MNE

MEG + EEG ANALYSIS & VISUALIZATION

History - Overview - Resources - Workflow

http://martinos.org/mne

http://perso.telecom-paristech.fr/~gramfort/mne/halifax2016/

MNE software for processing MEG and EEG data, A. Gramfort, M. Luessi, E. Larson, D. Engemann, D. Strohmeier, C. Brodbeck, L. Parkkonen, M. Hämäläinen, Neuroimage, 2014

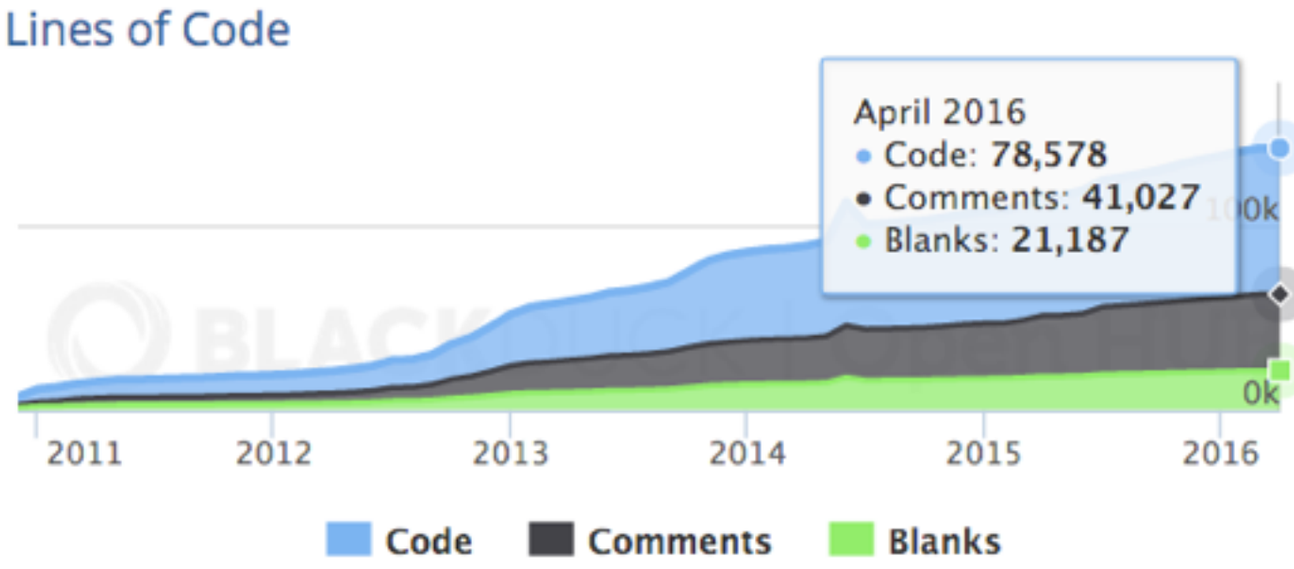
MEG and EEG data analysis with MNE-Python, A. Gramfort, M. Luessi, E. Larson, D. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, M. Hämäläinen, *Frontiers in Neuroscience, 2013*

What's the plan today

- MNE workflow + MNE "jargon"
- Sensor space analysis: Basic preprocessing and visualization
- Getting up to speed with Python

About the project

- MNE based on C code developed for ~15 years by MSH
- MNE-Python started in Dec. 2010 at MGH, Boston



In a Nutshell, MNE-Python...

... has had 11,232 commits made by 114 contributors representing 78,578 lines of code

... is mostly written in Python with a well-commented source code

... has a well established, mature codebase maintained by a very large development team with increasing Y-O-Y commits

... took an estimated 20 years of effort (COCOMO model) starting with its first commit in December, 2010 ending with its most recent commit 3 days ago

12 Month Summary

Apr 3 2015 — Apr 3 2016

3836 Commits

Up + 1358 (54%) from previous 12 months

63 Contributors

Up + 12 (23%) from previous 12 months

Source: https://www.ohloh.net/p/MNE

People

@agramfort

@mluessi

@Eric89GXL

@dengemann

@joewalter

@OlafHauk

@ jdammers

@christianmbrodbeck

@rgoj

@mainakjas

@t3on

@lauriparkkonen

@TalLinzen

@adykstra

@dgwakeman

@leggitta

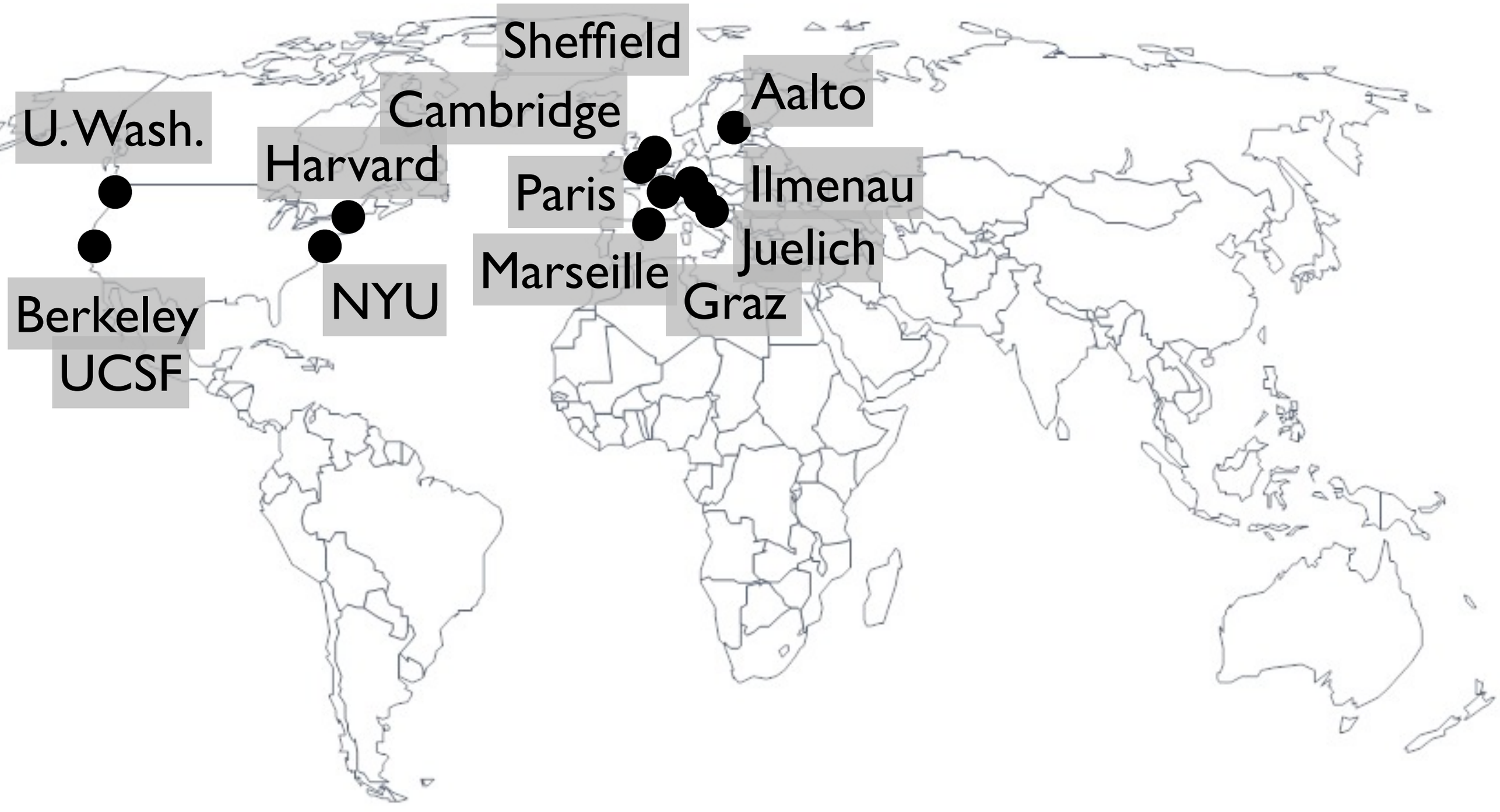
@mshamalainen

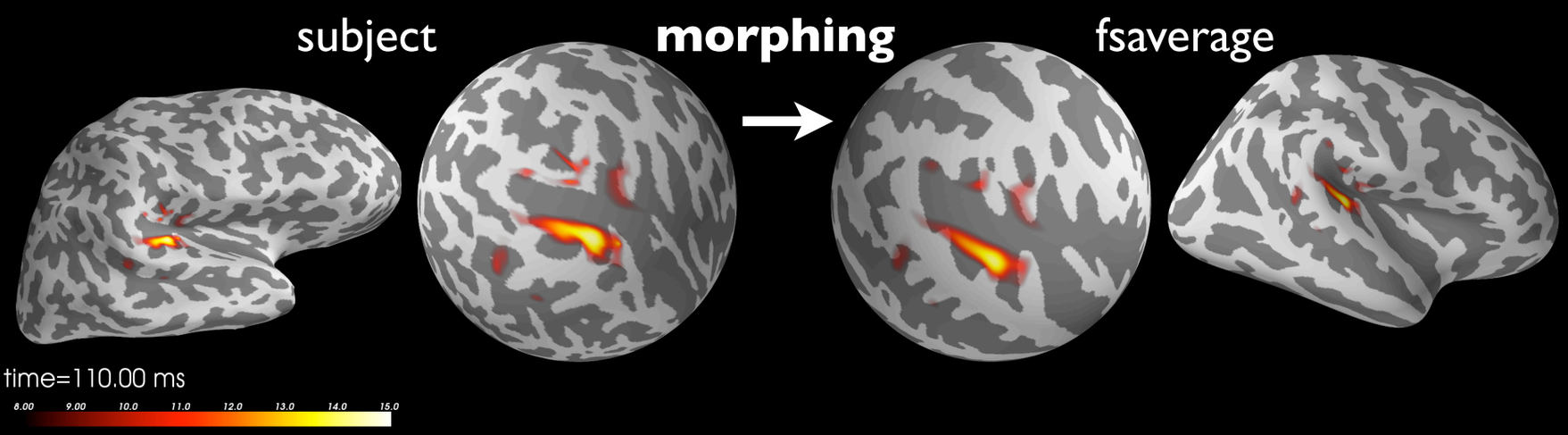
@kazemakase

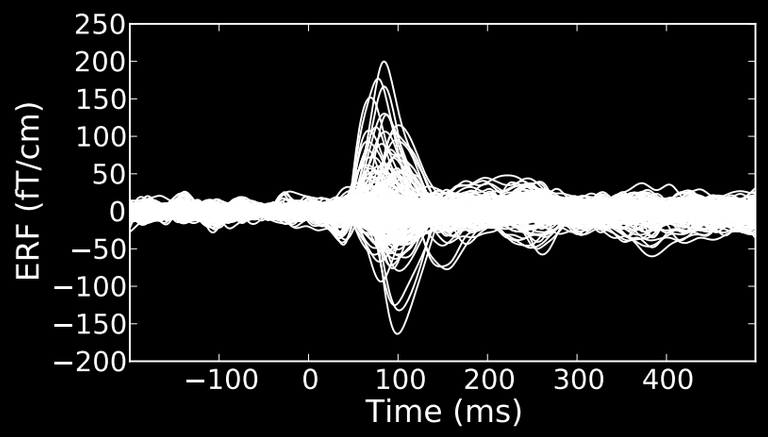
@you?

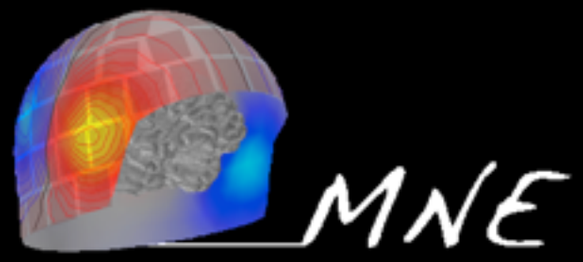
http://martinos.org/mne/stable/whats_new.html

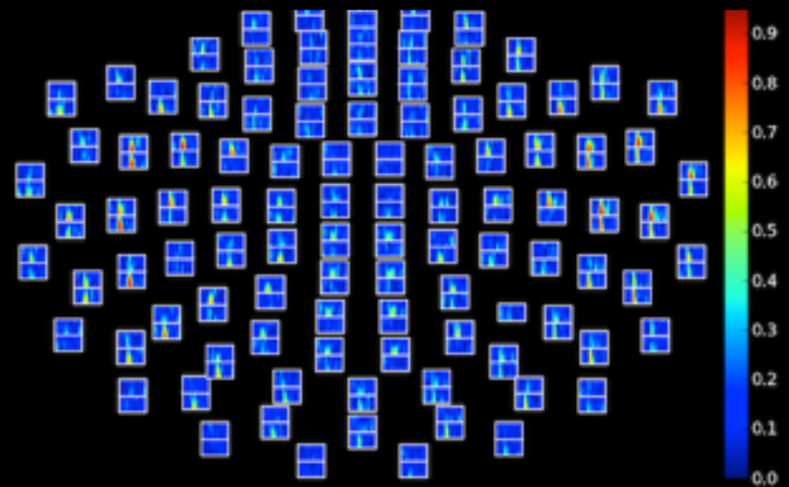
Development of the MNE software in 2016

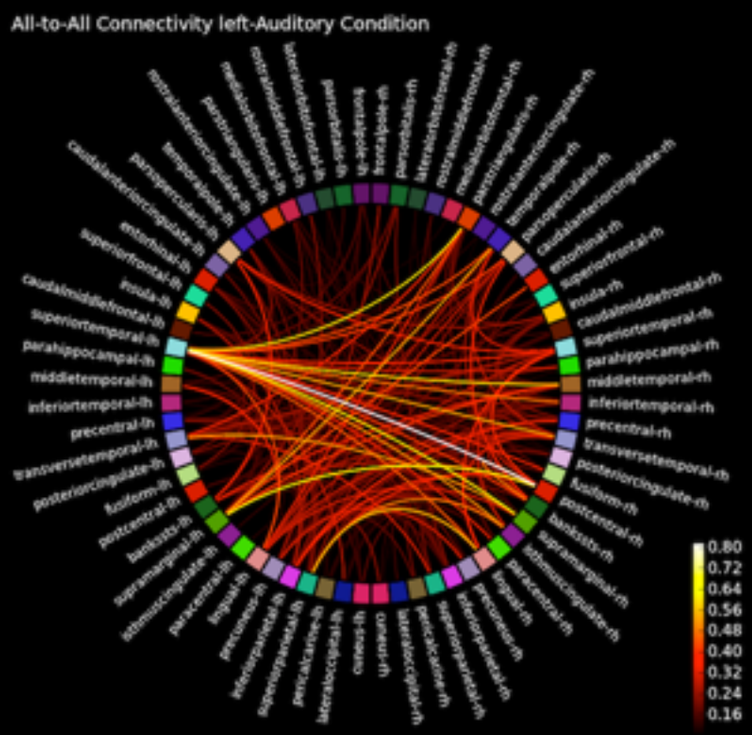


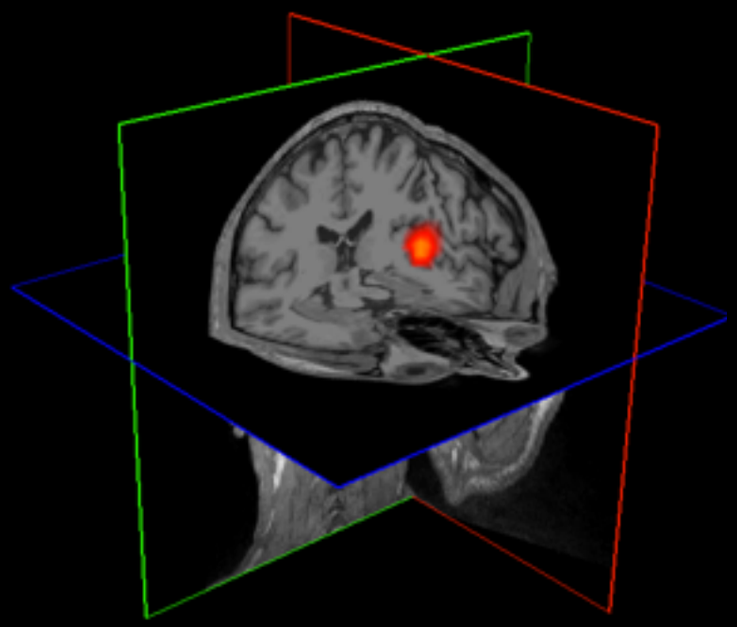

subject
morphing
fsaverage
time=110.00 ms
8.00 9.00 10.0 11.0 12.0 13.0 14.0 15.0

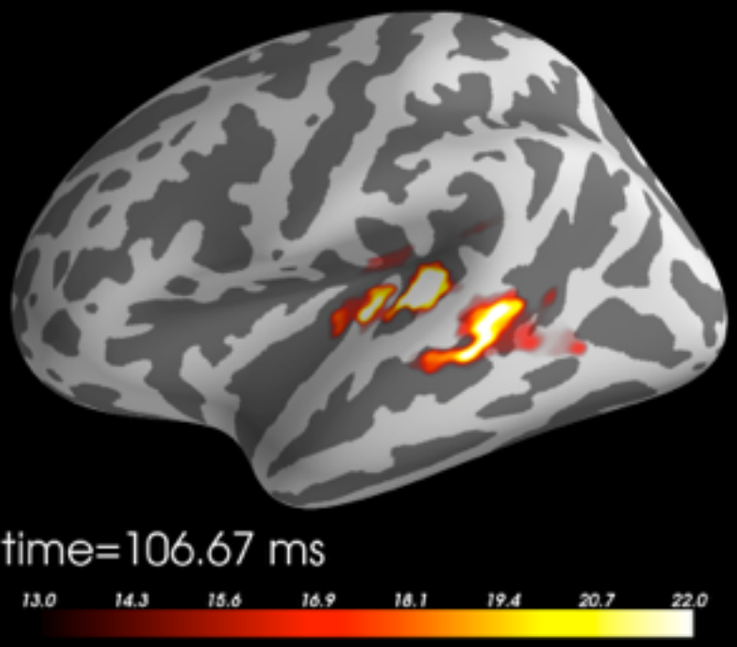

ERF (fT/cm)
250
200
150
100
50
0
−50
−100
−150
−200
−100 0 100 200 300 400
Time (ms)


MNE

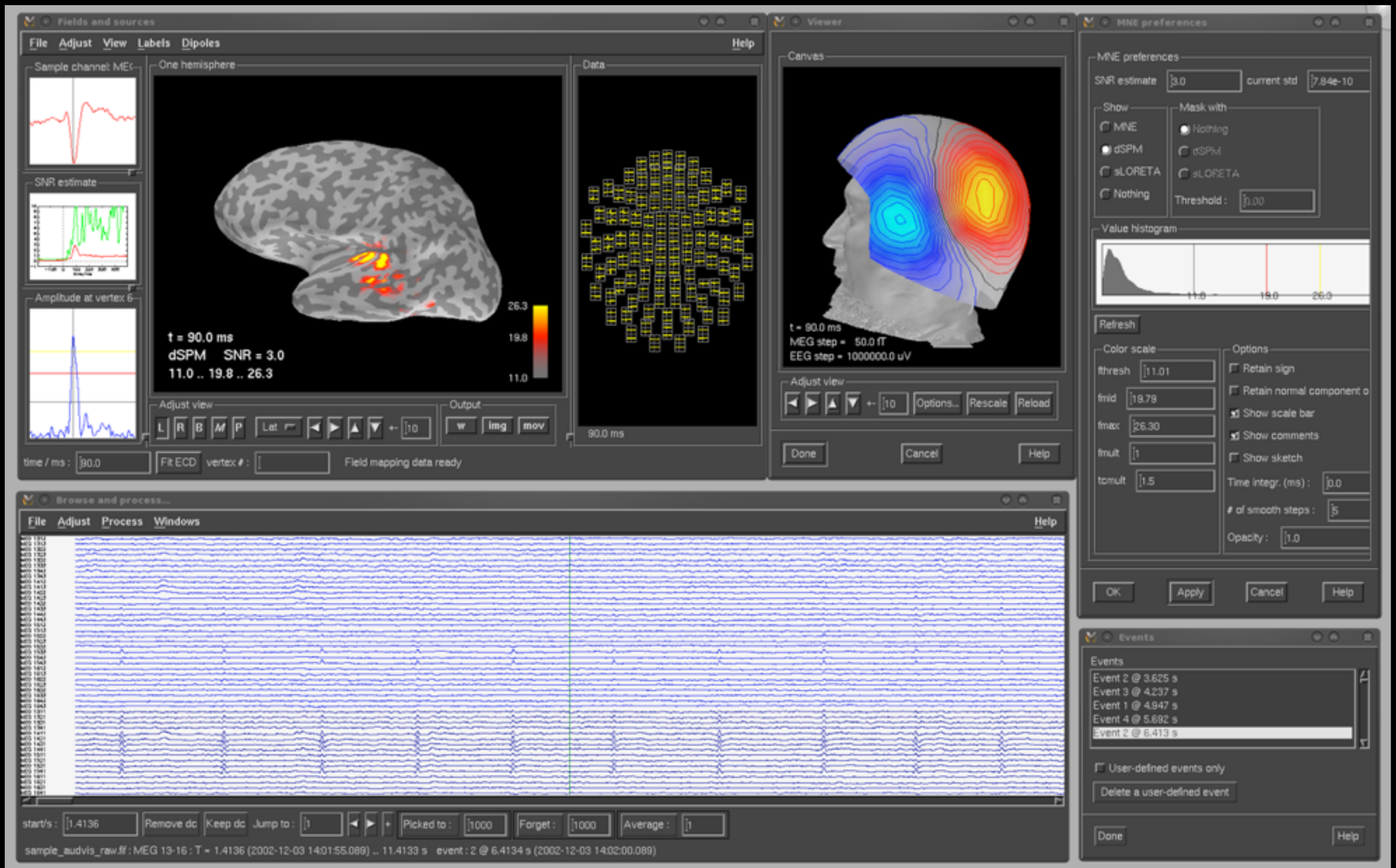

0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0


All-to-All Connectivity left-Auditory Condition
0.80
0.72
0.64
0.56
0.48
0.40
0.32
0.24
0.16



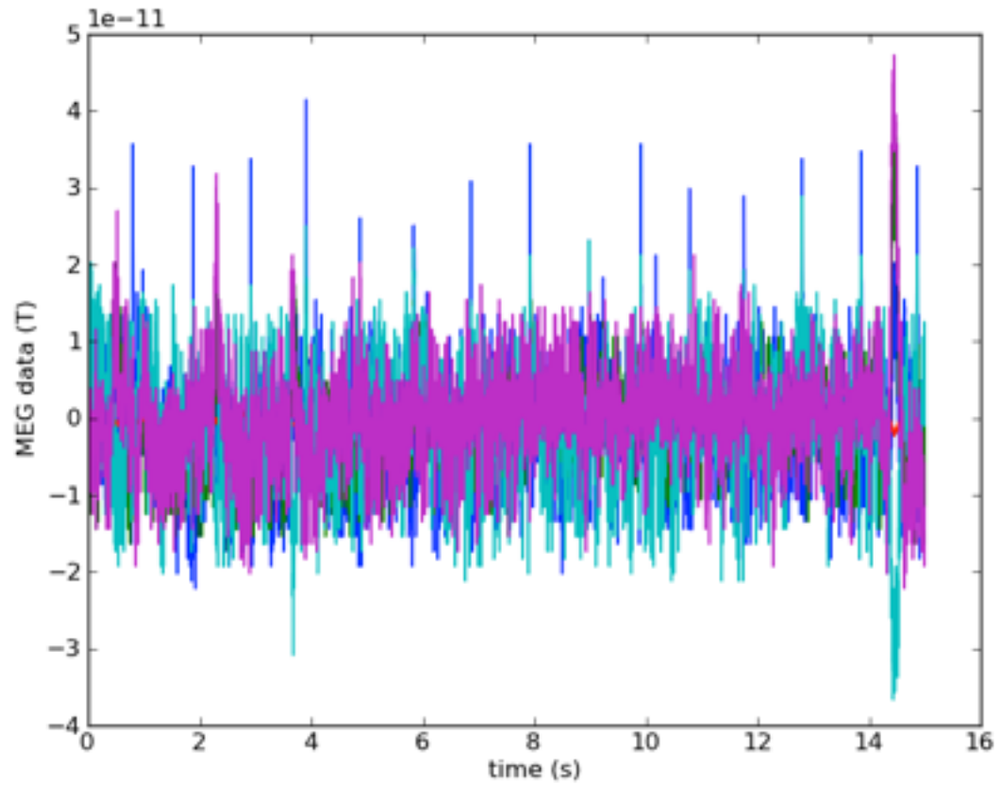

time=106.67 ms
13.0 14.3 15.6 16.9 18.1 19.4 20.7 22.0

MNE user interface



Scripting with Python

```
import matplotlib.pyplot as plt
import mne
raw = mne.io.Raw(fname)

picks = mne.pick_types(raw.info, meg='mag')
start, stop = raw.time_as_index([0, 15])  # read the first 15s of data
data, times = raw[picks[:5], start:(stop + 1)]  # take 5 first channels

plt.plot(times, data.T)
plt.xlabel('time (s)')
plt.ylabel('MEG data (T)')
```



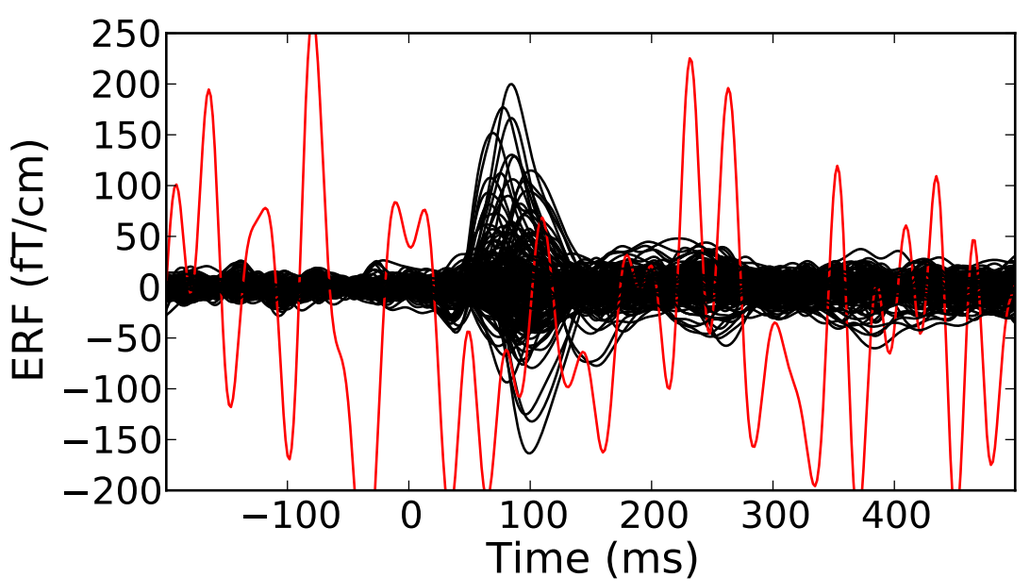
Filter data with Python

```
import mne

fname = 'raw.fif'
raw = mne.io.Raw(fname, preload=True)
```

```
# keep beta band
raw.filter(13.0, 30.0, filter_length=4096, n_jobs=8)
# or use IIR filter (instead of FFT)
raw.filter(13.0, 30.0, method='iir', n_jobs=8)

# save the result
raw.save('beta_raw.fif')
```

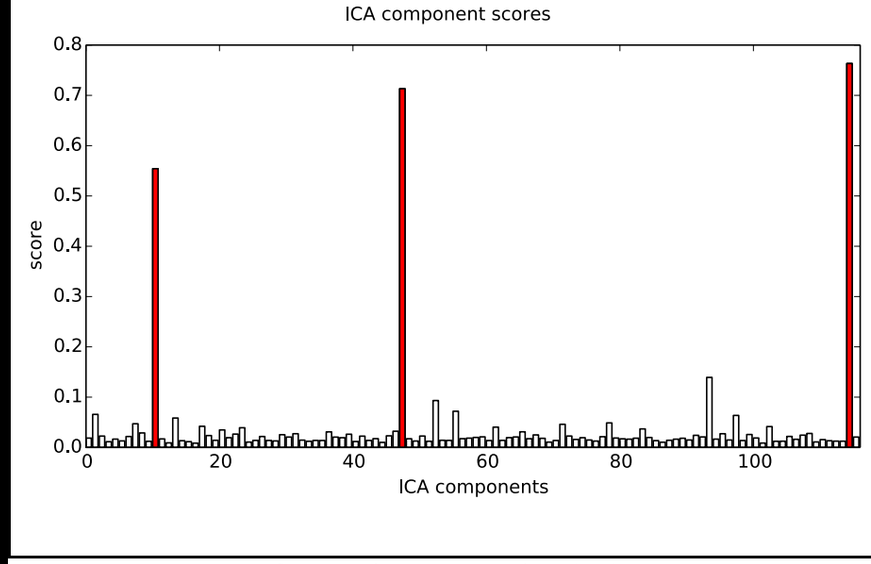
Computing ERF/ERP

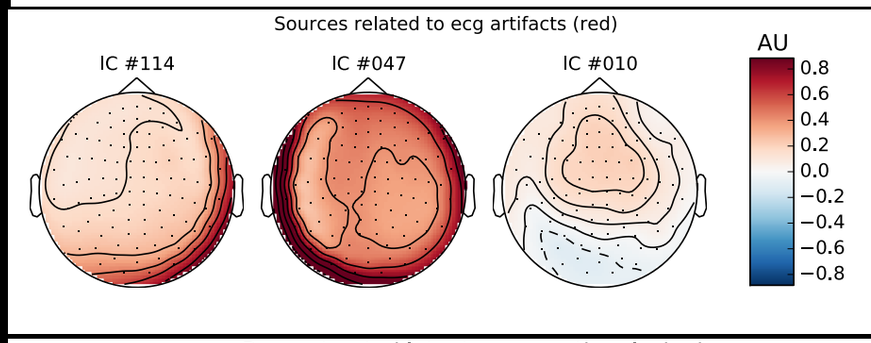
```
import mne

fname = 'raw.fif'
raw = mne.io.Raw(fname)
raw.info['bads'] = ['MEG 2443', 'EEG 053']  # mark bad channels

# extract epochs
picks = mne.pick_types(raw.info, meg=True, eeg=True, eog=True,
                       exclude=raw.info['bads'])
event_id, tmin, tmax = 1, -0.2, 0.5
events = mne.find_events(raw, stim_channel='STI 014')
epochs = mne.Epochs(raw, events, event_id, tmin, tmax, proj=True,
                    picks=picks, baseline=(None, 0), preload=True,
                    reject=dict(grad=4000e-13, mag=4e-12, eog=150e-6))

# compute evoked response
evoked = epochs.average()
# plot
evoked.plot()

# save them
epochs.save('event_%d-epo.fif' % event_id)
evoked.save('event_%d-ave.fif' % event_id)
```

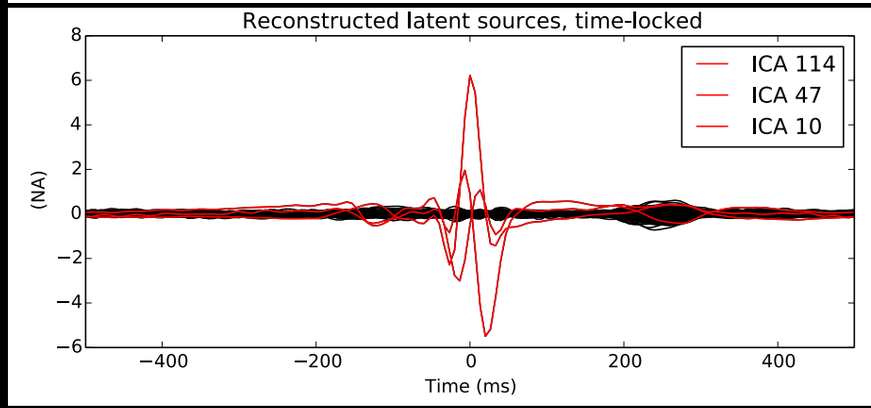


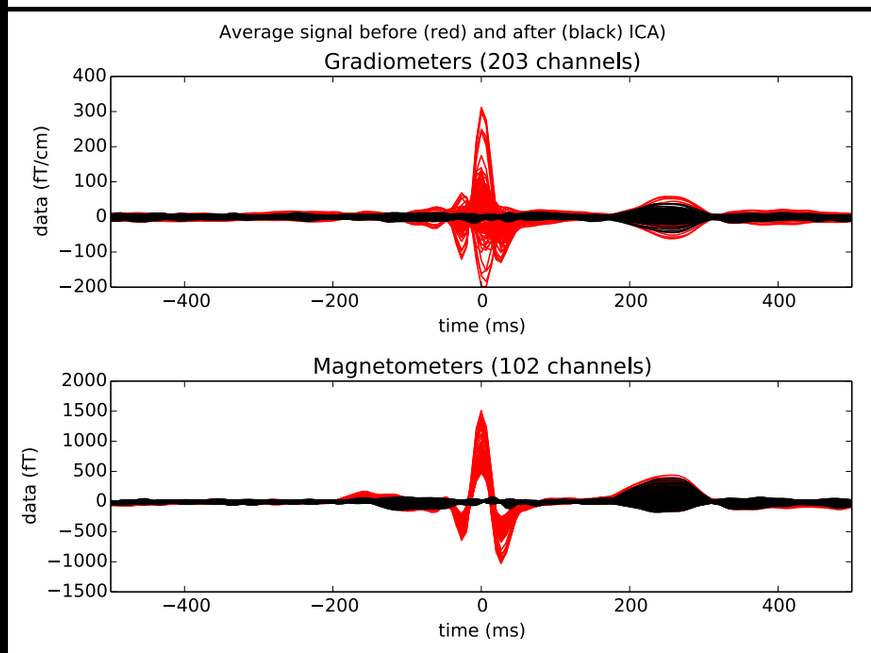
Remove artifacts using ICA

```
# Author: Denis Engemann <denis.engemann@gmail.com>
#
# License: BSD (3-clause)

raw = Raw(raw_fname, preload=True)
raw.filter(1, 30, method='iir')
picks = mne.pick_types(raw.info, meg=True, eeg=False, eog=True, ecg=True,
                       stim=False, exclude='bads')
events = mne.find_events(raw, stim_channel='STI 014')
event_id = dict(aud_l=1, aud_r=2, vis_l=3, vis_r=4)
reject = dict(eog=250e-6)
tmin, tmax = -0.5, 0.5

# 1) Fit ICA model using the FastICA algorithm
ica = ICA(n_components=0.95, method='fastica').fit(epochs)

# 2) Find ECG Artifacts
# generate ECG epochs to improve detection by correlation
ecg_epochs = create_ecg_epochs(raw, tmin=-.5, tmax=.5, picks=picks)
ecg_inds, scores = ica.find_bads_ecg(ecg_epochs)
ica.plot_scores(scores, exclude=ecg_inds)

title = 'Sources related to %s artifacts (red)'
ica.plot_components(ecg_inds, title=title % 'ecg')

# by default we expect 3 reliable ECG components
ica.exclude += ecg_inds[:3]

# 3) Assess component selection and unmixing quality
ecg_evoked = ecg_epochs.average() # estimate average artifact
ica.plot_sources(ecg_evoked)  # plot ECG sources + selection
ica.plot_overlay(ecg_evoked)  # plot ECG cleaning
```



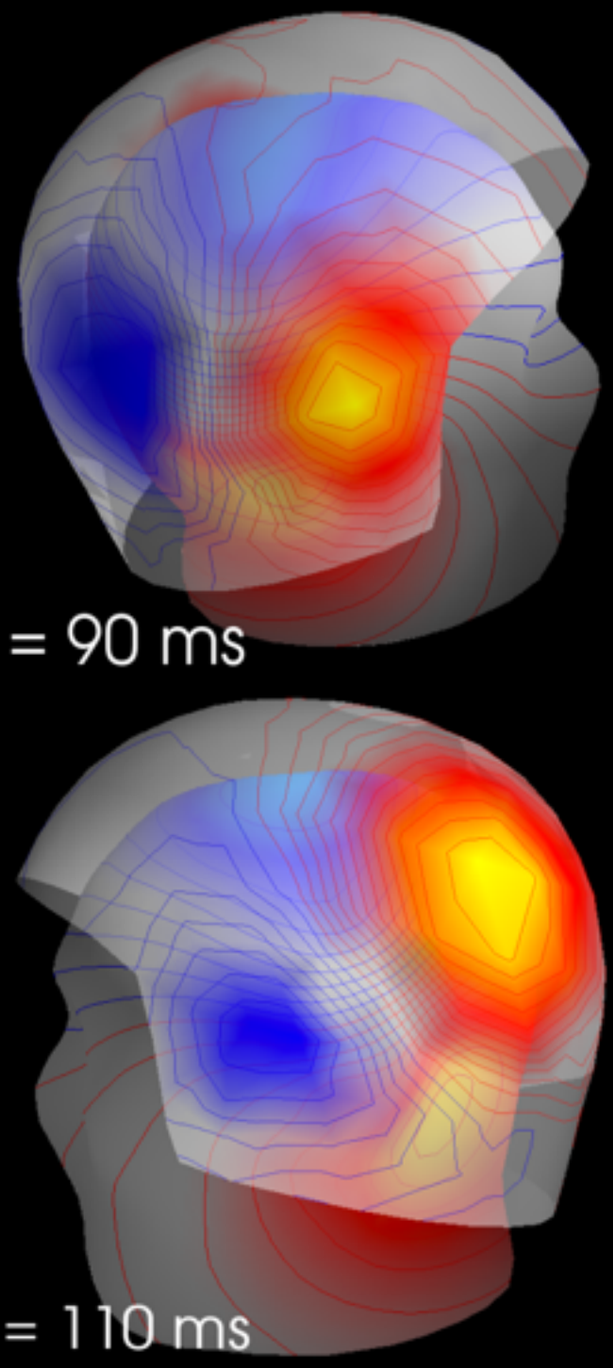






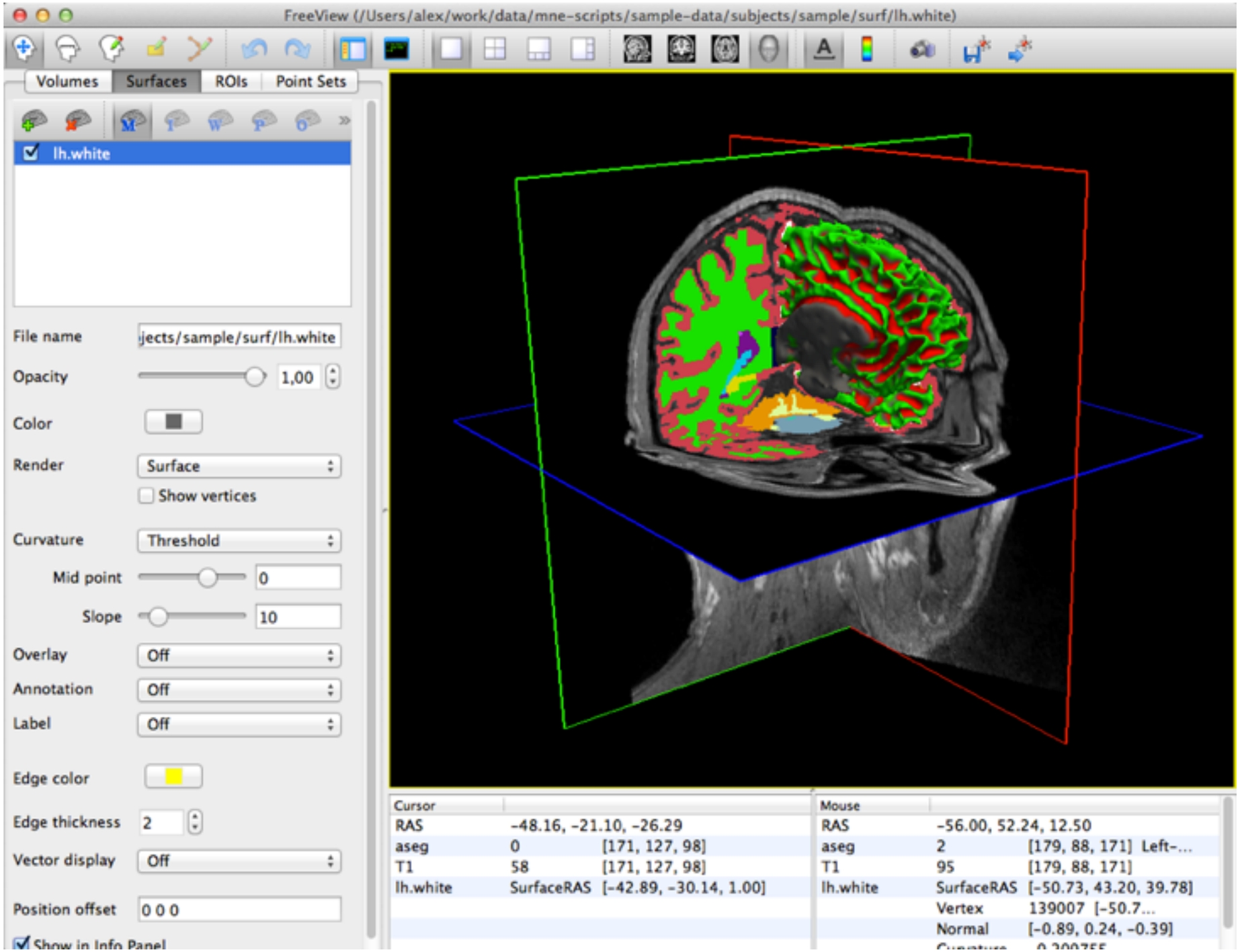
Project multiples sensor types on field map

```
"""
======================
Plot M/EEG field lines
======================

In this example, M/EEG data are remapped onto the
MEG helmet (MEG) and subject's head surface (EEG).
This process can be computationally intensive.
"""

# Authors: Eric Larson <larson.eric.d@gmail.com>
#          Denis A. Engemann <d.engemann@fz-juelich.de>
#          Alexandre Gramfort <alexandre.gramfort@telecom-paristech.fr>
# License: BSD (3-clause)
import mne

data_path = mne.datasets.sample.data_path()
subjects_dir = data_path + '/subjects'
evoked_fname = data_path + '/MEG/sample/sample_audvis-ave.fif'
trans_fname = data_path + '/MEG/sample/sample_audvis_raw-trans.fif'

# If trans_fname is set to None then only MEG estimates can be visualized
setno = 'Left Auditory'

evoked = mne.read_evoked(evoked_fname, setno=setno,
                         baseline=(-0.2, 0.0))

# Compute the field maps to project MEG and EEG data to MEG helmet
# and scalp surface
maps = mne.make_field_map(evoked, trans_fname=trans_fname,
                          subject='sample', subjects_dir=subjects_dir,
                          n_jobs=1)

# explore several points in time
[evoked.plot_field(maps, time=time) for time in [0.09, .11]]
```



Built on top of FreeSurfer

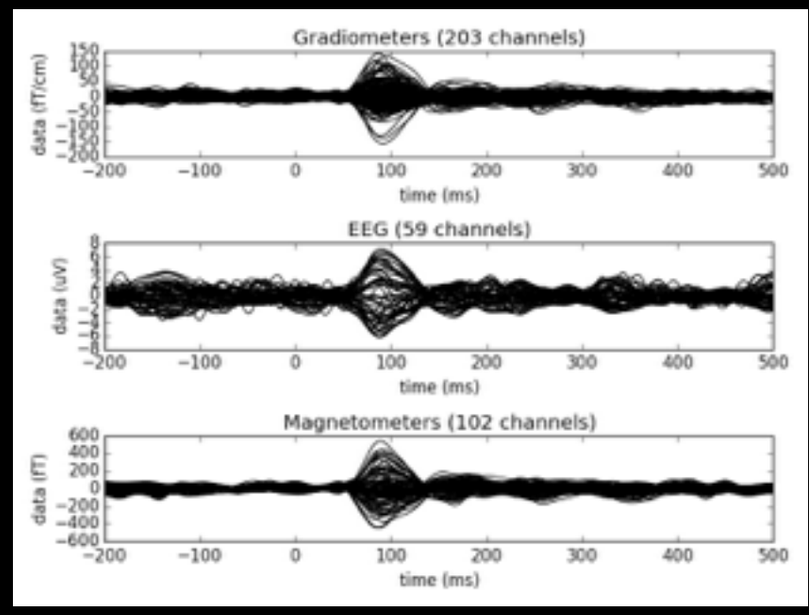
```
$   recon-all -s ${SUBJECT} -i xy000000.nii -all
```

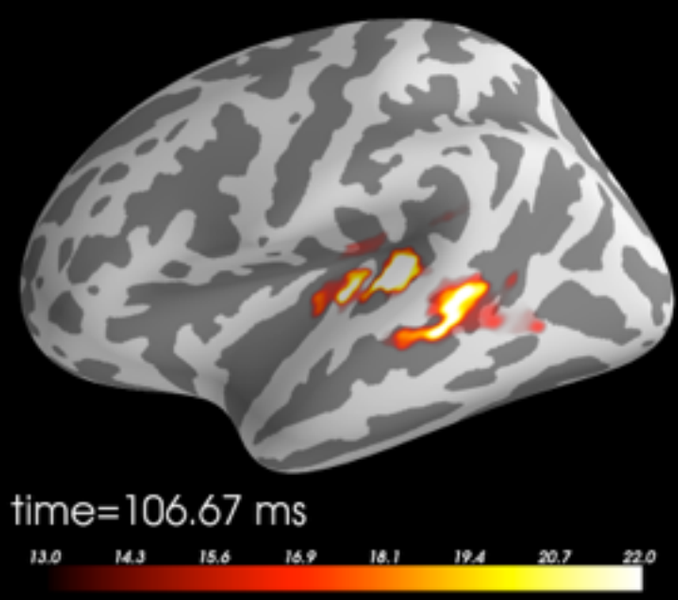


http://surfer.nmr.mgh.harvard.edu/

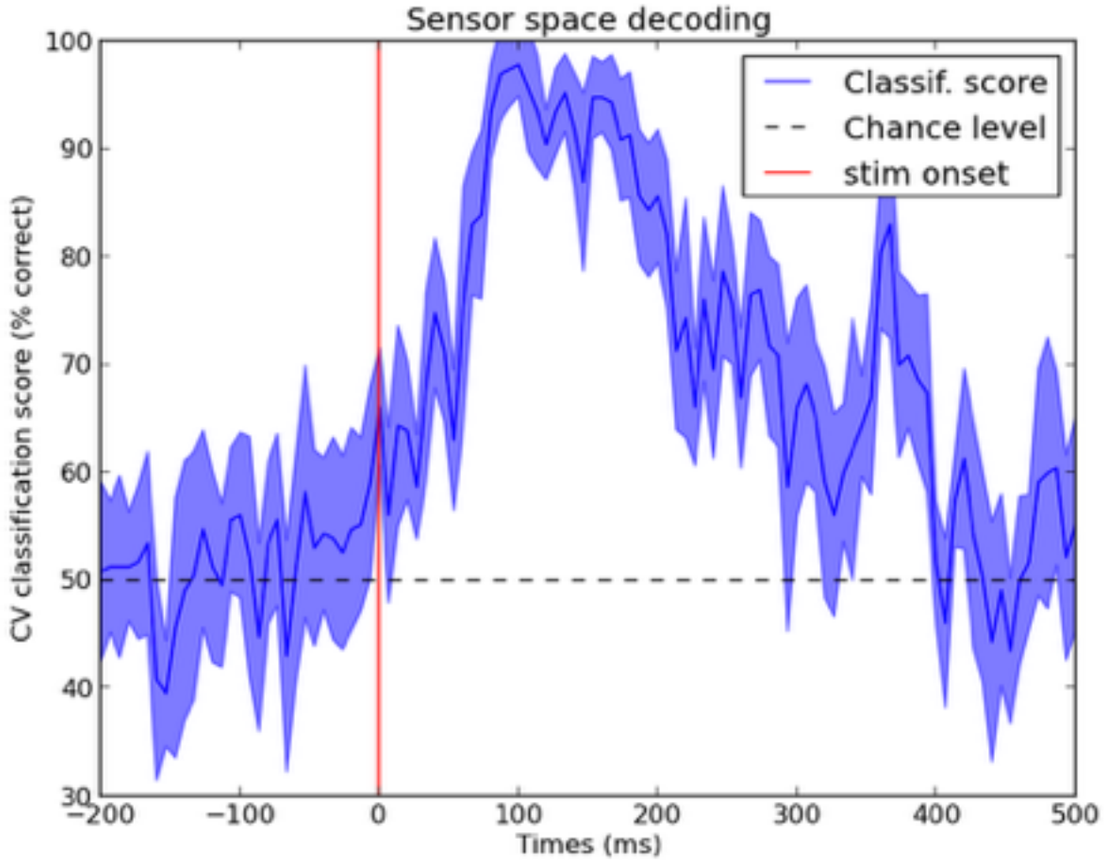
From raw data to dSPM solutions in ~30 lines

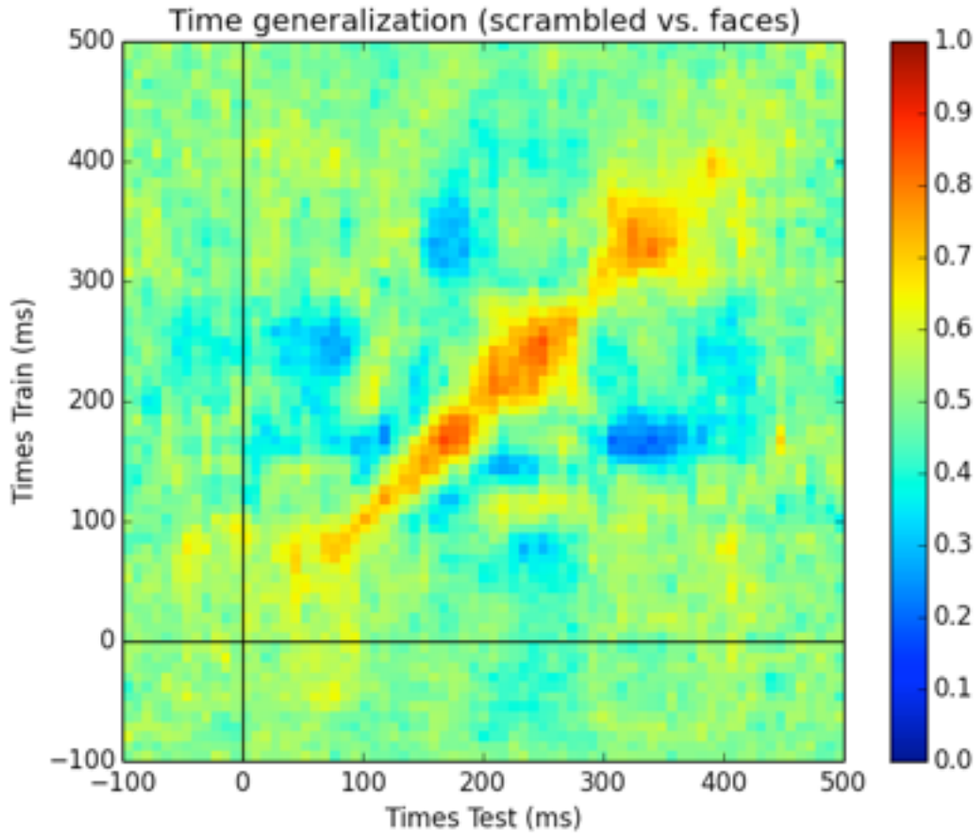
```
import mne

# load data
fname = 'raw.fif'
raw = mne.io.Raw(fname)
raw.info['bads'] = ['MEG 2443', 'EEG 053']  # mark bad channels

# band-pass filter data in beta band, and save it
raw.filter(13.0, 30.0, filter_length=4096, n_jobs='cuda')
raw.save('beta_raw.fif')

# extract epochs
picks = mne.pick_types(raw.info, meg=True, eeg=True, eog=True)
events = mne.find_events(raw)
epochs = mne.Epochs(raw, events, event_id=1, tmin=-0.2, tmax=0.5, proj=True,
                    picks=picks, baseline=(None, 0), preload=True,
                    reject=dict(grad=4000e-13, mag=4e-12, eog=150e-6))

# compute evoked response and noise covariance, and plot evoked
evoked = epochs.average()
cov = mne.compute_covariance(epochs, tmax=0)
evoked.plot()

# compute inverse operator
fwd_fname = 'sample_audvis-meg-eeg-oct-6-fwd.fif'
fwd = mne.read_forward_solution(fwd_fname, surf_ori=True)
inv = mne.minimum_norm.make_inverse_operator(raw.info, fwd, cov, loose=0.2)

# compute inverse solution
stc = mne.minimum_norm.apply_inverse(evoked, inv, lambda2=1 / 3.0 ** 2,
                                     method='dSPM')

# morph it to average brain for group study
stc_avg = mne.morph_data('sample', 'fsaverage', stc, 5, smooth=5)
stc_avg.plot()
```





Decoding





http://martinos.org/mne/stable/auto_examples/index.html#decoding-mvpa

MNE web report

report.html

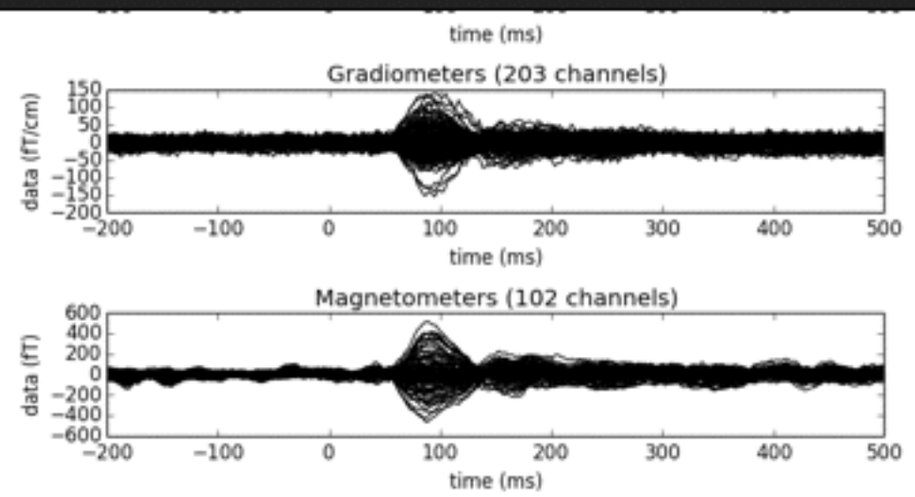
file:///Users/alex/work/data/mne-scripts/sample-data/report.html

Reader

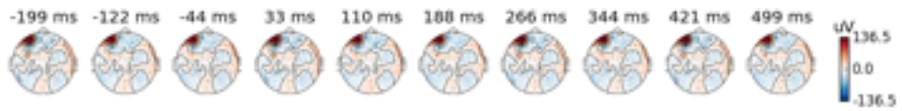
> MNE Report for ...MNE-sample-data

raw | events | evoked | covariance | forward | inverse

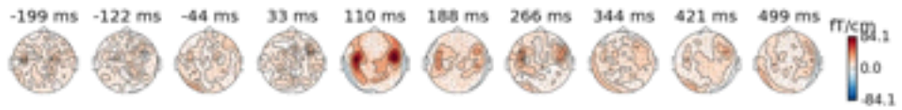
CONTENTS



Topomap (ch_type = eeg)



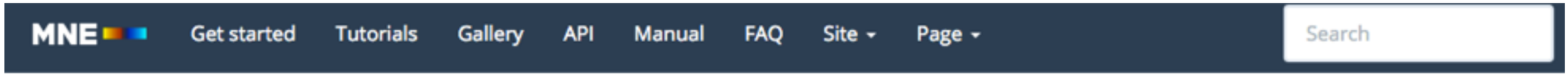
Topomap (ch_type = grad)



https://dl.dropboxusercontent.com/u/3915954/report.html

Getting help

http://martinos.org/mne/



MNE

MEG + EEG ANALYSIS & VISUALIZATION

MNE is a community-driven software package designed for for **processing electroencephalography (EEG) and magnetoencephalography (MEG) data** providing comprehensive tools and workflows for:

1. Preprocessing
2. Source estimation
3. Time–frequency analysis
4. Statistical testing
5. Estimation of functional connectivity
6. Applying machine learning algorithms
7. Visualization of sensor- and source-space data

MNE includes a comprehensive Python package (provided under the simplified BSD license), supplemented by tools compiled from C code for the LINUX and Mac OSX operating systems, as well as a MATLAB toolbox.

Documentation

- Getting Started
- What's new
- Cite MNE
- Related publications
- Tutorials
- Examples Gallery
- Manual
- API Reference
- Frequently Asked Questions
- Advanced installation and setup
- MNE with CPP

Mailing list:

http://mail.nmr.mgh.harvard.edu/mailman/listinfo/mne_analysis

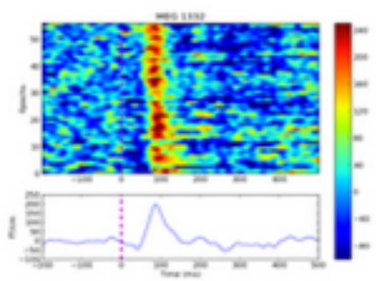
Getting inspired...

http://martinos.org/mne/auto_examples/index.html

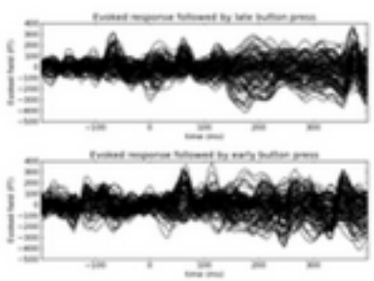
Examples

General examples

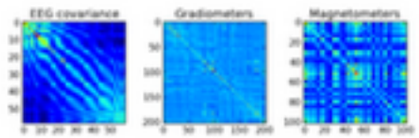
General-purpose and introductory examples to MNE.



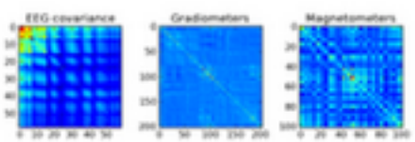
Visualize channel over epochs as an image



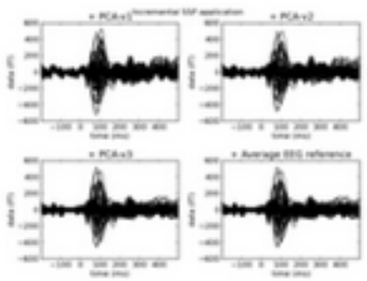
Define target events based on time lag, plot evoked response



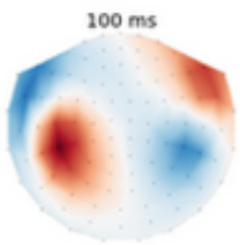
Estimate covariance matrix from Epochs baseline



Estimate covariance matrix from a raw FIF file



Create evoked objects in delayed SSP mode



Plotting topographic maps of evoked data

Previous topic

Tutorial: MEG and EEG data processing with MNE and Python

Next topic

Visualize channel over epochs as an image

Quick search

Sending feedback

https://github.com/mne-tools/mne-python

This repository | Search or type a command | Explore | Gist | Blog | Help | agramfort

mne-tools / mne-python

Unwatch 22 | Unstar 52 | Fork 53

Browse Issues | Milestones | New Issue

Everyone's Issues 30 | Assigned to you 0 | Created by you 7 | Mentioning you 11

No milestone selected

Labels

ENH 11
BUG 0
DOC 0
FIX 0

30 Open | 905 Closed | Sort: Newest

Close | Label | Assignee | Milestone

ENH/FIX: default EEG layout when no headshape ore coil positions are present #926
Opened by dengemann 2 days ago 8 comments

Segmentation fault on unit test #922
Opened by lulopolar 2 days ago 13 comments

BUG: coverage is not working with Travis .... #919
Opened by dengemann 4 days ago

ENH: Improve test coverage #909
Opened by Eric89GXL 7 days ago 11 comments

DOC: setting up Python #900
Opened by christianmbrodbeck 11 days ago 5 comments

So I know you better...

- Who knows what MEG is?
- Who knows what a FIF file is?
- Who has ever used mne_analyze?
- Who knows matlab? Fieldtrip/Brainstorm/SPM?
- Who has used dSPM? LCMV? DICS? Dipole fit?
- Who knows the difference between a gradiometer and a magnetometer?

MNE-C setup (bash)

```
$  cd MNE-2.7.4-3378-MacOSX-x86_64
$  export MNE_ROOT=${PWD}
$  . ${MNE_ROOT}/bin/mne_setup_sh
$  export SUBJECTS_DIR=/path/fs/recons
$  export SUBJECT=sample
```

You should have a folder MNE-sample-data

Check your setup with $ mne_analyze --help

MNE-C setup (tcsh)

```
$  cd MNE-2.7.4-3378-MacOSX-x86_64
$  setenv MNE_ROOT ${PWD}
$  source ${MNE_ROOT}/bin/mne_setup
$  setenv SUBJECTS_DIR /path/fs/recons
$  setenv SUBJECT=sample
```

MNE-Python setup

Assuming your scientific Python environment is setup:

```
$  pip install --upgrade mne
```

or

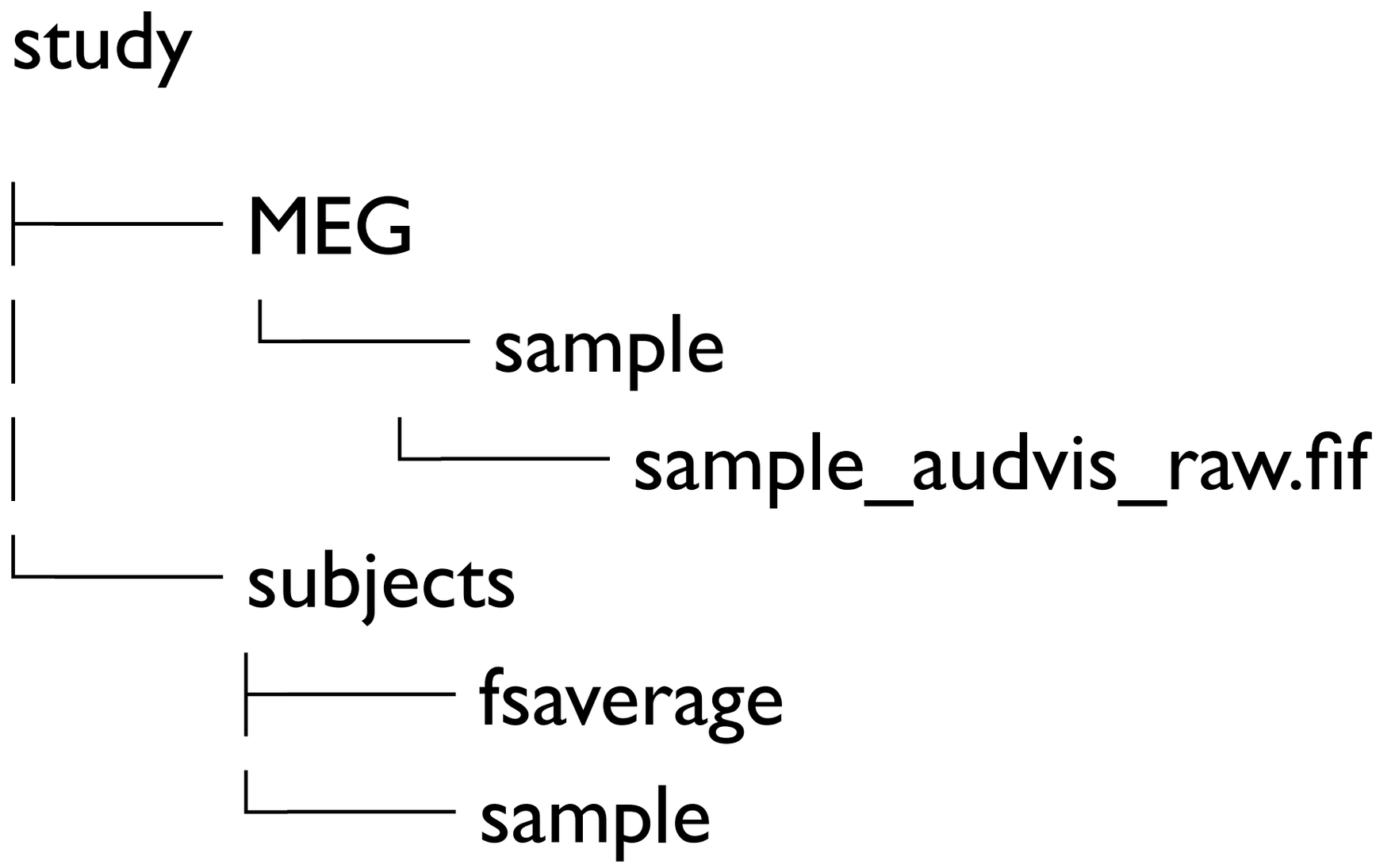
```
$  pip install --upgrade --user mne
```

If you don't have admin rights

Test your install

```
$  python -c "import mne; print mne.__version__"
0.8.6
```

Organizing your data



Browsing Raw data

$ mne_browse_raw

or

$ mne_browse_raw --raw sample_audvis_raw.fif

Remarks:

It is a FIF file

The 3 letters before .fif indicate the content (ave.fif for ERP/ERF, fwd.fif for forward solution, ica.fif for ICA solution, proj.fif for SSPs, etc.)

MNE-Python scripts

$ mne

Usage : mne command options

Accepted commands :

- browse_raw
- bti2fiff
- clean_eog_ecg
- compute_proj_ecg
- compute_proj_eog
- coreg
- flash_bem_model
- kit2fiff
- make_scalp_surfaces
- maxfilter
- report
- surf2bem

Example : mne browse_raw --raw sample_audvis_raw.fif

Getting help example : mne compute_proj_eog -h

Demo SSP for ECG/EOG

```
$ mne compute_proj_ecg -i sample_audvis_raw.fif --l-
freq 1 --h-freq 100 --rej-grad 3000 --rej-mag 4000
--rej-eeg 100 --average -c "MEG 1531" --ecg-h-freq
25 --tstart 5
```

Some links

- Documentation:
  - http://martinos.org/mne/ (general doc)
  - http://martinos.org/mne/stable/manual/index.html (manual)
  - http://martinos.org/mne/stable/tutorials.html (tutorials with code)
  - http://martinos.org/mne/auto_examples/index.html (python examples)
- Code:
  - https://github.com/mne-tools/mne-python (mne-python code)
  - https://github.com/mne-tools/mne-matlab (mne matlab toolbox)
  - https://github.com/mne-tools/mne-scripts (mne shell scripts)