BCILAB and Applications to EEG Cognitive Interfaces

Christian A. Kothe

SCCN, INC, UCSD

Outline

1. High-level View
2. Application Areas and Examples
3. Basic Underlying Theory
4. The BCILAB Toolbox
5. GUI and Scripting Tour
6. Methods Tour
7. Current and Future Directions

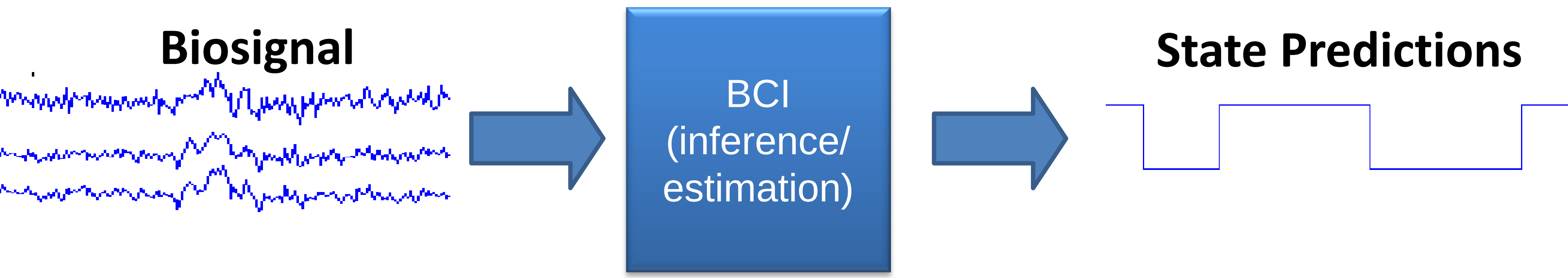
A. Further Reading

1 High-Level Overview

Swartz Center for Computational Neuroscience

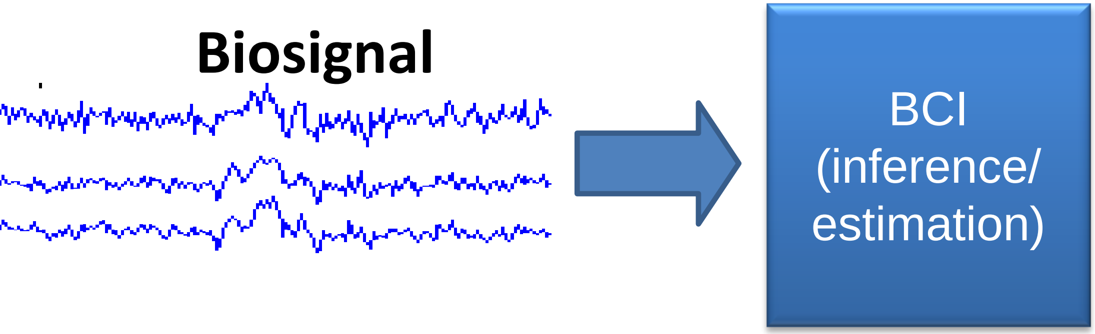
BCI: Our Working Definition

- “A system which takes a biosignal measured from a person and predicts (in real time / on a single-trial basis) some abstract aspect of the person's cognitive state.”



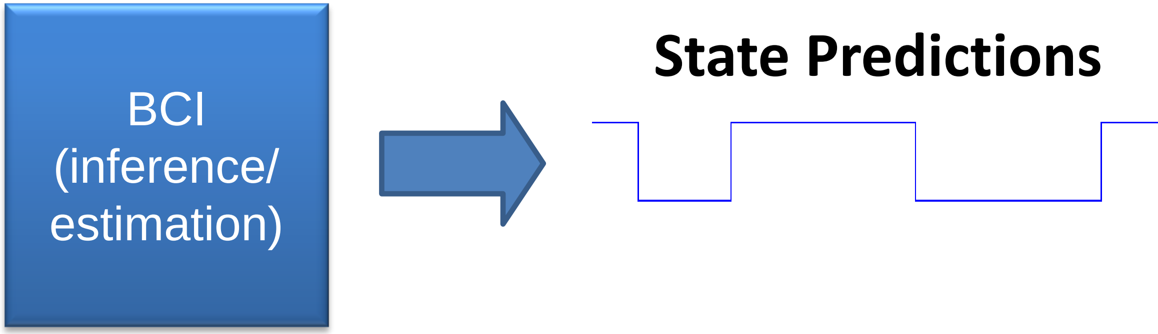
Biosignals and other Inputs

- **Brain Signals:** EEG, fNIRS, MEG, fMRI, ECoG, …
- **Peripheral Measures:** ECG, EMG, EOG, GSR, Respiration, Gaze/Pupillometry, Motion Capture
- **Context Information:** Program/System State, Vehicle Speed, …



BCI Estimates/Predictions

- Any aspect of the physical brain state that can be recovered from observable signals (discrete, continuous, multivariate, …)
- **Tonic state:** degree of “relaxation”, cognitive load,…
- **Phasic state:** attention deployment, imagined vowel
- **Event-related state:** surprised/not surprised, committed error, event noticed/not noticed, …



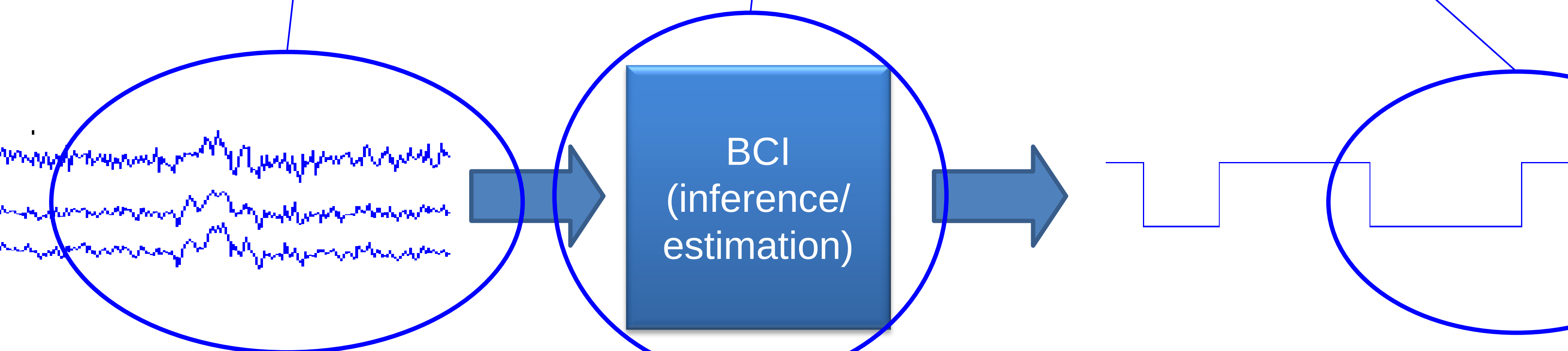
SCCN Software Tools for BCI

EEGLAB, MoBILAB, SIFT, …
(not discussed here today)

Lab Streaming Layer (LSL)
code.google.com/p/labstreaminglayer

Simulation and Neuroscience Application Platform (SNAP)
ftp://sccn.ucsd.edu/pub/SNAP

BCILAB
ftp://sccn.ucsd.edu/pub/bcilab
sccn.ucsd.edu/wiki/BCILAB



2 Application Areas and Examples

Communication and Control for the Severely Disabled

- Severe Disabilities: Tetraplegia, Locked-in syndrome
- **Speller Programs, Wheelchairs, Robots, …**

P300 Speller

KU Leuven

Brain2Robot (Fraunhofer FIRST)

Other Health Uses

- **Sleep Stage Recognition, Neurorehabilitation**

iBrain

Takata et al., 2011

Operator Monitoring

- **Braking Intent, Lane-Change Intent, Workload, Fatigue, Alertness, Attention, …**

Haufe et al., 2011

The MITRE Corp., 2011

Entertainment, Social, etc.

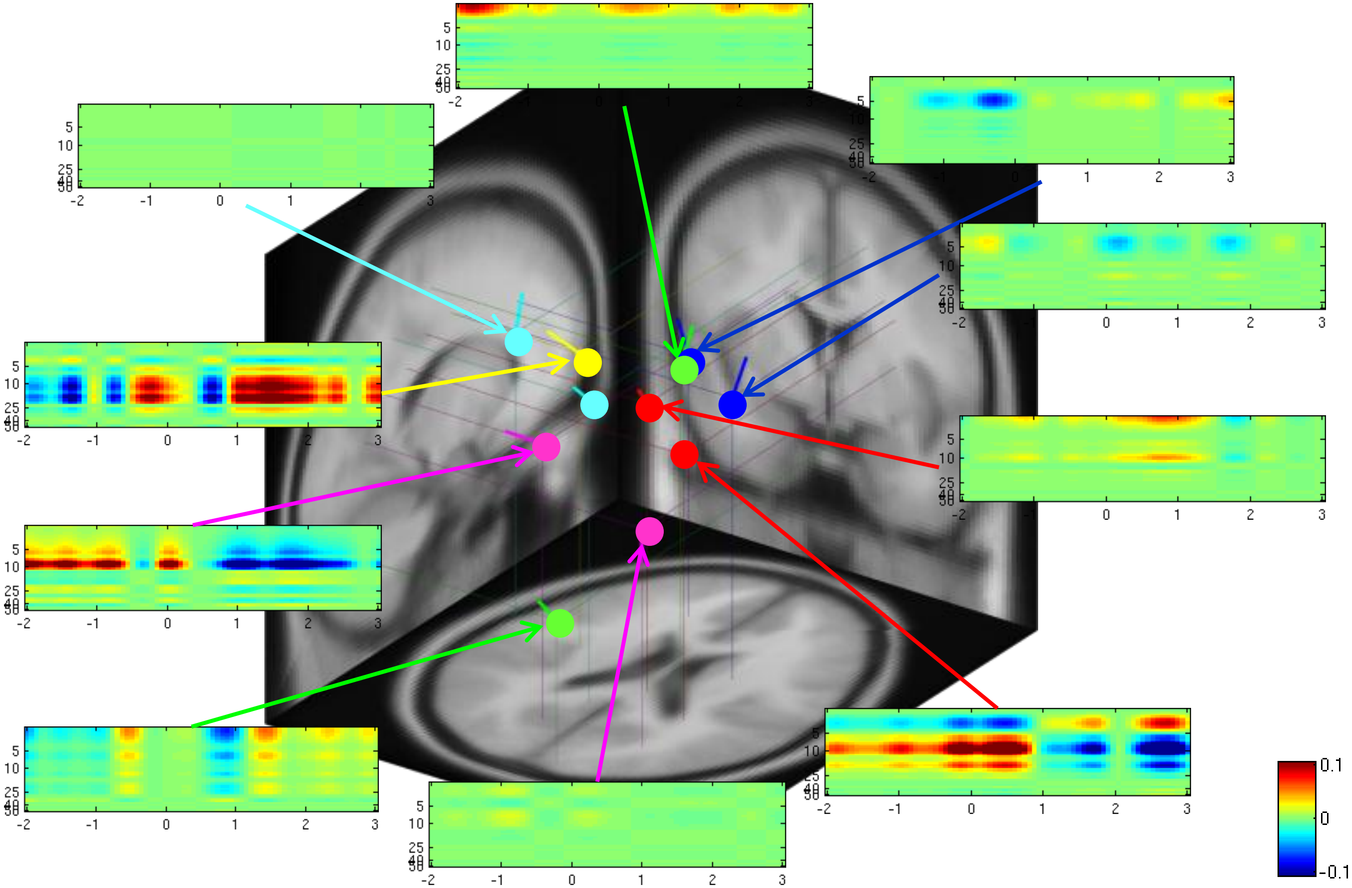
- **Control by Thought, Mood Assessment/Display**

Jedi Game Prototype

necomimi “neurowear”

Neuroscience

- **Multivariate Pattern Analysis / Brain Imaging**



Kothe et al., 2011

4 The BCILAB Toolbox

Software Environment For:

- **Brain-Computer Interface Design** (Cognitive Monitoring)
- **Methods Research**:
  - Design & rapid prototyping of new methods & methods from literature
  - Offline testing, performance evaluation & batch comparison, visualizations
  - Simulated online testing
- **Rapid Prototyping**:
  - Real-time use and testing of BCIs
  - Prototype deployment

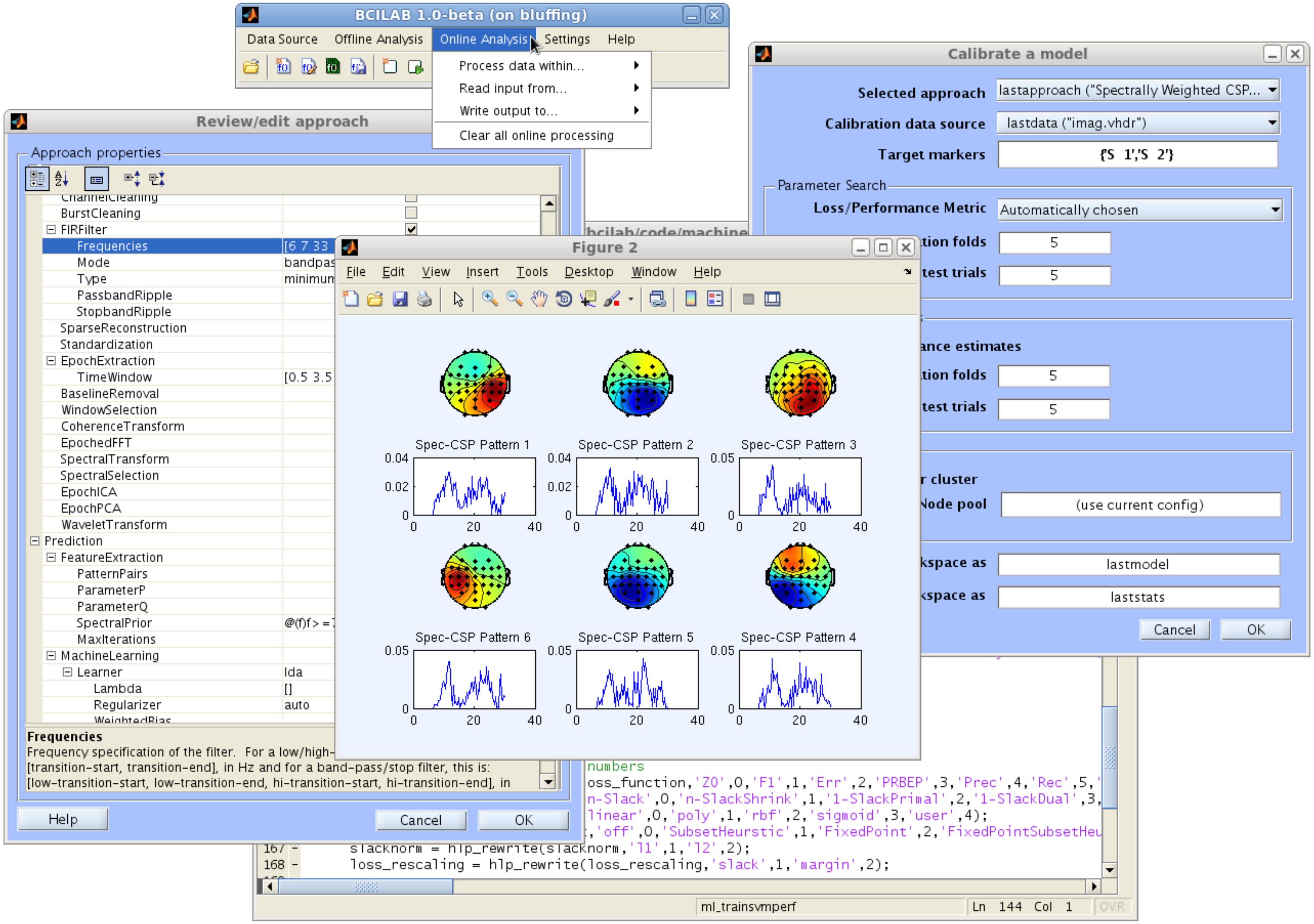
Basic Goals

- Usable by **beginners and experts** to serve both the EEGLAB community and advanced needs
- Include a **large array of methods**, both conventional and state-of-the art, to rapidly set up well-performing BCIs and conduct broad comparison studies
- Provide **convenient plugin frameworks and reusable backend tools** to allow for rapidly prototyping of methods

Facts & Figures

- Developed since 2010 at SCCN, UCSD (primarily by me)
- Precursor was the PhyPA toolbox (Kothe & Zander, 2006-’09)
- Built on top of EEGLAB (Delorme & Makeig, 2004)
- The largest open-source BCI toolbox by methods and algorithms (100+) as of 2011
- Offline and online processing both in MATLAB, same code base, Win/Linux/MacOS, 32/64bit
- Extensive documentation (hundreds of pages of help text, manual, wiki, 400+ lecture slides online)

BCILAB Sample GUI



[http://sccn.ucsd.edu/**wiki/BCILAB**](http://sccn.ucsd.edu/wiki/BCILAB)

ftp://sccn.ucsd.edu/pub/bcilab

BCILAB Sample Script

- Benchmarking 5 state-of-the-art methods on a 136-subject data set (on a cluster):

```
epoch = [-0.2 0.8];
wnds = [0.25 0.3;0.3 0.35;0.35 0.4; 0.4 0.45;0.45 0.5;0.5 0.55;0.55 0.6];

apps.wmeans_lda = {'Windowmeans' 'SignalProcessing',{'IIRFilter',{[0.1 0.5],'highpass'}, ...
    'EpochExtraction',epoch,'SpectralSelection',[0.1 15]},'Prediction',{'FeatureExtraction',{'wnds',wnds}}};
apps.wmeans_vblogreg = {'Windowmeans' 'SignalProcessing',{'IIRFilter',{[0.1 0.5],'highpass'}, ...
    'EpochExtraction',epoch,'SpectralSelection',[0.1 15]},'Prediction',{'FeatureExtraction',{'wnds',wnds}, ...
    'MachineLearning',{'Learner',{'logreg',[],'variant','vb-iter'}}}};
apps.dalfine = {'DALERP','SignalProcessing',{'EpochExtraction',epoch}, ...
    'Prediction',{'MachineLearning',{'Learner',{'dal','lambdas',2.^(10:-0.125:1),'solver','cg'}}}};
apps.raw_glc = {'DataflowSimplified' 'SignalProcessing',{'IIRFilter',{[0.1 0.5],'highpass'}, ...
    'EpochExtraction',epoch,'SpectralSelection',[0.1 15]}, ...
    'Prediction',{'MachineLearning',{'learner',{'dal',2.^(12:-0.125:1),'regularizer','glc', 'shape',[256 NaN]}}}};
apps.wavelet_glc = {'DataflowSimplified' 'SignalProcessing',{'IIRFilter',{[0.1 0.5],'highpass'}, ...
    'EpochExtraction',epoch,'SpectralSelection',[0.1 15],'wavelet','on'}, ...
    'Prediction',{'MachineLearning',{'learner',{'dal',2.^(12:-0.125:1),'regularizer','glc', 'shape',[256 NaN]}}}};

results = bci_batchtrain('Data','/data:/grainne/ERN/*.vhdr','Approaches',apps, ...
    'TargetMarkers',{{'S101','102'},{'S201','202'}});
```

Toolbox Organization

Framework

- GUI / Scripting Interfaces
- **Approach Definition**
- **Online Execution**
- **Offline Evaluation**
- **Visualization**

Plugins

Signal Processing
- ICA
- SSA
- FIR
- IIR
- FFT
- …

Machine Learning
- LDA
- QDA
- DAL
- GMM
- SVM
- …

BCI Paradigms
- CSP
- Spec-CSP
- ERP
- RSSD
- …

Devices
- TCP
- OSC
- BCI2000
- …

Infrastructure

- GUI generation
- cluster computing
- disk caching
- helper functions
- environment services

Dependencies

- CVX
- BNT
- EEGLAB
- GUI utils
- LIBSVM
- GLMNET
- …
- Driver I/O

3 A Close Look at Components

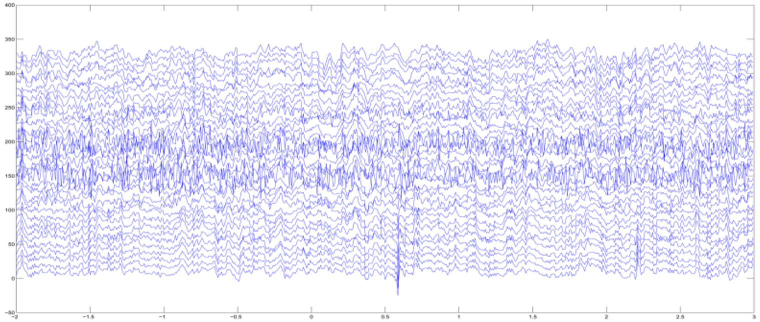
Plugins

- *Signal Processing*
  - ICA
  - SSA
  - FIR
  - IIR
  - FFT
  - …
- *Machine Learning*
  - LDA
  - QDA
  - DAL
  - GMM
  - SVM
  - …
- *BCI Paradigms*
  - CSP
  - Spec-CSP
  - ERP
  - RSSD
  - …
- *Devices*
  - TCP
  - OSC
  - BCI2000
  - …

Component 1: Predictive Mapping

Central Predictive Mapping

- A BCI (with limited memory of the past) can be viewed as a mathematical function *f*:

y = f(**X**); **X**=  y= "subj. excited" (+1)
"subj. not excited" (-1)

- The functional form is arbitrary, for example

$$y = \mathrm{sign}(\mathrm{var}(\boldsymbol{W}\boldsymbol{X}) + b)$$

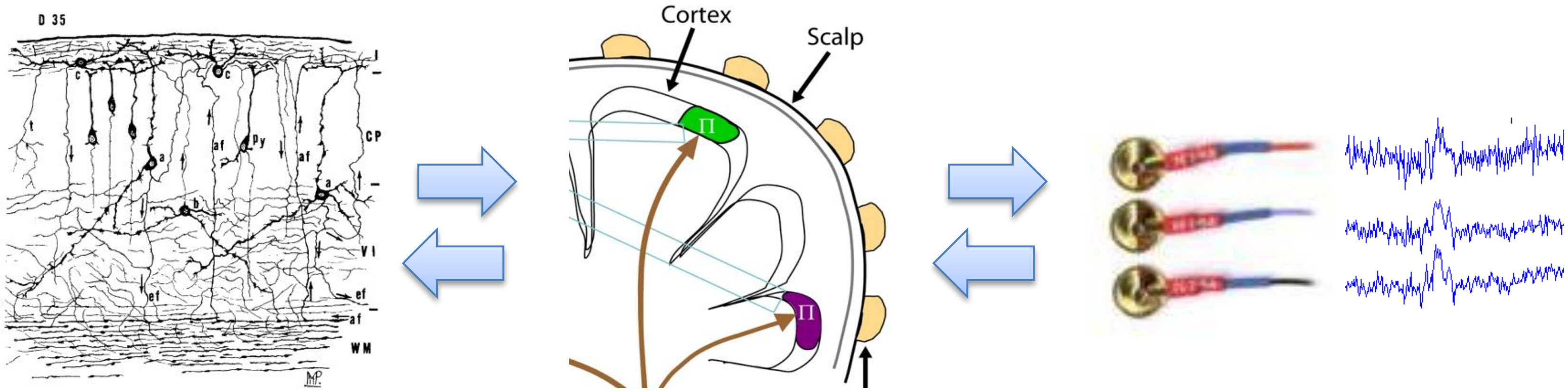
- The mapping involves free parameters, here **W** and b, and data from a *sliding window* **X**

Choice of a Functional Form

- Reflects the relationship between observation (data segment **X**) and desired output (cognitive state parameter y)

Choice of a Functional Form

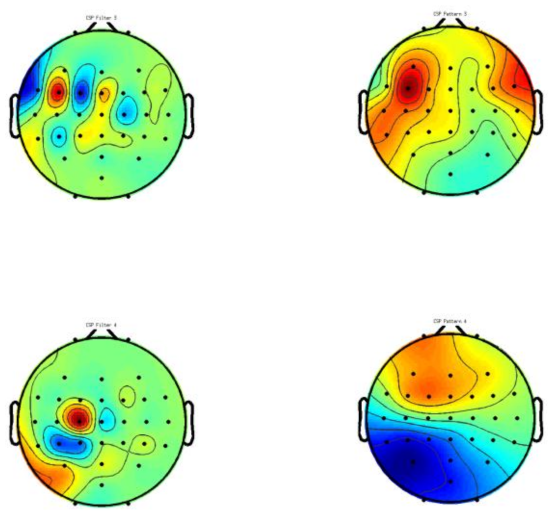
- Reflects the relationship between observation (data segment **X**) and desired output (cognitive state parameter y)
- Based on some assumed generative mechanism (forward model) – or ad hoc



- Remember: Functional form is the inverse mapping!

Key Ingredient: Spatial Filter

- Linear inverse of volume conduction effect between sources **S** and channels **X**
  $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}$ (forward)
  $\boldsymbol{S} = \boldsymbol{W}\boldsymbol{X}$ (inverse)

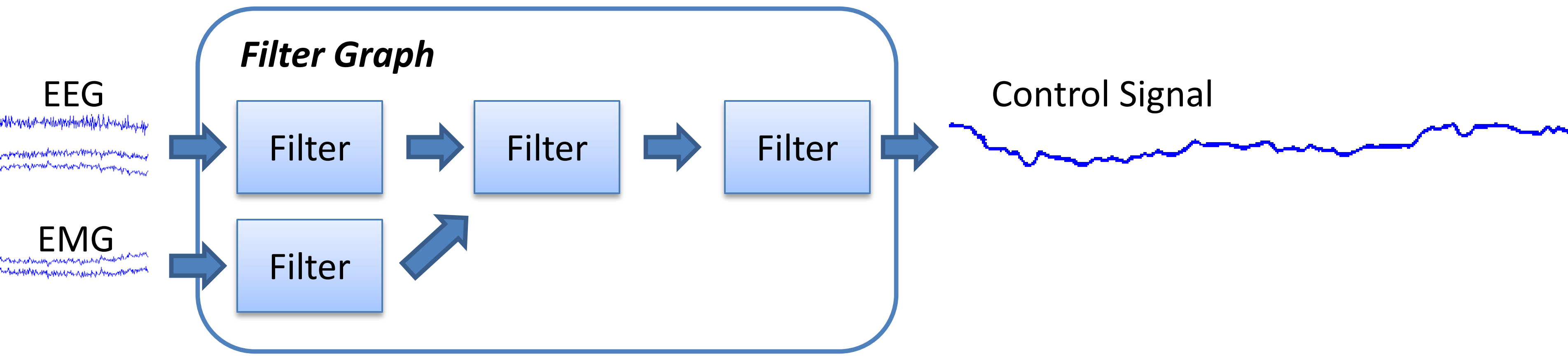


W **A=W^{-1}**

Component 2: Signal Processing

Role of Signal Processing

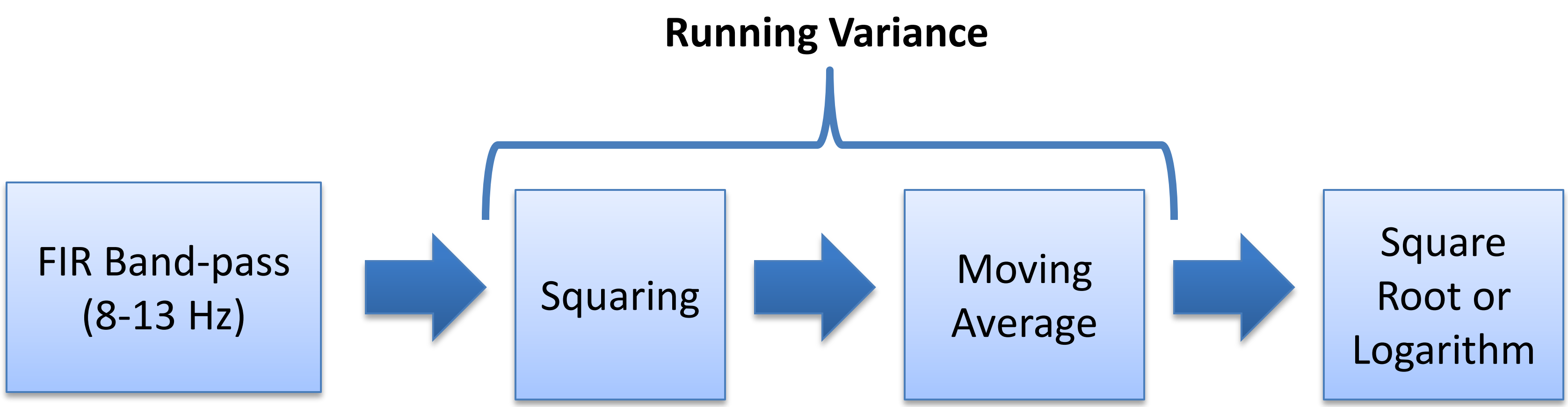
- BCILAB allows to implemented BCIs using a network of digital signal processing blocks ("filters")



- Relevant filter classes: *Spatial Filters, Temporal Filters, Spectral Filters, Spatio-Temporal Filters, Domain Transforms* (e.g. DFT)

Role of Signal Processing

- **Concrete Toy Example:** Feed the amplitude of a brain idle oscillation (e.g. 10 Hz alpha associated with relaxation) from one EEG channel back to the user/subject



$$y_i(n) = \sum_{k=0}^{m} b_k x_i\,(n-t)$$

$$y_i(n) = x_i\,(n)^2$$

$$y_i(n) = \frac{1}{m-1}\sum_{k=0}^{m} x_i\,(n-k)$$

$$y_i(n) = \log x_i\,(n)$$

Filter Components In Practice

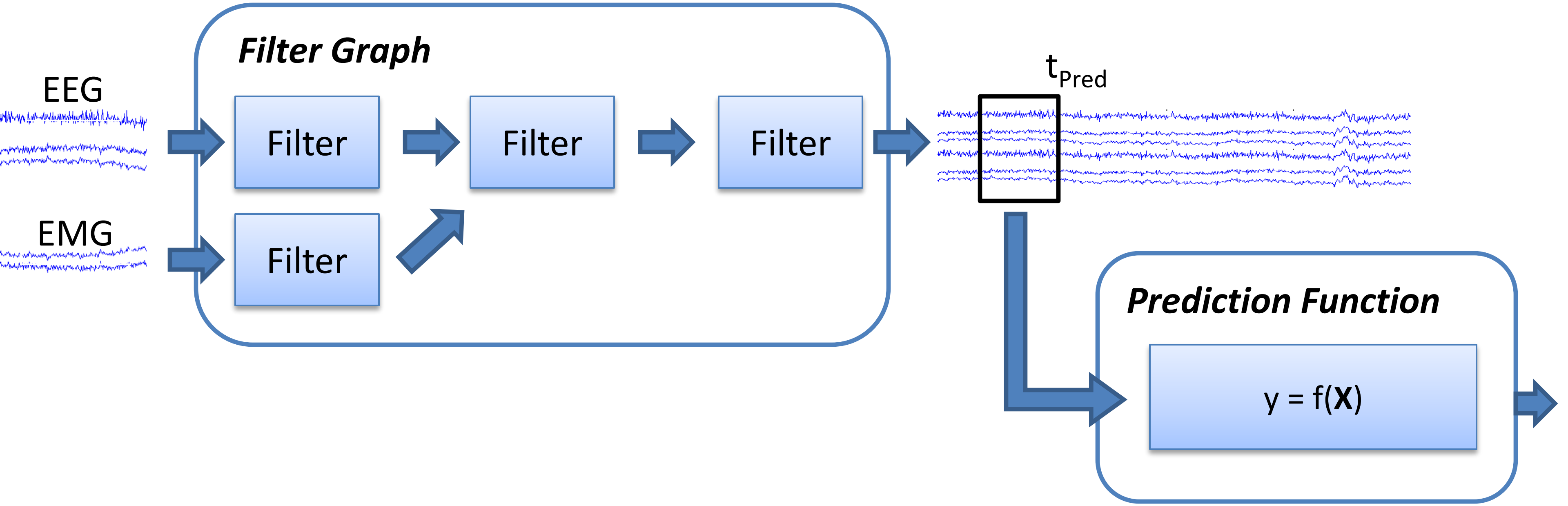
- Filters can operate on continuous signals…

- … or on segmented (“epoched”) signals:


```
EEG = flt_selchans(EEG,{‘C3’,’C4’,’Cz’})
[EEG,State] = flt_resample(EEG,200,State)
```

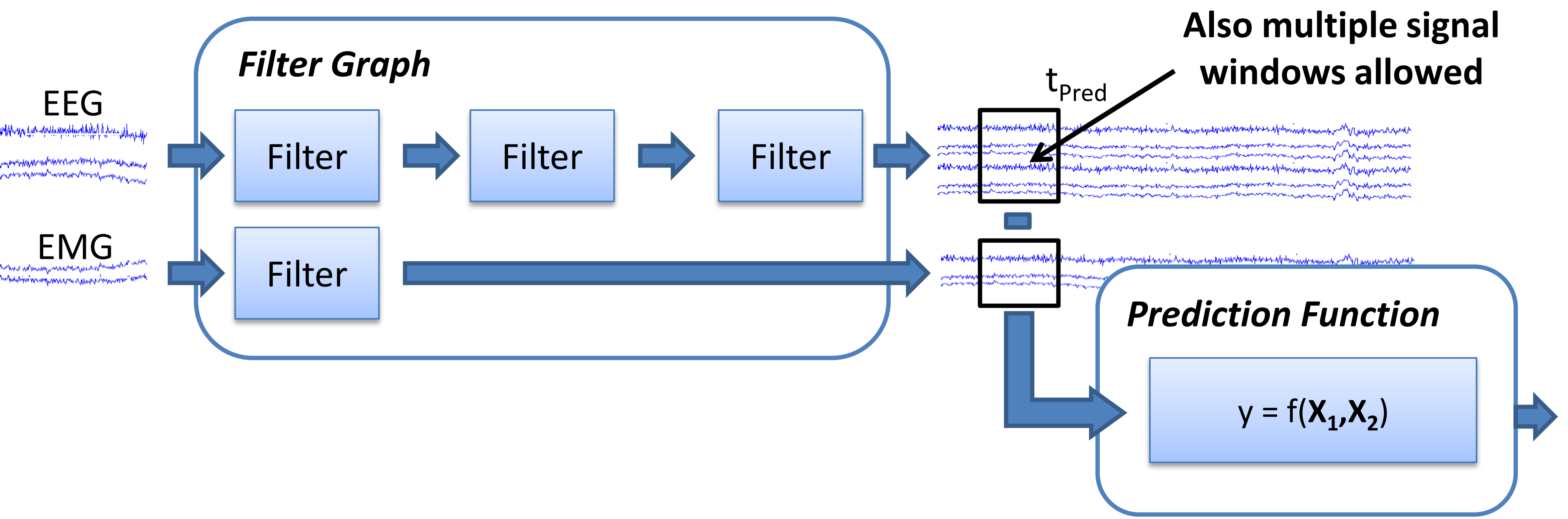
Combined Online Processing

- Both frameworks are complementary, rather than contradictory, and are in practice often used *in combination*, e.g. to minimize computational costs



Combined Online Processing

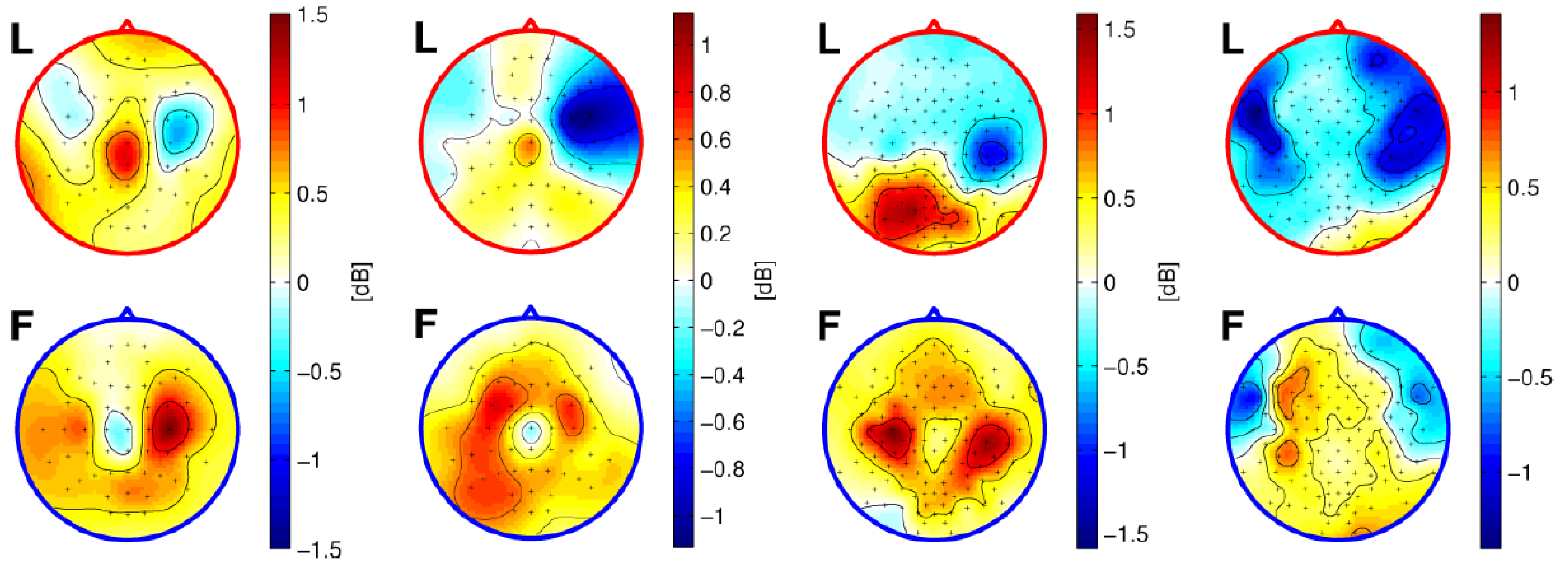
- Both frameworks are complementary, rather than contradictory, and are in practice often used *in combination*, e.g. to minimize computational costs



Component 3: Machine Learning

The Problem of Unknown Parameters

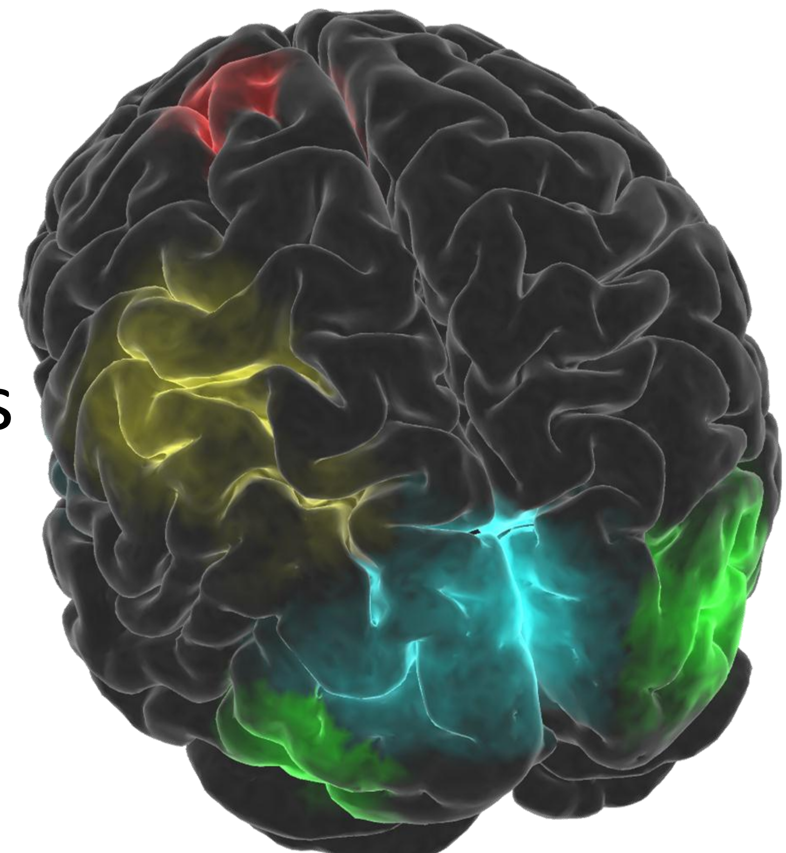
- Processing depends on unknown parameters (person-specific, task-specific, otherwise variable) – e.g., per-sensor weights as below:



Blankertz et al. 2007

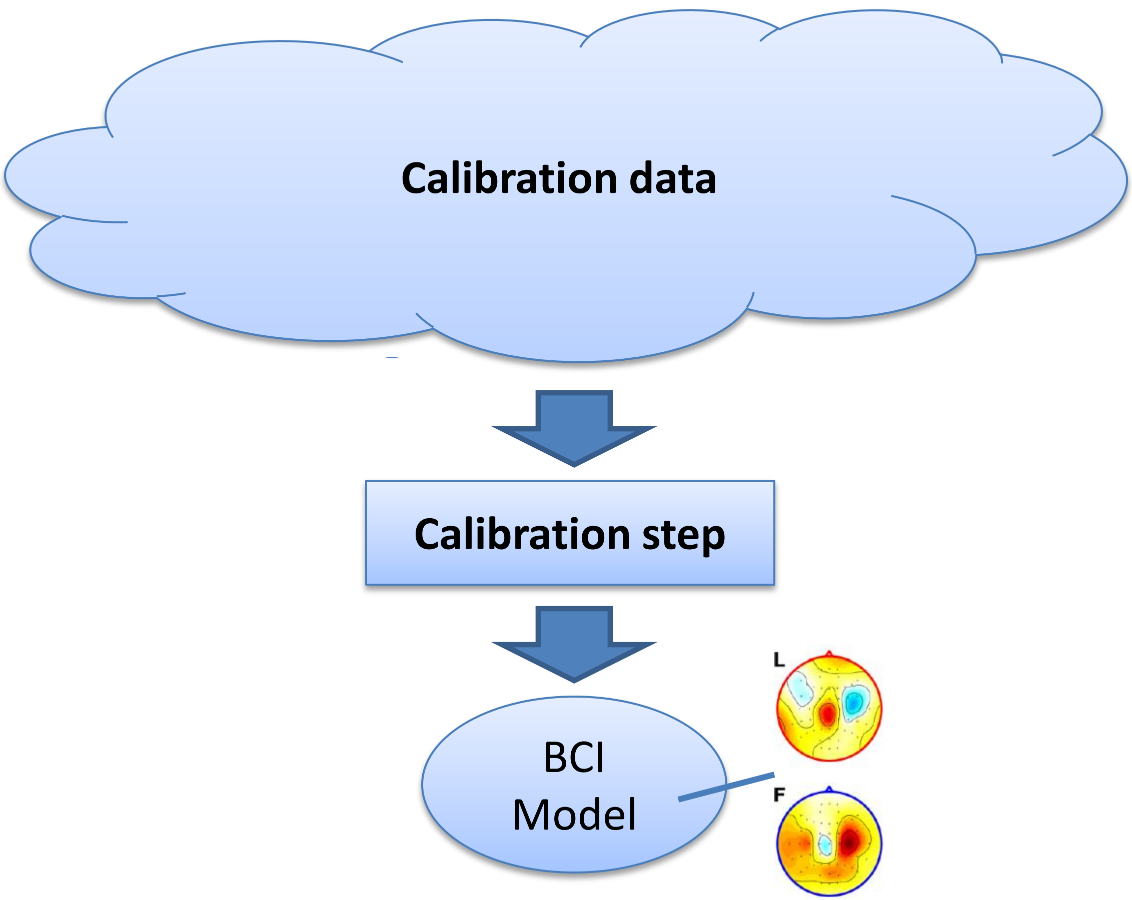
Reasons for Parameter Uncertainty

- Folding of cortex differs between any two persons
- Relevant functional map differs across individuals
- Sensor locations differ across recording sessions
- Brain dynamics are non-stationary at all time scales



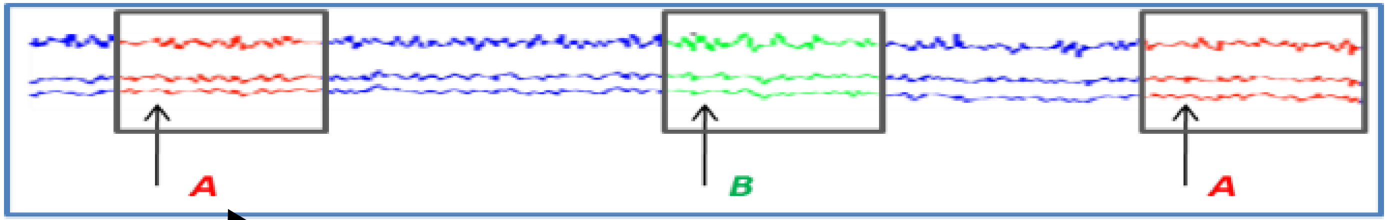
Solution: Calibration

- *Calibration / training data* can be used to estimate parameters, during a separate *calibration step*



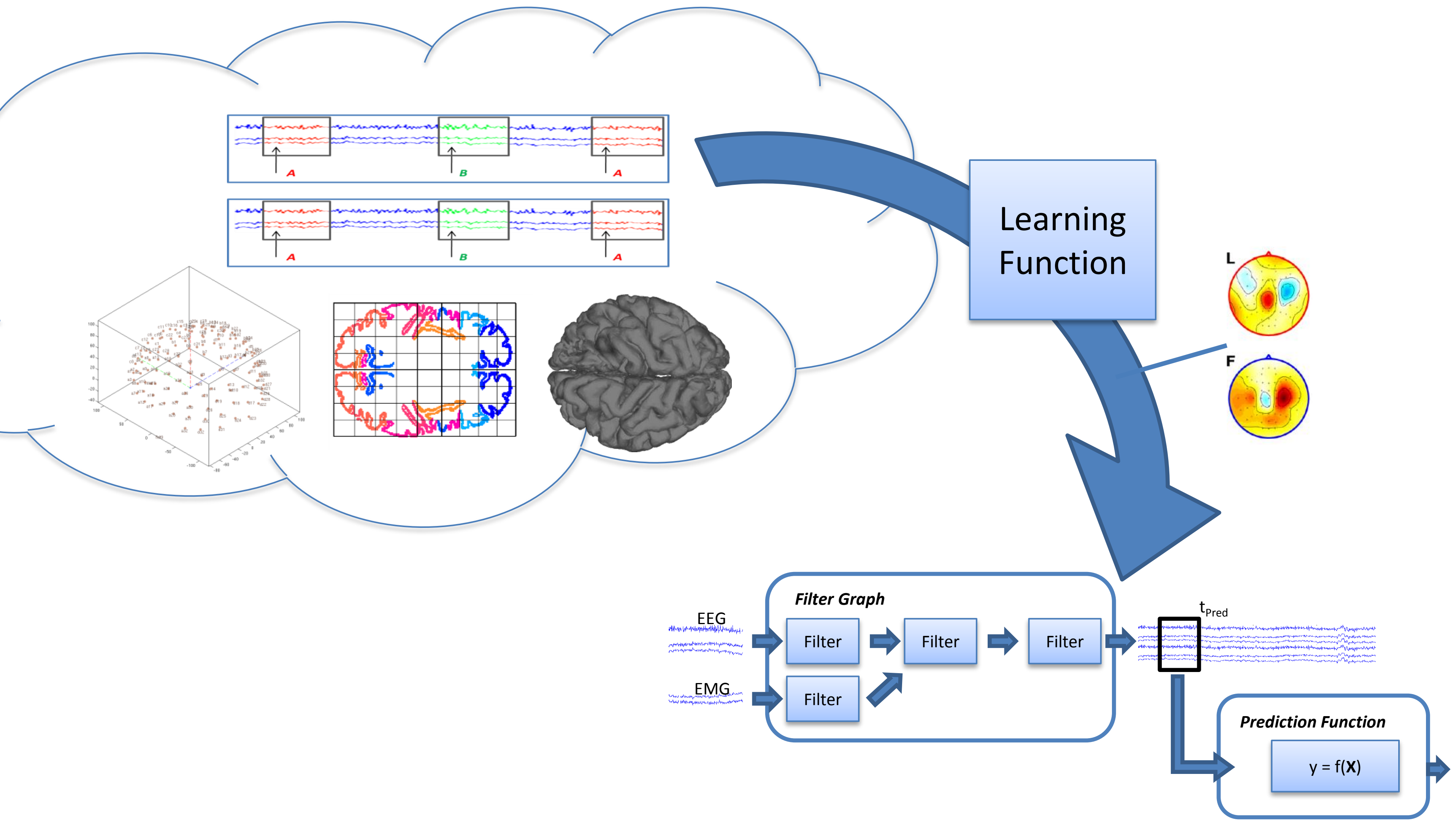
Calibration Data

- Many possible kinds of data could be used
- Best known type of calibration data: ***example data***, i.e. examples of EEG of a person being excited, not excited, etc.
- Collected in a special *calibration recording* (before actual online use of the BCI)



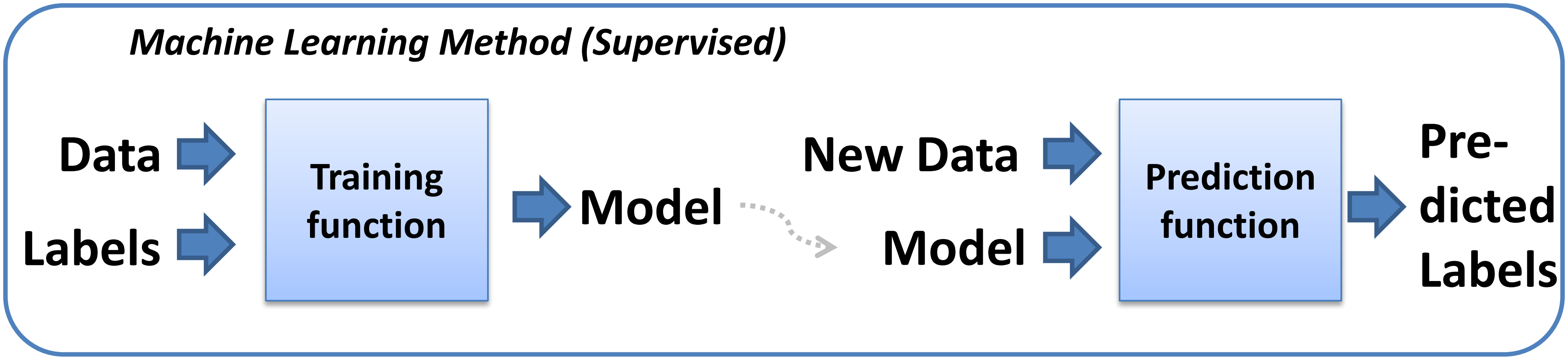
"target markers" **in BCILAB**

Big-Picture Information Flow



Machine Learning Framework

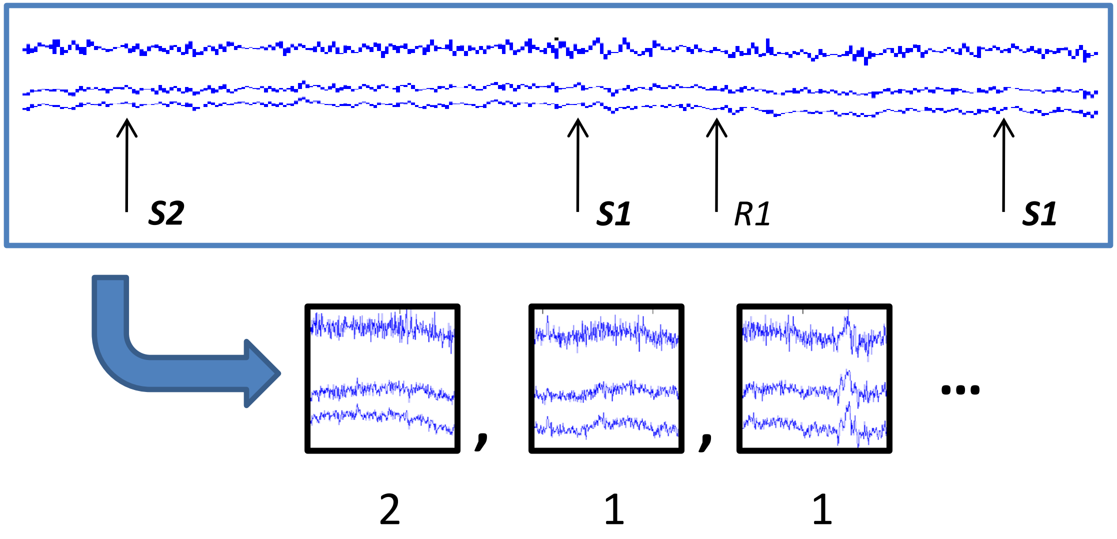
- Large field with 100s of algorithms (LDA, SVM, GMM, ANNs, logistic regression, …)
- Most methods conform to a common framework of a *training function* and a *prediction function*




```
M = ml_trainsvm(X,y, <extra parameters>)
y_pred = ml_predictsvm(X_new,M)
```

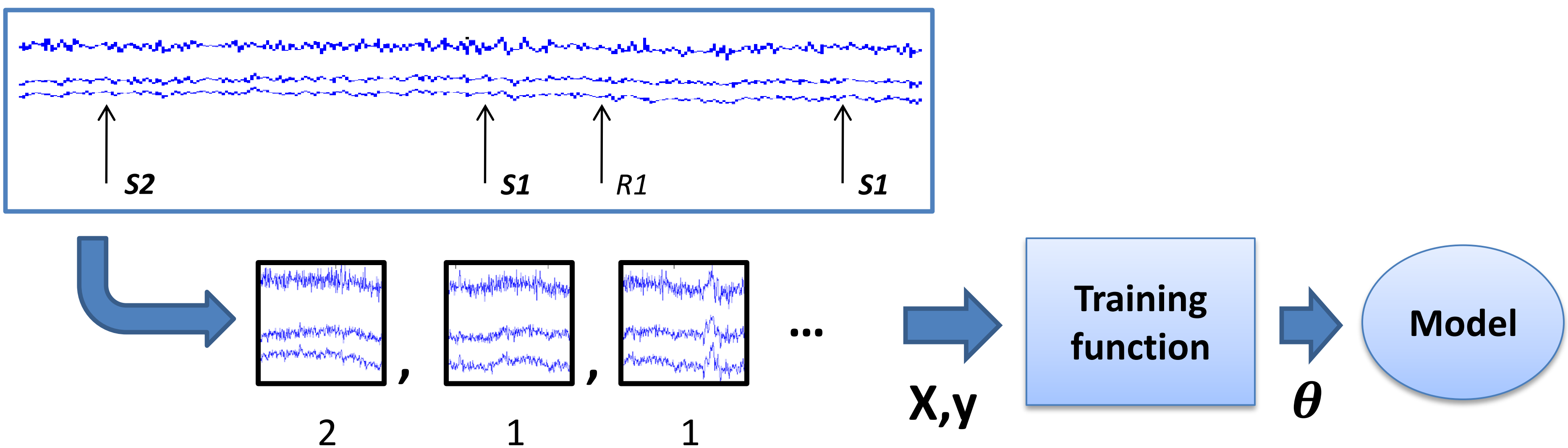
Machine Learning In Practice

- Often, one trial segment (sample) is extracted for every target marker in the calibration recording and is used as *training exemplar* $\boldsymbol{X}_k$
- Its associated label y_k can be deduced from the target marker



Machine Learning In Practice

- Often, one trial segment (sample) is extracted for every target marker in the calibration recording and is used as *training exemplar* $\boldsymbol{X}_k$
- Its associated label y_k can be deduced from the target marker

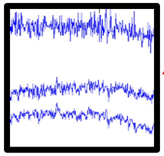


Component 4: Feature Extraction

Feature Extraction

- **Caveat:** Off-the-shelf machine learning methods often do *not work very well* when applied to raw signal segments of the calibration recording
  - too high-dimensional (too many parameters to fit)
  - too complex structure to be captured (too much modeling freedom, requires domain-specific assumptions)

1000s of degrees of freedom!



Feature Extraction

- **Typical Solution**: Introduce additional mapping (called *"feature extraction")* from raw signal segments onto feature vectors which extracts the *key features* of a raw observation
  - output is usually of lower dimensionality
  - hopefully statistically "better" distributed (easier to handle for machine learning)

Concrete Example Task

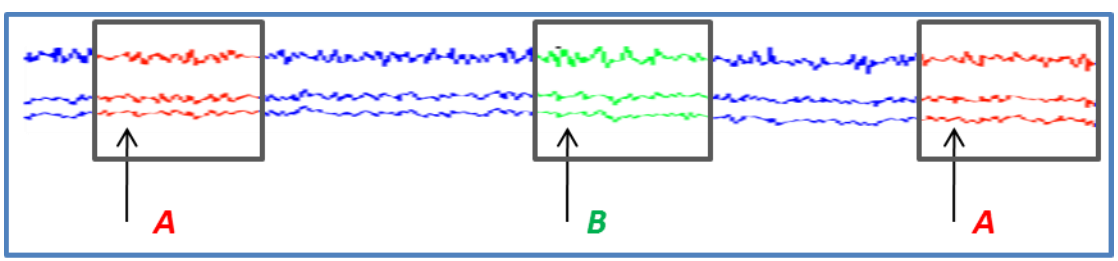
- **Flanker Task:** The experiment consists of a sequence of ca. 330 trials with inter-trial interval of 2s +/- 1.5s
- At the beginning of each trial, an arrow is presented centrally (pointing either left or right)
- The arrow is flanked by congruent or incongruent “flanker” arrows (preceding the center by a few ms):

← ← → ← ←

- The **subject is asked to press the left or right button, according to the central arrow direction, and makes frequent errors (ca. 25%)**

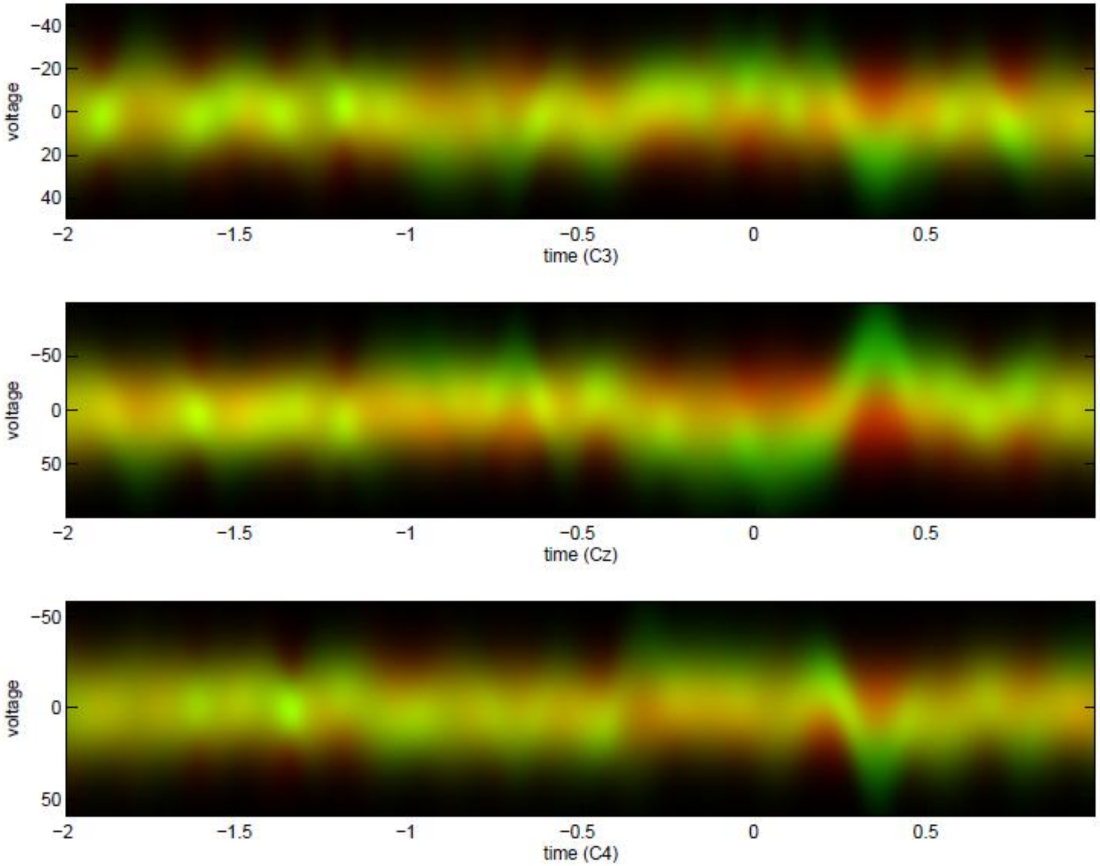
Approach

- Calibration recording is band-pass filtered between 0.5Hz and 15Hz
  - 0.5Hz lower edge removes drifts
  - 15Hz upper edge leaves enough room for sharp ERP features
- Epochs are extracted for each trial and label is set to A for incorrect trials and B for corrects



Actual Data

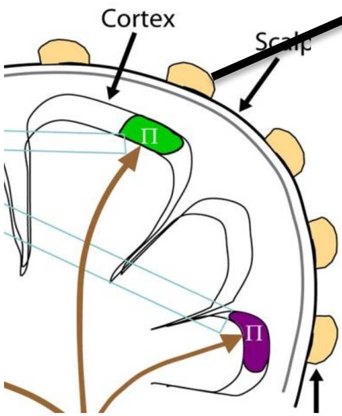
- Time courses for all trials super-imposed (color-coded by class) – but here different task

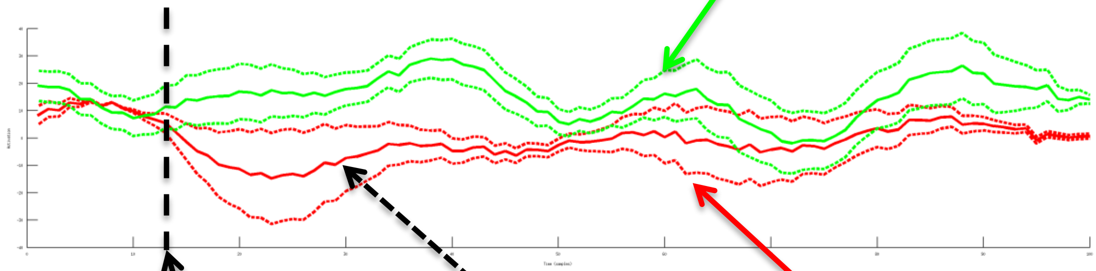


Swartz Center for Computational Neuroscience

Extracted Epochs

Channel time courses under Condition B



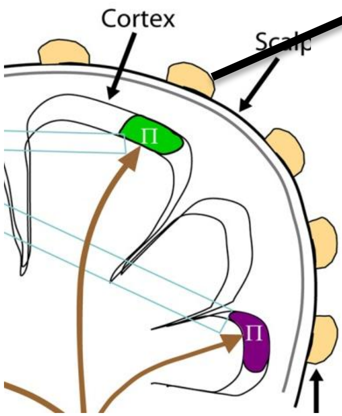


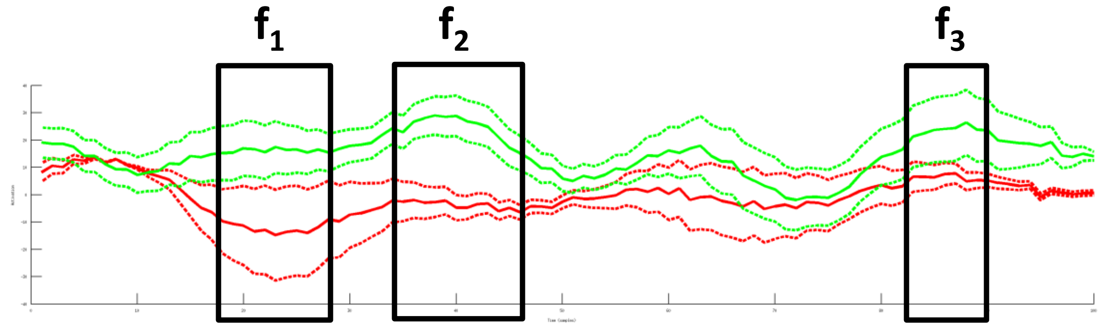
Channel time courses under Condition A

Response (A or B)

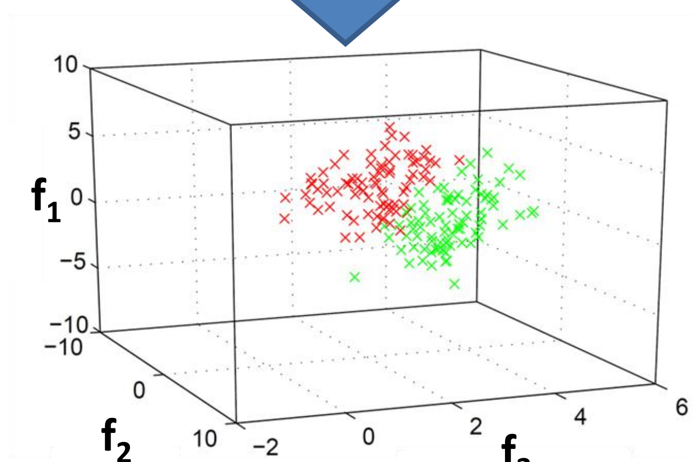
Three sample trials (out of 100) shown: mean, -1 std. dev, +1 std. dev

Extracting Linear Features



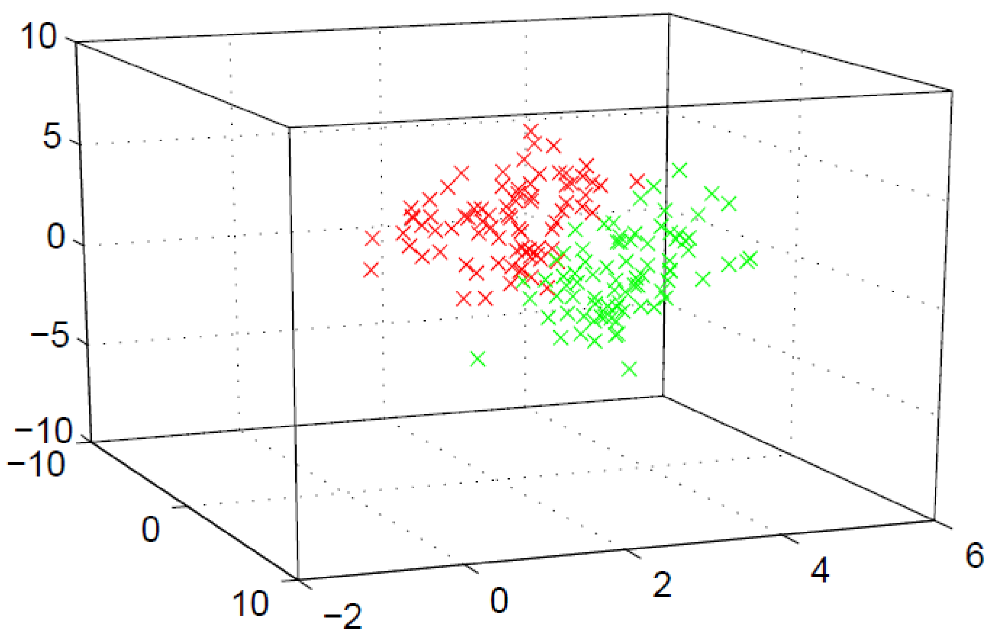


For each trial segment, calculate signal mean in 3 time sub-windows (→ 3-dim feature vector)



Resulting Feature Space

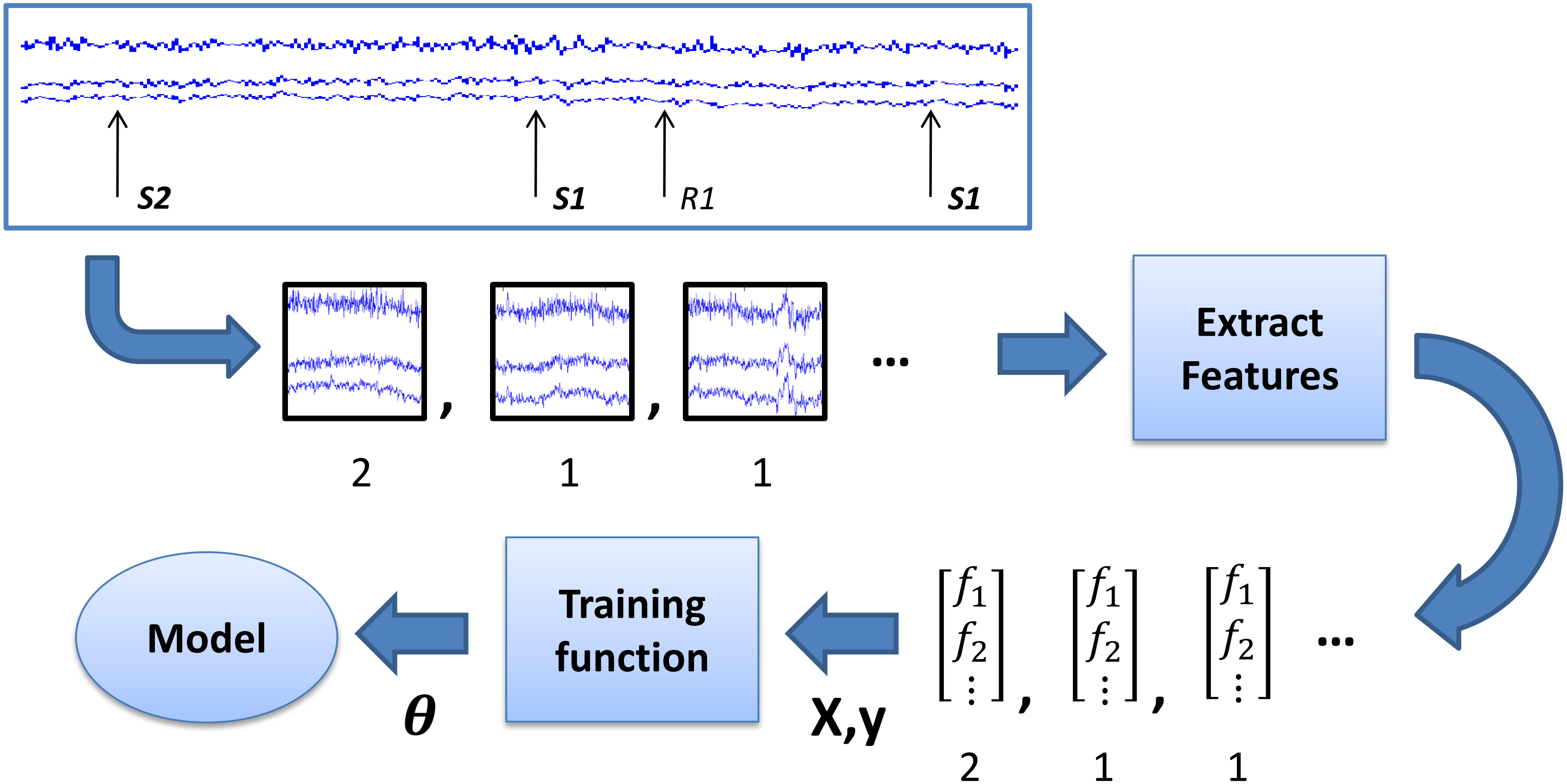
- Plotting the 3-element feature vectors for all error trials in red, and non-error trials in green, we obtain two distributions in a 3d space:



Note that across all channels this space has in fact 3 x #channels dimensions!

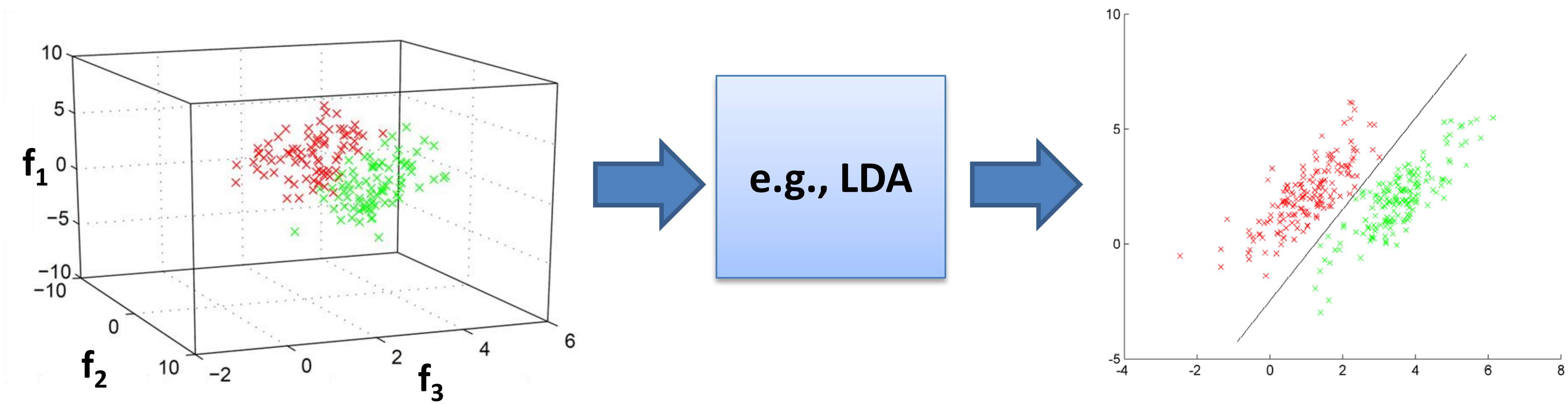
ML with Feature Extraction

- Including the feature extraction, the analysis process is as follows:



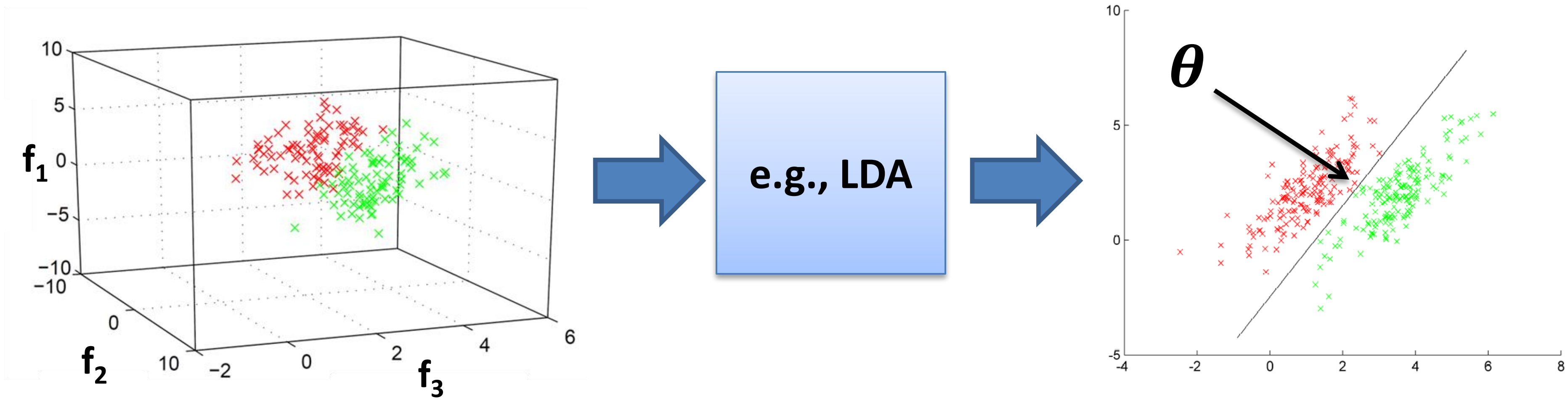
Machine Learning Continued

- The feature vectors are passed on to a machine learning function (e.g., Linear Discriminant Analysis)



Machine Learning Continued

- The feature vectors are passed on to a machine learning function (e.g., Linear Discriminant Analysis)
- … which determines a parametric predictive mapping

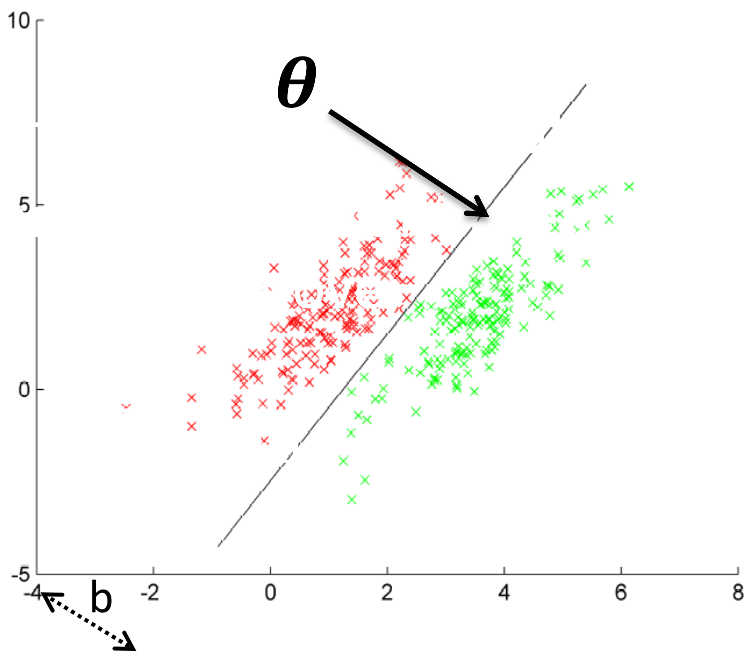


Simple 2-class LDA In a Nutshell

- Given feature vectors $\boldsymbol{x}_k$ (in vector form) in $\mathcal{C}_1$ and $\mathcal{C}_2$,

$$\boldsymbol{\mu}_i = \frac{1}{|\mathcal{C}_i|}\sum_{k\in\mathcal{C}_i}\boldsymbol{x}_k\,, \qquad \boldsymbol{\Sigma}_i = \sum_{k\in\mathcal{C}_i}(\boldsymbol{x}_k-\boldsymbol{\mu}_i)(\boldsymbol{x}_k-\boldsymbol{\mu}_i)^\intercal$$

$$\boldsymbol{\theta} = (\boldsymbol{\Sigma}_1+\boldsymbol{\Sigma}_2)^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1), \qquad \mathrm{b} = \boldsymbol{\theta}^\intercal(\boldsymbol{\mu}_1+\boldsymbol{\mu}_2)/2$$



Resulting Predictive Mapping and Model

- LDA produces parameters of a linear mapping

$$\mathrm{y} = \boldsymbol{\theta x} - b$$

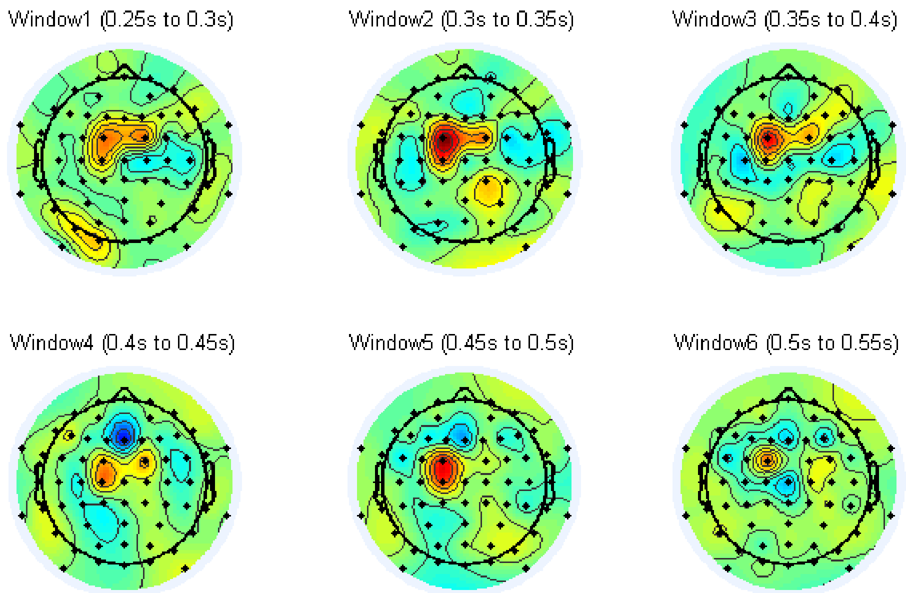
- For classification, the mapping is actually *non-linear*:

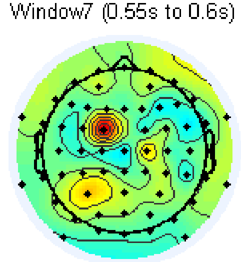
$$\mathrm{y} = \mathrm{sign}(\boldsymbol{\theta x} \; - b)$$

- The learned model with its person-specific parameters here consists of $(\boldsymbol{\theta}, b)$; generally it could include adapted signal-processing parameters, feature-extraction parameters, etc.

Spatial Filters Visualized

- Topographically mapped, the following filters emerge:

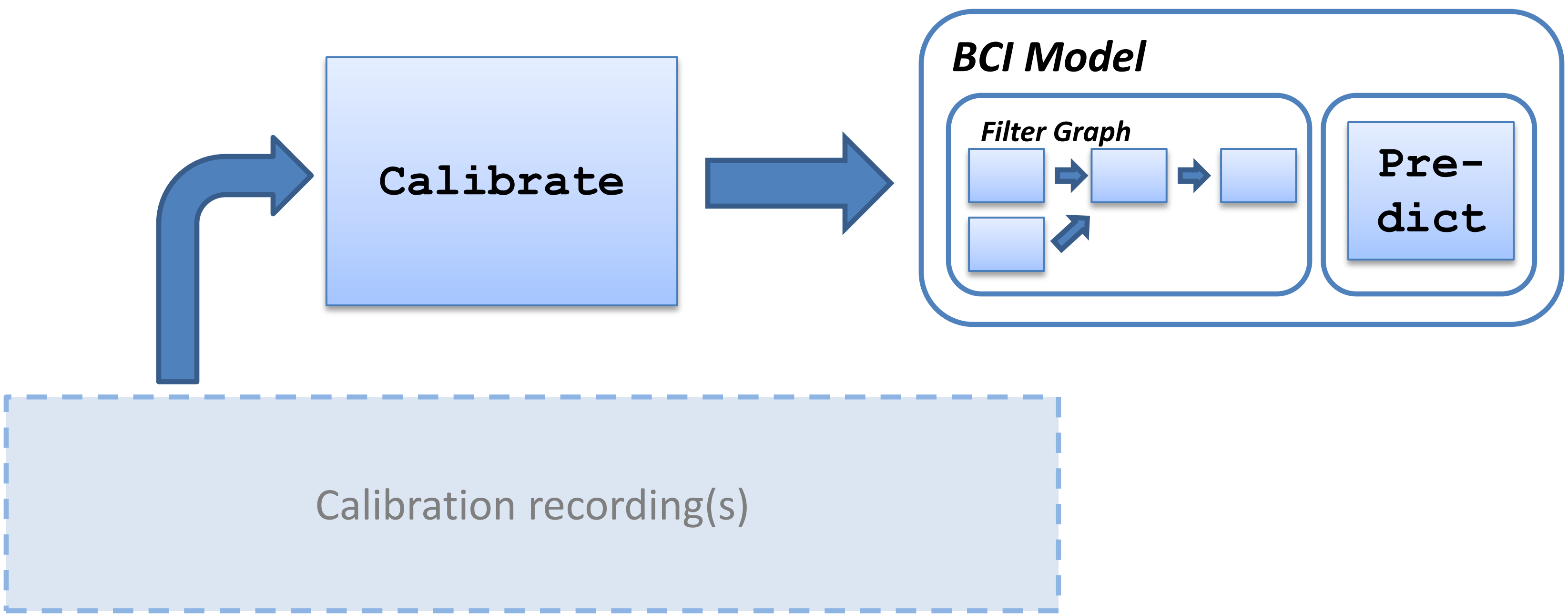




Note: This method (and its close relative using “shrinkage LDA” in particular) yield state-of-the-art Performance on ERPs.

Overall BCI Structure

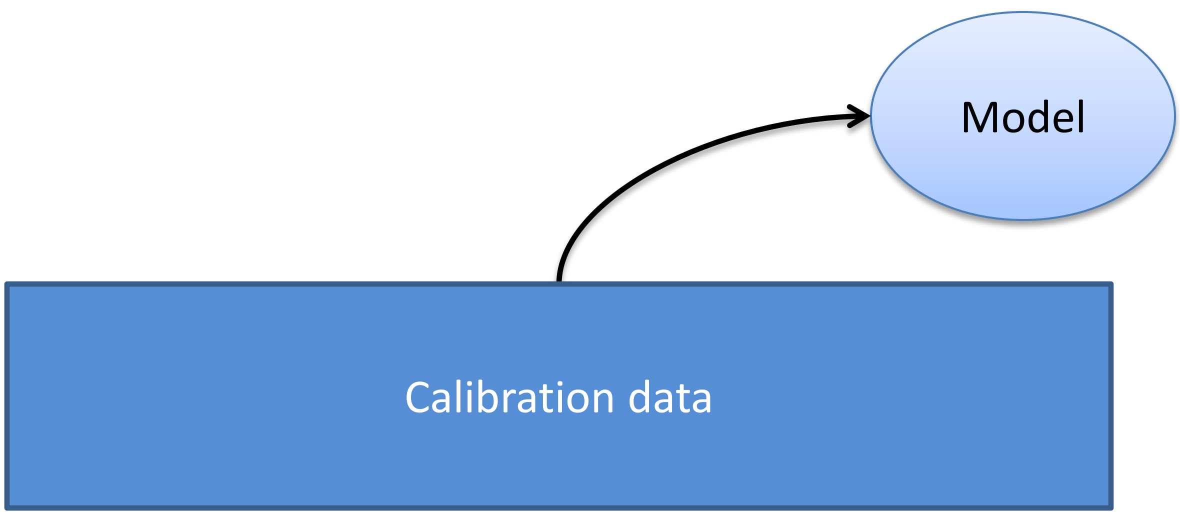
- **BCI paradigms** are BCILAB's way to *encapsulate all parts of a BCI approach into one unit* (e.g., signal processing, feature extraction, machine learning, …)



Component 5: BCI Performance Evaluation

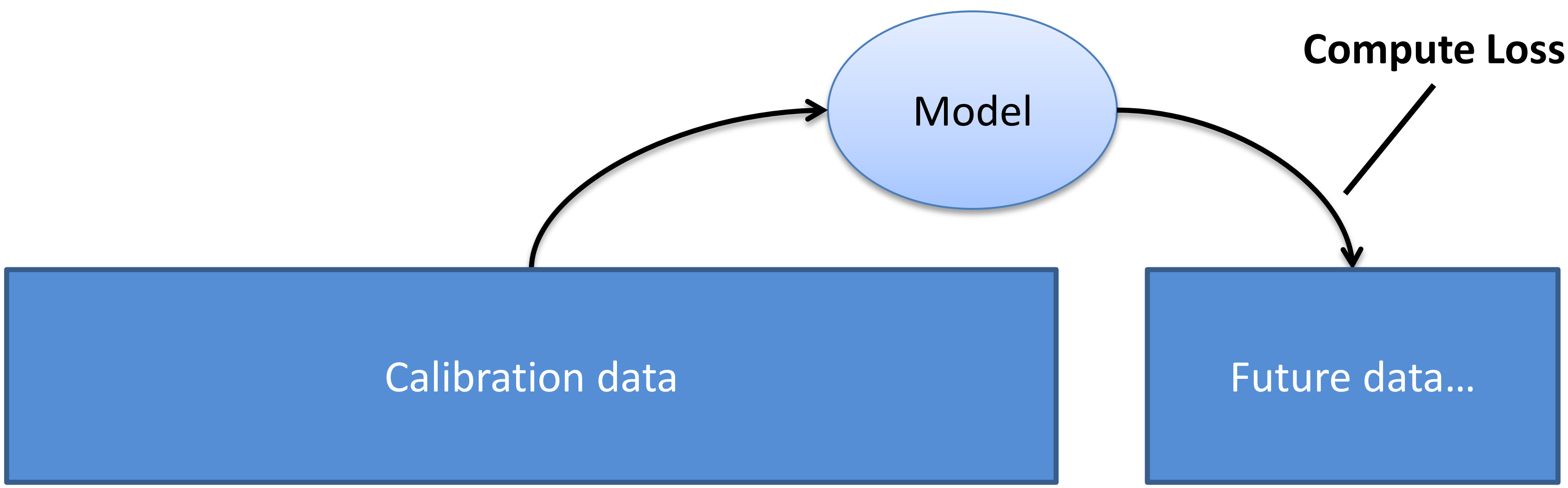
Overall Evaluation Strategies

- **When given calibration data and test data…**
- Estimate model parameters (spatial filters, statistics)



Overall Evaluation Strategies

- **When given calibration data and test data…**
- Estimate model parameters (spatial filters, statistics)
- Apply the model to new data (online / single-trial)
- Measure prediction performance or loss (e.g., mis-classification rate or mean-square error)

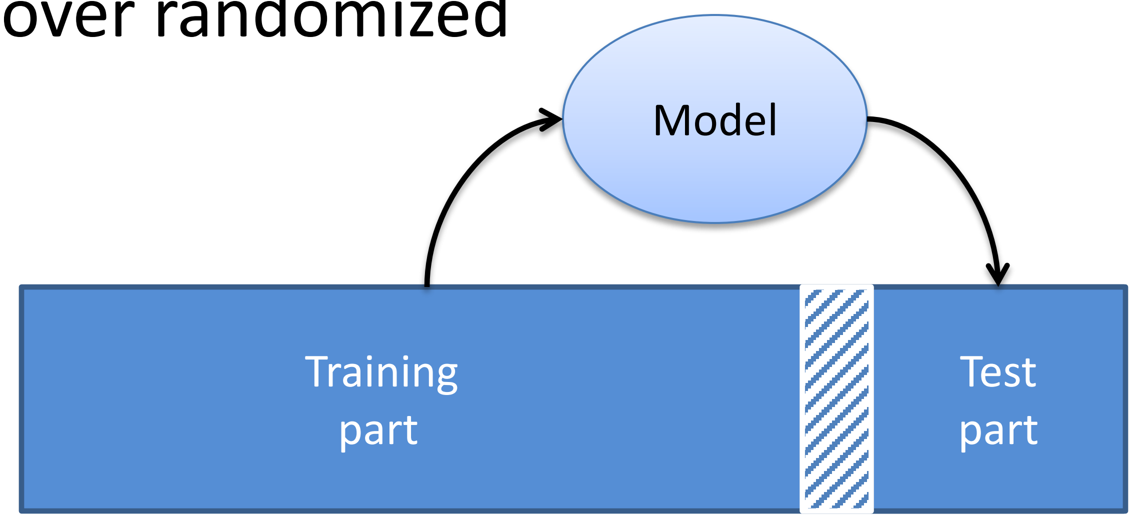


Overall Evaluation Strategies

- Some implemented loss measures (between known "ground-truth" target labels ***t*** and predicted labels ***p***) include *mean-square error*, *mis-classification rate*, *area under ROC curve*, and ca. a dozen others
- **Mean-Square Error:**
  – $L_{MSE}(\boldsymbol{p},\boldsymbol{t}) \; = \frac{1}{N}\sum_k (\boldsymbol{p}_k - \boldsymbol{t}_k)^2$
- **Mis-Classification Rate:**
  – $L_{MCR}(\boldsymbol{p},\boldsymbol{t}) \; = \; \frac{1}{N}\sum_k \begin{cases} 1, \boldsymbol{p}_k \neq \boldsymbol{t}_k \\ 0, \boldsymbol{p}_k = \boldsymbol{t}_k \end{cases}$

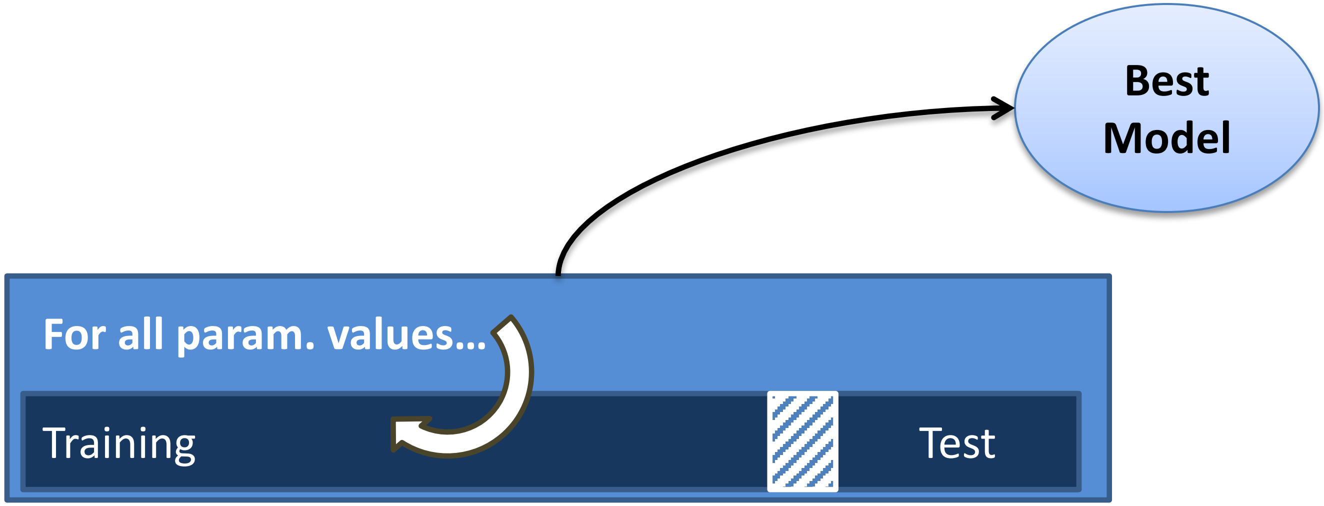
Overall Evaluation Strategies

- **What if there is no second data set?**
- split *one data set* repeatedly into training/test blocks systematically, a.k.a. *cross-validation*
- Each trial is used for testing once
- Time series data: Prefer block-wise cross-validation over randomized



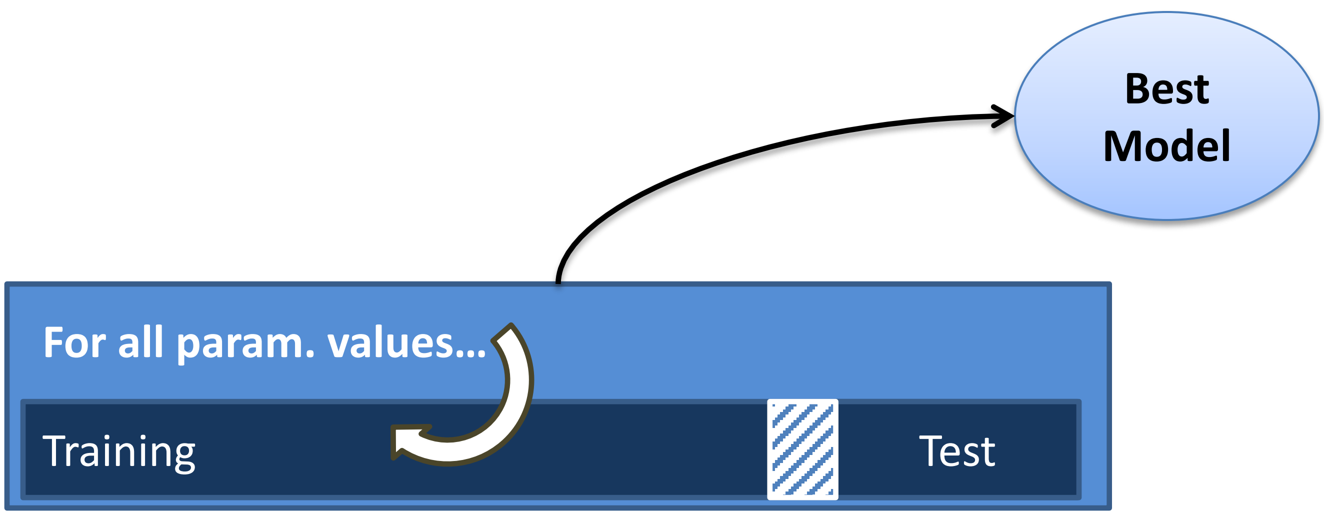
Overall Evaluation Strategies

- **Parameter search** can be done using cross-validation in a grid search (try all values of free parameters)
- Quite general (e.g. can search for best method)



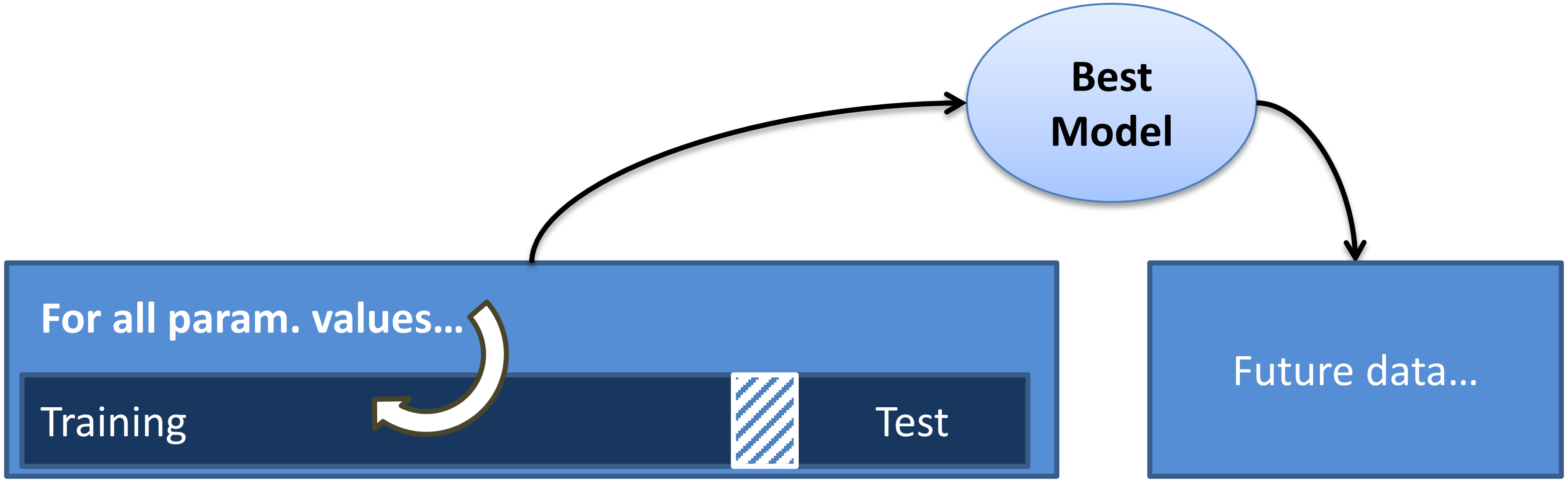
Overall Evaluation Strategies

- **Parameter search** can be done using cross-validation in a grid search (try all values of free parameters)
- Quite general (e.g. can search for best method)
- **However:** Cannot directly report "best performance" estimates (=cherry-picked)



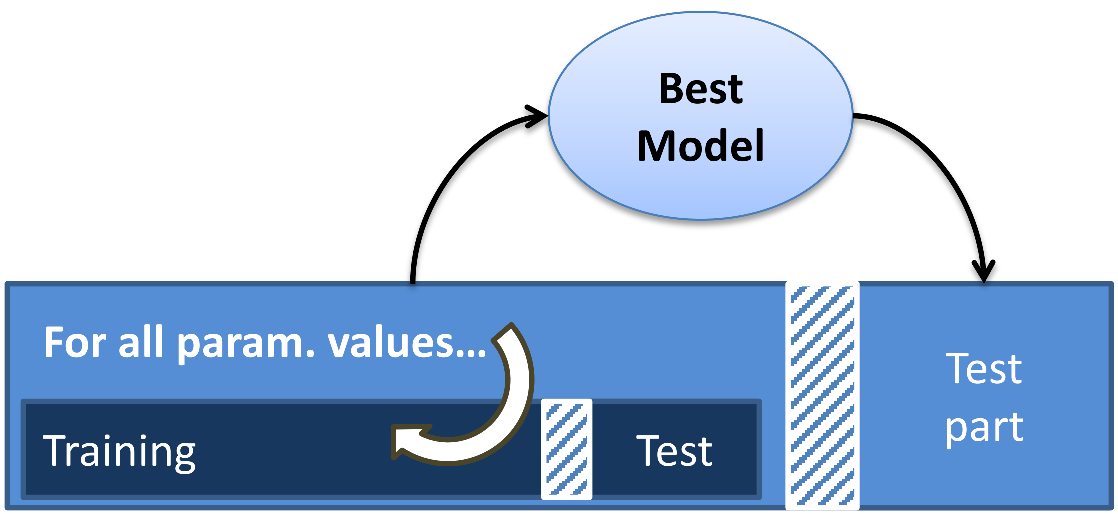
Overall Evaluation Strategies

- **Parameter search** can be done using cross-validation in a grid search (try all values of free parameters)
- Quite general (e.g. can search for best method)
- **However:** Cannot directly report "best performance" estimates (=cherry-picked), except on future data



Overall Evaluation Strategies

- **Parameter search** can be done using cross-validation in a grid search (try all values of free parameters)
- **Alternatively:** Parameter search can be nested *within* an outer cross-validation ("nested cross-validation")

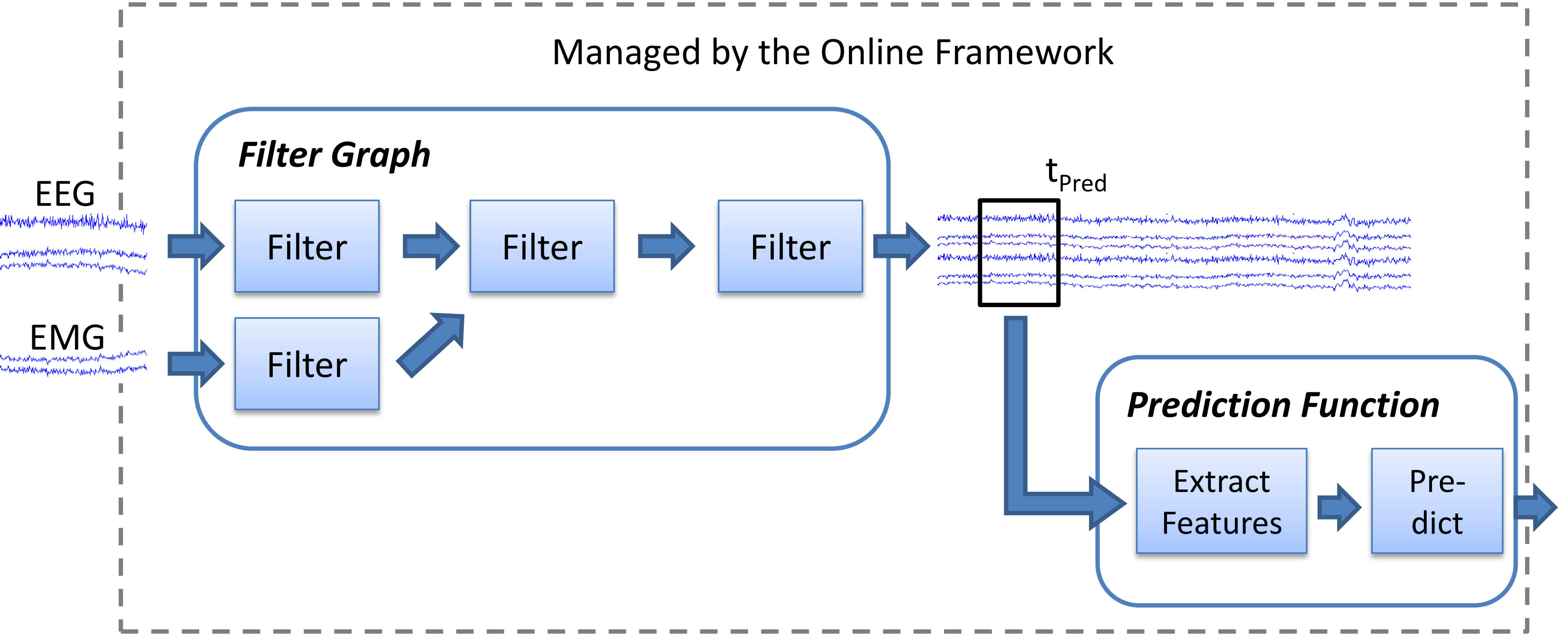


Overall Evaluation Strategies

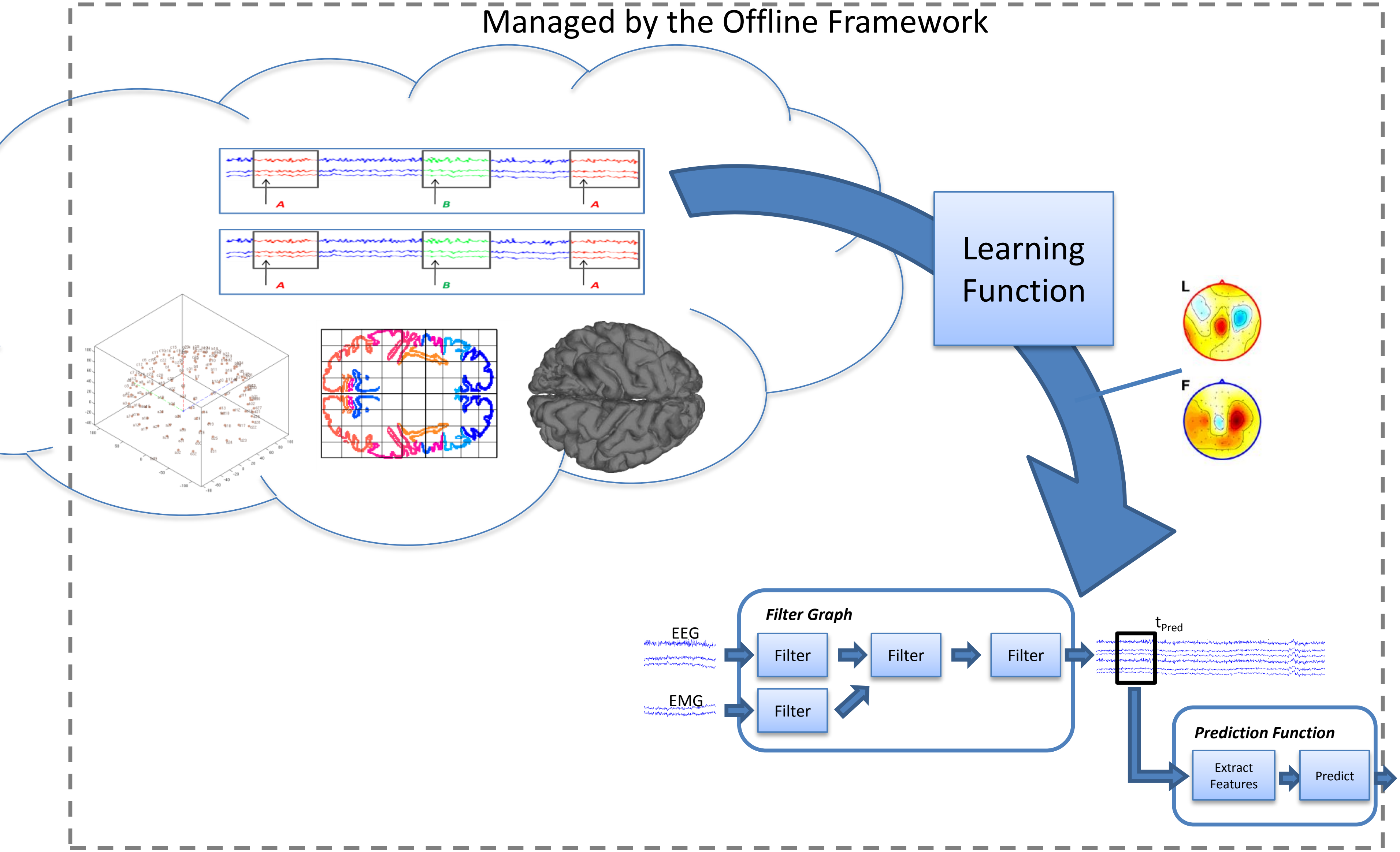
- The same strategies can be applied across a collection of data sets (e.g., multiple sessions or multiple subjects), for example “hold-one-subject-out”
- Cross-validation, grid search, nested cross-validation can be farmed out to a cluster in BCILAB, also to compiled workers (= no MATLAB license bottleneck)

Summary

Scope of the Online Framework

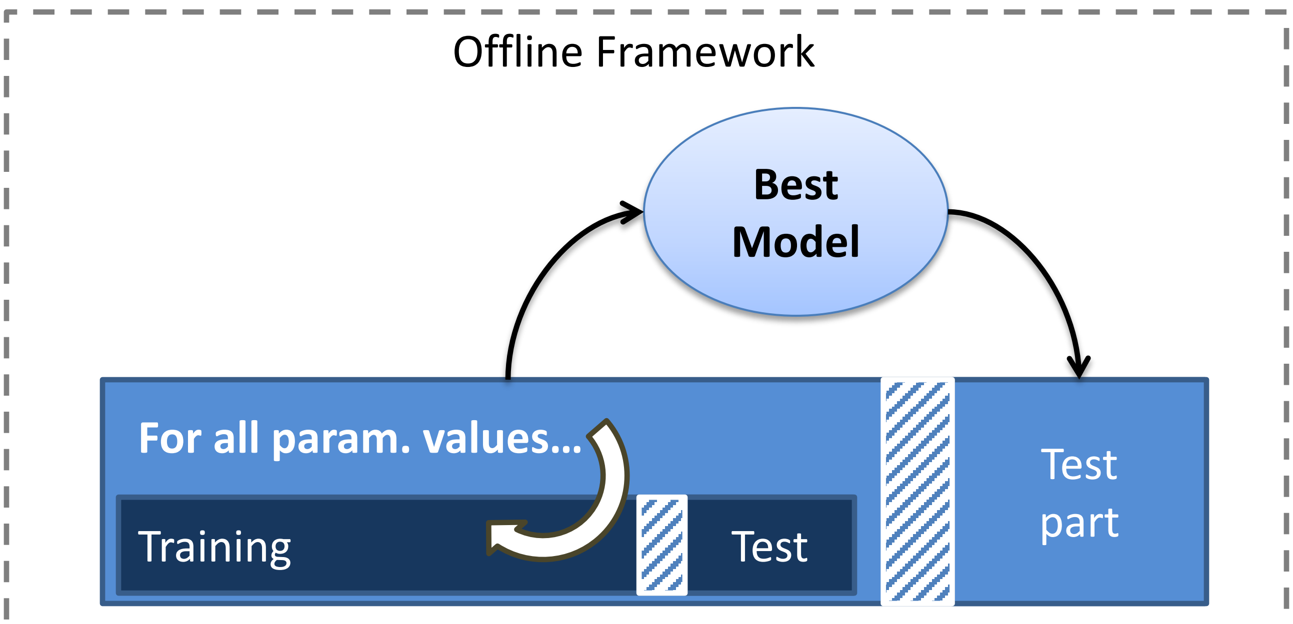


Scope of the Offline Framework

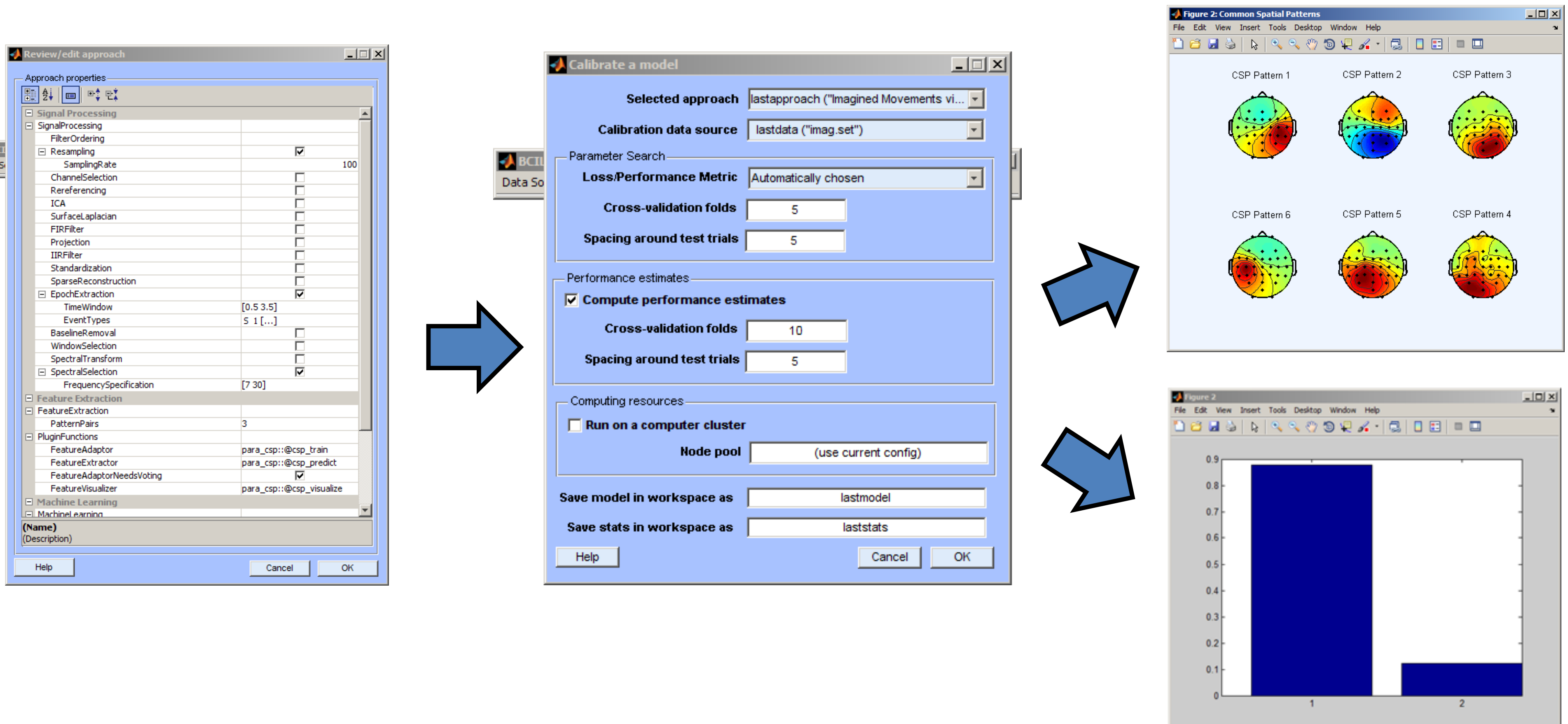


Scope of the Offline Framework

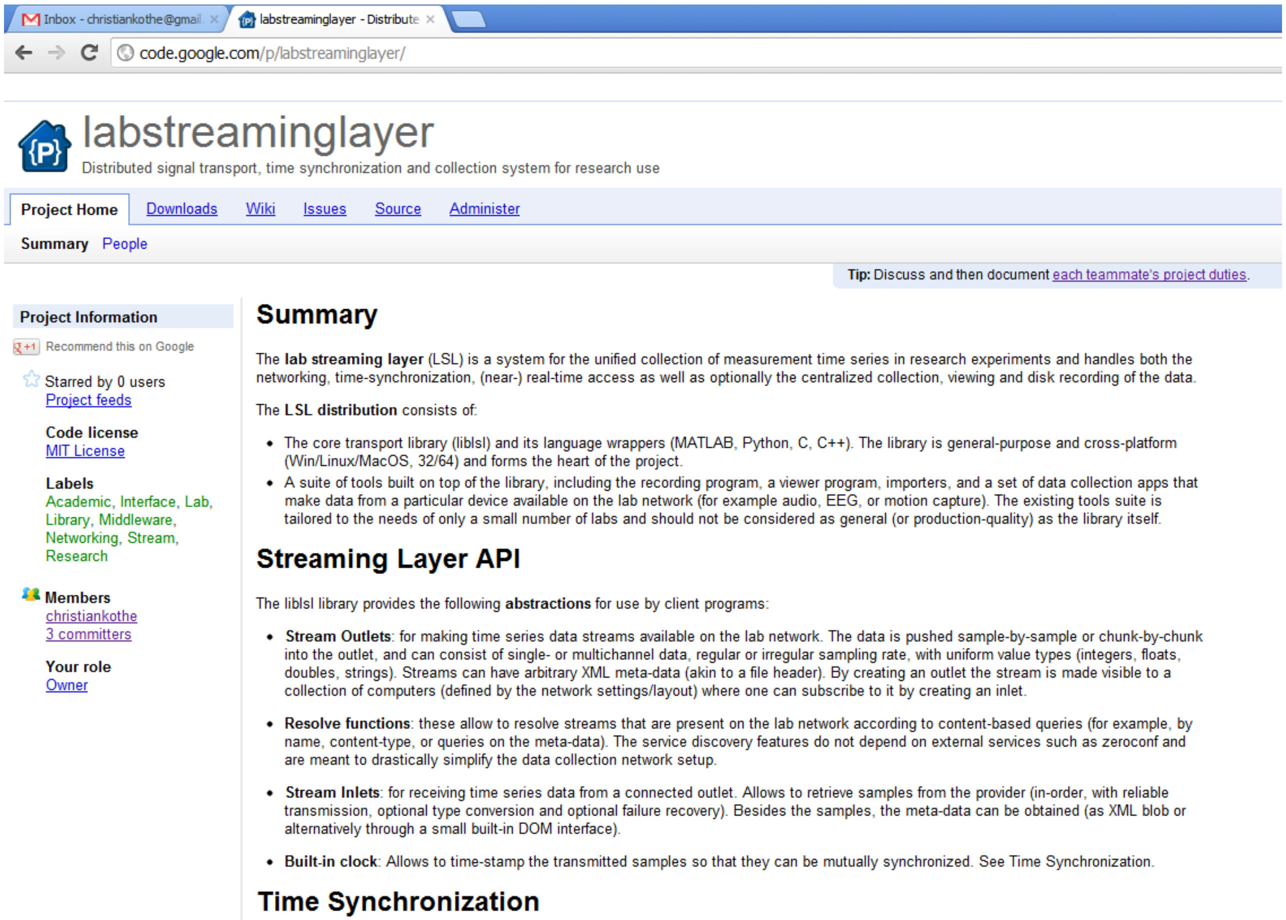
- **Also Covered:** Cross-validation, Grid Search, Nested Cross-Validation



5 GUI and Scripting Tour


Review/edit approach
Approach properties
Signal Processing
SignalProcessing
FilterOrdering
Resampling
SamplingRate
100
ChannelSelection
Rereferencing
ICA
SurfaceLaplacian
FIRFilter
Projection
IIRFilter
Standardization
SparseReconstruction
EpochExtraction
TimeWindow
[0.5 3.5]
EventTypes
S 1 [...]
BaselineRemoval
WindowSelection
SpectralTransform
SpectralSelection
FrequencySpecification
[7 30]
Feature Extraction
FeatureExtraction
PatternPairs
3
PluginFunctions
FeatureAdaptor
para_csp::@csp_train
FeatureExtractor
para_csp::@csp_predict
FeatureAdaptorNeedsVoting
FeatureVisualizer
para_csp::@csp_visualize
Machine Learning
MachineLearning
(Name)
(Description)
Help
Cancel
OK
Calibrate a model
Selected approach
lastapproach ("Imagined Movements vi...
Calibration data source
lastdata ("imag.set")
Parameter Search
Loss/Performance Metric
Automatically chosen
Cross-validation folds
5
Spacing around test trials
5
Performance estimates
Compute performance estimates
Cross-validation folds
10
Spacing around test trials
5
Computing resources
Run on a computer cluster
Node pool
(use current config)
Save model in workspace as
lastmodel
Save stats in workspace as
laststats
Help
Cancel
OK
Figure 2: Common Spatial Patterns
File Edit View Insert Tools Desktop Window Help
CSP Pattern 1
CSP Pattern 2
CSP Pattern 3
CSP Pattern 6
CSP Pattern 5
CSP Pattern 4
Figure 2
File Edit View Insert Tools Desktop Window Help
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
2

Getting Data In: The Lab Streaming Layer



Inbox - christiankothe@gmail × labstreaminglayer - Distribute ×

code.google.com/p/labstreaminglayer/

labstreaminglayer

Distributed signal transport, time synchronization and collection system for research use

Project Home Downloads Wiki Issues Source Administer

Summary People

Tip: Discuss and then document each teammate's project duties.

Project Information

Recommend this on Google

Starred by 0 users

Project feeds

Code license

MIT License

Labels

Academic, Interface, Lab, Library, Middleware, Networking, Stream, Research

Members

christiankothe

3 committers

Your role

Owner

Summary

The **lab streaming layer** (LSL) is a system for the unified collection of measurement time series in research experiments and handles both the networking, time-synchronization, (near-) real-time access as well as optionally the centralized collection, viewing and disk recording of the data.

The **LSL distribution** consists of:

- The core transport library (liblsl) and its language wrappers (MATLAB, Python, C, C++). The library is general-purpose and cross-platform (Win/Linux/MacOS, 32/64) and forms the heart of the project.
- A suite of tools built on top of the library, including the recording program, a viewer program, importers, and a set of data collection apps that make data from a particular device available on the lab network (for example audio, EEG, or motion capture). The existing tools suite is tailored to the needs of only a small number of labs and should not be considered as general (or production-quality) as the library itself.

Streaming Layer API

The liblsl library provides the following **abstractions** for use by client programs:

- **Stream Outlets**: for making time series data streams available on the lab network. The data is pushed sample-by-sample or chunk-by-chunk into the outlet, and can consist of single- or multichannel data, regular or irregular sampling rate, with uniform value types (integers, floats, doubles, strings). Streams can have arbitrary XML meta-data (akin to a file header). By creating an outlet the stream is made visible to a collection of computers (defined by the network settings/layout) where one can subscribe to it by creating an inlet.
- **Resolve functions**: these allow to resolve streams that are present on the lab network according to content-based queries (for example, by name, content-type, or queries on the meta-data). The service discovery features do not depend on external services such as zeroconf and are meant to drastically simplify the data collection network setup.
- **Stream Inlets**: for receiving time series data from a connected outlet. Allows to retrieve samples from the provider (in-order, with reliable transmission, optional type conversion and optional failure recovery). Besides the samples, the meta-data can be obtained (as XML blob or alternatively through a small built-in DOM interface).
- **Built-in clock**: Allows to time-stamp the transmitted samples so that they can be mutually synchronized. See Time Synchronization.

Time Synchronization

code.google.com/p/labstreaminglayer

Key Features

- System for the unified access to measurement time series from devices and applications (incl. events)
- Supports centralized collection, viewing and disk recording of the data (unified file format: XDF)
- Handles time-synchronization between multiple streams (to sub-ms precision, up to device uncertainty), networking, fault tolerance
- Library & Examples for C/C++/Python/MATLAB, Win/Linux/MacOS, 32/64bit
- Plugins for EEGLAB, BCILAB, MoBILAB

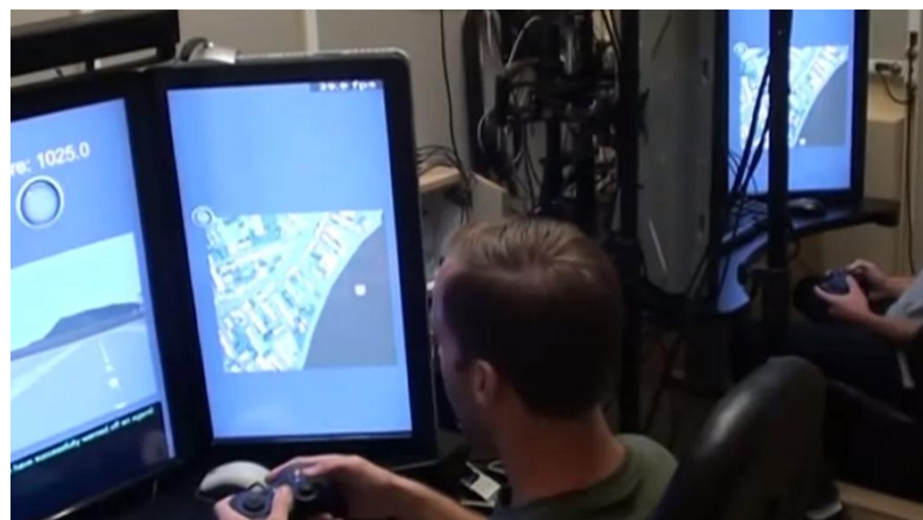
Currently Supported Hardware

- **EEG**: Biosemi, Cogionics, MINDO, BrainProducts, g.USBamp, Emotiv, Micromed, MindMedia, OpenEEG, TMSi, ANT Neuro ASALAB
- **Eye Tracking:** SR Research EyeLink, custom 2-camera setup
- **Motion Capture:** PhaseSpace, OptiTrack, Kinect, AMTI Force Plates
- **Human-Interface Devices:** Mice, Keyboards, Trackballs, Game Controllers, Wiimote and Expansions
- **Multimedia Devices:** PC-compatible sound cards, DirectShow-compatible video hardware
- **Untested:** ABM B-Alert, Enobio, Neuroscan Synamp, EGI AmpServer, Mitsar EEG, CTF/VSM, Tobii, SMI iViewX

Getting Data Out

- BCILAB provides several output protocols (e.g., TCP, OSC, LSL); also allows for custom extensions, e.g., for Presentation or ePrime
- Also supports SNAP natively (our Python-based stimulus-presentation environment)

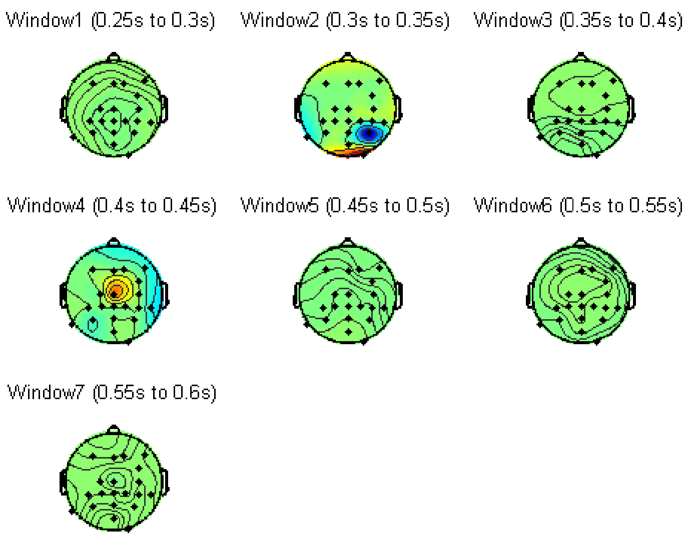
…



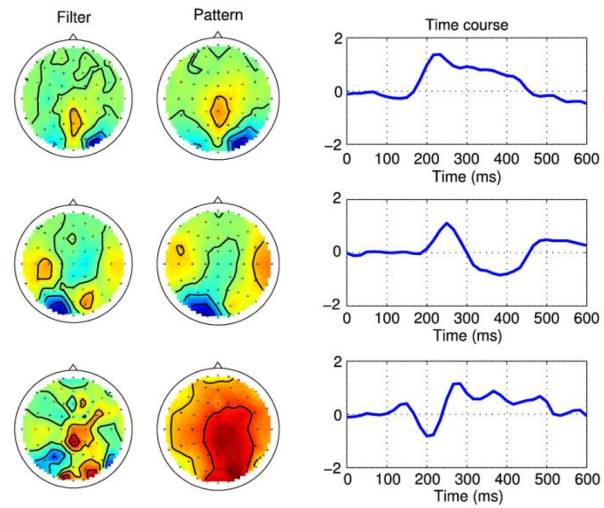
6 Methods Tour

Time-Domain / ERP Baseline

Windowed Means



DAL-ERP



(*)

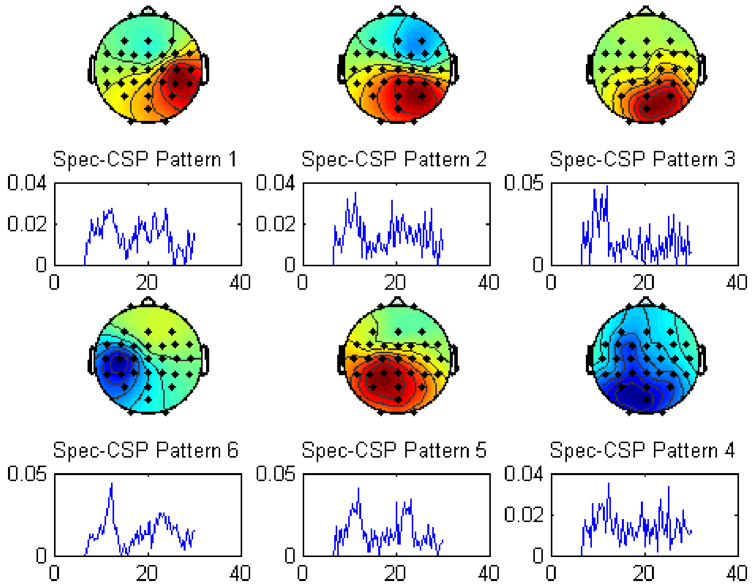
- Traditional linear classifier for event-locked brain responses, usually using LDA
- Time windows manually assigned
- Examples: error recognition, surprise

- State-of-the-art approach, no hand-tuned parameters
- Uses rank-regularized logistic or linear regression

(*image: Tomioka et al., 2010)

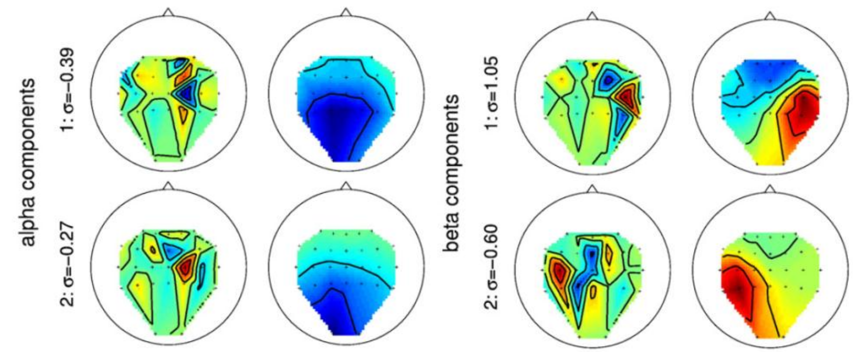
Oscillatory Processes Baseline

Common Spatial Patterns Family



- Filter-Bank CSP (FBCSP): multiple bands/windows
- Diagonal Loading CSP (DLCSP): cov. shrinkage
- Composite CSP (CCSP): covariance prior
- Tikhonov-regularized CSP (TRCSP): filter shrinkage
- Spectrally weighted CSP (Spec-CSP): learning spectral filters from the data

DAL-OSC



(*)

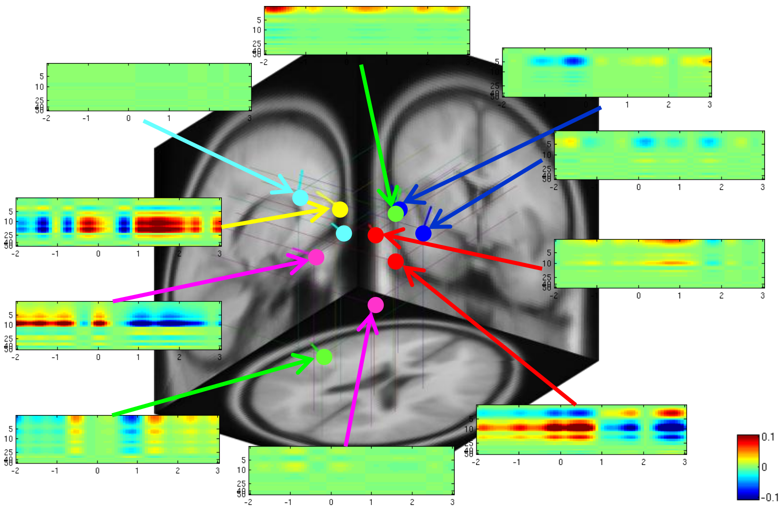
- State-of-the-art approach, no hand-tuned parameters
- Also uses rank-regularized logistic or linear regression
- Single-step approach, jointly optimal

(*image: Tomioka et al., 2010)

New Methods

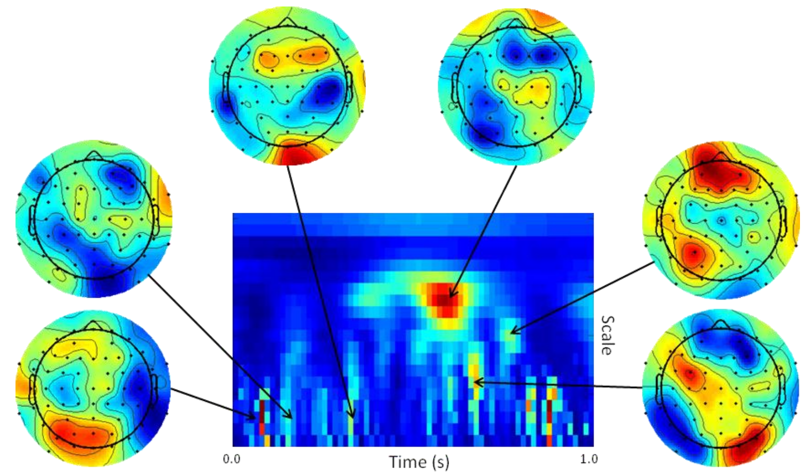
Methods for Oscillatory Analysis

(below: Regularized Spatio-Spectral Dynamics)



Methods for Time-Domain Analysis

(below: Wave Propagation Imaging)

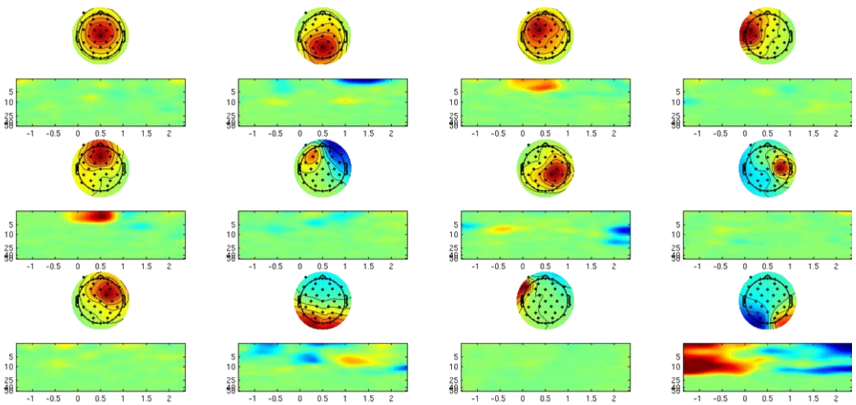


- Applicable to slowly-changing operator state and background activity as well as event-related transients
- RSSD is a pioneering method for learning full source-level time/frequency structure
- Examples: cognitive load, attention shifts
- Presented at ICON’11; methods and data papers in preparation

- Classify event-locked brain responses
- Best methods to date learn optimal evolving spatial filters (as above)
- Several methods in the same performance ballpark
- Examples: error recognition, surprise
- Benchmark paper in preparation

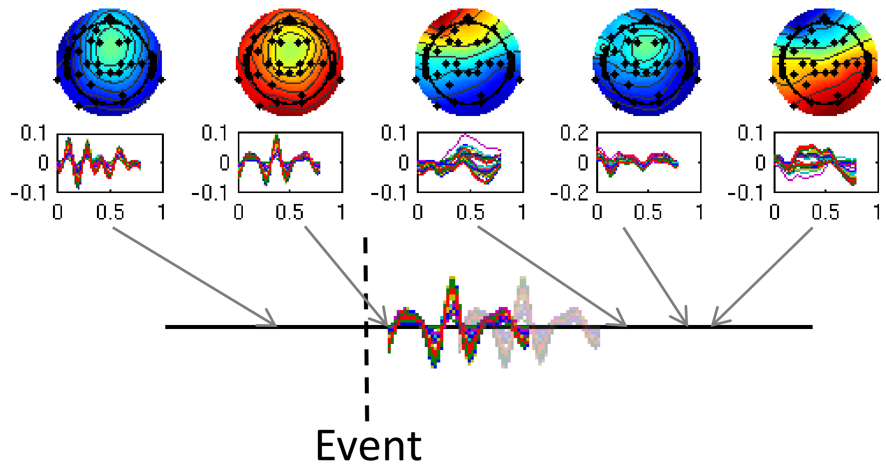
New Methods (Exploratory)

Spatio-Spectral Bayes



- A fully Bayesian version of RSSD aimed at neuroscientific modeling
- Allows for extensive statistical analysis of results
- Presented at Sloan-Swartz ‘11

Pattern Alignment Learning

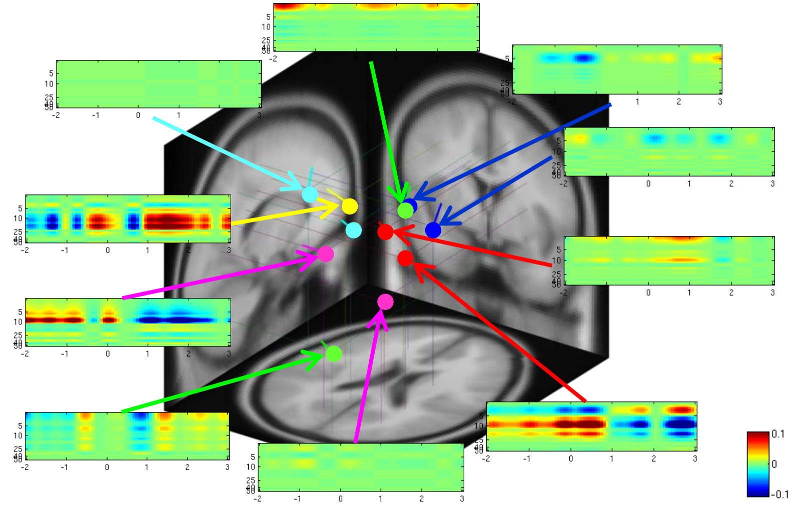


- Finds time-jittered brain processes associated with known events in the work environment
- Radically new approach using joint optimization
- Applications: target event detection and other event-related cognitive responses

7 Future Directions

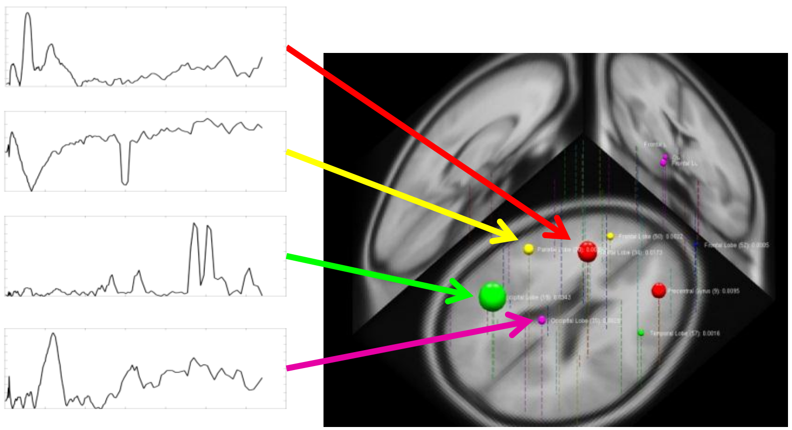
Current and Future Directions

- Making principled use of anatomical prior knowledge; requires that learned parameters are endowed with anatomically meaningful locations
- First step in this direction: RSSD, using Independent Component Analysis and Dipole Fitting to obtain localized parameters
- Use Beamforming, NFT, …



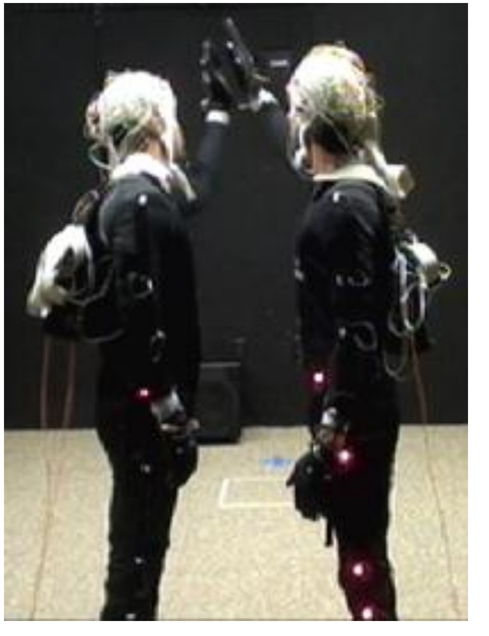
Current and Future Directions

- Learning models from data spanning multiple persons (using multi-task learning, empirical Bayesian methods, mixed-effects models, etc.)
- Currently only one such implementation in BCILAB (multi-subject-OSR)



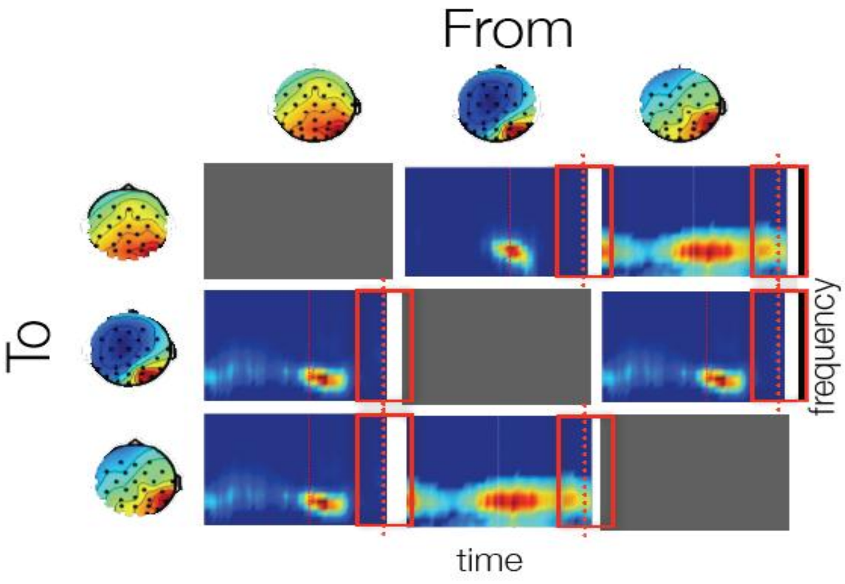
Current and Future Directions

- Integrating motion capture information and other peripheral and behavioral measures into BCIs (e.g., eye tracking, facial expression, …)
- Can explain away artifacts and interfering factors, contains rich information about cognitive state by themselves
- Requires deep integration with the MoBILAB toolbox



Current and Future Directions

- Leveraging Granger-causal effective connectivity measures as features for BCIs (using the SIFT toolbox)
- Connectivity contains far richer structure than univariate (per-source) measures



A Further Reading

These and Futher Slides:

ftp://sccn.ucsd.edu/pub/bcilab/

BCI Papers Worth Reading

- B. Blankertz, S. Lemm, M. Treder, S. Haufe, and K.-R. Mueller, "Single-trial analysis and classification of ERP components - A tutorial", NeuroImage, vol. 56, no. 2, pp. 814–825, May 2011.
- F. Lotte and C. Guan, “Regularizing common spatial patterns to improve BCI designs: unified theory and new algorithms,” IEEE Transactions on Biomedical Engineering, vol. 58, no. 2, pp. 355-362, Feb. 2011.
- R. Tomioka and K.-R. Mueller, A regularized discriminative framework for EEG analysis with application to brain-computer interface", NeuroImage, vol. 49, no. 1, pp. 415–432, 2010.
- B. Blankertz, G. Dornhege, M. Krauledat, K.-R. Mueller, and G. Curio, "The non-invasive Berlin brain-computer interface: Fast acquisition of effective performance in untrained subjects", NeuroImage, vol. 37, no. 2, pp. 539–550, Aug. 2007.
- M. Grosse-Wentrup, C. Liefhold, K. Gramann, and M. Buss, "Beamforming in noninvasive brain-computer interfaces", IEEE Trans. Biomed. Eng., vol. 56, no. 4, pp. 1209–1219, Apr. 2009.

BCI Surveys

- A. Bashashati, M. Fatourechi, R. K. Ward, and G. E. Birch, "A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals", J. Neural Eng., vol. 4, no. 2, pp. R32–R57, Jun. 2007.
- F. Lotte, M. Congedo, A. Lecuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain-computer interfaces", J. Neural Eng., vol. 4, no. 2, pp. R1–R13, Jun. 2007.
- S. Makeig, C. Kothe, T. Mullen, N. Bigdely-Shamlo, Z. Zhang, K. Kreutz-Delgado, "Evolving Signal Processing for Brain–Computer Interfaces", Proc. IEEE, vol. 100, pp. 1567-1584, 2012.

Interesting Technical Papers

- D.P. Wipf and S. Nagarajan, “A Unified Bayesian Framework for MEG/EEG Source Imaging,” NeuroImage, vol. 44, no. 3, February 2009.
- S. Haufe, R. Tomioka, and G. Nolte, “Modeling sparse connectivity between underlying brain sources for EEG/MEG,” Biomedical Engineering, no. c, pp. 1-10, 2010.
- S. Boyd, N. Parikh, E. Chu, and J. Eckstein, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” Information Systems Journal, vol. 3, no. 1, pp. 1-122, 2010.
- P. Zhao and B. Yu, “On Model Selection Consistency of Lasso,” Journal of Machine Learning Research, vol. 7 pp. 2541-2563, 2006.

Technical Papers, ct'd

- J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng, "Multimodal Deep Learning," in Proceedings of the 28th International Conference on Machine Learning, 2011.
- K. N. Kay, T. Naselaris, R. J. Prenger, and J. L. Gallant, "Identifying natural images from human brain activity," Nature, vol. 452, no. 7185, pp. 352-355, Mar. 2008.
- O. Jensen et al., "Using brain-computer interfaces and brain-state dependent stimulation as tools in cognitive neuroscience," Frontiers in Psychology, vol. 2, p. 100, 2011.
- D.-H. Kim, N. Lu, R. Ma,. Y.-S. Kim, R.-H. Kim, S. Wang, J. Wu, S. M. Won, H. Tao, A. Islam, K. J. Yu, T.-I. Kim, R. Chowdhury, M. Ying, L. Xu, M. Li, H.-J. Cung, H. Keum, M. McCormick, P. Liu, Y.-W. Zhang, F. G. Omenetto, Y Huang, T. Coleman, J. A. Rogers, "Epidermal electronics," Science vol. 333, no. 6044, 838-843, 2011.

Researchers to Watch

- Klaus-Robert Mueller et al. (TU Berlin) – one of the leading BCI groups
  http://www.bbci.de/publications.html
- Marcel van Gerven et al. (Donders) – BCI and Neuroscience with a Bayesian approach
  https://sites.google.com/a/distrep.org/distrep/publications
- Ryota Tomioka (U Tokyo) – known for some technical achievements
  http://www.ibis.t.u-tokyo.ac.jp/RyotaTomioka
- Karl Friston et al. (UC London) – working on relevant underpinnings for neuroimaging (outside BCI)
  http://www.fil.ion.ucl.ac.uk/Research/publications.html
- Leading Statisticians and Machine Learners: Michael I. Jordan, Andrew Ng, Lawrence Carin, Zoubin Ghahramani, Francis Bach, Geoffrey Hinton, Ruslan Salakhutdinov, Yeh Whye Teh, David Blei, …

Thanks!

Questions?