Background

You can access builtin documentation directly in Python. This is useful for quick reference. However, the website has many advantages like a better layout and links everywhere (https://mne.tools/).

The following sources may be particularly relevant for this assignment:

- Tutorials:
  - Preprocessing: Repairing artifacts with ICA
  - Working with continuous data: Working with events
  - Segmenting continuous data into epochs
  - Estimating evoked responses
- When you need to learn more about how a specific function works (e.g. it's used in a tutorial but not explained sufficiently) start in the API documentation

Note:

- Most MNE visualization functions return matplotlib figures. If you want to modify those figures you can use the matplotlib API. For example, you can access the axes on the figure through `Figure.axes`, and then you can use the matplotlib `Axes` API to add new elements.

General instructions

- Name your notebook `firstname-lastname-A1.ipynb`; e.g., `christian-brodbeck-A1.ipynb`
- Document your code: explain what it is doing
- Use the qestion identifiers in your answer ("1.1", "A" etc.)
- Before submitting, restart the kernel and re-run the whole notebook. This is to make sure your code runs in the sequence you are handing in. We want to see cell execution order numbers 1, 2, ...
- Submit an `ipynb` and a `PDF` version of your final notebook
- This is meant as an individual assignment and what you hand in should be your work
- Please do not use AI; you should be able to justify/explain what you write; if you have questions, please ask us

Setup

You can use your preferred method for creating the required environment. However, if you are starting new or facing problems, we recommend installing Mamba.

Start with an environment containing the basic libraries:

```
(base) $ mamba create -n 4cn3-a1 mne jupyterlab
```

Activate the new environment and start the notebook server:

```
(base) $ mamba activate 4cn3-a1
(4cn3-a1) $ jupyter lab
```

In addition to `mne` and `numpy`, you are allowed to import `matplotlib`, and the builtin `os` and `pathlib`. Do not import any other library.

In [1]:
```
import mne
import numpy


# Make sure you are using MNE-Python version 1.9
print(f"{mne.__version__=}")
```

```
mne.__version__='1.9.0'
```

Let MNE-Python download the sample data we will work with, and get the path:

In [2]:
```
data_path = mne.datasets.sample.data_path()
print(data_path)
```

```
/Users/christian/mne_data/MNE-sample-data
```

1. Evoked responses

1.1 Load/inspect data

Load the raw data file from the MNE sample data. The full path to the file is:

`ROOT/MEG/sample/sample_audvis_raw.fif`

1.2 Preprocessing

Filter the data with a band-pass filter, retaining data between 1 and 40 Hz.

Use ICA to remove the heart beat component from the data. For speed, use the FastICA algorithm. Make sure that your code is reproducible (i.e., the outcome will be the same every time when the code is run multiple times).

1.3 Events

Inspect the events in the dataset.

A, What types of events are there? How many events are there of each type? Plot the events on a timeline.

B. Investigate how the events are distributed in time: Are events of the same type clustered in time? Is the sequence of events random or predictable? Could this distributed in time affect brain responses?

1.4 Analyze evoked responses

Epoch the preprocessed data into data segments starting 200 ms before each event, and lasting until 700 ms after the event.

Initially, pick only the magnetometer data (`picks='mag'`). Pick left ear auditory stimluation. Visualize the time course of the evoked response using `Evoked.plot()` .

Assume that the brain response reflects a sequence of stages, each stage reflected by a specific configuration of current sources. We can visualize a snapshot of the response time course using a topographic plot. `Evoked.plot_joint()` will try to automatically find relevant time points (reflecting meaningful stages) in the response based on peaks in the RMS across sensors (also called the "Global Field Power").

A. Focus on the brain response within 150 ms of the stimulus. Are the peaks resulting from the automatic RMS selection informative?

Hint: Do all sensors exhibit the same general shape?

B. Still focusing on the brain response within 150 ms of the stimulus: Can you think of a better approach than the RMS to select informative peaks? Can you find and visualize a more informative sequence of peaks? (it's ok to select peaks manually with an articulated strategy)

From this, what can you find out about the sequence of events after left ear stimulation?

C. Do the same for right ear stimulation, and compare the two response profiles.

D. Do the same for visual stimulation. Compare visual to auditory responses. In what respects do they differ? In what respects are they similar?

E. Now do the previous analyses, but for EEG data. Would this lead to the same conclusions as the MEG analysis?