The minimum you need to know to classify electro-physiological brain signals

Jason Farquhar
Dept. Cognitive Artificial Intelligence (CAI)
Donders Center for Cognition
<J.Farquhar@donders.ru.nl>

Learning Goals

- Understand:
  - How the on-line real-time nature of BCI necessitates slightly different methods than off-line analysis
  - The purpose of each of the steps needed to process electropysiological data
  - The trade-offs inherent in skipping steps and/or using different methods at each step
- Know what the following terms mean:
  - Detrending, bad-channel identification, re-referencing, spectral-filtering, bad-trial identification, classifier-training

Today's Plan

- Overview : the BCI cycle
- Overview : 2 common BCI signals, ERPs and ERSPs
- Overview : Properties of electrophysiolical data and artifacts
- A simple BCI signal analysis pipeline:
  - Detrending / High-pass filtering – What? Why? How?
  - bad-channel removal – What? Why? How?
  - re-referencing/spatial-filtering – What? Why? How?
  - Spectral-filtering – What? Why? How?
  - Bad-trial identification/removal – What? Why? How?
  - ERSP feature-extraction & selection – What? Why? How?
  - Classifier training – What? Why? How?
- Summary

BCI signal analysis pipeline

1)Detrending

2)Bad-channel identify and remove

3)Re-referencing/Spatial Filtering

ERP

4) Spectral Filtering

5) Bad-trial ID & remove

ERSP

4) Feature extraction

5) Feature selection

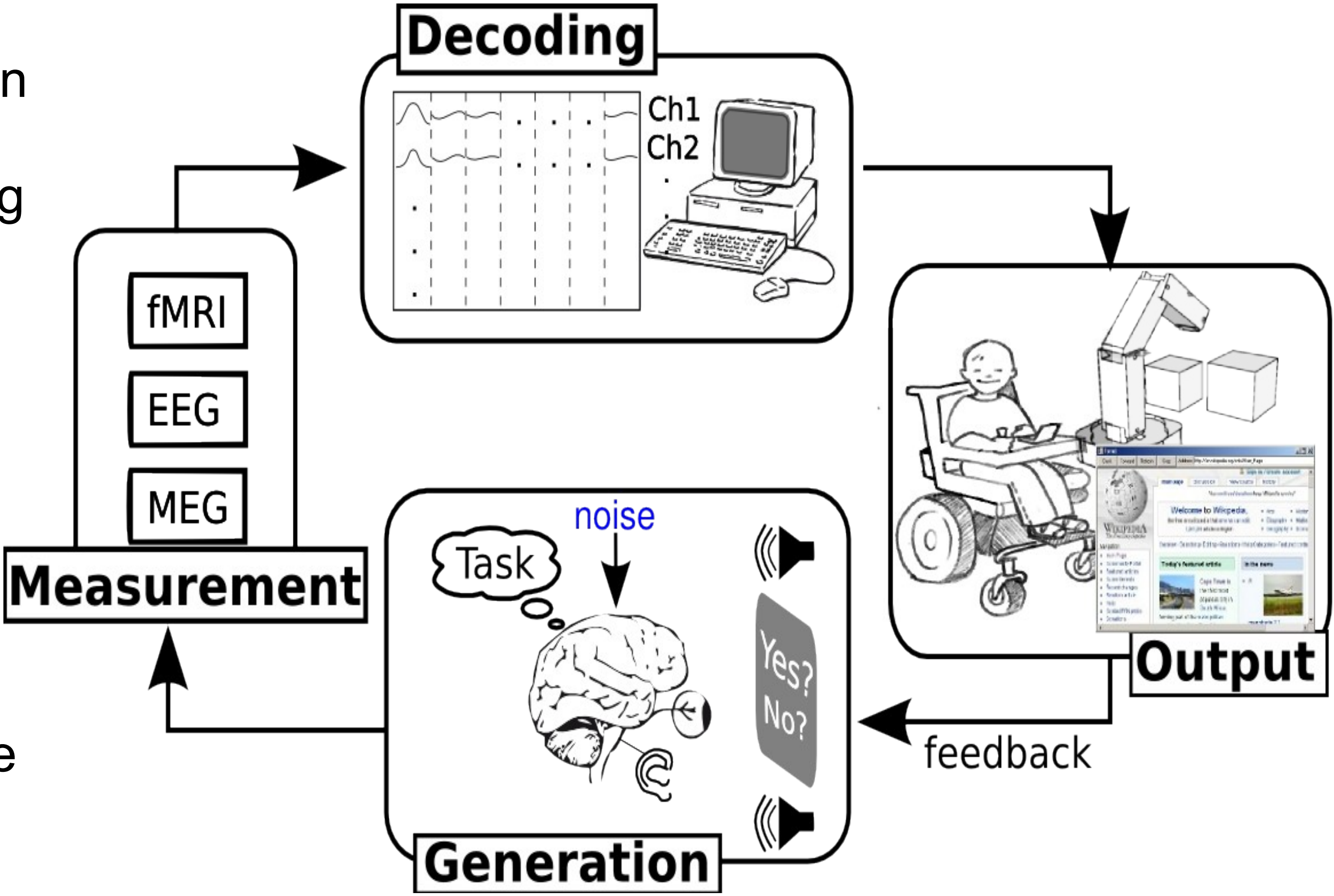
6) Bad-trial ID & rm

5/6) Classifier Training

The BCI Cycle

Fundamentally, BCI is a simple(?) engineering problem;

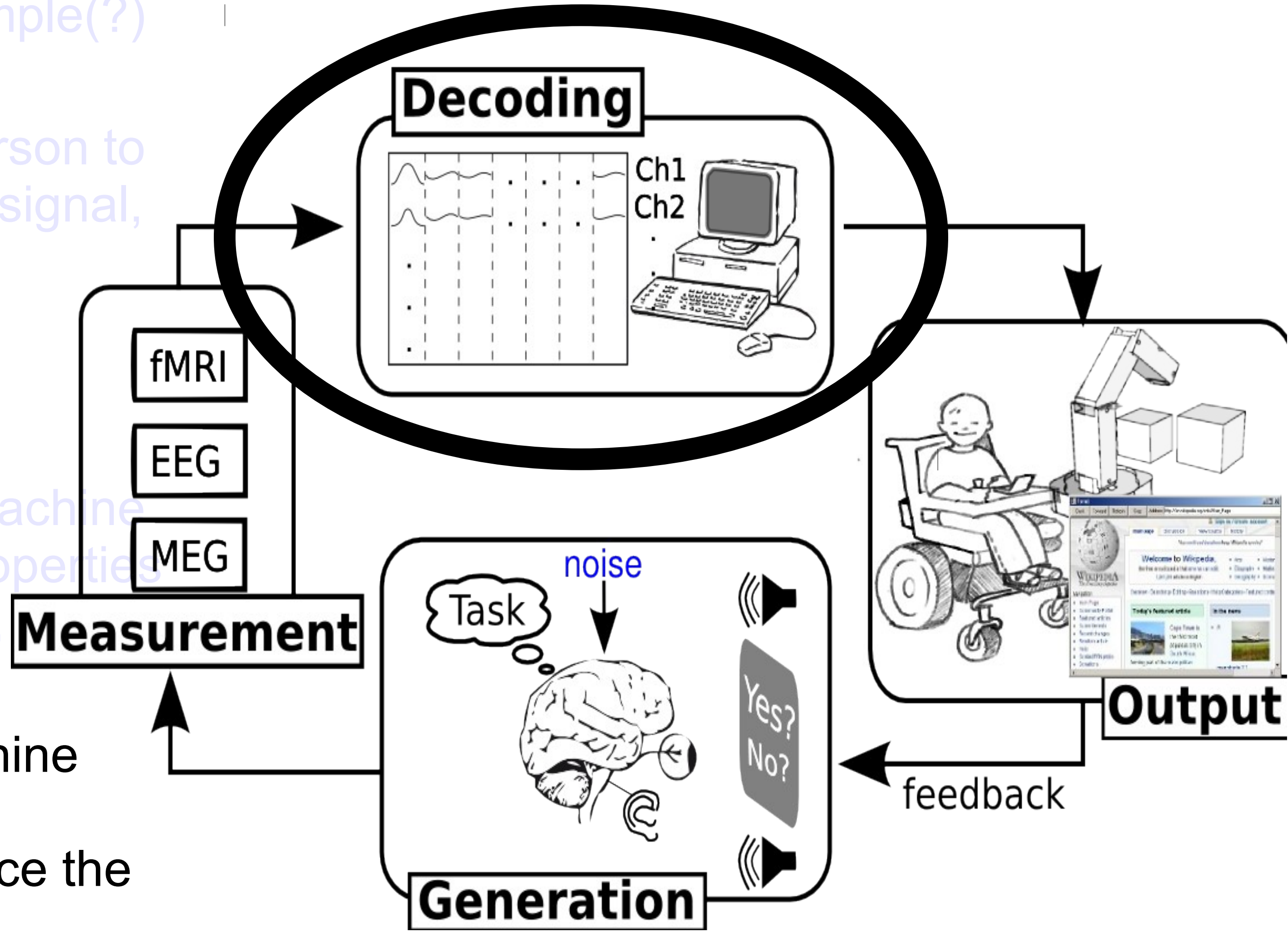
1) Generation: Get the person to produce a strong brain signal, either by performing an explicit mental-task, or through normal mental processes
2) Measurement: Build a machine able to measure the properties of their brain, e.g. EEG, MEG, fMRI
3) Decoding: Build a machine able to decode the measurements to deduce the users mental state
4) Output: Communicate the mental-state to the outside world.



The BCI Cycle

Fundamentally, BCI is a simple(?) engineering problem;

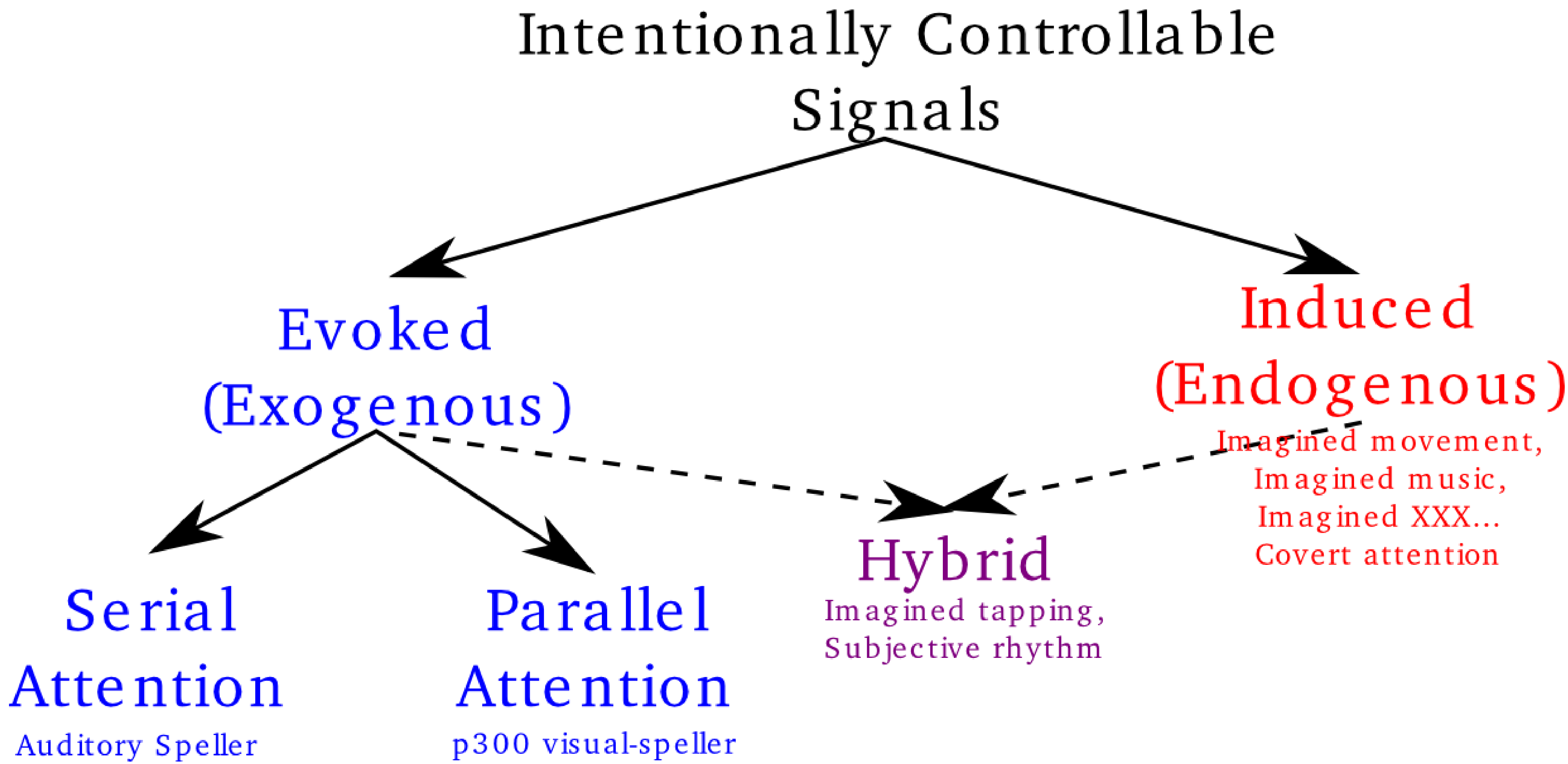
1) Generation: Get the person to produce a strong brain signal, either by performing an explicit mental-task, or through normal mental processes
2) Measurment: Build a machine able to measure the properties of their brain, e.g. EEG, fMRI
3) Decoding: Build a machine able to decode the measurements to deduce the users mental state
4) Output: Communicate the mental-state to the outside world.



2-Types BCI signal : ERP, ERSP

- Event Related Potentia (ERP)
  - or Evoked response BCIs
- Event Related Spectral Potential (ERSP)
  - Or Induced response BCIs

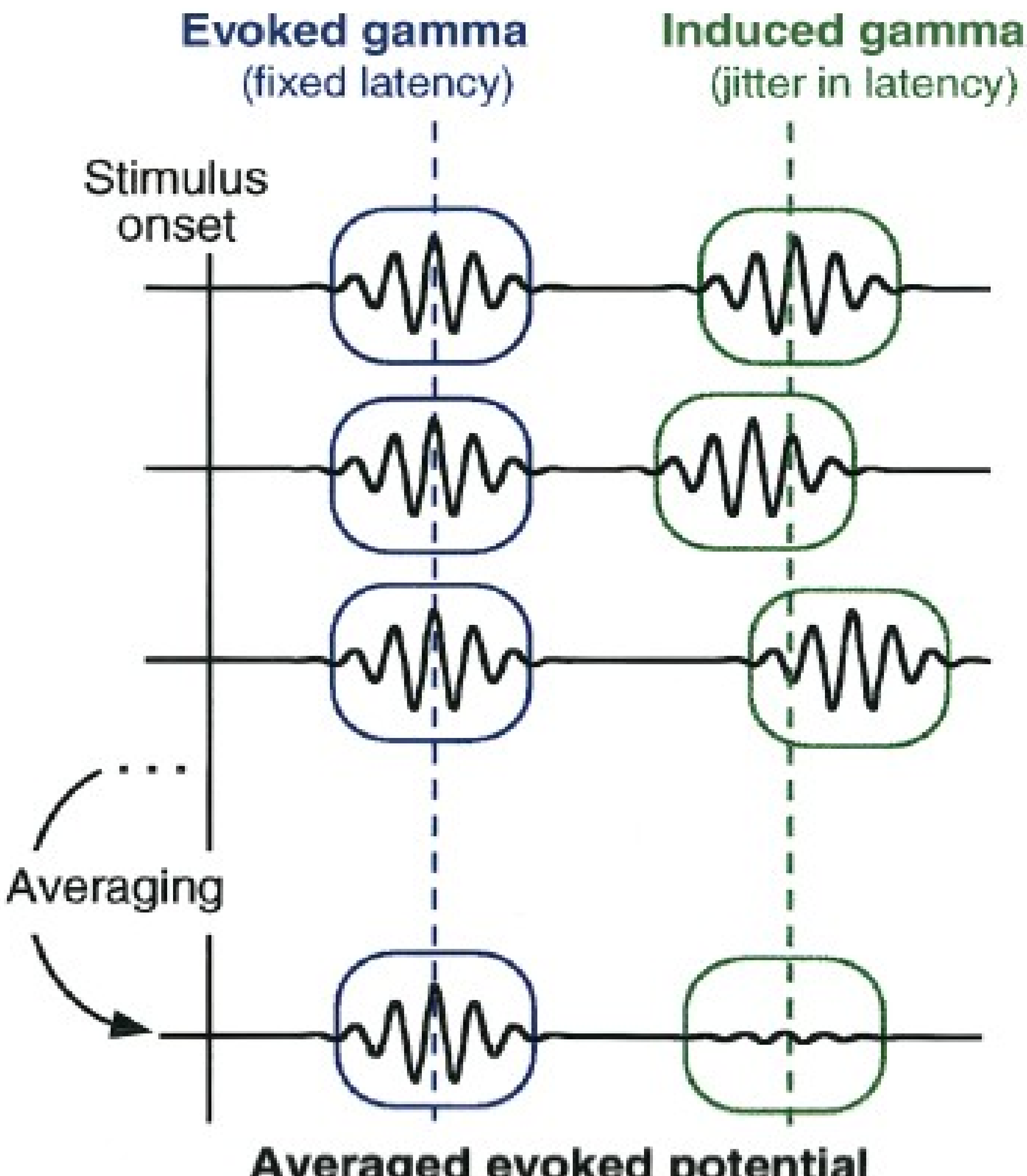
2-Types BCI signal : ERP, ERSP



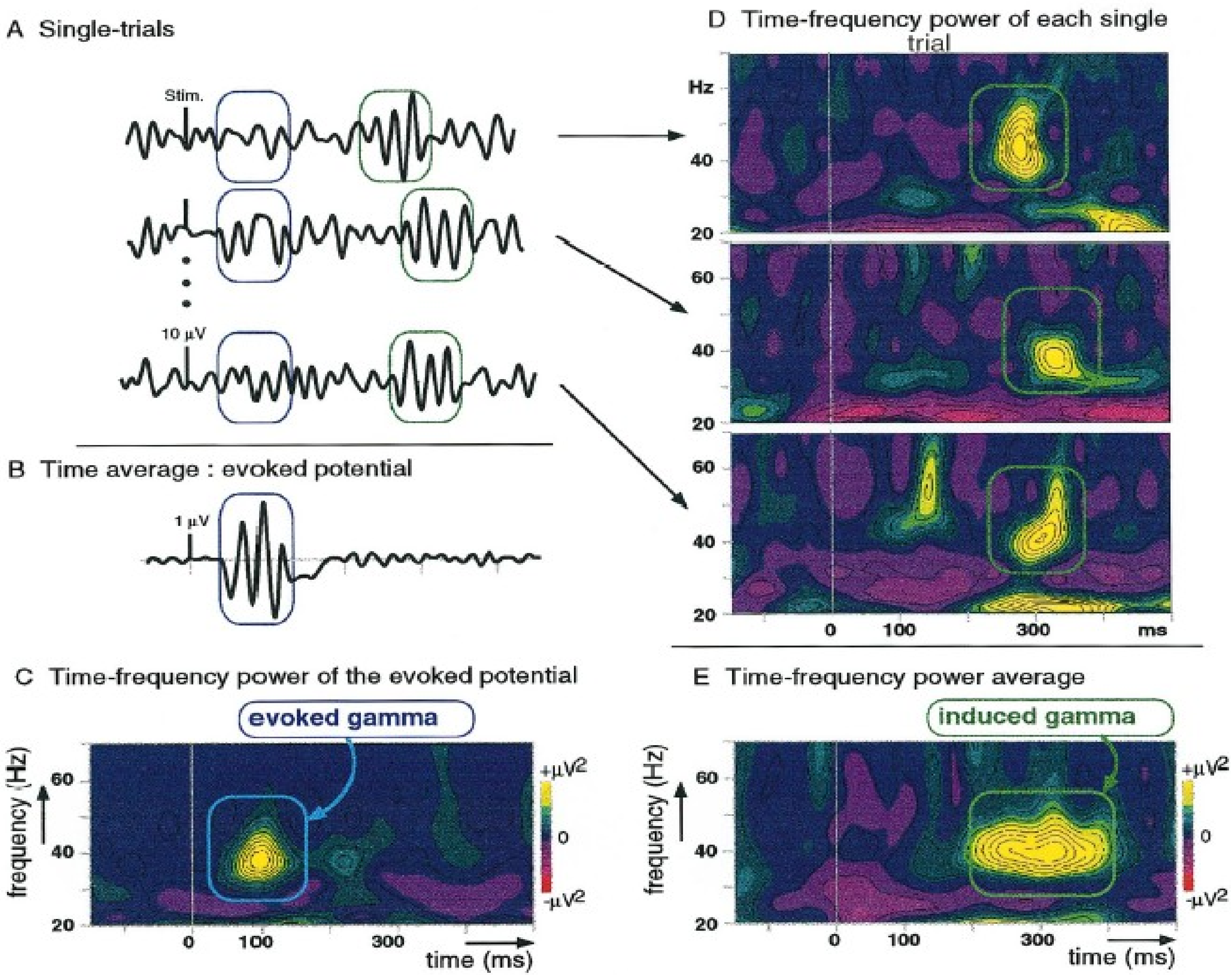
Evoked vs. Induced Responses

- Evoked Respones
  - Brain response to externally generated stimulus
  - Time-locked response (ERP)
  - Examples: P300, MMN, N200 etc.
- Induced Responses
  - Not time-locked response
  - Change in power at a particular frequency
  - External stimulus not-necessary
  - Examples: mu-ERD/S, Alpha changes, Gamma

Evoked vs. Induced Responses

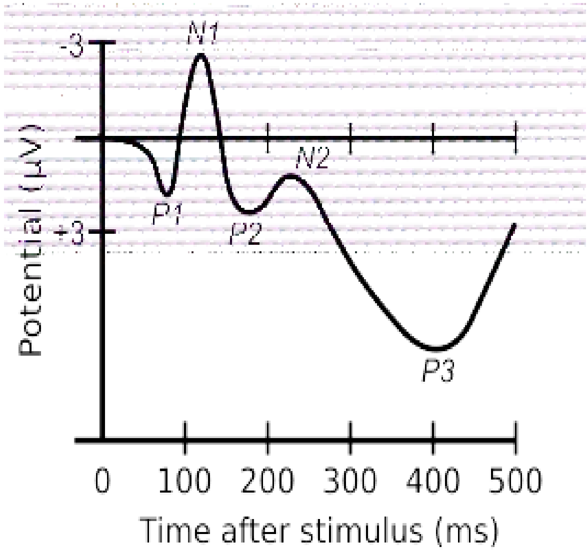


Evoked vs. Induced Response



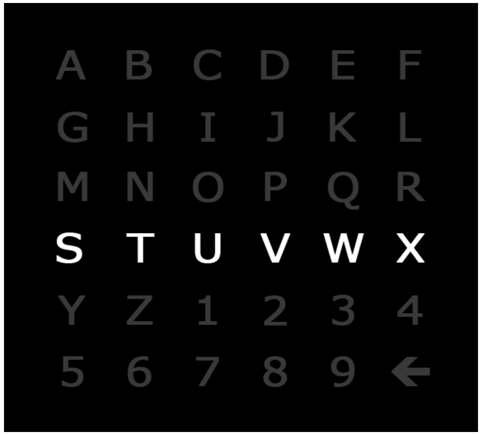
ERP BCIs

- ERP – time-locked signal generated in response to stimulus
- Location of response depends on modality of stimulus: visual, tactile, auditory
- Can depend on more than raw stimulus properties, e.g.
  - Syntatic processing, semantic processing, semantic updating, error-processing, surprise, etc.
- P300 – 'oddball' response
  - Evoked by: unusual stimulus, absence of stimulus
  - Influenced by: uncertainty resolved by event, importance of the stimulus
  - P300a – stimulus response
  - P300b – cognitive response



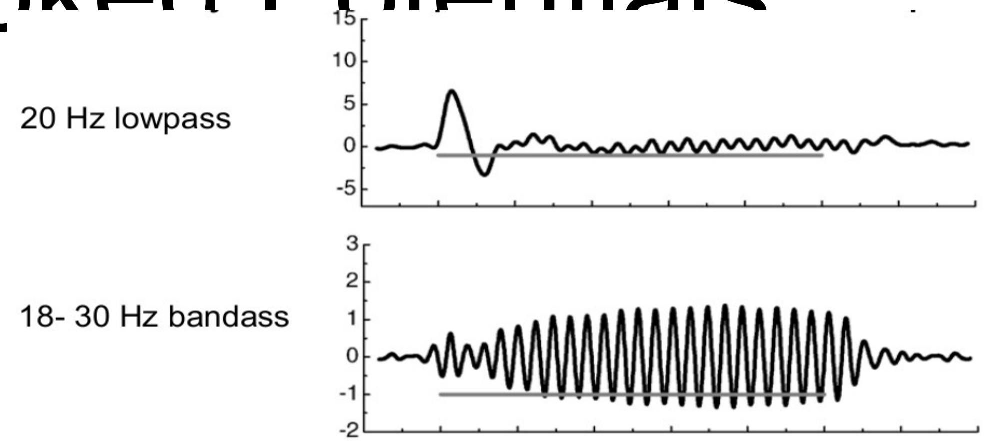
ERP BCIs (2)

- P300 - ERP applications
  - Cortically Coupled Computer Vision (CCCV/RSVP), Brain-fingerprinting
- BCI spelling (Farwell&Donchin) matrix speller
  - Parallel selective attention task
  - Each symbol has unique sequence of 'odd-ball' events
- Overt vs. covert attention
  - Most matrix speller performance based on eye-pointing **but** still works without it..



Special Case: Steady State Evoked Potentials

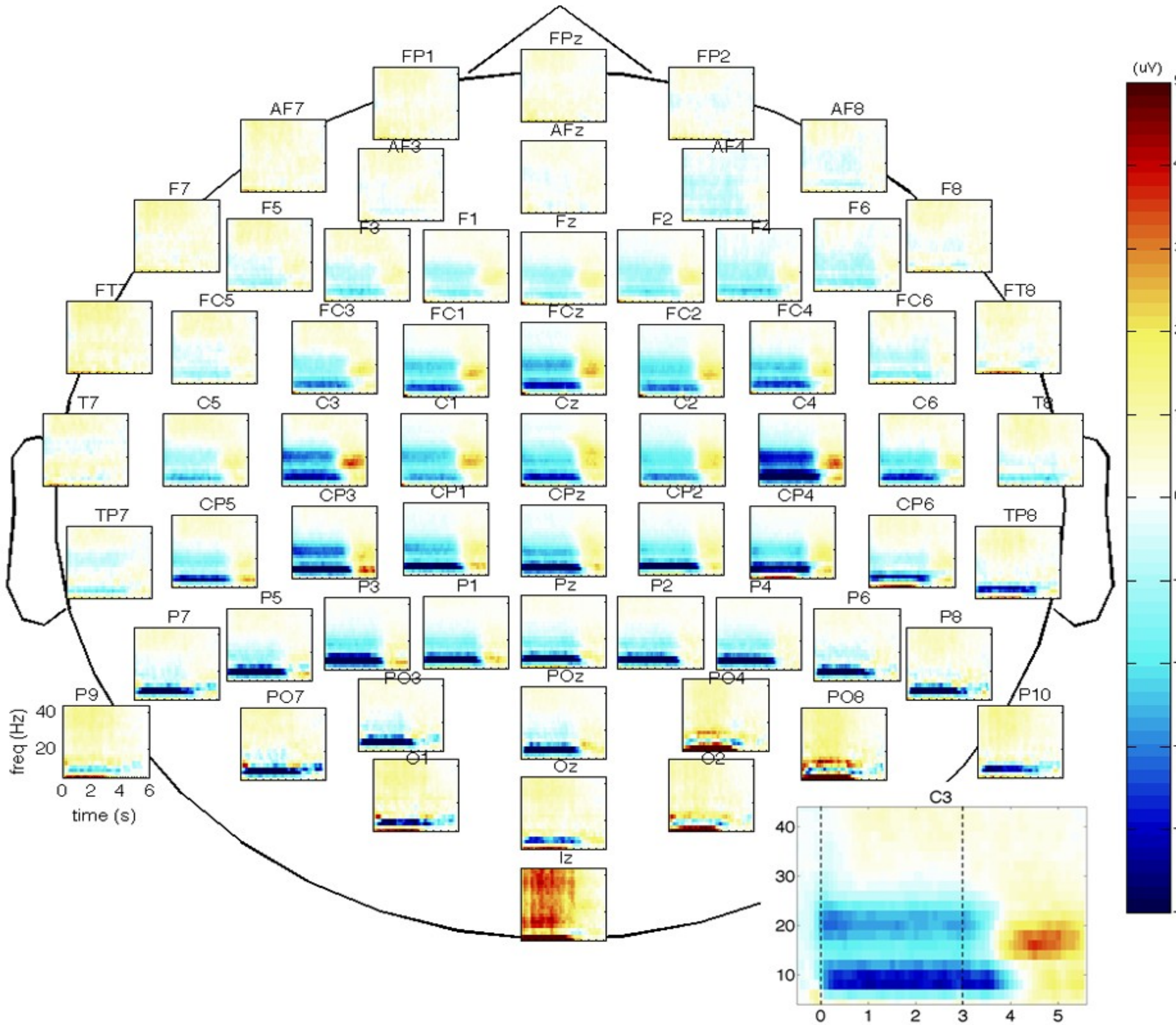
- Steady State Response
  - response to long-term periodic stimulus



- Modality dependent optimal frequency
  - Visual (VSSEP)– 16-20Hz, occiptial cortex
  - Auditory (ASSR) – AM modulation ~40Hz, auditory cortex
  - Tactile (SSSEP) – AM ~21Hz, contralatorial somatosensory cortex
  - Used to diagonise correct operation of the sensory system
- Response amplitude influenced by selective attention
  - Used for parallel/serial selective attention BCI

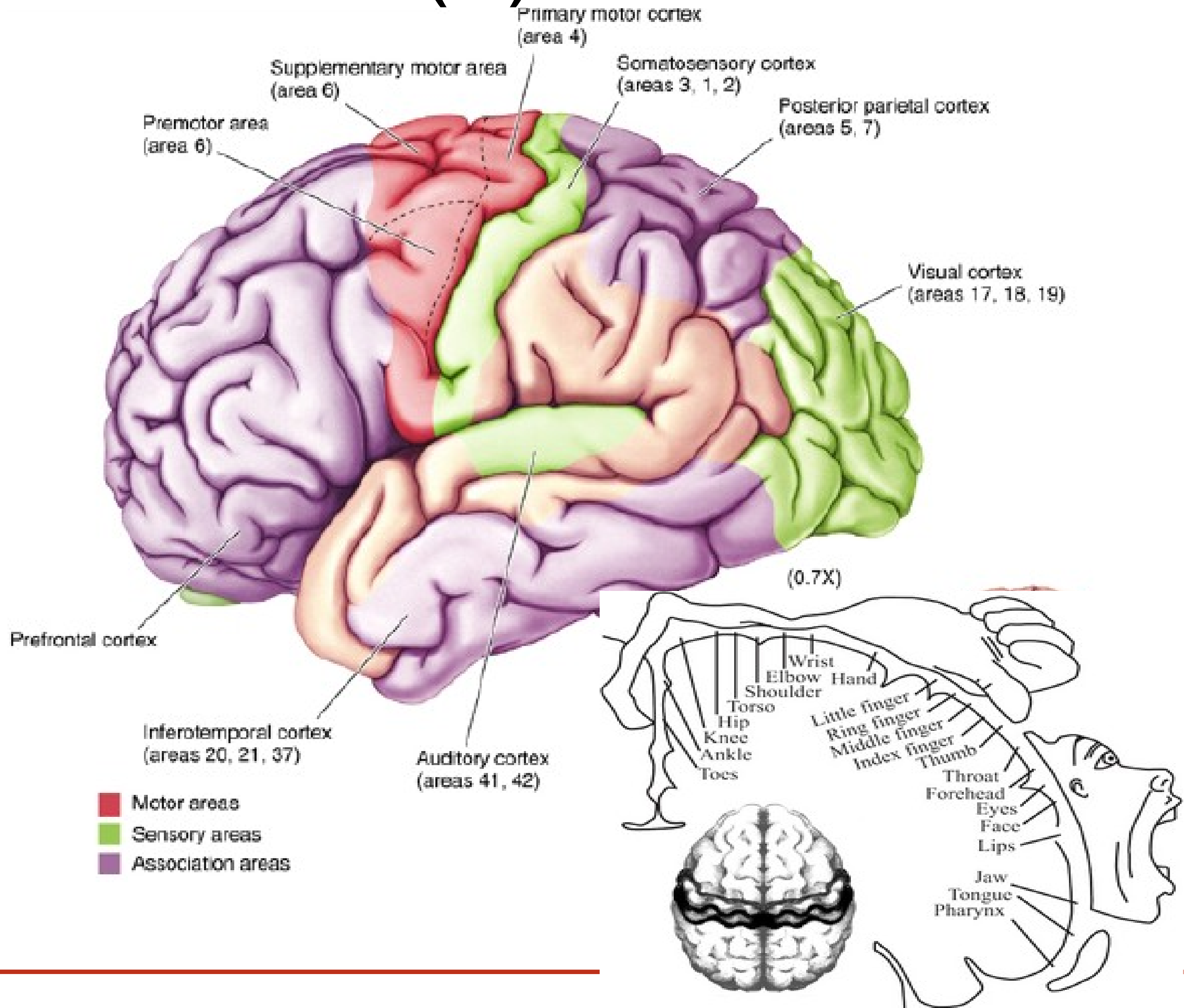
Induced Response / ERSP BCIs

- ERSP – task dependent change in the power of the signal in a particular frequency range
- Pseudonames:
  - Event Related De-Synchronisation (ERD/ERS)



ERSPs (2)

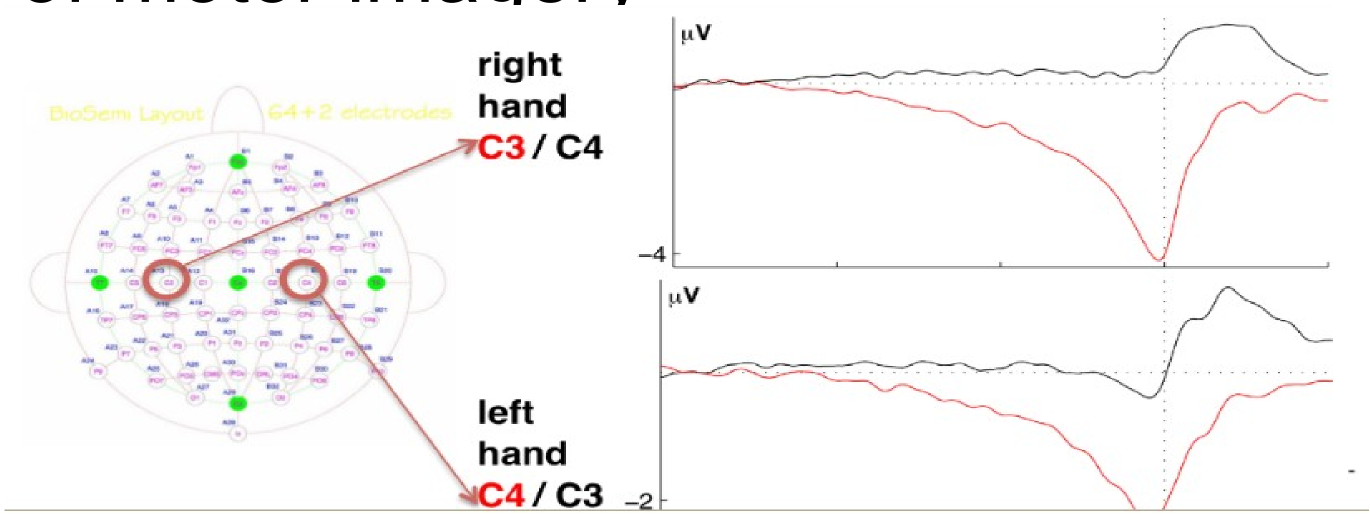
- Location and frequency of the response depends on the mental task the user is performing

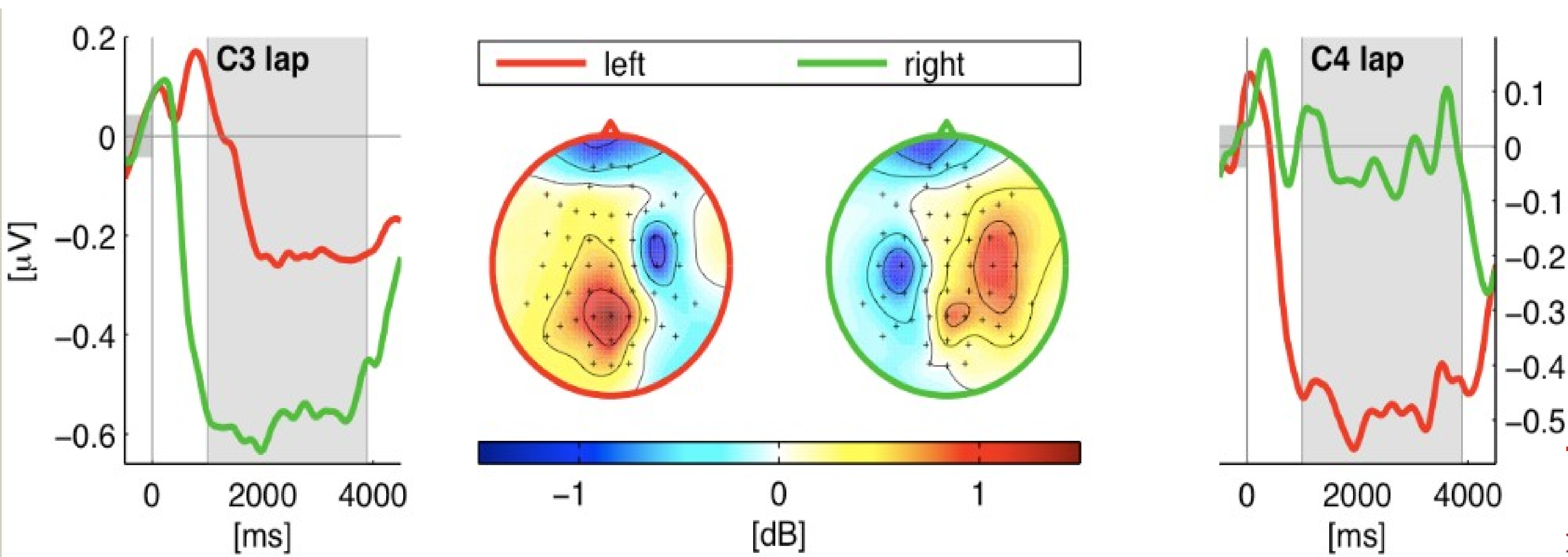


ERSP BCIs – Imagined Movements

- Brain signatures of motor imagery

Lateralised Readiness Potential (LRP)



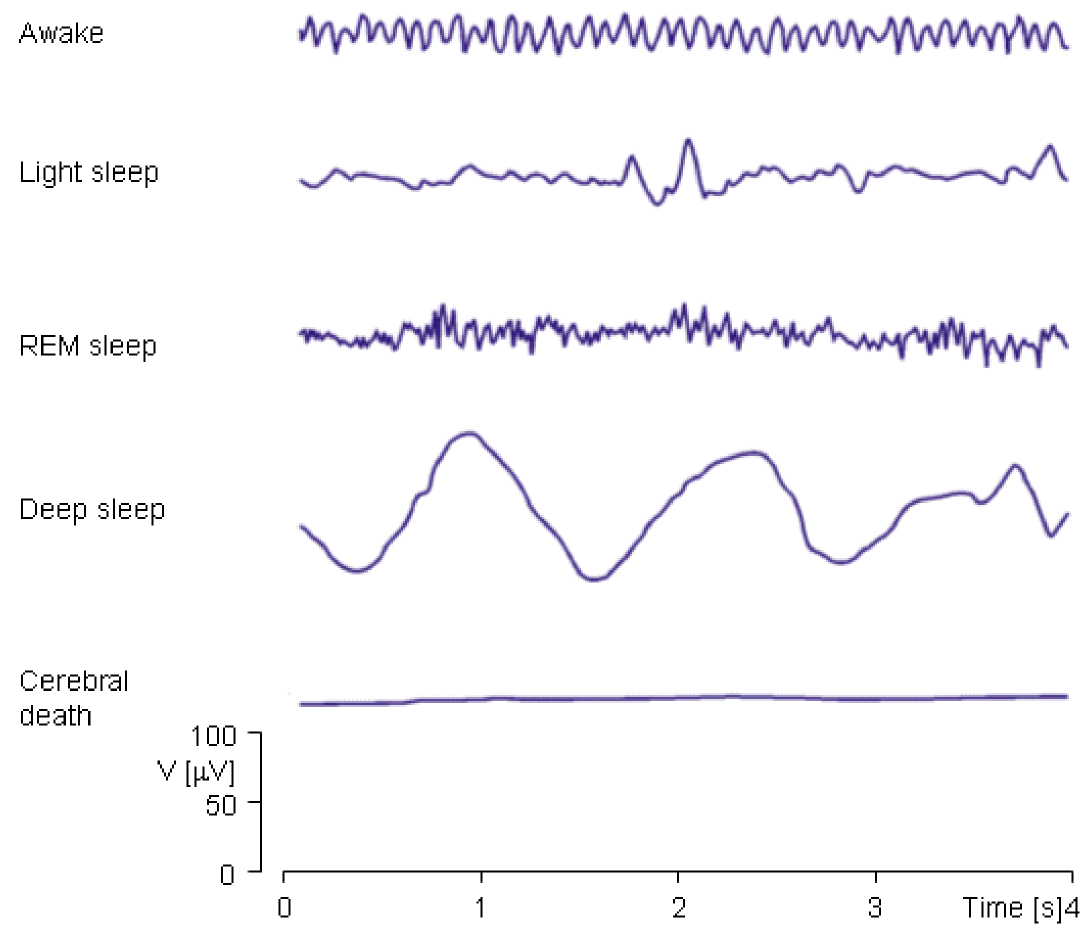


ERD/ERS

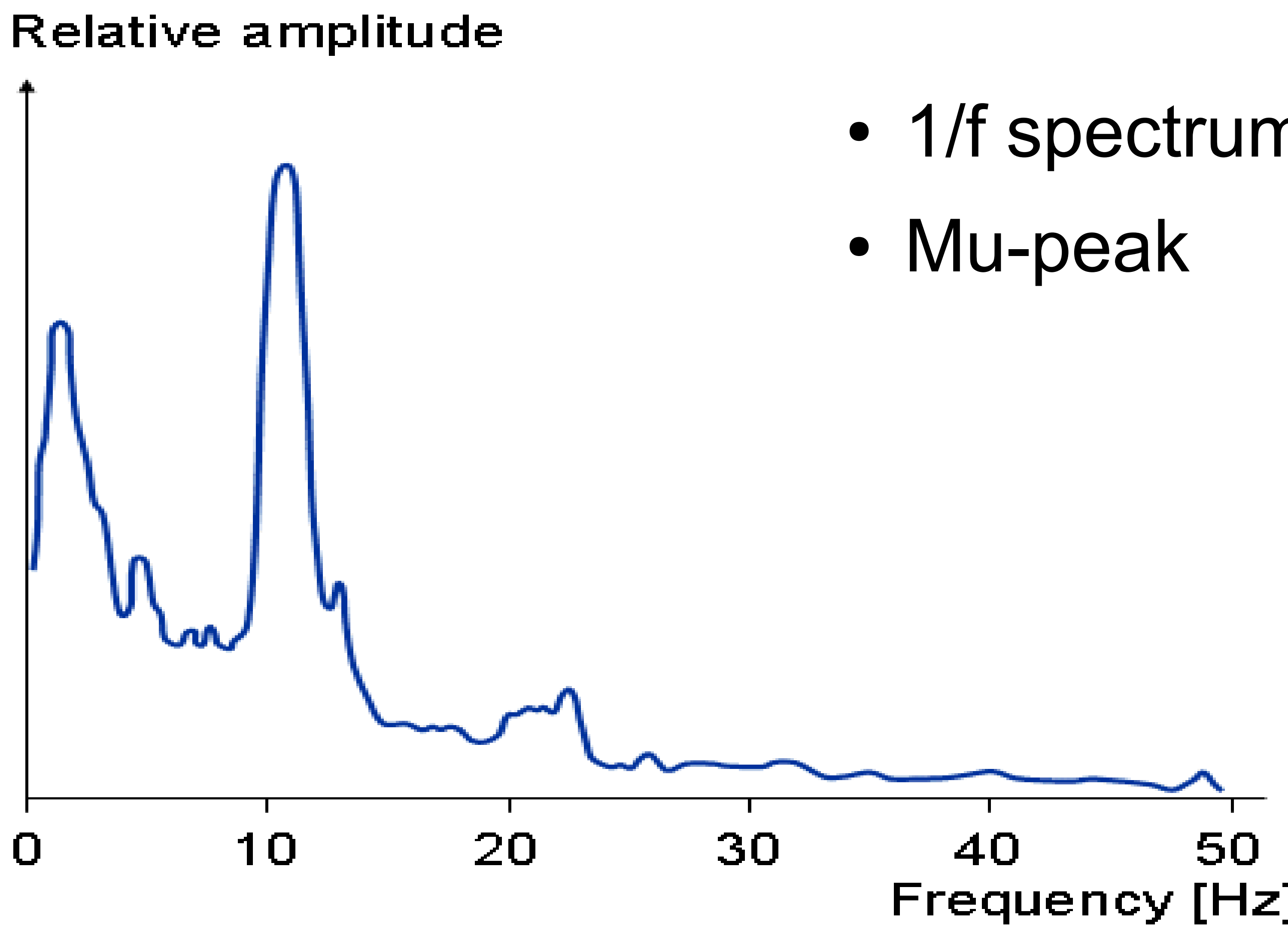
Basic properties of EEG/MEG data and artifacts

Basic Properties of the EEG

- Temporally
  - Fractal structure

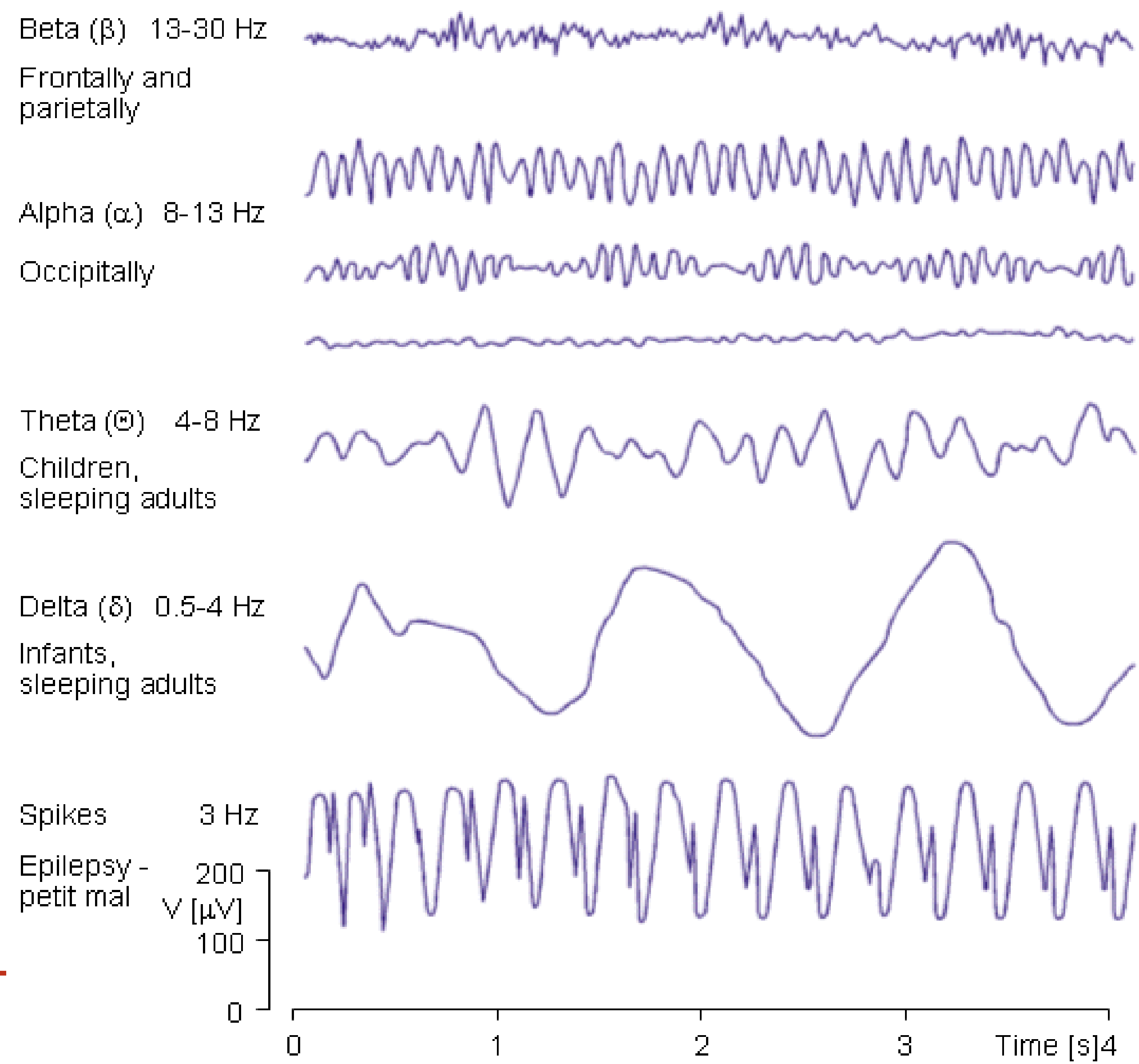


EEG Properties - Spectrally



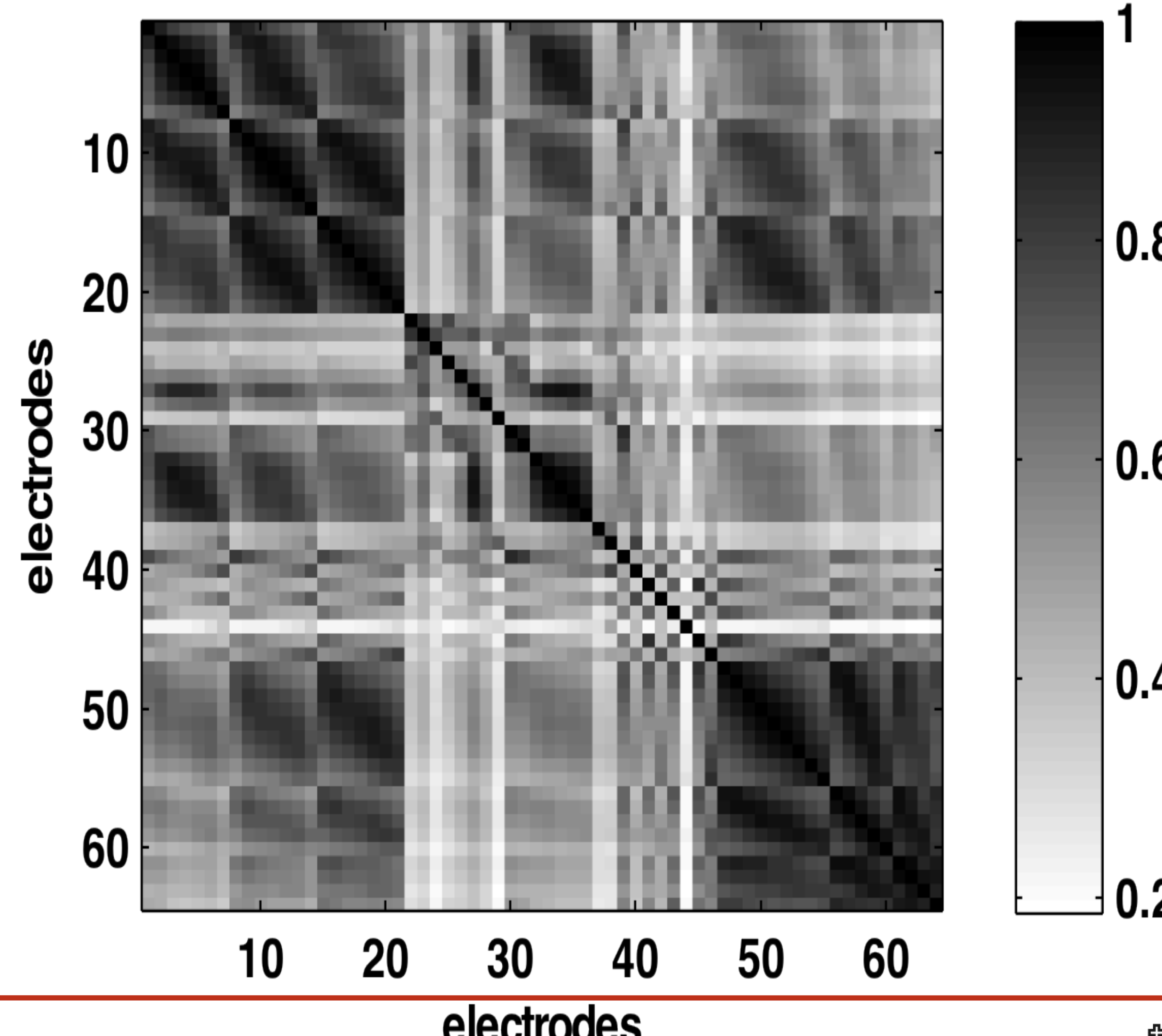
- 1/f spectrum
- Mu-peak

EEG properties - spectrally



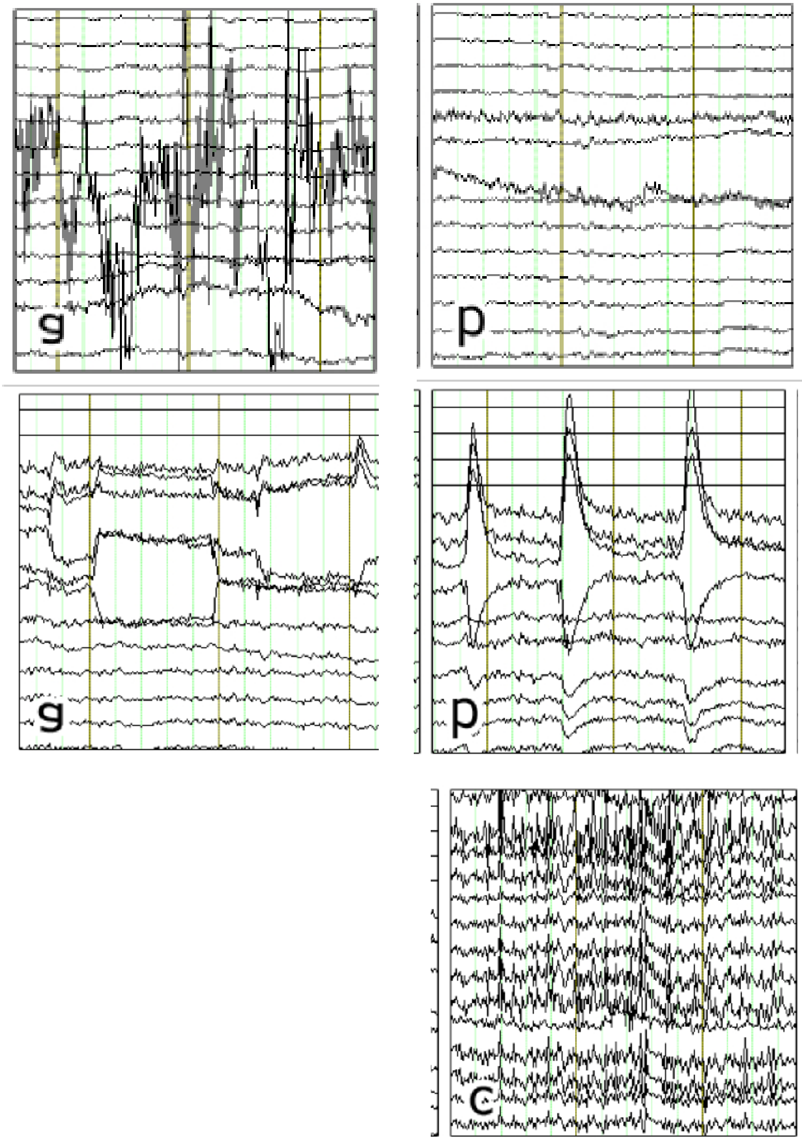
EEG Properties - Spatially

- High spatial correlation between electrodes



External Artifact Sources in EEG

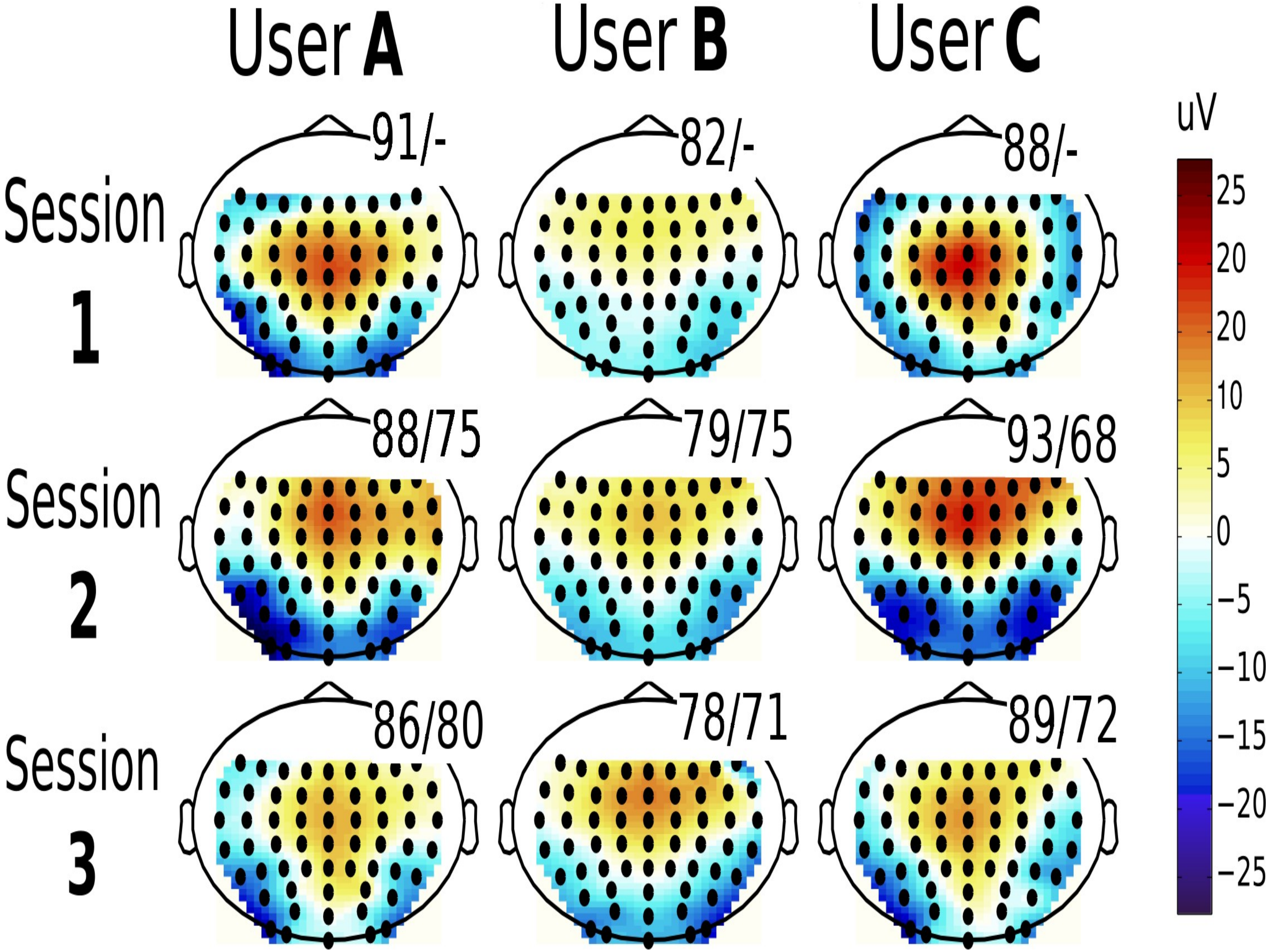
- Bad-channels
- 50Hz
- Slow electrode drifts
- Movement effects – head wiggling
- Eye-movement / Eye blinks
- Muscle movement
    - jaw-clenching
    - frowning
    - ear-waggling



Internal Artifact Sources in EEG

- Eyes open vs. Closed
- Relaxation (deep-breathing) vs. Stressed
- Hyperventilation

Very subject and session dependent



Basic BCI signal processing stages

BCI signal analysis pipeline

1)Detrending

2)Bad-channel identify and remove

3)Re-referencing/Spatial Filtering

ERP

4)Spectral Filtering

5) Bad-trial ID & remove

ERSP

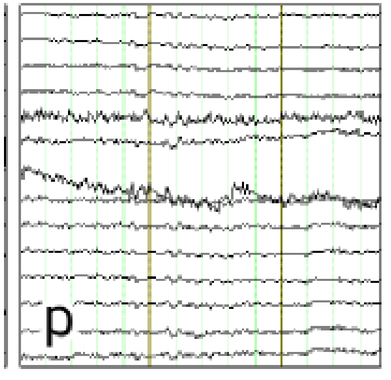
4) Feature extraction

5) Feature selection

6) Bad-trial ID & rm

5/6) Classifier Training

1) Detrending

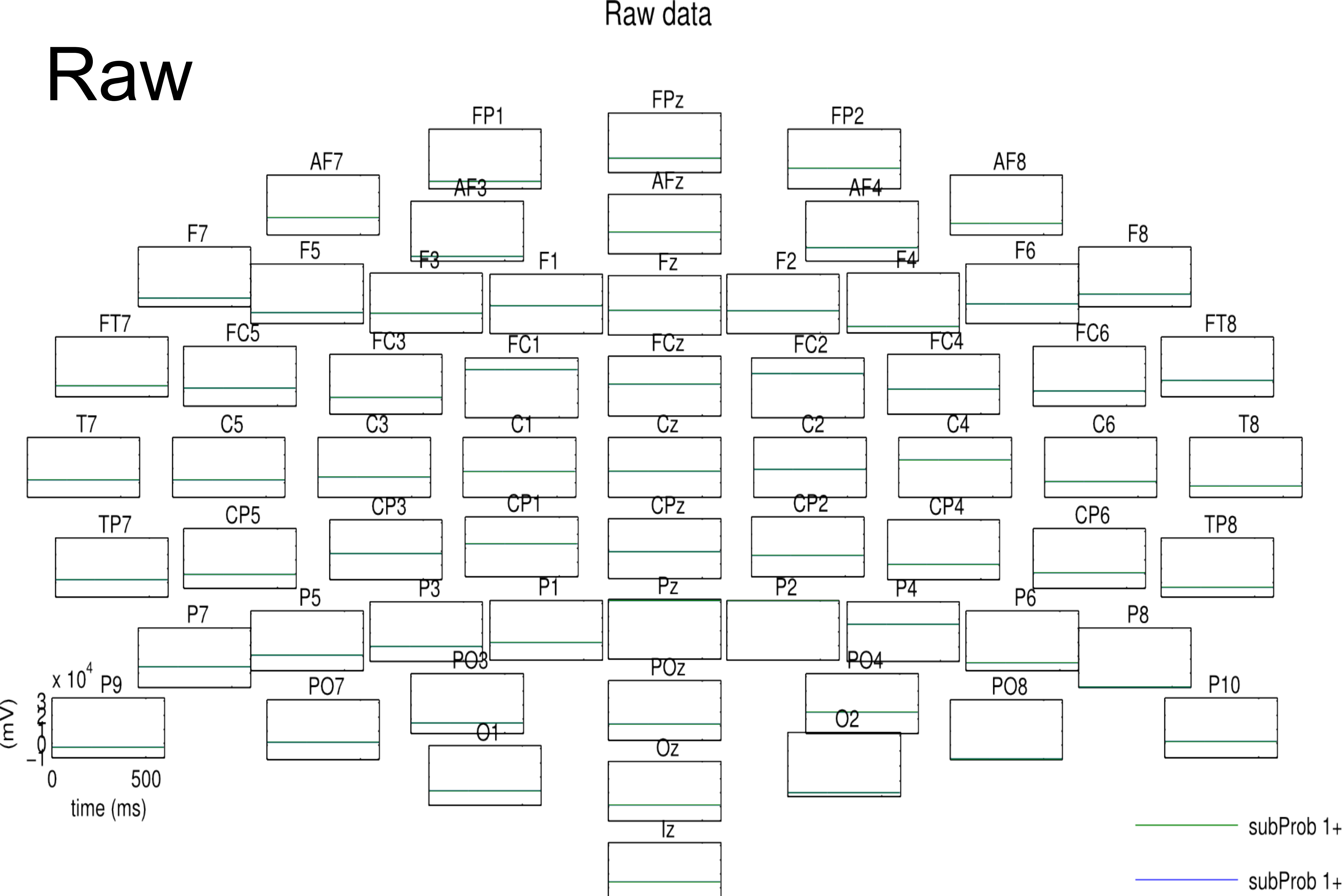


- Why?
  - Remove slow-drifts and arbitary offsets in EEG data
- How?
  - Compute and subtract linear trends for each channel and epoch

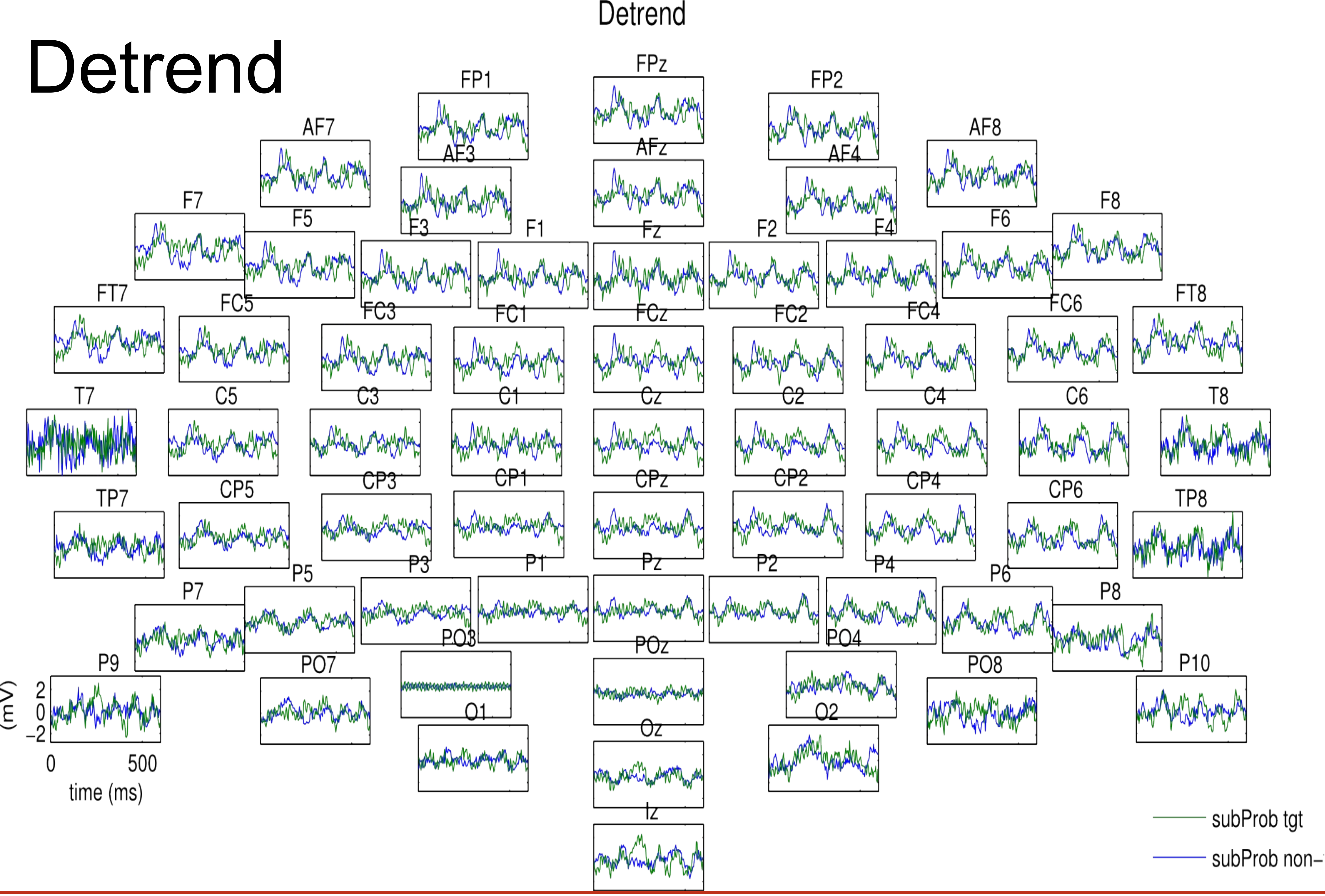
Key Function :

- X=detrend(X,dim); %Detrend data X along dimension dim

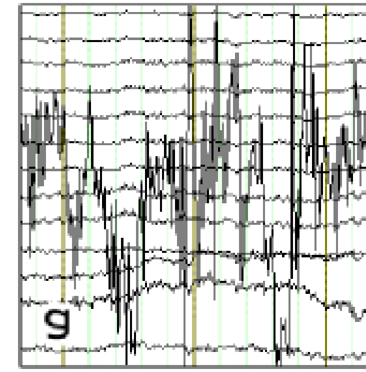
Raw


Raw data
FPz
FP1
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
x 10^4
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
3
2
1
0
−1
0
500
time (ms)
subProb 1+
subProb 1+

Detrend


Detrend
FP1
FPz
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
2
0
−2
0
500
time (ms)
subProb tgt
subProb non−

2) Bad channel identify and remove

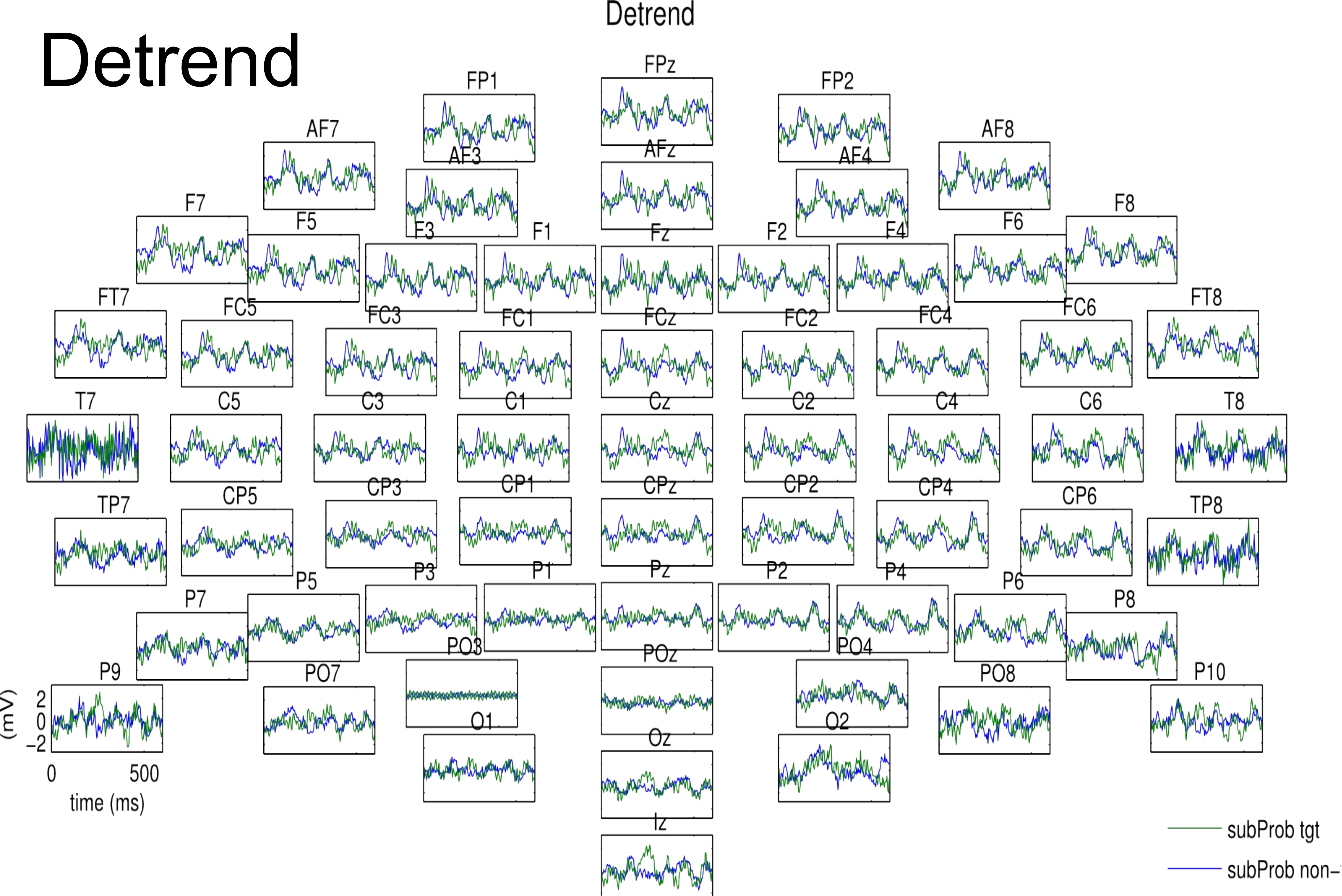


- Why?
  - Some channels don't connect well so only pickup noise. Leaving them in messes up the reset when spatial filtering.
- How?
  - Identify channels with excessively high power.
  - Specificially:
    - compute total power for each channel over all epochs
    - Compute mean channel power and variance in channel power
    - Remove any channels with power more than 3 std-dev from mean

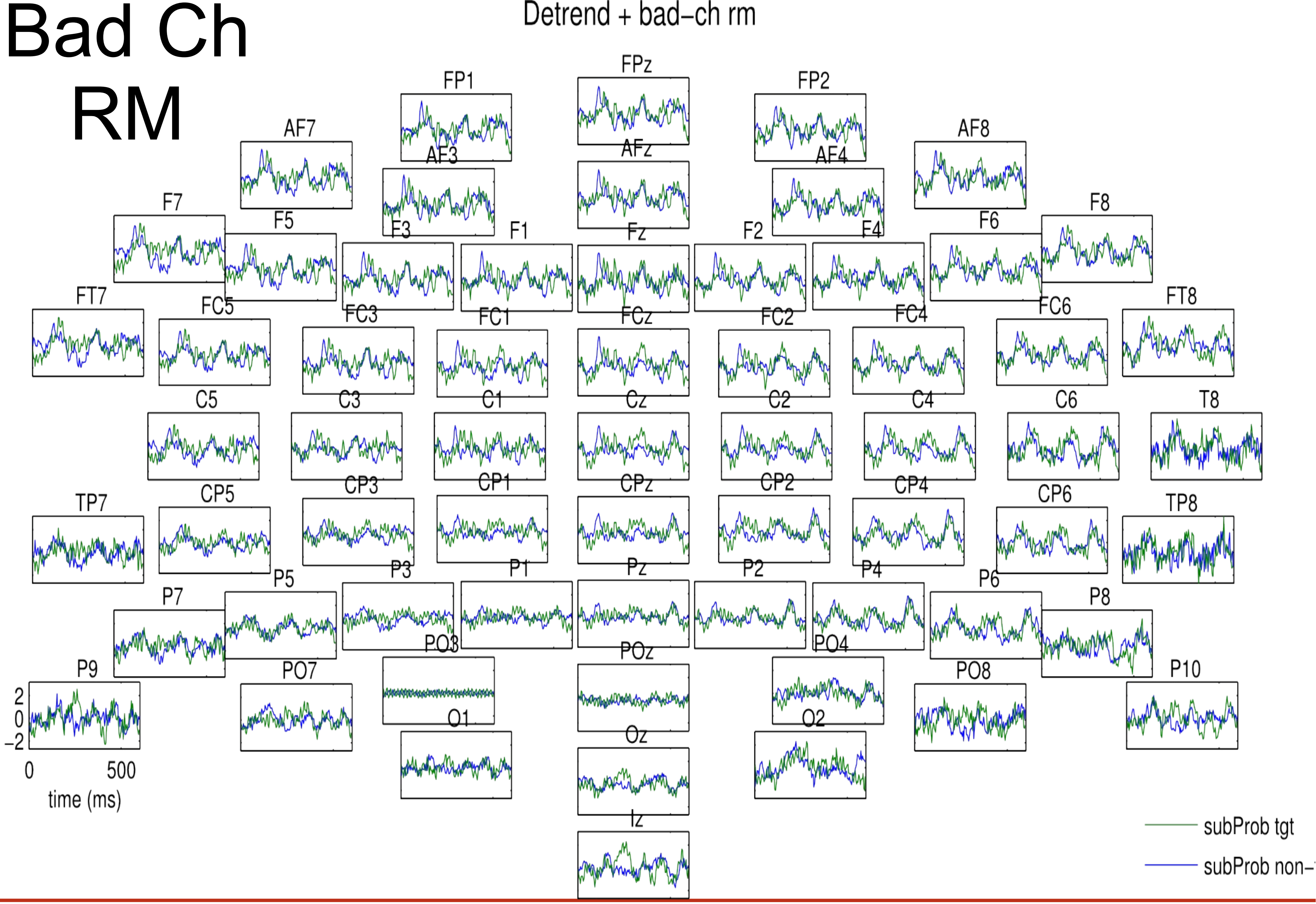
Key Function:

- badidx=idOutliers(*X*,*dim*,*thresh*); % identify outlining elements of *X* along *dim*

Detrend


Detrend
FP1
FPz
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
2
0
−2
0
500
time (ms)
subProb tgt
subProb non−

Bad Ch RM



Detrend + bad–ch rm
FPz
FP1
FP2
AF7
AF8
AF3
AFz
AF4
F7
F8
F5
F6
F3
F1
Fz
F2
F4
FT7
FT8
FC5
FC6
FC3
FC1
FCz
FC2
FC4
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
TP8
CP5
CP6
CP3
CP1
CPz
CP2
CP4
P7
P8
P5
P6
P3
P1
Pz
P2
P4
PO3
POz
PO4
P9
P10
PO7
PO8
O1
Oz
O2
Iz
(mV)
2
0
−2
0
500
time (ms)
subProb tgt
subProb non–

3) Spatial Filter

- Why?
    - EEG contains lots of signal from external noise sources – which is visible as a signal common to all channels
    - Volume conduction means nearby channels have high correlation
- How?
    - Common Average Reference (CAR) -- Remove common external signal by subtracting the average signal over channels from all channels
    - Surface Laplacian – Remove channel correlation (and common signal!) subtracting a local average signal (default)
    - Spatial Whiten – Remove channel correlation (and common signal!) using PCA to map to a co-ordinate set where channels are uncorrelated.

Key functions

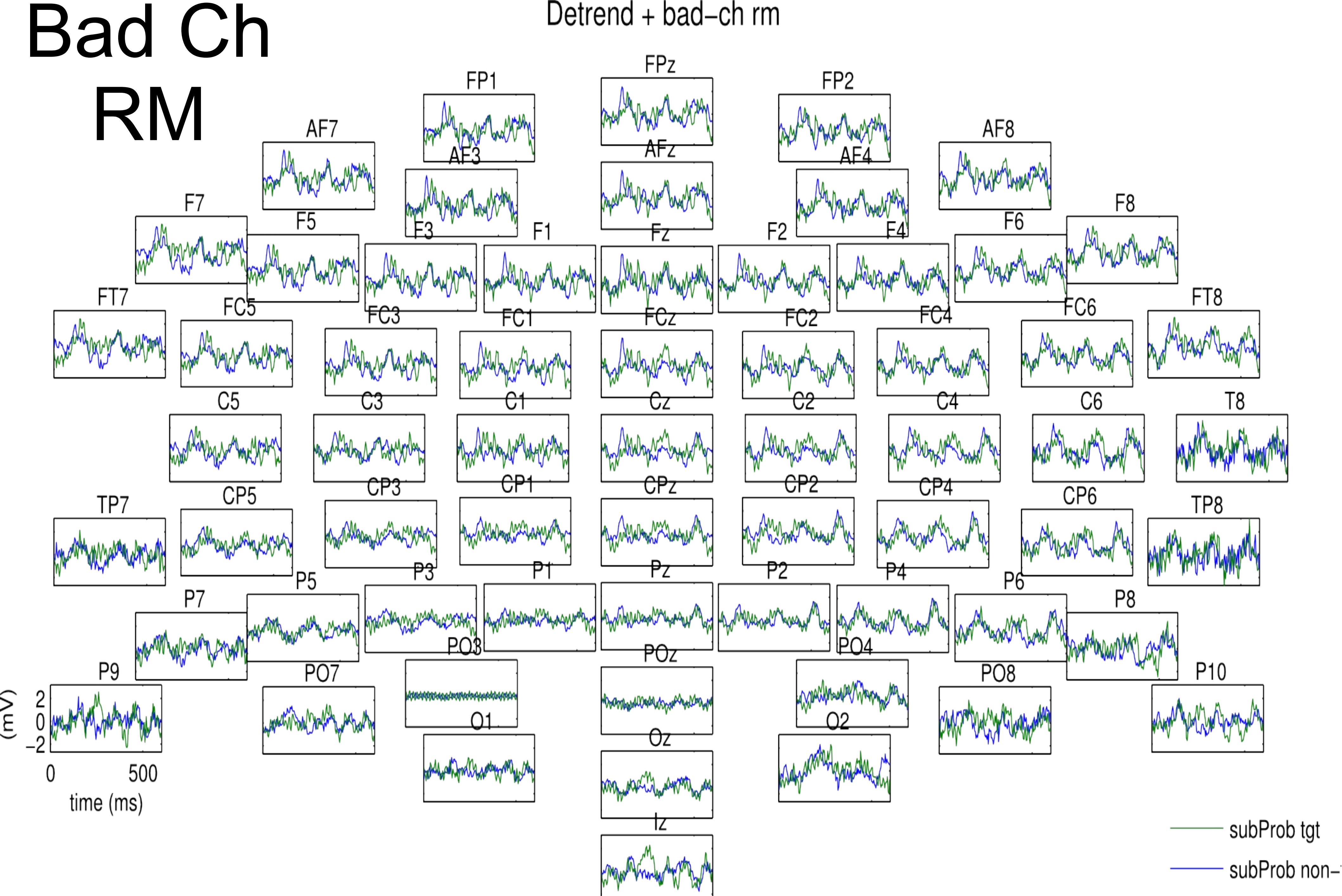
- X=repop(X,'-',mean(X,1));
  - subtract average over dimension 1
  - Alternative:
    - X=X-repmat(mean(X,1),[size(X,1),1,1]);
- R=sphericalSplineInterpolate(*chpos*,*chpos*,[],[],'slap');
  - compute Surface Laplacian (SLAP) spatial filter for electrodes at **3-d** positions chpos
  - R is a [nCh x nCh] spatial filter matrix
- *X*=tprod(*X*,[-1 2 3],*R*,[-1 1]);
  - Apply the spatial filter matrix *R* to the data *X*
  - Alternative:

Make 3d again

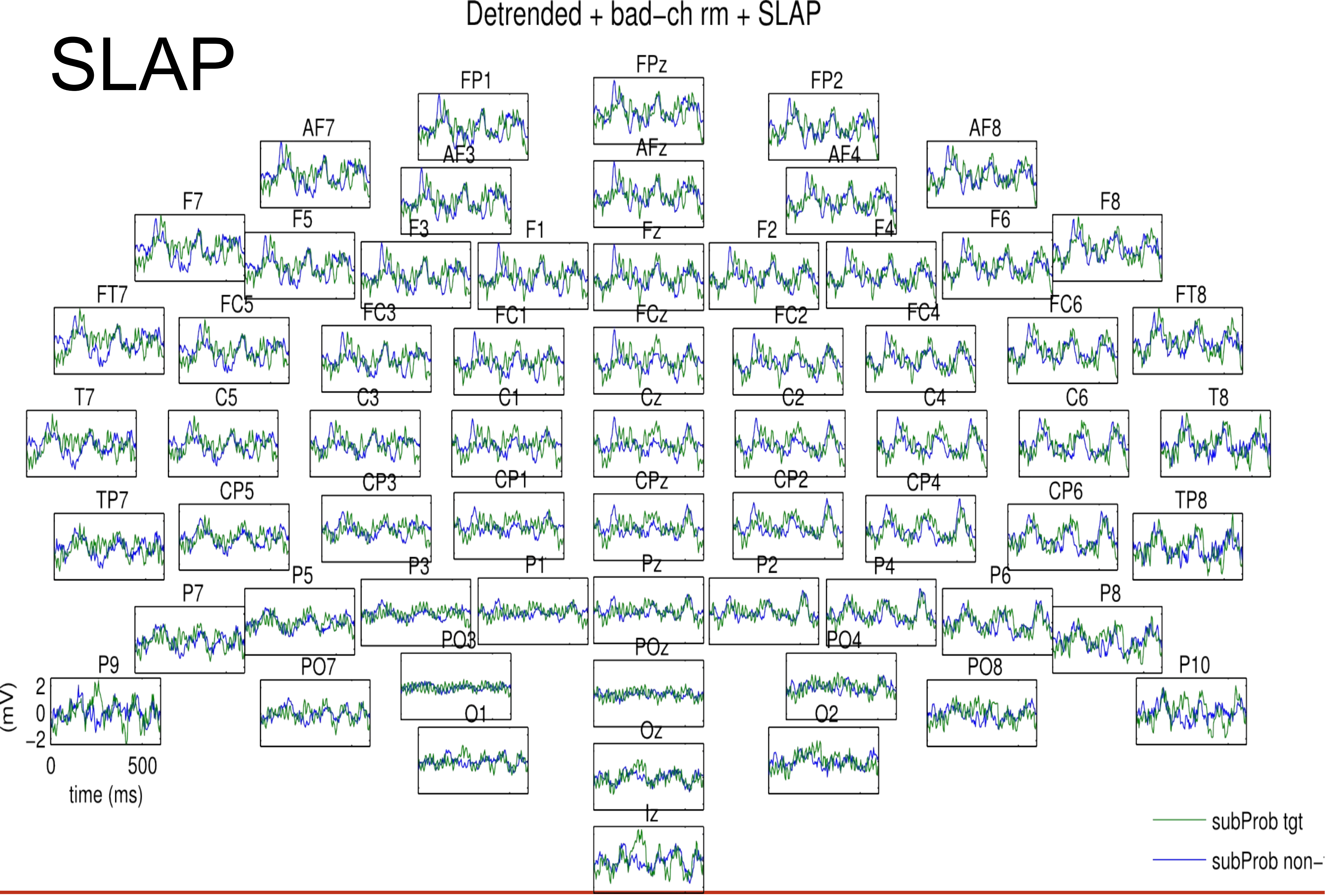
    - X=reshape(R'*reshape(X,size(X,1),[]),[size(R,2),size(X,1),size(X,2)])

Apply filter Make 2D

Bad Ch RM


Detrend + bad–ch rm
FPz
FP1
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
2
0
−2
0
500
time (ms)
subProb tgt
subProb non–

SLAP



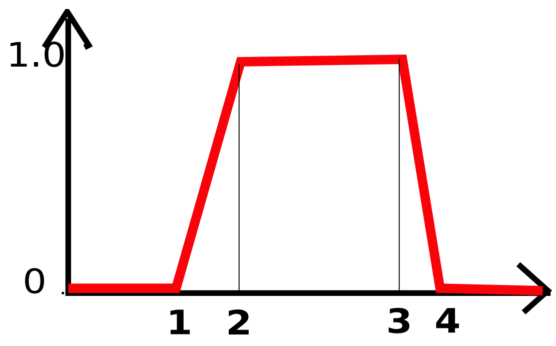
Detrended + bad–ch rm + SLAP
FPz
FP1
FP2
AF7
AFz
AF8
AF3
AF4
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
2
0
–2
0
500
time (ms)
subProb tgt
subProb non–

4) ERP Spectrally Filter

- Why?
  - Our signal of interest only occurs in a particular frequency range
  - Thus, any signal which occurs outside this frequency range **must** be noise, and should be removed
- How?
  - Apply a spectral filter to remove frequencies outside the range of interest

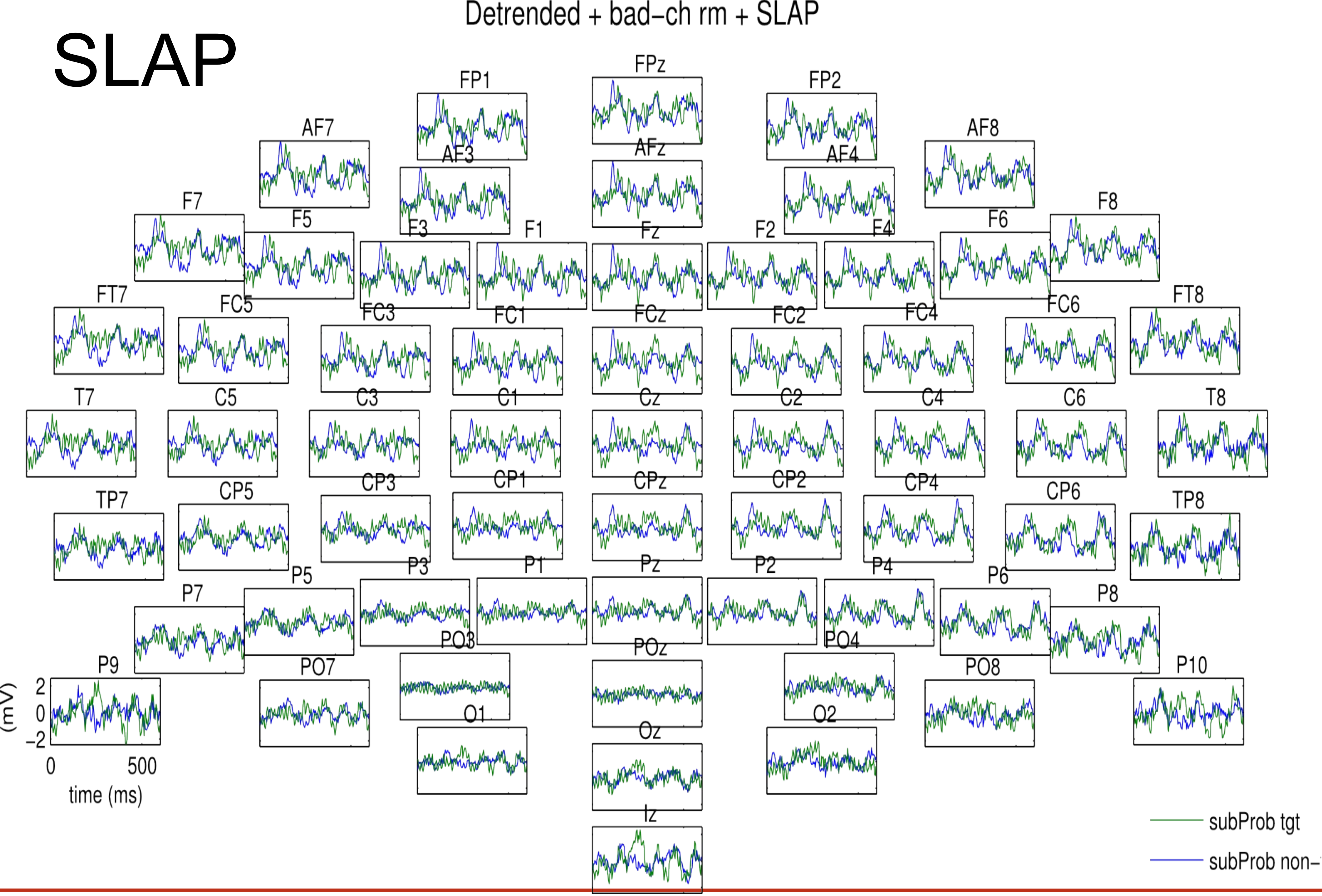
Key functions

- filt=mkFilter(size(X,2)/2,*bands*,fs/size(X,2));
  - Make a spectral domain filter with pass-band given by *bands*
  - *bands* has 4 elements



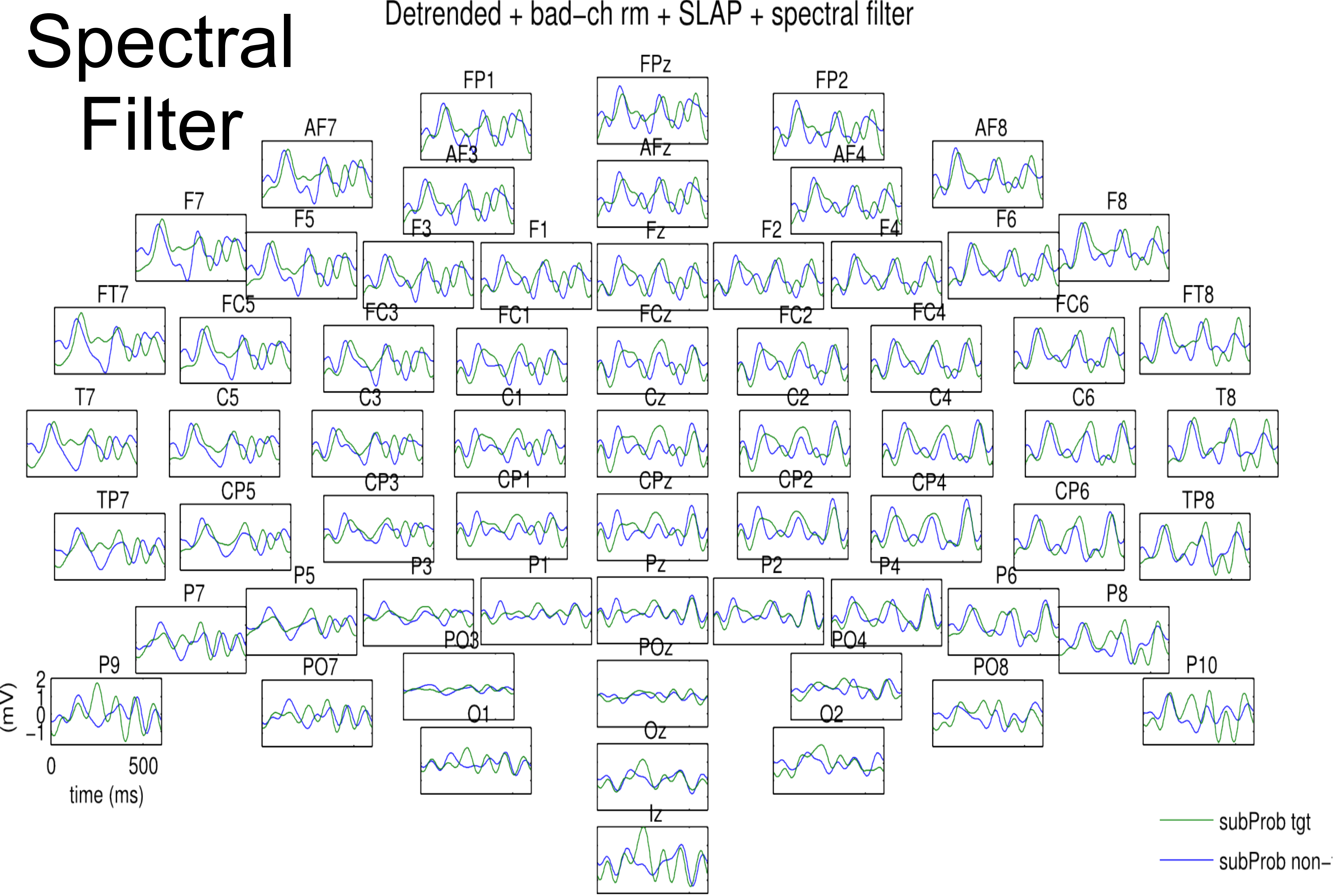
  - To reduce filtering artifacts ramps should be >1hz wide
- X=fftfilter(X,*filt*,[],2)
  - Apply the spectral domain filter *filt* to X along dim 2

SLAP

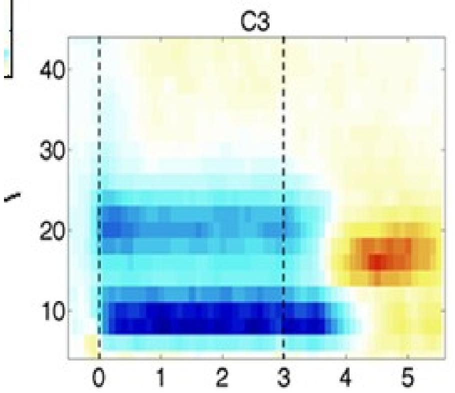


Detrended + bad−ch rm + SLAP
FPz
FP1
FP2
AF7
AF8
AF3
AFz
AF4
F7
F8
F5
F6
F3
F1
Fz
F2
F4
FT7
FT8
FC5
FC6
FC3
FC1
FCz
FC2
FC4
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(mV)
2
0
−2
0
500
time (ms)
subProb tgt
subProb non−

Spectral Filter



4) ERSP Feature Extraction



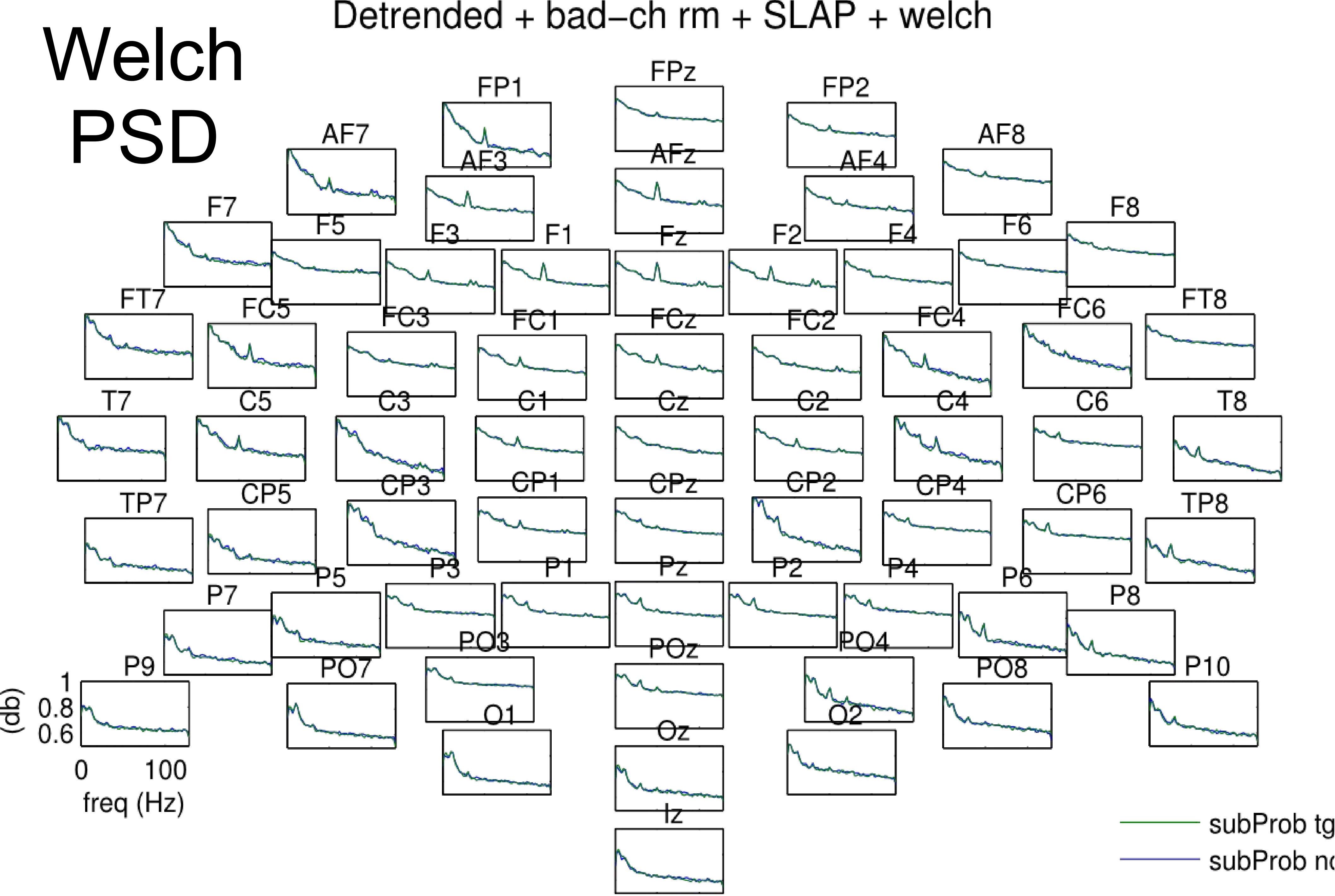
- Why?
  - Our signal of interest is **not** time locked, but a change in power in a particular frequency range
  - Thus, it cannot be detected from the raw time-domain features by a linear classifier
- How?
  - Use Welch's method to compute a high quality estimate of the signals power spectrum for each epoch, i.e. power in each frequency bin
  - N.B. To make the distribution of powers more Gaussian distributed, we use **log** power as the classification feature

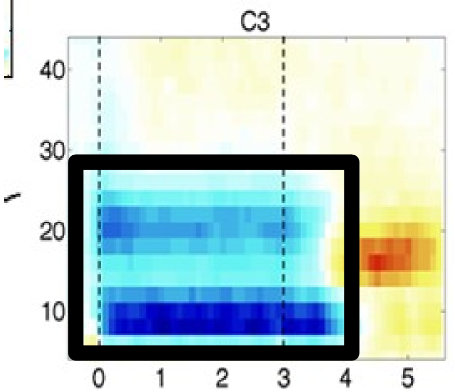
Key functions

- X=welchpsd(X,2,'width_ms',500,'fs',*fs*);
  - Compute the power-spectral-density (PSD) of X along dimension 2 using Welch's method[1] with *width_ms* sized windows.
    - N.B. Output spectral resolution = 1000/*width_ms* Hz

[1] see en.wikipedia.org/wiki/Welch's_method

Welch PSD

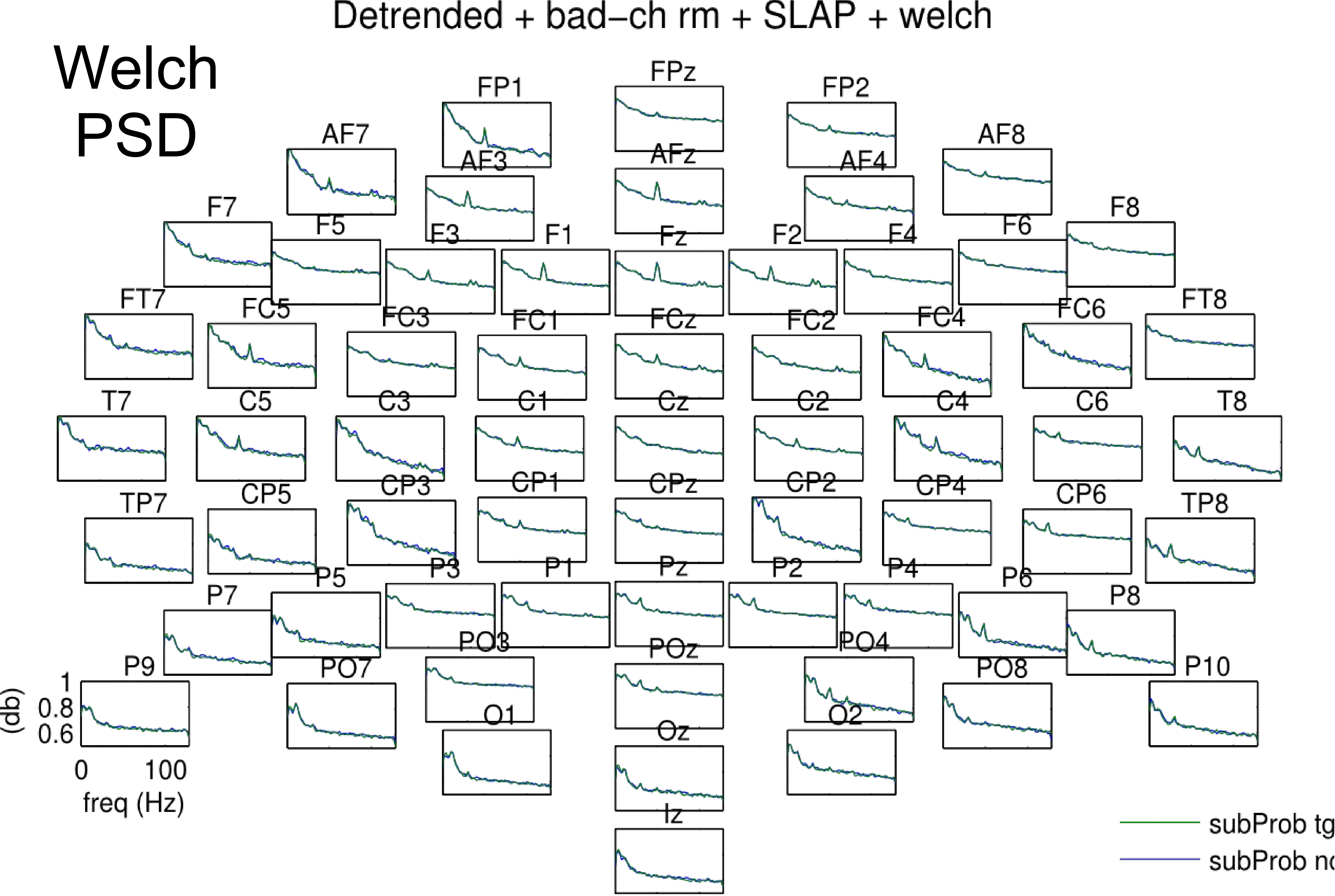

Detrended + bad–ch rm + SLAP + welch
FPz
FP1
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
(db)
1
0.8
0.6
0
100
freq (Hz)
subProb tg
subProb nc



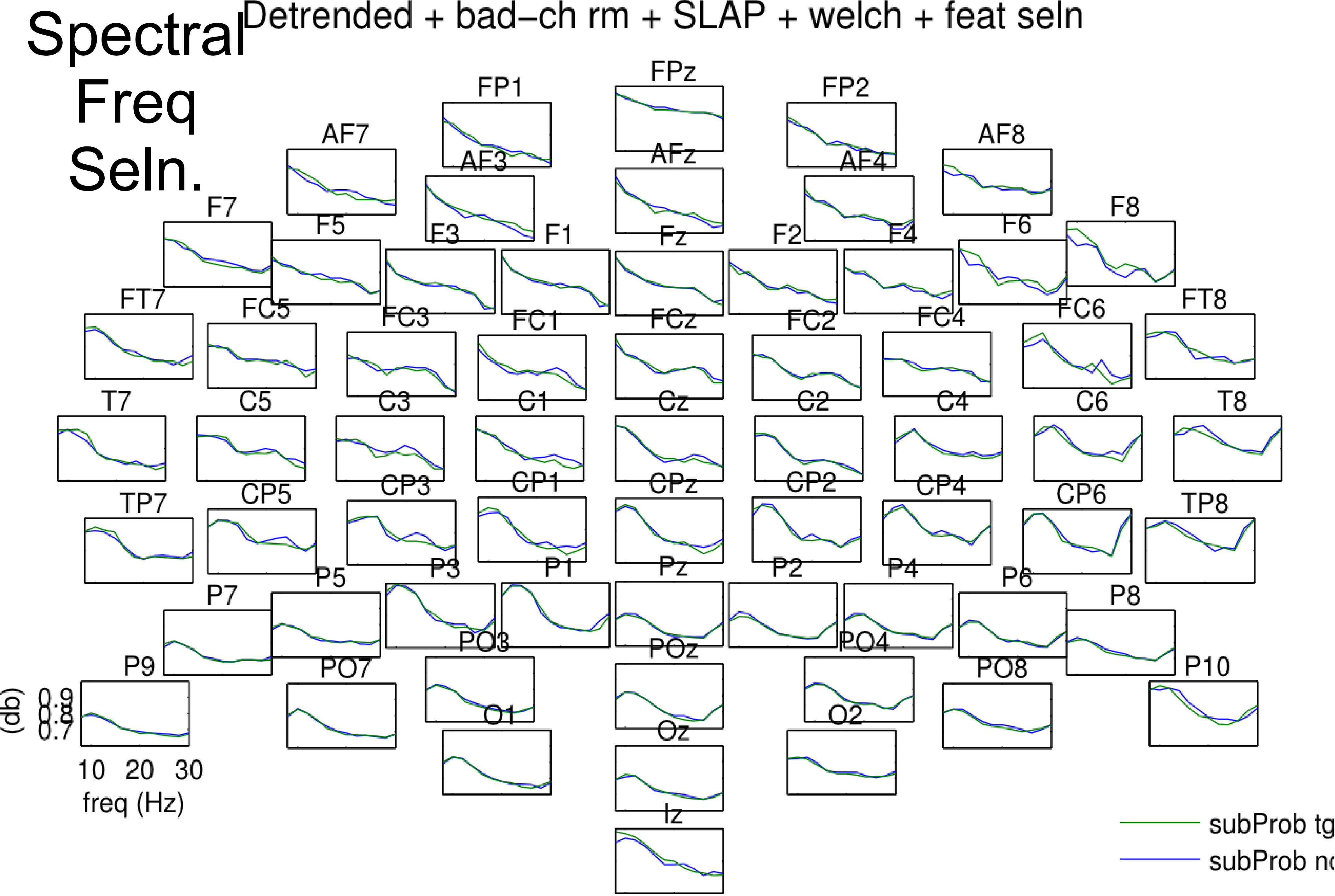
5) ERSP Feature Selection

- Why?
  - Our signal of interest only occurs in a particular frequency range
  - Thus, any signal which occurs outside this frequency range **must** be noise, and should be removed
- How?
  - Discard frequency bins from the power spectrum which occur outside the range of interest

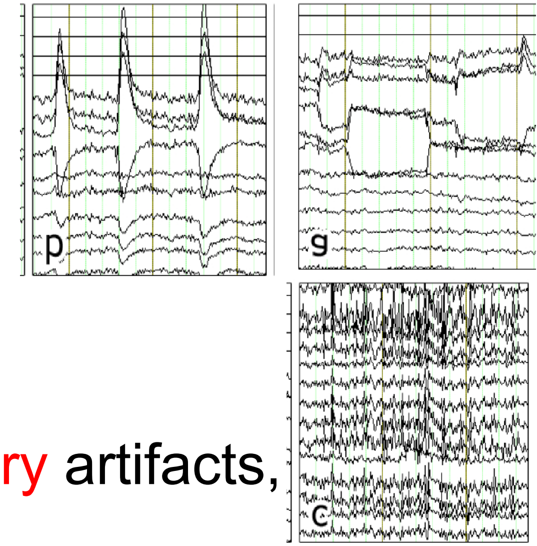
Welch PSD


Detrended + bad–ch rm + SLAP + welch
FP1 FPz FP2
AF7 AF3 AFz AF4 AF8
F7 F5 F3 F1 Fz F2 F4 F6 F8
FT7 FC5 FC3 FC1 FCz FC2 FC4 FC6 FT8
T7 C5 C3 C1 Cz C2 C4 C6 T8
TP7 CP5 CP3 CP1 CPz CP2 CP4 CP6 TP8
P7 P5 P3 P1 Pz P2 P4 P6 P8
P9 PO7 PO3 POz PO4 PO8 P10
O1 Oz O2
Iz
(db)
1
0.8
0.6
0
100
freq (Hz)
subProb tg
subProb nc

Spectral Freq Seln.



5/6) Bad trial identify and remove

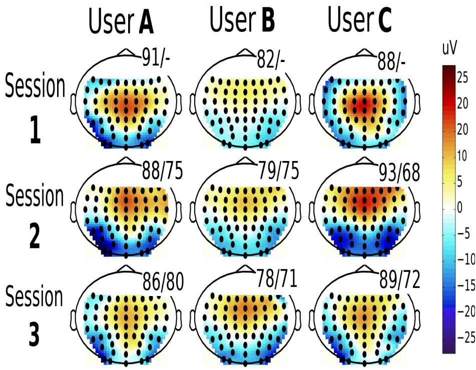


- Why?
  - Some trials (blocks of time) are just corrupted by tempory artifacts, e.g. eye-blinks.
  - Leaving these in will cause the classifier to produce garbage which leads to output mistakes.
  - Generally Better to identify produce a null-output than make a mistake... (particularly in sequence based BCIs like visual spellers)
- How?
  - Identify time points with excessively high power.
  - Specificially:
    - compute total power for each trial (time-block)
    - Compute mean trial power and variance in power
    - Remove any channels with power more than 3 std-dev greater than

Key Function :

- badidx=idOutliers(*X*,*dim*,*thresh*); % identify outlining elements of *X* along *dim*

6/7) Train Classifier



- Why?
  - Account for high variability in signal and noise properties, by building a special purpose detector for this subject on this day.
- How?
  - Train a linear logistic regression classifier with quadratic regularisation on the gathered examples
  - With 10-fold cross validation to select the optimal regularisation strength
  - N.B. Many other classifers possible... doesn't seem to matter much so long as its regularised

Key functions

- X=cvtrainLinearClassifier(X,Y,*Cs*,*nFold*);
  - Train a linear classifier on data X for labels Y
  - default *quadratically (L_2)regularised [1] logistic regression*[2]
  - estimate regularisation parameter using *nFold* cross-validation [3]
  - *Cs* is the set of regularisation parameters to test (automatically set if empty)

- Note: specify alternative classifiers using:
  - cvtrainLinearClassifier(X,Y,[],10,'objFn',*objFn*)
  - Where *objFn* is the name of a classifier, e.g. 'lr_cg','l2svm_cg','klr_cg','rkls_cg',etc.
  - N.B. Use option 'compKernel',1 for **kernel** methods [4], SVM, KLR

[1] see en.wikipedia.org/wiki/Regularization_%28mathematics%29
[2] see en.wikipedia.org/wiki/Logistic_regression
[3] see en.wikipedia.org/wiki/Cross-validation_(statistics)
[4] see en.wikipedia.org/wiki/Kernel_methods

Classifier Output

Regularisation Strength

High ... low

outer fold, (all data)

Folds

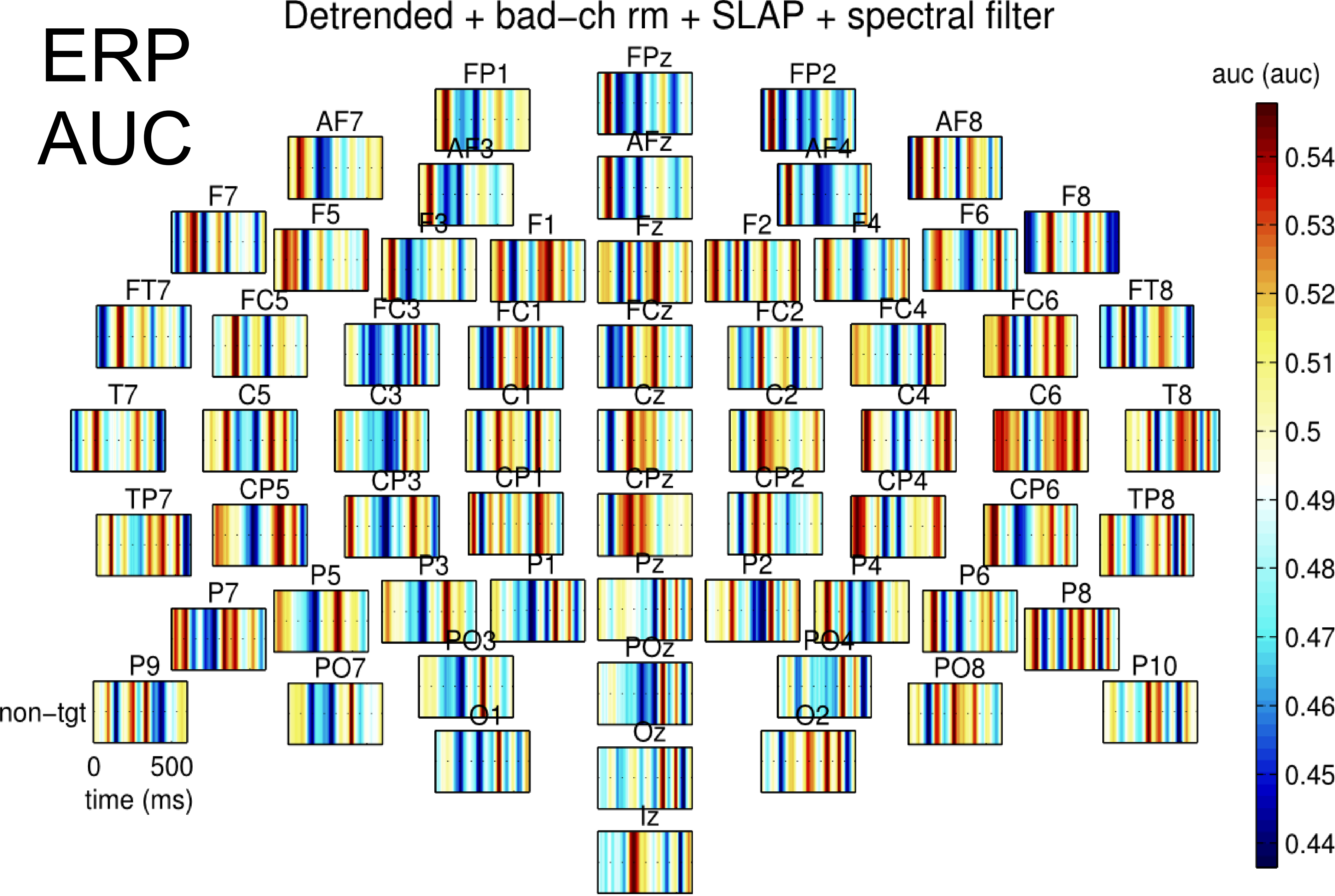
	High					low
(out)	0.51/NA	0.93/NA	0.95/NA	0.95/NA	0.95/NA	0.95/NA
(1)	0.50/0.50	0.93/0.91	0.95/0.95	0.95/0.95	0.95/0.95	0.95/0.95
(2)	0.50/0.50	0.93/0.92	0.95/0.94	0.95/0.94	0.96/0.94	0.96/0.94
(3)	0.50/0.50	0.92/0.96	0.95/0.99	0.95/0.99	0.95/1.00	0.95/1.00
(4)	0.50/0.50	0.93/0.93	0.95/0.97	0.95/0.97	0.95/0.98	0.95/0.98
(5)	0.50/0.50	0.92/0.93	0.96/0.94	0.95/0.93	0.96/0.93	0.96/0.93
(6)	0.50/0.50	0.93/0.91	0.95/0.96	0.95/0.97	0.95/0.97	0.95/0.97
(7)	0.50/0.50	0.92/0.94	0.95/0.96	0.95/0.96	0.95/0.96	0.95/0.96
(8)	0.50/0.50	0.93/0.91	0.96/0.94	0.95/0.94	0.95/0.94	0.95/0.94
(9)	0.50/0.50	0.92/0.94	0.95/0.98	0.95/0.97	0.95/0.97	0.95/0.97
(10)	0.50/0.50	0.94/0.89	0.96/0.89	0.96/0.89	0.96/0.89	0.96/0.89

(ave)	0.50/0.50	0.93/0.92	0.95/0.95	0.95/0.95	0.95/0.95	0.95/0.95

Training Performance

Testing Performance

ERP
AUC



Detrended + bad–ch rm + SLAP + spectral filter
FP1
FPz
FP2
AF7
AF3
AFz
AF4
AF8
F7
F5
F3
F1
Fz
F2
F4
F6
F8
FT7
FC5
FC3
FC1
FCz
FC2
FC4
FC6
FT8
T7
C5
C3
C1
Cz
C2
C4
C6
T8
TP7
CP5
CP3
CP1
CPz
CP2
CP4
CP6
TP8
P7
P5
P3
P1
Pz
P2
P4
P6
P8
P9
PO7
PO3
POz
PO4
PO8
P10
O1
Oz
O2
Iz
non–tgt
0
500
time (ms)
auc (auc)
0.54
0.53
0.52
0.51
0.5
0.49
0.48
0.47
0.46
0.45
0.44

Key functions

- *clsfr*=train_erp_clsfr(X,Y,...);
  - Apply default ERP pipeline to X and train a linear classifier.
  - *Clsfr* contains the classifier and pipeline info
- f=apply_erp_clsfr(X,*clsfr*)
  - Apply ERP pipeline and classifier in *clsfr* to new data X, return classifier prediction *f* as a **decision value**
- *clsfr*=train_ersp_clsfr(X,Y,...);
  - Apply default ERSP pipeline to X and train a linear classifier
  - *Clsfr* contains classifier and pipeline info
- f=apply_ersp_clsfr(X,*clsfr*)
  - Apply ERSP pipeline and classifier in *clsfr* to new data X, return classifier prediction *f* as a **decision value**

Summary : Decoding

- 2 main types of BCI signal : ERP, ERSP
- EEG artifacts and noise mean need particular processing steps to to correct.
- Simple pipeline of processing removes these artifacts:
  - Detrend / high-pass filter
  - Identify and remove bad channels
  - Spatial filter / reference
  - Spectral filter (ERP)
  - Compute power spectrum and select frequencies (ERSP)
  - Identify and remove bad trials
  - Regularised linear classifier training / application