Tutorial 3

Computing time-series in brain regions

By Isotta Rigoni & Nicolas Roehri

Brain Dynamics on the Connectome
Summer School 2021

LEMANIC NEURO SCIENCE

UNIL | Université de Lausanne

ReproNim

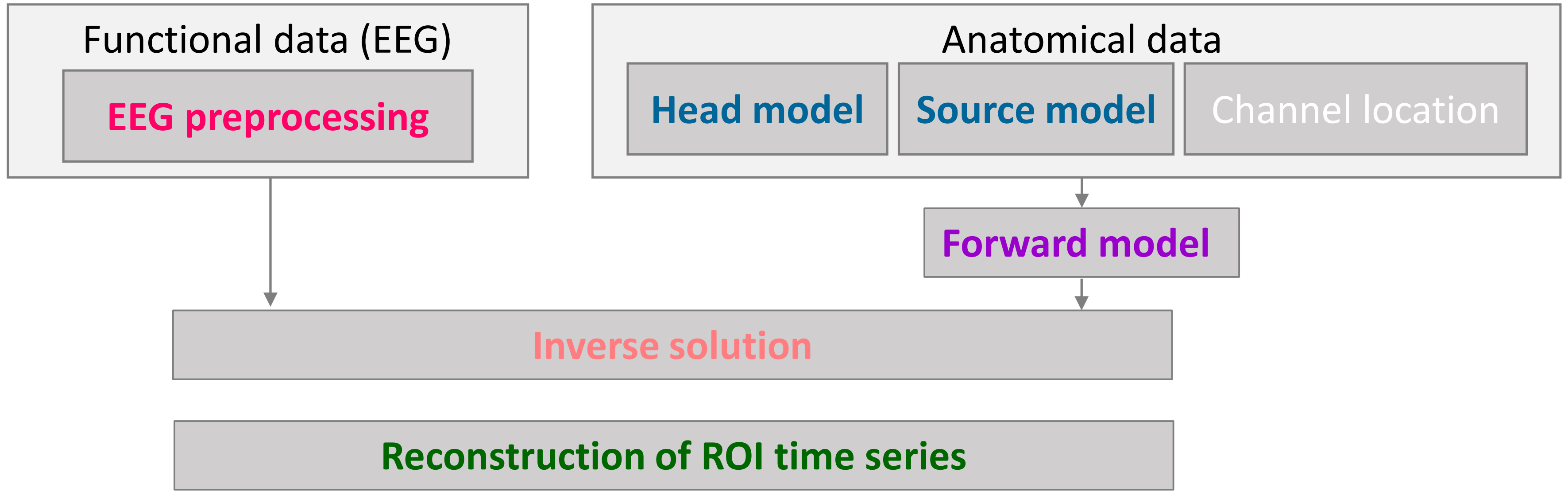
incf

FNSNF
FONDS NATIONAL SUISSE
SCHWEIZERISCHER NATIONALFONDS
FONDO NAZIONALE SVIZZERO
SWISS NATIONAL SCIENCE FOUNDATION

Outline

EEG preprocessing
MRI preprocessing
Inverse solution
ROI time courses

Tutorial 3 – from scalp to sources



EEG preprocessing
Data import, data filtering, bad channel identification, artefact removal and rejection

MRI preprocessing
MRI segmentation for head model computation and grid definition for source model computation

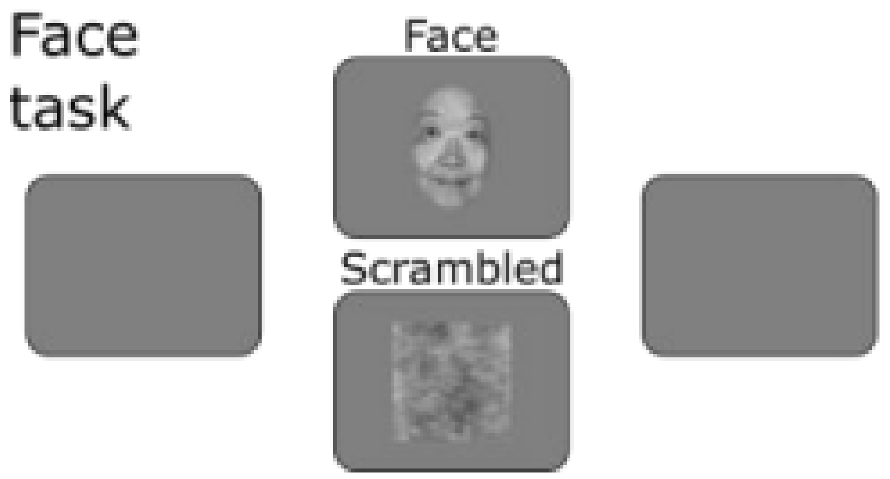
Solution of the inverse problem
Leadfield matrix extraction, spatial filter estimation

Reconstruction of ROI time series
Parcellation selection, dimensionality reduction from sources to ROIs

Outline
EEG preprocessing
MRI preprocessing
Inverse solution
ROI time courses

Tutorial 3 – the stimulus

- Visual evoked potentials (VEP)
- FACES = Female and male faces cropped with a Gaussian kernel to smooth borders
- SCRAMBLED = Phase spectra randomization of original images
- 4 blocks of 150 trials each (300 faces, 300 scrambled)



500 ms **200 ms** **1000 ms to respond** **…600-900 ms before the new trial…**

Pascucci et al 2021, BioRxiv

EEG – first steps/ data import and filtering

<u>Import the data</u>

- BDF data are the voltages between each electrode and CMS (50 Hz, ADC noise)
  $\rightarrow$ rereference the data to A1 (Cz) to remove CMS contribution

`ft_preprocessing(cfg)`

<u>Downsample</u>

- To 250 Hz, reduce computational load

`ft_resampledata(cfg,data)`

<u>High-pass filter</u>

- Above 1 Hz; eye-blinks are identified later by ICA

`ft_preprocessing(cfg)`

<u>Segment the data</u>

- Read event definition from sub-01_task-faces_events.tsv (onset + value)
- Identify trials corresponding to FACES (value=1) and SCRAMBLE (value=0)
  $\rightarrow$ Define epoch length ([prestim poststim])
- Actually segment continuous data

`ft_read_event(filename)`

`ft_trialfun_gen(cfg)`

`ft_redefinetrial(cfg,data)`

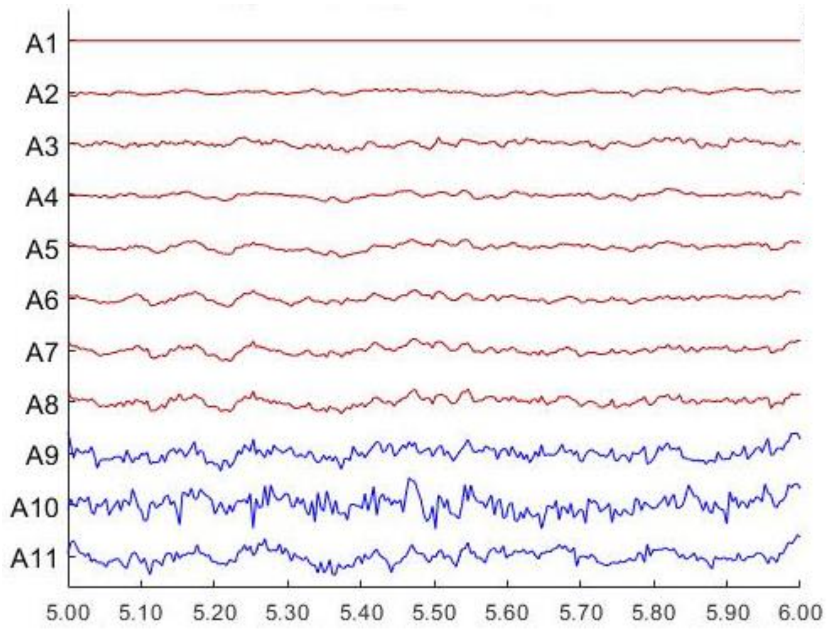
EEG – bad channels

What they are

- Channels with low SNR in general (due for example to high impedance…)

How to identify them

- Channels with low correlation with their neighbours
- Channels with SD very different from that of the others
- Visually



Read bad channels

- Channels are labelled as ‘bad’ in sub-01_task-faces_channels.tsv

Only keep the good channels

```
ft_selectdata(cfg,data)
```

EEG – ICA for artefact removal

What is it

- Separates source signals from mixed signals (cocktail party problem)
- Several algorithm exists: FastICA, Infomax (used here)
- Needs to be done after bad channels removal!

```
cfg = [];
cfg.method = 'runica';
cfg.channel = {'all' '-A1'};
cfg.updatesens = 'no';
ICs = ft_componentanalysis(cfg, data);
```

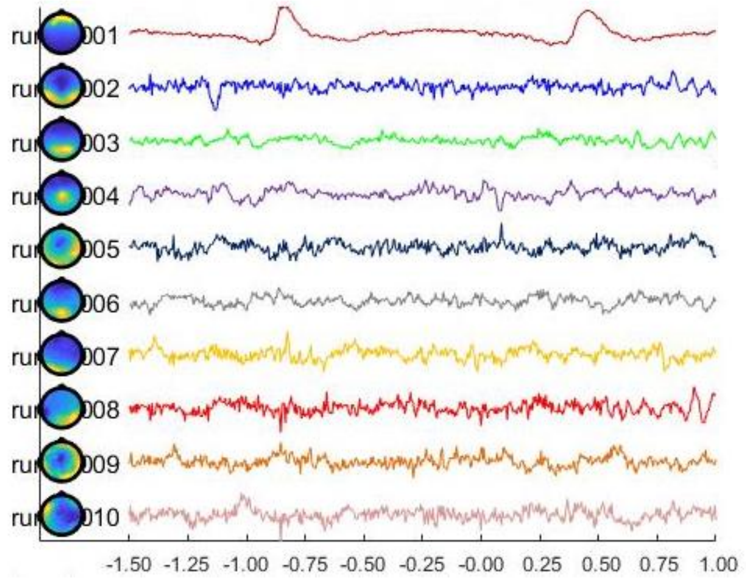
What it yields

- Components that are non-Gaussian and statistically independent from each other

Select the ICs that capture artefacts

- Eye-blinks
- Muscle twitches

```
cfg = [];
cfg.layout = layout;
cfg.viewmode = 'component';
cfg.continuous='no';
cfg.component = 1:10;
cfg.allowoverlap='yes';
ft_databrowser(cfg, ICs)
```



```
ft_rejectcomponent(cfg, ICs, data)
```

Finally remove the components

- Backprojects the components and substracts them from the data

Outline

EEG preprocessing

MRI preprocessing

Inverse solution

ROI time courses

EEG – final steps

<u>Define neighbouring electrodes</u>

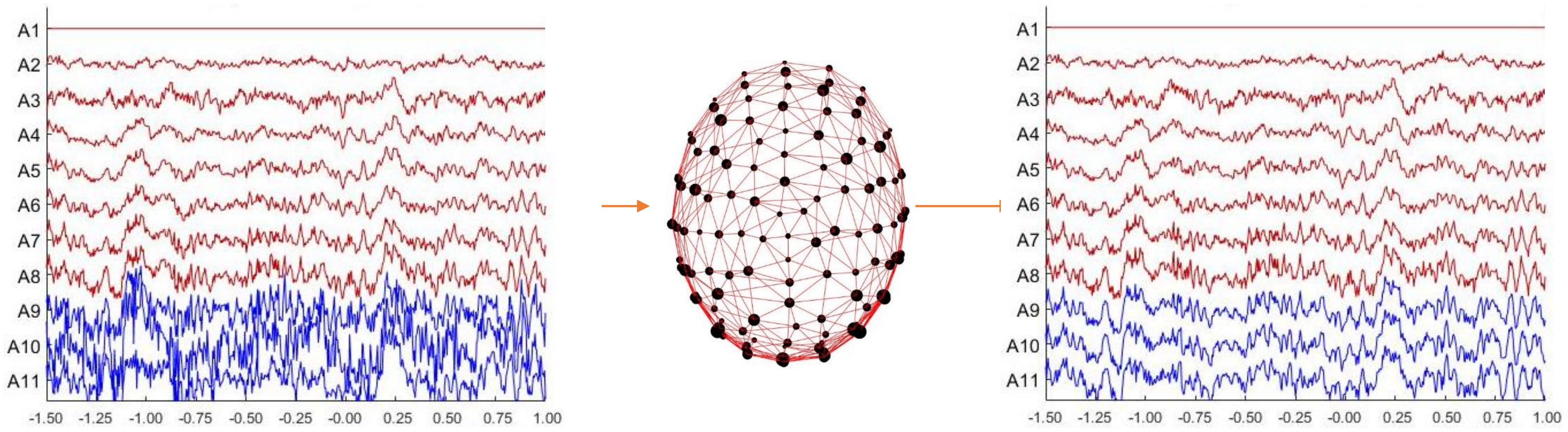
```
ft_prepare_neighbours(cfg,data)
```

- Needed for the bad channel interpolation

```
ft_channelrepair(cfg,data)
```

<u>Interpolate bad channels</u>

- Replace the bad channels with the spherical interpolation of the neighbours



<u>Average re-reference</u>

```
ft_preprocessing(cfg)
```

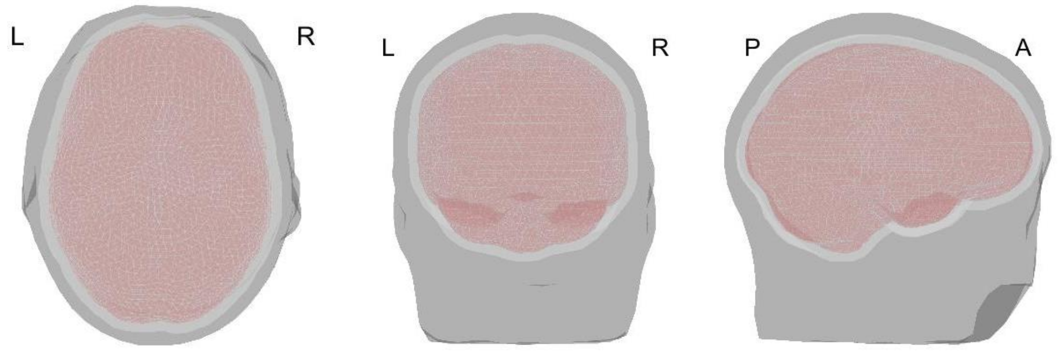
- Needed for the source localization

Outline
EEG preprocessing
MRI preprocessing
Inverse solution
ROI time courses

MRI - head model

What is it

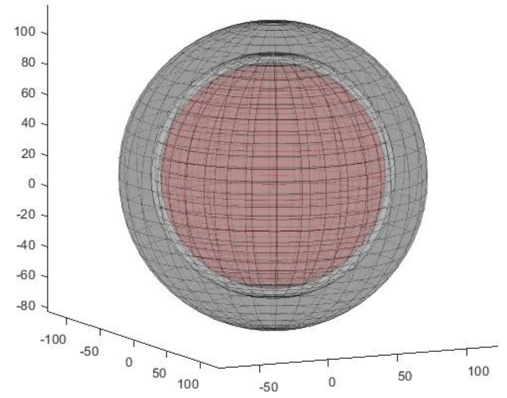
- Model that describes the electrical properties of the head
- Necessary to solve the forward model



How to estimate it

- MRI import `ft_read_mri(mri_filename)`
- MRI segmentation to extract surfaces of skin, skull and brain `ft_volumesegment(cfg, mri)`
- Create these surfaces using a triangular mesh `ft_prepare_mesh(cfg,segmentedmri)`
- 3 concentric spheres: assign a conductivity to each layer/surface

`ft_prepare_headmodel(cfg, mesh)`

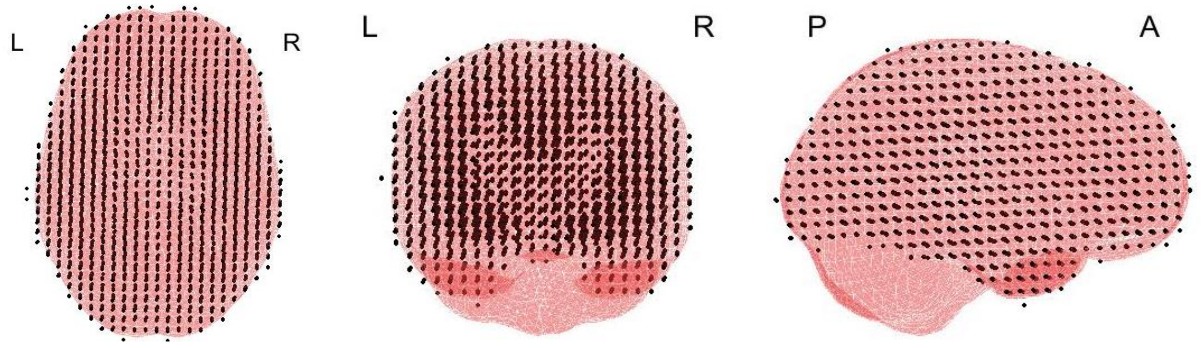


Outline
EEG preprocessing
MRI preprocessing
Inverse solution
ROI time courses

MRI - source model

What is it

- Model that describes the position of the current dipoles
- Dipoles are placed in the grey matter (not in the basal ganglia)
- Dipoles have not a fix orientation
  (1 point source=3 dipoles, one for each axis)
- Necessary to solve the forward model



How to estimate it

- Import atlas.nii and GM.nii (cmp output)
- Remove basal ganglia ROI from GM
- Coregister (linearly) with template to have a correct grid orientation
- Create the source model using a 6 mm-resolution grid of unfixed dipoles

```
ft_read_mri(mri_filename)
```

```
mask=ismember(atlas.anatomy, ROI2remove);
GM.anatomy(mask)=0;
```

```
cfg=[];
cfg.resolution=6;
cfg.unit='mm';
cfg.tight='yes';
cfg.mri=realign_reslice_GM;
ft_prepare_sourcemodel(cfg)
```

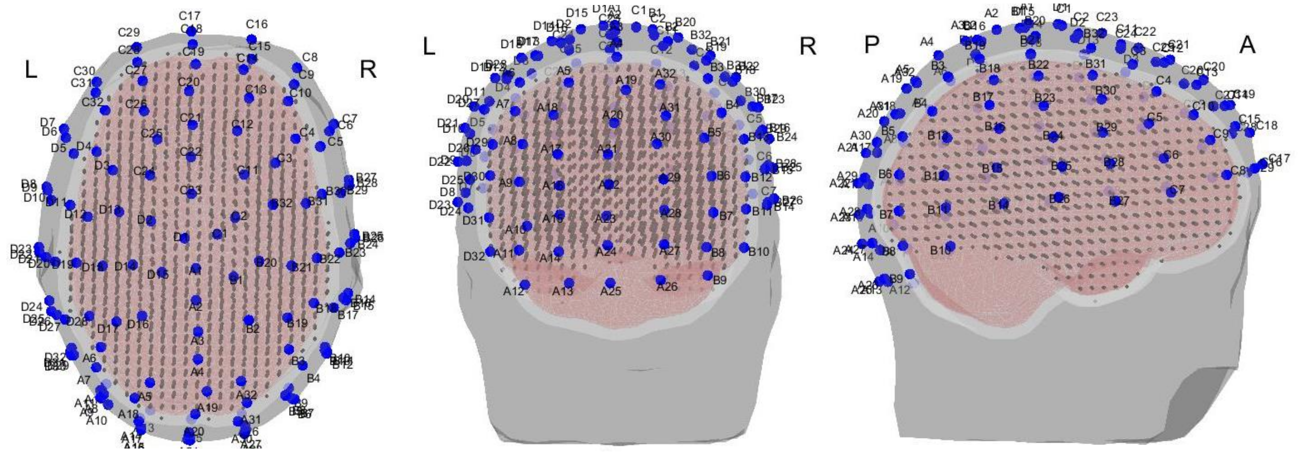
Solution of the inverse problem– leadfield matrix extraction

What is it

- Matrix (A) that describes the contribution of the source activity $s(t)$ to each scalp electrode, so that the distributed source model describing the EEG scalp signals $x(t)$ holds true:

$$x(t) = A * s(t) + n(t)$$

- For each source, the leadfield matrix (X*3) represents the field distribution (the potential) across all X sensors due to a unitary current dipole in its x,y,z-orientations
- Requires (head model) + (source model) + (sensors location)



How to estimate it

```
cfg = [];
cfg.elec = data.elec;
cfg.sourcemodel = sourcemodel;
cfg.headmodel = headmodel;
Leadfield = ft_prepare_leadfield(cfg);
```

Solution of the inverse problem – spatial filters estimation

What is it

- The spatial filers W are spatial maps –one per source dipole component and per source position- so that the following holds true

$$s(t) = W * x(t)$$

- Obtained with 'imaging' approach (i.e. by inversion of the distributed source model: MNE, wMNE) or with a 'scanning' approach (beamformer)

- The solution is ill-posed (S>>X)

How to estimate it

- Exact low-resolution electromagnetic tomography

```
spatial_filters = ft_inverse_eloreta(leadfield, elec,
hdmodel, dat, eye(length(data_reref.label)), 'keepfilter',
'yes','lambda', 0.05);
```

- Estimate the spatial filters W

ROI time courses

What is it

- For each ROI, 'summarise' the 3d-time series of the sources belonging to that ROI in a single time course
- Need (atlas from cmp) + (spatial filters) + (clean EEG data)

How to estimate it

- Import atlas.nii and set the basal ganglia ROI to zero

```
ft_read_mri(mri_filename)
```

```
mask=ismember(atlas.anatomy,ROI2remove);
atlas.anatomy(mask) = 0;
```

- Assign ROI label to each source point (for each source point, find the 26 closest voxels around each it and assign the label of the most present tissue around the source point)

```
assigned_label2source_points(sourcemodel,
nifti_atlas, tbl_idx2label)
```

- Concatenate time courses

```
time_courses = cat(2, data.trial{:});
```

- For each ROI, use Singular Value Decomposition (SVD) to go from many sources → to one time-course
  - $X = U \times S \times V^t \Leftrightarrow X \times V = U \times S$
  - Concatenate the spatial filters of the given ROI
  - Apply the inverse transformation
  - Get the first left singular vector

```
for ROI_id = 1:n_ROIs
    curr_ROI_bln = strcmp(src_label, ROIs_lbl{ROI_id});
    curr_filters = filters(curr_ROI_bln);
    curr_filters = cat(1,curr_filters{:});

    ESI_tmp = time_courses.'*curr_filters.';
    [ROI_traces(ROI_id,:),S,V] = svds(ESI_tmp,1);
end
```

- Segment the ROI time courses in the original epochs