Acquisition and Analysis of Neuronal Data 2010
BCI – Lecture #12

Carmen Vidaurre

Machine Learning Laboratory, Berlin Institute of Technology

carmen.vidaurre@tu-berlin.de

09-Jul-2010

Today's Topic

Methods:

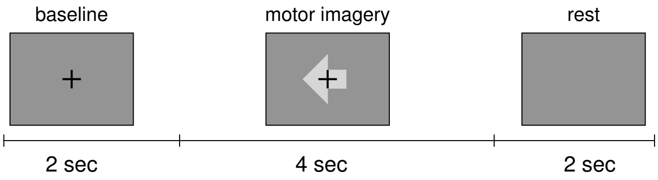
- Adaption of Fisher's Discriminant Classifier.
- In particular, iterative adaption of means and inverse (extended) covariance matrix.

Real world application:

- Classification of motor imagery conditions in a BCI paradigm.
- Update of the classifier to changes occuring during the experimental session.

Experimental Design

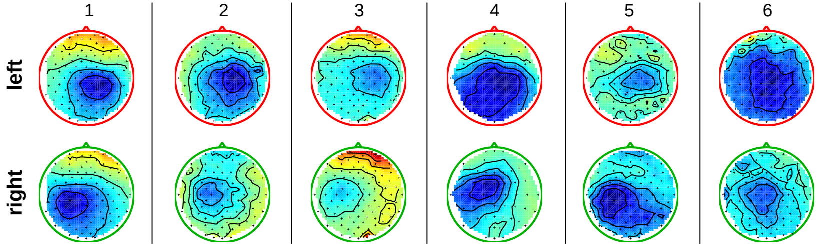
Subject sitting relaxed in a chair with armrests.
Visual cues (arrows) indicate which type of motor *imagery* is to be performed: left hand, right hand, right foot.
Every 15 trials, a break of 15 s is given. In total 105 trials of each motor imagery condition are recorded.
— Pause of several hours —–
Visual cues are provided again.



Note: today's data is artificially modified to increase the difference between the two recordings.

Reminder: Subject-to-Subject Variability

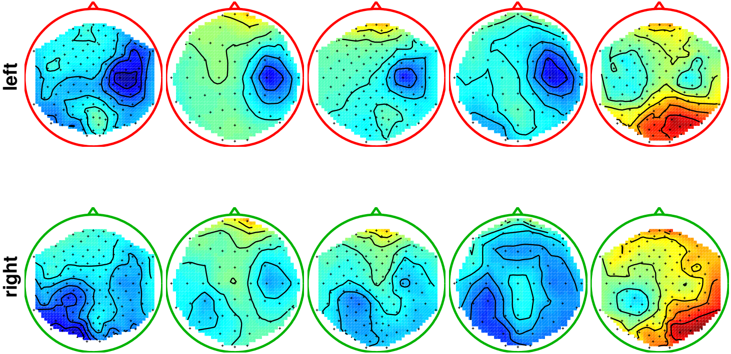
- Experiment: 6 subjects performed left vs. right hand finger tapping.
- Even though the task involves a highly **overlearned motor competence**, the **averaged** brain patterns exhibit a great diversity between subjects:



➤ An optimal system needs adaption for each user.

Reminder: Session-to-Session Variability

- Experiment: **One subject** imagined left vs. right hand movements on different days.
- Even though each ERD map represents an **average** across 140 trials, they exhibit an apparent diversity.

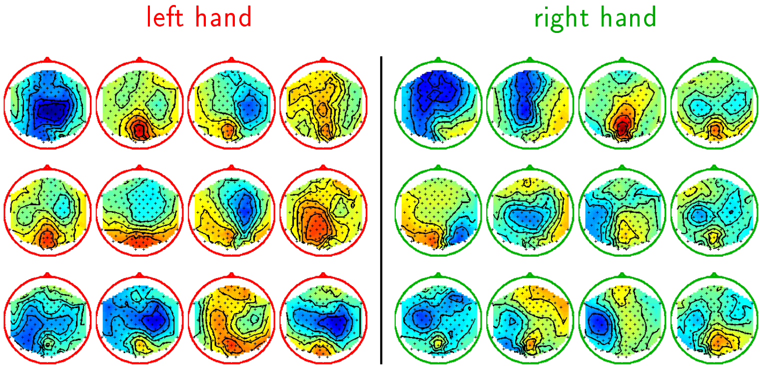


➤ An optimal system needs adaption for (or within?) each session.

Reminder: Trial-to-Trial Variability

In this lesson we will take care of the changes within the session.

- Experiment: One subject imagined left vs. right hand movements.
- Topographies show power in the **alpha band** during trials of 3.5 s.
- They exhibit an extreme diversity, although recorded from **one subject** on **one day**.



Why do we need to adapt?

EEG changes:

- *Class related* short-term changes: performance of different mental tasks.
- *Class related* long-term changes: due to feedback training (learning). Mean of the features.
- *Class unrelated* long-term changes: e.g. fatigue or lack of concentration. Co-Variance of the features.
- Variation of other *noise sources*: e.g. changing impedance of the electrodes.

Reminder: Fisher's Discriminant Analysis

Let $\mathbf{x}_k$ be feature vectors of two conditions (k in $\mathcal{C}_1$ resp. $\mathcal{C}_2$) and define

$$\mu_i = \frac{1}{|\mathcal{C}_i|} \sum_{k \in \mathcal{C}_i} \mathbf{x}_k,$$

$$S_i = \sum_{k \in \mathcal{C}_i} (\mathbf{x}_k - \mu_i)(\mathbf{x}_k - \mu_i)^\top$$

$$\mathbf{w} = (S_1 + S_2)^{-1}(\mu_1 - \mu_2)$$

Note: the vectors are column vectors.

Fisher's Discriminant: today's variation

Today we use a equivalent variation:

$$
\begin{aligned}
S &= \sum_{k \in C_1, C_2} (\mathbf{x}_k - \mu_i)(\mathbf{x}_k - \mu_i)^\top \\
\mathbf{w}' &= S^{-1}(\mu_1 - \mu_2) \\
\mathbf{w}' &= \text{constant} \cdot \mathbf{w}
\end{aligned}
$$

With "some" mathematical effort one can show that the classification result with both variations is the same.

Reminder: FD for Classification

Let $\mathbf{x}_k \in \mathbb{R}^m$ be feature vectors of two classes ($k \in \mathcal{C}_1$ resp. $k \in \mathcal{C}_2$). Then the FD vector $\mathbf{w}$ as defined above separates $\mathbb{R}^m$ in two classes by virtue of the decision function:

$$f : \mathbb{R}^m \to \mathbb{R}; \quad \mathbf{z} \mapsto \begin{cases} -1 & \text{if } \mathbf{w}^\top \mathbf{z} + b < 0 \\ 1 & \text{else} \end{cases}$$

The bias can, e.g., be chosen as $b = -\mathbf{w}^\top(\mu_1 + \mu_2)/2$.

To estimate the bias in today's variation, we use the pooled mean instead of the average of the class means:

$$\mu = \frac{1}{N} \cdot \sum_{k \in C_1, C_2} \mathbf{x}_k$$

Note: FDA is equivalent to Linear Discriminant Analysis.

Mean estimation

Mean estimation of a stochastic (random) process $x(t)$: at t, $x(t)$ is observed, with N observations. The mean value estimate μ_x is

$$\text{mean}(x) = \mu_x = \frac{1}{N}\sum_{t=1}^{N} x(t) = E\langle x(t)\rangle$$

For a time-varying estimation, we need a (sliding) window:

$$\mu_x(t) = \frac{1}{\sum_{i=0}^{n-1} w_i}\sum_{i=0}^{n-1} w_i \cdot x(t-i) \qquad t \geq n$$

where n is the width of the window and w_i are the weighting factors.

Mean Estimation

Commonly: rectangular window, $w_i = 1$

$$\mu_x(t) = \frac{1}{n} \sum_{i=0}^{n-1} x(t-i) \qquad t \geq n$$

Recursive formula for the rectangular window approach:

$$\mu_x(t) = \mu_x(t-1) + \frac{1}{n} \cdot (x(t) - x(t-n)) \qquad t \geq n$$

Need to keep the n past sample values in memory and an initial $\mu_x(0)$.

Mean Estimation

Next formula needs no memory of past values of x:

$$\mu_x(t) = (1 - UC) \cdot \mu_x(t-1) + UC \cdot x(t) \qquad t \geq 1 \quad (1)$$

UC= update coefficient of an exponential weighting window. One needs an initial estimate $\mu_x(0)$.

$$w_i = UC \cdot (1 - UC)^i \qquad i \in \{0, \ldots, n-1\}$$

Mean Estimation

Table: Computational effort of mean estimators (per dimension and time step).

Method	Memory effort	Computational effort
stationary	O(1)	O(1)
weighted sliding window	O(n)	O(n)
rectangular sliding window	O(n)	O(n)
recursive (only for rectangular)	O(n)	O(1)
adaptive (exponential window)	O(1)	O(1)

Note: if the window length and UC are properly chosen, a similar characteristic can be obtained.

Variance Estimation

The overall variance σ_x^2 of $x(t)$ can be estimated with

$$\text{var}(x) = \sigma_x^2 = \frac{1}{N}\sum_{t=1}^{N}(x(t) - \mu_x)^2 = E\langle(x(t) - \mu_x)^2\rangle$$

An adaptive estimator for the variance is this one

$$\sigma_x(t)^2 = (1 - UC)\cdot\sigma_x(t-1)^2 + UC\cdot(x(t) - \mu_x(t))^2 \qquad t \geq 1 \tag{2}$$

One needs the initial $\sigma_x(0)^2$ and $\mu_x(1)$.

Note: this variance estimator is biased. In order to obtain an unbiased estimator, one must multiply the result by $N/(N-1)$.

Variance Estimation

$$\sigma_x^2 = \frac{1}{N} \sum_{t=1}^{N} x(t)^2 - \mu_x^2$$

Alternatively, one can also compute the mean square

$$MSQ_x(t) = (1 - UC) \cdot MSQ_x(t-1) + UC \cdot x(t)^2 \tag{3}$$

One needs $MSQ_x(0)$ as initial condition.
The variance can be obtained by

$$\sigma_x(t)^2 = MSQ_x(t) - \mu_x(t)^2 \tag{4}$$

Variance-Covariance Estimation

Remember FDA, the covariances between the various dimensions are of interest. The (stationary) variance-covariance matrix:

$$\text{cov}(x) = \boldsymbol{\Sigma}_x = \frac{1}{N}\sum_{t=1}^{N}(\boldsymbol{x}(t) - \boldsymbol{\mu}_x) \cdot (\boldsymbol{x}(t) - \boldsymbol{\mu}_x)^\top$$

Variances: diagonal elements. Off-diagonal, element $S_{i,j}$ covariance between the i-th and j-th element.
An adaptive estimator of the covariance matrix:

$$\boldsymbol{\Sigma}_x(t) = (1-UC)\cdot\boldsymbol{\Sigma}_x(t-1)+UC\cdot(\boldsymbol{x}(t)-\boldsymbol{\mu}_x(t))\cdot(\boldsymbol{x}(t)-\boldsymbol{\mu}_x(t))^\top$$

t is the sample time, UC is the update coefficient. Necessary $\boldsymbol{\Sigma}_x(0)$ and $\boldsymbol{\mu}(1)$.

Variance-Covariance Estimation

Estimating the covariance implies estimating mean values as well. To avoid this we define the *extended covariance matrix* (ECM) $\boldsymbol{E}$ as

$$ECM(x) = \boldsymbol{E}_x = \sum_{t=1}^{N_x} [1; \boldsymbol{x}(t)] \cdot [1; \boldsymbol{x}(t)]^\top =$$

$$= N_x \cdot \left[\begin{array}{c|c} 1 & \boldsymbol{\mu}_x^\top \\ \hline \boldsymbol{\mu}_{\boldsymbol{x}} & \boldsymbol{\Sigma}_{\boldsymbol{x}} + \boldsymbol{\mu}_x \boldsymbol{\mu}_x^\top \end{array} \right] \tag{5}$$

Remember to divide through N_x.

Variance-Covariance Estimation

Adaptive ECM estimator:

$$\boldsymbol{E}_x(t) = (1-UC)\cdot\boldsymbol{E}_x(t-1)+UC\cdot[1;\boldsymbol{x}(t)]\cdot[1;\boldsymbol{x}(t)]^\top \qquad t \geq 1 \tag{6}$$

t is the sample time, UC is the update coefficient. Necessary $E_x(0)$.
For the exercise: remember to normalize initial conditions!!

Adaptive Inverse Covariance Matrix Estimation

FDA needs the computation of $\mathbf{\Sigma}^{-1}$. We can extract $\mathbf{\Sigma}$ from the ECM (divided through N_x) and compute its inverse

$$\mathbf{\Sigma}^{-1} = \text{inv}(\text{ECM}(2\text{:end},2\text{:end}) - \text{ECM}(2\text{:end},1) \cdot \text{ECM}(1,2\text{:end}))$$

Needs an explicit matrix inversion -> **computational effort**.

Adaptive Inverse Covariance Matrix Estimation

But $\mathbf{\Sigma}^{-1}$ can be obtained **without** an explicit matrix inversion.
Let's see first what the inverse of ECM is:

$iECM = \boldsymbol{E}^{-1} = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]^{-1}$ with the inverse of a block matrix

$$\left[\begin{array}{c|c} A^{-1} + A^{-1}BS^{-1}CA^{-1} & -A^{-1}BS^{-1} \\ \hline -S^{-1}CA^{-1} & S^{-1} \end{array}\right]$$
$$= \boxed{\left[\begin{array}{c|c} 1 + \boldsymbol{\mu}_x^\top \mathbf{\Sigma}_x^{-1} \boldsymbol{\mu}_x & -\boldsymbol{\mu}_x^\top \mathbf{\Sigma}_x^{-\top} \\ \hline -\mathbf{\Sigma}_x^{-1} \boldsymbol{\mu}_x^\top & \mathbf{\Sigma}_x^{-1} \end{array}\right]} \tag{7}$$

with $S = D - CA^{-1}B$

Adaptive Inverse Covariance Matrix Estimation

Now we obtain the adaptively estimated $iECM = \boldsymbol{E}^{-1}$.

Applying the matrix inversion lemma to equation (6)
$\boldsymbol{A} = (\boldsymbol{B} + \boldsymbol{UDV})$
The inverse is:

$$\begin{aligned} \boldsymbol{A}^{-1} &= (\boldsymbol{B} + \boldsymbol{UDV})^{-1} = \\ &= \boldsymbol{B}^{-1} - \boldsymbol{B}^{-1}\boldsymbol{U}\left(\boldsymbol{D}^{-1} + \boldsymbol{VB}^{-1}\boldsymbol{U}\right)^{-1}\boldsymbol{VB}^{-1} \end{aligned} \quad (8)$$

Adaptive Inverse Covariance Matrix Estimation

We identify the matrices in (8) as follows:

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{E}(t)\\ \boldsymbol{B} &= (1-UC)\cdot\boldsymbol{E}(t-1)\\ \boldsymbol{U} &= \boldsymbol{V}^\top = \boldsymbol{x}(t)\\ \boldsymbol{D} &= UC\end{aligned}$$

UC: update coefficient, $\boldsymbol{x}(t)$: the current sample vector.
Substituting in Eq. 8 the adaptive inverse covariance matrix is:

$$\boldsymbol{E}(t)^{-1} = \frac{\left(\boldsymbol{E}(t-1)^{-1} - \frac{UC}{(1-UC)+UC\cdot\boldsymbol{x}(t)^\top\cdot\boldsymbol{v}}\cdot\boldsymbol{v}\cdot\boldsymbol{v}^\top\right)}{1-UC} \tag{9}$$

with $\boldsymbol{v} = \boldsymbol{E}(t-1)^{-1}\cdot\boldsymbol{x}(t)$ and $\boldsymbol{x}(t)^\top\cdot\boldsymbol{v}$ is a scalar. You need an estimate of $\boldsymbol{E}(0)^{-1}$.

Adaptive Inverse Covariance Matrix Estimation

iECM can become asymmetric and singular. Avoid it like this:

$$\boldsymbol{E}(t)^{-1} = \frac{\left(\boldsymbol{E}(t)^{-1} + \boldsymbol{E}(t)^{-\top}\right)}{2} \tag{10}$$

Now, the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ can be obtained by estimating the extended covariance matrix and decomposing it according to equation (7).

$\boldsymbol{\Sigma}^{-1} =$ iECM(2:end,2:end)

Adaptive Inverse Covariance Matrix Estimation

For the usual covariance we follow the same procedure:

$$\mathbf{\Sigma}(t)^{-1} = \frac{\left(\mathbf{\Sigma}(t-1)^{-1} - \frac{UC}{(1-UC)+UC\cdot(\boldsymbol{x}(t)-\boldsymbol{\mu}(t))^{\top}\cdot\boldsymbol{v}} \cdot \boldsymbol{v} \cdot \boldsymbol{v}^{\top}\right)}{1-UC} \tag{11}$$

with $\boldsymbol{v} = \mathbf{\Sigma}(t-1)^{-1} \cdot (\boldsymbol{x}(t) - \boldsymbol{\mu}(t))$ and $(\boldsymbol{x}(t) - \boldsymbol{\mu}(t))^{\top} \cdot \boldsymbol{v}$ is a scalar. You need an estimate of $\mathbf{\Sigma}(0)^{-1}$ and $\boldsymbol{\mu}(1)$. You need to reinforce symmetry as well.

Reminder: Training CSP-based Classification

- Determine most discriminative frequency band,
- band-pass filter EEG in that band,
- extract single trials using the time interval in which ERD/ERS is expected,
- calculate and select CSP filters,
- and apply them to EEG single trials,
- calculate the log variance within trials.

To obtain a low dimensional feature vector per trial.

—(The data of the exercise is pre-processed until here)—

- Train a linear classifier like Fisher's Discriminant on the features (w/o shrinkage).

Updating and applying the Classifier

Trial by trial:

- Compute features: filter in time (frequency band) and space (CSP filters), compute variance and log -> already pre-processed!
- Update the trained classifier using the current test feature vector (note that you do not use class labels).
- Apply the new classifier in the next test feature vector.

We need some delay! Only apply the classifier to the features of the next trial.