OHBA M/EEG Analysis Workshop

Mark Woolrich

Diego Vidaurre

Andrew Quinn

Romesh Abeysuriya

Robert Becker

UNIVERSITY OF OXFORD

OHBA
Oxford Centre for Human Brain Activity

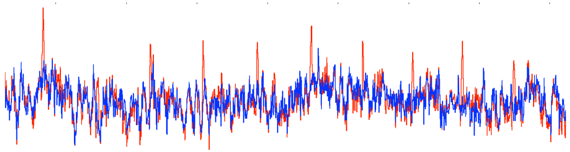
Workshop Schedule

- Tuesday
  - Session 1: Preprocessing, manual and automatic pipelines
  - Session 2: Task data analysis in sensor space (subject level)
  - Session 3: Task data analysis in source space (subject level)
- Thursday
  - Session 4: Task data analysis group level
  - Session 5: Connectivity analysis in source space, subject and group level
  - Session 6: Hidden-Markov-Modeling in rest and task data

Tuesday’s Schedule

09:30 Session 1 lecture
10:00 **Coffee Break**
10:10 Session 1 practical
11:30 Session 2 lecture
12:00 **Lunch (not provided)**
13:00 Session 2 practical
14:20 **Coffee Break**
14:30 Session 3 lecture
15:00 Session 3 practical
16:30 Finish

OSL Preprocessing



Robert Becker

OSL workshop

OHBA, Oxford

25.04.17

UNIVERSITY OF OXFORD

OHBA
Oxford Centre for Human Brain Activity

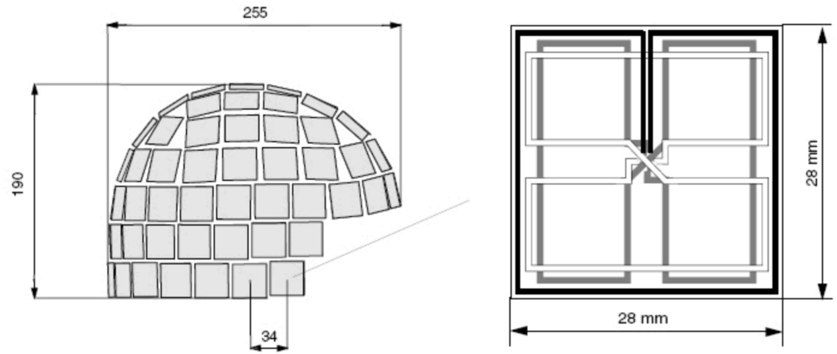
OSL = OHBA's software library

OSL is written in matlab but uses also a number of different toolboxes:

- SPM
- fieldtrip
- FSL
- osl-core (incl. OPT, OAT, ROInets, GLEAN, HMM-MAR etc)
- Utilities
- …

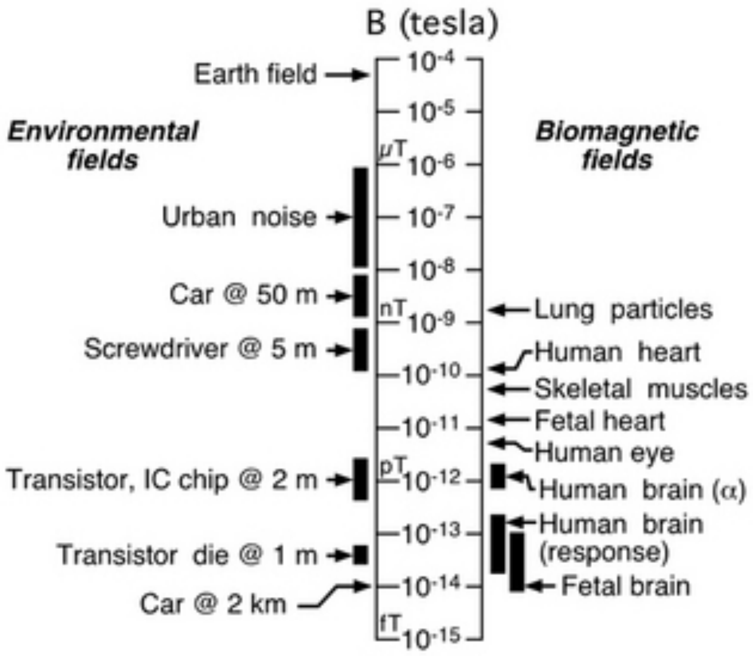
Provides complete pipeline to process and analyse your MEG data

MEG basics: How we measure



- Elekta Neuromag system has 306 helium-cooled SQUID sensors: 102 radial magnetometers and 204 planar gradiometers
- Magnetometers measure magnetic field strength perpendicular to surface of sensors (z-axis), gradiometers measure magnetic field changes (x and y direction)
- Has also EEG recording capabilities + ECG and EOG

MEG and artefacts: What we (want to) measure



This !

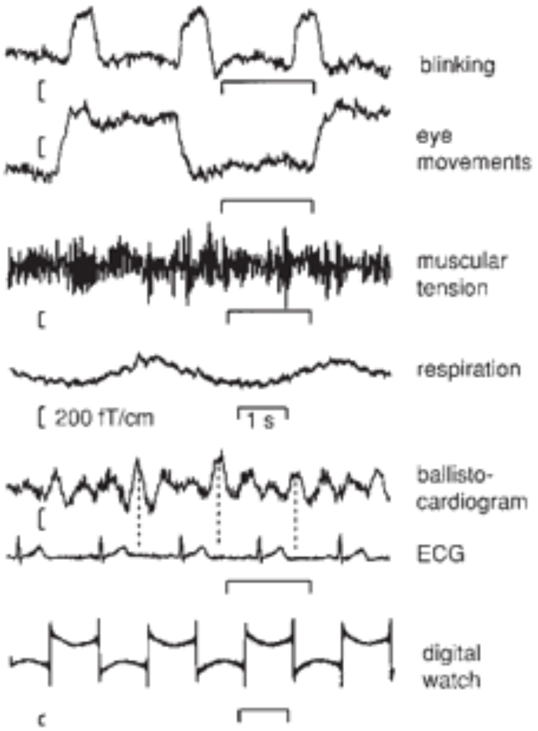
Session 1: Overview

- Introduction: typical MEG artefacts
- Manual Preprocessing:
  - Using the maxfilter
  - Filtering and downsampling
  - Visual Inspection of data: OSLview
  - De-noising data: artefact rejection by using independent component analysis (AFRICA)
- Automated Preprocessing:
  - Using OSL's preprocessing tool (OPT)

MEG and artefacts: Common sources

- Biological artifacts
  - Saccades, blinks, microsaccades
  - Muscular artefacts
  - Heartbeat
  - Respiration

- Electrical/other
  - 50 Hz line noise
  - Scanner artifacts (jumps, spikes)
  - Channel saturation
  - MRI magnetisation
  - Subject movements

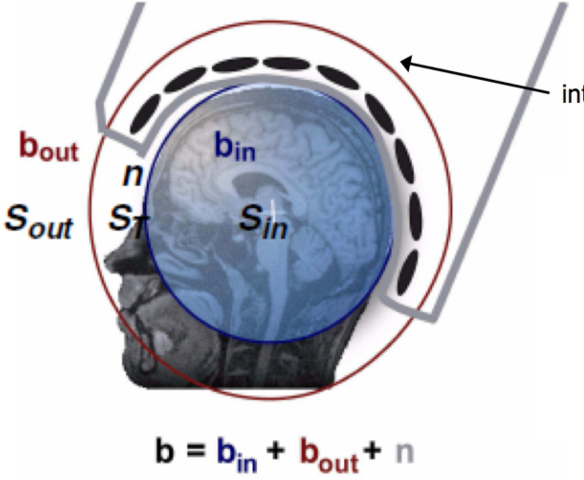


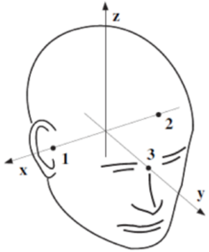
Strategies for dealing with artefacts

- Before and during acquisition:
  - Avoid them (eye blinks, movements etc.)!
- Post-acquisition:
  - Maxfilter noise suppression
  - Filter out problematic frequencies
  - Removing bad periods by visual inspection
  - Use ICA to remove problematic components
  - --> the more you know about them, the better, e.g by recording ECG and EOG (see later)

Manual preprocessing: Maxfilter

- Maxfilter is proprietary software provided by Elekta for their scanner, which implements a signal space separation (SSS) algorithm to reduce external noise (b_{out}):





Maxfilter can also do:

Movement compensation
Alignment
Detect bad channels
Downsample data

Manual preprocessing: Filtering and downsampling

- Efficient for removing of low-frequency drifts and line-noise

- Downside: Might distort your signal of interest:
  - Task data: Phase, latency of evoked fields
  - Rest data: Connectivity measures?

- → Conservative use of filtering; always inspect your data after filtering!

- Downsampling: optional

Manual preprocessing: Visual inspection for bad periods

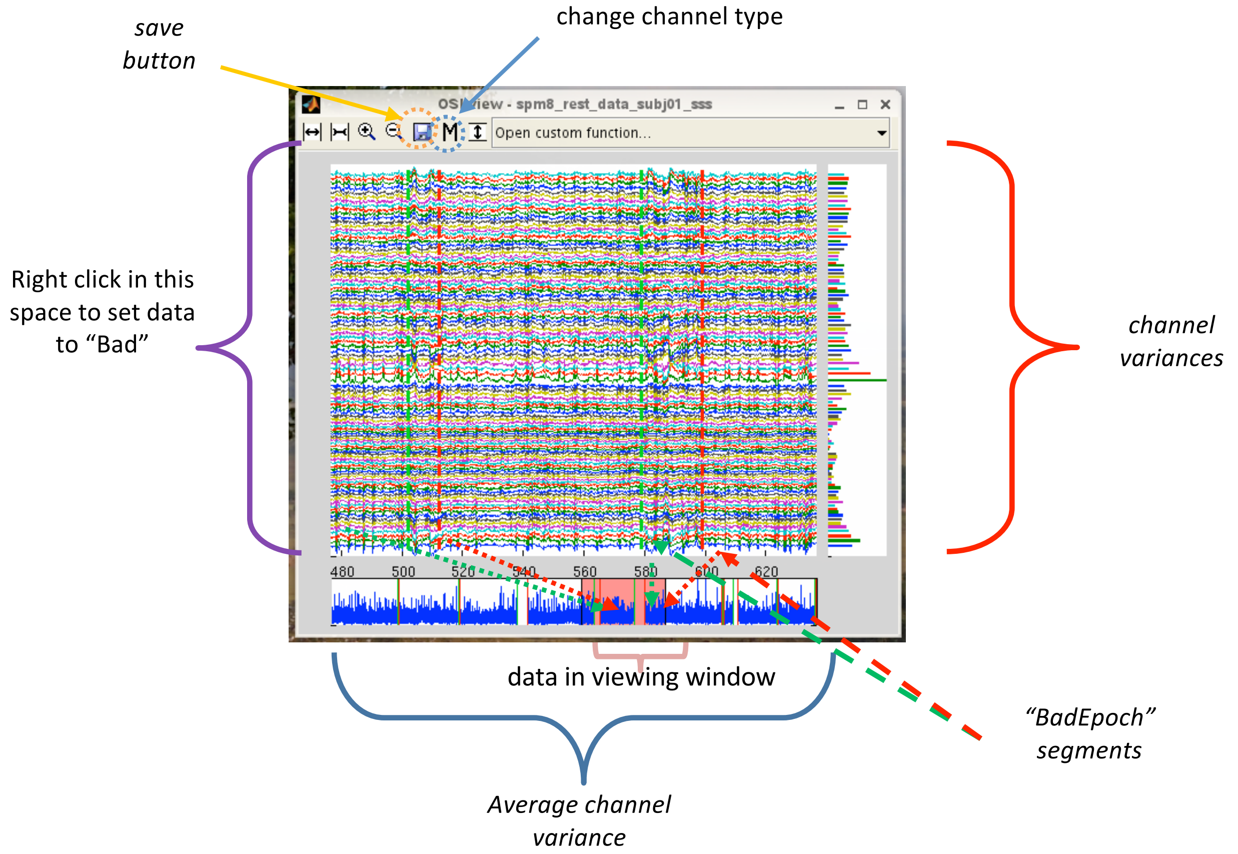
- Important step during manual preprocessing

- But: Equally important to double-check performance during automatic preprocessing!

- Downside: Complete loss of rejected data (channels / periods / trials)

- We will use our OSLview data viewer to do this:

Manual preprocessing: Visual inspection with OSLview

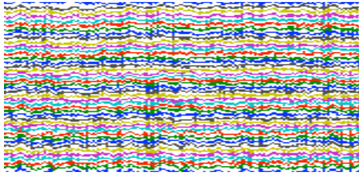


Manual preprocessing: **AfRICA**: **A**rte**f**act **r**ejection using **I**ndependent **C**omponent **A**nalysis (AfRICA)

ICA is a data-driven BBS algorithm to split our MEG data (Y)

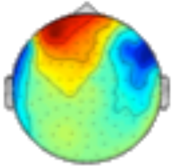
$N_{samples}$

$N_{channels}$ **Y**



into a number of temporally independent components (S) by identifying the mixing matrix (A).

$$= A \times S$$

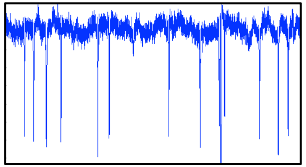


$N_{components}$

$N_{channels}$ **A** x

$N_{samples}$

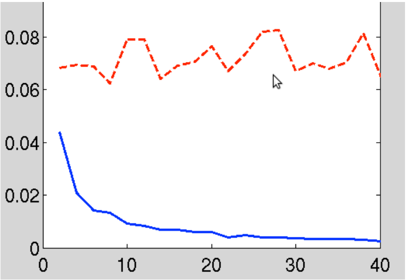
$N_{components}$ **S**



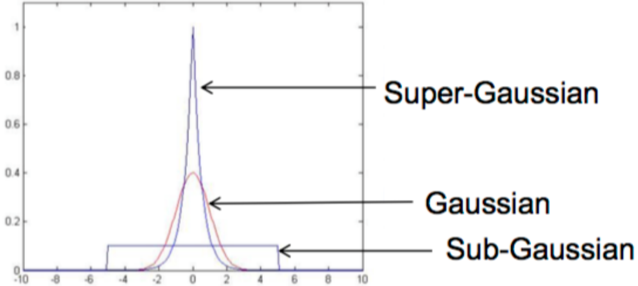
How ICA works

The key ingredient: A mixture of signals is always **more Gaussian** than the underlying signals (aka the central limit theorem).

Non-Gaussianity
(kurtosis)



Number of mixed signals ↑

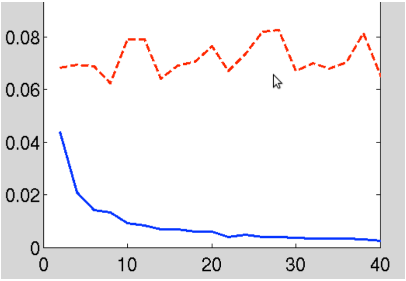


- Means: By searching for the set of maximally non-Gaussian signals we can reverse the mixing process and recover our unknown sources.

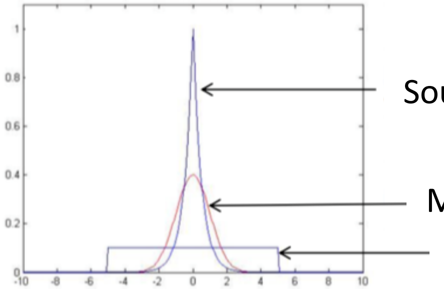
How ICA works

The key ingredient: A mixture of signals is always **more Gaussian** than the underlying signals (aka the central limit theorem).

Non-Gaussianity (kurtosis)



Number of mixed signals ↑



- Means: By searching for the set of maximally non-Gaussian signals we can reverse the mixing process and recover our unknown sources.
- Does not work with Gaussian sources …

AfRICA: Classifying components

- Now, having independent components is great, but which of the sources are artefacts?

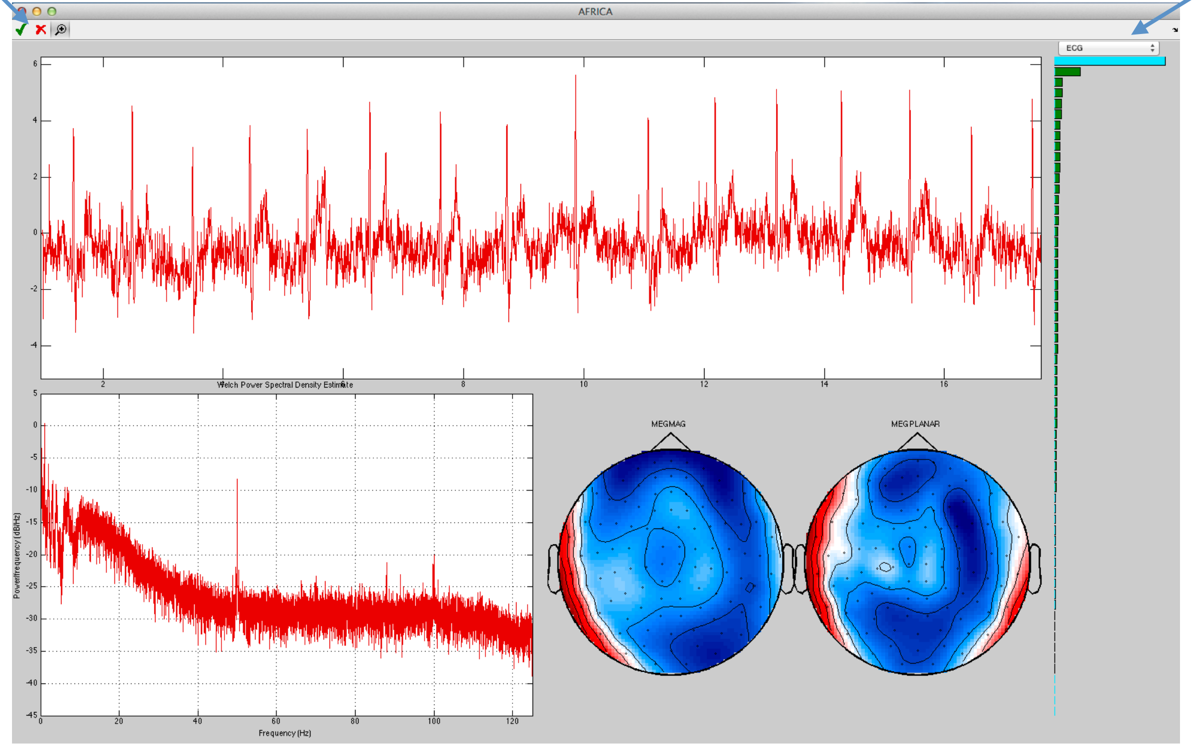
- AfrICA offers ways to guide you:
    - Visual inspection
        - Time course, spectrum, topography
    - Correlation with external signals
        - If you have acquired ECG or EOG then AfRICA will sort the indepent components according to their similarity

AfrICA: User interface

Set component as bad (or revert to good)

metrics (e.g. kurtosis, ECG, EOG, mains)

time course



power spectrum

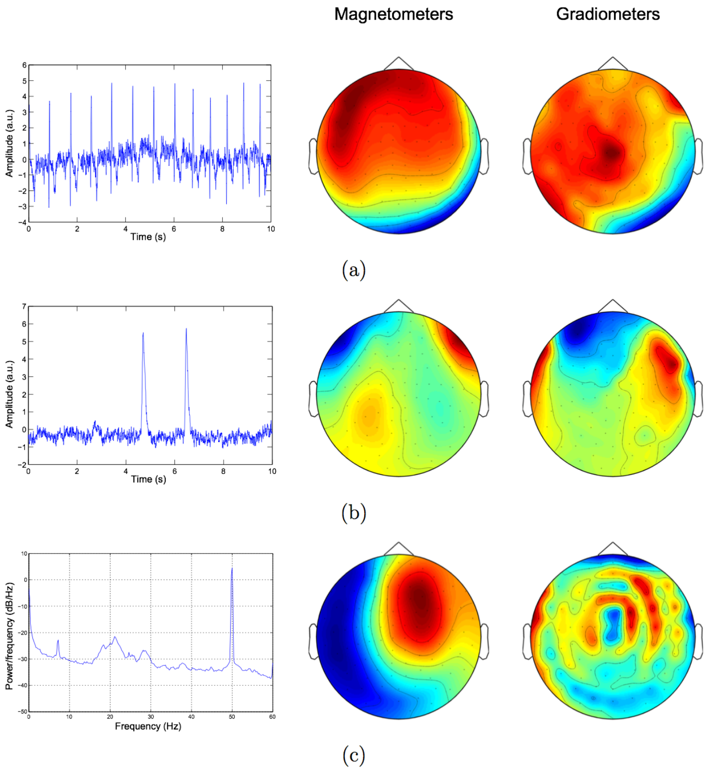
sensor topographies

AfrICA: Typical components

Cardiac component

Eye blink component

Mains (50 Hz) component



AfrICA: Removing components

- Once we know the contribution of the bad components

$$Y_{artif} = A_{artif} * S_{artif}$$

to our data, we can simply remove them:

$$Y_{clean} = Y - Y_{artif}$$

- This linear operation can be rewritten as a matrix operation from Y to Y_{clean}
- → Whenever you want, you can simply transform your original data *Y* into Y_{clean} by applying that matrix multiplication on the fly, i.e. online, no need for a new data set!
- That’s called an ‘online montage’!

Your output after running the manual pipeline

- You should have created new files with a prefix indicating the preprocessing performed on the data, .e.g
  - ‘f’ for filtering
  - ‘d’ for downsampling
  - ‘A’ for the AFRICA denoised data
  - ‘e’ for epoching
- Same naming convention for the automated OPT pipeline!

Automatic preprocessing with OPT

- Wouldn’t it be great to do all the previous steps just automatically and lean back?

- OSL’s preprocessing tool (OPT) allows you to do that!

OPT’s fully automated pipeline:

OPT runs through the following pipeline steps (any of which can be optionally turned off):

- Elekta Neuromag data: Runs the "Double Maxfilter Procedure"
- Conversion of data into SPM format
- Downsampling
- Filtering (hi-pass, notch, …)
- Marking bad segments
- Automated AFRICA denoising
- Coregistration (needed if intending to do subsequent analysis in source space)
- Epoching (If appropriate)
- Automated outlier trial and channel rejection

OPT data input

Data can be input as:

- Either (only for Elekta Neuromag data):
- - the full path of the raw fif files (pre-SSS) to pass to the Maxfilter
- Or:
- - the full path of the input files that will be passed to the SPM convert function (for Elekta Neuromag data this will be post-SSS .fif files
- Or:
- - the full path of the (already converted) SPM MEEG files

Run OPT

Use osl_check_opt call to setup an OPT struct:

- *opt= osl_check_opt(opt);*
- Requires only limited mandatory settings
- Fills other field with default values (which can then be adjusted before running)

Use osl_run_opt to run an OPT:

- *opt=osl_run_opt(opt);*

OPT output

- Results are stored in the directory specified in opt.dirname, with a ‘.opt’ suffix
- *opt=osl_run_opt(opt)* returns a results field: *opt.results*

- This contains:
- *opt.results.logfile* (file containing the matlab text output)
- *opt.results.report*: (Web page report with diagnostic plots)
- *opt.results.spm_files*: (list of SPM MEEG object files for the continuous data, e.g. to pass into an OAT analysis)
- *opt.results.spm_files_epoched*: (list of SPM MEEG object files for the epoched data, e.g. to pass into an OAT analysis)

Today’s practicals

- Practicals + data are on the OSL Wiki

- Practical corresponds to this lectures and has two parts:

- 1) Manual Preprocessing Pipeline

- 2) Automated Preprocessing Pipeline (OPT)

Recommended reading

- Look at and use the OSL Wiki!
- Independent Component Analysis (easy, but a whole book)
  - Independent Component Analysis – A Tutorial Introduction – James V. Stone
- fastICA & ICASSO (advanced)
  - Hyvärinen, A., 1999. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. Neural Netw. 10 (3), 626–634.
- ICA de-noising in MEG (relevant)
  - Mantini, D., et al. 2011. A Signal-Processing Pipeline for Magnetoencephalography Resting-State Networks. Brain Connectivity, 1(1), 49–59.
- I want a demo!
  - Finnish cocktail party here: https://research.ics.aalto.fi/ica/cocktail/cocktail_en.cgihttps://research.ics.aalto.fi/ica/cocktail/cocktail_en.cgi