Introduction to EEG, MEG and analysis with the FieldTrip toolbox

Robert Oostenveld

Donders Institute, Radboud University, Nijmegen, NL
NatMEG, Karolinska Institute, Stockholm, SE

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

What kind of signals are generated in the brain

How do we record those signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Talk outline

What kind of signals are generated in the brain

How do we record those signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Talk outline

What kind of signals are generated in the brain

How do we record those signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Talk outline

What kind of signals are generated in the brain

How do we record those signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

M/EEG signal characteristics considered during analysis

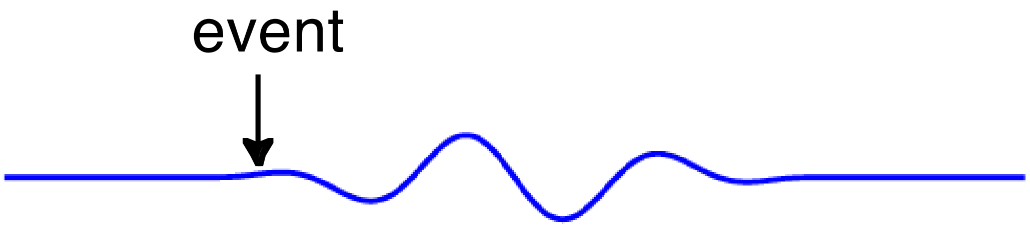
timecourse of activity
-> ERP

spectral characteristics
-> power spectrum

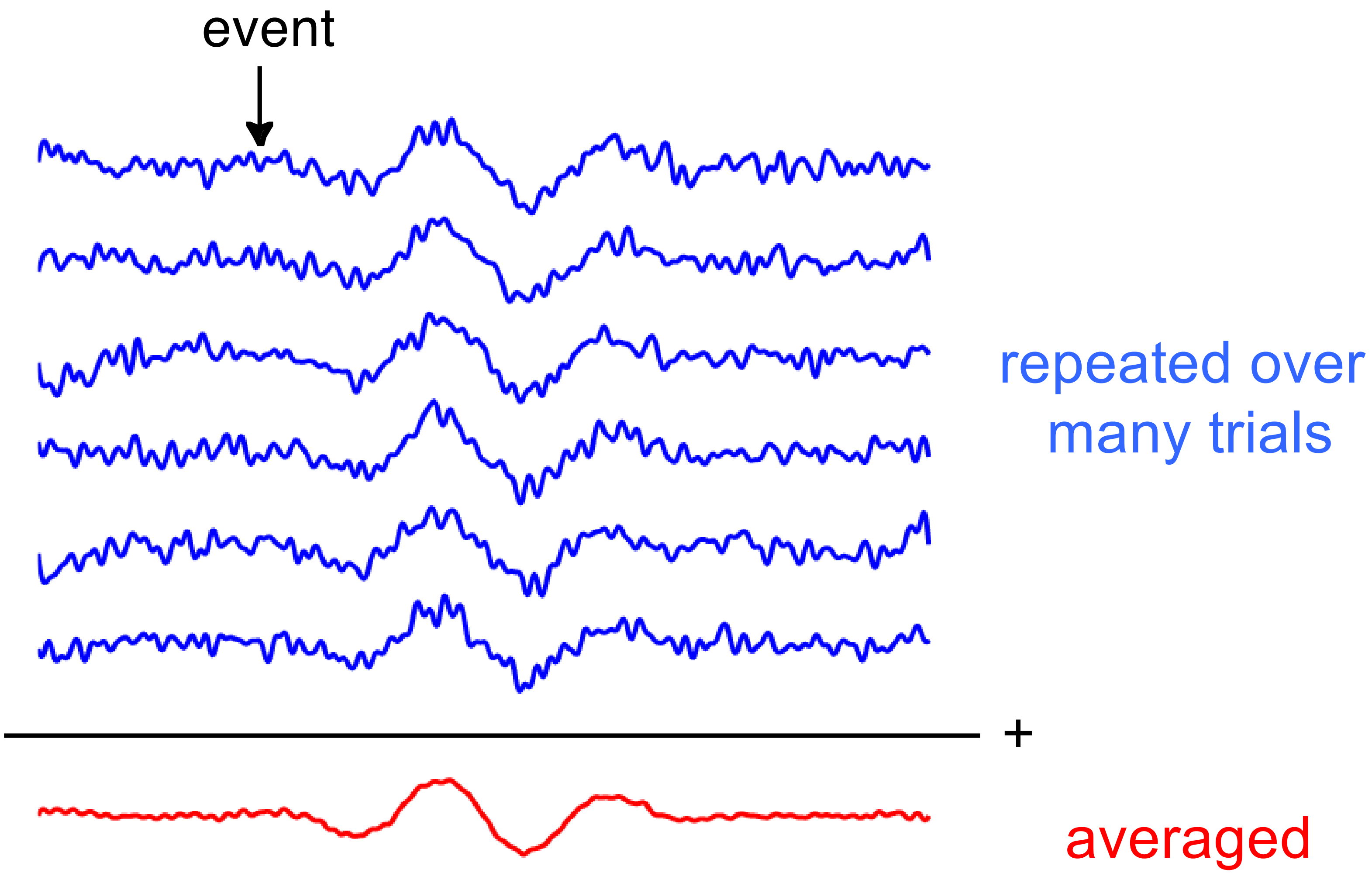
temporal changes in power
-> time-frequency response (TFR)

spatial distribution of activity over the head
-> source reconstruction

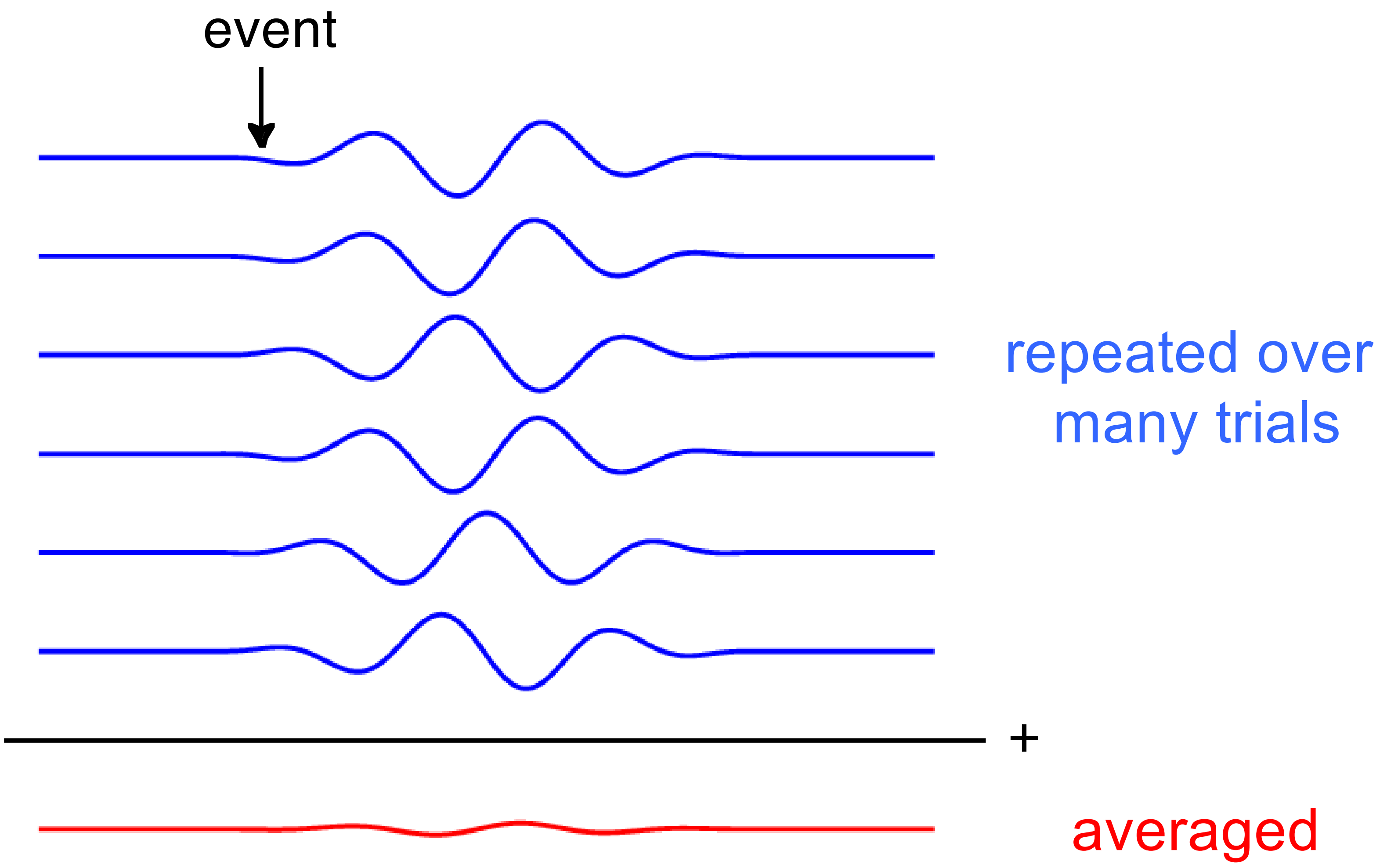
Evoked activity



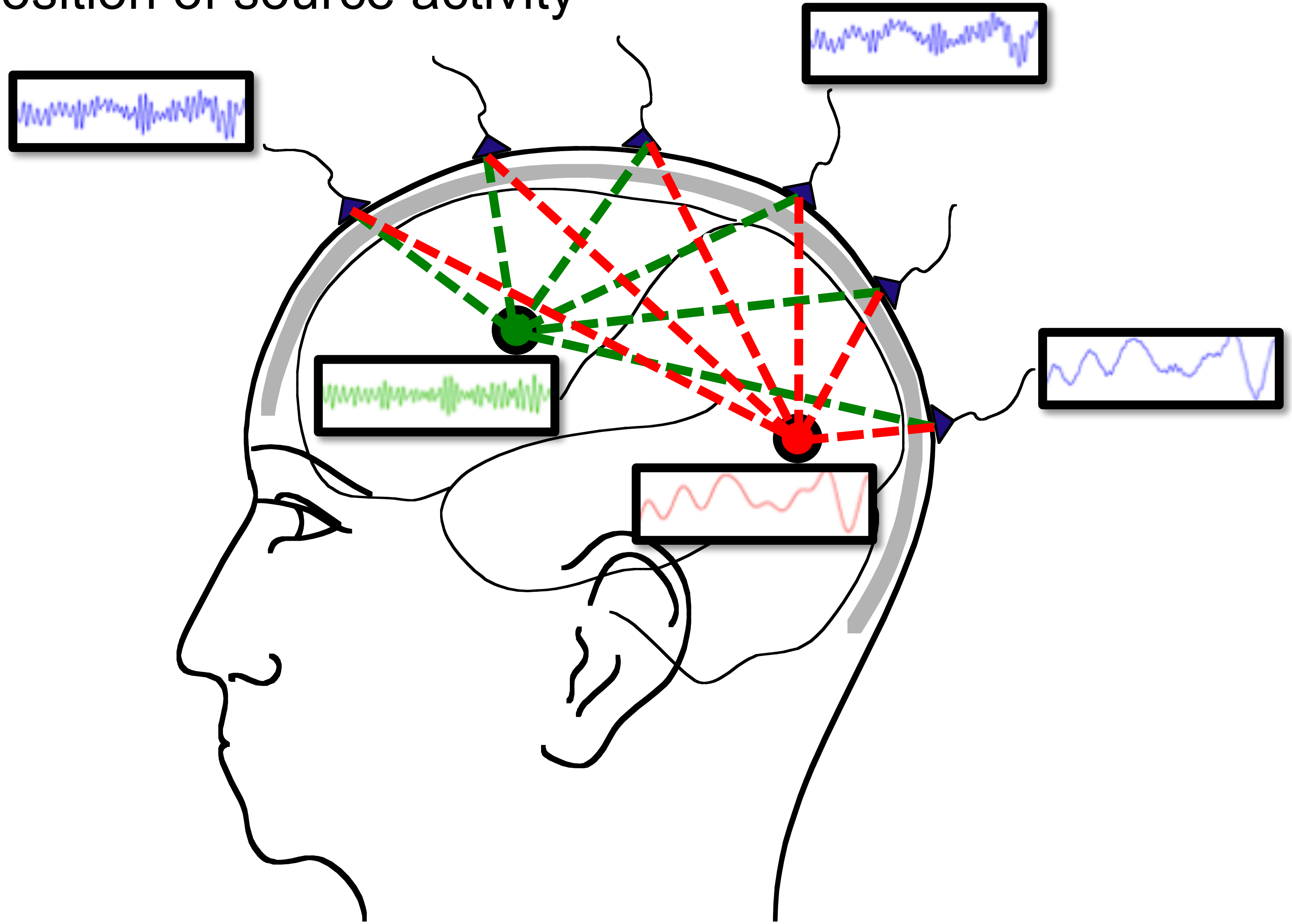
Evoked activity



Induced activity



Superposition of source activity



Separating activity of sources

- 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

What kind of signals are generated in the brain

How do we record those signals

Analyzing those signals with FieldTrip

Background on the FieldTrip toolbox

Some FieldTrip basics

dataout = functionname(cfg, datain, …)

functionname(

dataout = func

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

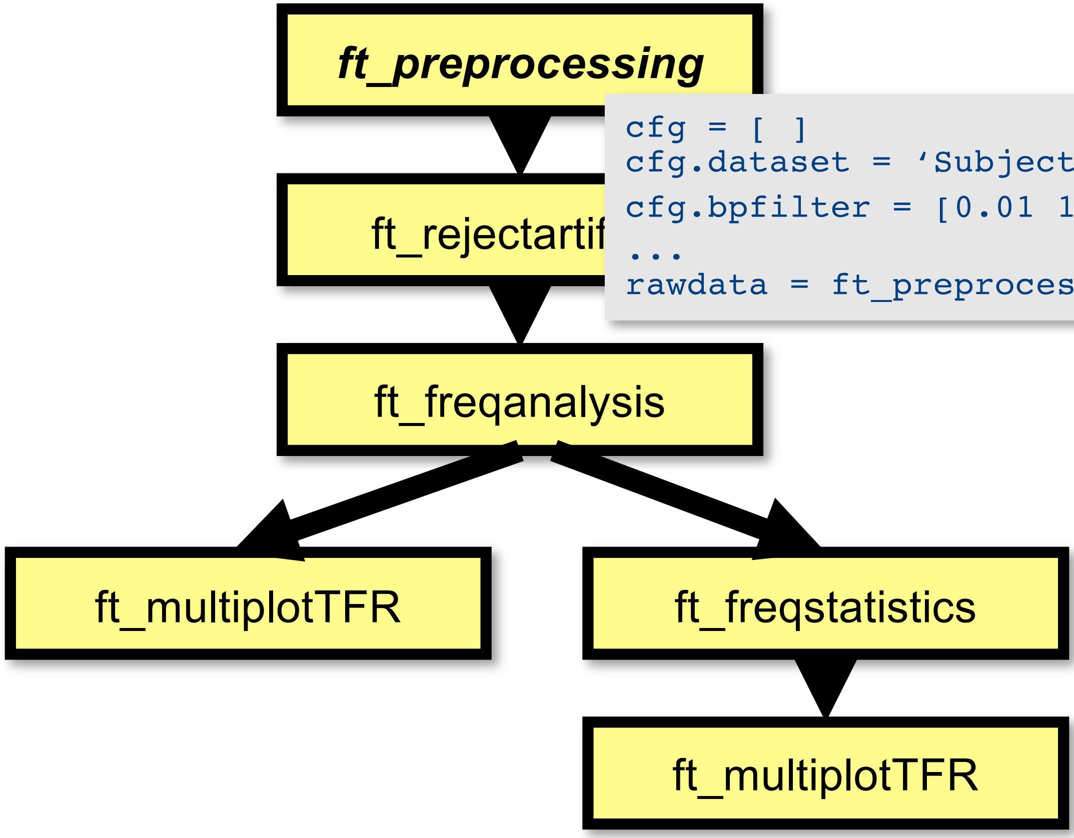
```
FT_PREPROCESSING reads MEG and/or EEG data according to user-specified
trials and applies several user-specified preprocessing steps to the
signals.

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



ft_preprocessing

ft_rejectartif

ft_freqanalysis

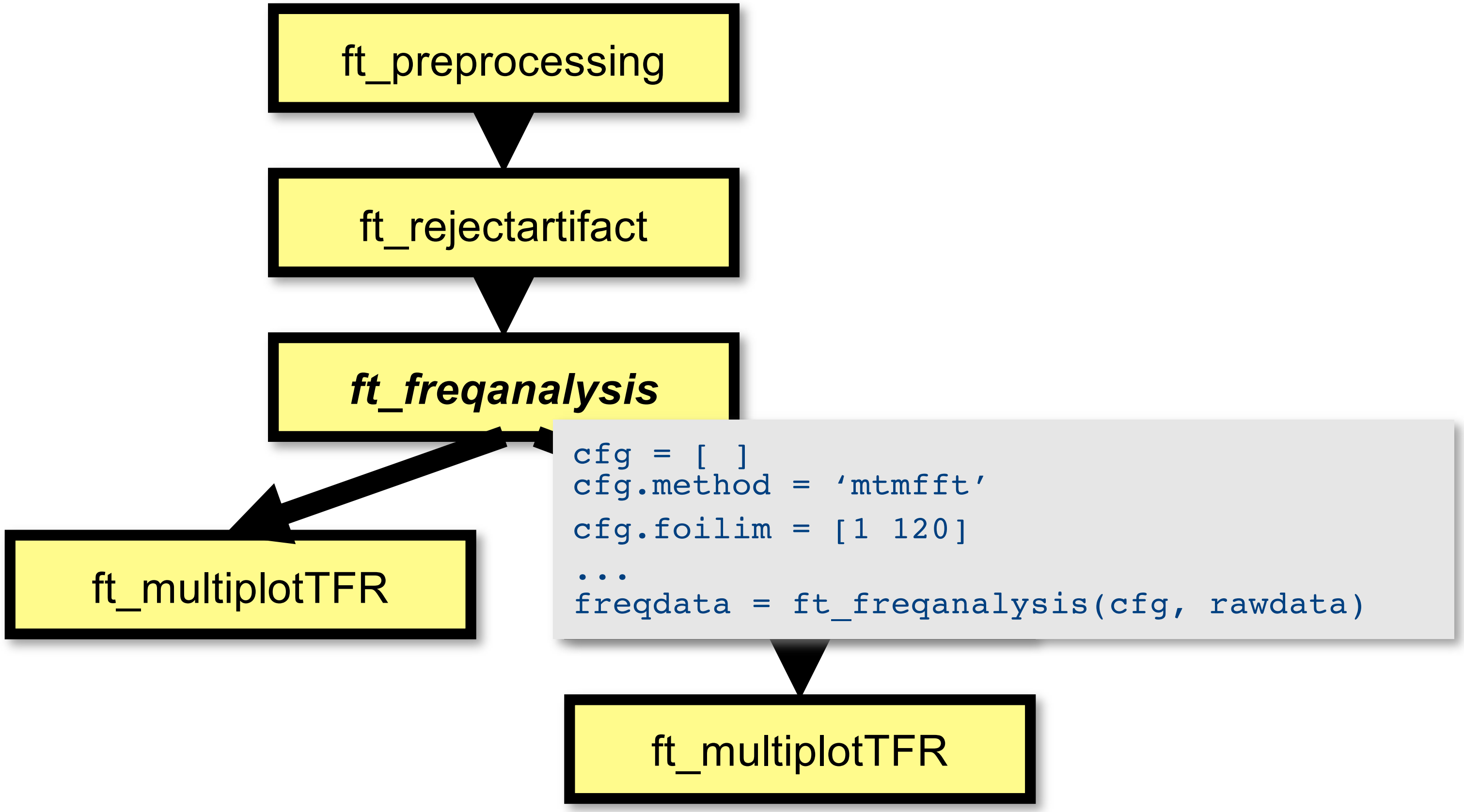
ft_multiplotTFR

ft_freqstatistics

ft_multiplotTFR

```
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

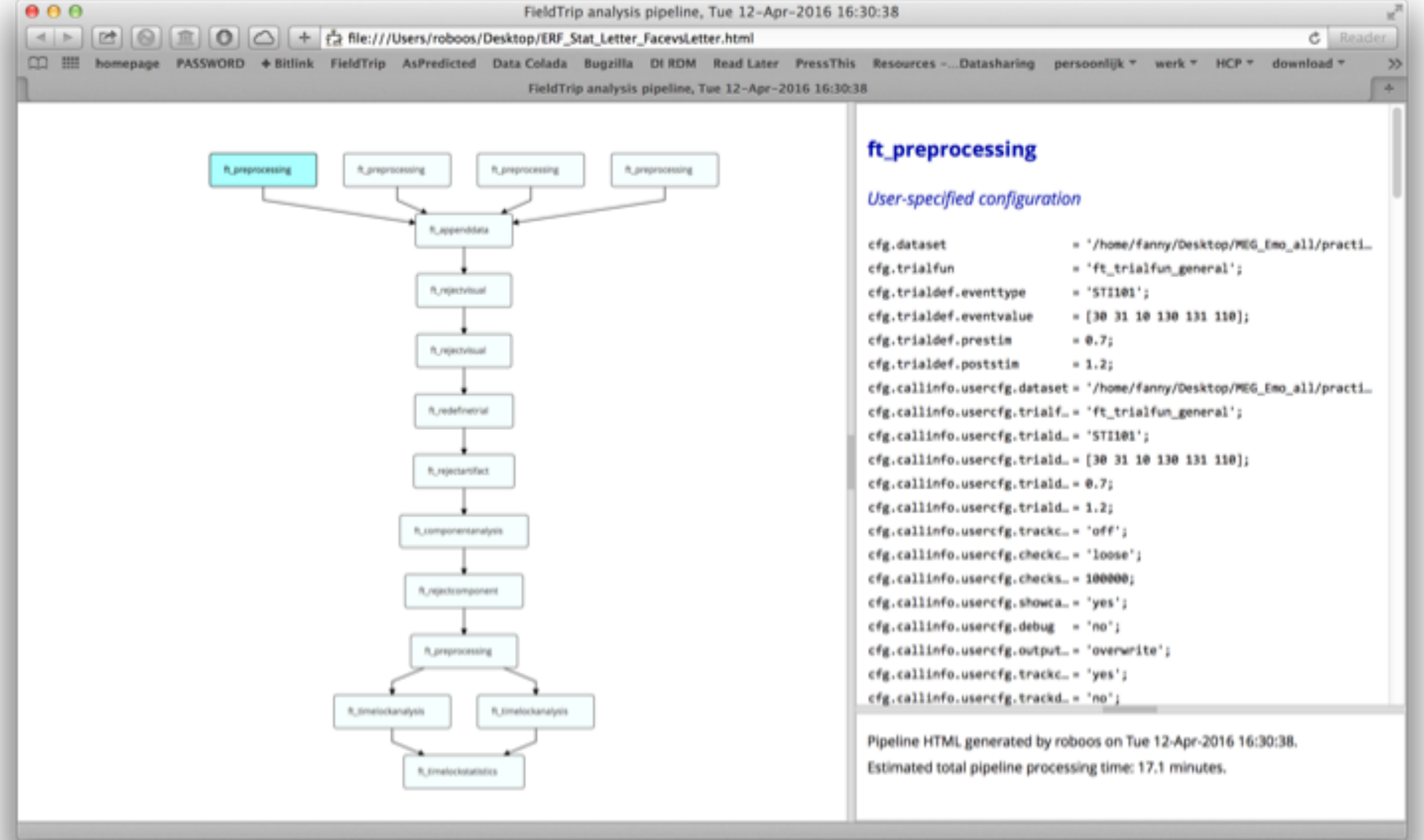
Raw data structure

```
rawData =
       label: {151x1 cell}
       trial: {1x80 cell}
        time: {1x80 cell}
     fsample: 300
         hdr: [1x1 struct]
         cfg: [1x1 struct]
```

Event related response

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

Keeping track of your analysis



ft_preprocessing

User-specified configuration

```
cfg.dataset                = '/home/fanny/Desktop/MEG_Emo_all/practi…
cfg.trialfun               = 'ft_trialfun_general';
cfg.trialdef.eventtype     = 'STI101';
cfg.trialdef.eventvalue    = [30 31 10 130 131 110];
cfg.trialdef.prestim       = 0.7;
cfg.trialdef.poststim      = 1.2;
cfg.callinfo.usercfg.dataset = '/home/fanny/Desktop/MEG_Emo_all/practi…
cfg.callinfo.usercfg.trialf… = 'ft_trialfun_general';
cfg.callinfo.usercfg.triald… = 'STI101';
cfg.callinfo.usercfg.triald… = [30 31 10 130 131 110];
cfg.callinfo.usercfg.triald… = 0.7;
cfg.callinfo.usercfg.triald… = 1.2;
cfg.callinfo.usercfg.trackc… = 'off';
cfg.callinfo.usercfg.checkc… = 'loose';
cfg.callinfo.usercfg.checks… = 100000;
cfg.callinfo.usercfg.showca… = 'yes';
cfg.callinfo.usercfg.debug   = 'no';
cfg.callinfo.usercfg.output… = 'overwrite';
cfg.callinfo.usercfg.trackc… = 'yes';
cfg.callinfo.usercfg.trackd… = 'no';
```

Pipeline HTML generated by roboos on Tue 12-Apr-2016 16:30:38.

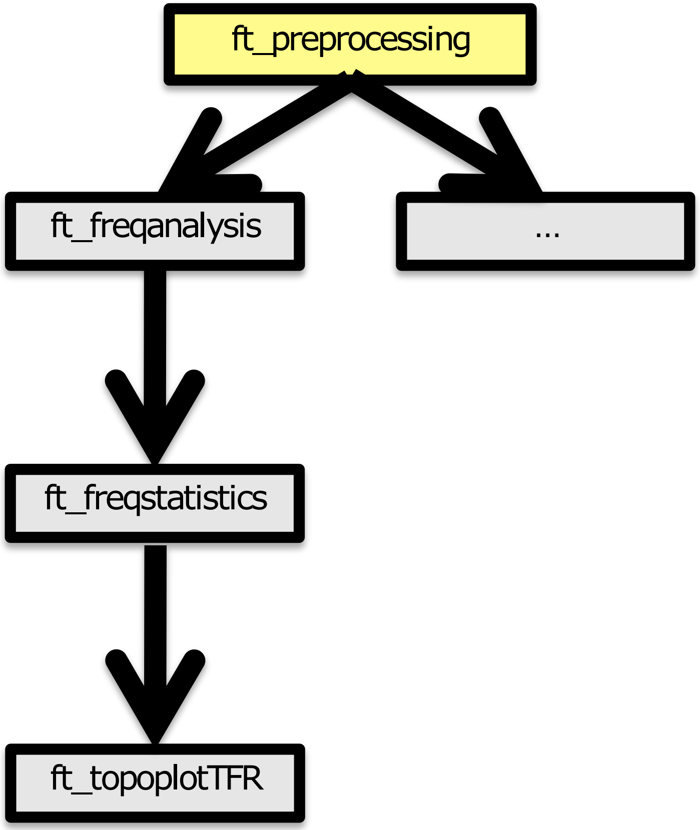
Estimated total pipeline processing time: 17.1 minutes.

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

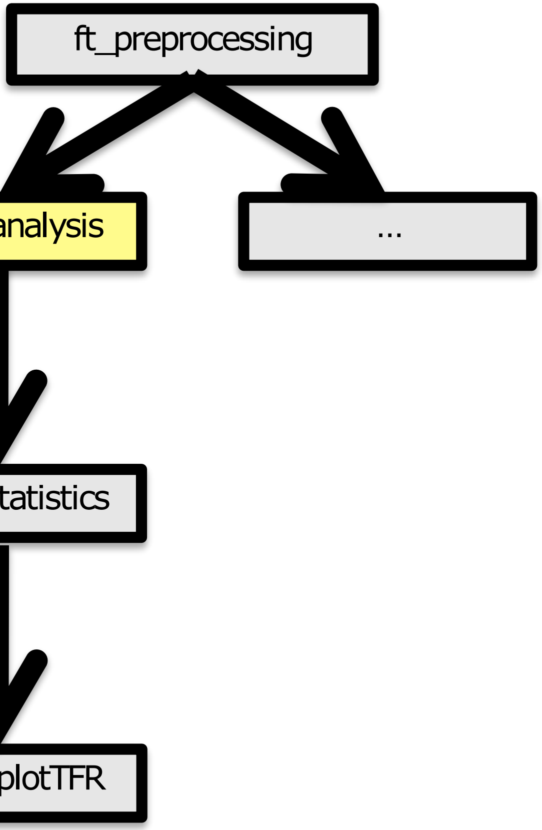
ft_topoplotTFR

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)
```

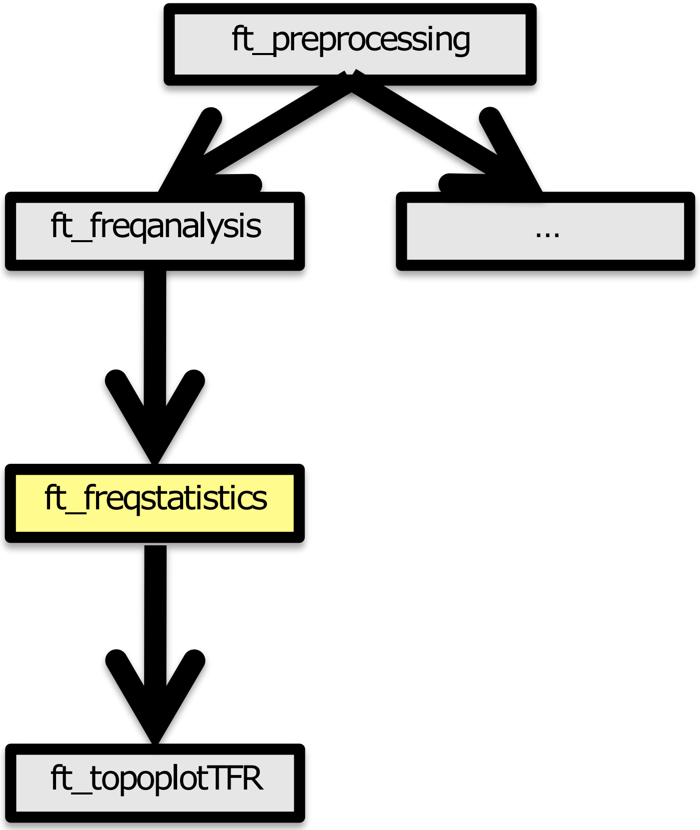


Example use in scripts

```
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

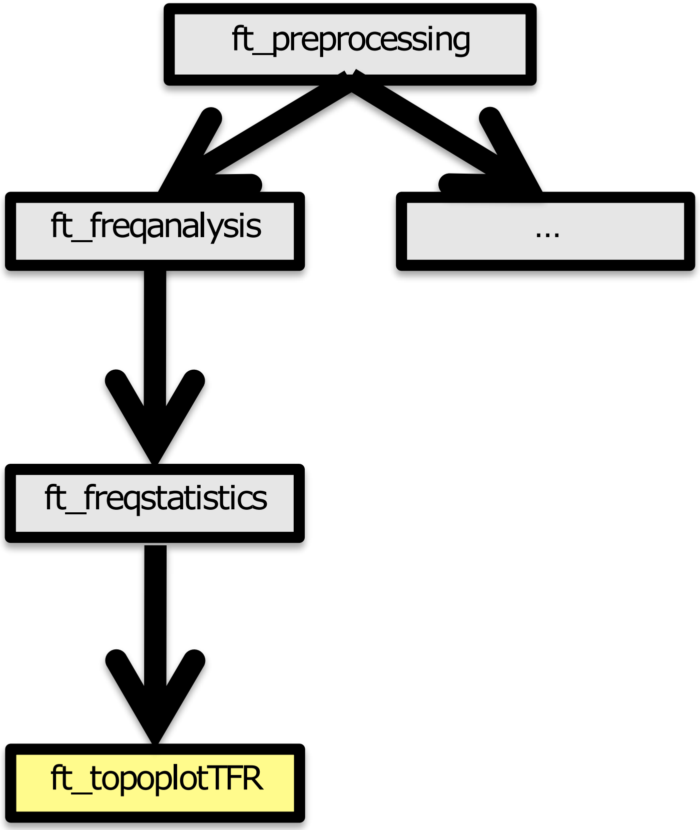
ft_topoplotTFR

Example use in scripts

```
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)
```

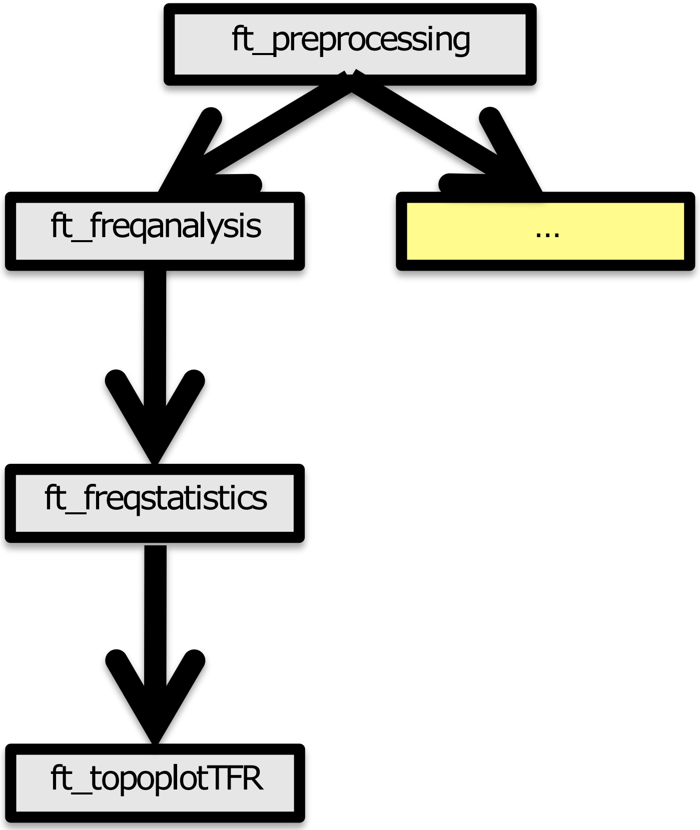


Example use in scripts

```
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

```
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)
```

Example use in distributed computing

```
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)
```

FieldTrip is a toolbox

the data and the separate functions are in your hands

the scripts depend on the data properties, your computer and on your programming skills and style

scripts correspond to analysis protocols

- scripts can be reviewed by supervisors
- scripts are often shared with colleagues
- scripts can be published/released

Starting point for MEG Nord 2018 workshops

"*Group Analysis in FieldTrip of Time-Frequency Responses: A Pipeline for Reproducibility at Every Step of Processing, Going From Individual Sensor Space Representations to an Across-Group Source Space Representation*" by Lau Moller Andersen https://doi.org/10.3389/fnins.2018.00261

The analysis scripts have been published and are available from https://github.com/ualsbombe/omission_frontiers

The original raw data has been shared on https://doi.org/10.5281/zenodo.998518