0. Setup

Initialization:
- FieldTrip top-level folder for ft_defaults.m
- MEGneto top level for megne2setup.m
- SPM T1 template visible

MATLAB input
- project_path
- analysis_name
- rawdata_path
- mri_path
- overwrite

megne2setup.m
- Run ft_defaults.m to add the right subfolders within FieldTrip.
- Create directories for project
- Initialize JSON config file

paths (struct)

out

1,2,3. Processing

fcp_1_TaskEpoching.m

1. **Setup**
   - Read in config settings
   - Load and match PIDs from MEG and MRI folders
   - Create list of subjects with full sets of data
2. **Plot triggers**
   - Generate and save plot of event triggers
3. **Epoch continuous data into trials**
   - Set up trial definitions for epoching
   - Epoch events list to define time windows corresponding to each trial and overall number of trials; save to config
4. **Head motion**
   - Identify trials with excessive head motion
   - Reject those trials, save filtered version
5. **Artifacts**
   - Check for muscle, jump artifacts
   - Reject those trials with artifacts, save filtered version
6. **Bad channels**
   - Identify and save list of bad channels
   - Throw warning if there are many

   Outputs
   - Plots:
     - Triggers for each event type
     - Head motion visualization
   - Epochs (into fcp_1_output JSON):
     - All trials
     - Filtered for head motion (HM)
     - Filtered for HM, muscle, jump artifacts
   - List of bad channels (saved to its own JSON)

fcp_2_PreprocessingICA.m

1. **Setup**
   - As in fcp_1
2. **Noise reduction**
   - Load gradiometer info and compute 3rd order gradients from CTF to account for noise
3. **ICA**
   - Remove bad channel signals altogether from analysis (they are dependent on neighbor sensor signals, thus redundant to include in ICA)
   - Run ICA

   Outputs
   - ICA components for each participant

fcp_2_5_checkpoint.m

- Human to identify ICA components that are artifacts or too noisy.
- Back-project to clean ICA

Outputs
- Bad ICA components to JSON file
- ICA-denoised data

fcp_3_ChannelRepair.m

1. **Setup**
   - As in fcp_1
2. **Bad channel repair**
   - Replace bad channel signal with average (or other option) of neighbours
   ----
   Outputs
   - Final preprocessed data output into *.mat file
   - fcp_2_output JSON tracking configurations, paths
   - Bad ICA components to JSON file

---

4. Beamforming

fcp_4_beamforming.m

1. **Setup**
   - Read in config settings
   - Load and match PIDs from MEG and MRI folders
   - Create list of subjects with full sets of data; note that participants may be removed after preprocessing due to insufficient number of trials leftover (e.g., too much noise)
2. **Head model preparation**
   - Load and segment T1 template brain
   - Construct head model and do necessary unit conversions
   - Construct dipole grid in template brain
3. **Check alignment**
   - Load and segment participant MRIs, load preprocessed MEG data
   - Construct subject-specific head, source models
   - Check alignment between subject and template head model
   - Check alignment source model and head model
   - Save images
4. **Source reconstruction**
   - Compute lead field matrix
   - Run source analysis, reduce to dominant orientation
   - Load desired atlas and create binary masks to define valid voxels within head model
   - Interpolate functional data onto anatomical data
   - Further interpolation onto AAL-defined regions
   - Return these results

   Outputs (for each participant)
   - catmatrix = individual source timeseries in *.mat files: (timepoints in 1 trial) x (num. trials) x (num. AAL nodes)
   - srate = sampling rate
   - coords = coordinate of source points w/in
   - Individual and template head model alignment, *.png
   - Source and head model alignment, *.png

---

5. Frequency and Connectivity Analysis

fcp_5_freqanalysis.m

1. **Setup**
   - As in fcp_1 and load in virtual sensor data from fcp_4
2. **Power spectrum**
   - Perform time window analysis
   - Baseline correct the data to control for general/random spikes in power
   ----
   Outputs
   - 4-D matrix containing power spectrum data. Matrix dimensions are: [participants] x [regions] x [frequency] x [time]
   - Users can plot frequency by time for a specific region of a given participant's data by applying a plotting function on a slice of the matrix

fcp_5_taskconnectivity.m

1. **Setup**
   - As in fcp_1
   - Create filter coefficients
2. **Analysis**
   - Load participant source timeseries matrices (catmatrix)
   - Apply Hilbert filter to isolate frequency band
   - Return PLI or PLV connectivity between pairs of AAL nodes
   - Repeat for each trial and frequency band
   - Save individual adjacency matrix
   - Then, repeat for all participants
   - Take average across trials for each participant and assemble all-participant adjacency matrix
   ----
   Outputs
   - Individual adjacency matrix (nodes x nodes x trial x freq. band)
   - All participant adjacency matrix (nodes x nodes x participants x band)

- ft_defaults
- path_check
- path_generation
- save_to_json

- check_csv_has_empty
- detectBadChannels
- ds_pid_match
- ft_artifact_jump
- ft_artifact_muscle
- ft_definetrial
- ft_preprocessing
- ft_rejectartifact
- ft_read_event
- ft_read_header

- HeadMotionTool
- load_config
- load_participants
- plotTriggers
- save_to_json
- write_match_if_not_empty

- ds_pid_match
- ft_channelrepair
- ft_channelselection
- ft_componentanalysis
- ft_denoise_synthetic
- ft_prepare_neighbours
- ft_preprocessing
- ft_rejectcomponent
- ft_resampledata
- ft_selectdata
- ft_topoplotIC

- inputBADchannels
- load_config
- load_participants
- saveSensorsToFile
- save_to_json
- write_match_if_not_empty

- ds_pid_match
- ft_convert_units
- ft_freqanalysis
- ft_plot_mesh
- ft_plot_sens
- ft_plot_vol
- ft_prepare_headmodel
- ft_prepare_leadfield
- ft_prepare_sourcemodel
- ft_read_atlas
- ft_read_mri
- ft_resampledata

- ft_sourceanalysis
- ft_sourcedescriptives
- ft_sourceinterpolate
- ft_sourceplot
- ft_timelockanalysis
- ft_volumelookup
- ft_volumesegment
- load_config
- load_participants
- write_match_if_not_empty

- ds_pid_match
- ft_connectivityanalysis
- load_config
- load_participants
- write_match_if_not_empty

2020-12-02