Donders Institute
for Brain, Cognition and Behaviour

FieldTrip

Introduction to EEG/MEG and the FieldTrip toolbox

Robert Oostenveld

Donders Institute for Brain, Cognition and Behaviour
Radboud University
Nijmegen, The Netherlands

What is FieldTrip

a MATLAB toolbox for the analysis of MEG, EEG and animal electrophysiology data

can import data from many different file formats

contains algorithms for spectral analysis, source reconstruction, statistics, connectivity, …

Talk outline

Recording signals from the brain

Temporal and spatial features of the signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Talk outline

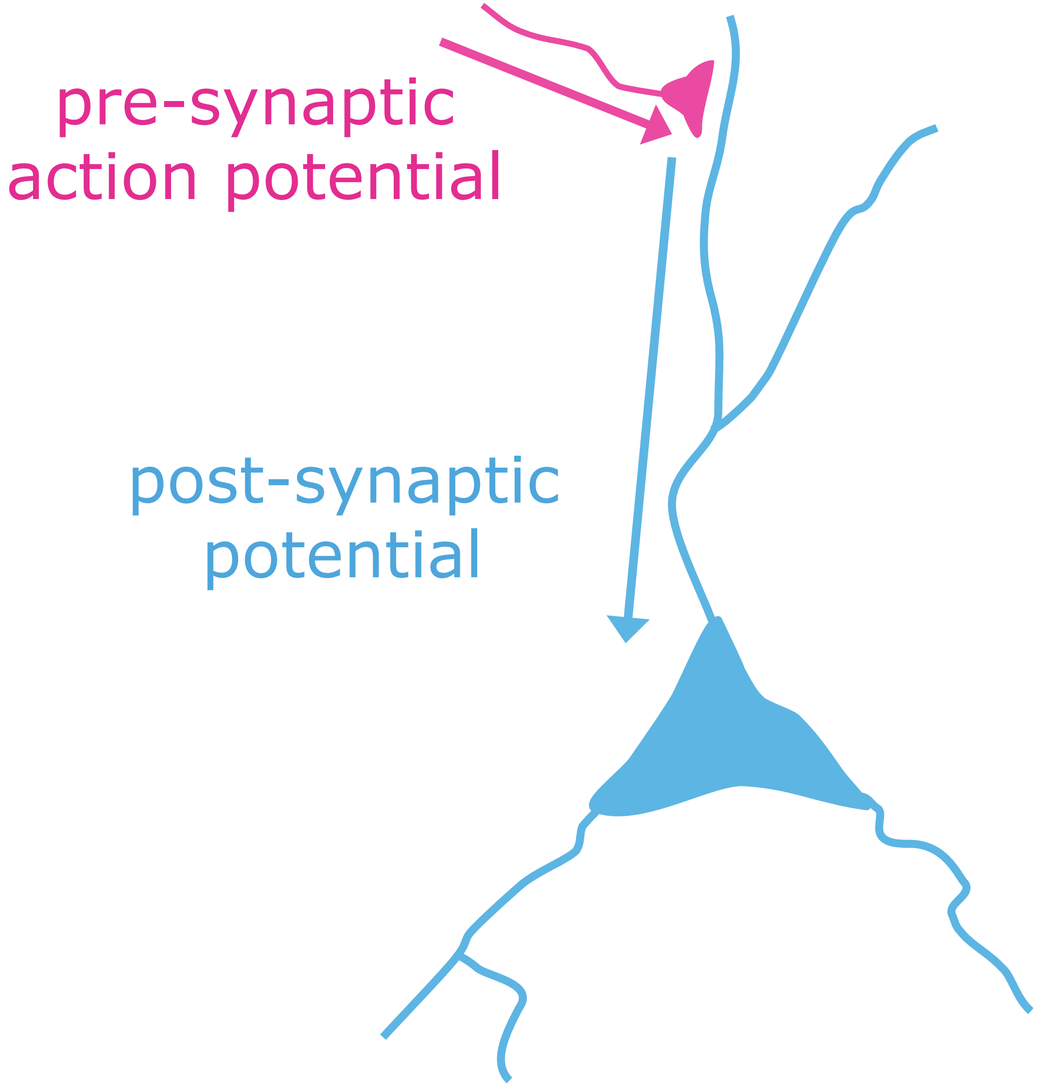
Recording signals from the brain

Temporal and spatial features of the signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

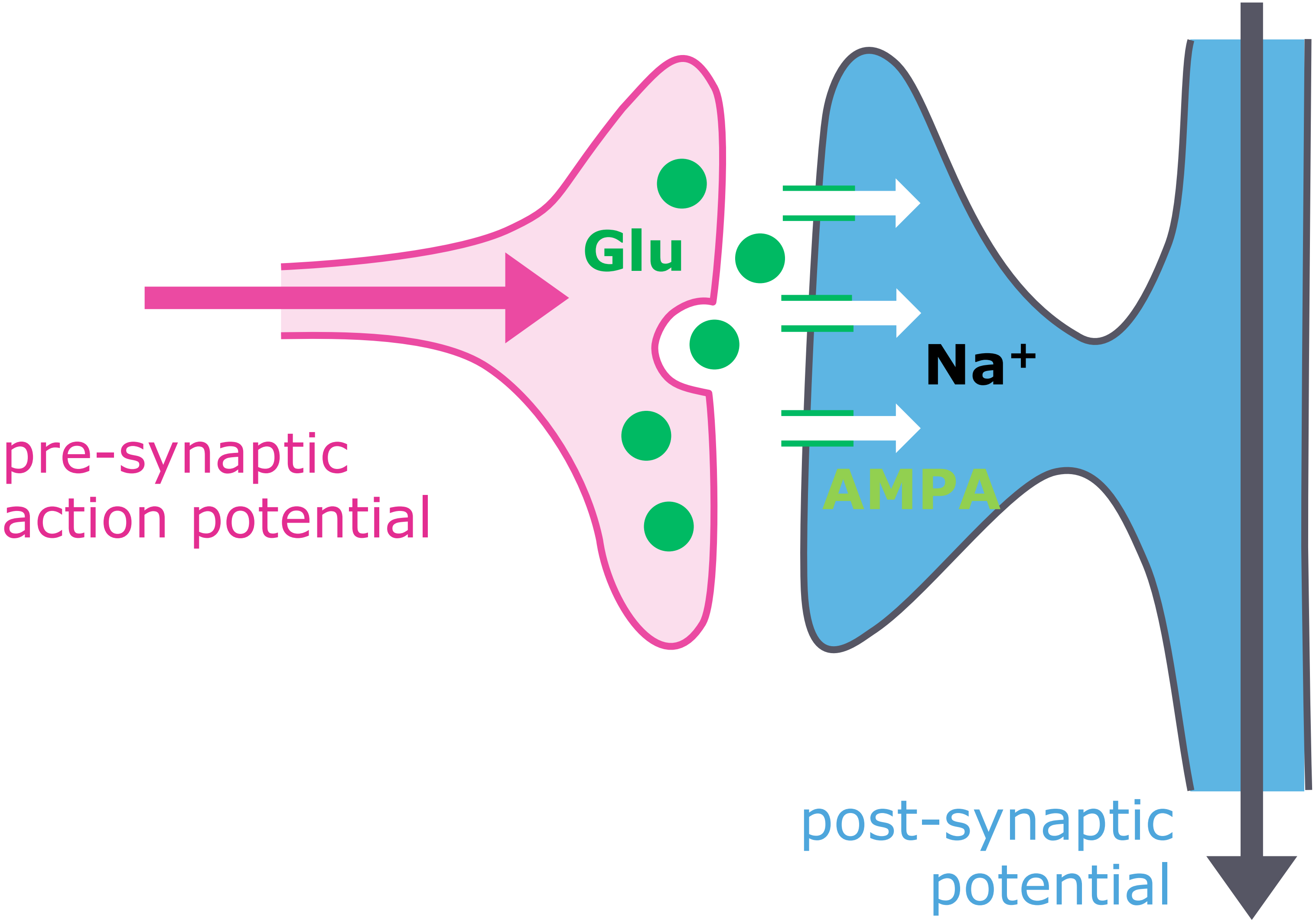
What produces the electric current and magnetic field



pre-synaptic action potential

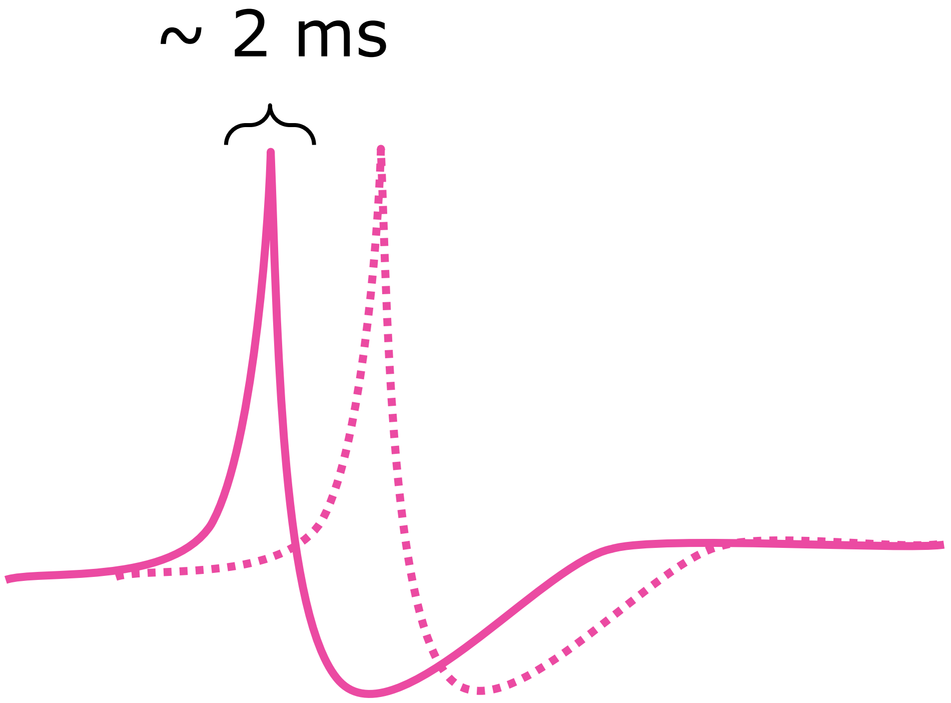
post-synaptic potential

What produces the electric current and magnetic field

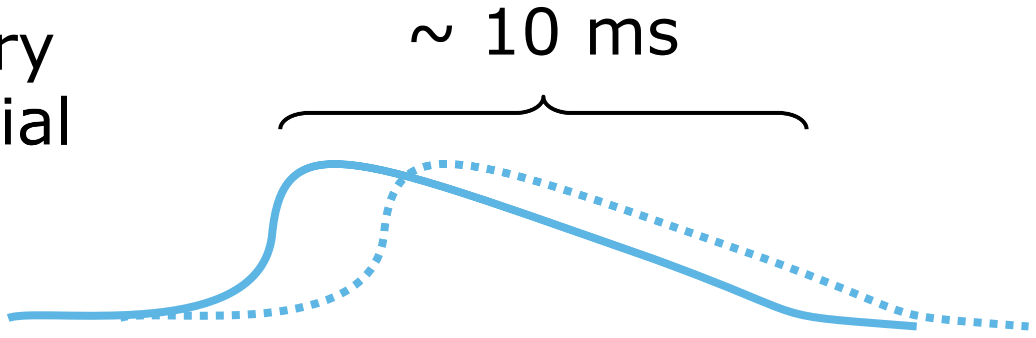


What produces the electric current and magnetic field

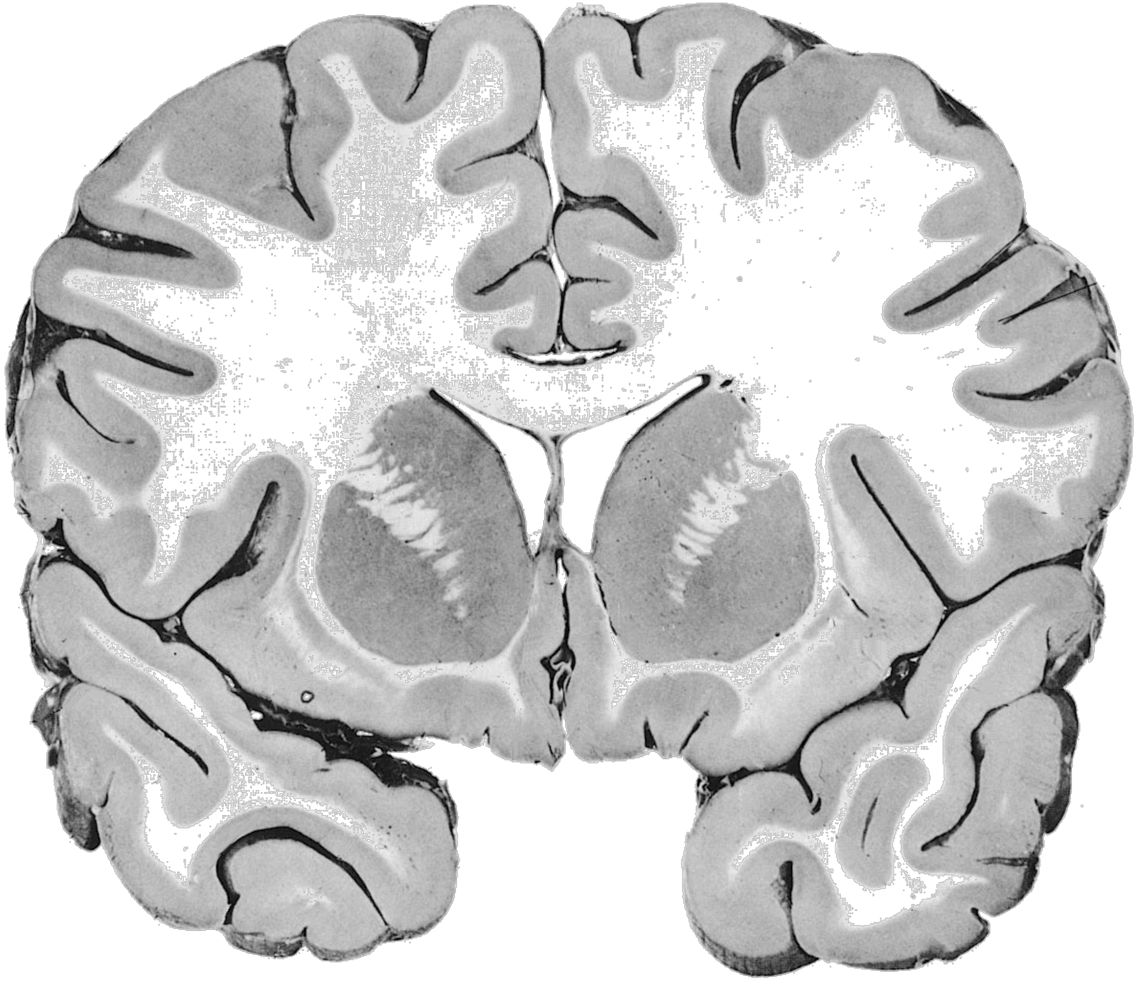
action potential



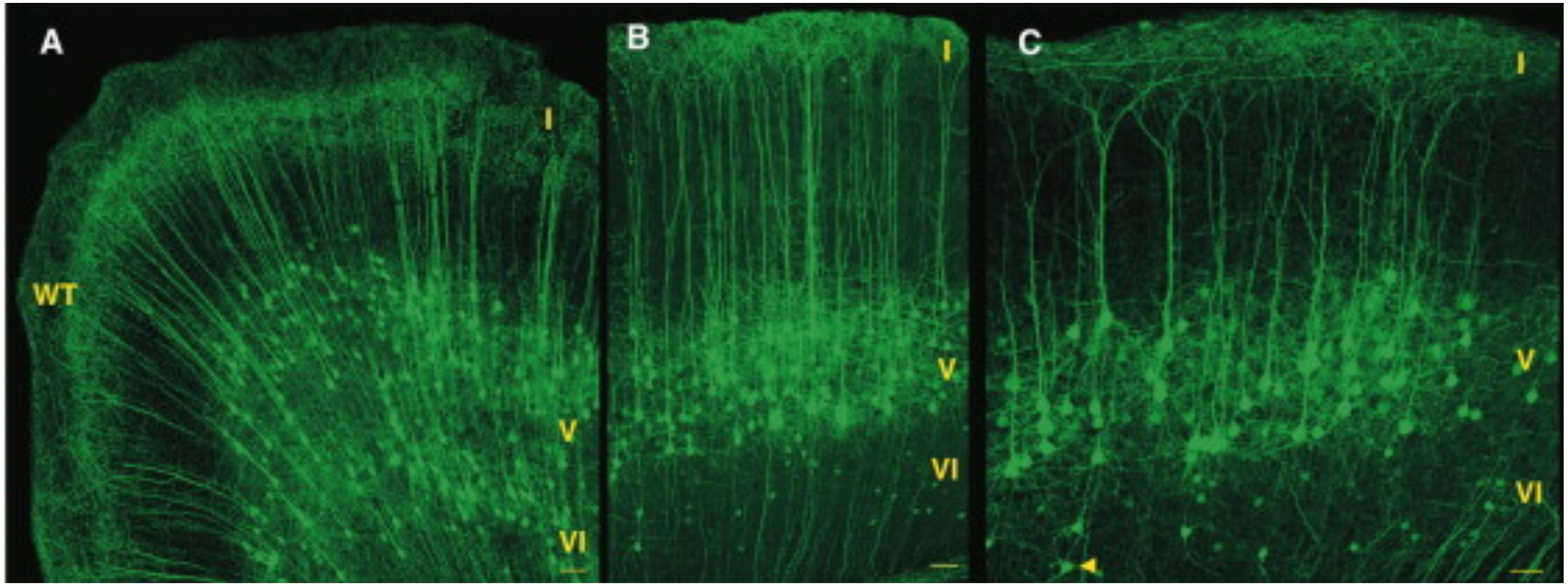
exitatory or inhibitory
post-synaptic potential



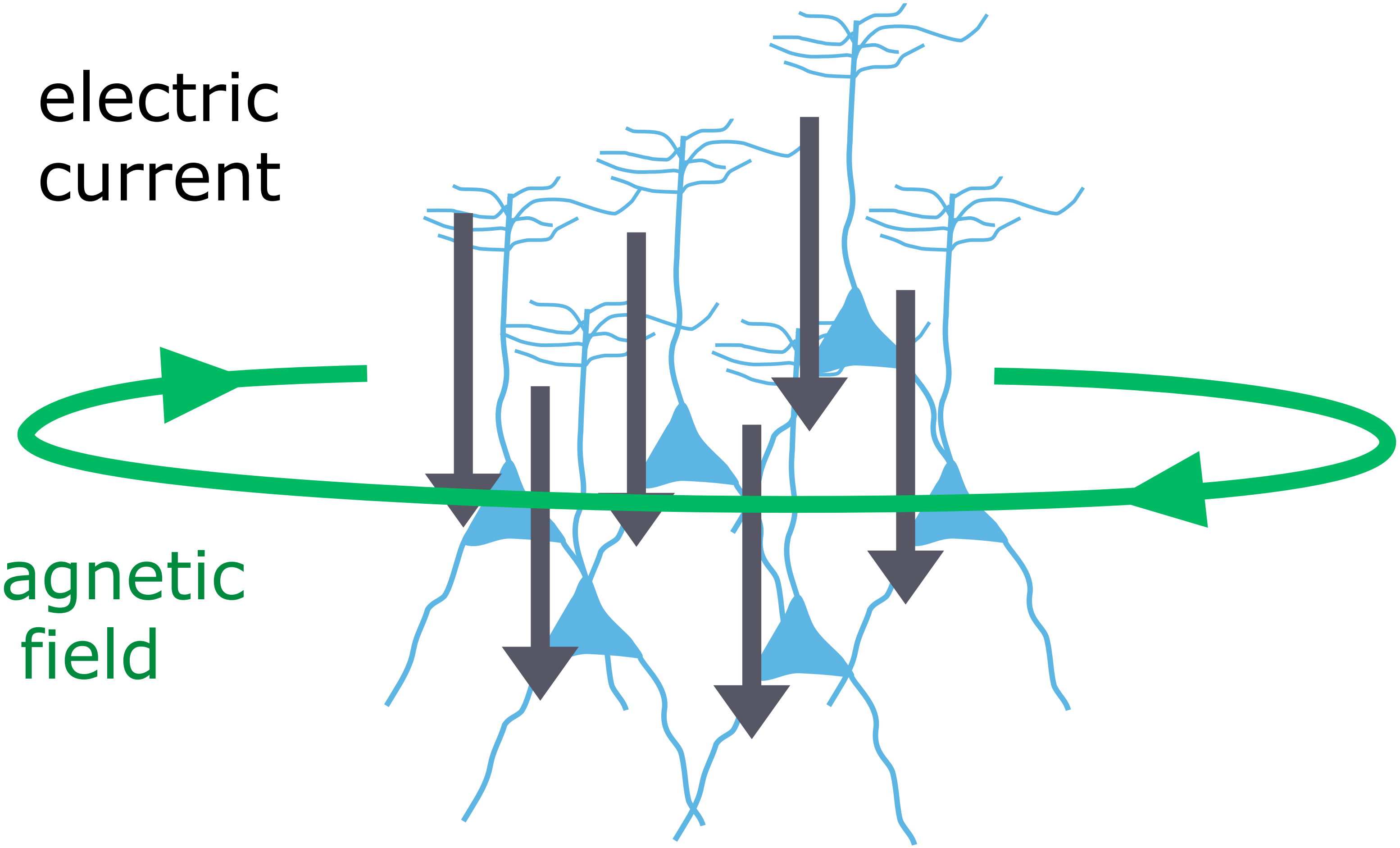
What produces the electric current and magnetic field



What produces the electric current and magnetic field



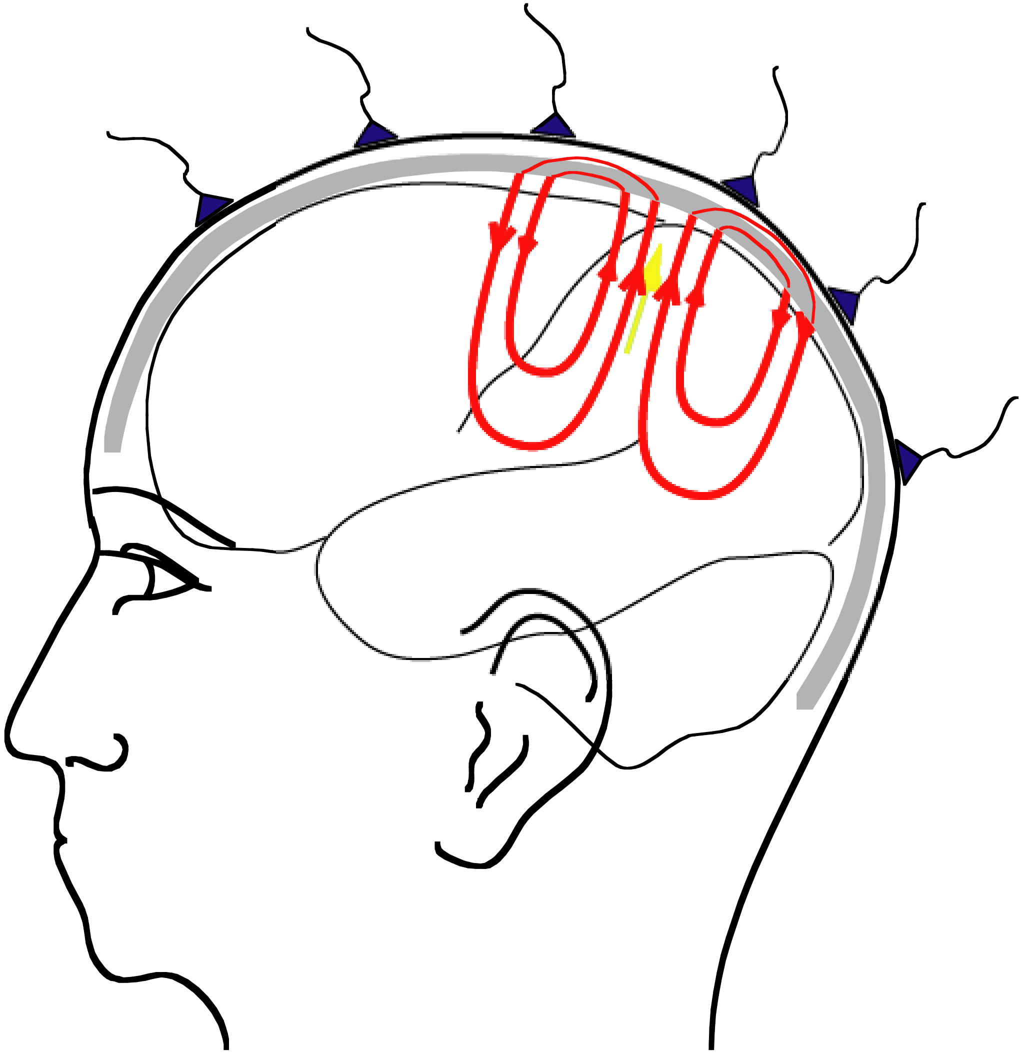
What produces the electric current and magnetic field



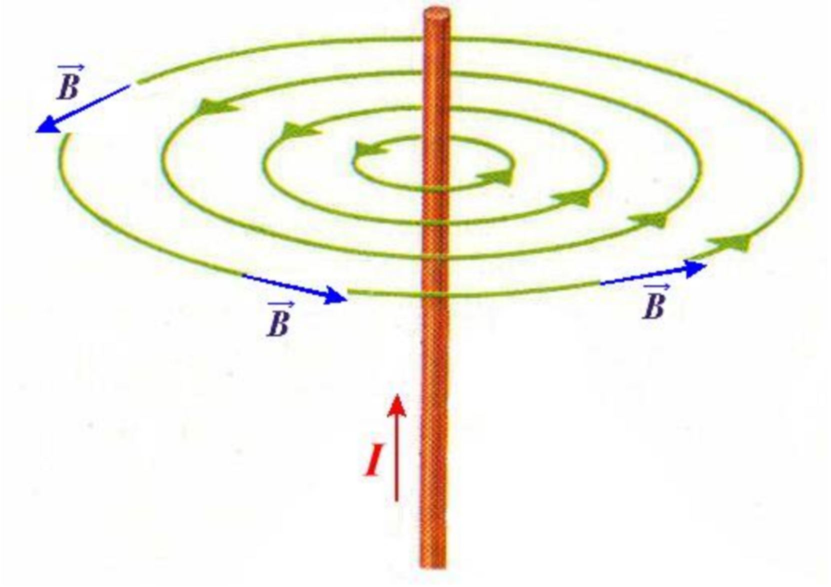
electric current

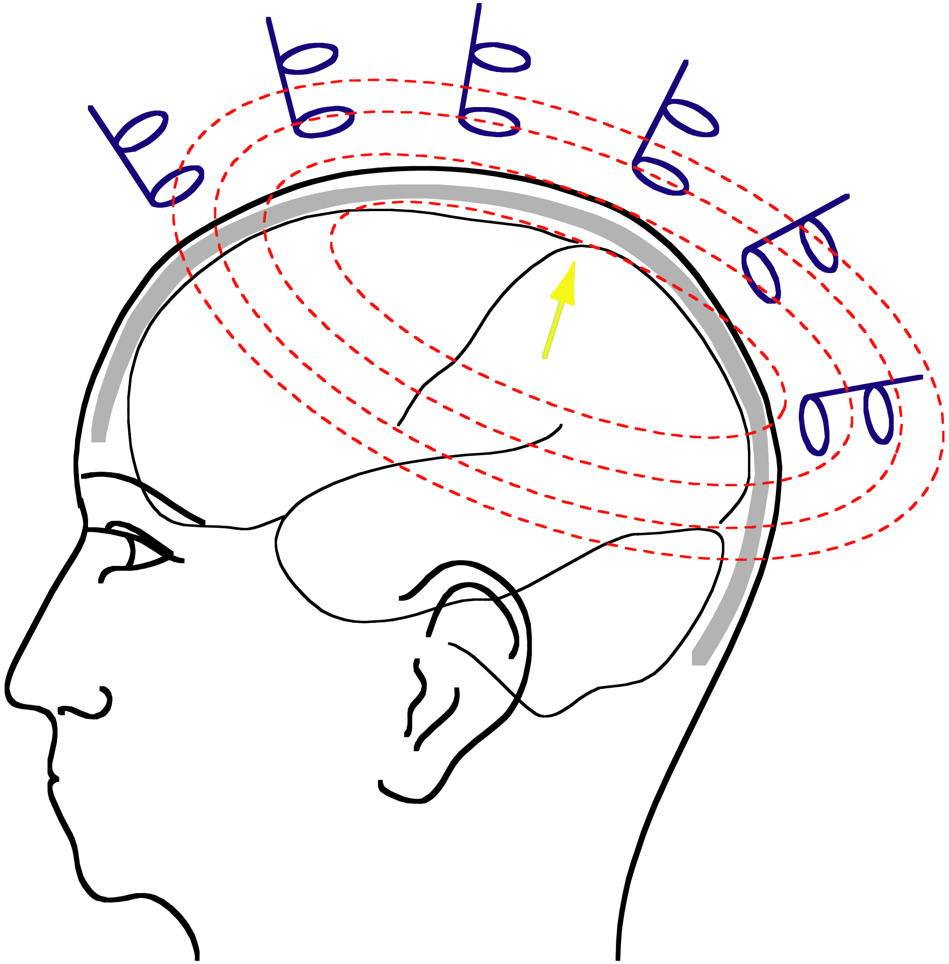
magnetic field

EEG volume conduction



Electric current → magnetic field





Technical challenges of MEG

Sensitive magnetic detectors

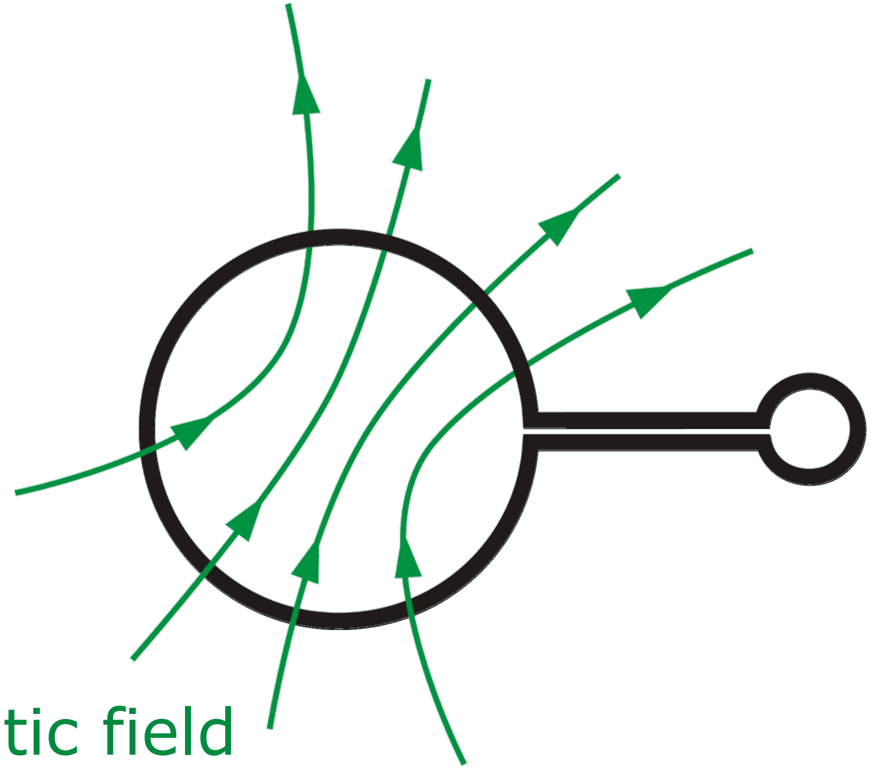
superconductive sensors

Deal with environmental noise

shielding

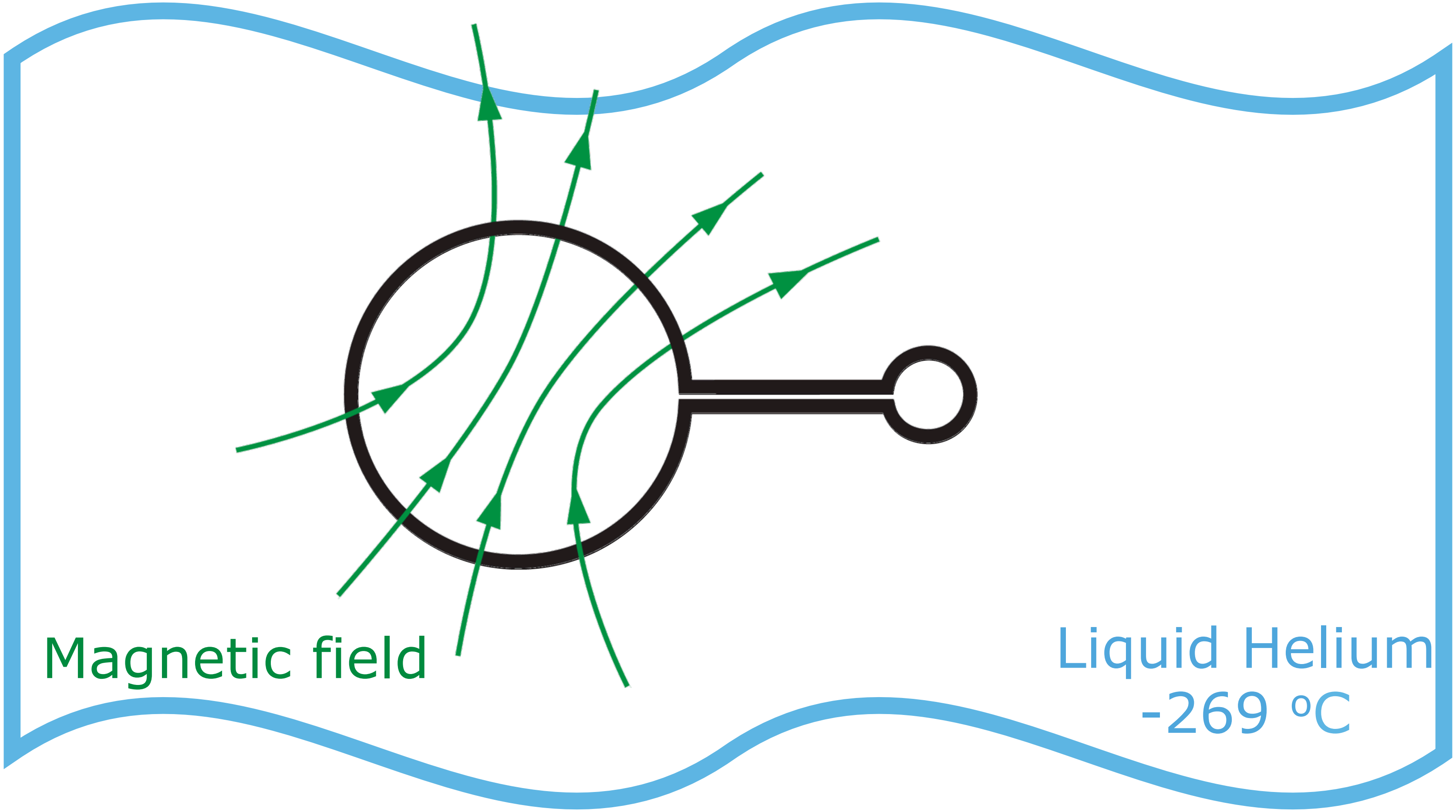
sensor design

MEG sensors - magnetometer

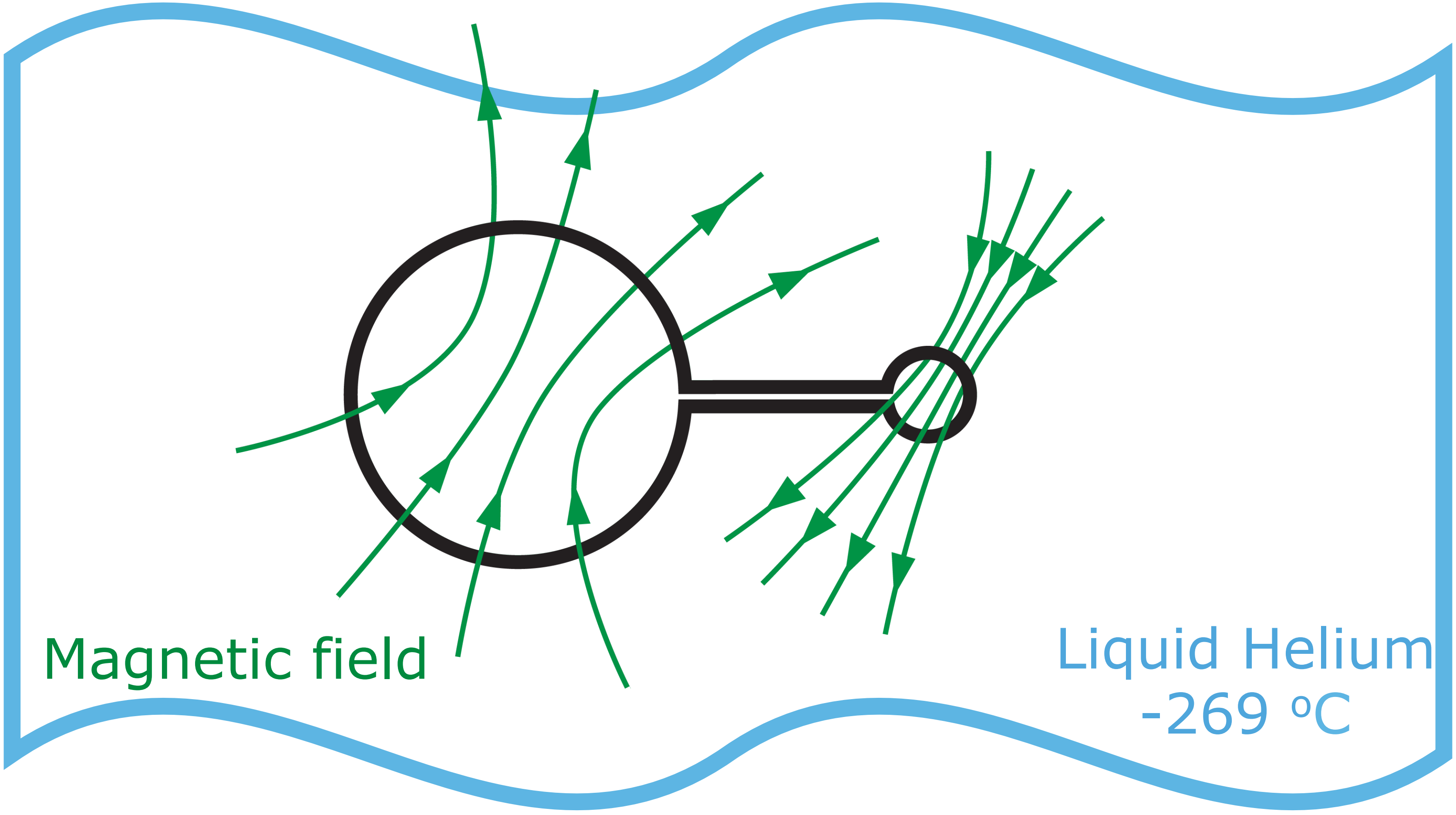


Magnetic field

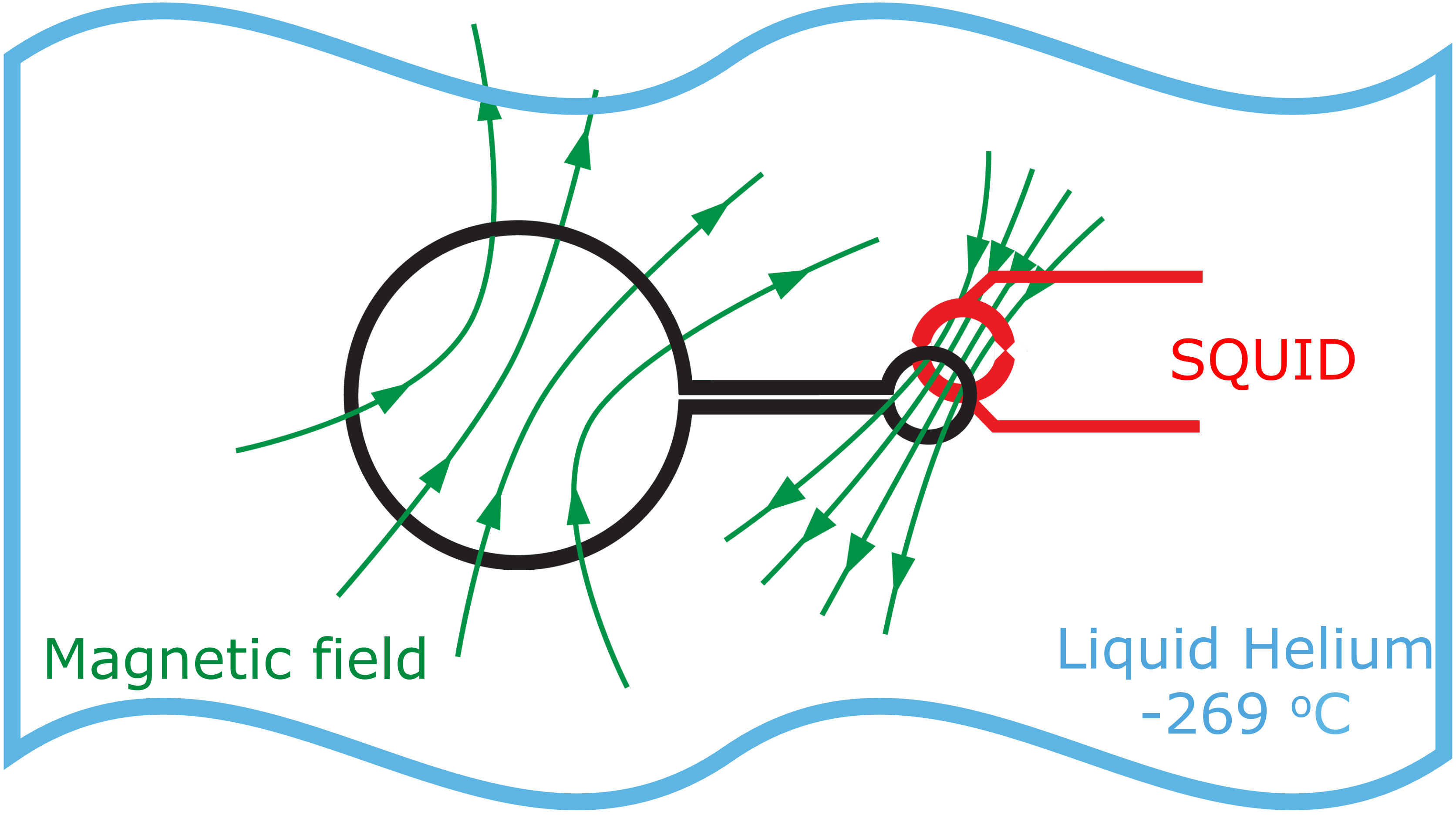
MEG sensors - magnetometer



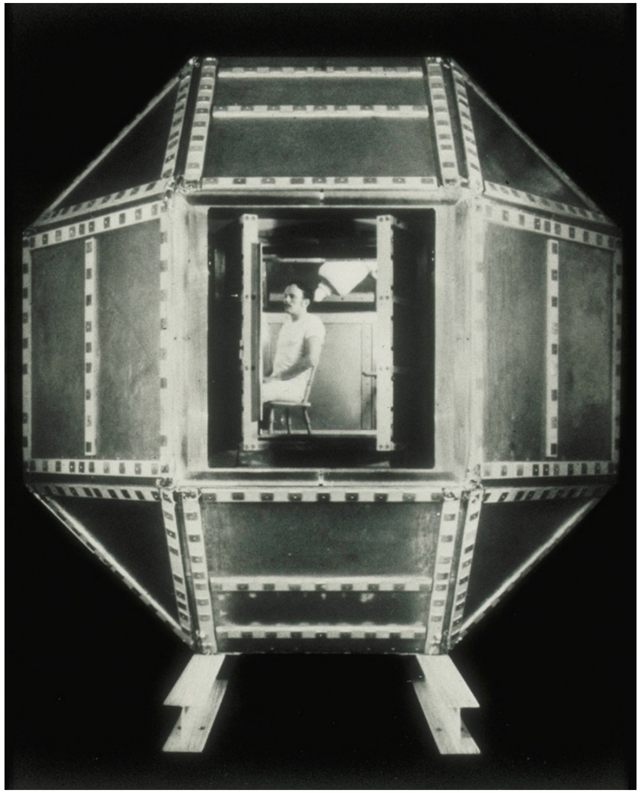
MEG sensors - magnetometer



MEG sensors - magnetometer



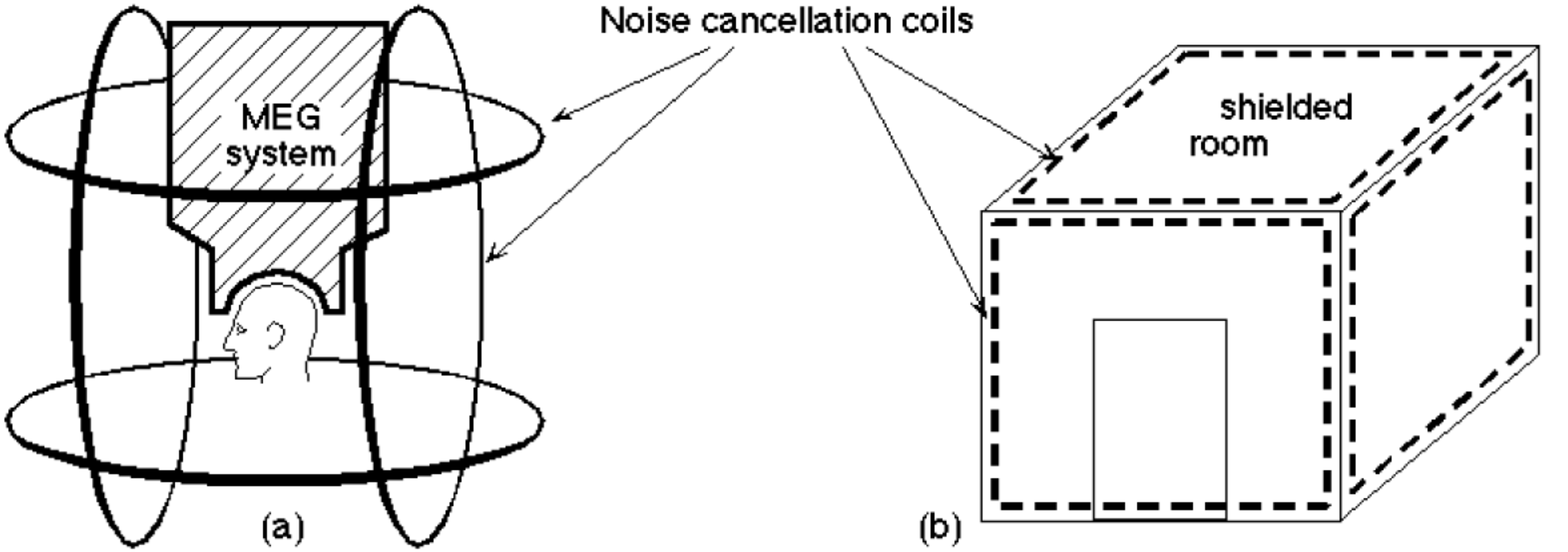
Passive Shielding



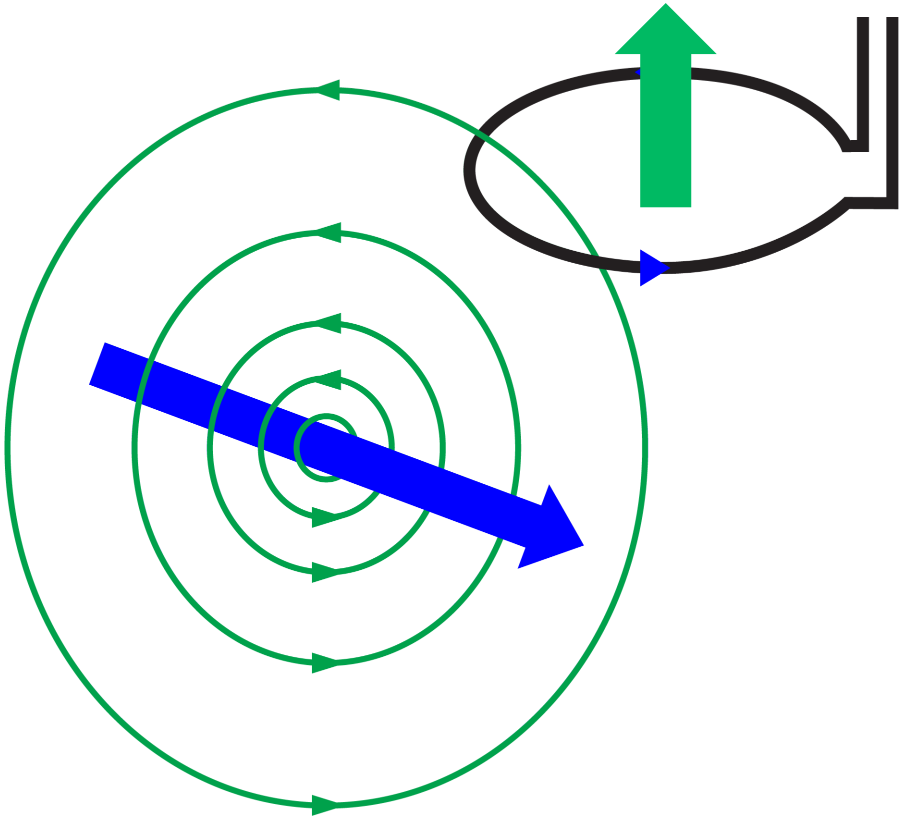
The magnetically shielded room built by David Cohen at MIT's Francis Bitter National Magnet Laboratory in 1969.

The magnetically shielded room at NatMEG in Stockholm (2013).

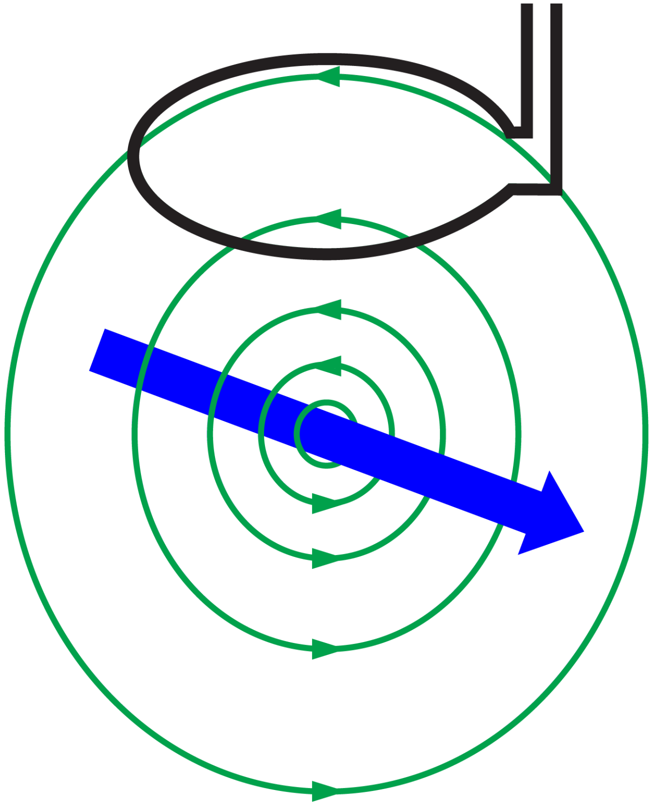
Active Shielding



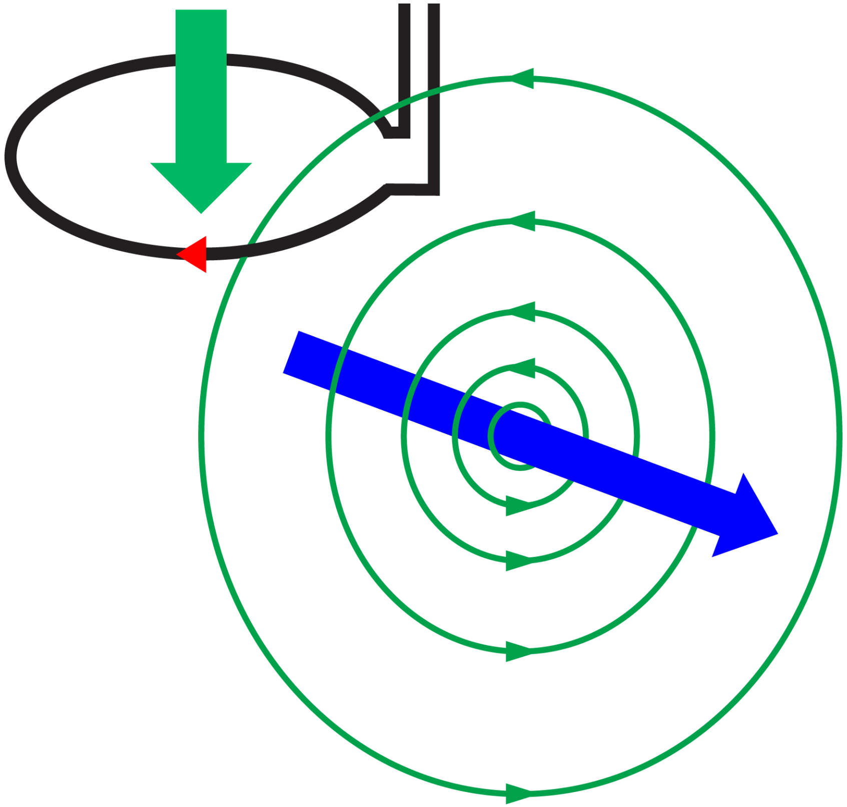
Magnetometer

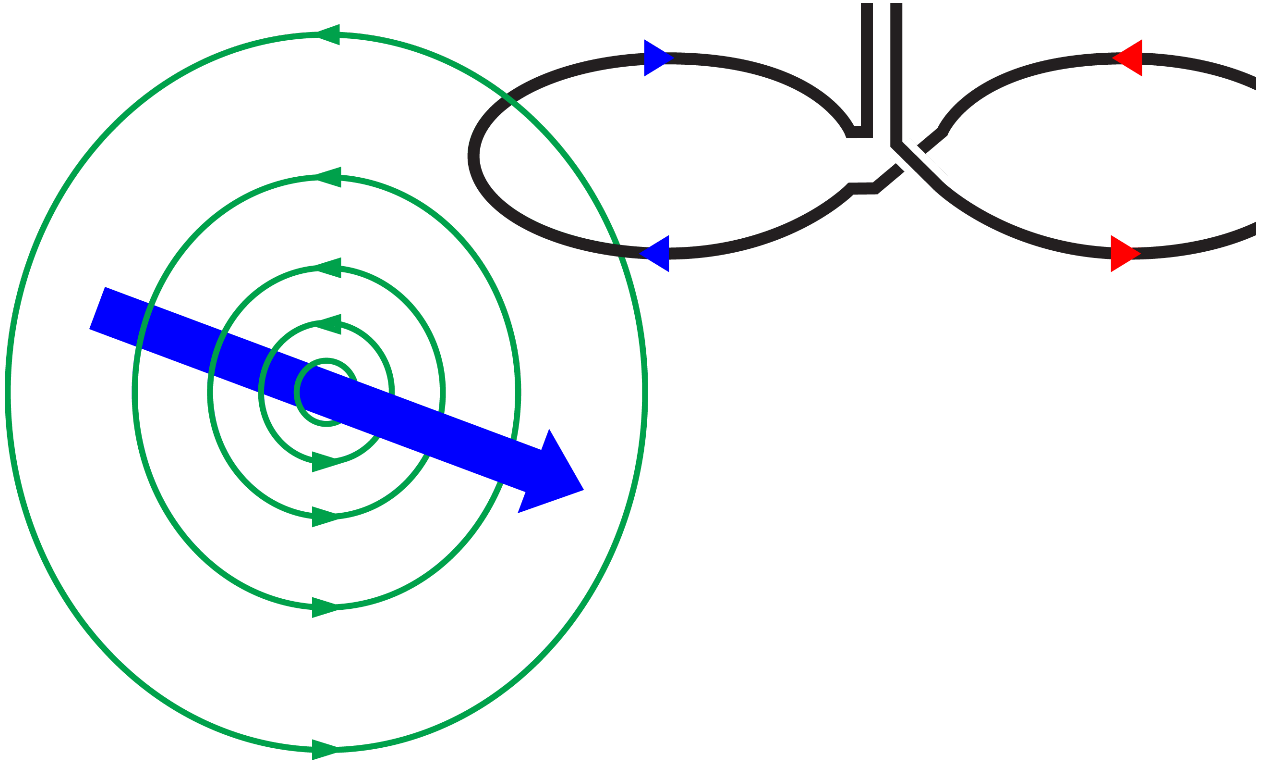


Magnetometer

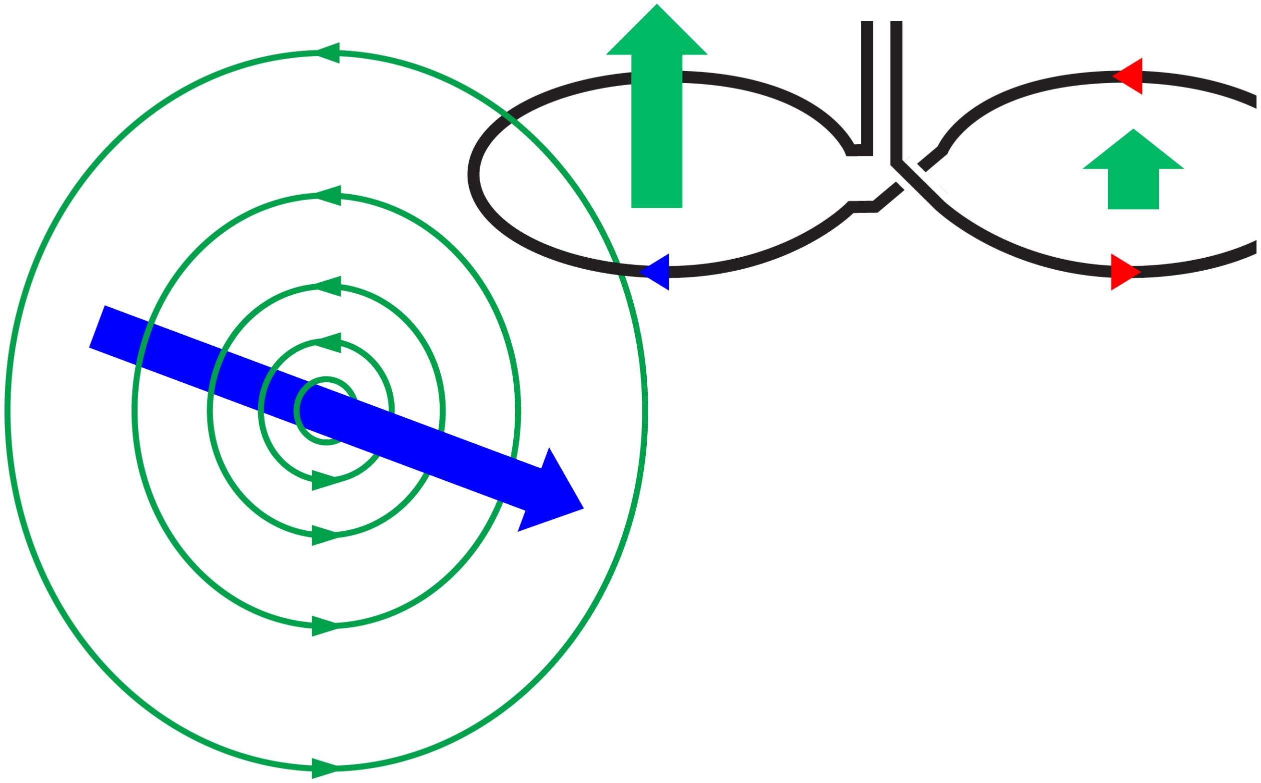


Magnetometer

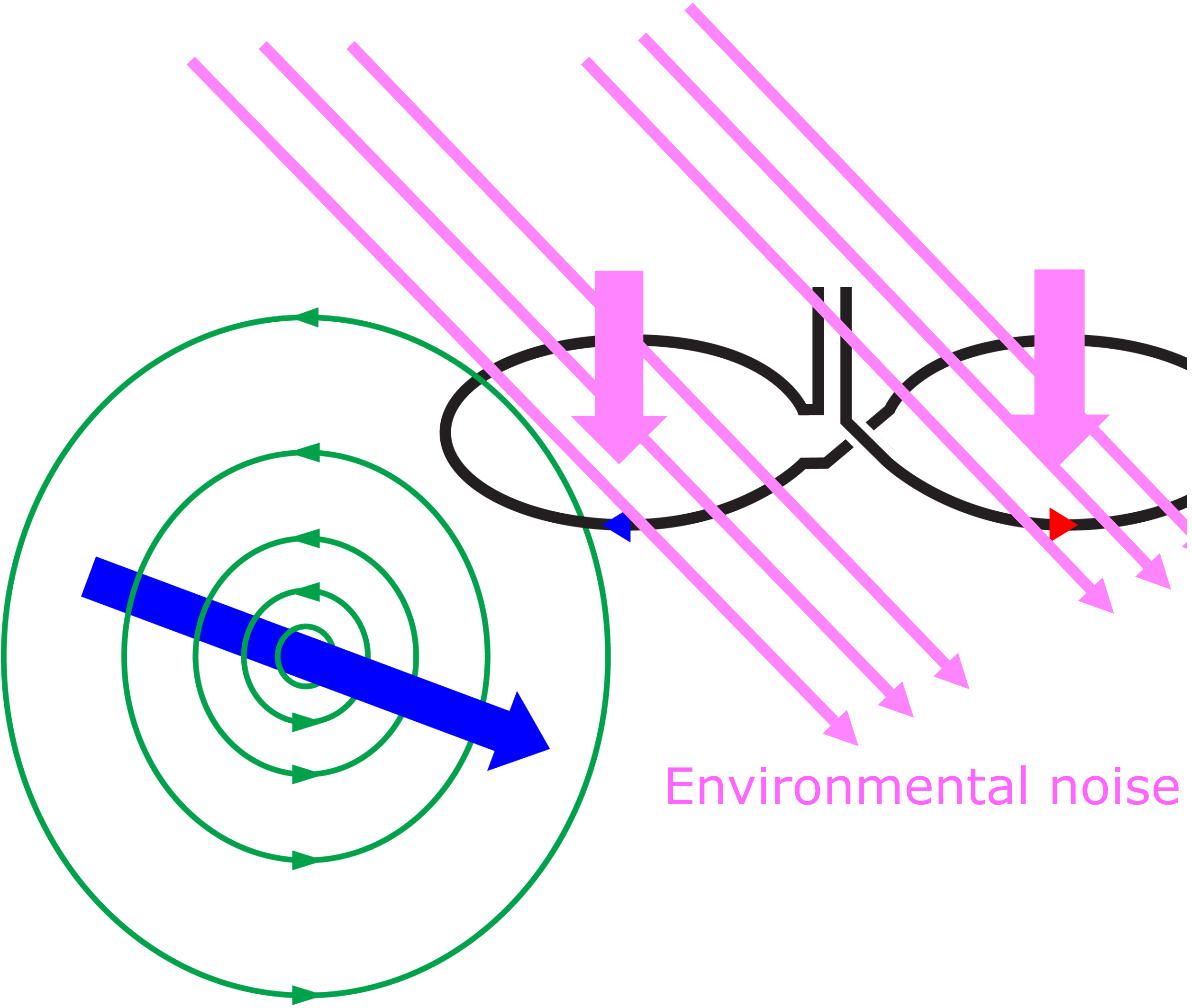




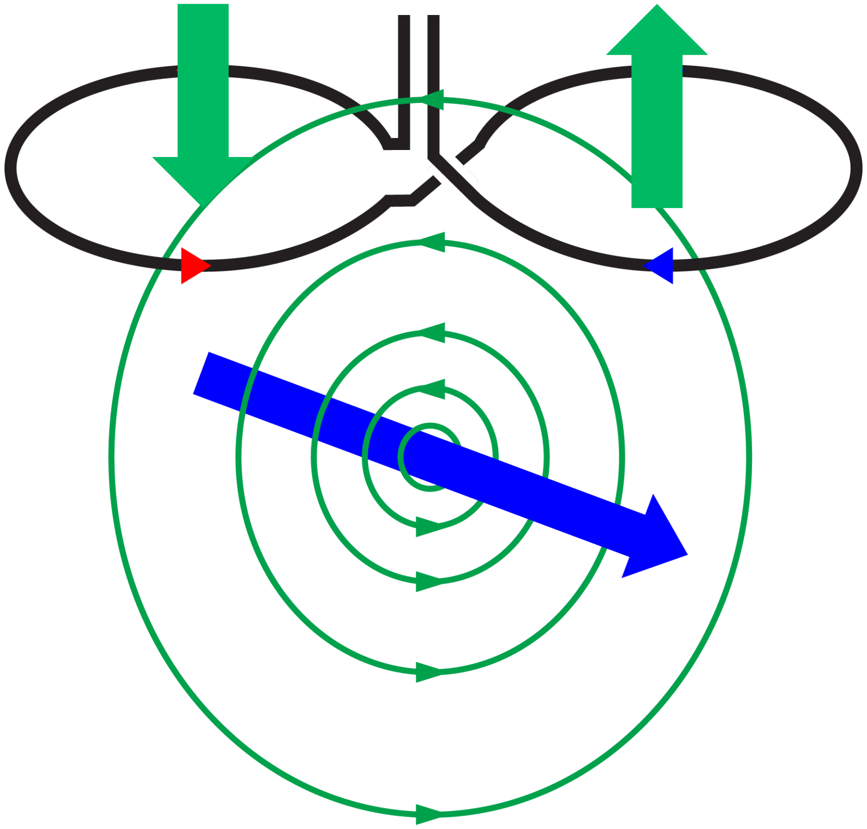
Planar gradiometer



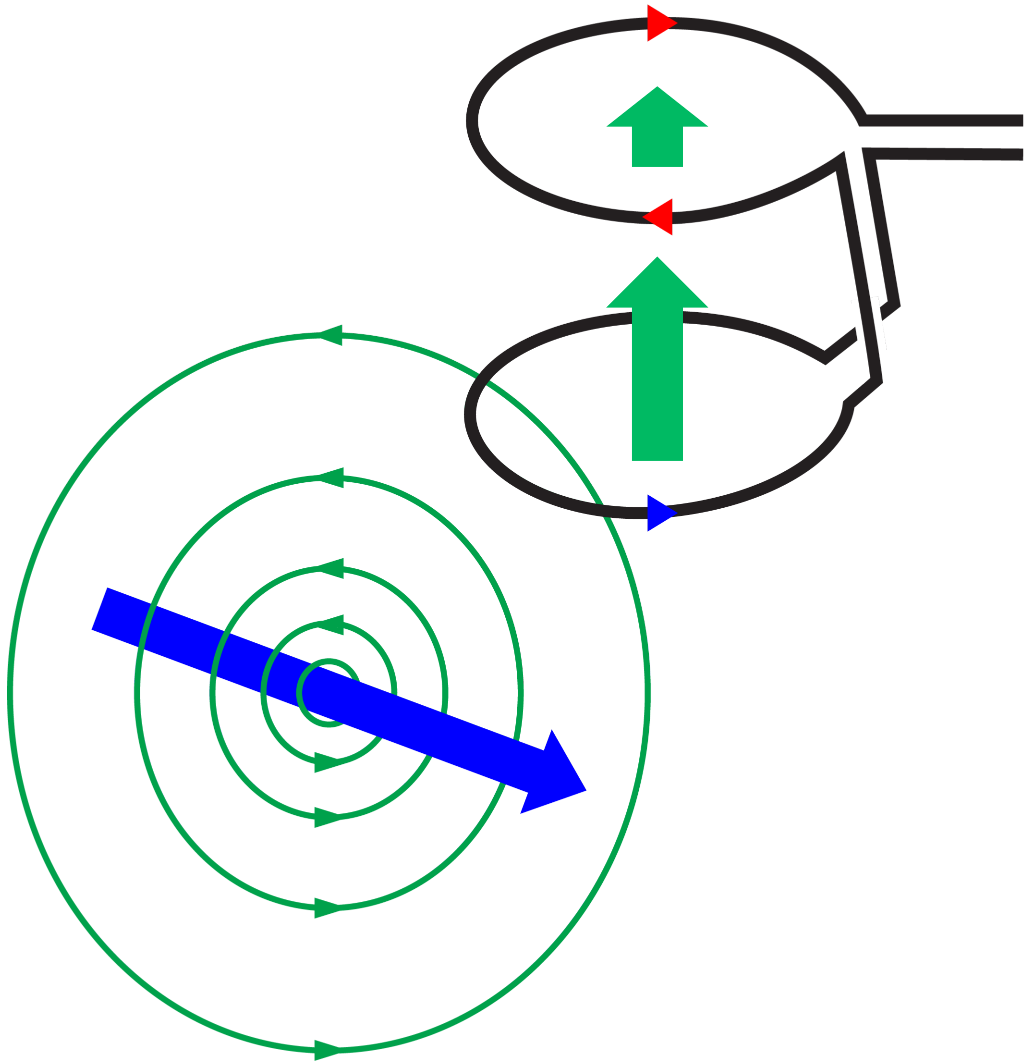
Planar gradiometer



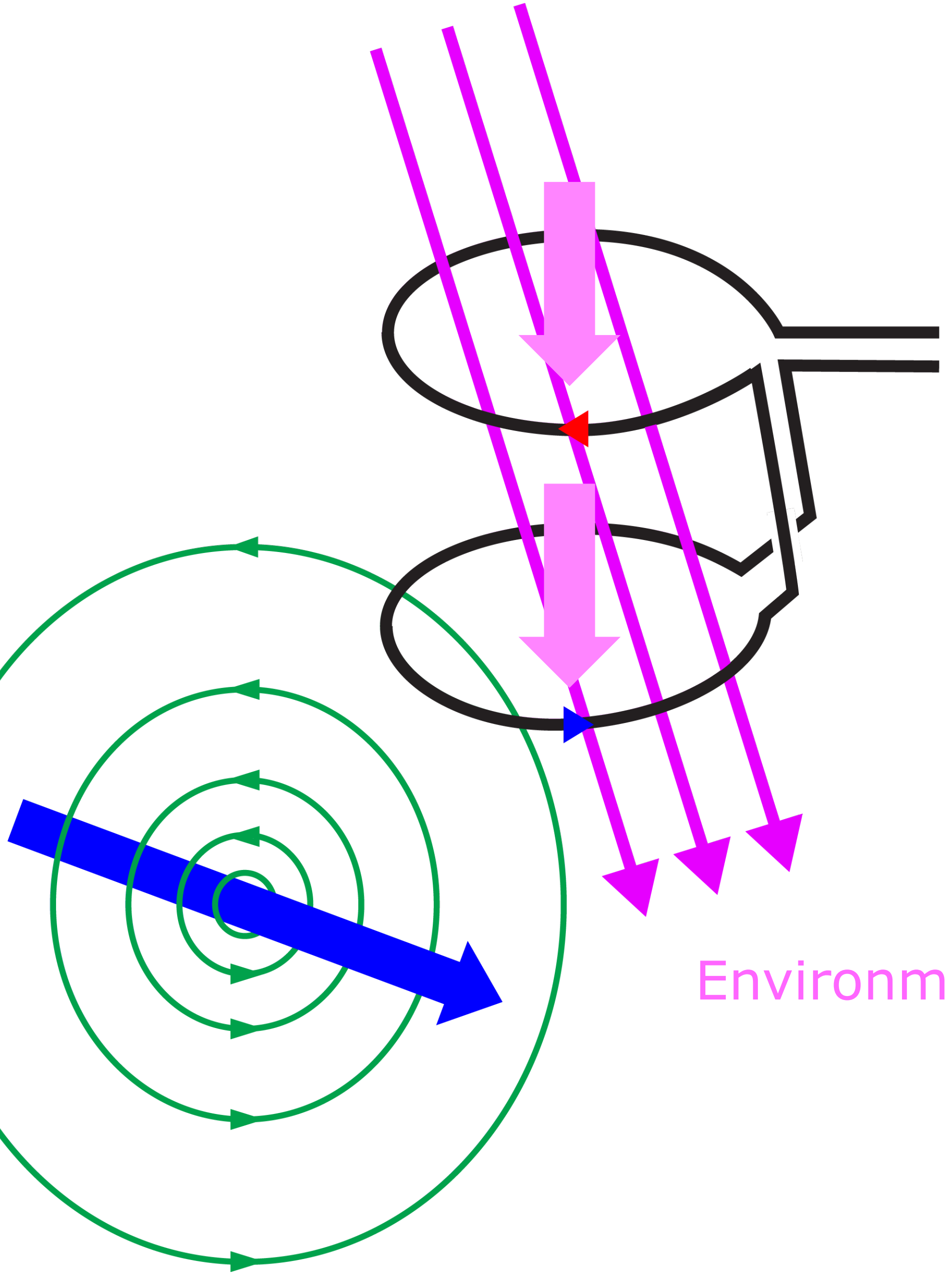
Planar gradiometer



Axial gradiometer

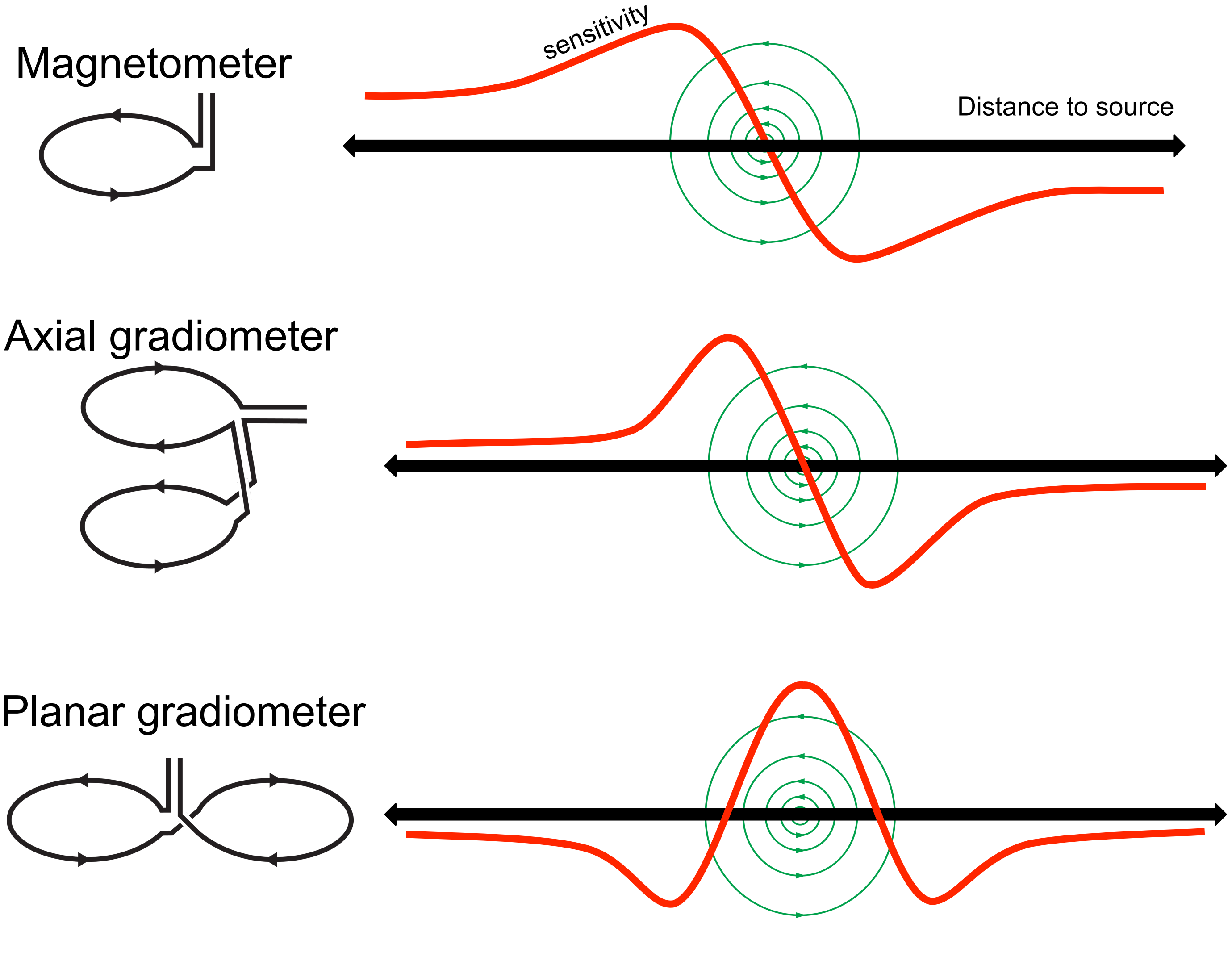


Axial gradiometer



Environmental noise

Different sensitivity profiles



Magnetometer

Axial gradiometer

Planar gradiometer

Talk outline

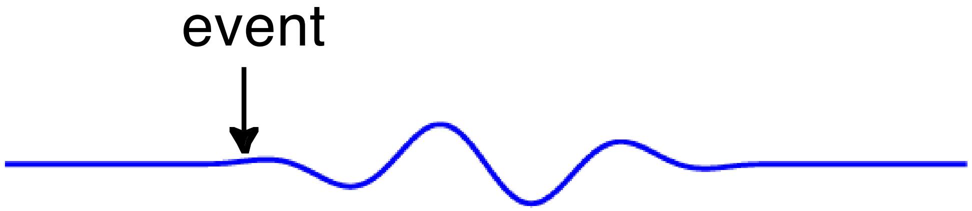
Recording signals from the brain

Temporal and spatial features of the signals

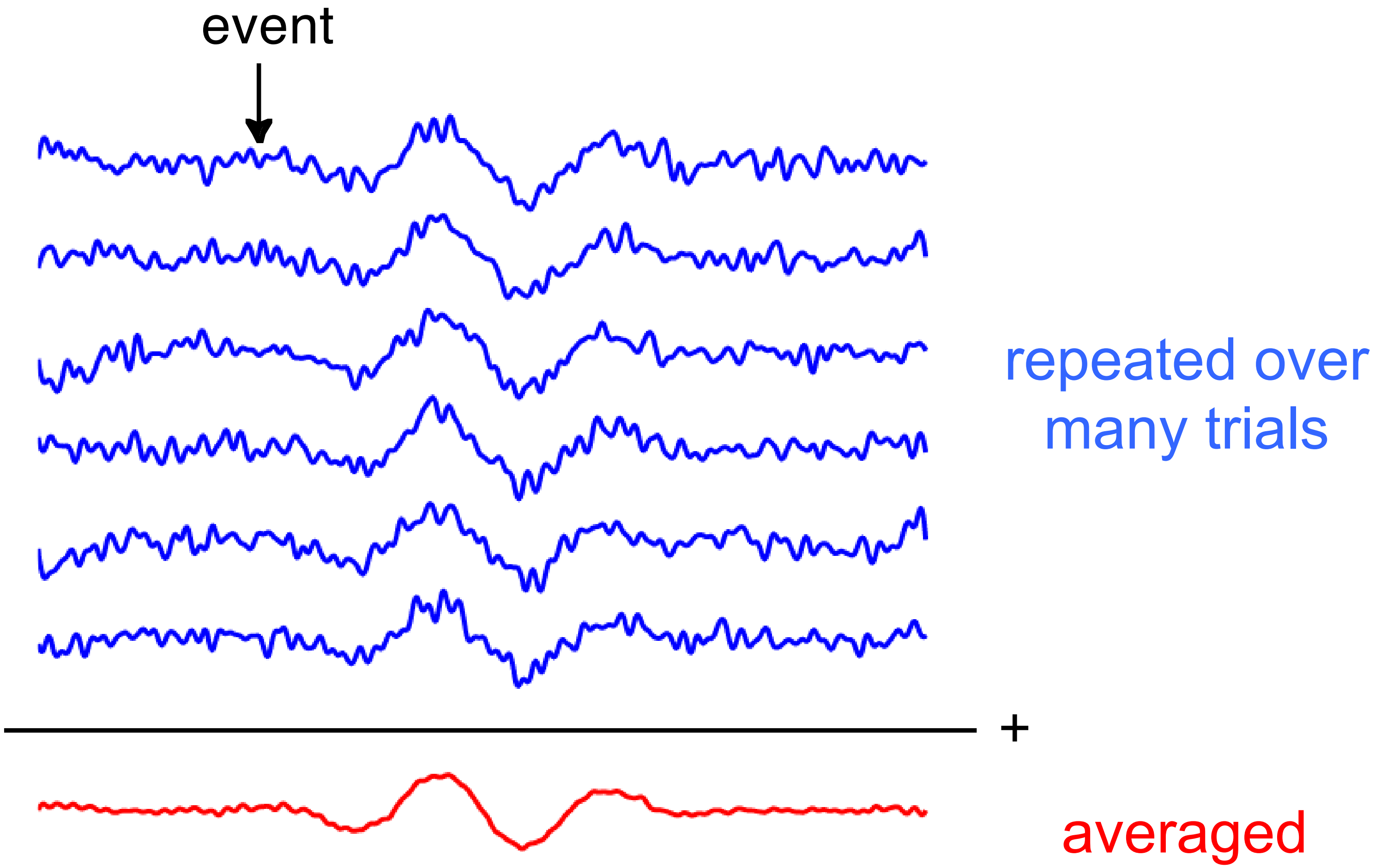
Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

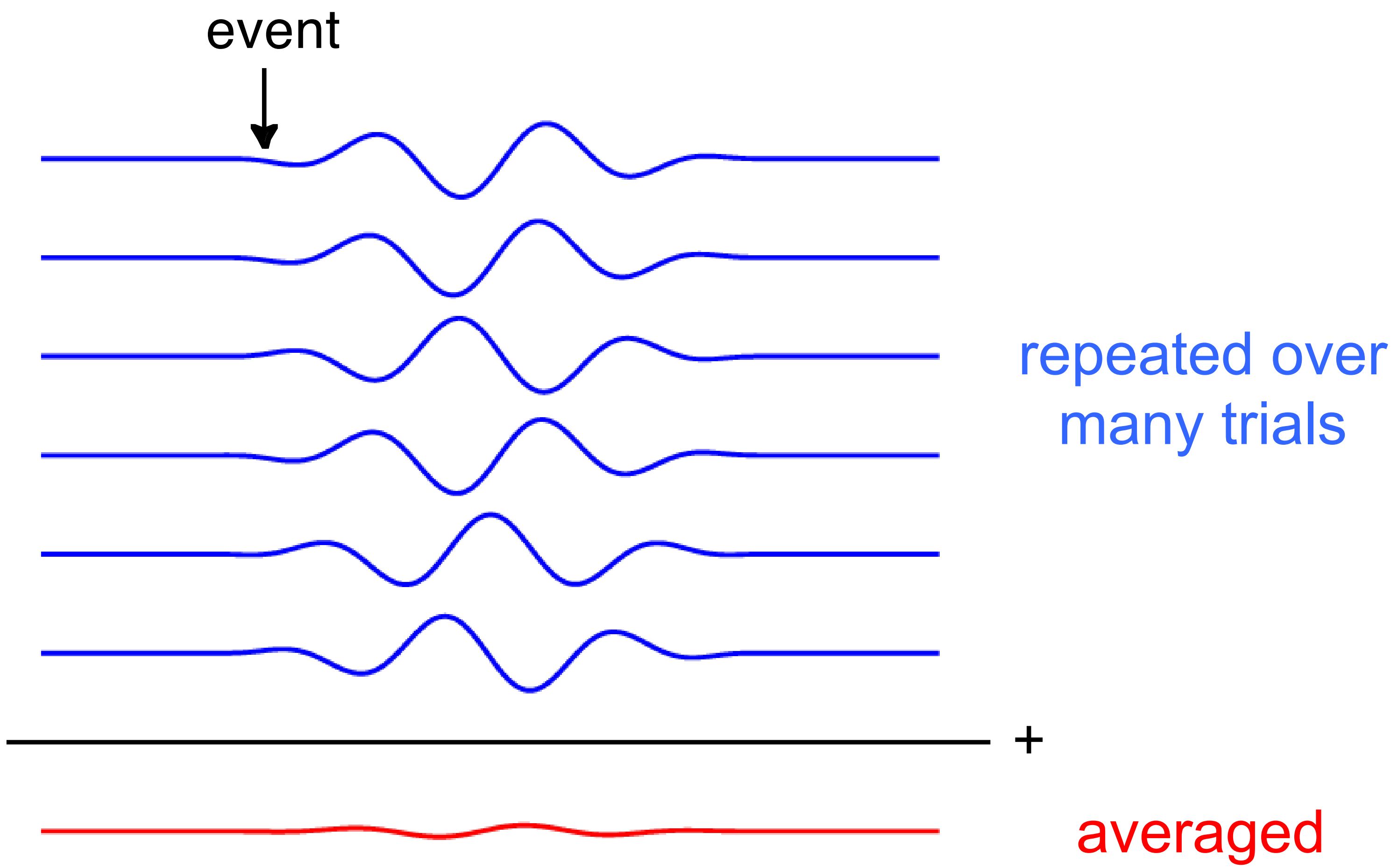
Evoked activity



Evoked activity



Induced activity



M/EEG signal characteristics considered during analysis

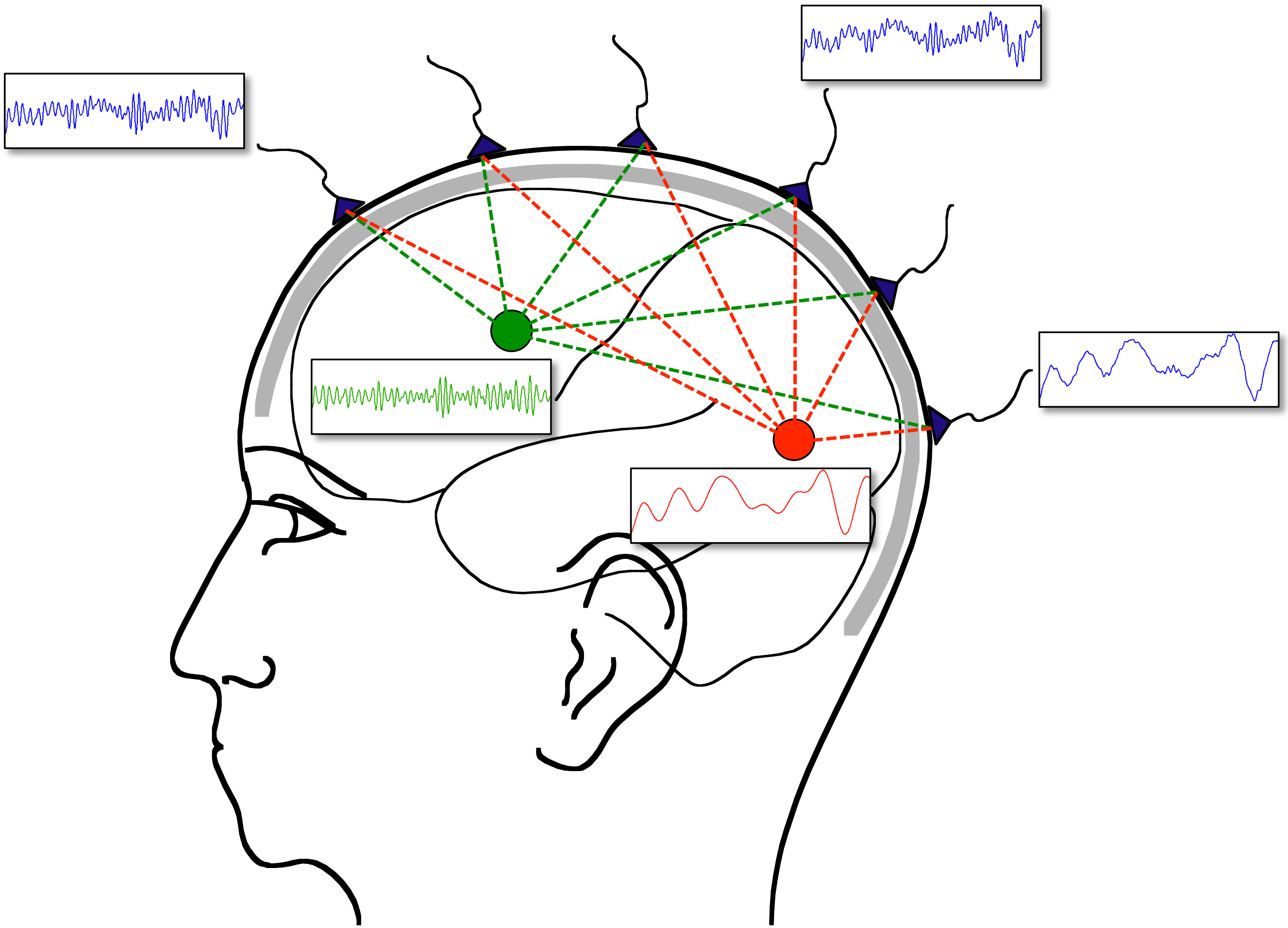
timecourse of activity
-> ERP

spectral characteristics
-> power spectrum

temporal changes in power
-> time-frequency response (TFR)

spatial distribution of activity over the head
-> source reconstruction

Superposition of source activity



Separating activity of different sources (and noise)

- Use the temporal aspects of the data at the channel level
  - ERF latencies
  - ERF difference waves
  - Filtering the time-series
  - Spectral decomposition

- Use the spatial aspects of the data
  - Volume conduction model of head
  - Estimate source model parameters

Talk outline

Recording signals from the brain

Temporal and spatial features of the signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Some FieldTrip basics

```
dataout = functionname(cfg, datain, …)
```

```
functionname(cf
```

```
dataout = functi
```

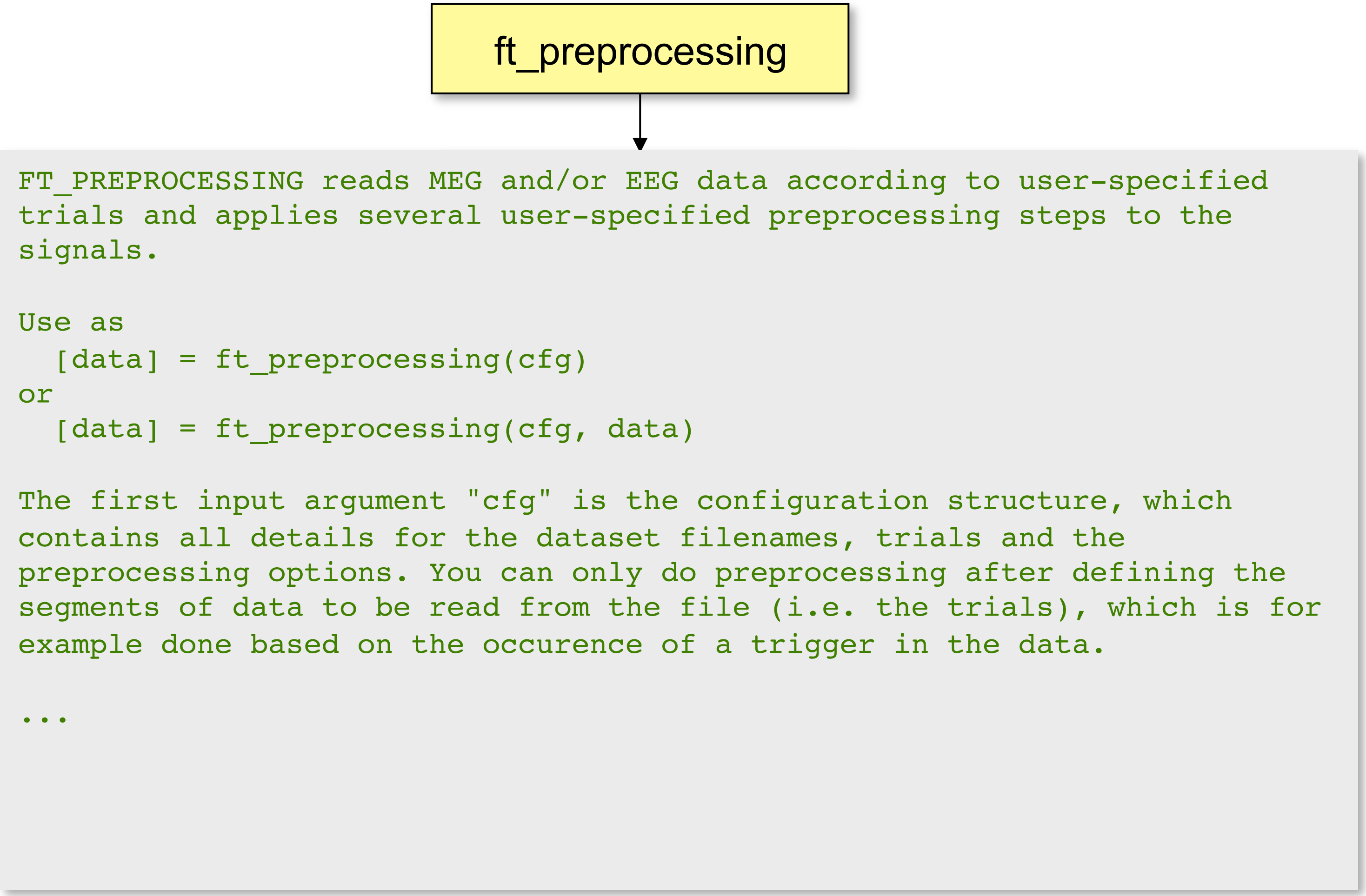
the “cfg” argument is a configuration structure, e.g.

```
cfg.channel = {‘C3’, C4’, ‘F3’, ‘F4’}
cfg.foilim = [1 70]
```

FieldTrip v.s. default Matlab

```
dataout = functionname(cfg, datain, …)

          cfg.key1 = value1
          cfg.key2 = value2

dataout = functionname(datain, ‘key1’, ‘value1’, …)
```

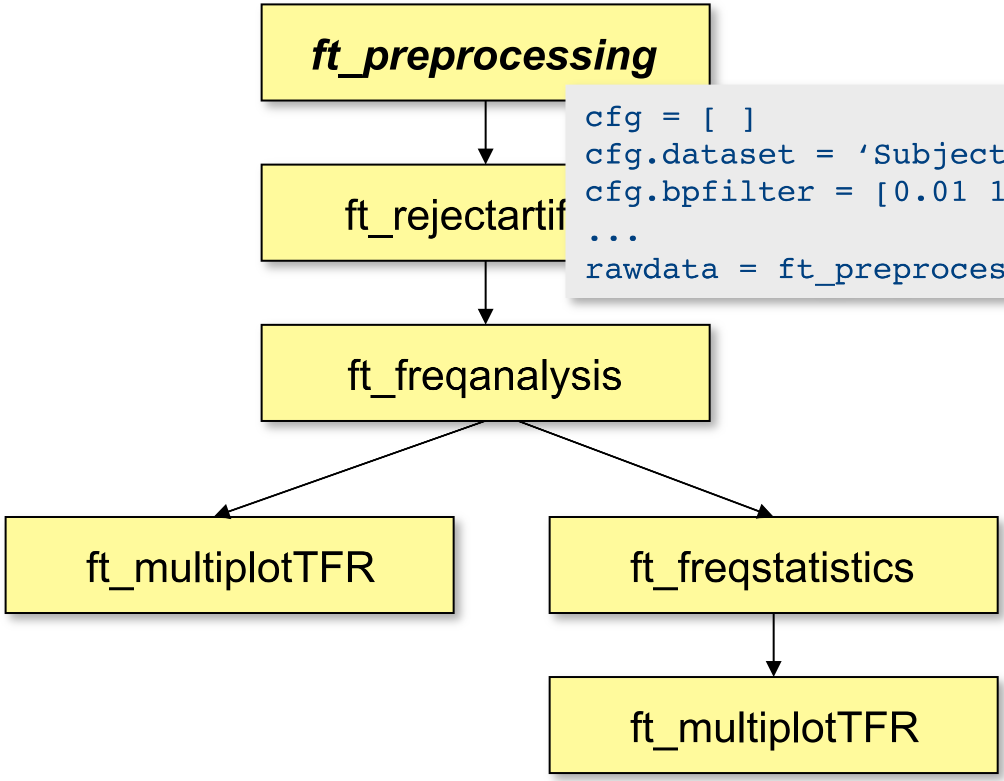
Using functions in an analysis protocol



ft_preprocessing

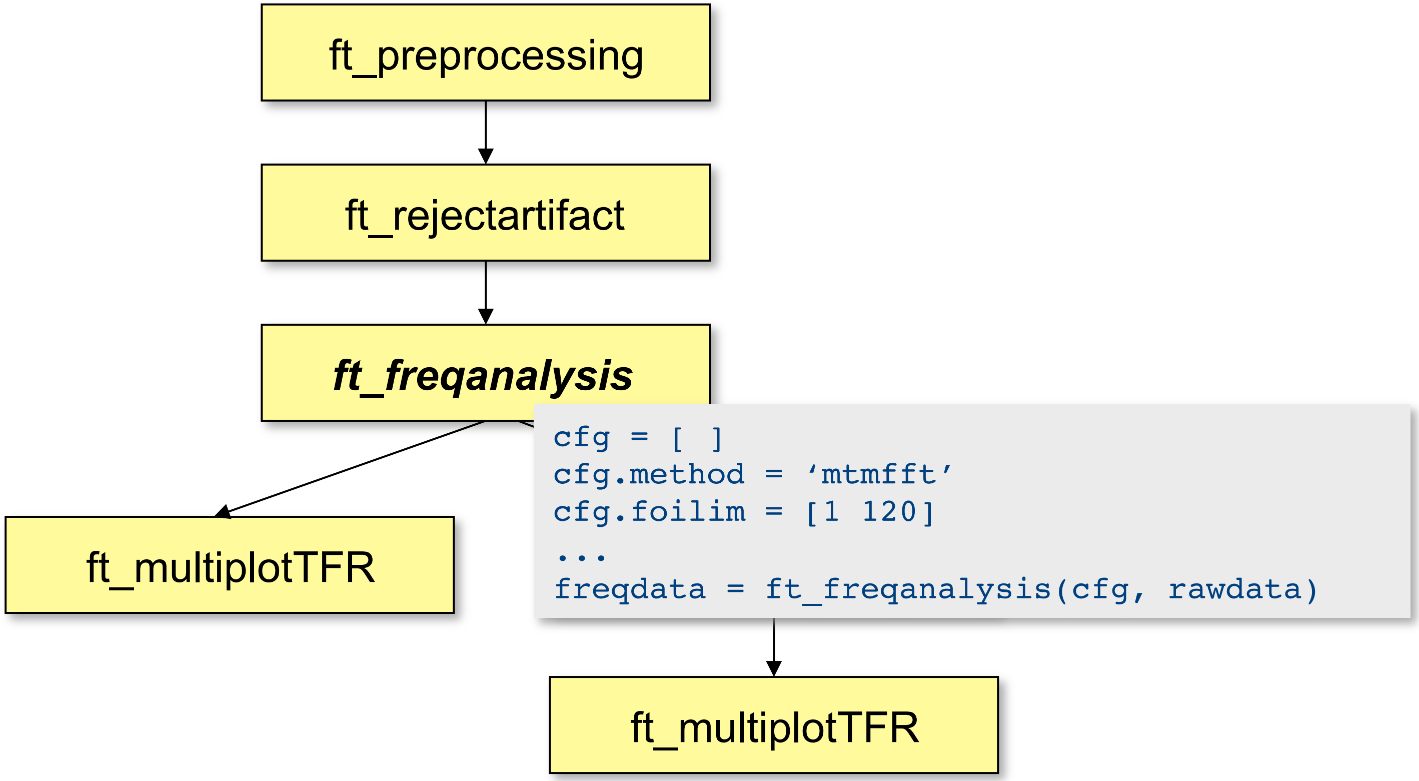
```
FT_PREPROCESSING reads MEG and/or EEG data according to user-specified
trials and applies several user-specified preprocessing steps to the
signals.

Use as
  [data] = ft_preprocessing(cfg)
or
  [data] = ft_preprocessing(cfg, data)

The first input argument "cfg" is the configuration structure, which
contains all details for the dataset filenames, trials and the
preprocessing options. You can only do preprocessing after defining the
segments of data to be read from the file (i.e. the trials), which is for
example done based on the occurence of a trigger in the data.

...
```

Using functions in an analysis protocol



```
cfg = [ ]
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)
```

Using functions in an analysis protocol



ft_preprocessing

ft_rejectartifact

ft_freqanalysis

ft_multiplotTFR

```
cfg = [ ]
cfg.method = 'mtmfft'
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)
```

ft_multiplotTFR

Raw data structure

```
rawData =
      label: {151x1 cell}
      trial: {1x87 cell}
       time: {1x87 cell}
        hdr: [1x1 struct]
        cfg: [1x1 struct]
```

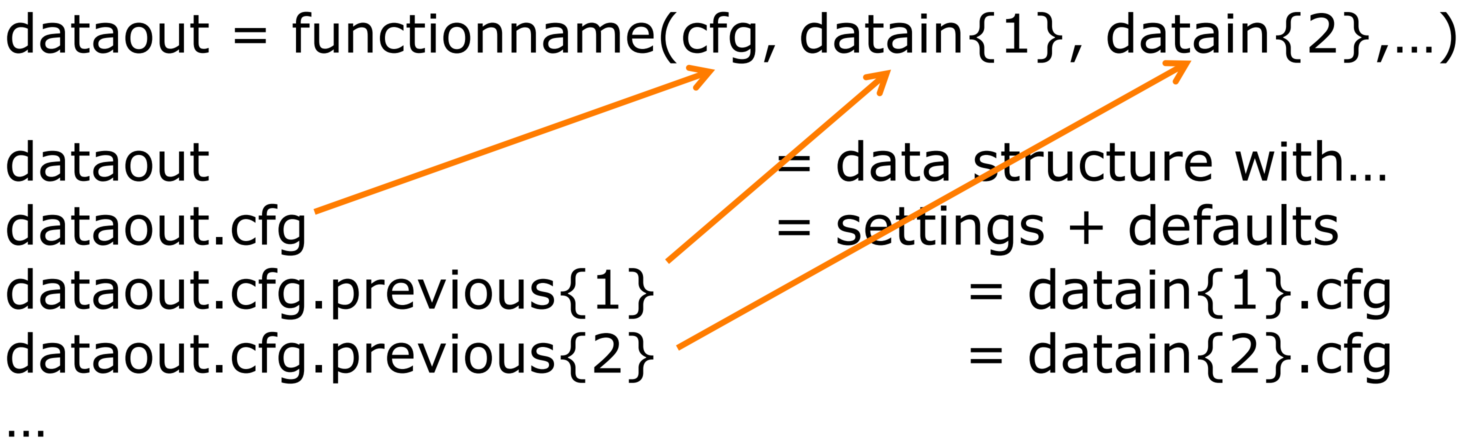
Event related response

```
erpData =
      label: {151x1 cell}
        avg: [151x900 double]
        var: [151x900 double]
       time: [1x900 double]
     dimord: 'chan_time’
        cfg: [1x1 struct]
```

Keeping track of your analysis

input cfg structure specifies parameters

output cfg structure keeps history



```
dataout = functionname(cfg, datain{1}, datain{2},…)

dataout                        = data structure with…
dataout.cfg                    = settings + defaults
dataout.cfg.previous{1}             = datain{1}.cfg
dataout.cfg.previous{2}             = datain{2}.cfg
…
```

details of computations are kept with data

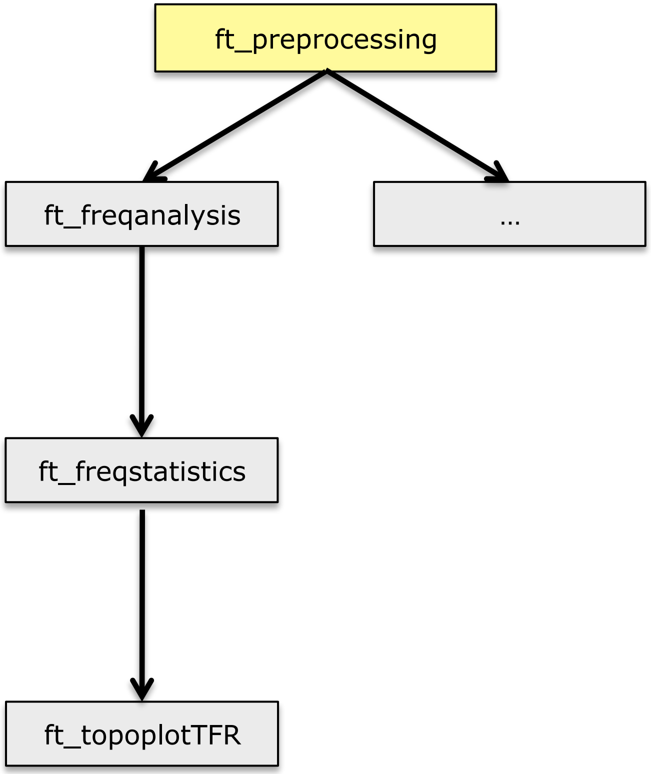
previous data is not kept, but can be reconstructed using cfg.previous.previous…

Example use in scripts

```
cfg = []
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)
```

```
cfg = []
cfg.method = ‘mtmfft’
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)
```

```
cfg = []
cfg.method = ‘montecarlo’
cfg.statistic = ‘indepsamplesT’
cfg.design = [1 2 1 2 2 1 2 1 1 2 ... ]
...
freqstat = ft_freqstatistics(cfg, freqdata)
```

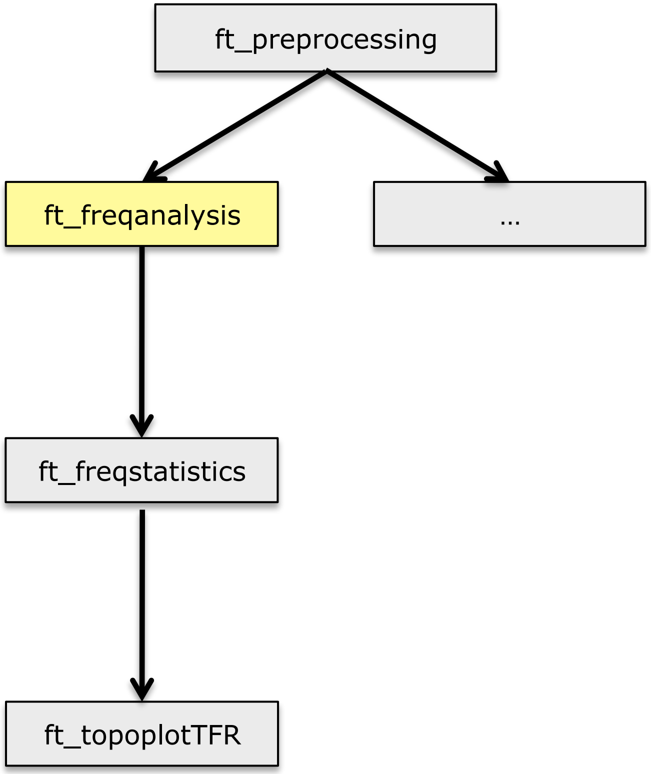


Example use in scripts

```
cfg = []
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)
```

```
cfg = []
cfg.method = ‘mtmfft’
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)
```

```
cfg = []
cfg.method = ‘montecarlo’
cfg.statistic = ‘indepsamplesT’
cfg.design = [1 2 1 2 2 1 2 1 1 2 ... ]
...
freqstat = ft_freqstatistics(cfg, freqdata)
```

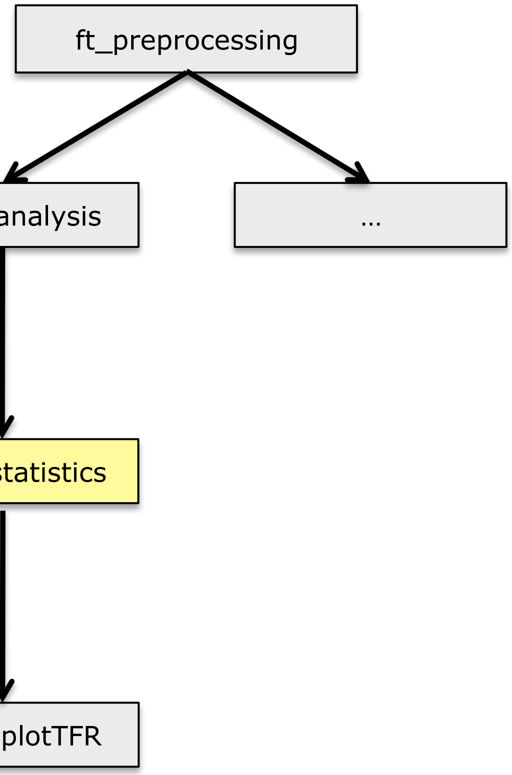


Example use in scripts

```
cfg = []
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)
```

```
cfg = []
cfg.method = ‘mtmfft’
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)
```

```
cfg = []
cfg.method = ‘montecarlo’
cfg.statistic = ‘indepsamplesT’
cfg.design = [1 2 1 2 2 1 2 1 1 2 ... ]
...
freqstat = ft_freqstatistics(cfg, freqdata)
```



ft_preprocessing

ft_freqanalysis

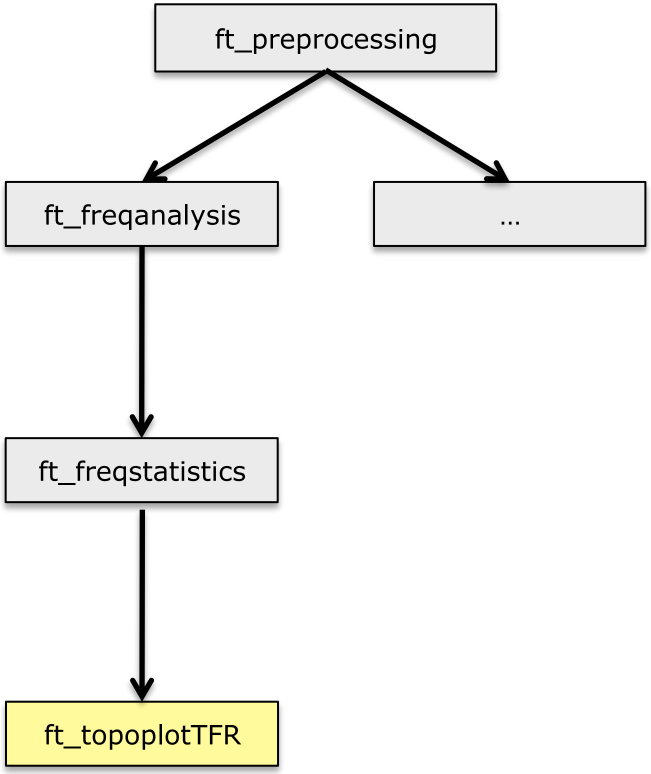
...

ft_freqstatistics

ft_topoplotTFR

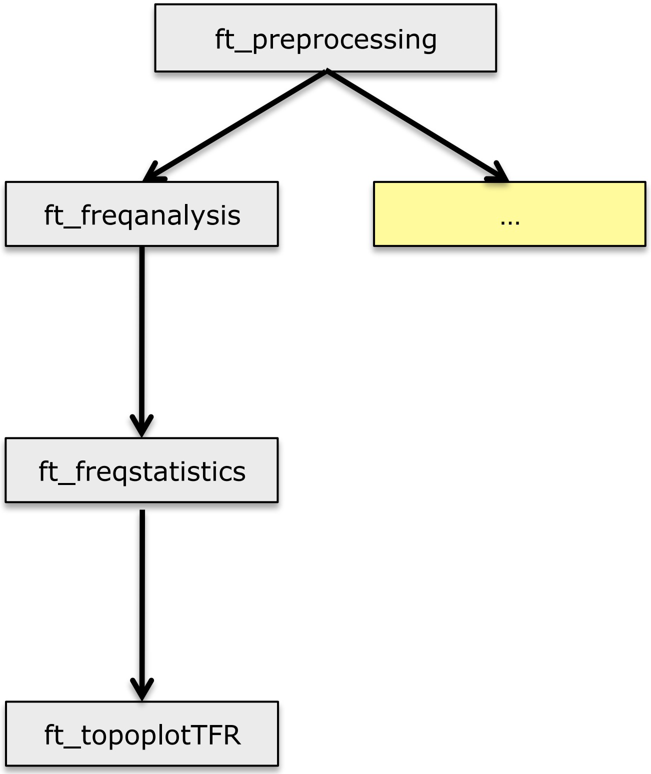
Example use in scripts

```
cfg = []
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)


cfg = []
cfg.method = ‘mtmfft’
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)


cfg = []
cfg.method = ‘montecarlo’
cfg.statistic = ‘indepsamplesT’
cfg.design = [1 2 1 2 2 1 2 1 1 2 ... ]
...
freqstat = ft_freqstatistics(cfg, freqdata)
```



Example use in scripts

```
cfg = []
cfg.dataset = ‘Subject01.ds’
cfg.bpfilter = [0.01 150]
...
rawdata = ft_preprocessing(cfg)


cfg = []
cfg.method = ‘mtmfft’
cfg.foilim = [1 120]
...
freqdata = ft_freqanalysis(cfg, rawdata)


cfg = []
cfg.method = ‘montecarlo’
cfg.statistic = ‘indepsamplesT’
cfg.design = [1 2 1 2 2 1 2 1 1 2 ... ]
...
freqstat = ft_freqstatistics(cfg, freqdata)
```



ft_preprocessing

ft_freqanalysis

...

ft_freqstatistics

ft_topoplotTFR

Example use in scripts

```
subj = {‘S01.ds’, ‘S02.ds’, …}
trig = [1 3 7 9]

for s=1:nsubj
for c=1:ncond

  cfg = []
  cfg.dataset   = subj{s}
  cfg.trigger   = trig(c)
  rawdata{s,c} = ft_preprocessing(cfg)

  cfg = []
  cfg.method    = ‘mtmfft’
  cfg.foilim    = [1 120]
  freqdata{s,c} = ft_freqanalysis(cfg, rawdata{s,c})

end
end
```

Example use in scripts

```
subj = {‘S01.ds’, ‘S02.ds’, …}
trig = [1 3 7 9]

for s=1:nsubj
for c=1:ncond

  cfg = []
  cfg.dataset   = subj{s}
  cfg.trigger   = trig(c)
  rawdata       = ft_preprocessing(cfg)


  filename = sprintf(‘raw%s_%d.mat’, subj{s}, trig(c));
  save(filename, ‘rawdata’)


end
end
```

Example use in distributed computing – with qsub

```
subj = {‘S01.ds’, ‘S02.ds’, …}
trig = [1 3 7 9]

for s=1:nsubj
for c=1:ncond

  cfgA{s,c} = []
  cfgA{s,c}.dataset    = subj{s}
  cfgA{s,c}.trigger    = trig(c)
  cfgA{s,c}.outputfile = sprintf(‘raw%s_%d.mat’, subj{s}, trig(c))

  cfgB{s,c} = []
  cfgB{s,c}.dataset    = subj{s}
  cfgB{s,c}.trigger    = trig(c)
  cfgB{s,c}.inputfile  = sprintf(‘raw%s_%d.mat’, subj{s}, trig(c));
  cfgB{s,c}.outputfile = sprintf(‘freq%s_%d.mat’, subj{s}, trig(c));

end
end

qsubcellfun(@ft_preprocessing, cfgA)
qsubcellfun(@ft_freqanalysis,  cfgB)
```

Example use in distributed computing – with dfeval

```
subj = {‘S01.ds’, ‘S02.ds’, …}
trig = [1 3 7 9]

for s=1:nsubj
for c=1:ncond

  cfgA{s,c} = []
  cfgA{s,c}.dataset    = subj{s}
  cfgA{s,c}.trigger    = trig(c)
  cfgA{s,c}.outputfile = sprintf(‘raw%s_%d.mat’, subj{s}, trig(c))

  cfgB{s,c} = []
  cfgB{s,c}.dataset    = subj{s}
  cfgB{s,c}.trigger    = trig(c)
  cfgB{s,c}.inputfile  = sprintf(‘raw%s_%d.mat’, subj{s}, trig(c));
  cfgB{s,c}.outputfile = sprintf(‘freq%s_%d.mat’, subj{s}, trig(c));

end
end

dfeval(@ft_preprocessing, cfgA)
dfeval(@ft_freqanalysis,  cfgB)
```

FieldTrip is a toolbox

- the data and the separate functions are in the hands of the end-users

- the scripts depend on the data properties, available memory and programming skills and style

- scripts correspond to analysis protocols
  - scripts can be reviewed by supervisors
  - scripts are often shared with colleagues

Finding your way around in the FieldTrip toolbox

Matlab

help functionname

edit functionname

Website

http://www.fieldtriptoolbox.org

Email discussion list

Expertise in your local group

Talk outline

Recording signals from the brain

Temporal and spatial features of the signals

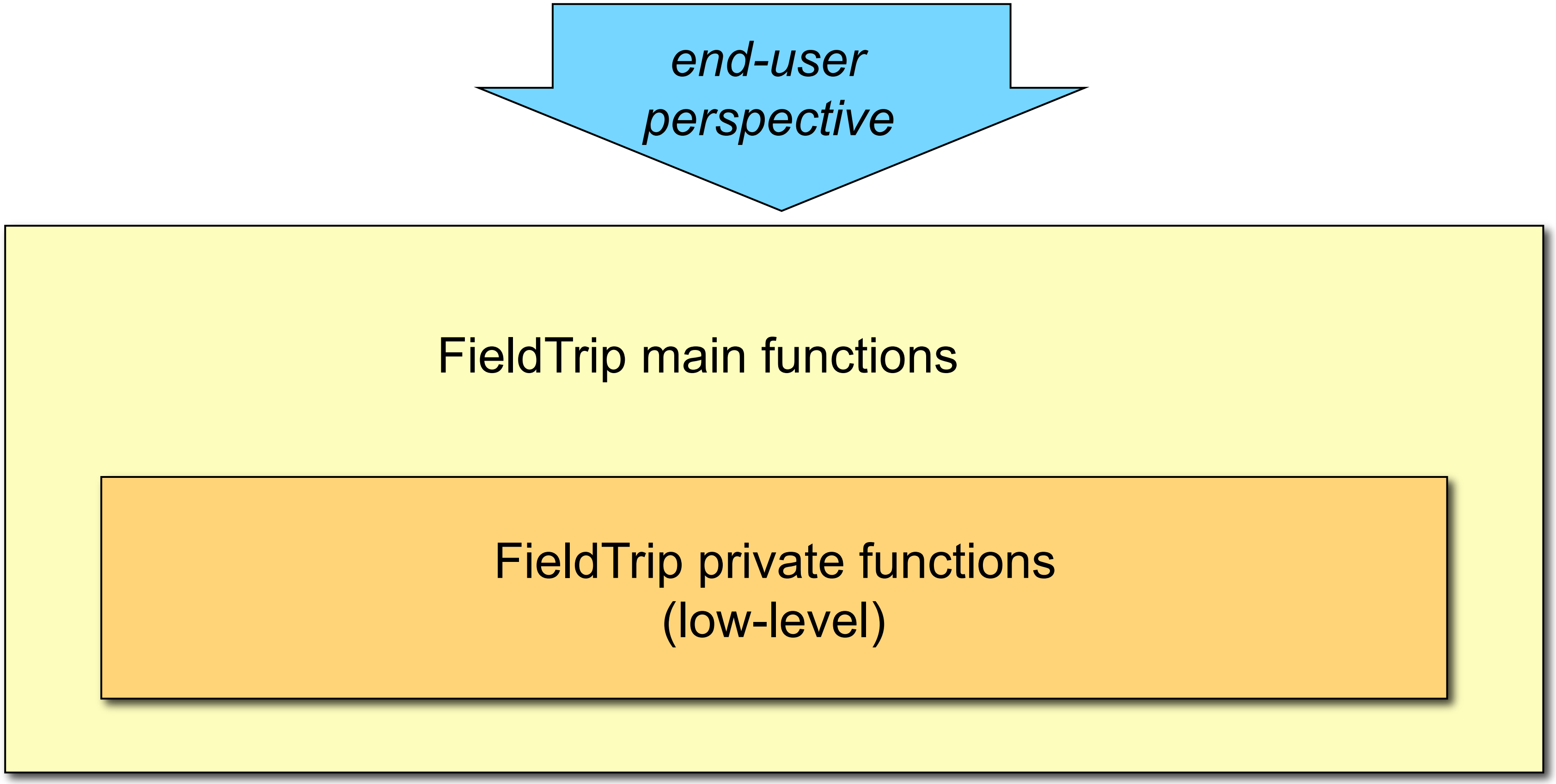
Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Who is the audience?

- experimental neuroscientists
  - no graphical user interface
  - more dedicated and ambitious researchers
- developers of other software packages
  - SPM
  - EEGLAB
  - BESA
  - BCI2000
  - SIMBIO
  - OpenMEEG
- developers of analysis methods

FieldTrip toolbox structure - at a glance

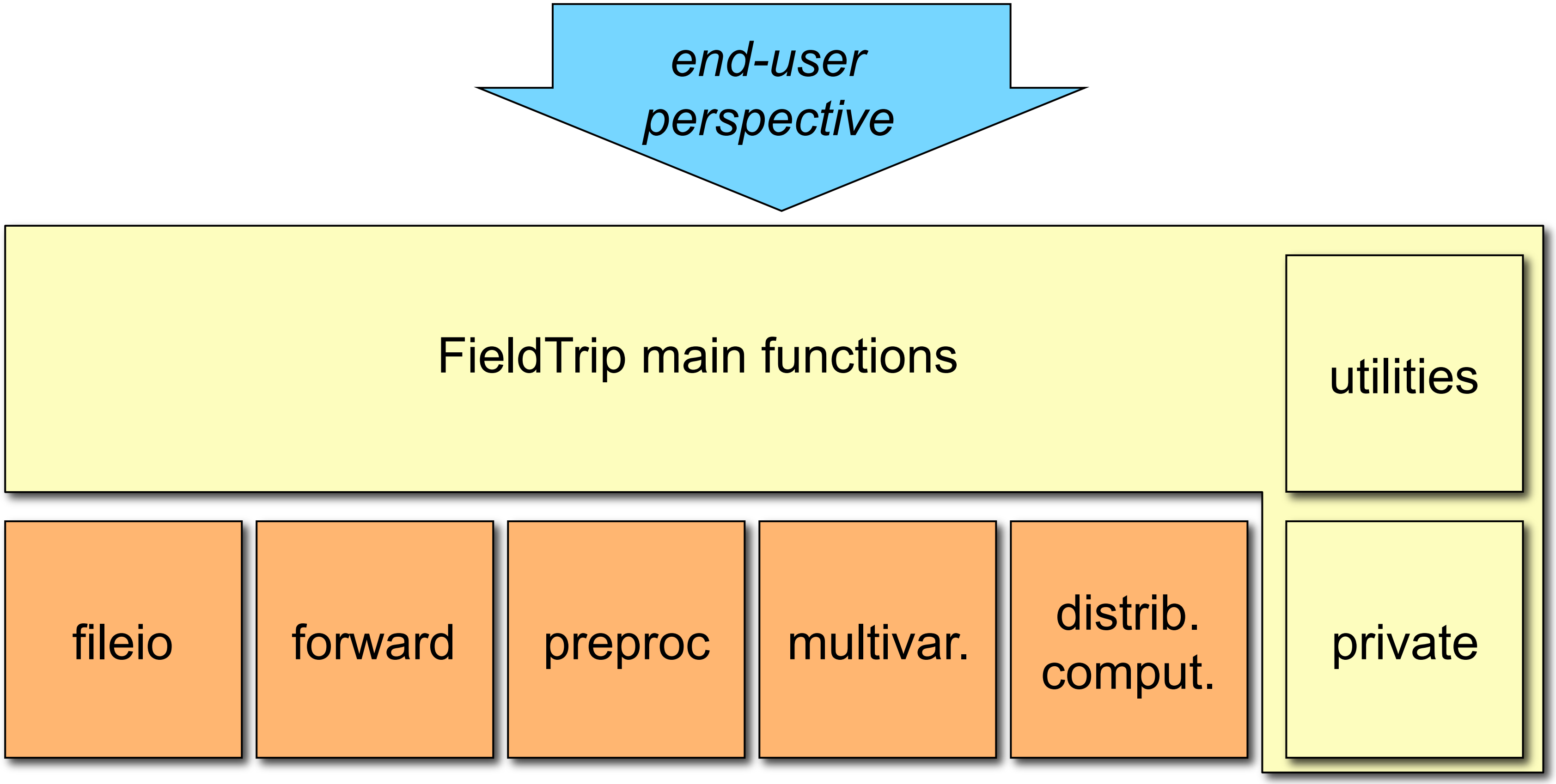


end-user perspective

FieldTrip main functions

FieldTrip private functions (low-level)

FieldTrip toolbox structure - a closer look



end-user perspective

FieldTrip main functions

utilities

fileio

forward

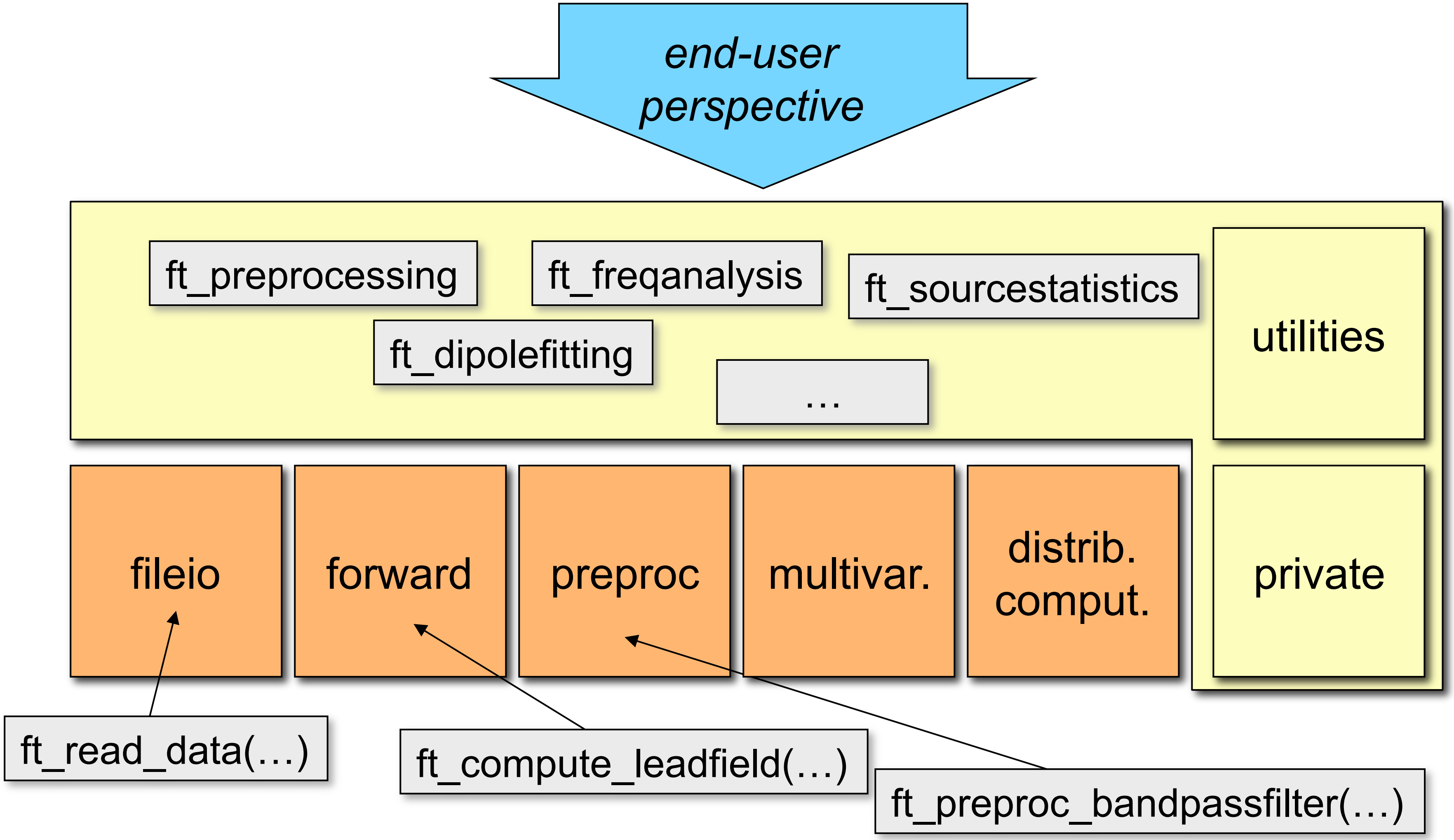
preproc

multivar.

distrib. comput.

private

FieldTrip toolbox structure - a closer look



Summary

Recording signals from the brain

Temporal and spatial features of the signals

Analyzing those signals with FieldTrip

Background on the FieldTrip project

Hands-on

Selecting pieces of data

Preprocessing

Averaging

Plotting