Lecture 4: Adaptivity and Machine Learning

Introduction to Modern Brain-Computer Interface Design

Christian A. Kothe
SCCN, UCSD

Outline

1. Adaptivity in BCIs
2. Machine Learning
3. Concrete Case Study
4. Performance Evaluation


Swartz
Center for
Computational
Neuroscience

4.1 Adaptivity in BCIs

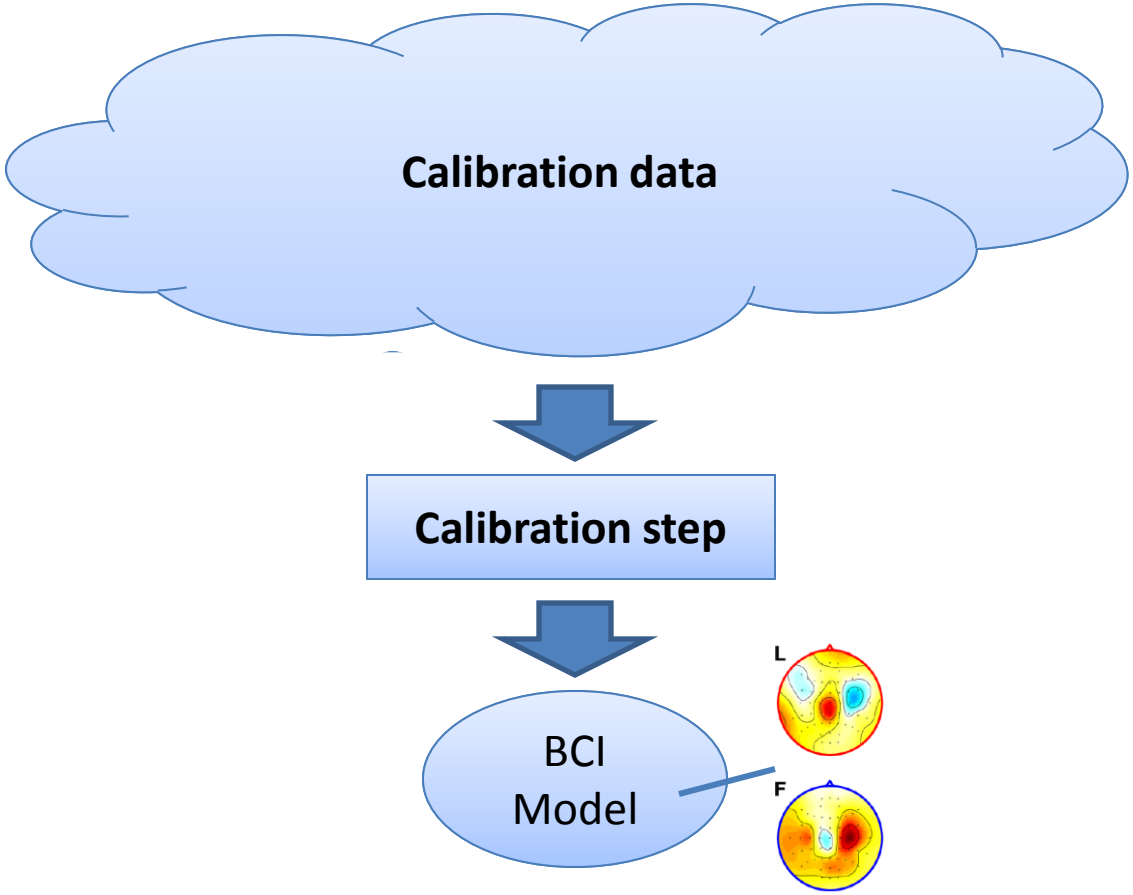
Unknown Parameters

- for most BCI questions and implementations, the parameters leading to best accuracy (**W**,b, …) are *a priori* unknown…
  - Depend on highly variable factors (e.g., sensor placement, subject state)
  - Different for every person, task, montage, etc.
  - Depend on hard-to-measure factors (e.g., brain functional map)
  - Depend on expensive-to-measure factors (e.g., brain folding)

- How to solve this problem?

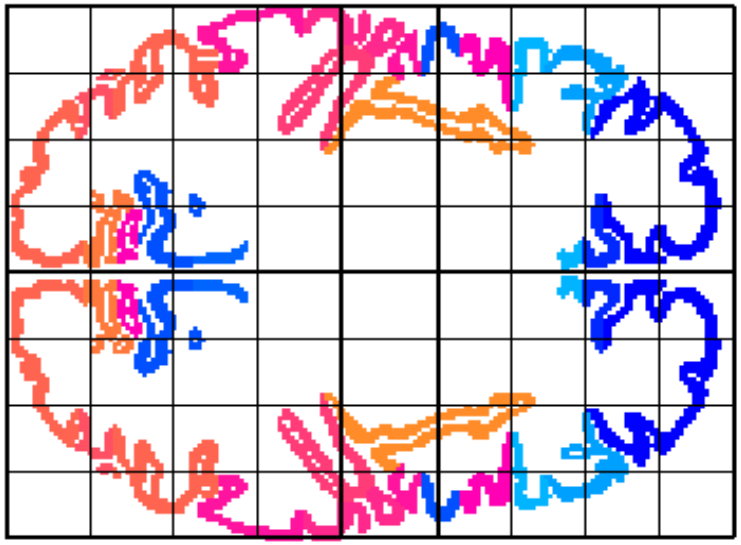
Model Calibration

- Can use *calibration / training data* to estimate parameters from, and a separate *calibration step*



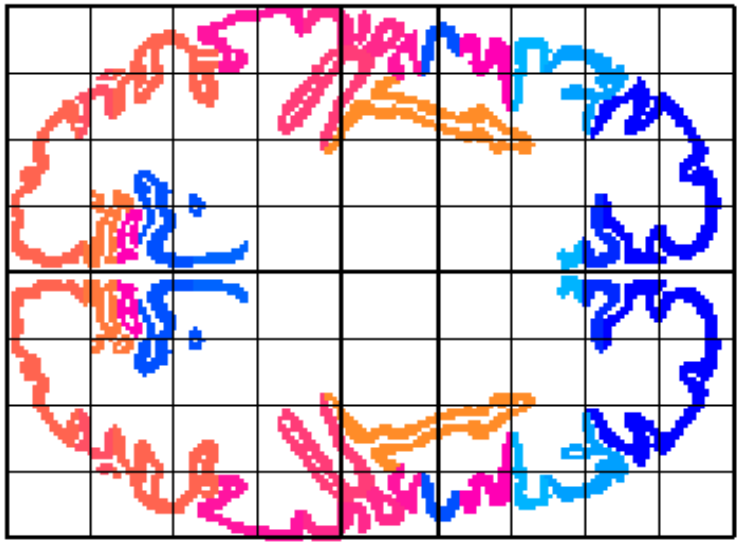
Prior Knowledge

- Prior knowledge is neuroscientific, such as:
  - Anatomical atlases (e.g. Talairach, LONI)
  - Functional atlases (if available)



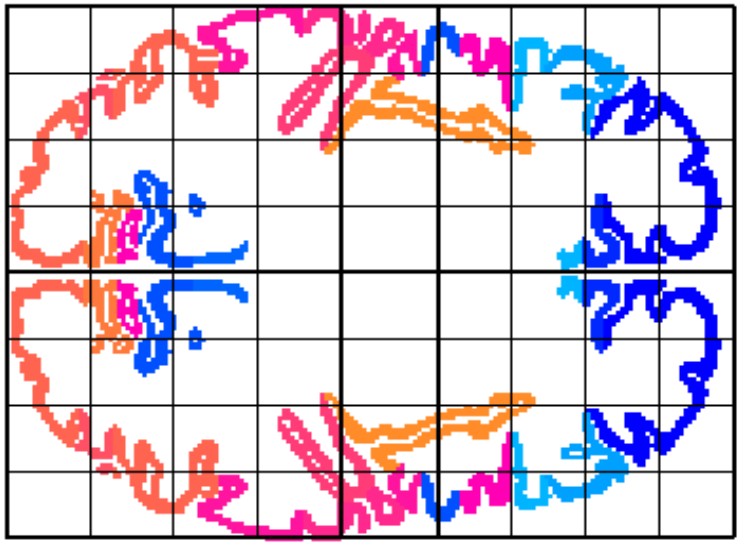
Prior Knowledge

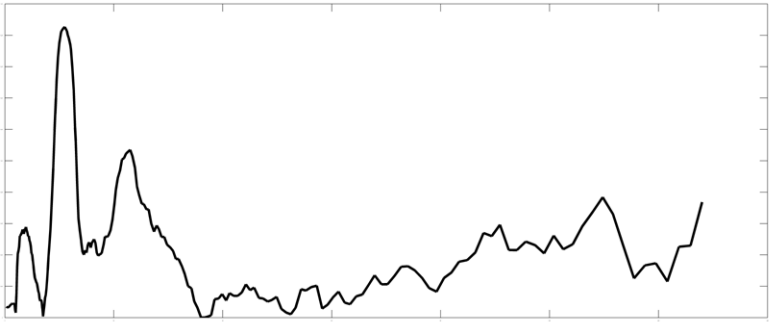
- Prior knowledge is neuroscientific, such as:
  - Anatomical atlases (e.g. Talairach, LONI)
  - Functional atlases (if available)
  - Timing information (e.g. neural latencies, reaction times)



Prior Knowledge

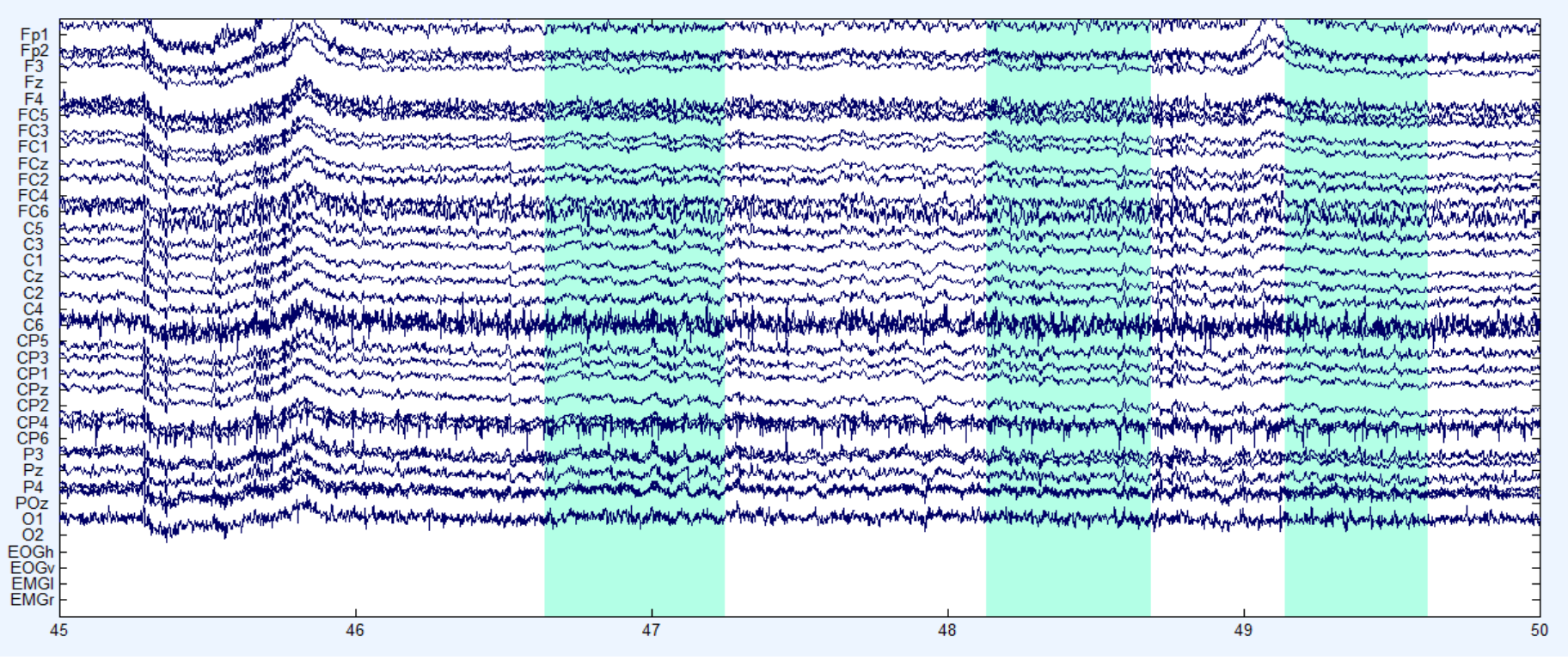
- Prior knowledge is neuroscientific, such as:
  - Anatomical atlases (e.g. Talairach, LONI)
  - Functional atlases (if available)
  - Timing information (e.g. neural latencies, reaction times)
  - Frequency bands of oscillatory processes (alpha, beta, theta, …)





Calibration Data

- Example/calibration data is used to calculate optimal parameters of a BCI, and is *extremely important*



The Ideal Calibration Data

- Collected with the same/similar measurement apparatus as used for online runs
  - otherwise extra transformations and uncertainty incurred
- Comprises *multiple independent realizations / repetitions / trials* (to quantify variability)
  - one-shot learning (one exemplar) is *much* harder

The Ideal Calibration Data

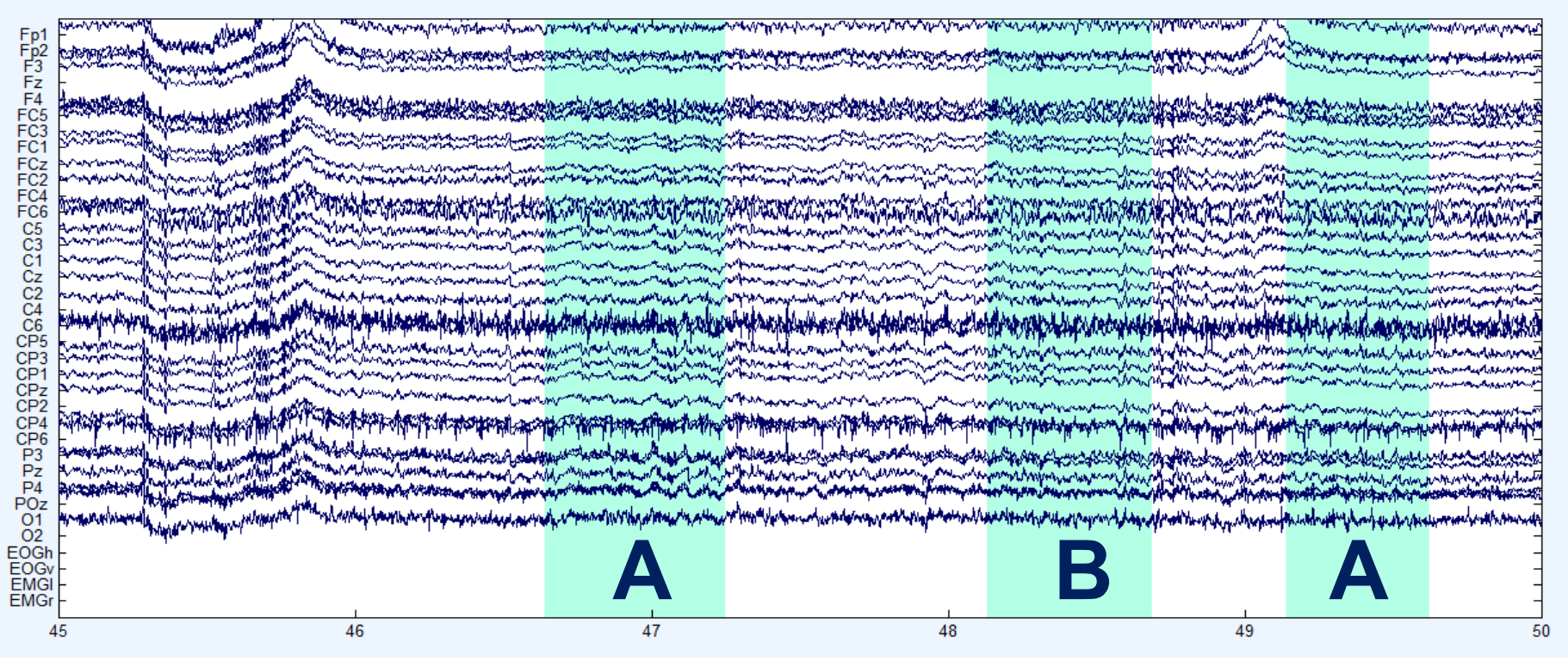
- Collected under conditions that are as close to those of the online runs as possible (i.e., drawn from the same statistical distribution)
  - Same person is preferable
  - Same sensor arrangement is preferable
  - Same session is preferable
  - Task parameters (stress level, …) should be similar
- Obviously a cost/benefit tradeoff:
  - Would trade off some performance for being able to reuse one recording for multiple sessions and persons

The Ideal Calibration Data

- If there is a bias (e.g., different session), data should cover multiple realizations (e.g., multiple sessions) to capture variability
- A plain EEG recording is “unlabeled” (no knowledge about the association between raw observed signal and the cognitive state variable of interest)
- Labeled data (person is “surprised” / “not surprised”) is *far* more useful than unlabeled

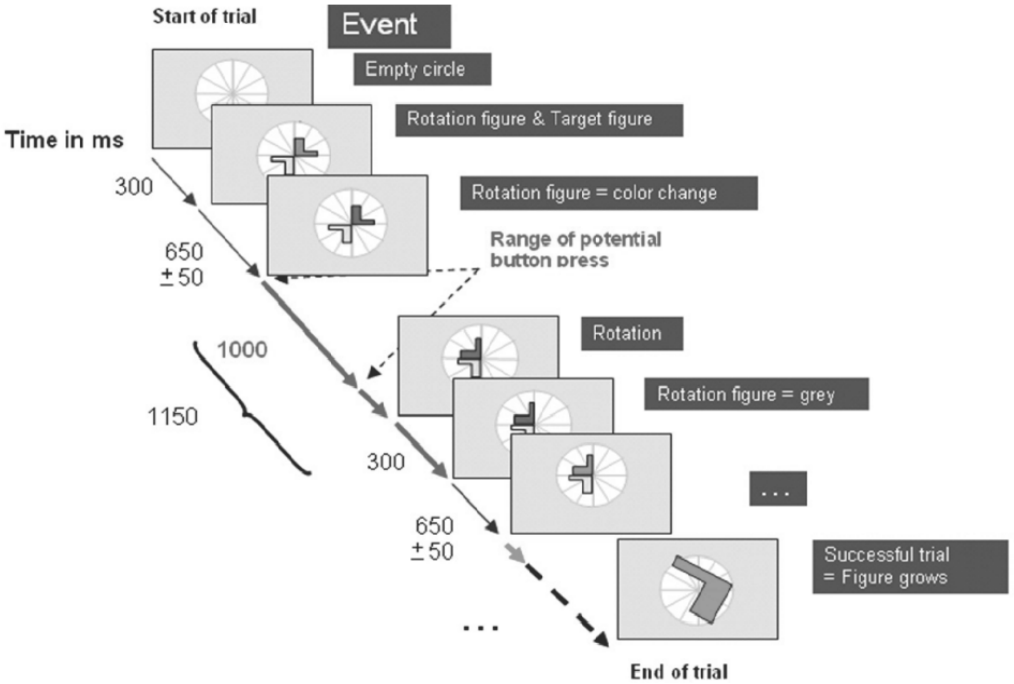
The Ideal Calibration Data

- Labels are assigned per realization (e.g., per trial) and *index the output that the BCI shall produce for this class of data*



Summary

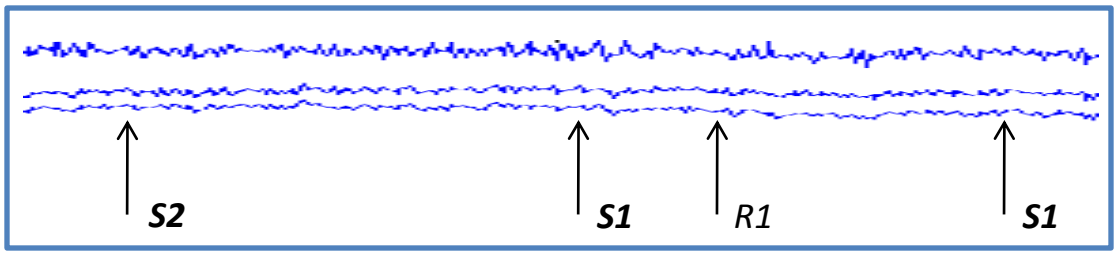
- The required data to calibrate a BCI resembles data produced by *controlled psychological experiments*



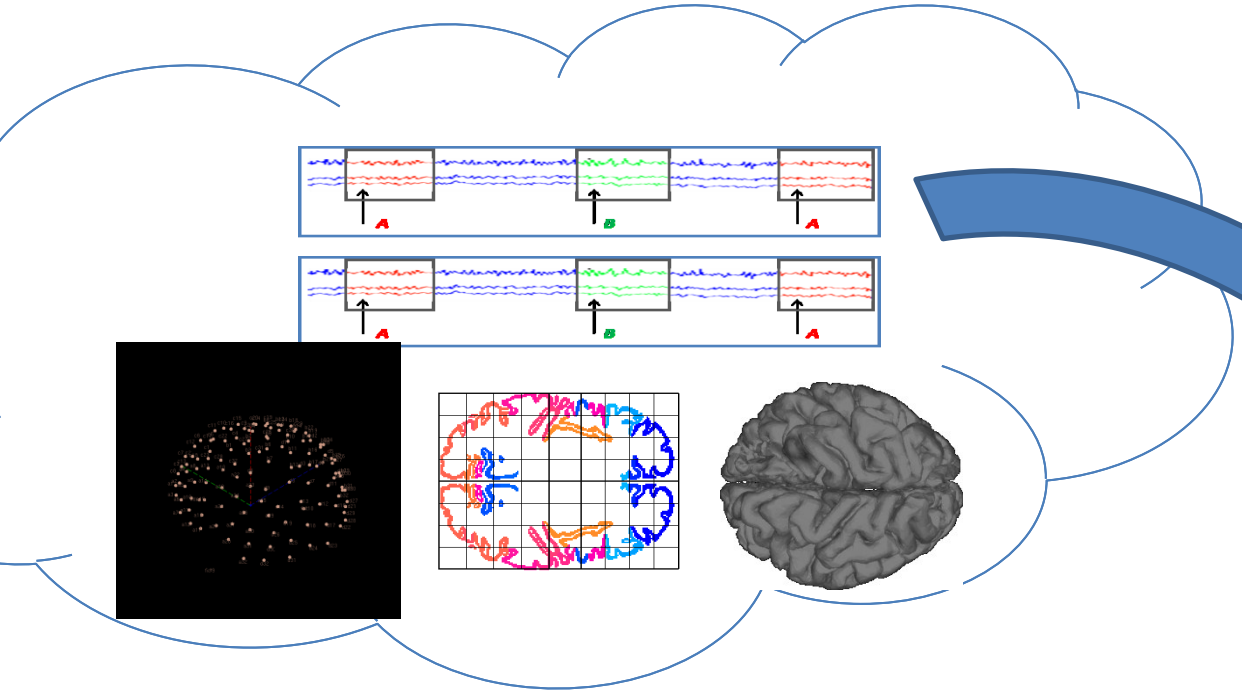
Zander et al., 2010

Summary

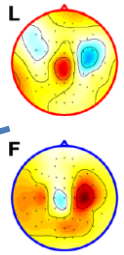
- Features
  - continuous EEG (or other)
  - multiple trials/blocks (capturing variation)
  - randomized (eliminating confounds)
  - event markers to encode cognitive state conditions of interest, e.g., stimuli/responses (called "*target markers*" in BCILAB)
- Can also be used for offline performance tests

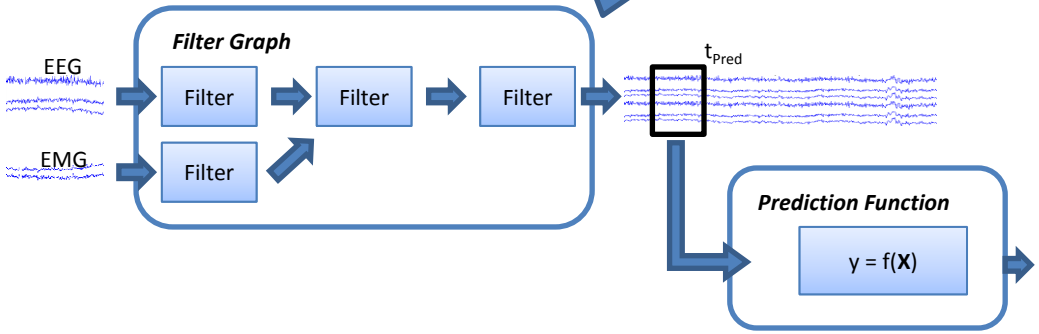


Big Picture



Learning Function



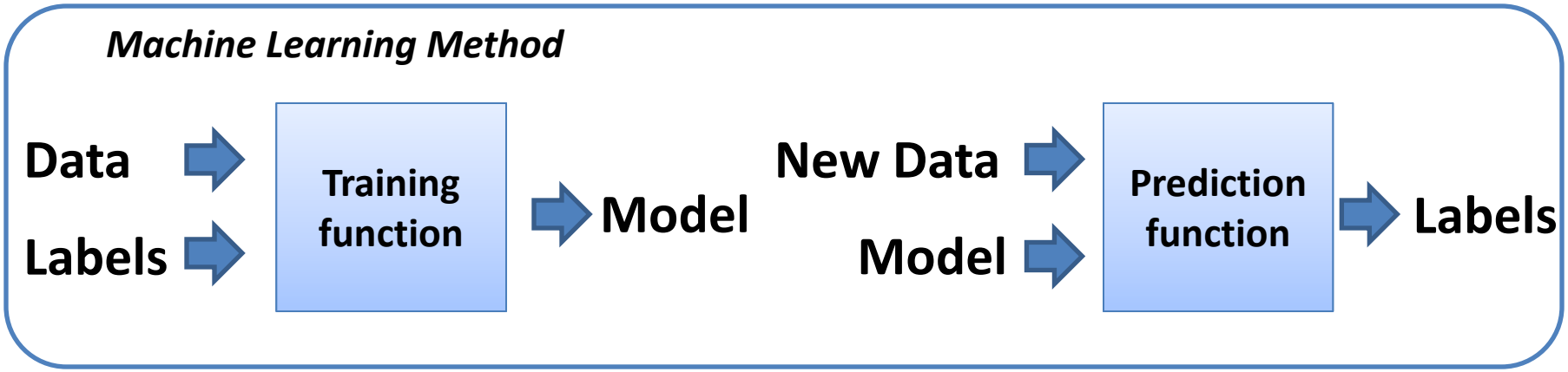



Swartz
Center for
Computational
Neuroscience

4.2 Machine Learning

Machine Learning

- Large field with 100s of algorithms
- Most methods conform to a common framework of a *training function* and a *prediction function*



- Intermediate model parameters $\boldsymbol{\theta}$ capture the learned relationship
- Data $\boldsymbol{X} \in \mathbb{R}^{N \times F}$ and Labels / target values $\boldsymbol{y} \in \mathbb{R}^{N \times D}$
  N = #trials, F = #features, D = #output dims.

Machine Learning

- The Machine Learning Framework is largely *trial-based* (learning from exemplars)
- Most methods come in form of two functions: *learning function* and *prediction function*
- Learning function is often *far* more complex than the prediction function

Sub-Types In ML

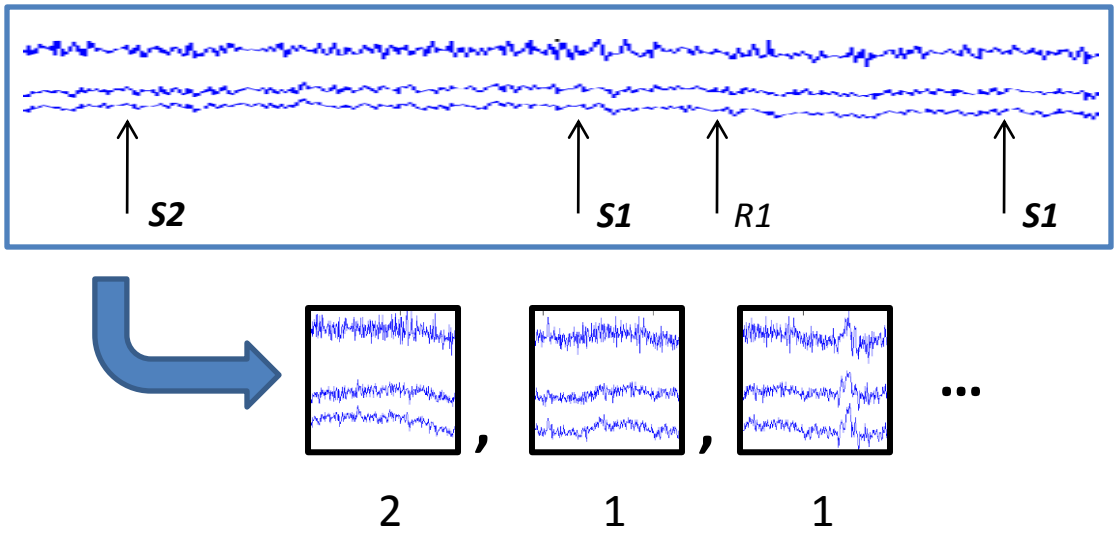
- **Supervised Learning:** given a set of (input,output) pairs as training data, learn a parametric (or "non-parametric") model M that encodes the mapping from input to output
- **Unsupervised Learning:** given a set of training examples, learn the structure in the input space (e.g. clusters, manifolds, probability density)
- **Semi-Supervised Learning:** Some training examples have labels, others do not
- **Others:** e.g., Active Learning, Online Learning, …

Related Areas

- Probability Theory
- Statistics
- Optimization
- Neural Networks
- Artificial Intelligence, …

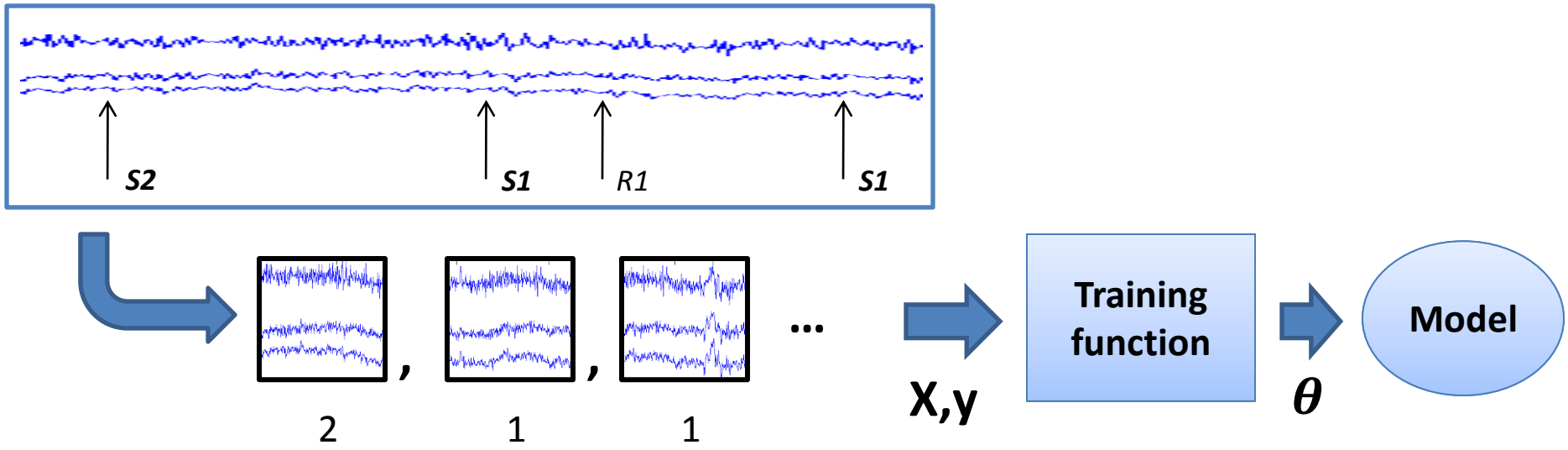
Using Machine Learning

- 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



Using Machine Learning

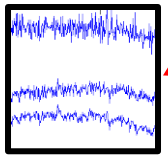
- The training function computes a parameter (here $\boldsymbol{\theta}$) of the prediction function *such that the performance on the given example data is optimal*
- What is considered optimal depends on *extra assumptions* (a.k.a. *priors*)



Detour: 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)

1000s of degrees of freedom!



Detour: Feature Extraction

- **Solution**: Introduce additional mapping (called *"feature extraction")* from raw signal segments onto feature vectors
  - output is often of lower dimensionality
  - hopefully statistically "better" distributed (easier to handle for machine learning)

Examples of Feature Extraction

- Depends on the process of interest (e.g. oscillation, ERP peak, complex phenomenon)
- For oscillations, e.g.:
  - log-Variance (logarithm yields more convenient data distributions)
  - Part of the Fourier spectrum
- For ERPs, e.g.:
  - Peak latency, height, width (artificial example)
  - Mean in one or more time ranges relative to an event
  - Subset of Wavelet coefficients


Swartz
Center for
Computational
Neuroscience

Swartz Center for Computational Neuroscience

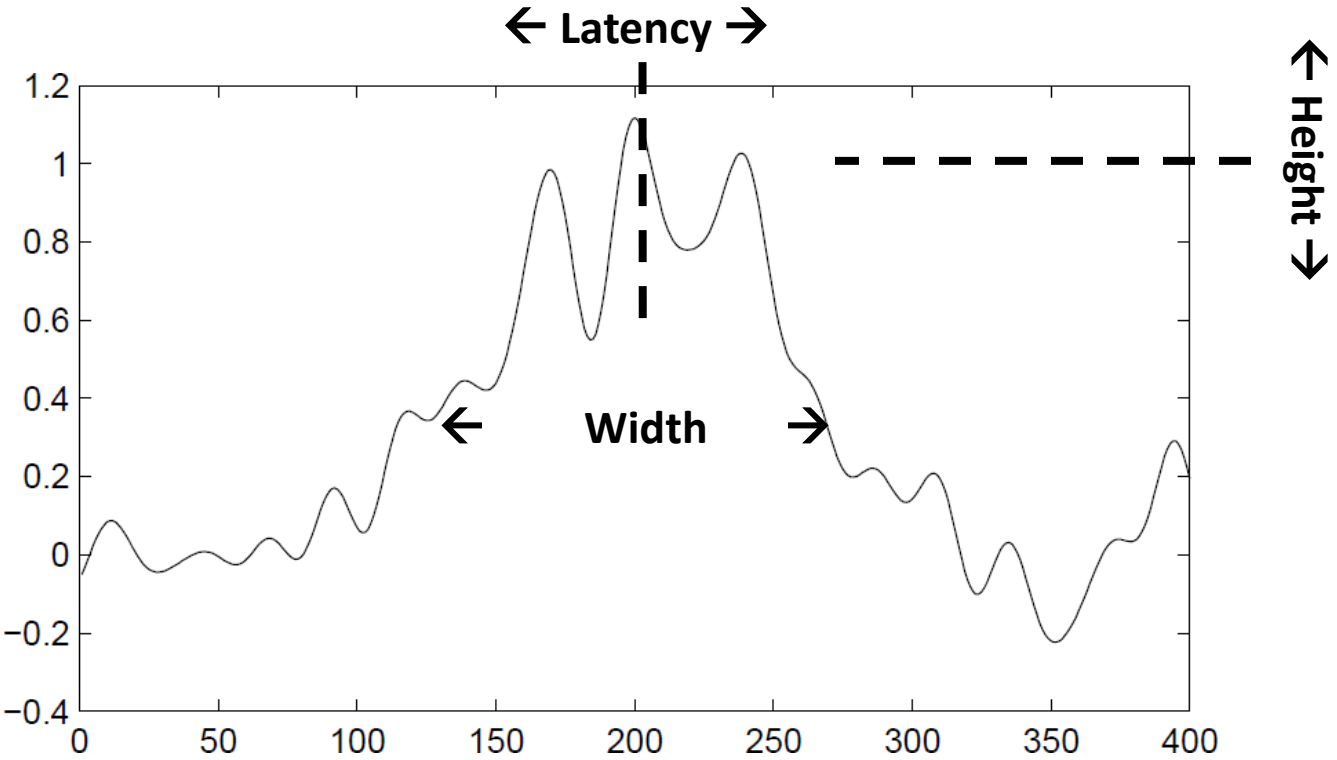
4.3 Concrete Case Study

Example Calibration Problem

- **Task:** A person is presented with a sequence of 300 images (one ever 2 seconds). Half of the images are exciting, the other half are not. One channel of EEG (at Cz location) is recorded.
- **Question:** How to design a BCI that can determine whether a person is shown an exciting or a non-exciting image?
- **Approach:** For each trial k, cut out an epoch $\mathbf{X}_k$ of 1s length, extract a short vector of features $\mathbf{f}_k$, and assign a label y_k in {E,NE}. Use machine learning to find an optimal statistical mapping from $\mathbf{f}_k$ onto y_k.

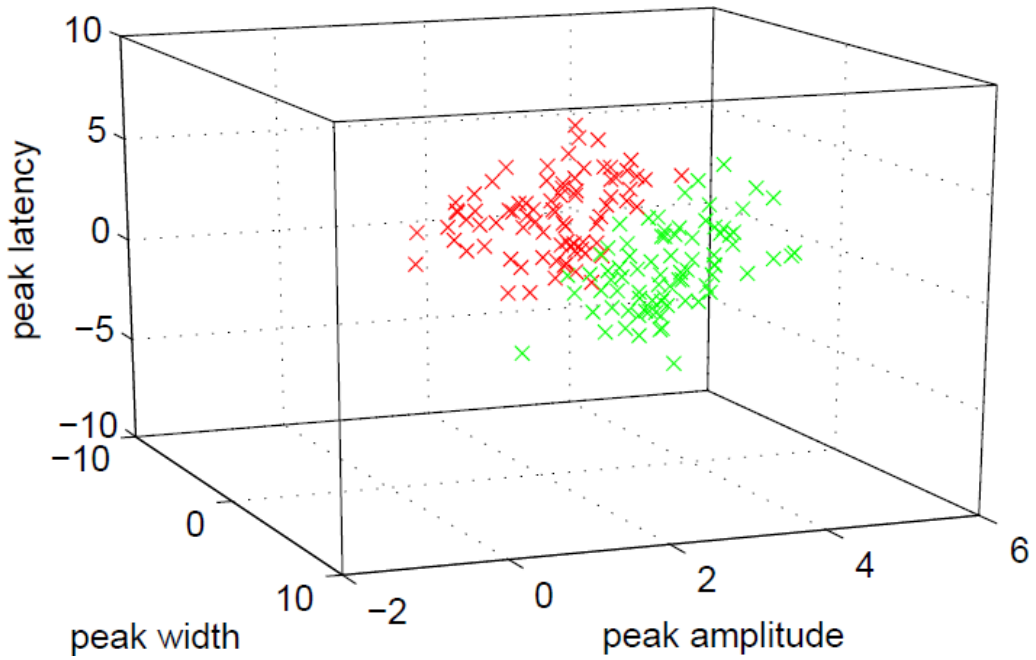
Extracting Features of a Peak

- A supposed characteristic peak in a time window (relative to an event) could be characterized by three parameters:



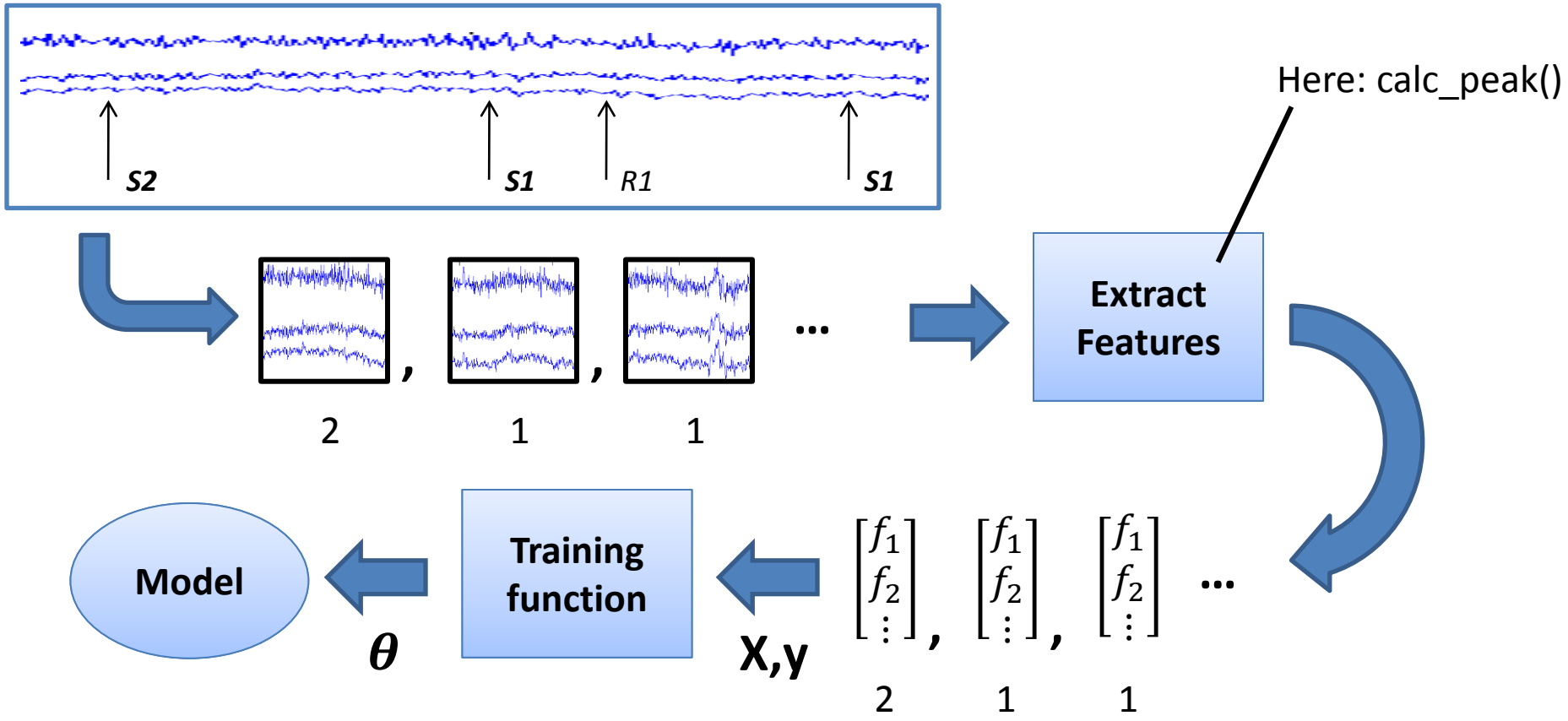
Resulting Feature Space

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



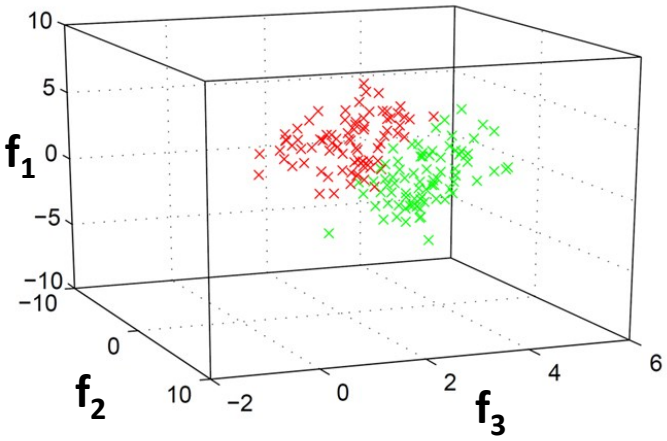
ML with Feature Extraction

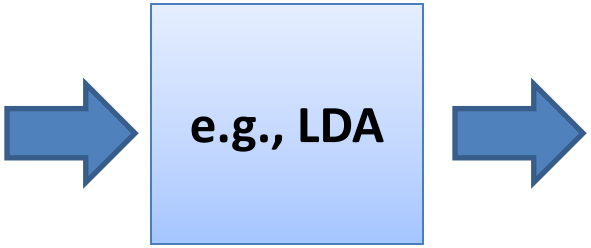
- Including the feature extraction, the analysis process is as follows:

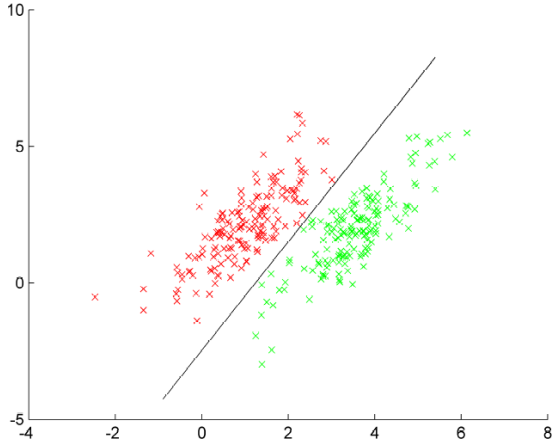


Using Machine Learning

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

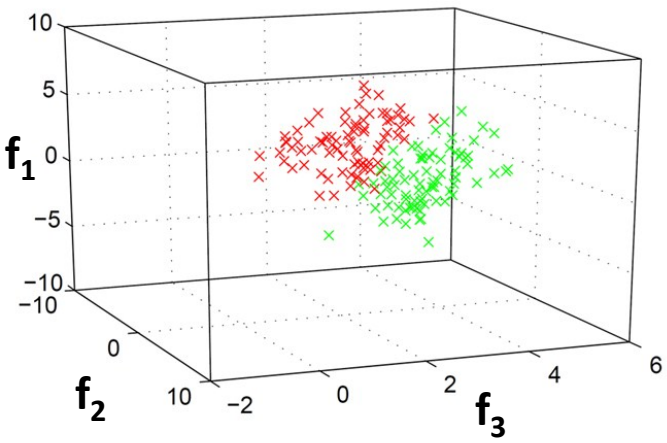


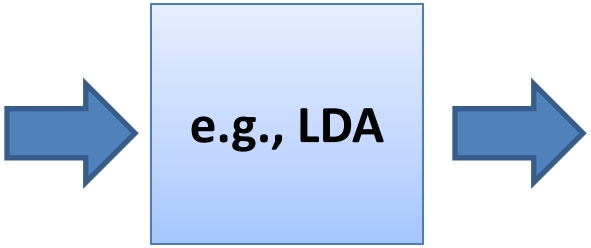


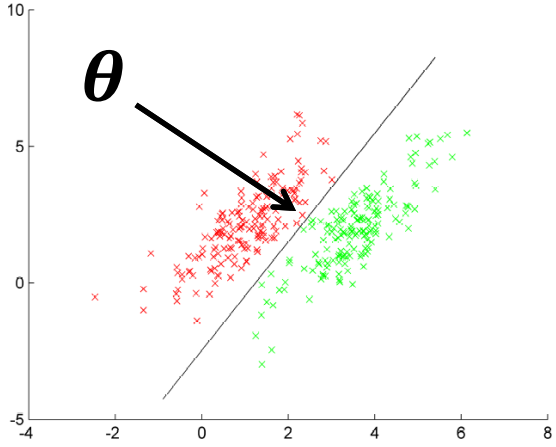


Using Machine Learning

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





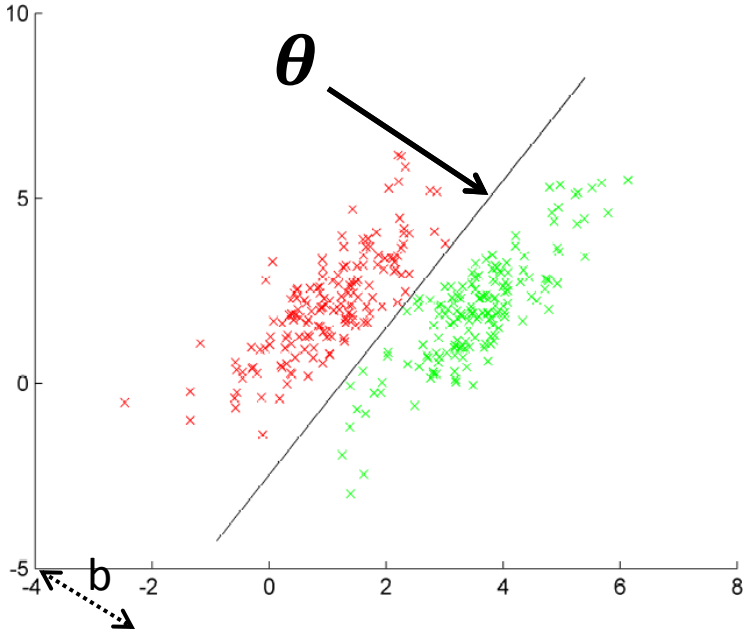


LDA In a Nutshell

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

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

$$\boldsymbol{\theta} = (\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1), \qquad \mathrm{b} = -\boldsymbol{\theta}^{\mathsf{T}}(\boldsymbol{\mu}_1+\boldsymbol{\mu}_2)/2$$



Resulting Predictive Map

- LDA generates parameters of a linear mapping

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

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

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

More on LDA

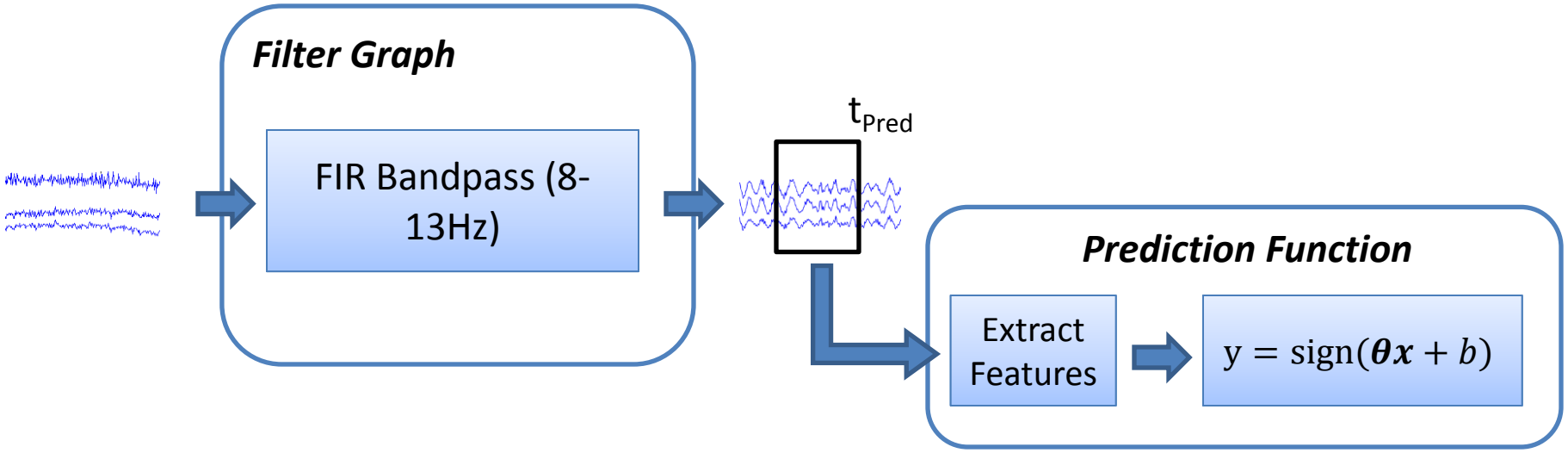
- **Assumptions:**
  - Data in each class is distributed according to a Gaussian distribution
  - Shape of the distribution is identical for all classes
- **Benefits:**
  - Simple, fast and optimal in the large-sample limit if assumptions are true
- **Problems:**
  - Very sensitive to outliers (non-Gaussian)
  - Covariance matrix estimates become unreliable / unusable for too few trials and too many dimensions

Putting it Together

1. Apply band-pass filter to calib. recording
2. Extract epochs relative to target markers
3. Extract features for each epoch (here: peaks)
4. Submit all feature vectors & target labels to LDA to calculate $\boldsymbol{\theta}$ and b of the predictive mapping

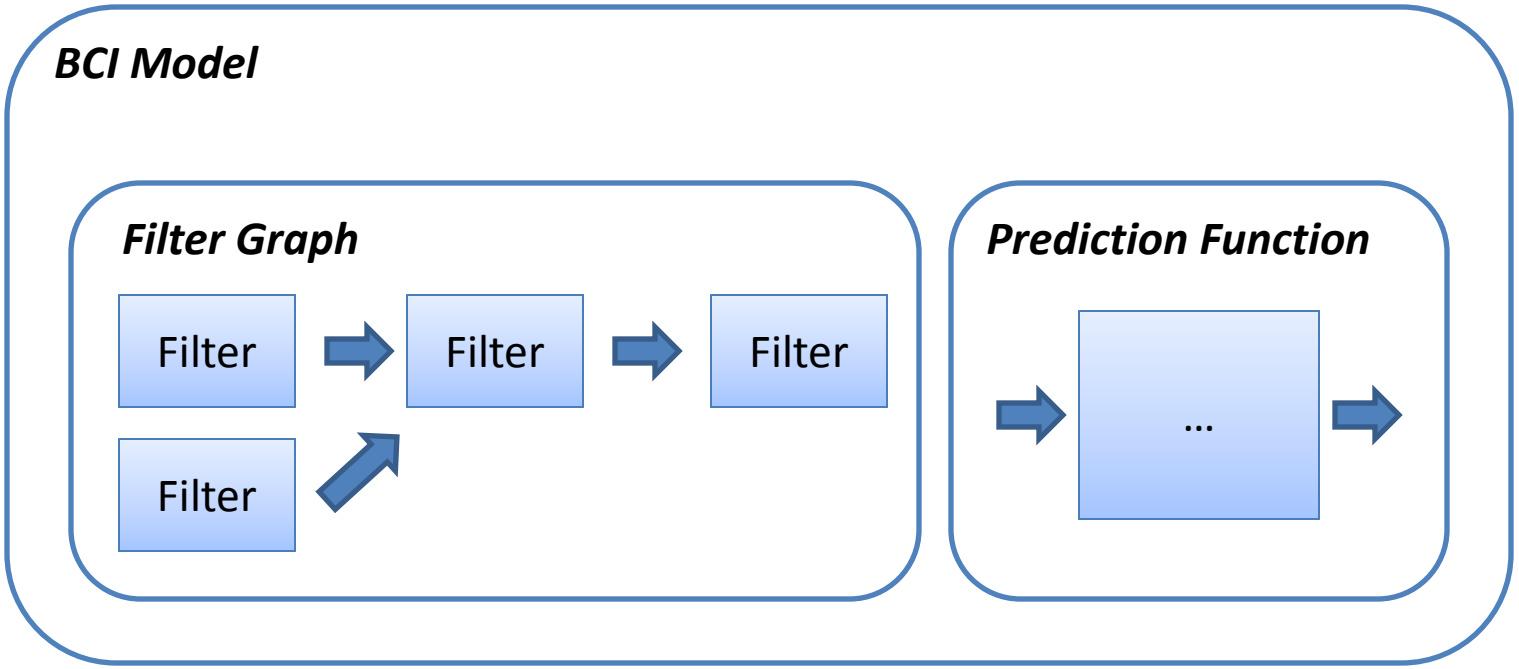
Putting it Together

- For online operation, the overall prediction function consists of the feature extraction followed by the predictive map
- This yield a primitive “excitement detector”



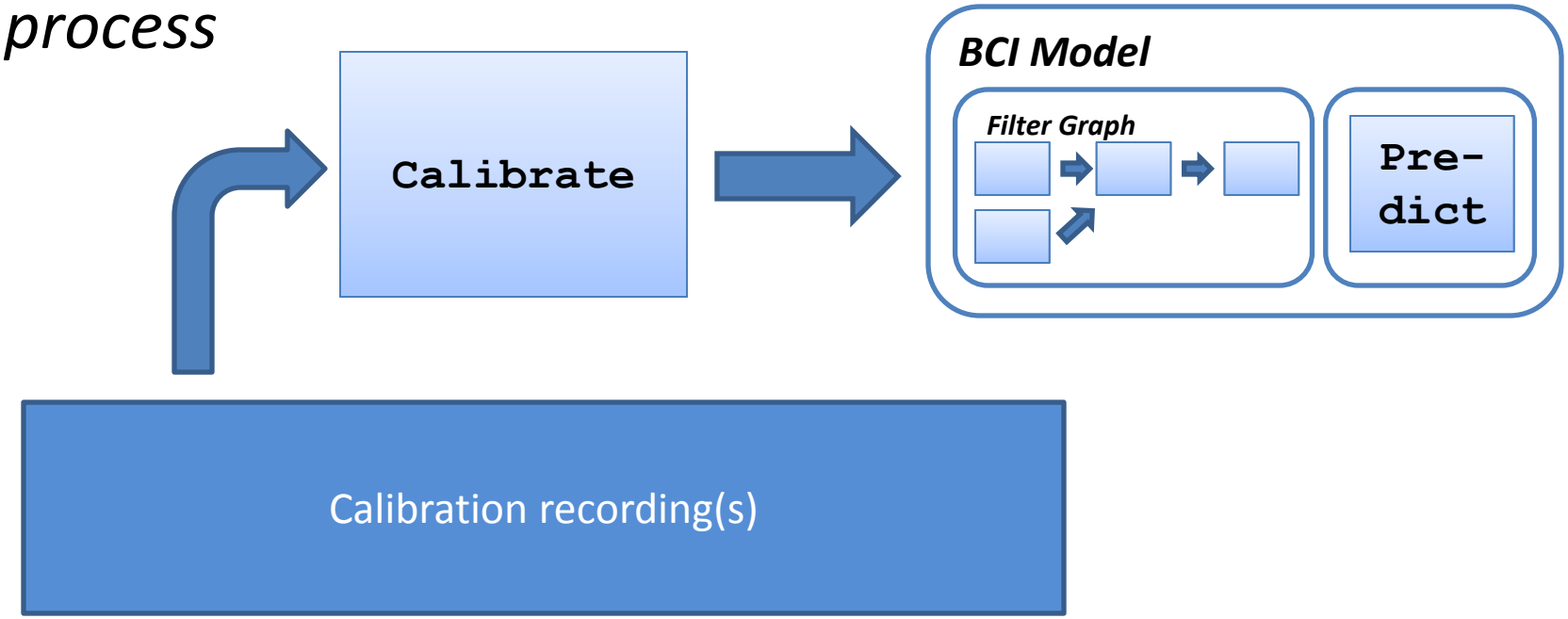
BCI Models

- BCIs are described by "BCI models" that specify both the *filter graph* and the *prediction function* (incl. its parameters)
- These models are the result of the calibration step



“BCI Paradigms”

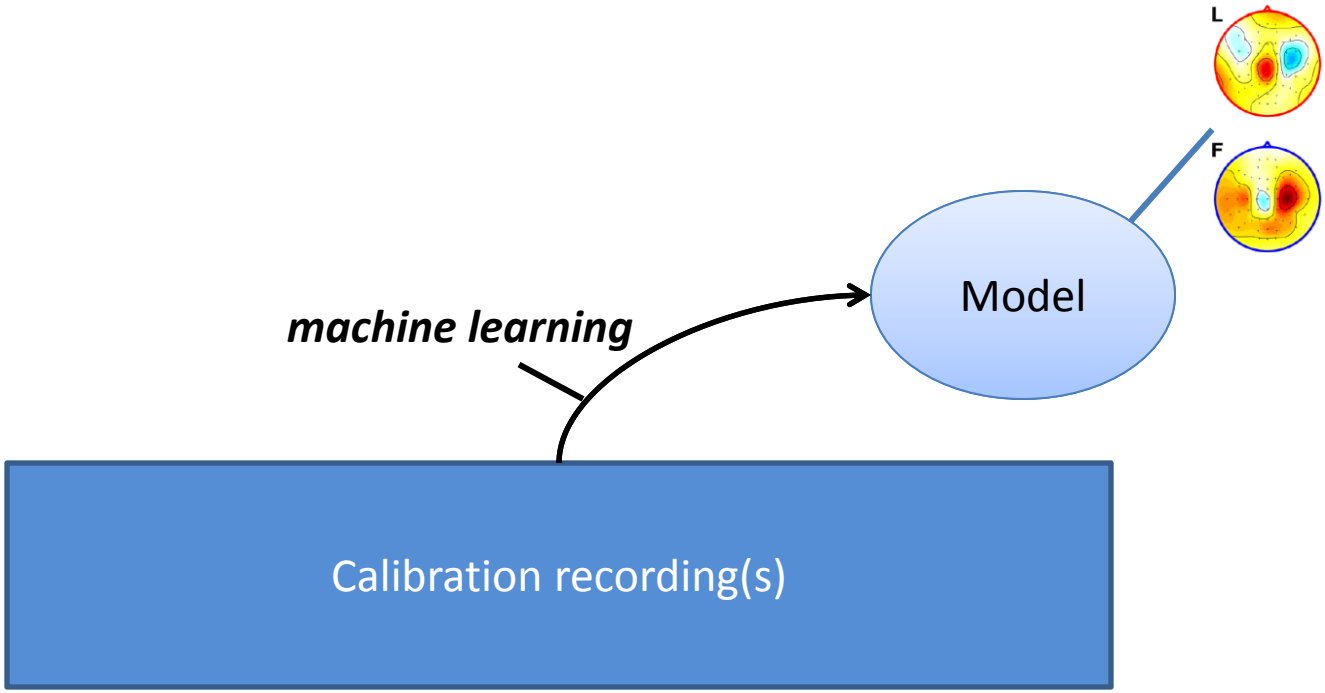
- BCI paradigms are a notion that was first developed in the BCILAB framework
- A BCI paradigm is the *full description and codification of a particular type of calibration and prediction process*




Swartz
Center for
Computational
Neuroscience

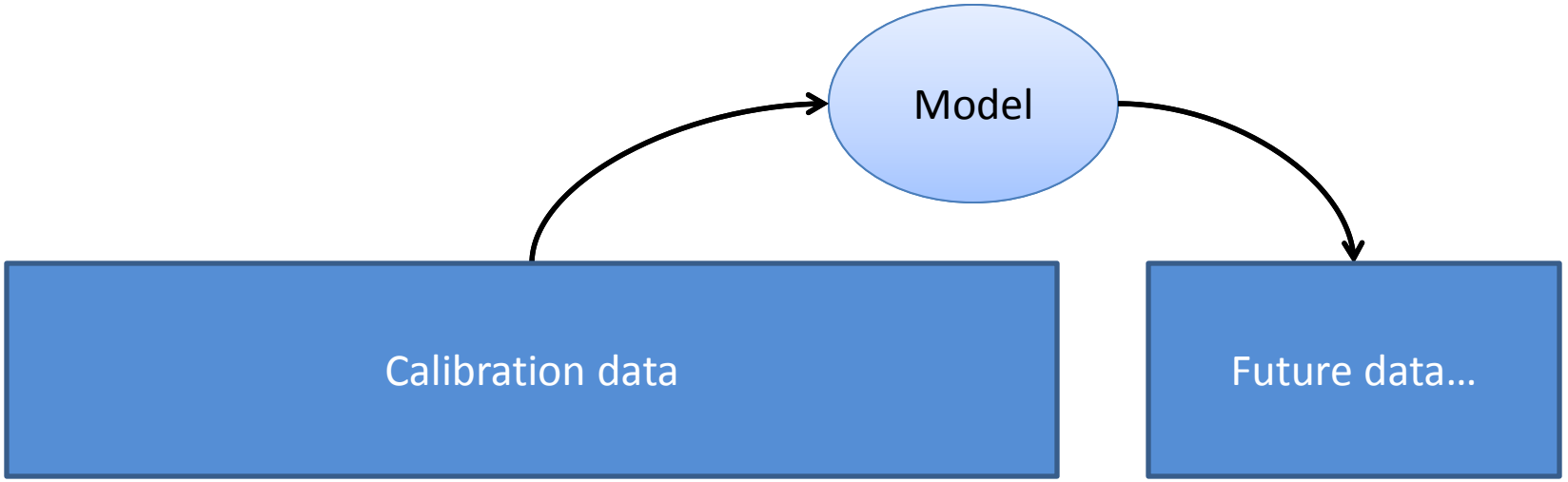
4.4 Performance Evaluation

How to Evaluate a Model?



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



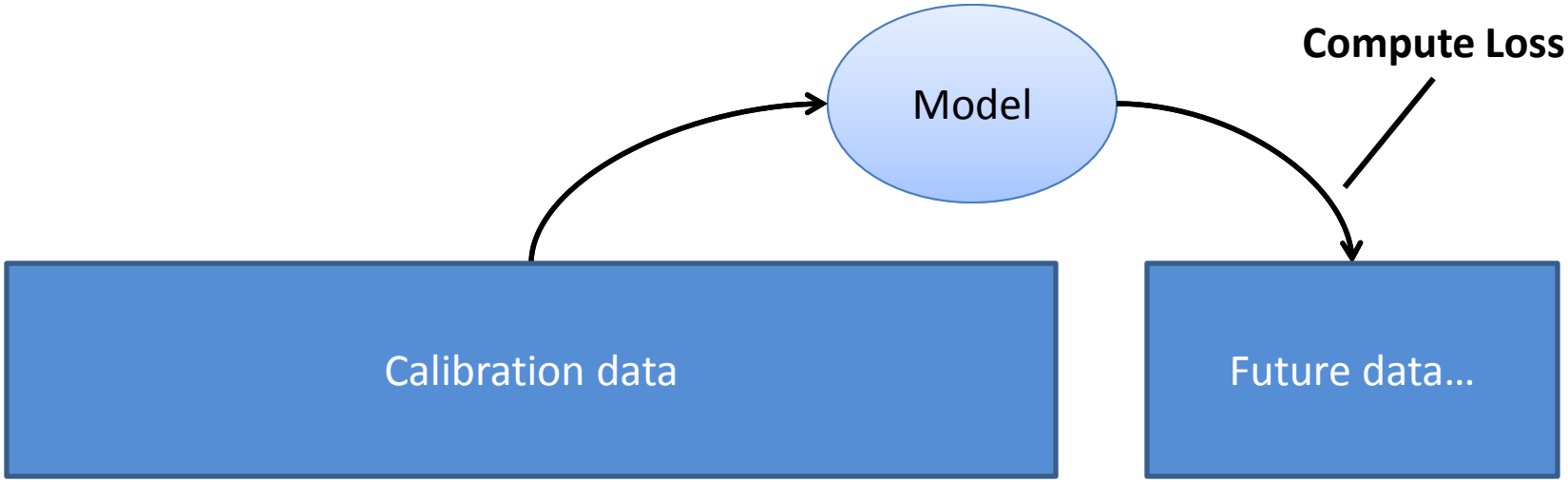
Overall Evaluation Strategies

- Note: Overall *loss estimate* between a vector of predictions $\boldsymbol{p}$ and a vector of targets $\boldsymbol{t}$ is a summary statistic

- **Mean-Square Error:**
  - $L_{MSE}(\boldsymbol{p}, \boldsymbol{t}) = \frac{1}{N}\sum_k (\boldsymbol{p}_k - \boldsymbol{t}_k)^2$
- **Mis-Classification Rate:**
  - $L_{MCR}(\boldsymbol{p}, \boldsymbol{t}) = \frac{1}{N}\sum_k \begin{cases} 1, \boldsymbol{p}_k \neq \boldsymbol{t}_k \\ 0, \boldsymbol{p}_k = \boldsymbol{t}_k \end{cases}$

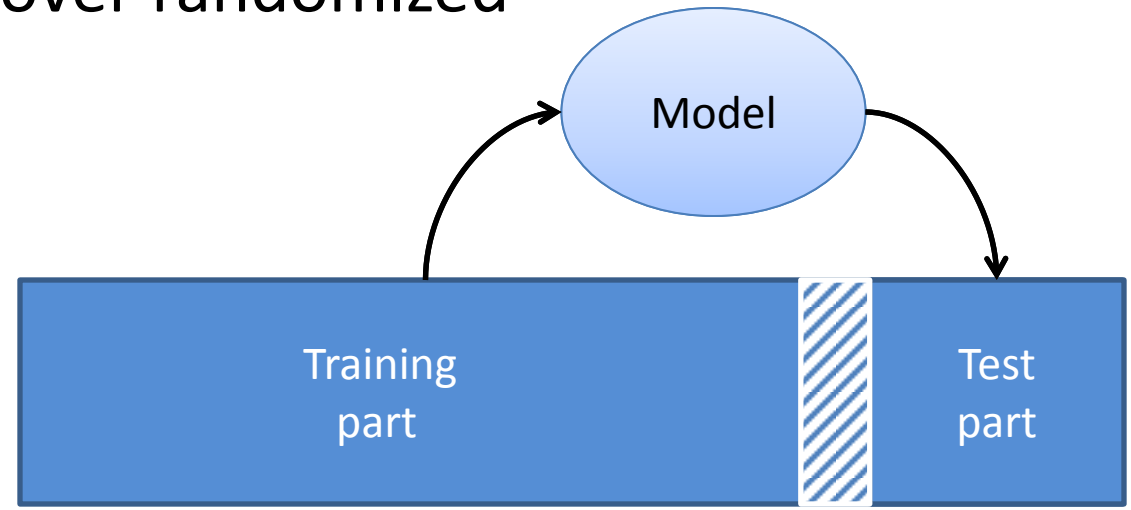
Overall Evaluation Strategies

- **What if there is no second data set?**



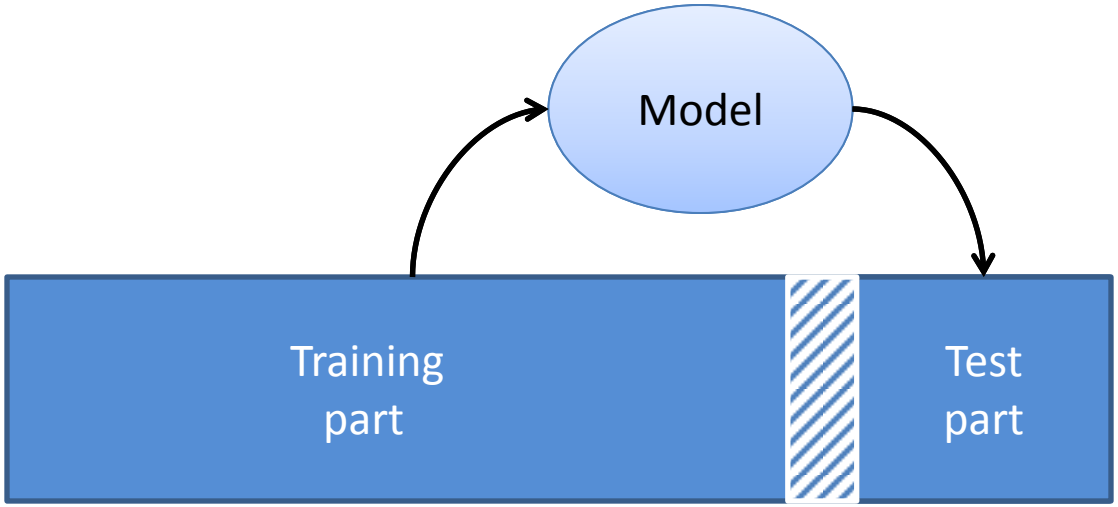
Overall Evaluation Strategies

- **Alternative:** 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



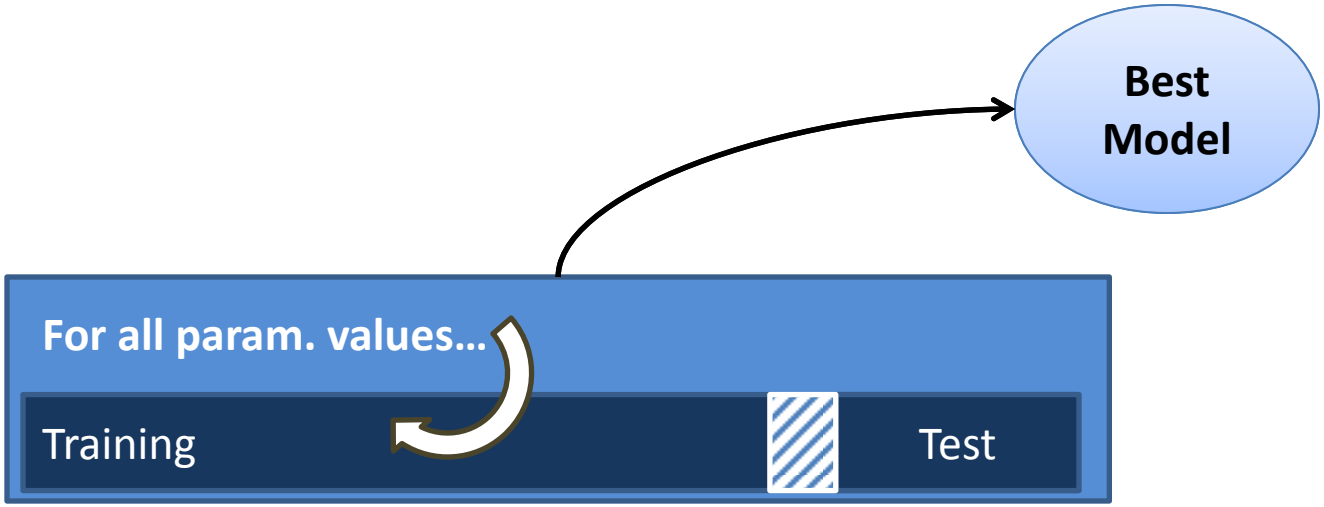
Overall Evaluation Strategies

- **Consideration:** Since neighboring trials are more closely related than training and future online data, *leave a margin of several trials/seconds between training and test*
- Standard splitting schemes: 5x, 10x



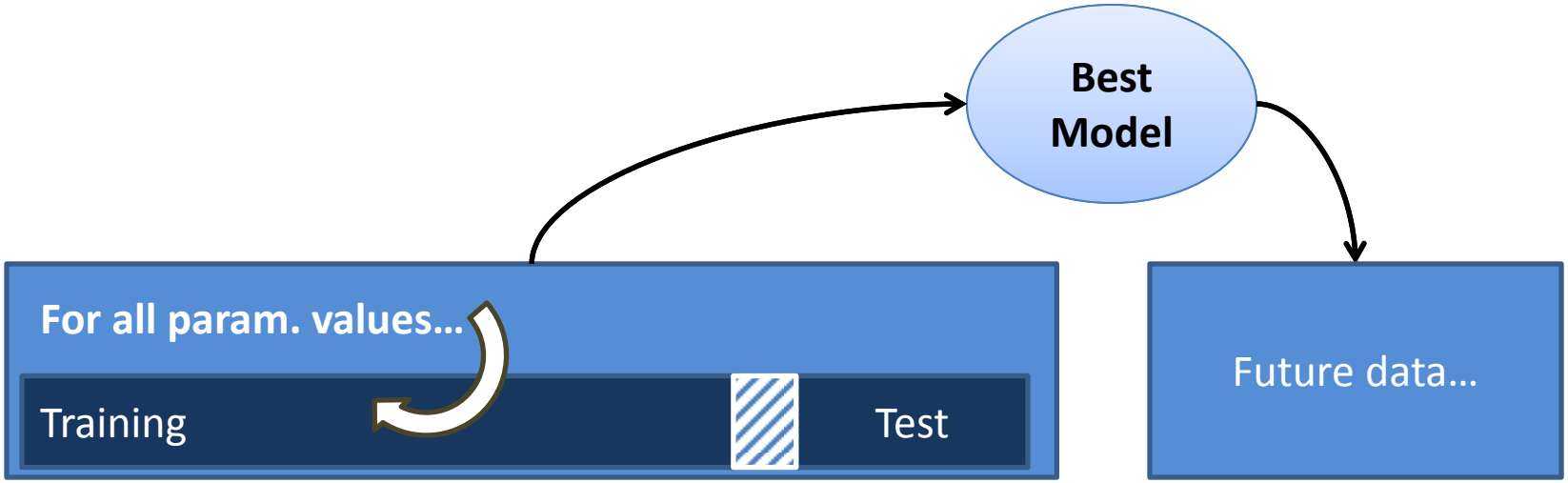
Overall Evaluation Strategies

- **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)



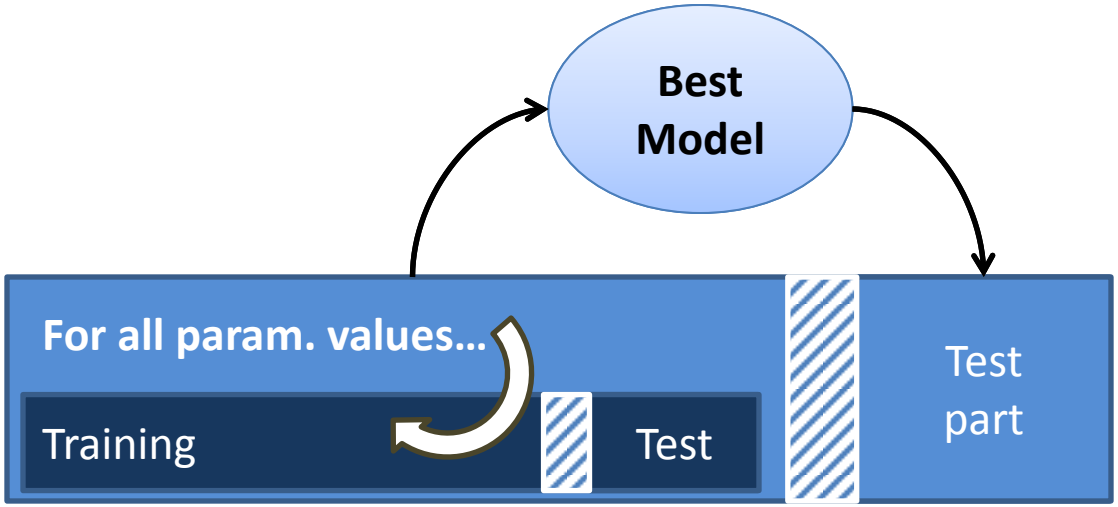
Overall Evaluation Strategies

- **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), 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")




Swartz
Center for
Computational
Neuroscience

L4 Questions?