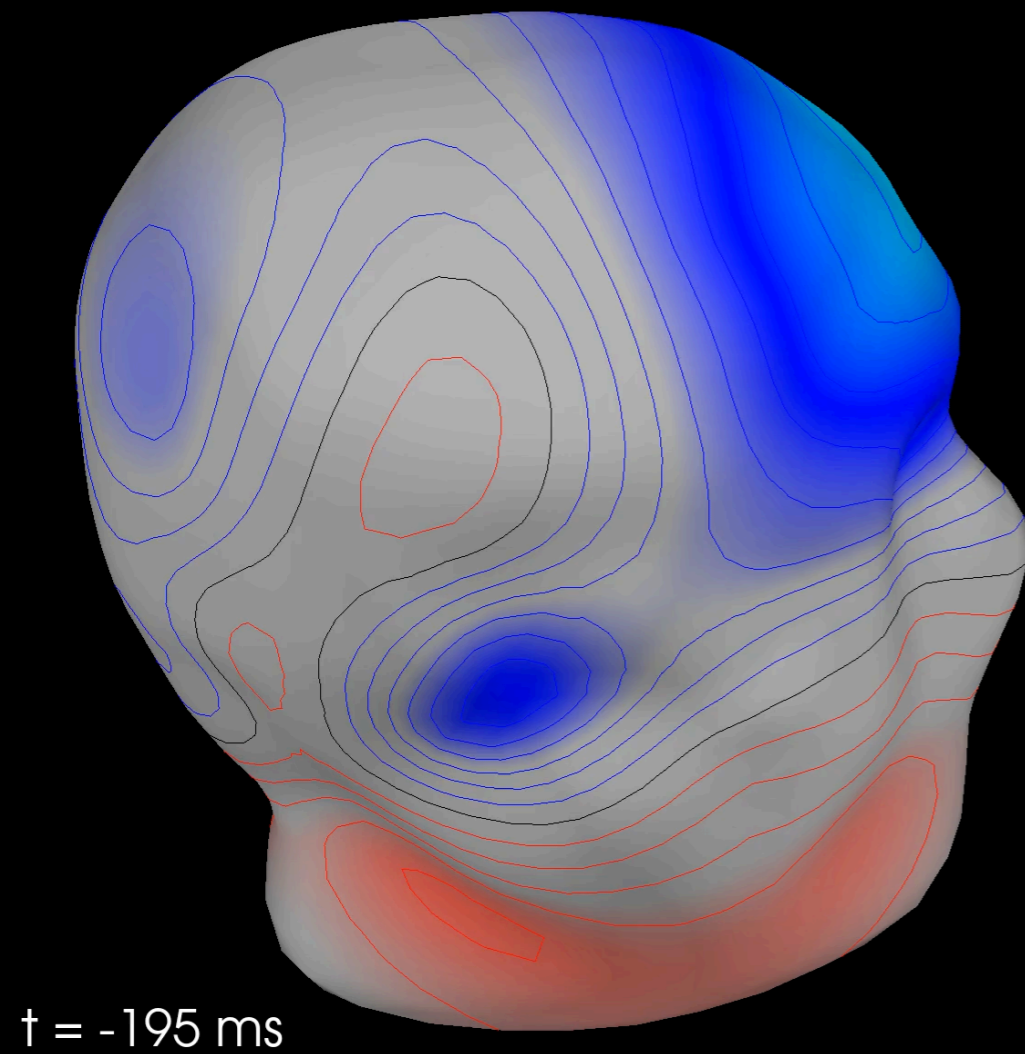
Electroencephalography (EEG) Electric Potential [V]



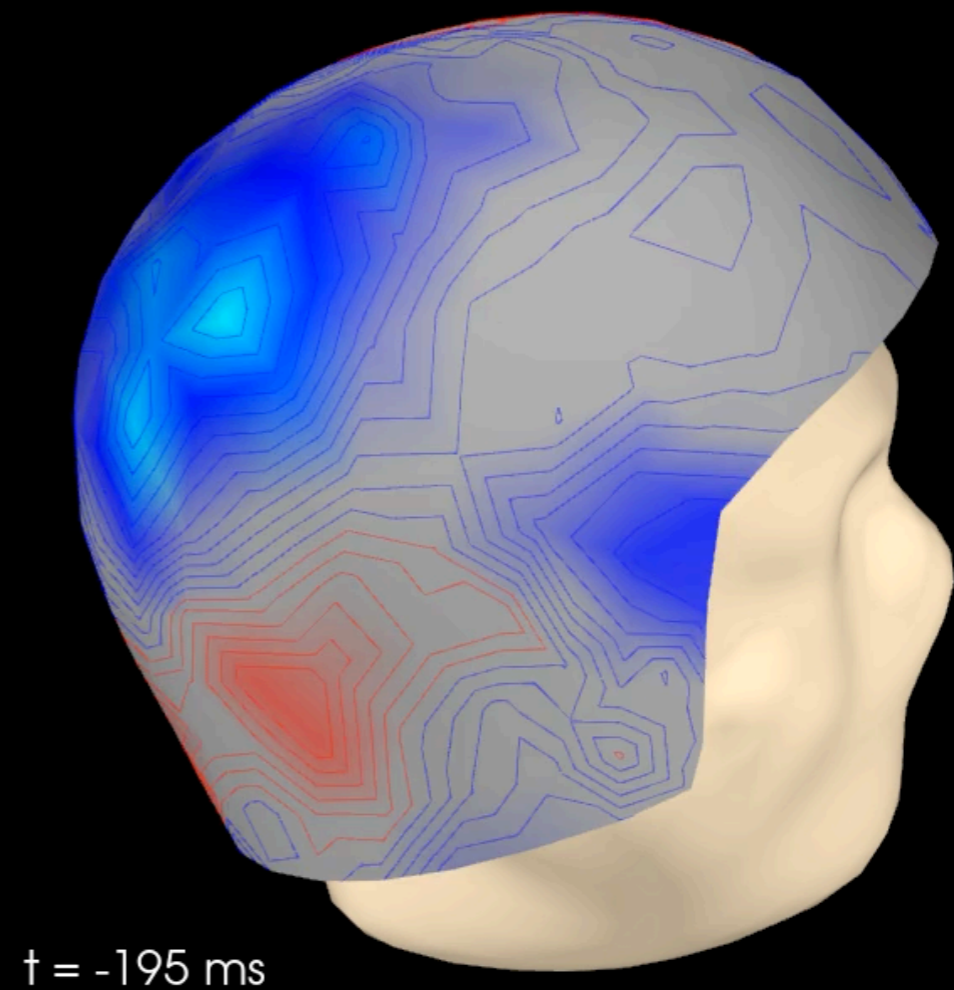
Magnetoencephalography (MEG) Magnetic field [T]



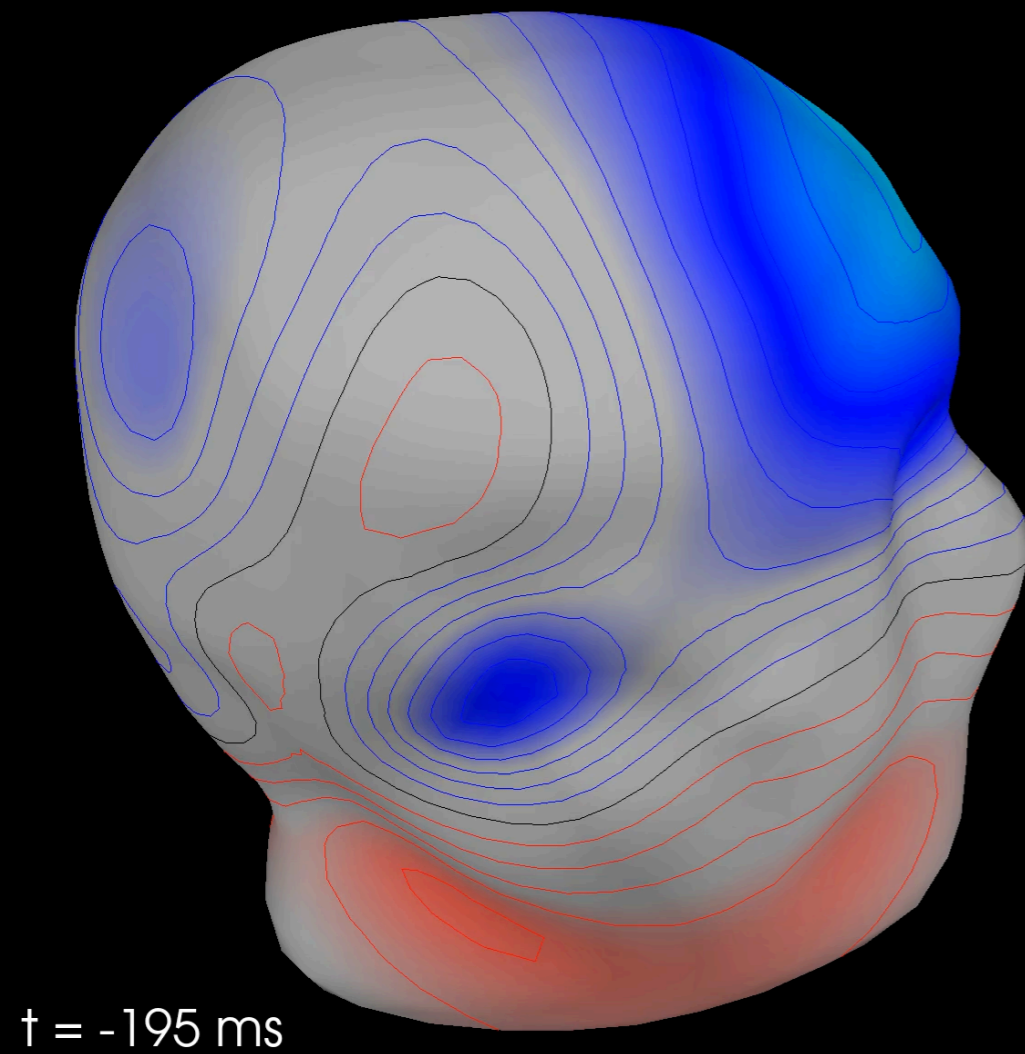
Electroencephalography (EEG) Electric Potential [V]



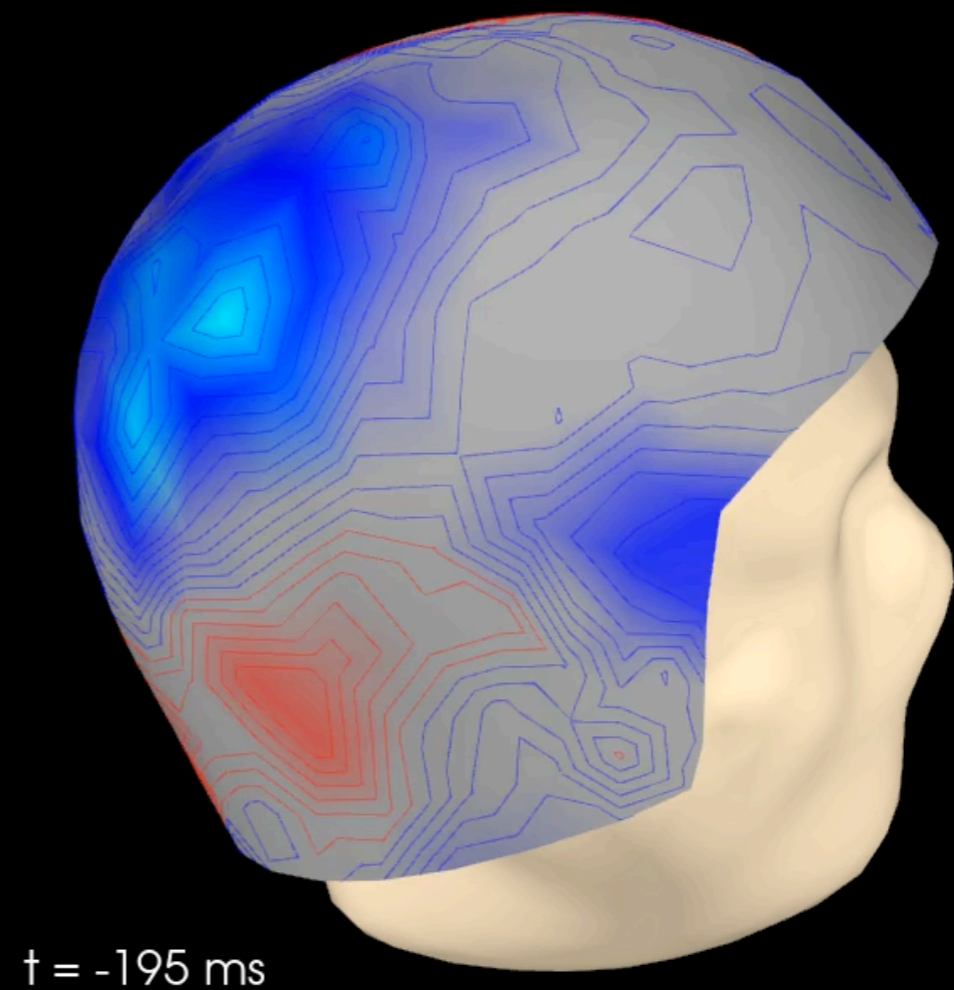
Magnetoencephalography (MEG) Magnetic field [T]



Electroencephalography (EEG) Electric Potential [V]



Magnetoencephalography (MEG) Magnetic field [T]

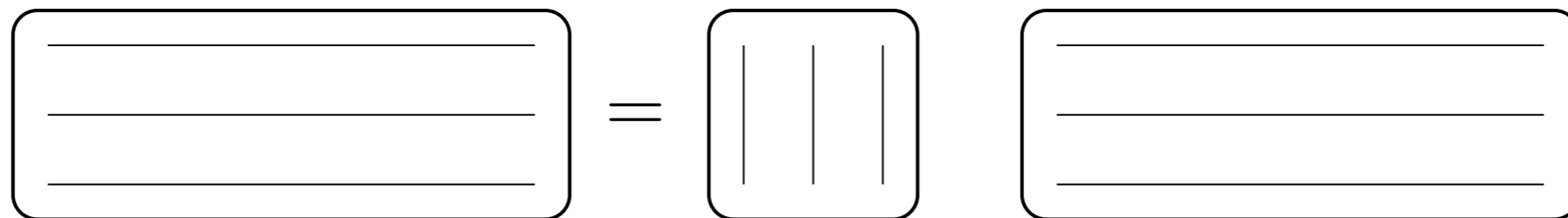


Outline: From unsupervised to self-supervised

Unsupervised learning: From ICA to multiview ICA

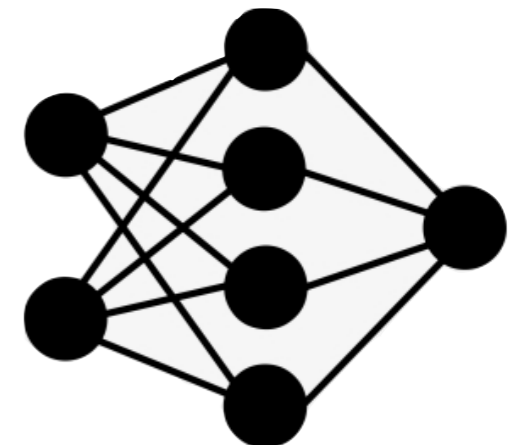
P. Ablin, J-F Cardoso, A. Gramfort (2017), **Faster independent component analysis by preconditioning with Hessian approximations**, IEEE Trans. Sig. Proc.

H. Richard, L. Gresele, A. Hyvärinen, B. Thirion, A. Gramfort, P. Ablin (2020), **Modeling Shared Responses in Neuroimaging Studies through MultiView ICA**, Proc. NeurIPS



Self-supervised learning on EEG data

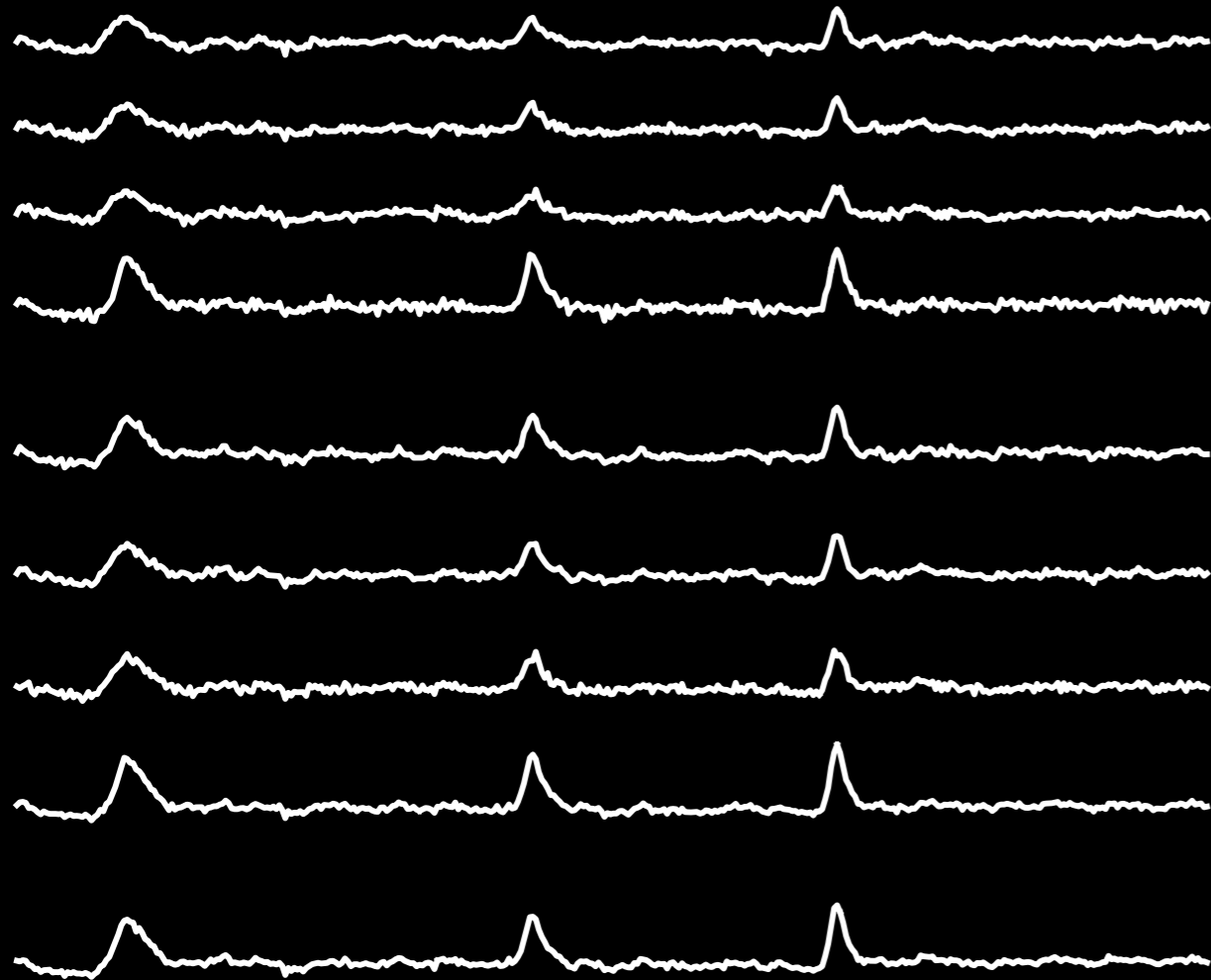
Banville, H., Chehab, O., Hyvärinen, A., Engemann, D. and Gramfort, A. (2020), **Uncovering the structure of clinical EEG signals with self-supervised learning**, Journal of Neural Engineering



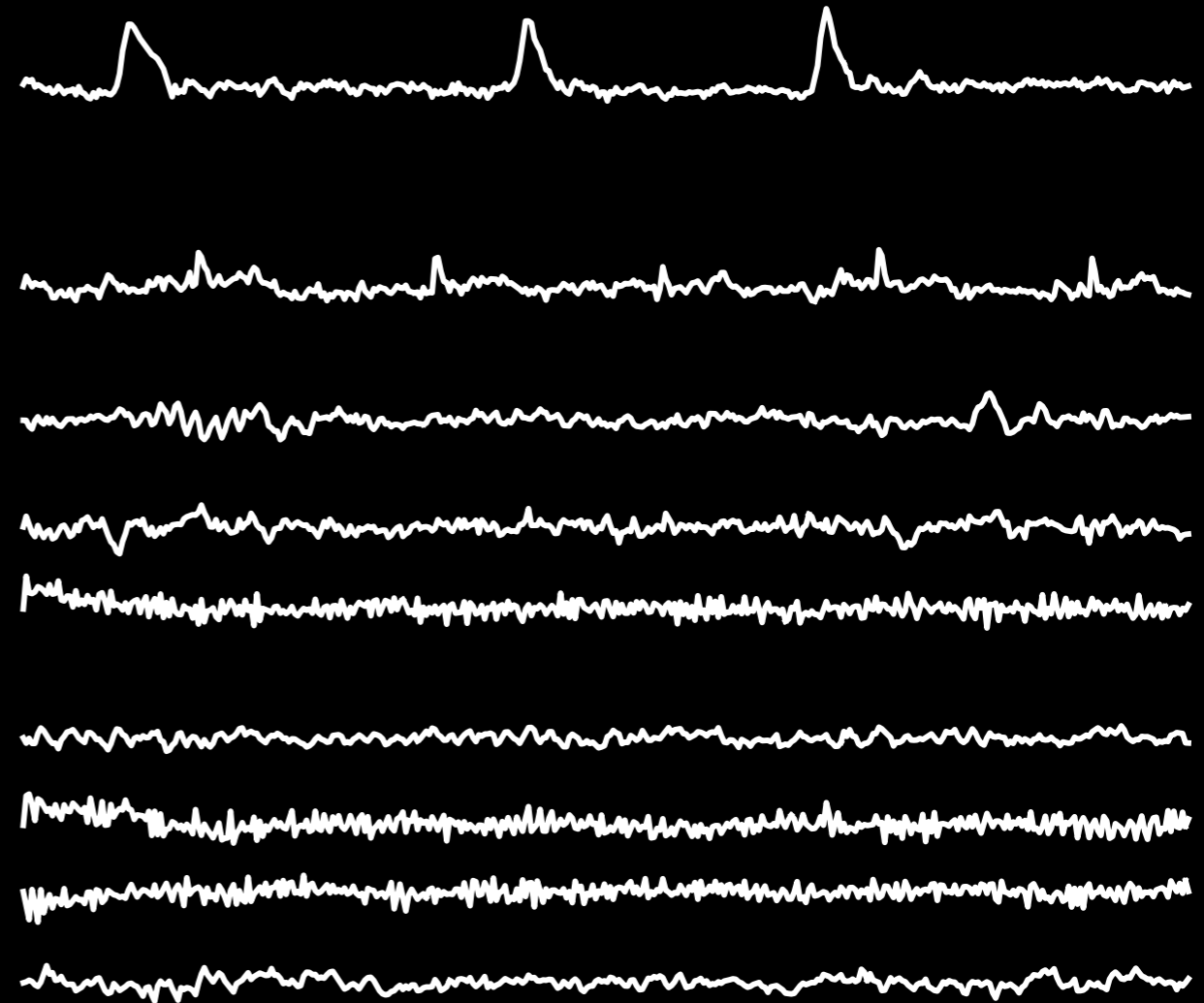
1

From Independent Component Analysis (ICA) to multiview ICA

Observations (raw EEG)



ICA recovered sources



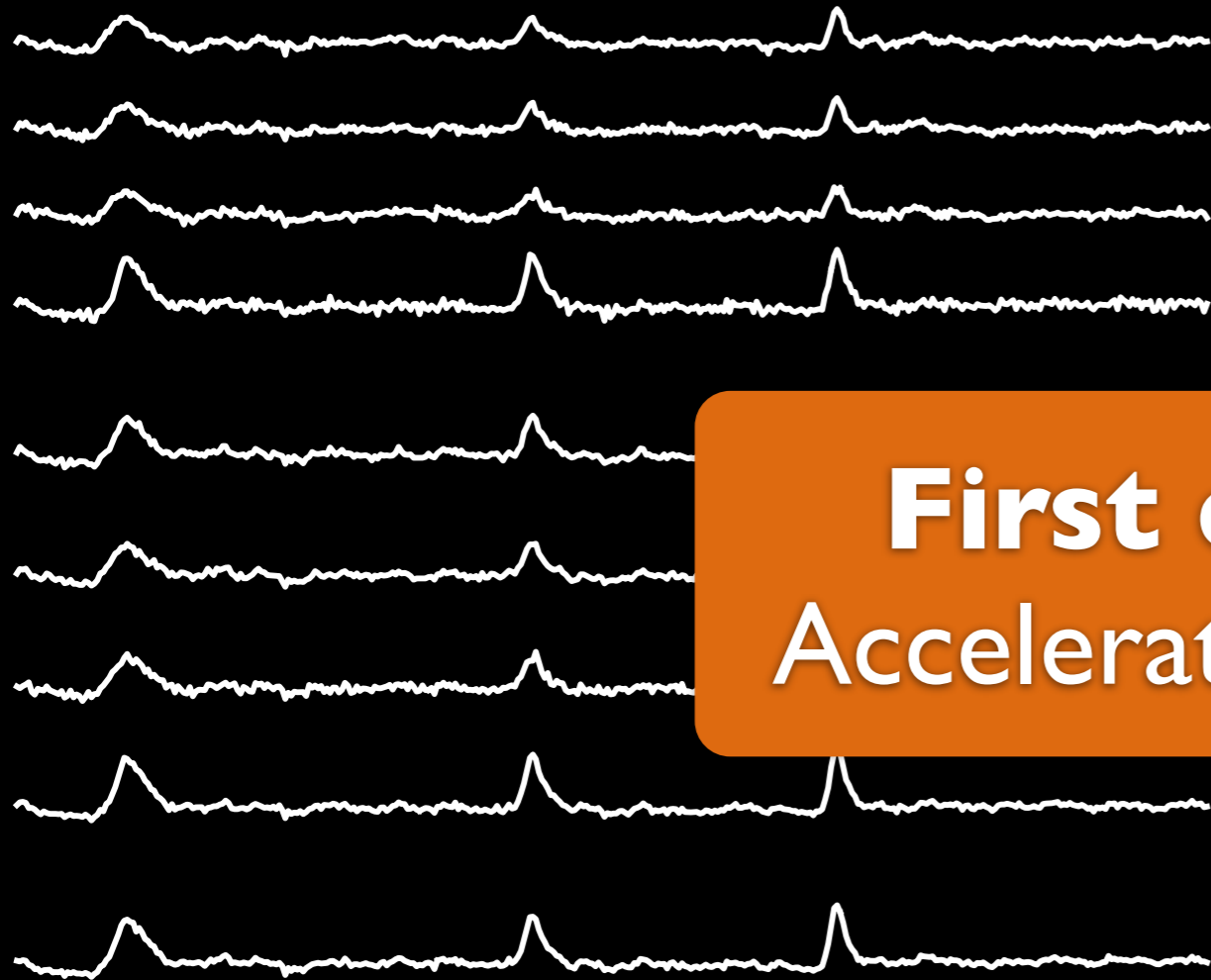
*[Faster independent component analysis by preconditioning with Hessian approximations,
P. Ablin, J.-F. Cardoso & A. Gramfort 2017 IEEE Trans. Signal Processing]*

Code: <https://pierreablin.github.io/picard>

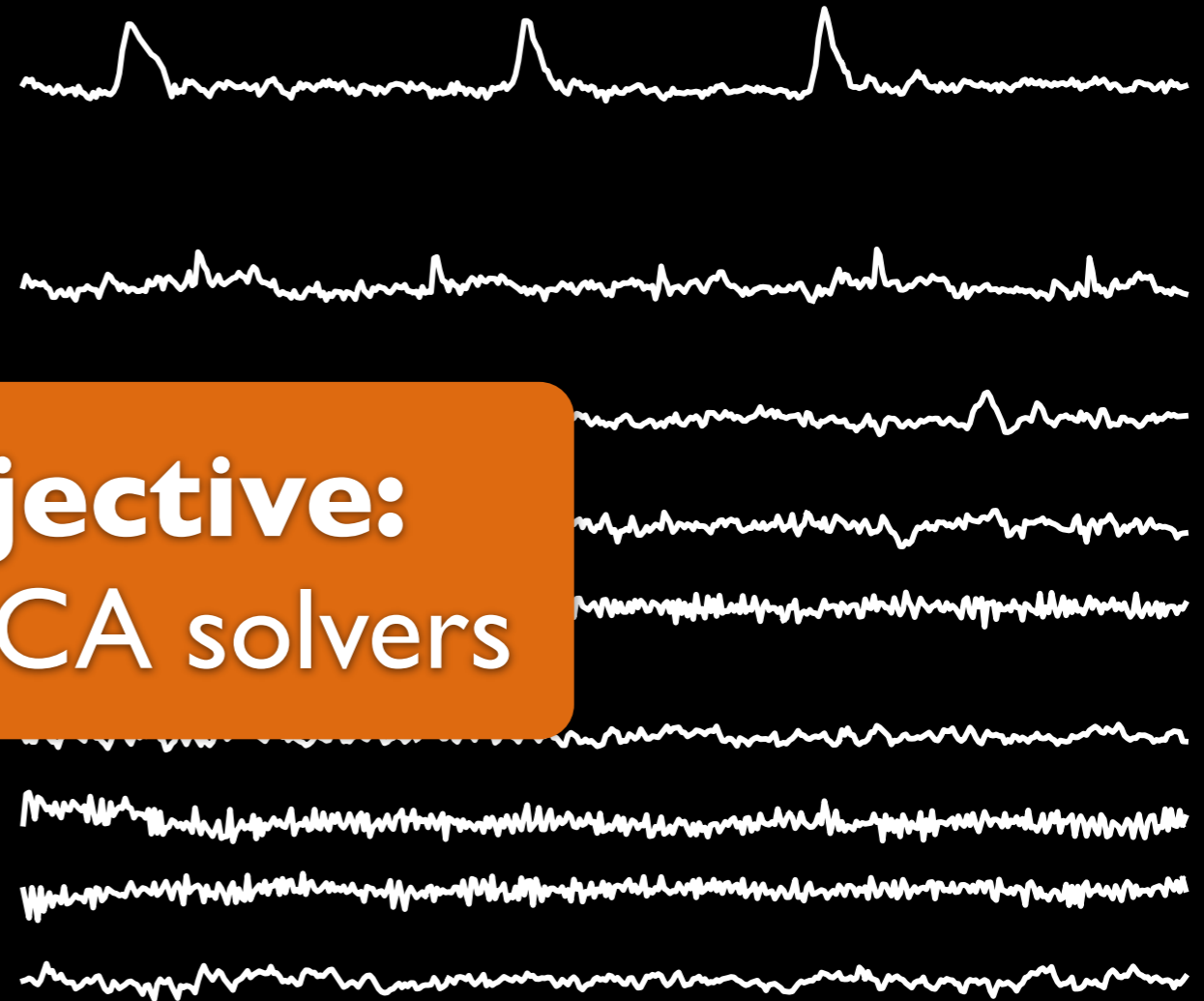
1

From Independent Component Analysis (ICA) to multiview ICA

Observations (raw EEG)



ICA recovered sources



First objective:
Accelerate ICA solvers

*[Faster independent component analysis by preconditioning with Hessian approximations,
P. Ablin, J.-F. Cardoso & A. Gramfort 2017 IEEE Trans. Signal Processing]*

Code: <https://pierreablin.github.io/picard>

Linear ICA model: $\mathbf{X} = \mathbf{A}\mathbf{S}$

- **Assumption:** Observed signals are a linear mix of independent identically distributed signals.

$$\mathbf{X} \in \mathbb{R}^{N \times T} = \mathbf{A} \in \mathbb{R}^{N \times N} \mathbf{S} \in \mathbb{R}^{N \times T}$$

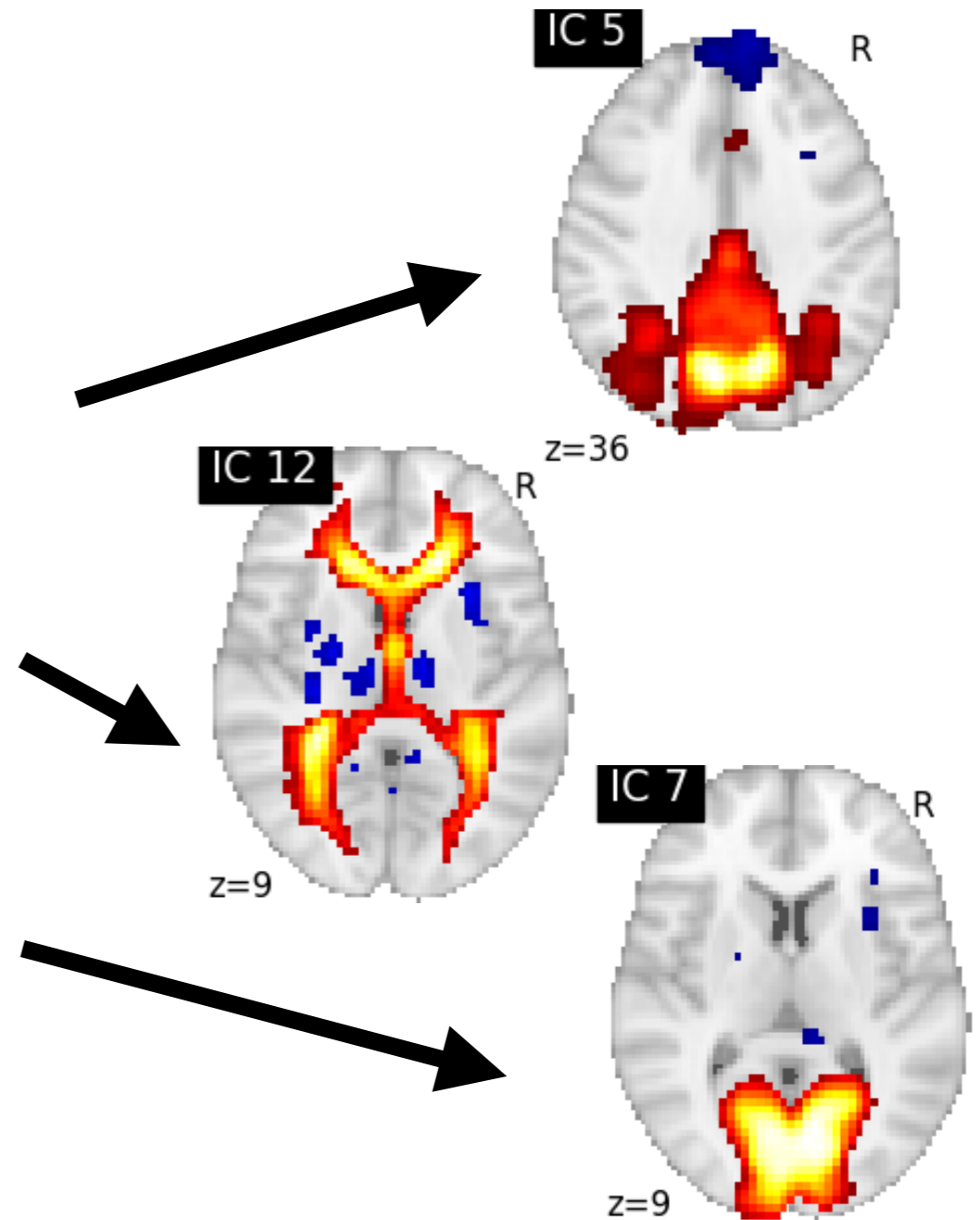
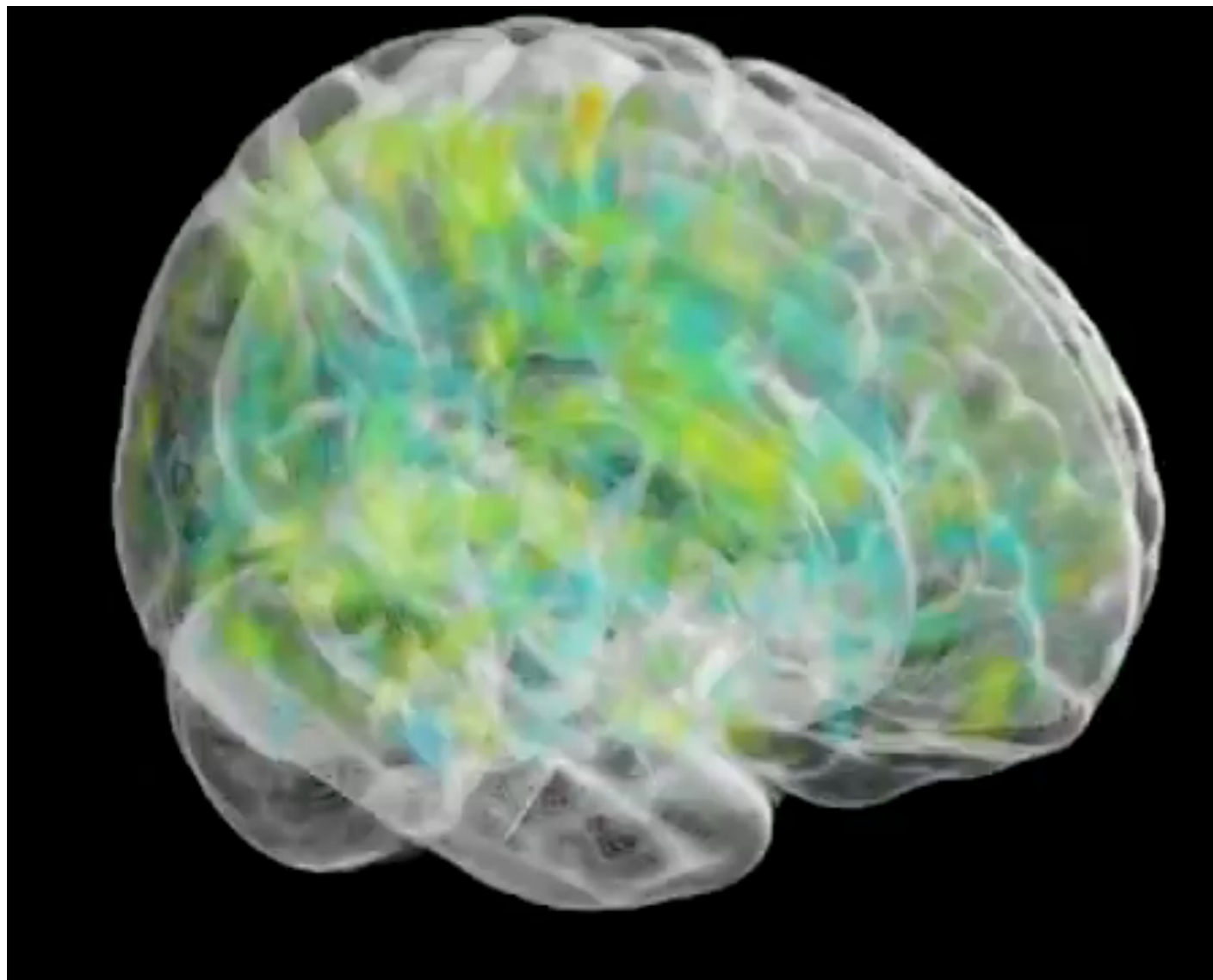
- N : Number of signals
- T : Number of samples
- \mathbf{X} : Observed signals
- \mathbf{S} : Independent **sources** signals
- \mathbf{A} : **Mixing matrix**

Both \mathbf{A} and \mathbf{S} are unknown

[Jutten & Herault 91, Bell & Sejnowski 1995, Hyvärinen 1997]

Why should you care?

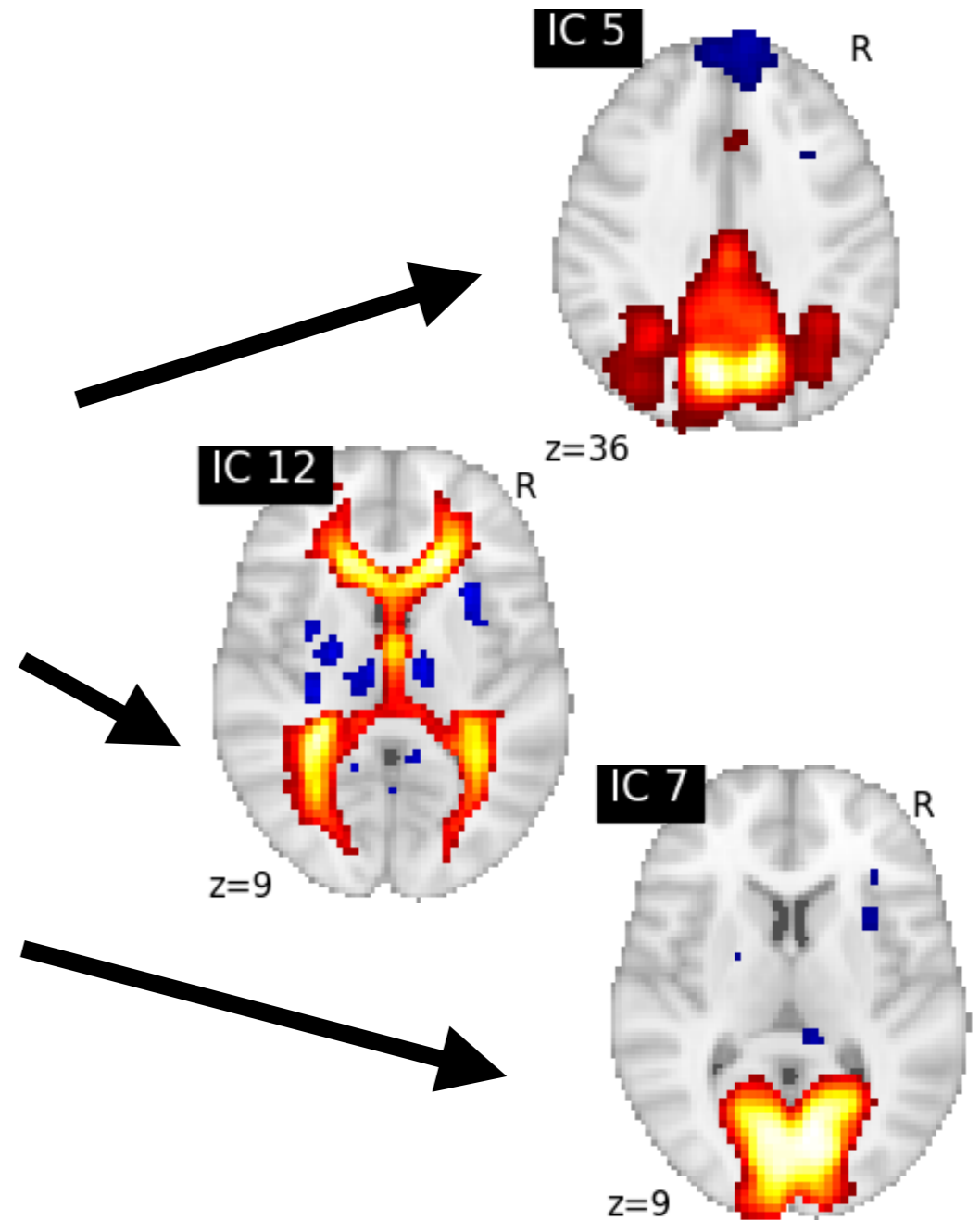
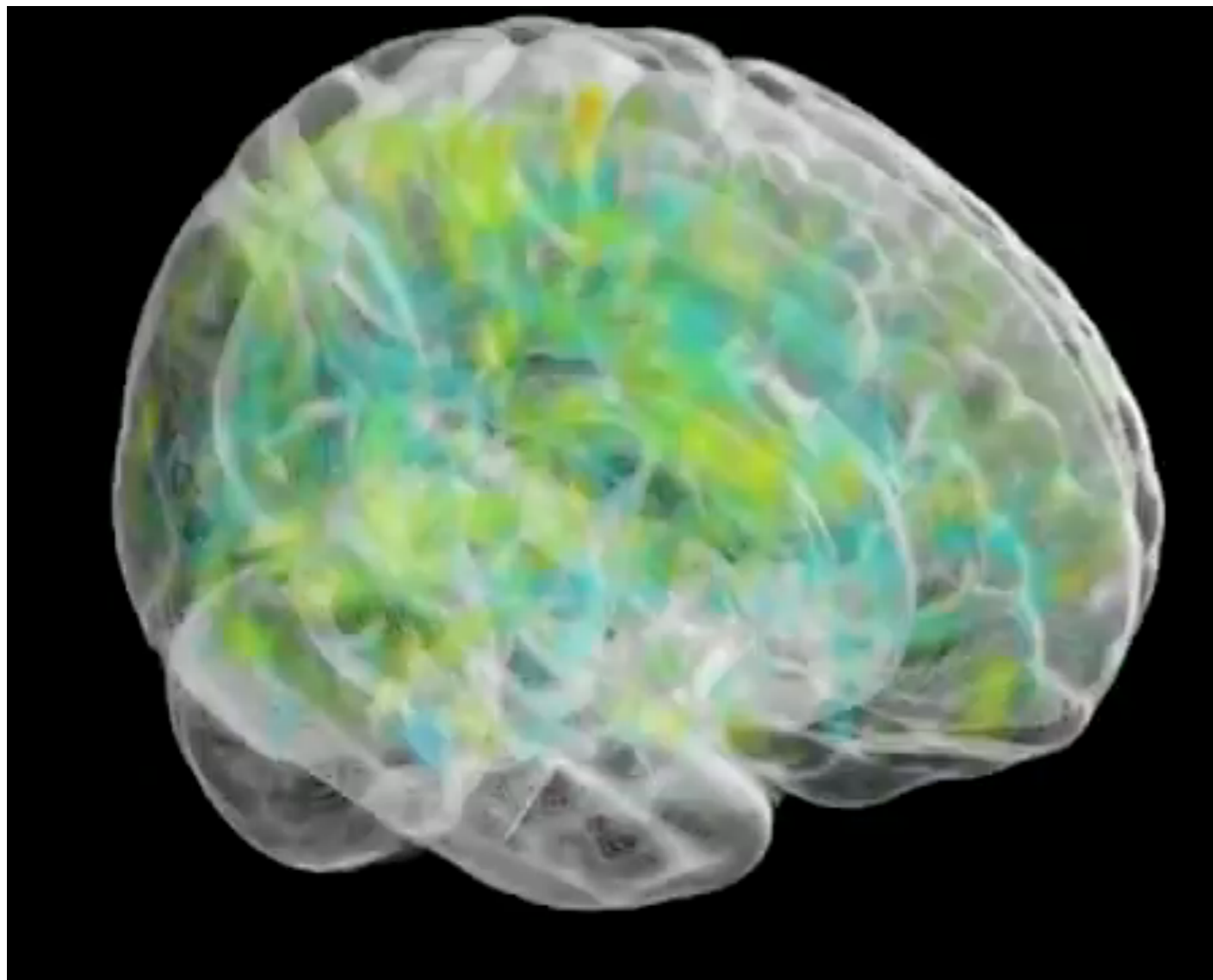
- Used in neuroimaging to find “brain networks” (not neural networks...) in functional MRI data



Source: http://nilearn.github.io/auto_examples/03_connectivity/plot_canica_resting_state.html

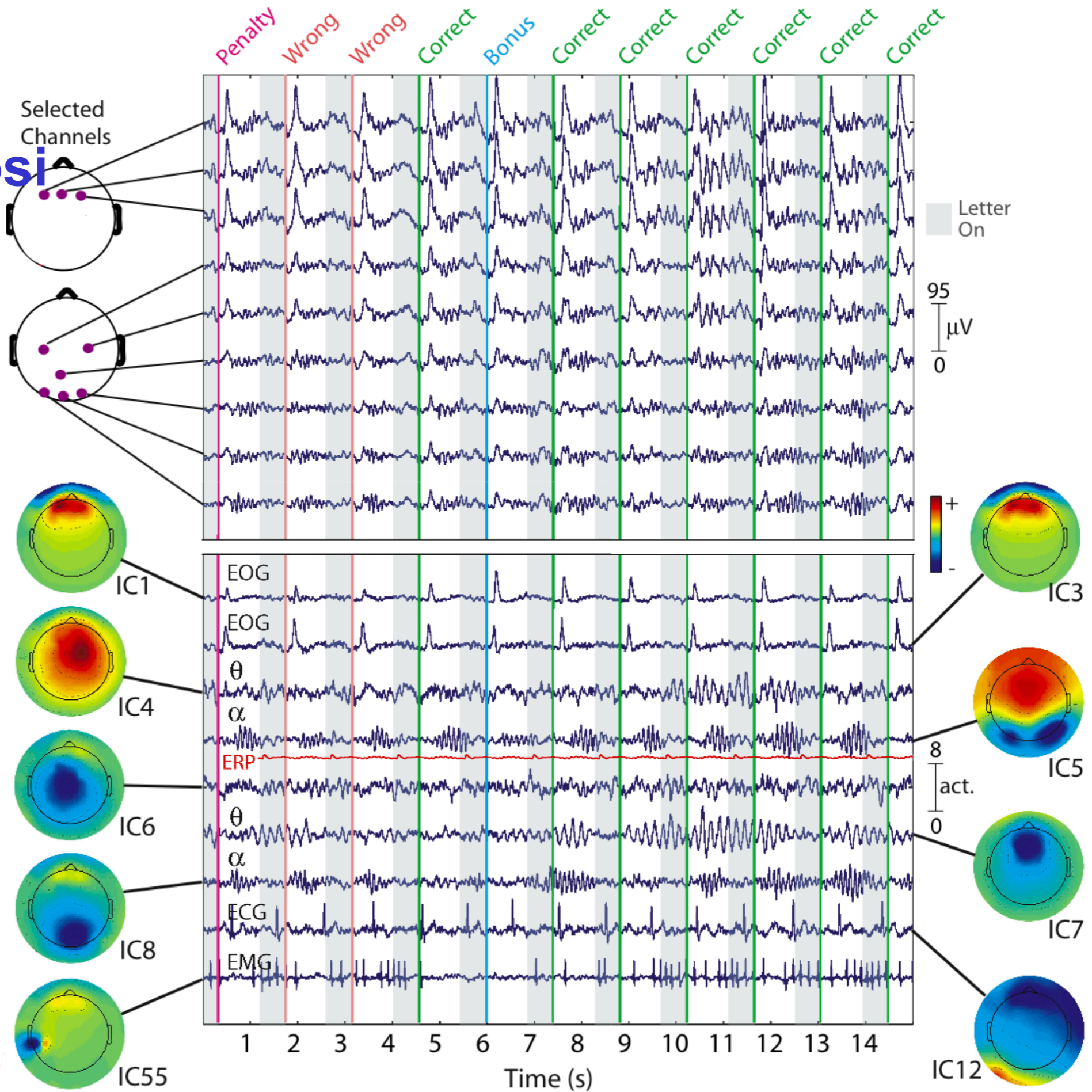
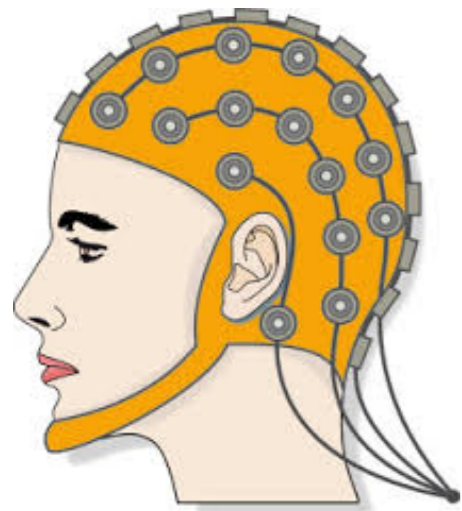
Why should you care?

- Used in neuroimaging to find “brain networks” (not neural networks...) in functional MRI data



Source: http://nilearn.github.io/auto_examples/03_connectivity/plot_canica_resting_state.html

Sample EEG Decomposition



X

\hat{S}

Principles of (non-Gaussian) ICA

$$\mathbf{X} = \mathbf{A}\mathbf{S}$$

Objective: Find \mathbf{W} s.t. $\mathbf{W}\mathbf{X}$ has maximally independent rows.

Model: Let $\mathbf{Y} = \mathbf{W}\mathbf{X}$ denote the unmixed data

$$p(\mathbf{Y}(t)) = p_1(\mathbf{Y}_1(t)) \dots p_N(\mathbf{Y}_N(t))$$

Principles of (non-Gaussian) ICA

$$\mathbf{X} = \mathbf{A}\mathbf{S}$$

Objective: Find \mathbf{W} s.t. $\mathbf{W}\mathbf{X}$ has maximally independent rows.

Model: Let $\mathbf{Y} = \mathbf{W}\mathbf{X}$ denote the unmixed data

$$p(\mathbf{Y}(t)) = p_1(\mathbf{Y}_1(t)) \dots p_N(\mathbf{Y}_N(t))$$

Likelihood: With $\mathbf{Y} = g(\mathbf{X}) = \mathbf{W}\mathbf{X}$

$$p(\mathbf{X}) = p(g(\mathbf{X})) | \det J_{g(\mathbf{X})} | = p(\mathbf{Y}) | \det \mathbf{W} |$$

cf. RealNVP [Dinh et al. 2016]

Principles of (non-Gaussian) ICA

$$\mathbf{X} = \mathbf{A}\mathbf{S}$$

Objective: Find \mathbf{W} s.t. $\mathbf{W}\mathbf{X}$ has maximally independent rows.

Model: Let $\mathbf{Y} = \mathbf{W}\mathbf{X}$ denote the unmixed data

$$p(\mathbf{Y}(t)) = p_1(\mathbf{Y}_1(t)) \dots p_N(\mathbf{Y}_N(t))$$

Likelihood: With $\mathbf{Y} = g(\mathbf{X}) = \mathbf{W}\mathbf{X}$

$$p(\mathbf{X}) = p(g(\mathbf{X})) | \det J_{g(\mathbf{X})} | = p(\mathbf{Y}) | \det \mathbf{W} |$$

leads to:

cf. RealNVP [Dinh et al. 2016]

$$\mathcal{L}(\mathbf{W}) = \log | \det \mathbf{W} | + \mathbb{E}_t \left(\sum_{i=1}^N \log p_i(\mathbf{Y}_i(t)) \right)$$

[Pham & Garat 1997]

Optimization problem

$$\min_{\mathbf{W} \in \mathbb{R}^{N \times N}} -\log |\det \mathbf{W}| - \frac{1}{T} \sum_{t=1}^T \left(\sum_{i=1}^N \log p_i(\mathbf{Y}_i(t)) \right)$$

Optimization problem

$$\min_{\mathbf{W} \in \mathbb{R}^{N \times N}} -\log |\det \mathbf{W}| - \frac{1}{T} \sum_{t=1}^T \left(\sum_{i=1}^N \log p_i(\mathbf{Y}_i(t)) \right)$$

- Infomax model [Bell & Sejnowski 1995]

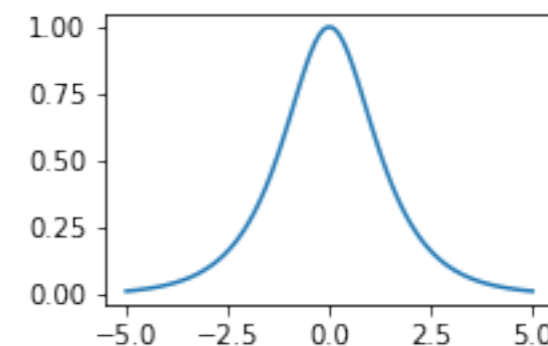
Density

$$p_i(\cdot) \propto \frac{1}{\cosh(\cdot)}$$

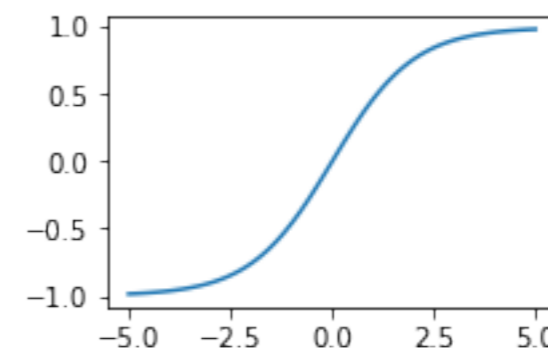
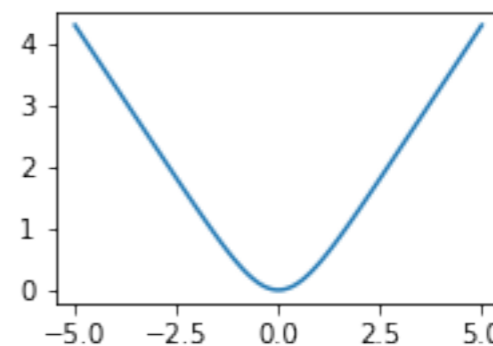
$$-\log(p_i(\cdot)) = \log(\cosh(\cdot)) + c$$

Score function:

$$\psi_i(\cdot) = -\log(p_i(\cdot))' = \tanh(\cdot/2)$$



Heavy-tail
“sparse”



Geometry of the problem

- non-convex (multiple minima)
- Optimization on the invertible matrices manifold
- Use of a **relative** framework

Absolute

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \delta \mathbf{W}$$

Relative

$$\mathbf{W}_{n+1} = (\mathbf{I}_N + \delta \mathbf{W}) \mathbf{W}_n$$

Relative gradient / Hessian

Relative matrix form Taylor expansion:

$$\mathcal{L}((I + \mathcal{E})W) = \mathcal{L}(W) + \langle G | \mathcal{E} \rangle + \frac{1}{2} \langle \mathcal{E} | H | \mathcal{E} \rangle + \mathcal{O}(\|\mathcal{E}\|^3)$$

- Gradient is a $N \times N$ matrix:

$$G_{ij} = E[\psi_i(y_i)y_j] - \delta_{ij}$$

simple
expressions

- Hessian is a $N \times N \times N \times N$ Fourth order tensor.

$$H_{ijkl} = \delta_{il}\delta_{jk} + \delta_{ik}E[\psi'_i(y_i)y_jy_l]$$

(relative) Newton method

$$H_{ijkl} = \delta_{il}\delta_{jk} + \delta_{ik}E[\psi'_i(y_i)y_jy_l]$$

One iteration: $W_{n+1} = (I - H^{-1}G)W_n$

Problem:

Large linear system / regularization needed

Newton method is possible but not practical

(relative) Newton method

$$H_{ijkl} = \delta_{il}\delta_{jk} + \delta_{ik}E[\psi'_i(y_i)y_jy_l]$$

One iteration: $W_{n+1} = (I - H^{-1}G)W_n$

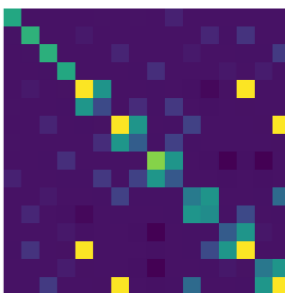
Problem:

Large linear system / regularization needed

Newton method is possible but not practical

but if model holds:

$$\delta_{ik}E[\psi'_i(y_i)y_jy_l] = \delta_{ik}\delta_{jl}E[\psi'_i(y_i)]E[y_j^2]$$



(relative) Newton method

$$H_{ijkl} = \delta_{il}\delta_{jk} + \delta_{ik}E[\psi'_i(y_i)y_jy_l]$$

One iteration: $W_{n+1} = (I - H^{-1}G)W_n$

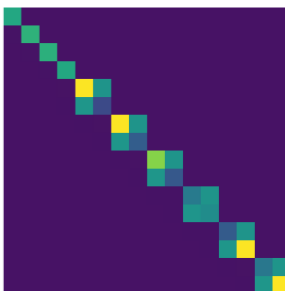
Problem:

Large linear system / regularization needed

Newton method is possible but not practical

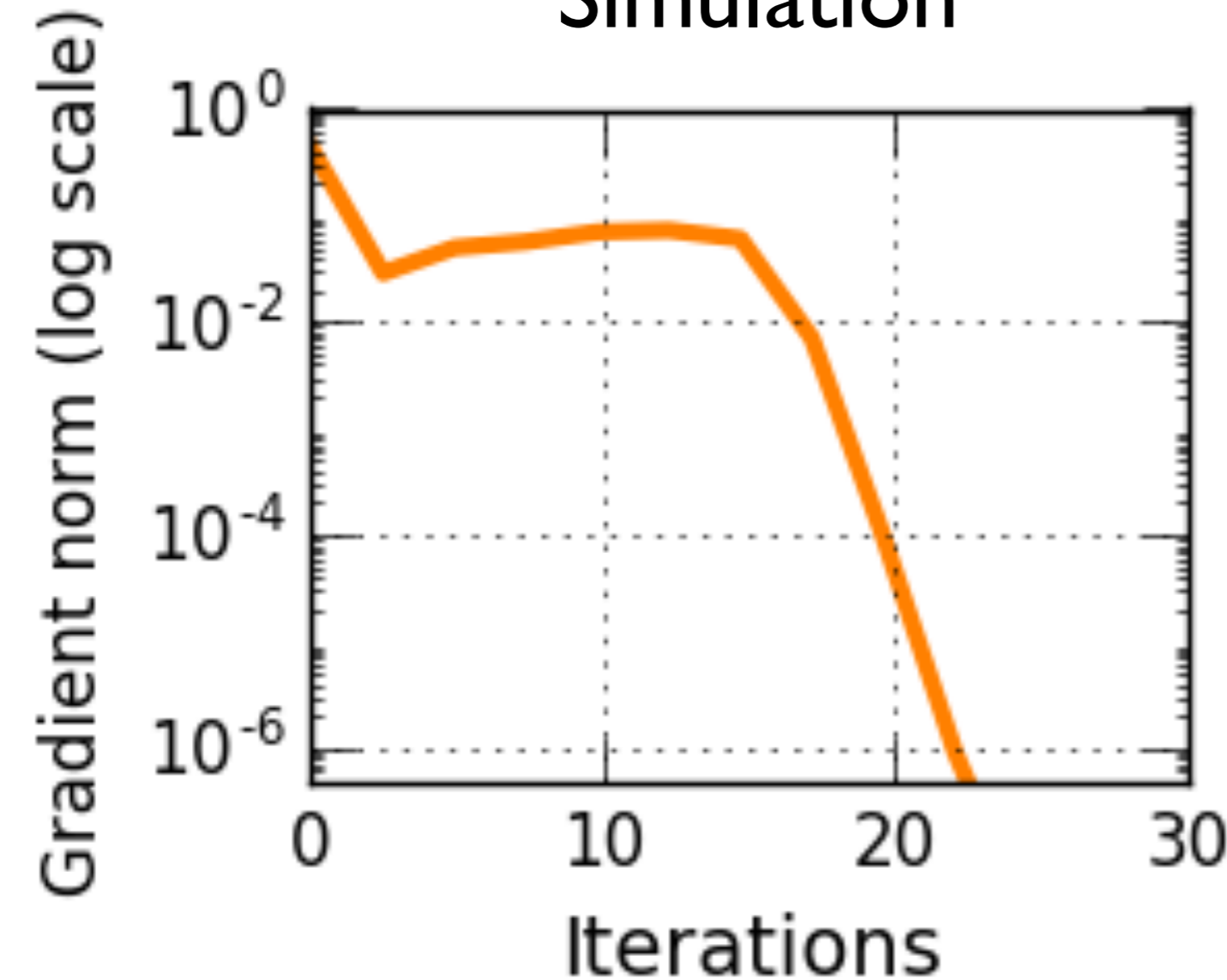
but if model holds:

$$\delta_{ik}E[\psi'_i(y_i)y_jy_l] = \delta_{ik}\delta_{jl}E[\psi'_i(y_i)]E[y_j^2]$$

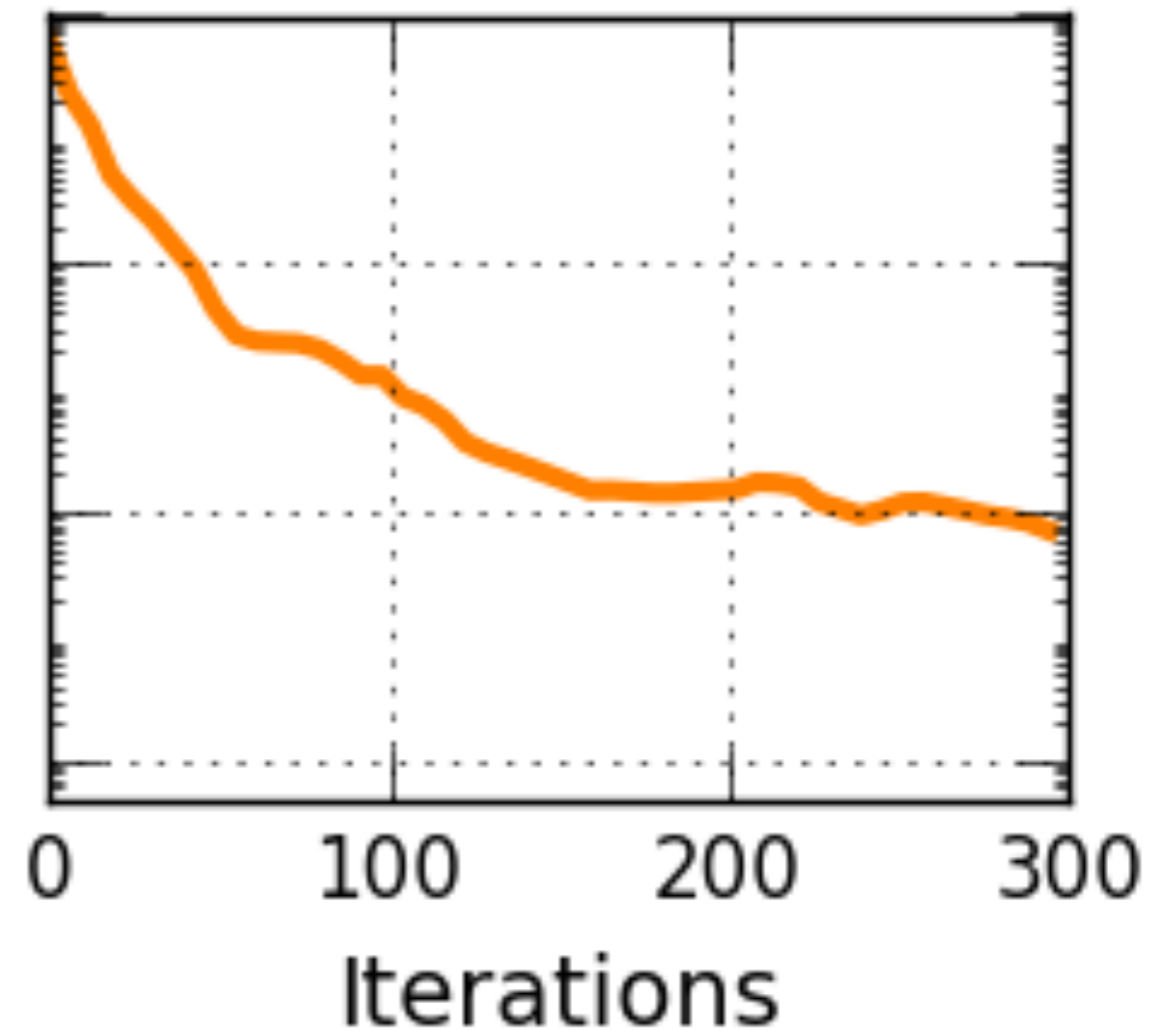


But...

Simulation



EEG



It does not work with real data !

L-BFGS algorithm with Hessian approx.

Idea:

- Combine L-BFGS with Hessian approx.
- Replace diagonal initial guess by Hessian approx.
- The rest is the same (although written with relative gradients)...



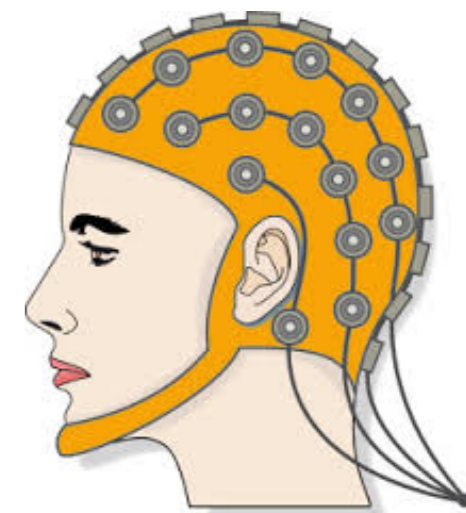
[Liu, D. C., & Nocedal, J. « On the limited memory BFGS method for large scale optimization. » *Mathematical programming*, 1989]

[P.Ablin, J.-F. Cardoso, A. Gramfort,
Faster ICA by preconditioning with Hessian approximations, IEEE TSP 2017]

Real data

- Oracle gradient descent
- Truncated Newton method
- Simple quasi-Newton H2
- Simple quasi-Newton H1
- Trust region ICA
- Infomax
- L-BFGS
- Picard H1
- Picard H2

EEG



Functional MRI

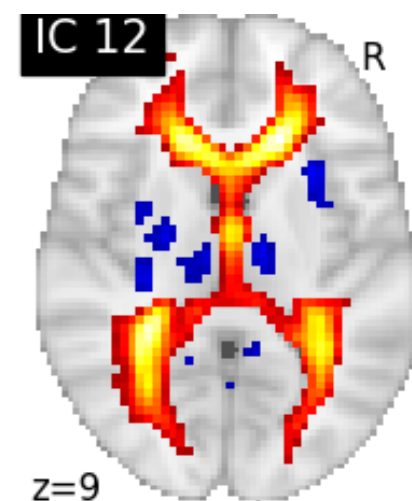
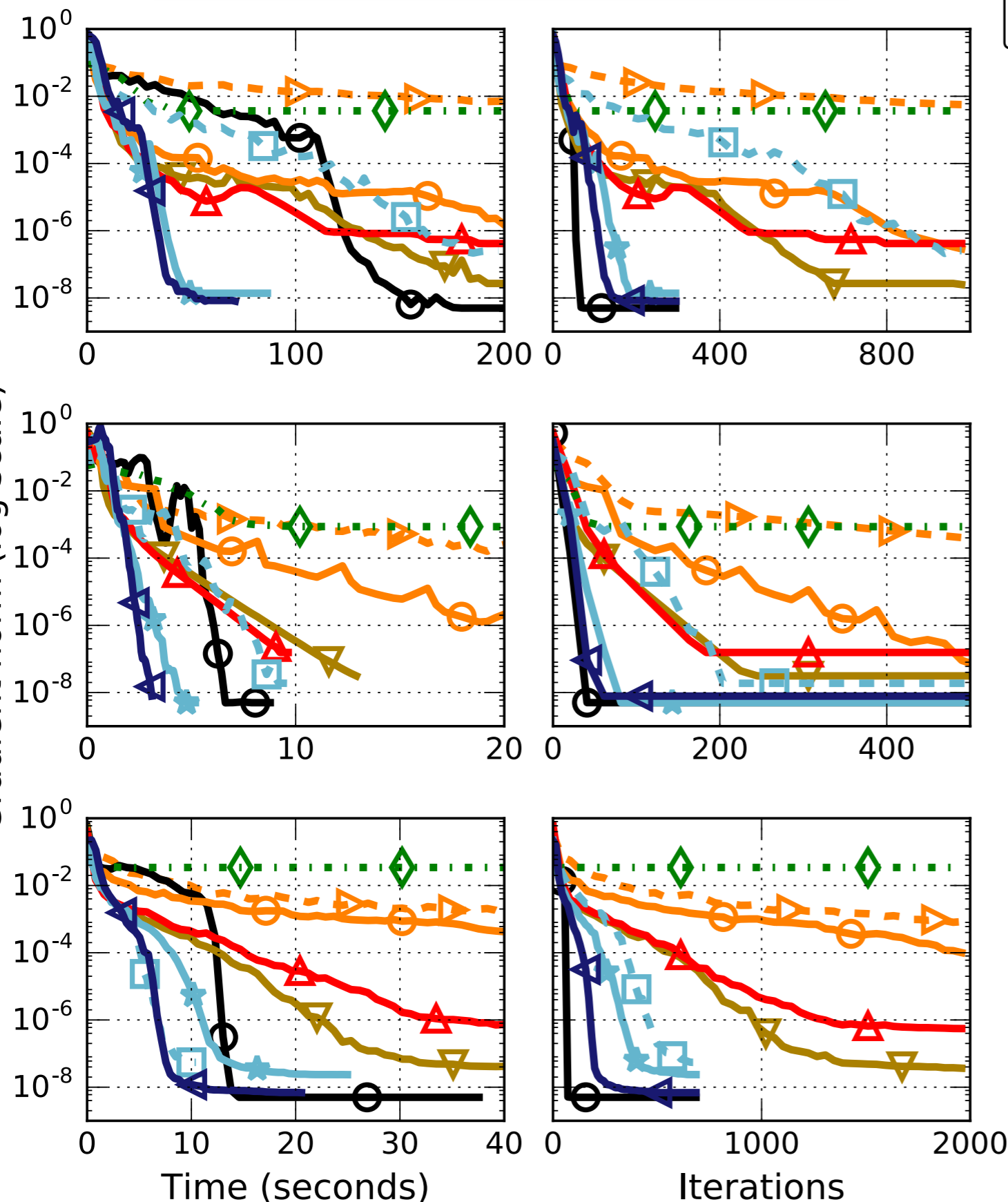
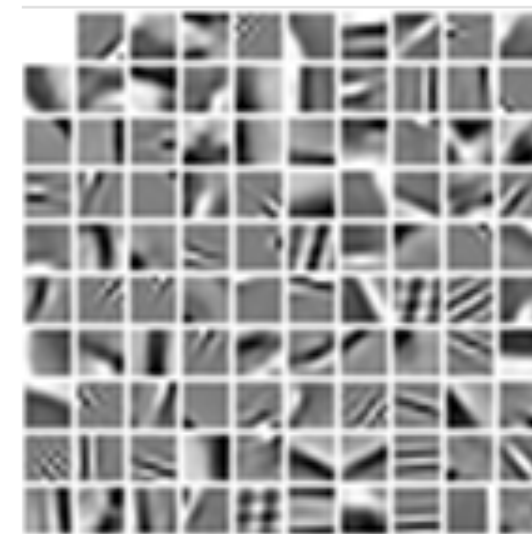


Image patches

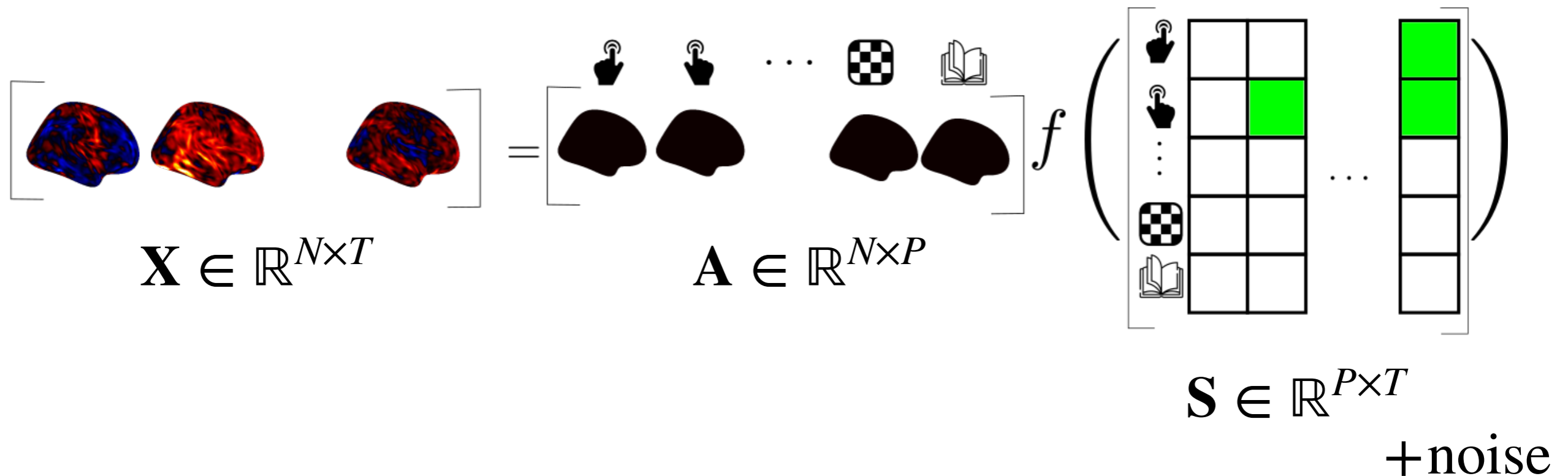


What if you have multiple sets \mathbf{X} , e.g.
from a population?
Multiview ICA

*[Modeling Shared Responses in Neuroimaging Studies through MultiView ICA
Richard, H., Gresele, L., Hyvärinen, A., Thirion, B., Gramfort, A., Ablin, P. (2020). Proc. NeurIPS]
[Shared Independent Component Analysis for Multi-Subject Neuroimaging
Richard, H., Ablin, P., Thirion, B., Gramfort, A., Hyvärinen, A. (2021). Proc. NeurIPS]*

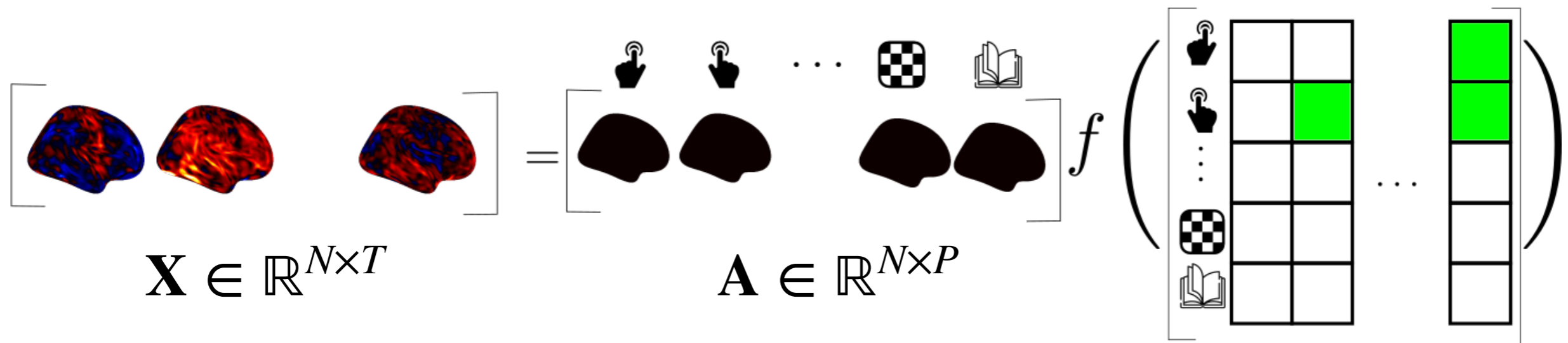
Towards naturalistic stimuli

Controlled stimuli

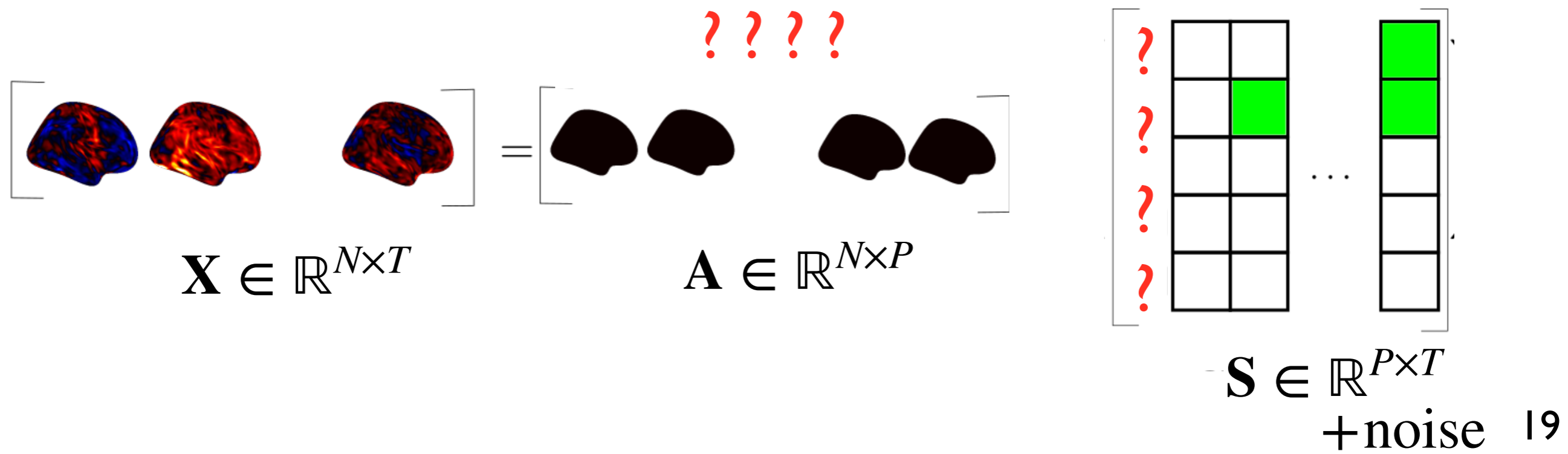


Towards naturalistic stimuli

Controlled stimuli

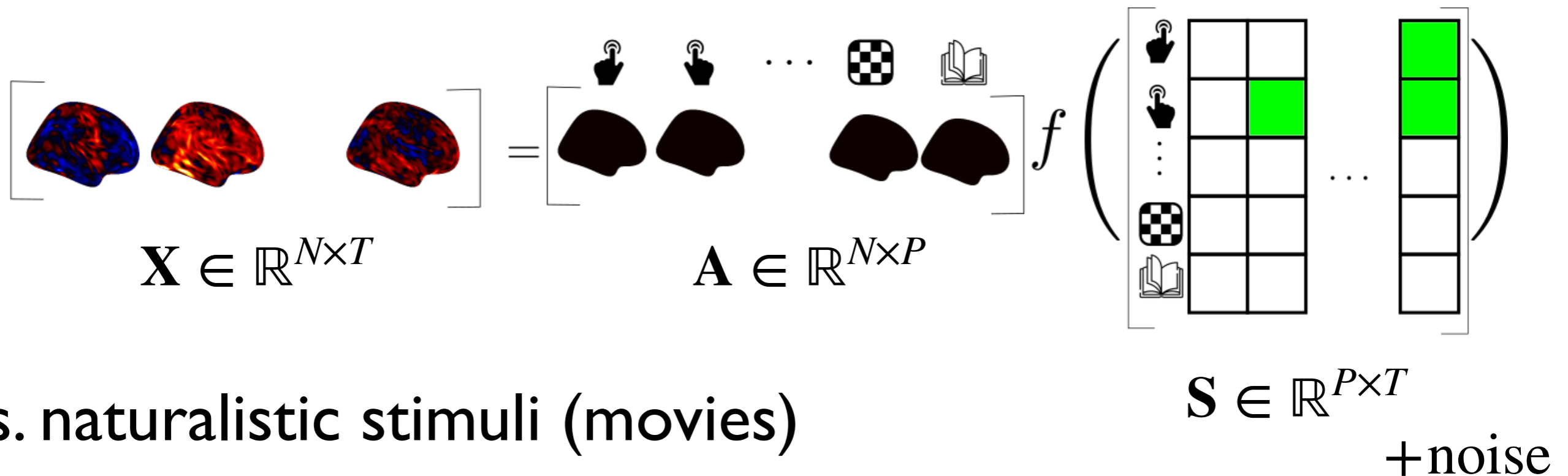


vs. naturalistic stimuli (movies)

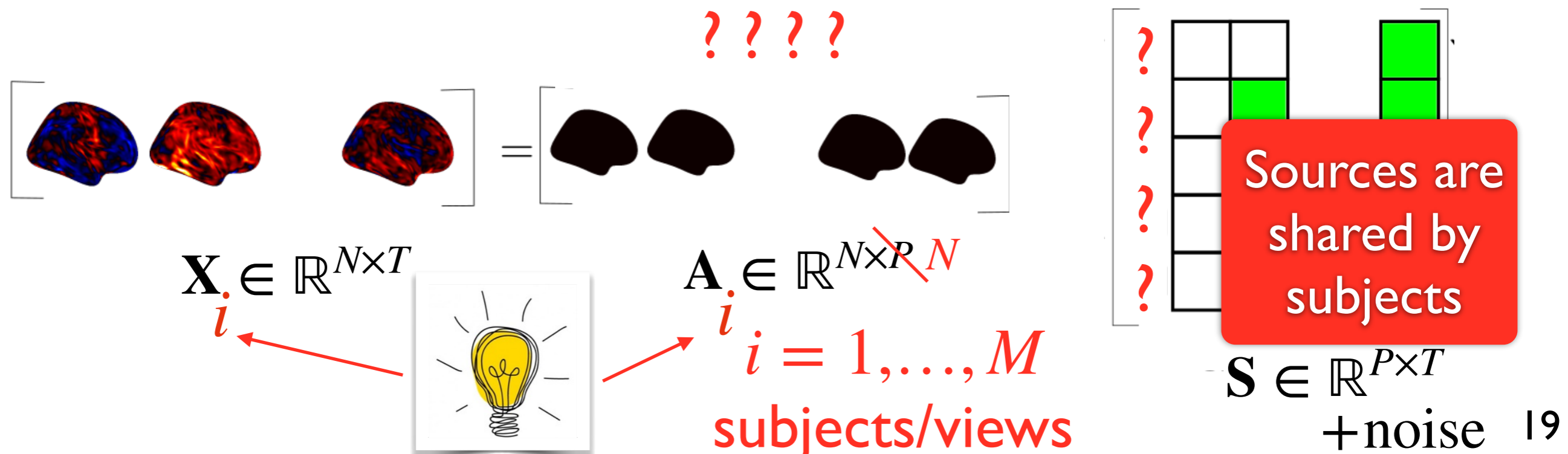


Towards naturalistic stimuli

Controlled stimuli



vs. naturalistic stimuli (movies)



Multiview ICA

Model: $\mathbf{X}^i = \mathbf{A}^i(\mathbf{S} + \mathbf{N}^i)$, $i = 1, \dots, M$ \mathbf{X}^i Data for subject i

Assumptions:

density as in Infomax ($1/\cosh$)

Source independence: $p(\mathbf{S}(t)) = q(\mathbf{S}_1(t)) \dots q(\mathbf{S}_N(t))$

Gaussian noise: $\mathbf{N}_j^i(t) \sim \mathcal{N}(0, \sigma^2)$ independent across components
and independent from sources

Multiview ICA

Model: $\mathbf{X}^i = \mathbf{A}^i(\mathbf{S} + \mathbf{N}^i)$, $i = 1, \dots, M$ \mathbf{X}^i *Data for subject i*

Assumptions:

density as in Infomax ($1/\cosh$)

Source independence: $p(\mathbf{S}(t)) = q(\mathbf{S}_1(t)) \dots q(\mathbf{S}_N(t))$

Gaussian noise: $\mathbf{N}_j^i(t) \sim \mathcal{N}(0, \sigma^2)$ *independent across components and independent from sources*

$$\mathcal{L}(\mathbf{W}^i) = - \underbrace{\sum_{i=1}^M \log |\det \mathbf{W}^i| + f(\tilde{\mathbf{S}})}_{\text{"ICA term"}} + \frac{1}{2\sigma^2} \underbrace{\sum_{i=1}^M \|\mathbf{W}^i \mathbf{X}^i - \tilde{\mathbf{S}}\|^2}_{\text{"Error term"}}$$

$$\mathbf{W}^i \quad \text{Unmixing matrix for subject } i \quad \tilde{\mathbf{S}} = \frac{1}{M} \sum_{i=1}^M \mathbf{W}^i \mathbf{X}^i \quad f(\mathbf{S}) = \sum_{j=1}^N f(\mathbf{S}_j)$$

$$f(\mathbf{S}_j) = \int \exp \left(-\frac{1}{2\sigma^2} M z^2 \right) q(\mathbf{S}_j - z) dz$$

Smoothed density

Optimization (same recipe)

Objective: Minimize the negative log-likelihood \mathcal{L}
using a block-coordinate Newton descent

$$\mathbf{W}^i = (\mathbf{I} + \rho \mathbf{D}^i) \mathbf{W}^i$$

Multiplicative updates

$$\mathbf{D}^i = -(\tilde{\mathbf{H}}^i)^{-1} \mathbf{G}^i$$

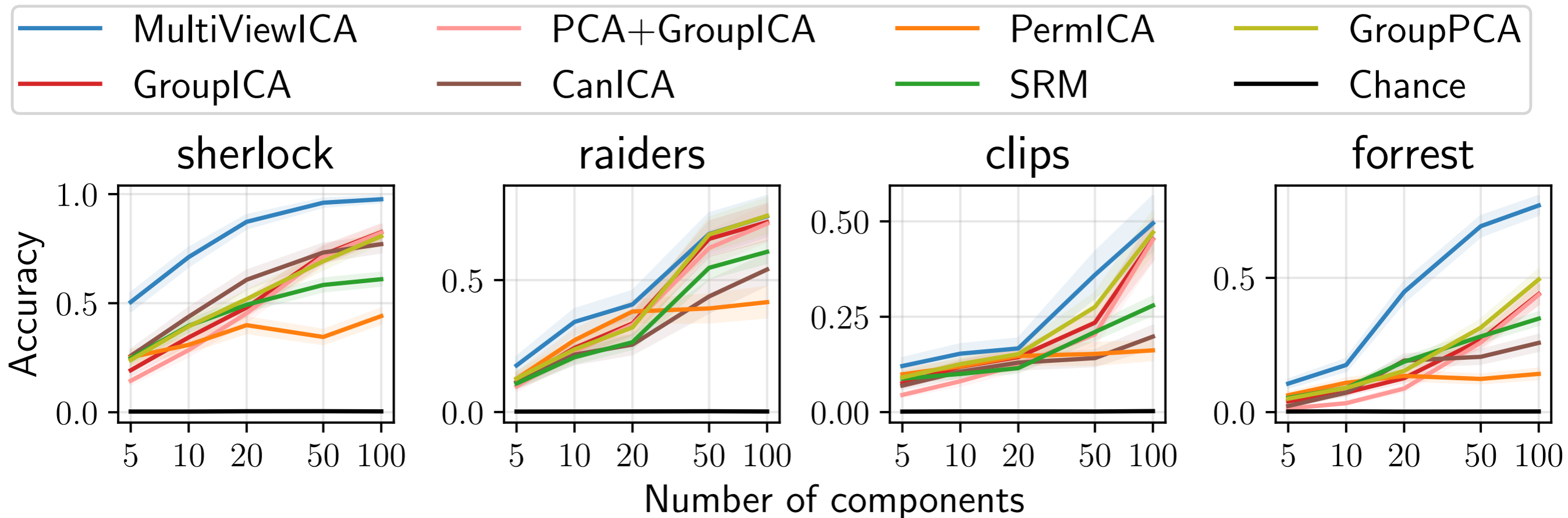
Relative “Newton descent” direction

$$\tilde{\mathbf{H}}^i$$

*Sparse Hessian ($N \times N \times N \times N$)
Approximation with N^2 non-zeros*

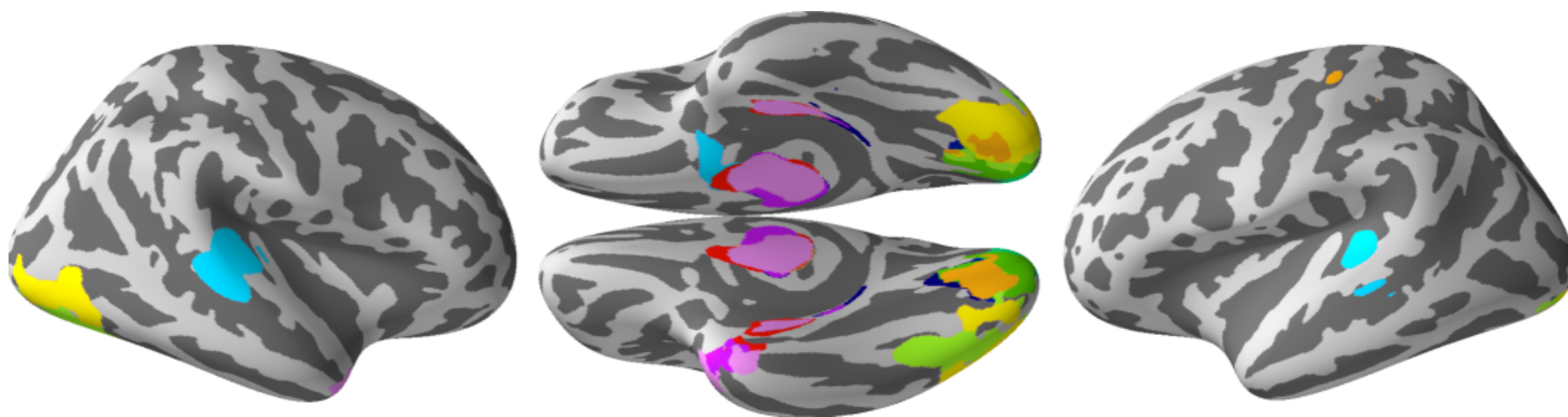
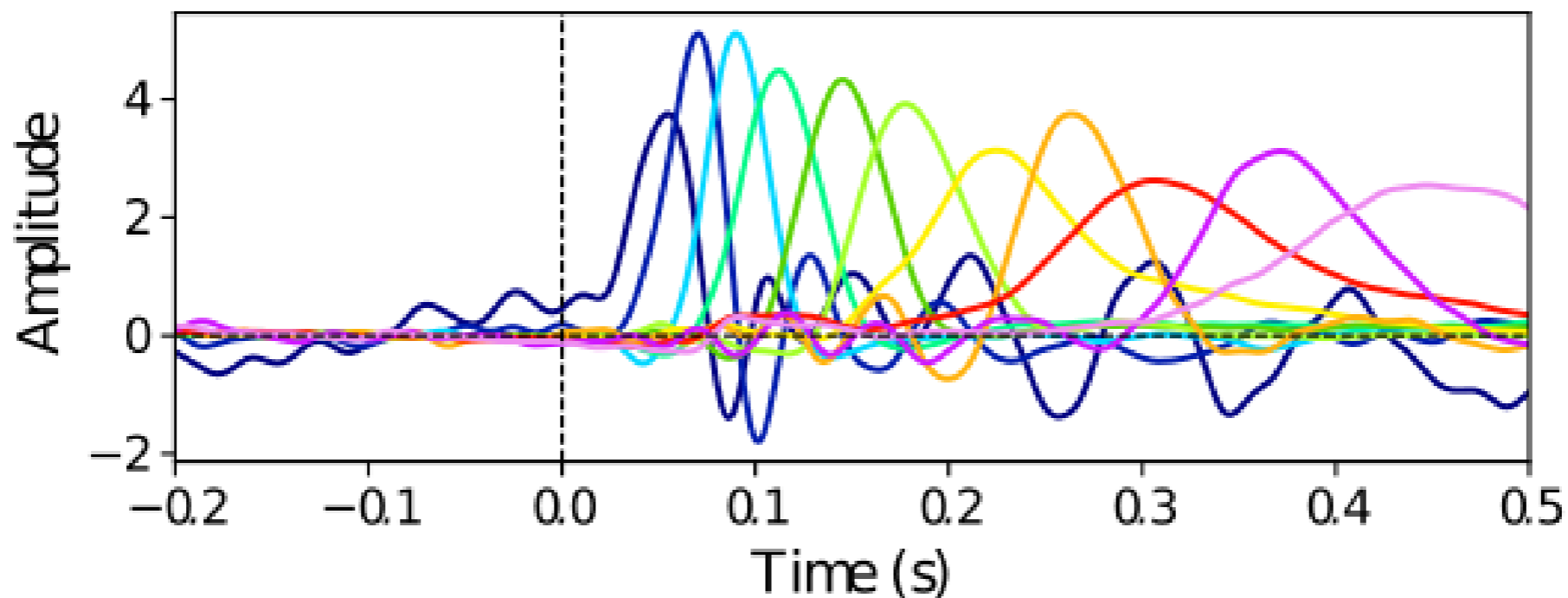
fMRI time segment matching

We select a target time-segment (9 consecutive timeframes) in the shared responses and try to localize the corresponding time-segment in the sources of the left-out subject using a maximum-correlation classifier

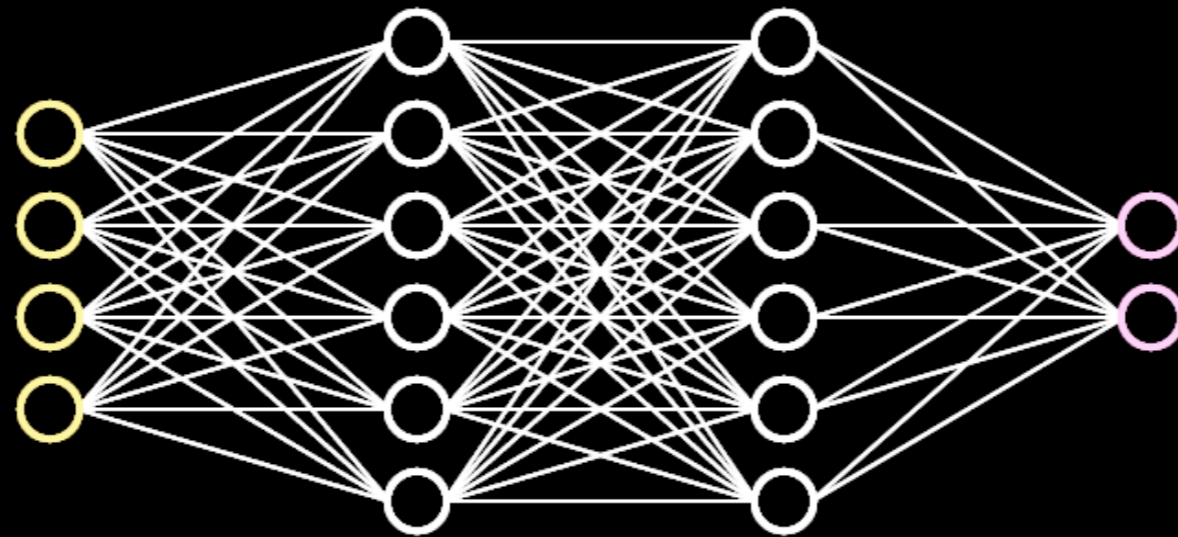


Experiment on MEG data

MEG data from Cam-CAN dataset
with 200 subjects attending to audiovisual stimuli



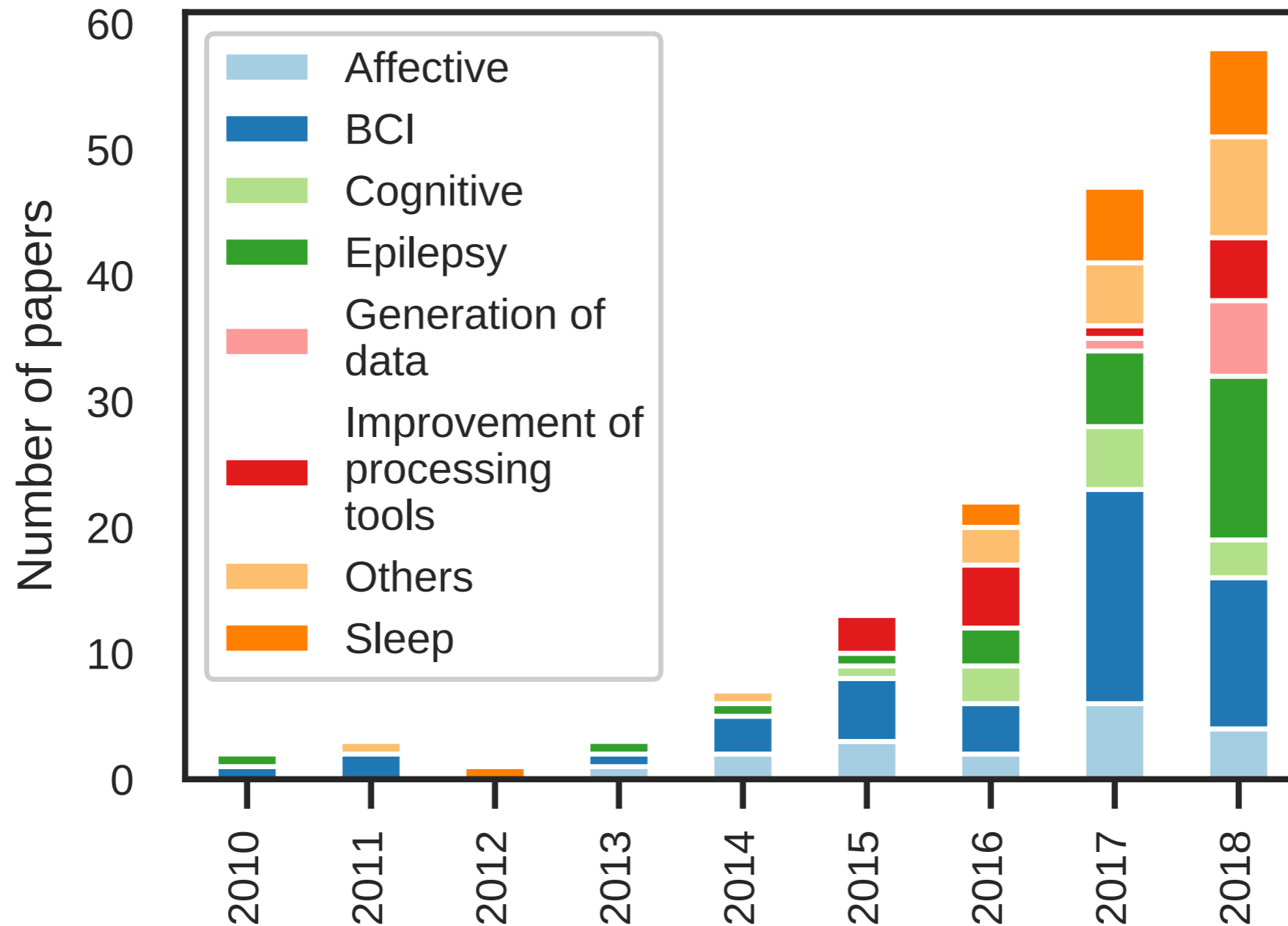
Self-supervised learning on EEG



Uncovering the structure of clinical EEG signals with self-supervised learning
Banville, H., Chehab, O., Hyvärinen, A., Engemann, D. and Gramfort, A. (2020)
Journal of Neural Engineering & ArXiv abs/2007.16104

Self-supervised representation learning from electroencephalography signals
Banville, H., Albuquerque, I., Moffat, G., Engemann, D. and Gramfort, A. (2019)
Proc. Machine Learning for Signal Processing (MLSP) .

Deep Learning papers on EEG



[Deep learning-based electroencephalography analysis: a systematic review
Roy, Y., Banville, H., Albuquerque, I., Gramfort, A., Falk, T. and Faubert, J. (2019)
Journal of Neural Engineering 16: (051001).]

Self-supervision



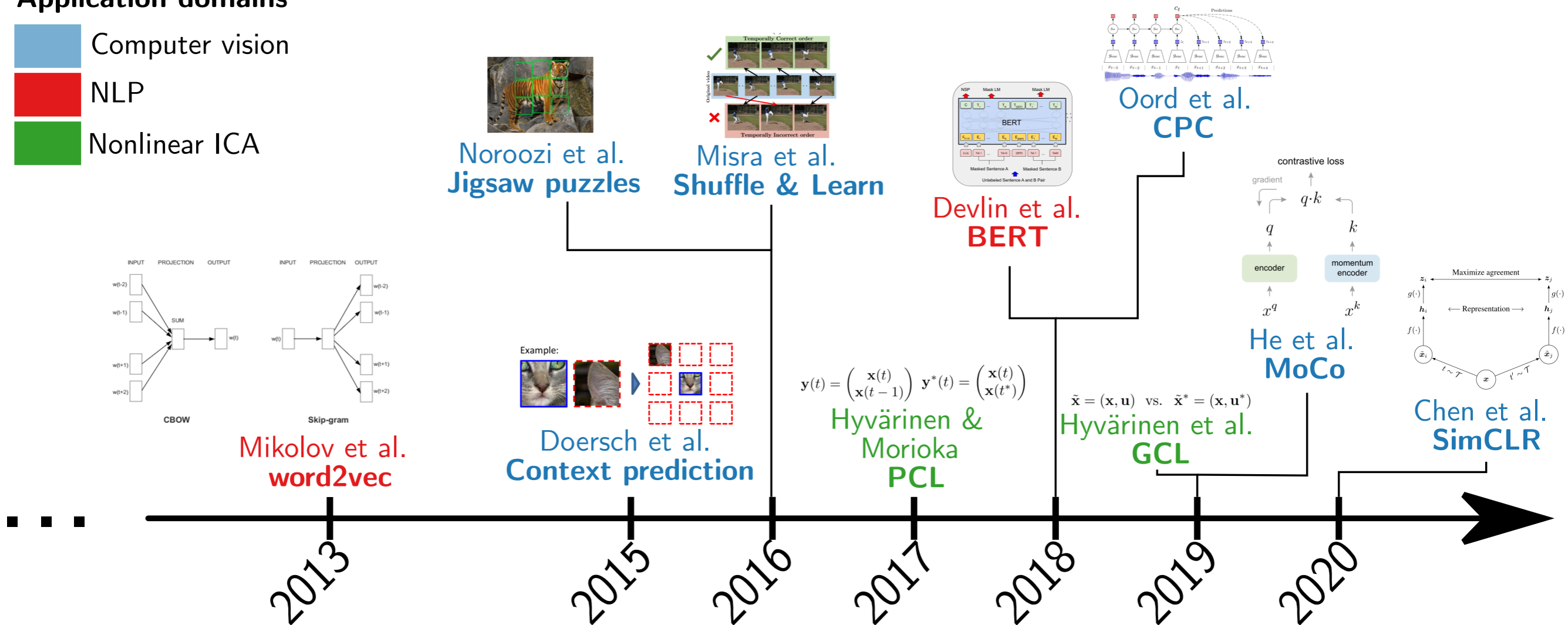
[Noroozi & Favaro 2016] use a deep neural network to solve the Jigsaw puzzle



Use the **structure** of the data to pretrain a **feature extractor** with a supervised *pretext task* — then use the features on a *downstream task*.

(Partial) History of SSL beyond EEG

Application domains



Remarks: Heavily based on contrastive learning and possibly data augmentation techniques

Polysomnography (PSG)

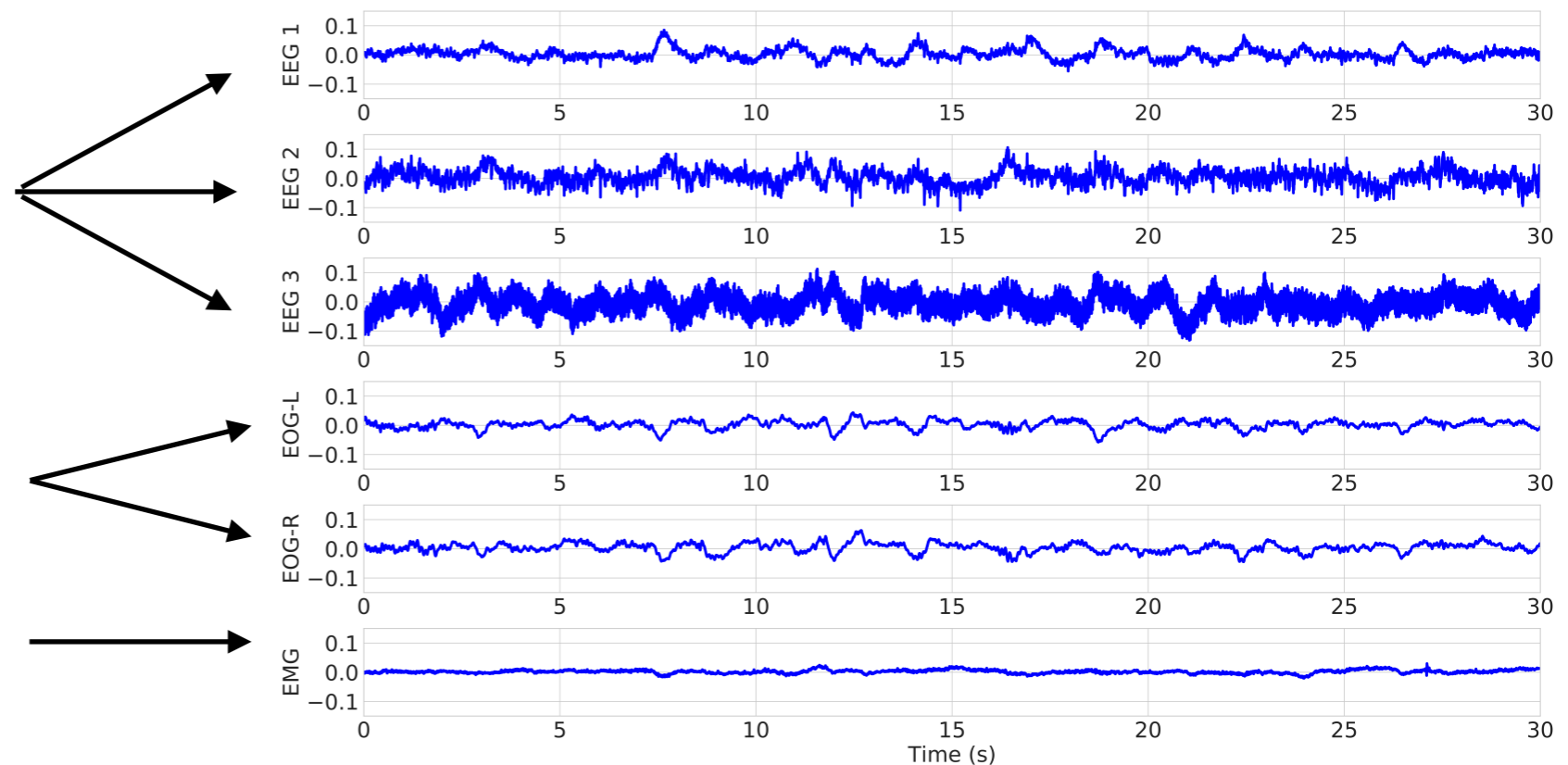
- Clinical exam
- Electrophysiological signals



Electro-encephalography
(EEG)

Electro-oculography
(EOG)

Electro-myography
(EMG)



Polysomnography (PSG)

- Clinical exam
- Electrophysiological signals

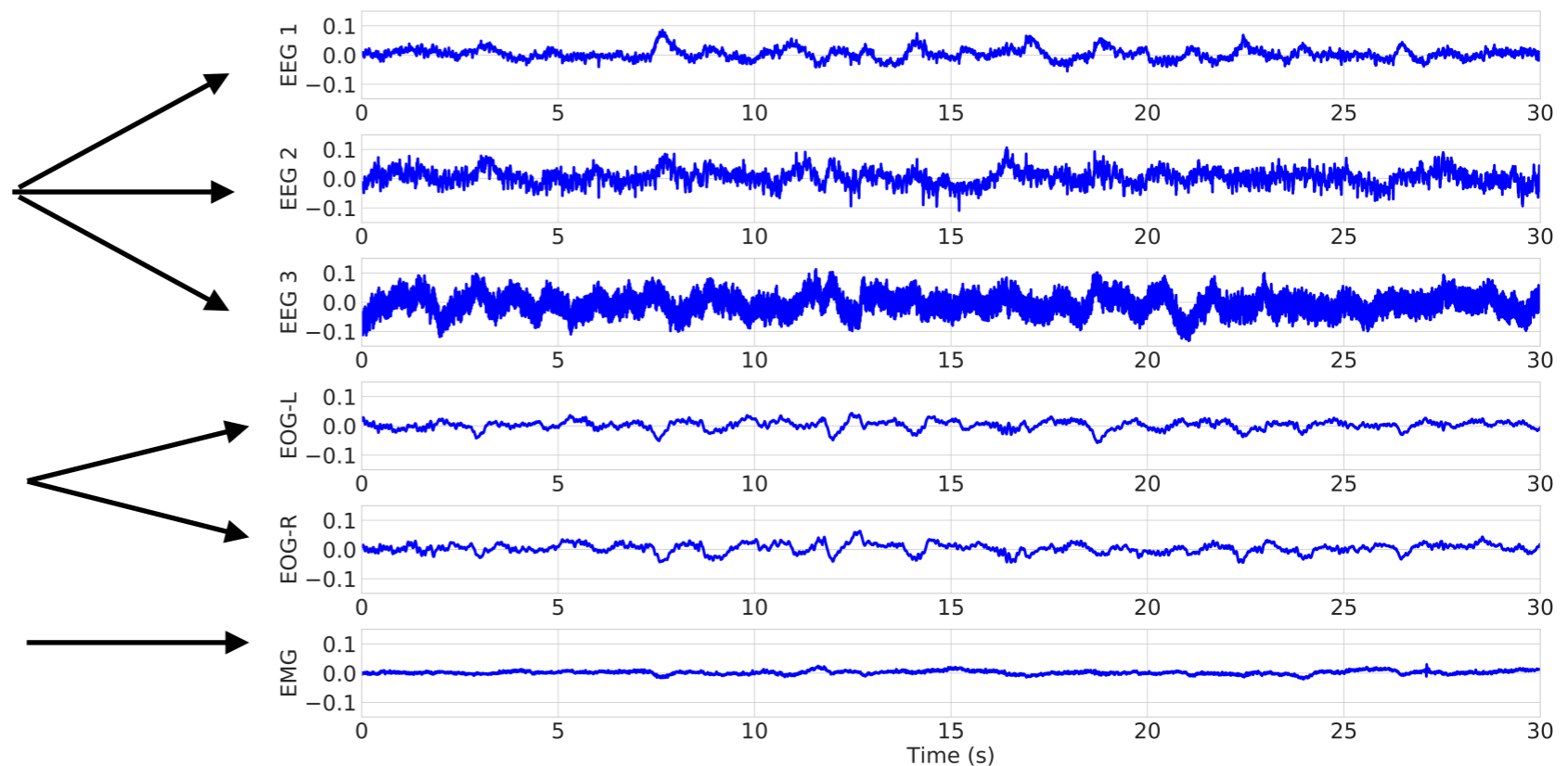
Routinely annotated by
sleep experts



Electro-encephalography
(EEG)

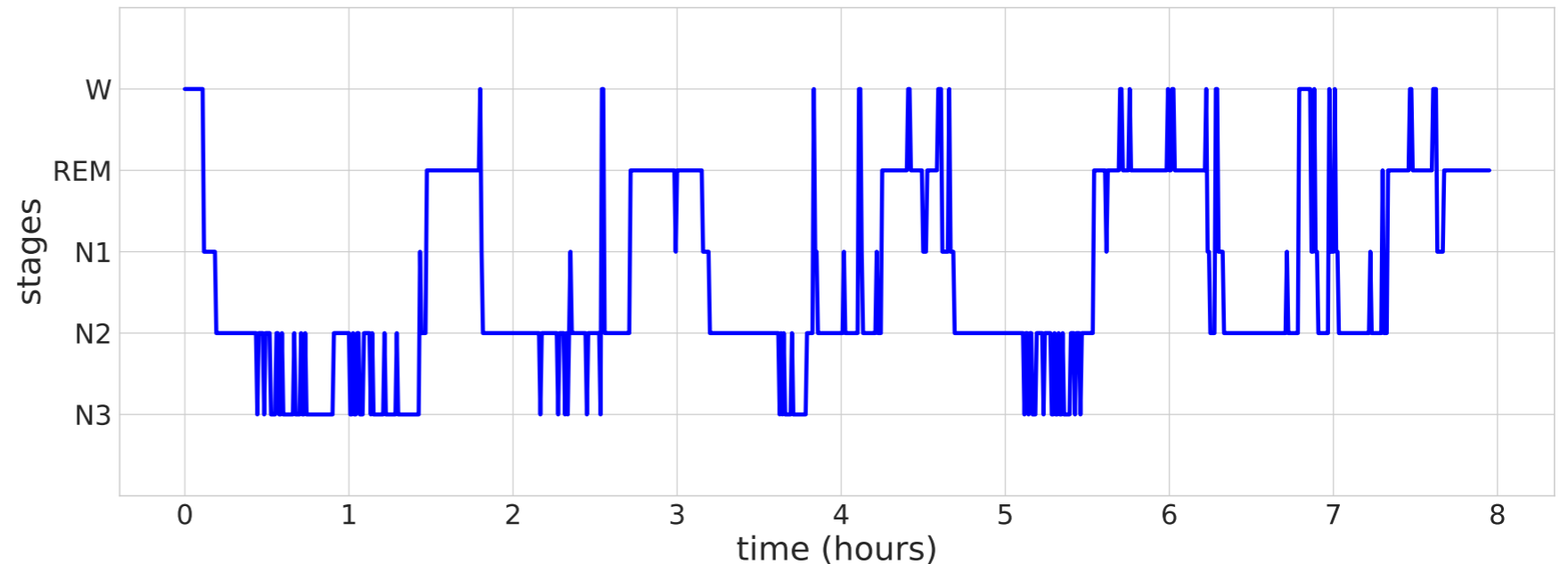
Electro-oculography
(EOG)

Electro-myography
(EMG)



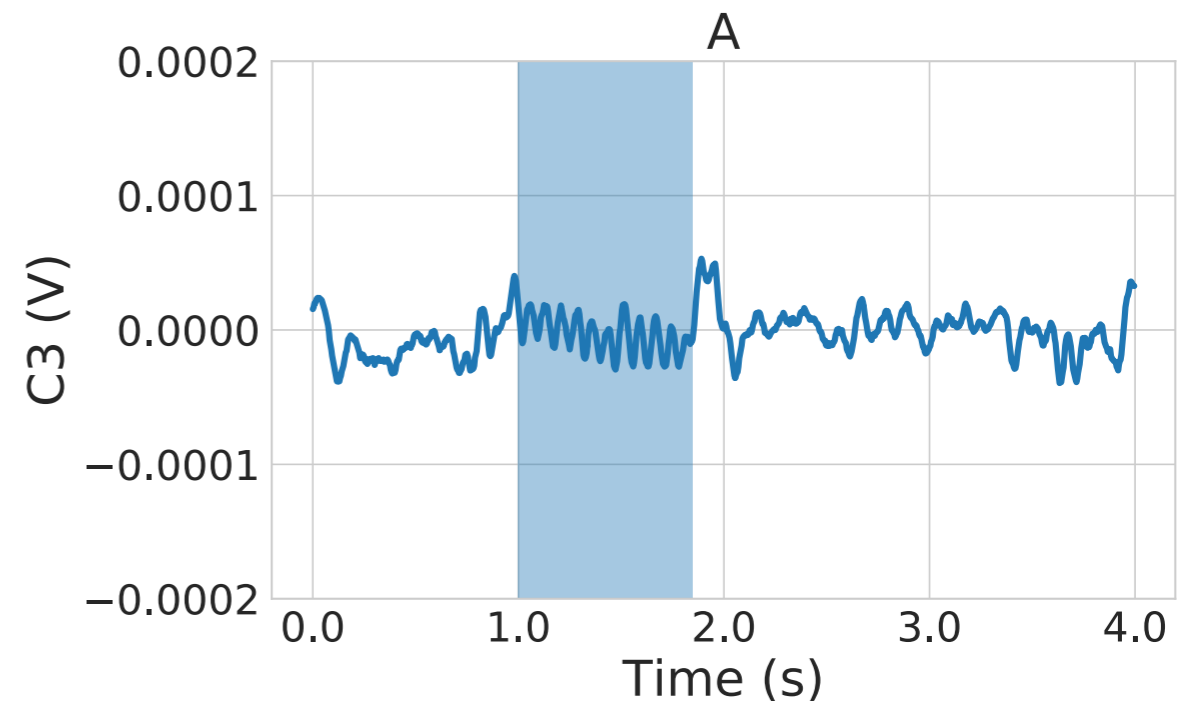
2 types of annotations

Hypnogram of sleep stages



[S. Chambon et al. (2018), *IEEE Trans. Neural Systems and Rehabilitation Engineering*]

Micro-events: Spindles, K-complex etc.

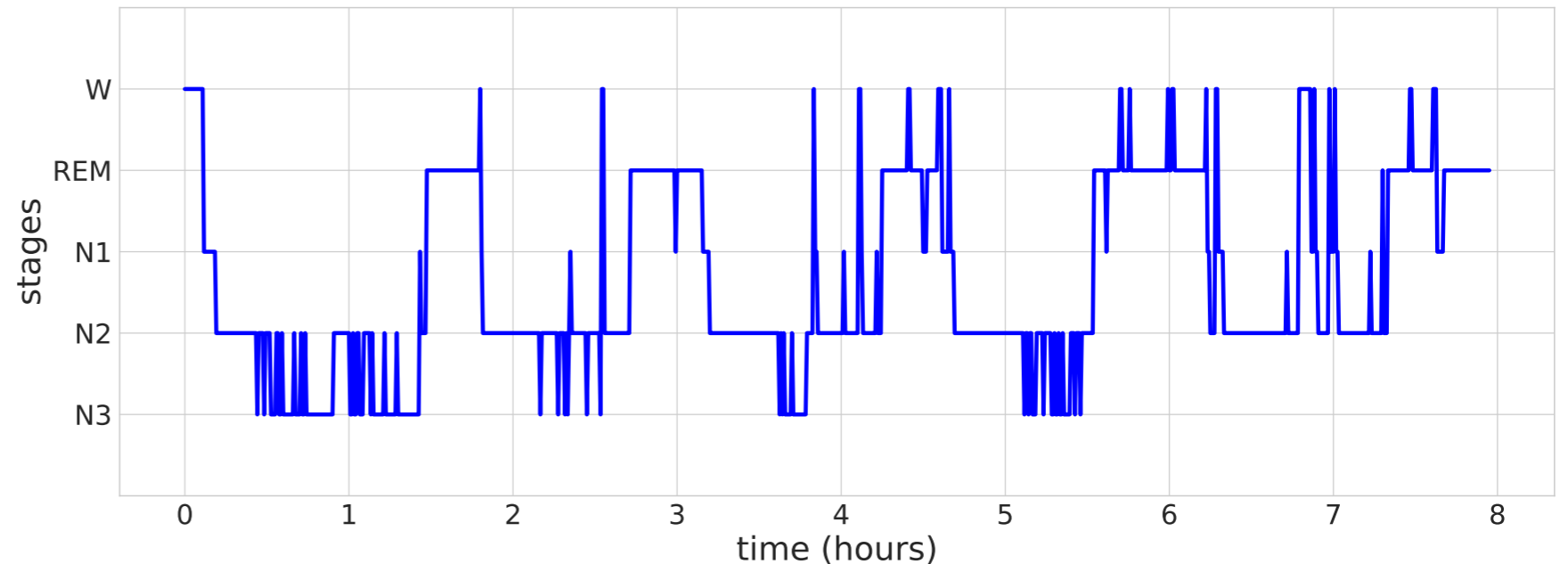


[S. Chambon et al. (2018), *J. of Neuroscience Methods*]

2 types of annotations

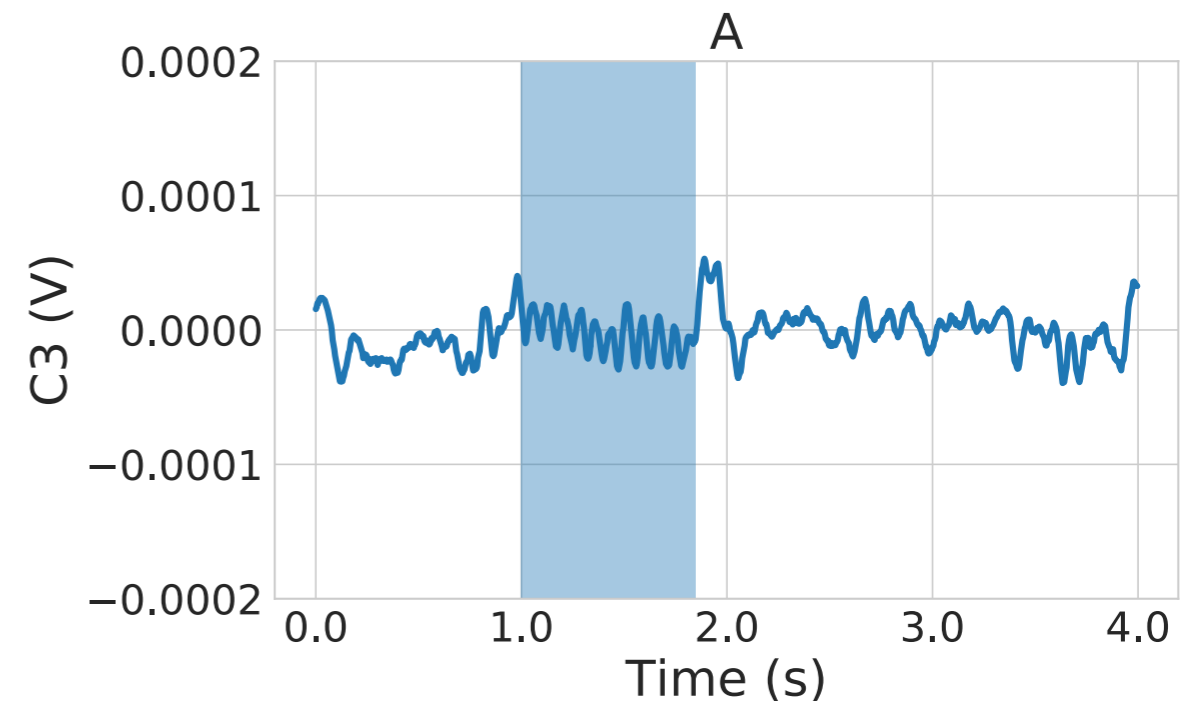
Hypnogram of sleep stages

**Classification
problem**



[S. Chambon et al. (2018), IEEE Trans. Neural Systems and Rehabilitation Engineering]

Micro-events: Spindles, K-complex etc.

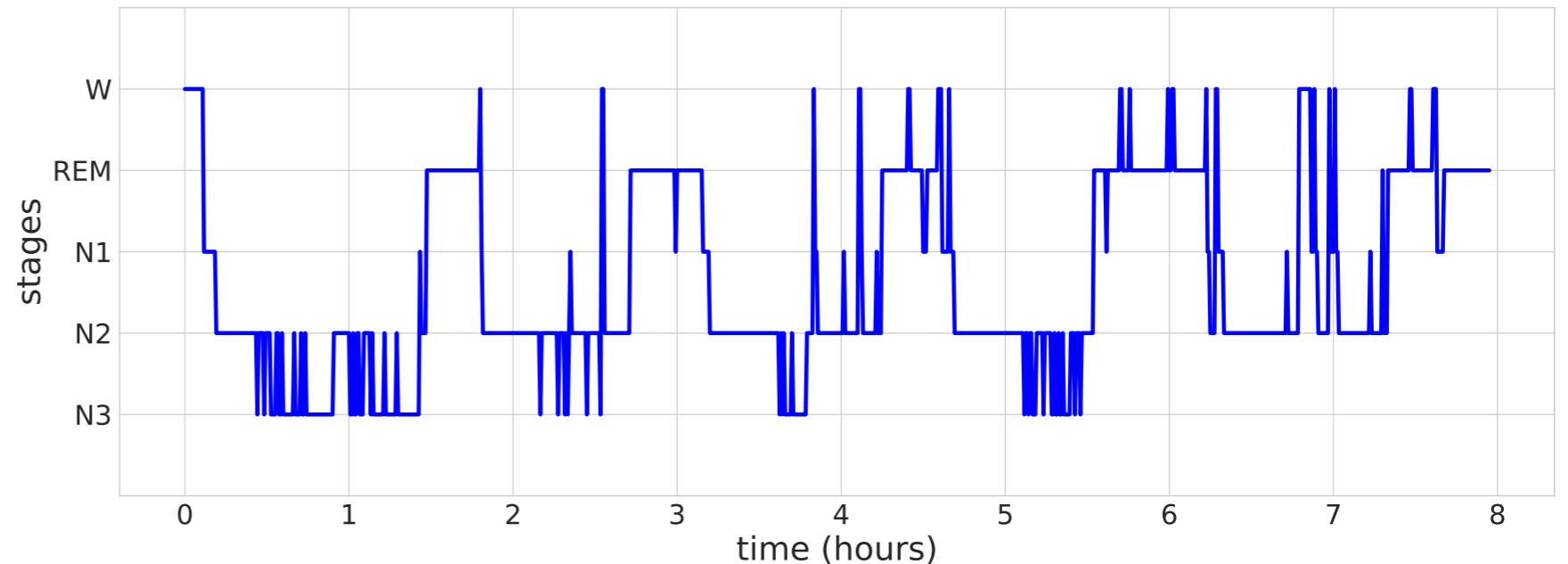


[S. Chambon et al. (2018), J. of Neuroscience Methods]

2 types of annotations

Hypnogram of sleep stages

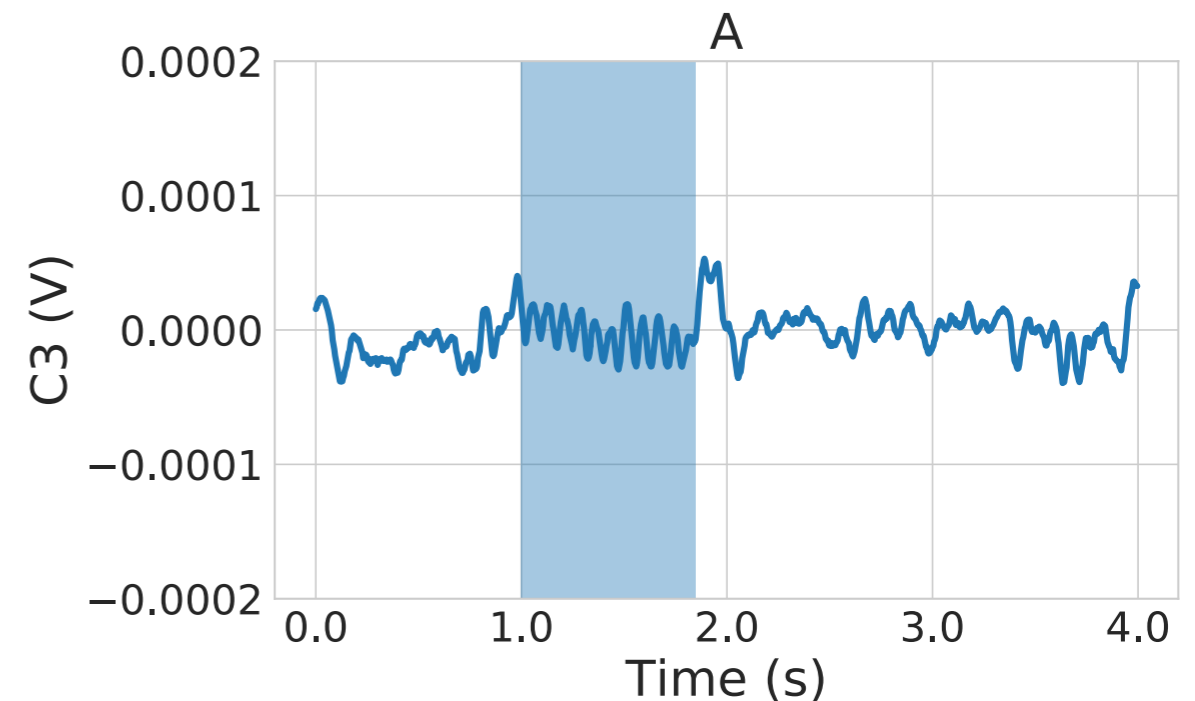
**Classification
problem**



[S. Chambon et al. (2018), IEEE Trans. Neural Systems and Rehabilitation Engineering]

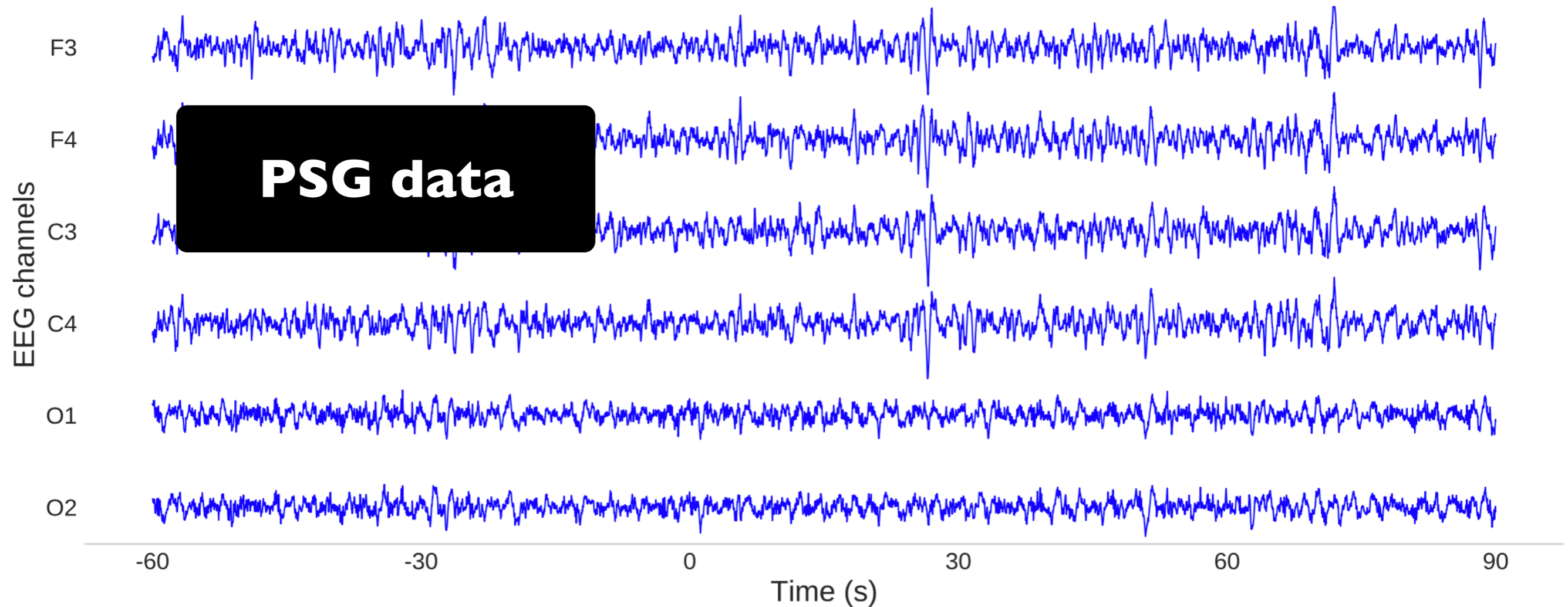
Micro-events: Spindles, K-complex etc.

**Joint detection and
classification problem**

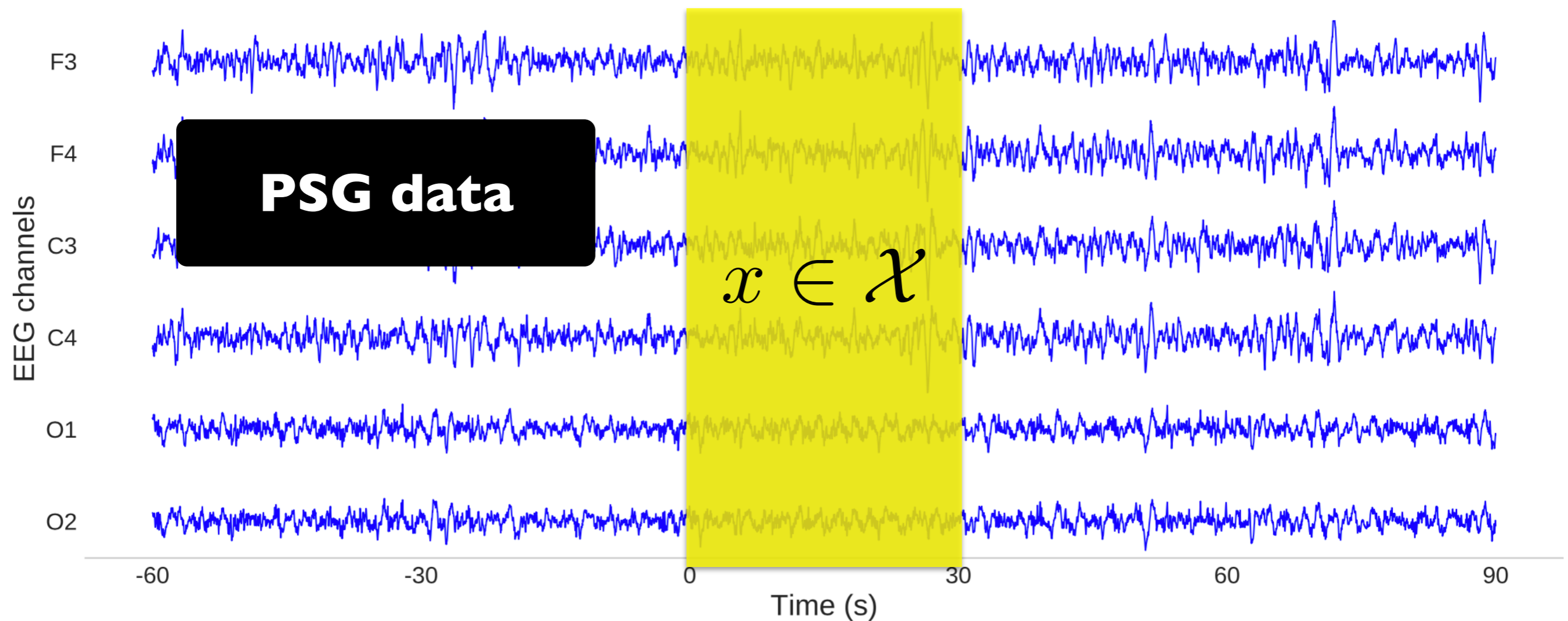


[S. Chambon et al. (2018), J. of Neuroscience Methods]

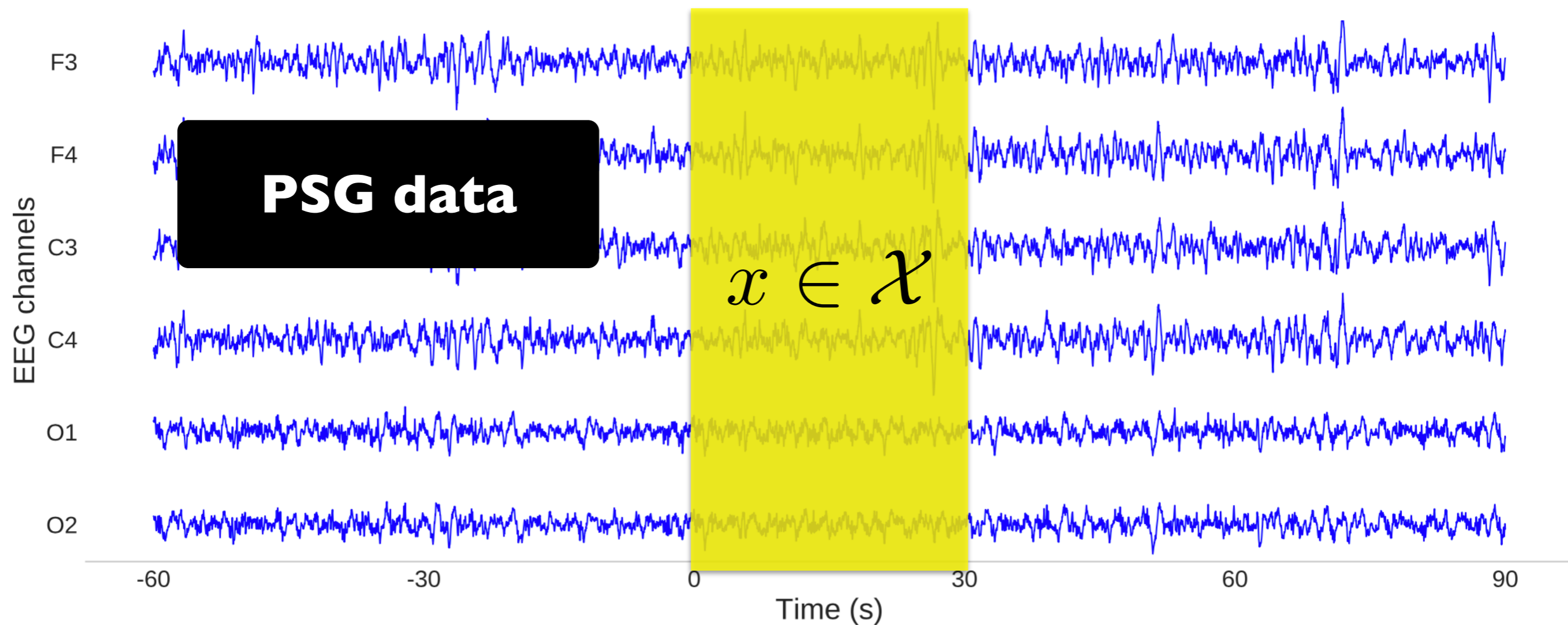
Sleep scoring: Downstream Task



Sleep scoring: Downstream Task



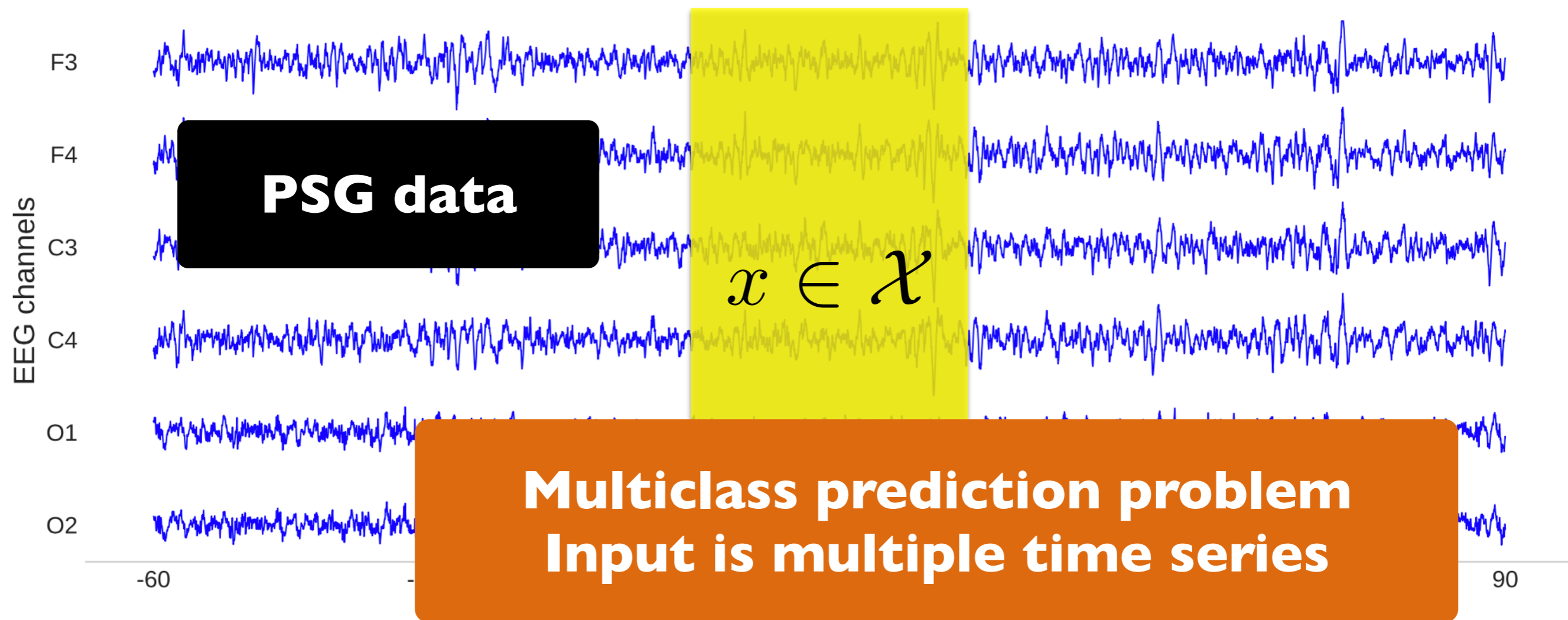
Sleep scoring: Downstream Task



$$\text{Learn: } \hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$$

$$\mathcal{Y} = \{\text{Awake, REM, Stage 1, Stage 2, etc.}\}$$

Sleep scoring: Downstream Task



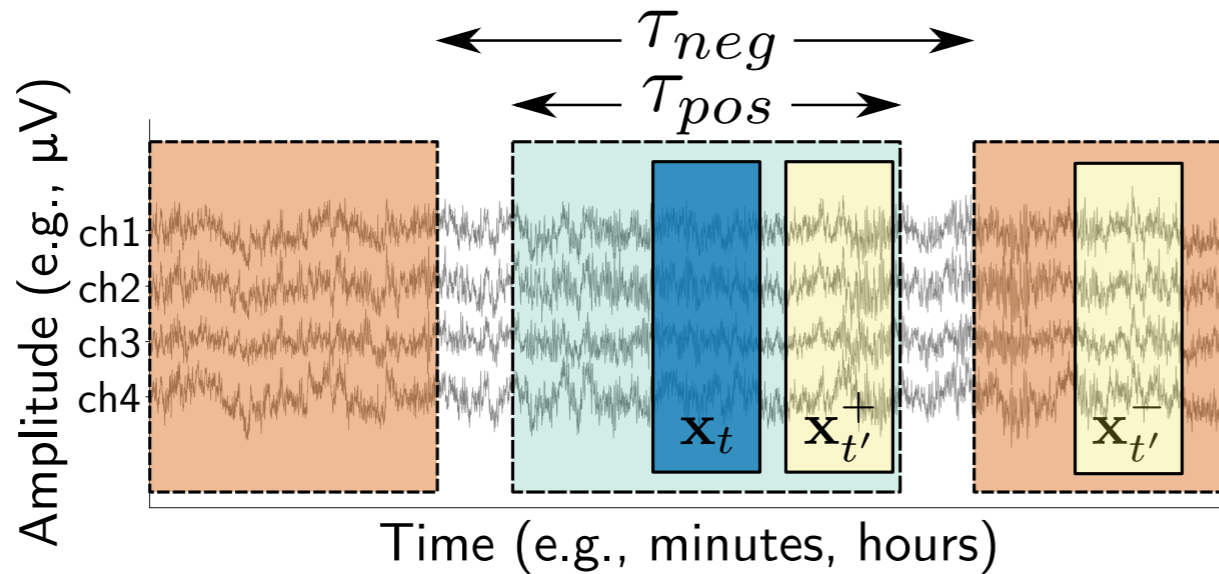
$$\text{Learn: } \hat{f} : \mathcal{X} \rightarrow \mathcal{Y}$$

$$\mathcal{Y} = \{\text{Awake, REM, Stage 1, Stage 2, etc.}\}$$

Pretext Task

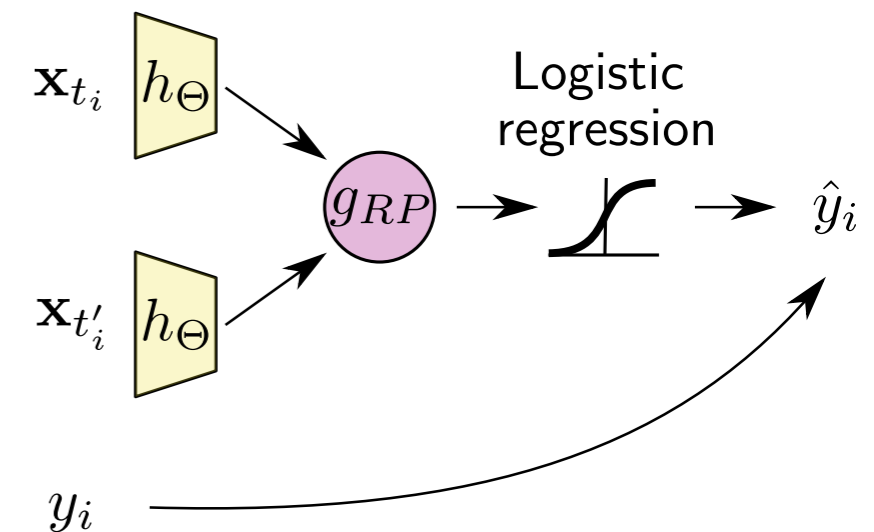
Relative
positioning (RP)

1 Sampling



$$y_i = \begin{cases} 1, & \text{if } |t_i - t'_i| \leq \tau_{pos} \\ -1, & \text{if } |t_i - t'_i| > \tau_{neg} \end{cases}$$

2 Training



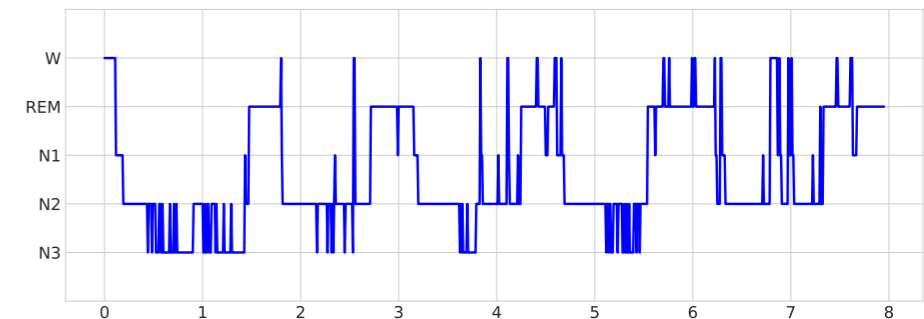
Predict if 2 windows of data are close in time

Related to PCL [Hyvärinen et al. 2017]

Downstream tasks on clinical EEG

Sleep staging:

Predict sleep stage from EEG
(5-class: W, N1, N2, N3, R)



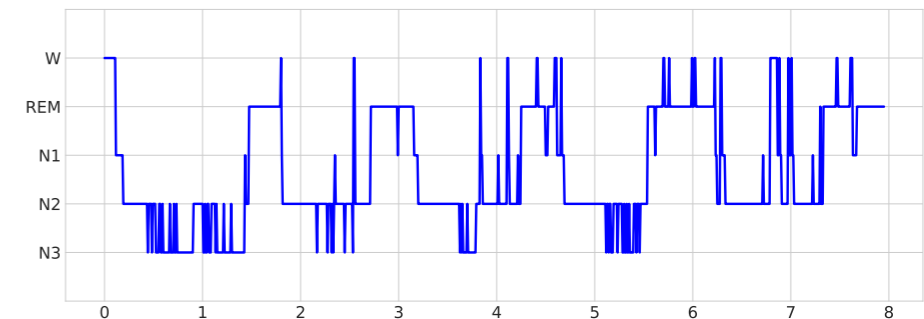
Hypnogram

- **Dataset:** Physionet Challenge 2018 (PC18) [Ghassemi et al. 2018]

Downstream tasks on clinical EEG

Sleep staging:

Predict sleep stage from EEG
(5-class: W, N1, N2, N3, R)



Hypnogram

- **Dataset:** Physionet Challenge 2018 (PC18) [Ghassemi et al. 2018]

Pathology detection:

Is someone's EEG pathological?
(2-class: normal, abnormal)



- **Dataset:** TUH Abnormal EEG (TUHab) [López 2017]

Datasets

Two public datasets with hundreds of recordings:

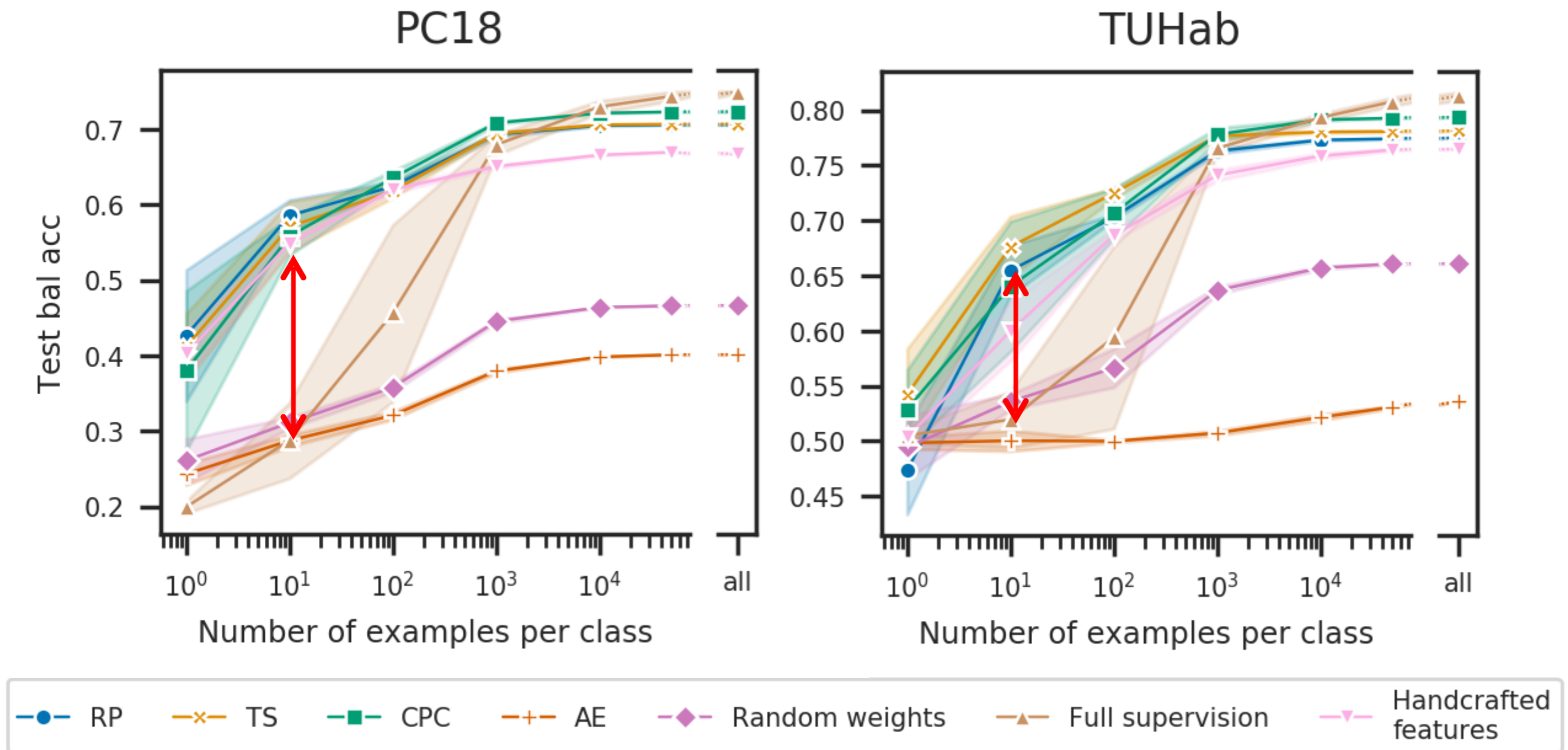
- **Sleep staging:** Physionet Challenge 2018 (PC18) [Ghassemi et al. 2018]
- **Pathology detection:** TUH Abnormal EEG (TUHab) [López 2017]

PC18 (train)	
	# windows
W	158,020
N1	136,858
N2	377,426
N3	102,492
R	116,872
Total	891,668
<hr/>	
# unique subjects	994
# recordings	994
Sampling frequency	200 Hz
# EEG channels	6
Reference	M1 or M2

	TUHab (train)	TUHab (eval)
	# recordings	# recordings
Normal	1371	150
Abnormal	1346	126
Total	2717	276
<hr/>		
# unique subjects	2329	
# recordings	2993	
Sampling frequency	250, 256, 512 Hz	
# EEG channels	27 to 36	
Reference	Common average	

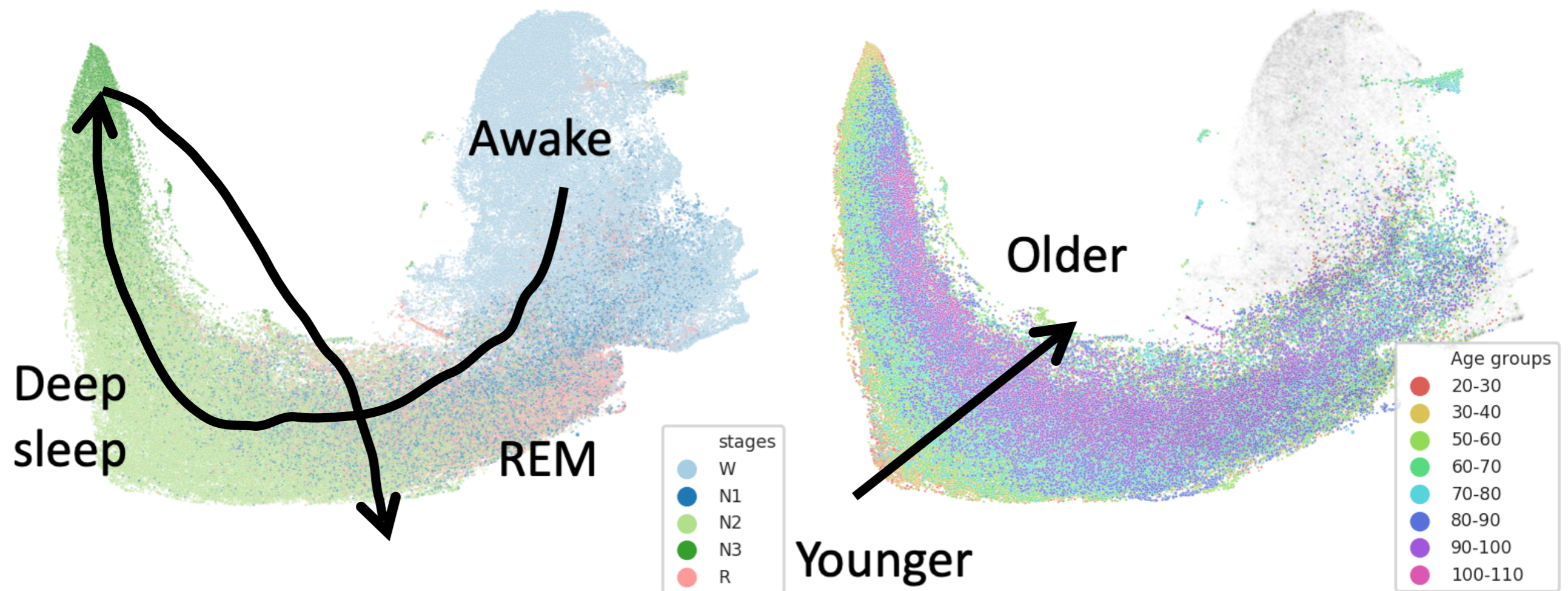
Results: Prediction accuracy

Using a CNN (2 conv/relu/mp) layers described in [Chambon et al. 2017]



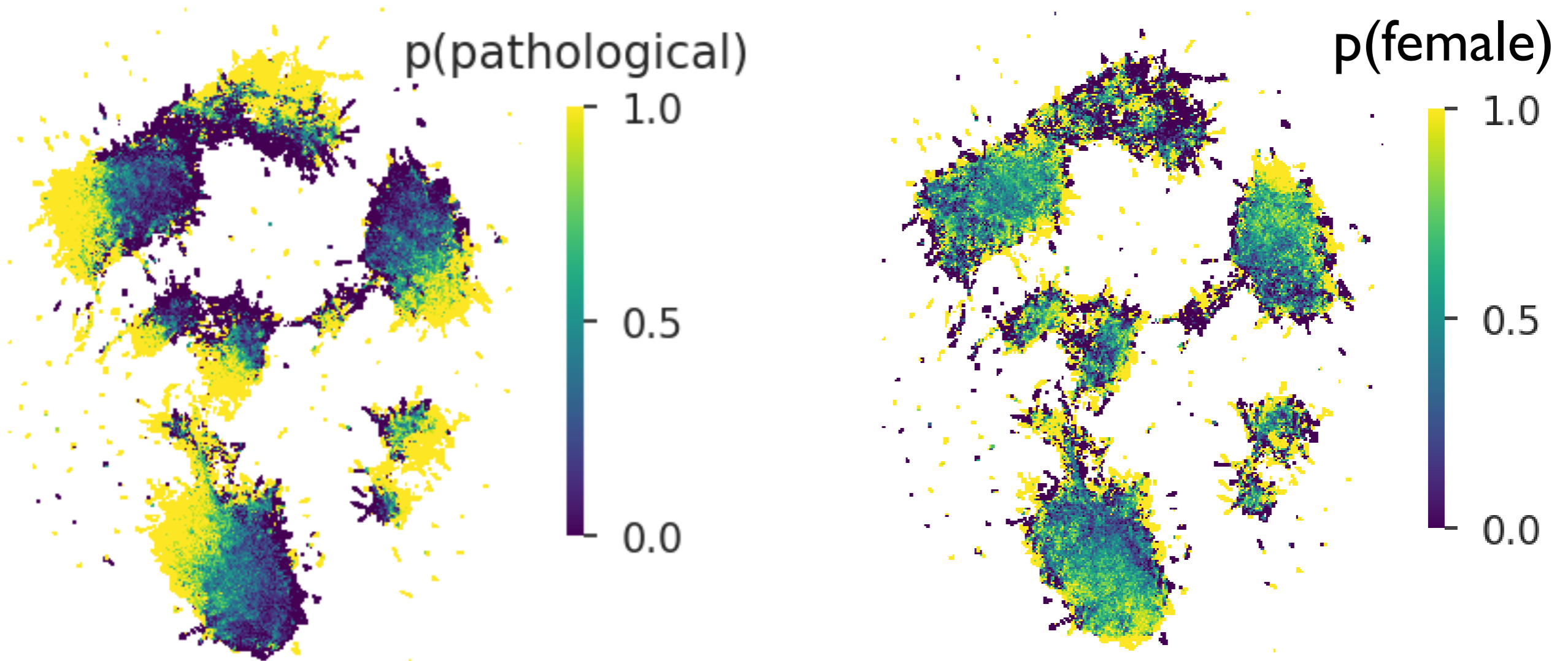
SSL is **better than full supervision** when limited data is available, and **competitive** when all data is available.

Results on sleep EEG



SSL can **uncover structure** without human supervision

Results on TUH data



SSL can **uncover clinically-relevant structure**
without human supervision

A young person with short, curly brown hair and black-rimmed glasses is looking directly at the camera. They are wearing a light blue shirt. In the foreground, their hands are positioned over a beige, vintage-style computer keyboard. The background is a dark blue-grey gradient with vertical streams of white and yellow characters, including numbers (0-9), letters (a-z), and symbols (x, c, w, u, t), falling like digital rain. An orange rounded rectangle is overlaid on the lower half of the image, containing white text.

“All models are wrong but some come with good open source implementation and good documentation so use those.”



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: *SVM, nearest neighbors, random forest, ...*

— Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: *SVR, ridge regression, Lasso, ...*

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: *k-Means, spectral clustering, mean-shift, ...*

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: *PCA, feature selection, non-*

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: *grid search, cross validation*

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: *preprocessing, feature extraction.*

http://mne.tools

The screenshot shows the mne.tools website in a web browser. The browser's address bar displays 'mne.tools'. The website header features the 'MNE' logo with a rainbow gradient bar to its right. Navigation links include 'Install', 'Documentation', 'API Reference', 'Get help', and 'Development'. A blue button shows the current version 'v0.23.4' with a dropdown arrow. Social media icons for GitHub, Twitter, and Discord are also present.

On the left side, there is a search bar labeled 'Search the docs ...'. Below it, a section titled 'Version 0.23.4' contains a list of links: 'Tutorials', 'Changelog', 'Get help', 'Cite', and 'Contribute'.

The main content area features a large 'MNE' logo with a rainbow gradient. Below the logo, the text 'MEG + EEG ANALYSIS & VISUALIZATION' is displayed. A paragraph describes the package as an 'Open-source Python package for exploring, visualizing, and analyzing human neurophysiological data: MEG, EEG, sEEG, ECoG, NIRS, and more.'

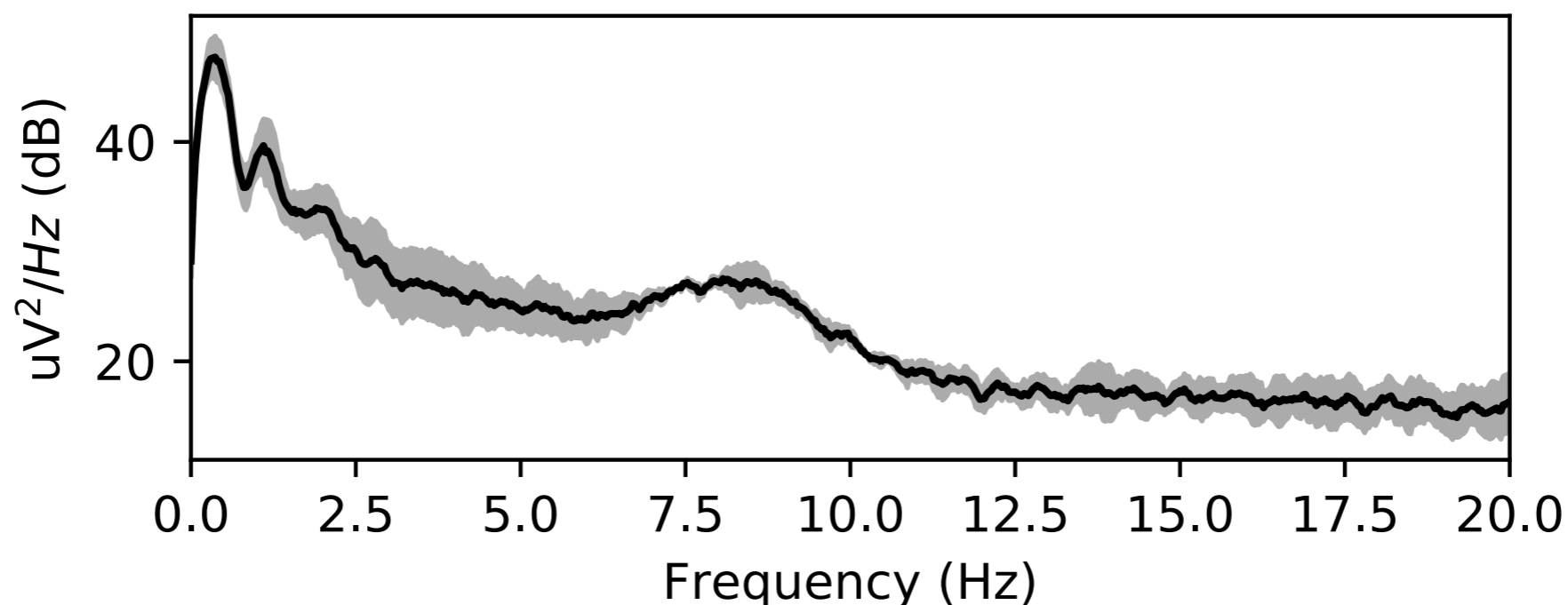
At the bottom, there are six feature cards arranged in a 2x3 grid:

- Source Estimation**: Distributed, sparse, mixed-norm, beamformers, dipole fitting, and more. (Background image: Brain surface map)
- Machine Learning**: Advanced decoding models including time generalization. (Background image: Heatmap with 'Temporal generalization' text)
- Encoding Models**: Receptive field estimation with optional smoothness priors. (Background image: Heatmap with 'Simulated STRF' text)
- Statistics**: (Background image: Brain surface map)
- Connectivity**: (Background image: Brain surface map with 'All-time' and 'LibraryCondition (PLI)' text)
- Data Visualization**: (Background image: Brain surface map)



Working with EEG sleep data in 7 lines of Python code

```
>>> import mne
>>> raw = mne.io.read_raw_edf('SC4001E0-PSG.edf')
>>> annot = mne.read_annotations('SC4001EC-Hypnogram.edf')
>>> raw.set_annotations(annot)
>>> events, event_id = mne.events_from_annotations(raw)
>>> epochs = mne.Epochs(raw, events, event_id, tmin=0.,
                        tmax=30., baseline=None)
>>> epochs['Sleep stage W'].plot_psd(fmax=20., picks=[0, 1])
```



<https://braindecode.org>

The screenshot shows a web browser window displaying the Braindecode 0.5.1 documentation. The browser's address bar shows `braindecode.org`. The page title is "Sleep staging on the Sleep Physionet dataset — Braindecode 0.5.1 documentation". The left sidebar contains a navigation menu with the following items:

- Braindecode 0.5.1
- Search docs
- Basic trialwise decoding
- More data-efficient "cropped decoding"
- Your own datasets through MNE
- Your own datasets through Numpy
- Braindecode examples
 - Split Dataset Example
 - Custom Dataset Example
 - Load and save dataset example
 - MNE Dataset Example
 - MOABB Dataset Example
 - Regression example on fake data
- Sleep staging on the Sleep Physionet dataset
 - References
 - Loading and preprocessing the dataset
 - Create model
 - Training

The main content area shows the breadcrumb navigation: [» Braindecode examples](#) » Sleep staging on the Sleep Physionet dataset, with a [View page source](#) link. Below this is a blue "Note" box with the text: "Click [here](#) to download the full example code". The main heading is "Sleep staging on the Sleep Physionet dataset". The text below the heading reads: "This tutorial shows how to train and test a sleep staging neural network with Braindecode. We follow the approach of ¹ on the openly accessible Sleep Physionet dataset ^{1 2}." The "References" section lists three references:

- [1] (^{1,2,3,4,5,6}): Chambon, S., Galtier, M., Arnal, P., Wainrib, G. and Gramfort, A. (2018) A Deep Learning Architecture for Temporal Sleep Stage Classification Using Multivariate and Multimodal Time Series. *IEEE Trans. on Neural Systems and Rehabilitation Engineering* 26: (758-769)
- [2] : B Kemp, AH Zwinderman, B Tuk, HAC Kamphuisen, JJJ Oberyé. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the EEG. *IEEE-BME* 47(9):1185-1194 (2000).
- [3] : Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng C-K, Stanley HE. (2000) PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101(23):e215-e220

<https://braindecode.org>

The screenshot shows a web browser window with the URL braindecode.org. The page title is "Self-supervised learning on EEG with relative positioning — Braindecode 0.5.1 documentation". The left sidebar contains a list of examples under the heading "Braindecode examples". The main content area shows the breadcrumb "» Braindecode examples » Self-supervised learning on EEG with relative positioning" and a link to "View page source". A blue note box contains the text "Click [here](#) to download the full example code". The main heading is "Self-supervised learning on EEG with relative positioning". The text below the heading explains that this example shows how to train a neural network with self-supervision on sleep EEG data, following the relative positioning approach of ¹ on the Sleep Physionet dataset ^{2 3}. A section titled "Self-supervised learning" defines SSL as a learning paradigm that leverages unlabelled data to train neural networks, describing the pretext task and its relation to the downstream task.

Braindecode examples

- Split Dataset Example
- Custom Dataset Example
- Load and save dataset example
- MNE Dataset Example
- MOABB Dataset Example
- Regression example on fake data
- Sleep staging on the Sleep Physionet dataset
- Trialwise Decoding on BCIC IV 2a Dataset
- Cropped Decoding on BCIC IV 2a Dataset
- Process a big data EEG resource (TUH EEG Corpus)
- Benchmarking eager and lazy loading

Self-supervised learning on EEG with relative positioning

- Loading and preprocessing the dataset
- Creating the model
- Training
- Visualizing the results

» Braindecode examples » Self-supervised learning on EEG with relative positioning

[View page source](#)

Note

Click [here](#) to download the full example code

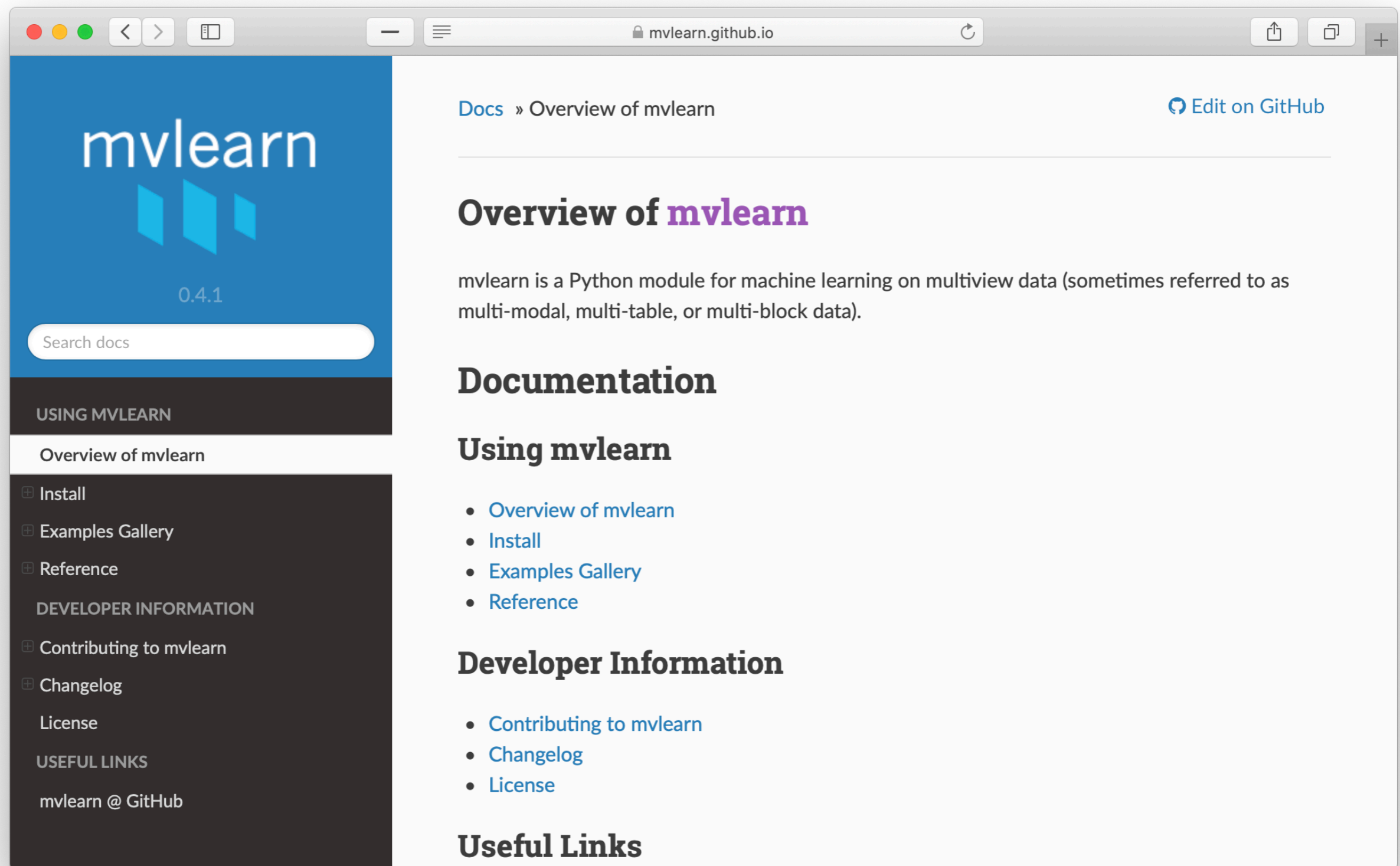
Self-supervised learning on EEG with relative positioning

This example shows how to train a neural network with self-supervision on sleep EEG data. We follow the relative positioning approach of ¹ on the openly accessible Sleep Physionet dataset ^{2 3}.

Self-supervised learning

Self-supervised learning (SSL) is a learning paradigm that leverages unlabelled data to train neural networks. First, neural networks are trained on a “pretext task” which uses unlabelled data only. The pretext task is designed based on a prior understanding of the data under study (e.g., EEG has an underlying autocorrelation structure) and such that the processing required to perform well on this pretext task is related to the processing required to perform well on another task of interest. Once trained, these neural networks can be reused as feature extractors or weight initialization in a “downstream task”, which is the task that we are actually interested in (e.g., sleep staging). The pretext task step can help reduce the quantity of labelled data needed to perform well on the downstream task and/or improve downstream performance as compared to a strictly supervised

<https://mvlearn.github.io>



*[Perry, Ronan, et al. "mvlearn: Multiview Machine Learning in Python."
Journal of Machine Learning Research 22.109 (2021): 1-7.]*

Thanks !

P. Ablin, J-F Cardoso, A. Gramfort (2017), **Faster independent component analysis by preconditioning with Hessian approximations**, IEEE Trans. Sig. Proc.

Richard, H., Gresele, L., Hyvärinen, A., Thirion, B., Gramfort, A., Ablin, P. (2020), **Modeling Shared Responses in Neuroimaging Studies through MultiView ICA**, Proc. NeurIPS

Richard, H., Ablin, P., Thirion, B., Gramfort, A., Hyvärinen, A., P. (2021), **Shared Independent Component Analysis for Multi-Subject Neuroimaging**, Proc. NeurIPS

Banville, H., Chehab, O., Hyvärinen, A., Engemann, D. and Gramfort, A. (2020), **Uncovering the structure of clinical EEG signals with self-supervised learning**, J. Neural Engineering

Contact

<http://alexandre.gramfort.net>

GitHub : @agramfort 

Twitter : @agramfort 

anr

Support

ERC SLAB, ANR-14-NEUC-0002-01
NIH R01 MH106174, ANR AI Chaire BrAIN, ANR AI-Cog

erc