Time Series Analysis with FieldTrip

Thanks to: Robert Oostenveld

Some FieldTrip basics

Some FieldTrip basics

dataout = functionname(cfg, datain, …)

functionname(cfg, datain, …)

dataout = functionname(cfg)

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

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

FieldTrip data structures

- Contain all relevant details
- As small as possible

```
rawData =
        label: {151x1 cell}
        trial: {1x87 cell}
         time: {1x87 cell}
      fsample: 300
          hdr: [1x1 struct]
          cfg: [1x1 struct]
```

FieldTrip data structures

- Contain all relevant details
- As small as possible

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

Keeping track of what you've done…

- cfg structure binds it together
- 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 = preprocessing(cfg)

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

cfg = [ ]
cfg.channels = ‘EEG1020’
…
erpdata = timelockanalysis(cfg, rawdata)
```

Example use in scripts

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

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

  cfg = [ ]
  cfg.dataset = subjlist{s}
  cfg.trigger = triglist(c)
  rawdata{s,c} = preprocessing(cfg)

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

end
end
```

Example use in scripts

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

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

  cfg = [ ]
  cfg.dataset = subjlist{s}
  cfg.trigger = triglist(c)
  rawdata = preprocessing(cfg)

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

  …

end
end
```

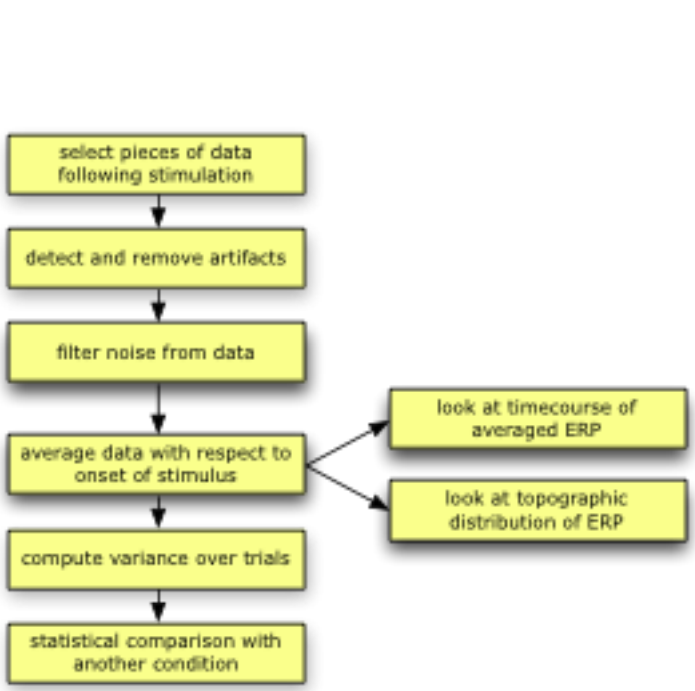
Example use in scripts

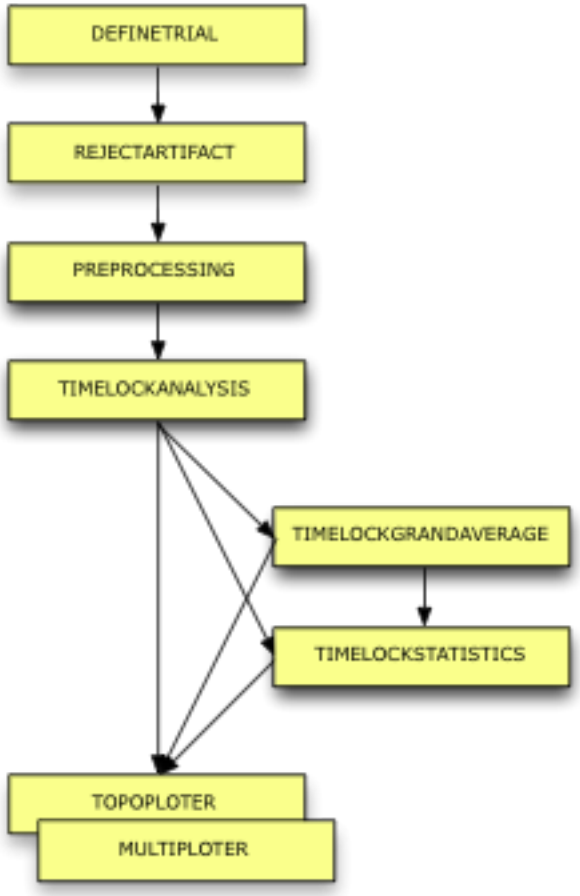
```
function mypreprocessing(subj, trig)
  cfg = [ ]
  cfg.dataset = subj
  cfg.trigger = trig
  cfg.bpfilter = [0.01 100]
  rawdata = preprocessing(cfg)
  save(sprintf(…))


% subsequently in script
for s=1:nsubj
for c=1:ncond
  mypreprocessing(…)
  myfreqanalysis(…)
end
end
```

FieldTrip is a toolbox

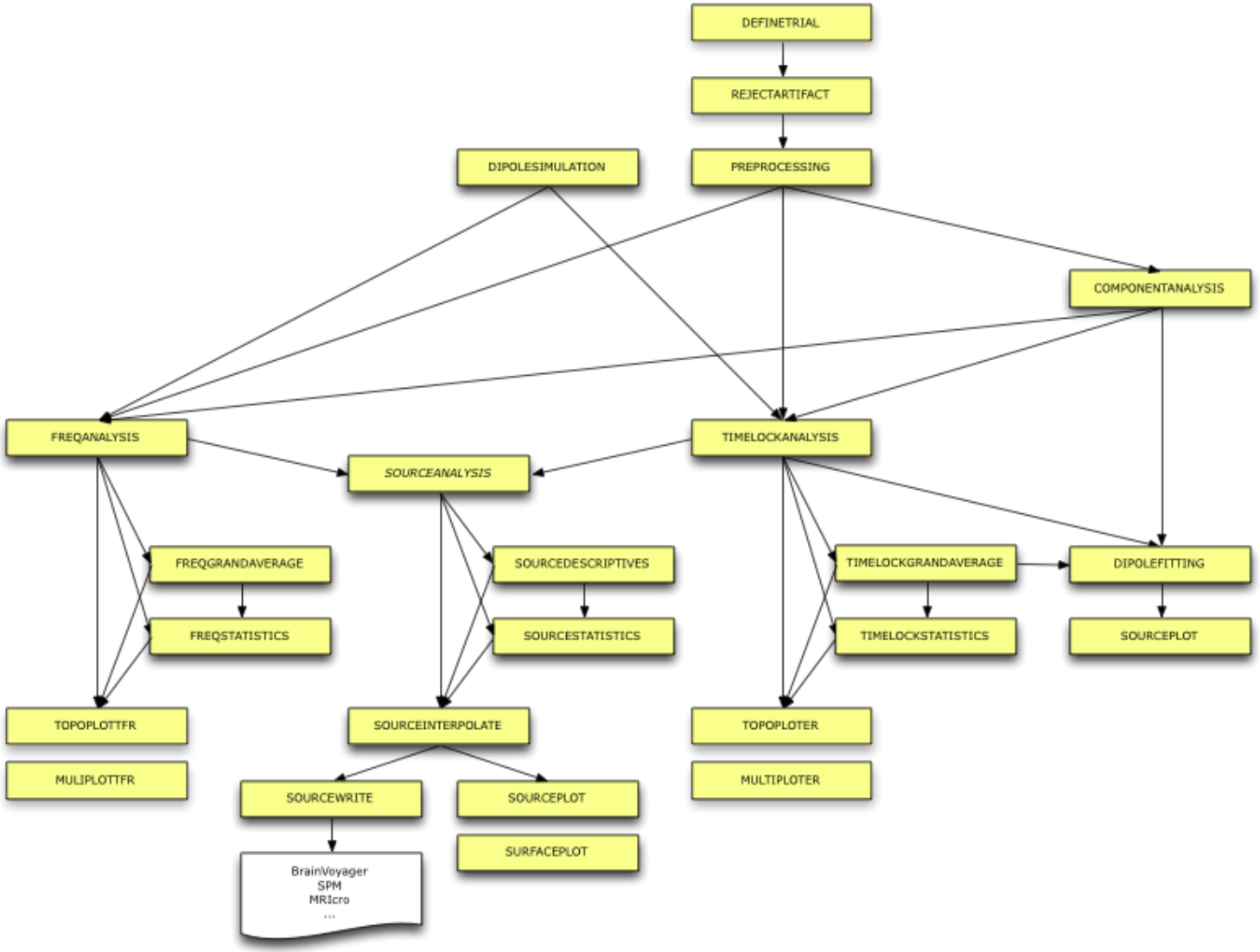
- the data and the separate functions are in the hands of the end-users
- the scripts depend on the data properties, available memory and programming skills and style
- scripts are often shared
- scripts correspond to analysis protocols

One-to-one mapping between analysis steps and FT functions





Overview of main functions



Most important take home message of the day:

http://fieldtrip.fcdonders.nl/

Most important take home message of the day (part II):

If you want to run fieldtrip within the department, there's a recent version on:
/home/jan/public/

Any questions sofar?

...

Go to FT01.m

Spectral analysis

- what
  - decompose data in sine and cosine “basis functions”
  - c.f. fMRI, general linear model with sine and cosine regressors
- why
  - oscillatory phenomena
  - recurrent networks
  - periodic activation

Recipe

- frequency analysis
  - take a piece of sampled data
  - try to explain it with a full set of sine and cosine functions
- time-frequency analysis
  - take a smaller piece of the sampled data
  - try to explain it with a full set of sine and cosine functions
  - go to the next timepoint and repeat…

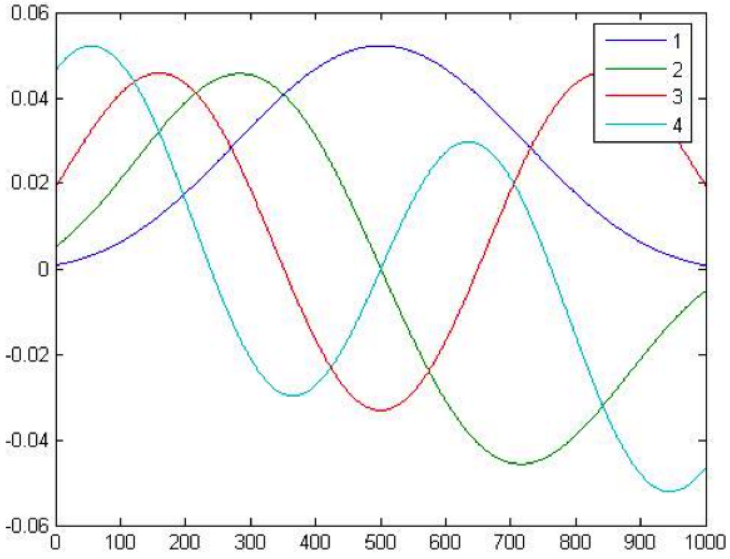
Tapers (windowing functions)

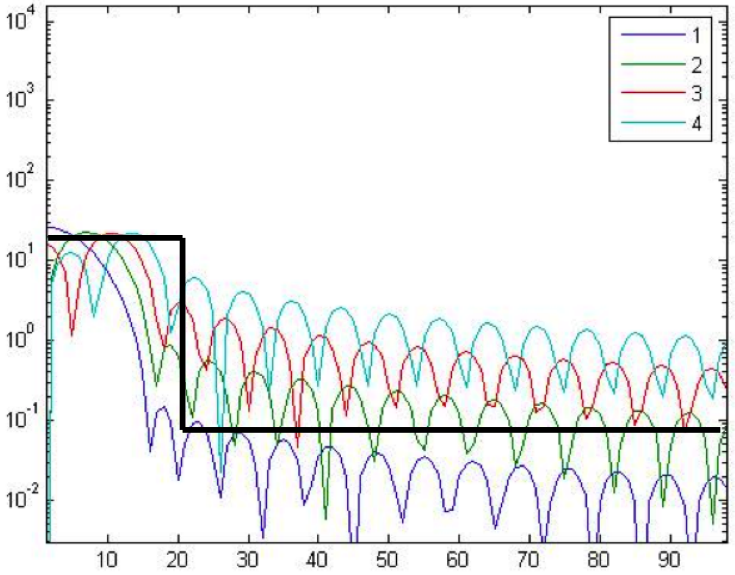
- Discrete Fourier transform
- Limited time available for FFT
  - boxcar window

Multitapers

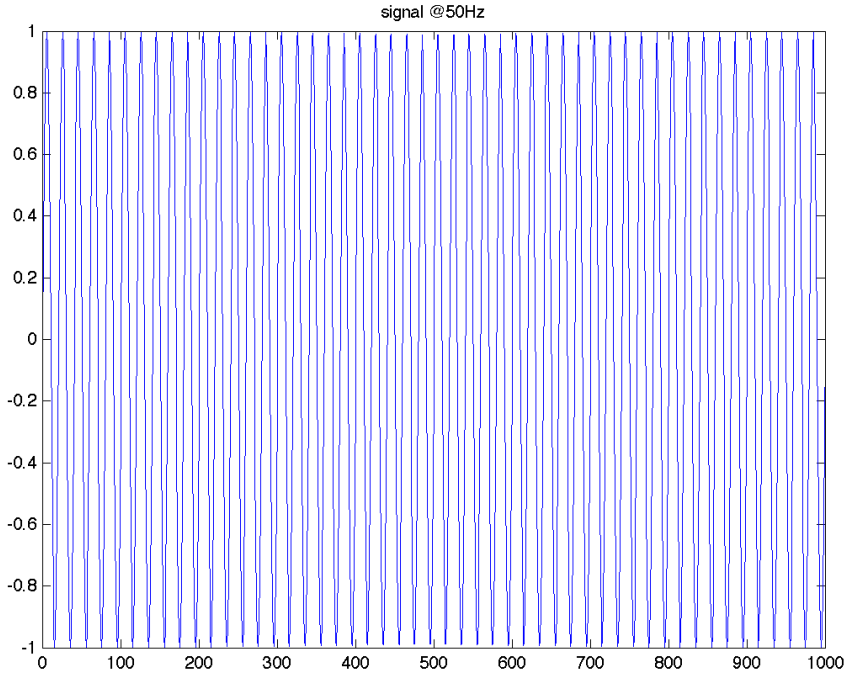
- Make use of more than one taper and combine their properties
- Used for spectral smoothing in the frequency domain
- Shape and number of tapers determines the frequency smoothing
- Instead of “smoothing” one can also say “controlled leakage”

Multitapers

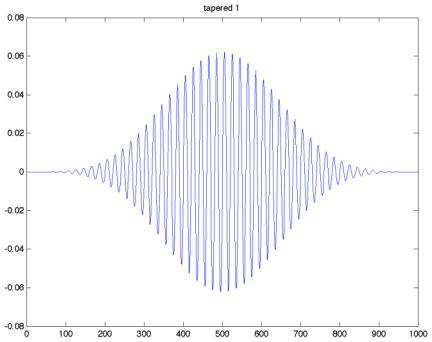


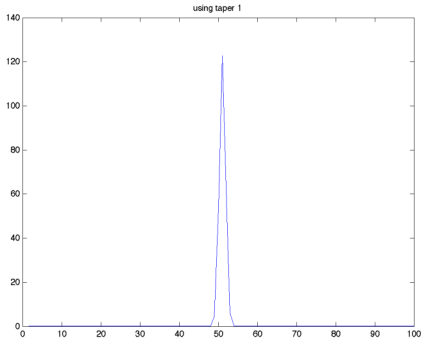


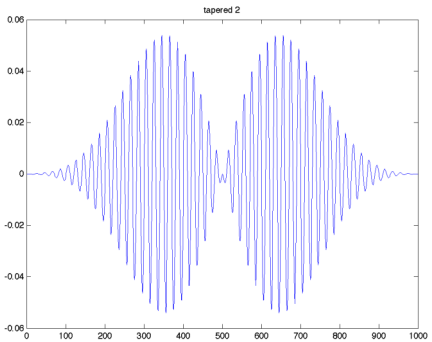
Multitapers demo

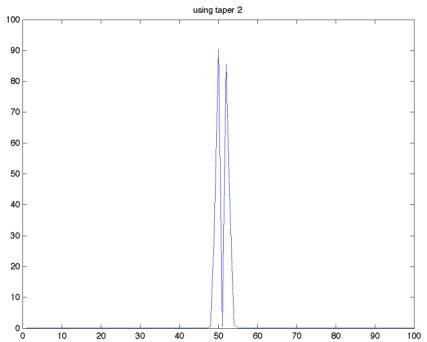


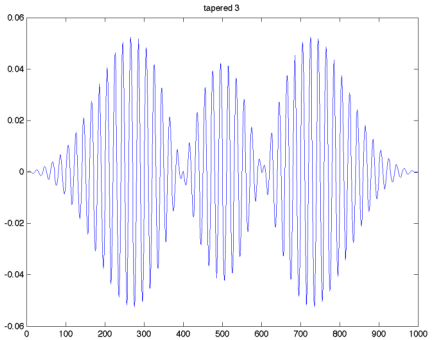
tapered signal spectral estimate

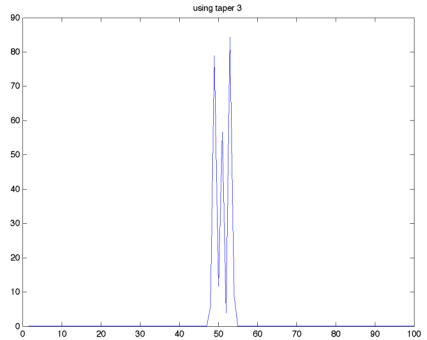




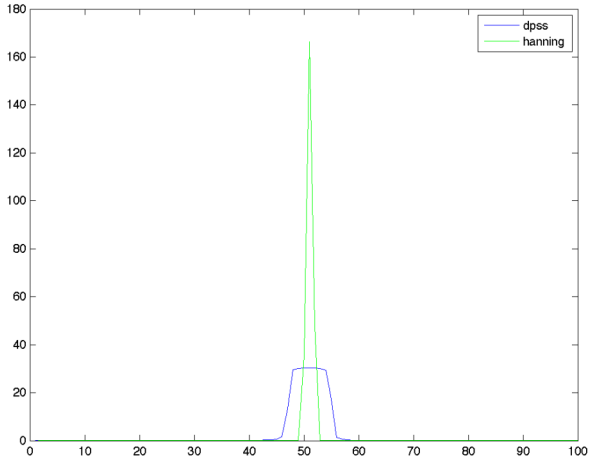








averaged spectral estimate



Multitapering effect

- narrowband signal
  - controlled spectral leakage
- broadband signal
  - spectral smoothing/concentration
- not only for power, also coherence
- in case of wavelet analysis: additional trade-off between time- and frequency resolution

Go to the hands-on…
FT02_multitaper.m