

# FrankLab

FL fMRI data organization and scripts

<b>Installation</b>	<b>3</b>
<b>General FrankLab data organization</b>	<b>4</b>
<b>Getting started</b>	<b>6</b>
Defining .cfg files for a new study or a new subject	6
Overview of FL commands	8
<b>Importing data for a new subject (fl IMPORT)</b>	<b>9</b>
General information	9
Configuration file defining subject information	10
Example (EXAMPLE01.cfg)	10
Definition details	11
Configuration file defining experimental design information	12
Example (design_A.cfg)	12
Definition details	14
<b>Preprocessing your data (fl PREPROCESSING)</b>	<b>15</b>
General information	15
Configuration file defining preprocessing pipeline information	16
Example (preprocessing_mnispace.cfg)	16
Example (preprocessing_surfacespace.cfg)	17
Definition details	18

<b>Running first-level analyses (fl FIRSTLEVEL)</b>	<b>21</b>
General information	21
Configuration file defining first-level model estimation options	22
Example (firstlevel_speech.cfg)	22
Definition details	23
<b>Running second-level analyses (fl SECONDLEVEL)</b>	<b>25</b>
General information	25
Configuration file defining second-level analysis	26
Example (secondlevel_average.cfg)	26
Example (secondlevel_correlation.cfg)	27
Example (secondlevel_anova.cfg)	28
Definition details	29
Storing subject-level information	30
<b>Running second-level ROI-based analyses (fl SECONDLEVEL.ROI)</b>	<b>31</b>
General information	31
Configuration file defining ROI	32
Example (roi_atlas.cfg)	32
Example (roi_parcel.cfg; using subject-specific ROI files)	32
Definition details	33
<b>Optional/advanced commands</b>	<b>34</b>
<b>Example dataset directory structure</b>	<b>37</b>

# Installation

Copy FL software to installation folder ( e.g. /project/busplab/software/fl )

note: FL requires **SPM12**, **CONN**, and **Evlab17** packages already installed in your system. If they are not, copy them to their corresponding installation folders (e.g. /project/busplab/software/spm12, /project/busplab/software/conn, and /project/busplab/sotware/evlab17, respectively)

In Matlab type “addpath /project/busplab/software/fl” (without the quotes) to add FL to your Matlab path, and optionally type “savepath” to save these changes for future Matlab sessions as well

Optionally, delete the REPOSITORY entry in FL installation folder (e.g. /project/busplab/software/fl/REPOSITORY) and add in its place a soft-link to your desired location of the FL data repository (e.g. /project/busplab/software/fl/REPOSITORY -> /projectnb/busplab/FL ) where all of the experiment and subject data will be stored

# General FrankLab data organization

FrankLab organizes each experiment functional/anatomical data in a BIDS-compatible structure. FL software integrates all of the functionality necessary to: a) import and preprocess your functional and anatomical volumes; b) create Quality-Control plots and reports; c) perform first-level analyses estimating task-/condition- responses; and d) perform second-level analyses estimating population-level effects

An example of this BIDS-compatible folder structure for a single experiment would be the following:

Portion of data structure created by user (inputs to FL)

<b>EXAMPLE /</b>	<b>(experiment folder)</b>
code /	(optional directory for user-defined scripts)
config /	<b>(directory containing user-defined configuration files)</b>
<b>EXAMPLE01.cfg</b>	(lists subject 'EXAMPLE01' files -functional/anatomical/design/etc.)
<b>EXAMPLE02.cfg</b>	
<b>EXAMPLE03.cfg</b>	
design_A.cfg	(details experimental design onsets/durations/names)
design_B.cfg	
preprocessing_mnispace.cfg	(lists 'mnispace' preprocessing pipeline steps and options)
firstlevel_speech.cfg	(lists 'speech' firstlevel analysis options)
secondlevel_average.cfg	(lists 'average' secondlevel analysis options)
sub- <b>EXAMPLE01 /</b>	<b>(directory containing individual subject raw data)</b>
scanner /	(directory containing dicom -or other original files- from scanner)
anat / sub- <b>EXAMPLE01_T1w.nii</b>	
func / sub- <b>EXAMPLE01_run01_bold.nii</b>	
sub- <b>EXAMPLE01_run01_bold.json</b>	
[sub- <b>EXAMPLE01_run01_events.tsv</b> ]	
fmap / sub- <b>EXAMPLE01_run01_fmap.nii</b>	
fmap / sub- <b>EXAMPLE01_run01_vdm.nii</b>	
sub- <b>EXAMPLE02 /</b>	
sub- <b>EXAMPLE03 /</b>	
derivatives /	<b>(directory containing all derivative data)</b>
<b>mnispace /</b>	(individual preprocessing pipeline)
sub- <b>EXAMPLE01 /</b>	(individual subject)
anat	

```

func
surf
fmap
results/ firstlevel / (directory containing firstlevel analyses)
    speech (individual firstlevel analysis)
    qa/ (directory containing QA plots)
sub-EXAMPLE02 /
sub-EXAMPLE03 /
results / secondlevel /
    speech / average (individual secondlevel analysis)

```

Portion of data structure created by FL scripts

where the section in blue represents files/folders that are created by users as input to FL software, and the section in black represents files generated by different FL steps.

In this example there is a single study (StudyID = EXAMPLE) and three subjects (e.g. SubjectID = EXAMPLE01, EXAMPLE02, and EXAMPLE03), a single preprocessing pipeline (PipelineID = **mnispace**) and a single first-level model (ModelID = **speech**). The portion of the filenames and directories that are dependent on these Study/Subject/Pipeline/Model IDs are bolded to highlight what would be different for a different Study/Subject/Pipeline/Model.

# Getting started

## Defining .cfg files for a new study or a new subject

For a new study:

- A. Define your **study id** (just a keyword identifying your new study, e.g. ACE) and create a new experiment folder named `$/ACE` (note: `$/projectnb/busplab/FL/`; see REPOSITORY in installation section above)
- B. Create your general **experimental design files** `$/ACE/config/design_*.cfg`. This is where you specify the experimental paradigm during the scanner session, such as task/condition onsets, durations, modulators, etc. (see [EXAMPLE/config/design\\*.cfg](#) for a template)
- C. Create one or several **preprocessing pipeline files** `$/ACE/config/preprocessing_<pipeline>.cfg`. This is where you define your planned sequences of functional and structural preprocessing steps (see [software/fl/preprocessing\\_mnispace.cfg](#) and [software/fl/preprocessing\\_surfacespace.cfg](#) for default preprocessing pipelines for **volume-level** and **surface-level** analyses, respectively; you may directly copy these files or adapt them if needed)
- D. Create one or several **first-level model files** `$/ACE/config/firstlevel_<model>.cfg`. This is where you define contrasts of interest and subject-level model estimation options (see [EXAMPLE/config/firstlevel\\_speech.cfg](#) for a template)
- E. Create one or several **second-level model files** `$/ACE/config/firstlevel_<model>.cfg`. This is where you define group-level analysis details (see [EXAMPLE/config/secondlevel\\_average.cfg](#) for a template). Optionally, create one additional file `$/ACE/config/participants.tsv` containing all behavioral/demographic information for each subject

For a new subject:

- F. Define your subject id (just a keyword identifying a new subject as part of an existing study; e.g. ACE01), create a new subject folder `$/ACE/sub-ACE01/scanner`, and copy there all of the data from your scanner (e.g. dicoms or nifti files) (please note add 'sub-' prefix in folder name above to the subject ID)
- G. Optionally, when importing DICOM files, use the “fl convert.dicom ACE01” syntax to convert all dicom files to nifti format. This will also attempt to automatically identify the relevant functional/anatomical/dti dicom series (copying them to corresponding anat, func, dti folders)
- H. Create a new **data configuration file** `$/ACE/config/ACE01.cfg` with the same name as your subject ID. This file (see [EXAMPLE/config/EXAMPLE01.cfg](#) for a template) specifies the location/names of the dicom/nifti files copied/created in the steps above, and the location/names of the design\_\*.cfg files for this subject's functional runs  
note1: optionally (rarely), you may also create a file `$/ACE/config/preprocessing_ACE01_<pipelineID>*.cfg` if you need to define subject-specific preprocessing options, and/or a file `$/ACE/config/firstlevel_ACE01_<ModelID>*.cfg` if you need to define subject-specific contrasts or model-estimation options  
note2: if your structural data has been preprocessed using freesurfer, you may enter the freesurfer-generated mri/T1.mgz file in the #structurals field. This will also import the associated freesurfer information (surf/lh|rh.white, surf/lh|rh.pial, and surf/lh|rh.sphere.reg subject-coregistration information) during the fl import step, enabling surface-based analyses

All configuration (.cfg) files share the same format, consisting of simple text files containing *FieldNames* (field names have a # prefix, e.g. #functionals) followed by one or several *FieldValues*. *FieldValues* may be numeric (values/vectors/matrices entered as space separated rows of values) or strings (with lists of strings are entered as one string per line). Comments start with the % character and are simply disregarded by FL.

## Overview of FL commands

After creating all necessary configuration files, you may simply run any of the following ***fl*** commands (described in detail in the next sections) in order to:

- |                                   |   |
|-----------------------------------|---|
| 1) import each subject's data     | <b><i>fl IMPORT</i></b> <i>[SubjectID]</i>  |
| 2) preprocess each subject's data | <b><i>fl PREPROCESSING</i></b> <i>[SubjectID]</i> <i>[PipelineID]</i>                                 |
| 3) compute first-level analyses   | <b><i>fl FIRSTLEVEL</i></b> <i>[SubjectID]</i> <i>[PipelineID]</i> <i>[ModelID]</i>                   |
| 4) compute second-level analyses  | <b><i>fl SECONDLEVEL</i></b> <i>[StudyID]</i> <i>[PipelineID]</i> <i>[ModelID]</i> <i>[ResultsID]</i> |



# Importing data for a new subject (*fl IMPORT*)

## General information

Example syntax:

```
fl IMPORT EXAMPLE01
```

Imports new subject EXAMPLE01 into FL folders

- Expects `$/EXAMPLE/config/EXAMPLE01.cfg` file listing source dicom/nifti files and experimental design information
- Creates original version of dataset in `$/EXAMPLE/sub-EXAMPLE01`

Note1: the file **EXAMPLE01.cfg** should minimally include the source of structural and functional volumes for each subject (`#structurals` and `#functionals` fields). It may also include experimental design information (`#design` field), although this may alternatively be defined in separate model-specific files (see **firstlevel\_EXAMPLE01\_speech.cfg** references in the sections below). Last (and optionally), it may also include the source of fieldmap volumes for each subject (`#fmap_functionals` field) or already-computed voxel-displacement maps (`#unwarp_functionals` field)

General command syntax:

```
fl IMPORT [SubjectID] ...
```

**[SubjectID]** String identifying individual experiment/subject/session, and associated with an existing **[SubjectID].cfg** file defining this subject source dicom/nifti files and experimental design. The SubjectID string is composed by an **[StudyID]** part (non-numeric characters only, e.g. EXAMPLE, ACE, CCRS, etc.), and a subject identifier (which must start with a number but may contain other characters, e.g. 01, 01b, 01-session1, etc.). In the example above the SubjectID is “EXAMPLE01”, where the “EXAMPLE” token identifies the experiment ID, and the “01” token uniquely identifies one subject within this experiment (note that, following BIDS convention, several folders will be named “sub-[SubjectID]”; keep in mind that the “sub-” portion of these folder names is *not* part of the subject ID).

Optional additional inputs: `fieldName` and `fieldValue` pairs to FL (see *help FL*)

# Configuration file defining subject information

## Example (EXAMPLE01.cfg)

```
#structurals          % structural files (full path or partial within scanner folder)
anat/*.nii

#functionals          % functional files (full path or partial within scanner folder)
func/*.nii

#unwarp_functionals   % vdm files (full path or partial within scanner folder)
fmap/vdm5_run-26.nii

#RT                  % Repetition Time (in seconds)
1.8

#design.runs          % functional runs to include in design
1 2                  % (relative numbers, use #design.dicom_runs for dicom run numbers instead)

#design.files          % files containing onset/duration info (one per run)
design_runA.cfg
design_runB.cfg
```

## Definition details

**#functionals** : list of functional files, if appropriate

enter only one file per run/session

if starting from NIFTI files, enter here the functional files (e.g. /mydata/nii/984000-7.nii)

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

if starting from DICOM files (see 'dicoms' field below), enter here the runs that contain functional data

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

files may be specified by DICOM series numbers (e.g. entering 7 identifies files named \*-7.nii output by the dicom converter step)

files may be specified by filenames without extension (e.g. entering 984000-7 identifies the nifti files generated from 984000-7-\*.dcm)

enter simply \* to indicate all DICOM series

enter simply ? (or do not include the 'functionals' field) to let users select functional files interactively (GUI)

**#structurals** : list of structural files, if appropriate

enter either a single file (for session-invariant structurals) or one file per run (for session-specific structurals)

if starting from NIFTI files, enter here the functional files (e.g. /mydata/nii/984000-3.nii)

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

Note: to enable surface-based analyses, enter here the FreeSurfer-generated mri/T1.mgz file in FS output folder

if starting from DICOM files (see 'dicoms' field below), enter here the filenames/runs that contain structural data

files may be specified by DICOM series numbers (e.g. entering 3 identifies files named \*-3.nii output by the dicom converter step)

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

files may be specified by filenames without extension (e.g. 984000-3)

enter simply \* to indicate all DICOM series

enter simply ? to let user select functional files interactively (GUI)

**#dicoms** : list of dicom files, if appropriate

enter one file per run (only first file -1.dcm from each dicom series)

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

files may be specified explicitly (e.g. /mydata/dicoms/984000-1-1.dcm)

files may also include wildcards (e.g. /mydata/dicoms/\*-1.dcm)

**#RT** : functional data repetition time (single value in seconds)

**#design.runs** : list of runs/sessions to include in the model (default: all sessions identified as functional data during preprocessing step)

alternatively, use fieldname '#design.abs\_runs' or '#design.dicom\_runs' to specify absolute dicom run/session numbers

use fieldname '#design.rel\_runs' or '#design.runs' to specify relative run/session numbers (among those identified as functional data)

**#design.files** : list of design information configuration files (one file per run)

(optional: if planning to use the susceptibility-distortion correction step during realignment, and have already created voxel-displacement maps)

**#unwarp\_functionals**: list if voxel-displacement maps, if already created and appropriate (for preprocessing.steps=='realign&unwarp&fieldmap')

enter a single file or one file per run/session (vdm\* file)

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

note: explicitly entering these volumes here supersedes CONN's default option to search for/use vdm\* files in same directory as functional data)

(optional: if planning to use the susceptibility-distortion correction step during realignment, and have not created yet voxel-displacement maps)

**#fmap\_functionals**: list if fieldmap volumes, if appropriate (for preprocessing.steps==*'functional\_vdm\_create'*)

enter either a) magnitude1+phasediff images; b) real1+imag1+real2+imag2; or c) fieldmap (in Hz) fieldmap acquisition sequence volumes

(note: use #fmap\_functionals\_ss fieldname if entering session/run-specific sets of fieldmap files)

(optional: if planning to use subject-specific ROIs in second-level analyses)

**#rois.files** : list of subject-specific ROI files

if entering partial paths they are interpreted as relative to the sub-<SUBJECTID>/scanner folder

**#rois.names** : (optional) list of ROI names (defaults to name of ROI file, without file extension)

**#rois.labelfiles** : (optional for atlas files) list of text files describing ROI labels within each atlas file

## Configuration file defining experimental design information

Example (design\_A.cfg)

**#units scans**      % units of "onset time" numbers below

**#onsets**            % onset time, condition number

0 3

1 2

2 1

3 2

4 2

5 1

6 2

7 1

8 1

9 2

10 1

11 2

12 2

13 1

14 1

15 2

16 1

17 2

18 2  
19 1  
20 1  
21 2  
22 2  
23 1  
24 2  
25 2  
26 1  
27 2  
28 1  
29 1

#names            % condition names  
Speech Baseline Exclude

#durations        % condition durations  
1 1 1

## Definition details

**#units** : scans / secs : temporal units for onset/duration fields (default: scans)

**#onsets** : Nx2 array with onset times for each event/block in the first column and condition number for each event/block in the second column

**#durations**: Nx1 array with duration of each event/block  
OR 1xM array with duration of each condition

**#names** : 1xM cell array with condition names (note: condition names cannot contain whitespace characters)

**#orth** : (optional, when multiple within-condition effects) 1/0 array indicating whether within-condition regressors should be orthogonalized [1]

(optional: for temporal modulation)

**#tmod** : 1xM 1/0 array indicating whether condition M is modulated by time (if values >1 are entered they are interpreted as polynomial order of temporal modulation effects)

(optional: for parametric modulation by user-defined covariates)

**#pmod** : KxM 1/0 array indicating whether condition M is modulated by covariate K

**#pmod\_names** : 1xK cell array of covariate names (note: covariate names cannot contain whitespace characters)

**#pmod\_values** : NxK matrix of covariate values (e.g. reaction time)

**#pmod\_interaction** : 1xK 1/0 array indicating whether to include condition-by-covariate interactions for each covariate [1]

(optional: for non-parametric modulation; estimation of individual trial-level effects)

**#npmod** : 1xM 1/0 array indicating whether condition M is broken down into individual-trial effects

# Preprocessing your data (*fl PREPROCESSING*)

## General information

Example syntax:

```
fl PREPROCESSING EXAMPLE01 mnispace
```

Preprocess subject EXAMPLE01 using the 'mnispace' preprocessing pipeline

- Expects imported dataset in  $\$/EXAMPLE/sub-EXAMPLE01$  (created in Step 1 above)
- Expects  $\$/EXAMPLE/config/preprocessing\_mnispace.cfg$  file describing *mnispace* preprocessing steps
- Creates  $\$/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01$  directory with preprocessed data

Note1: if a file *preprocessing\_EXAMPLE01\_mnispace.cfg* exists within the config directory the preprocessing information within this file will be used for this subject. This information takes precedence over the general information contained in the *preprocessing\_mnispace.cfg* file. This option may be used in cases where preprocessing steps may be different among different subjects.

Note2: the initial/starting data for all preprocessing pipelines is always the data imported in Step 1 above. This data is copied to the *derivatives/mnispace/sub-\** folders before running preprocessing steps. If you prefer your initial data to be something else (e.g. data already partially preprocessed by a different pipeline), use instead the syntax: "*fl PREPROCESSING.BRANCH EXAMPLE01 otherpipeline mnispace*", changing *otherpipeline* to the name of your initial preprocessing pipeline.

General command syntax:

```
fl PREPROCESSING [SubjectID] [PipelineID] ...
```

**[SubjectID]** String identifying individual experiment/subject/session

**[PipelineID]** String identifying an individual preprocessing pipeline, and associated with an existing *preprocessing\_[PipelineID].cfg* file listing this preprocessing pipeline steps and options. Optionally, associated with an existing *preprocessing\_[SubjectID]\_[PipelineID].cfg* file (for subject-specific preprocessing steps)

Optional additional inputs: fieldName and fieldValue pairs to FL (see *help FL*)

See also: *fl PREPROCESSING.APPEND*, *fl PREPROCESSING.BRANCH*

## Configuration file defining preprocessing pipeline information

Example (preprocessing\_mnispace.cfg)

```
%% EXAMPLE .CFG FILE DEFINING PREPROCESSING PIPELINE
%% see "help fl" for details
%%
```

```
#steps                                % list of preprocessing steps
```

```
functional_label_as_original
```

```
functional_realign&unwarp
```

```
functional_art
```

```
functional_center
```

```
functional_segment&normalize_direct
```

```
functional_label_as_mnispace
```

```
functional_smooth
```

```
functional_smooth
```

```
functional_label_as_smoothed
```

```
#fwhm                                % parameters for 'functional_smooth' steps
```

```
4
```

```
6.9282
```



## Example (preprocessing\_surfacespace.cfg)

```
%% EXAMPLE .CFG FILE DEFINING PREPROCESSING PIPELINE
%% see "help fl" for details
%%
```

```
#steps                                % list of preprocessing steps
```

```
functional_label_as_original
```

```
functional_realign&unwarp&fieldmap
```

```
functional_art
```

```
functional_coregister_affine
```

```
functional_label_as_subjectspace
```

```
functional_surface_resample
```

```
functional_label_as_surfacespace
```

```
functional_surface_smooth
```

```
functional_surface_smooth
```

```
functional_label_as_smoothed
```

```
#diffusionsteps                       % parameters for 'functional_surface_smooth' steps
```

```
10
```

```
30
```

## Definition details

**#steps** : list of preprocessing steps (in order) to be performed. Available steps are:

**functional\_art** : functional identification of outlier scans (from motion displacement and global signal changes)

**functional\_bandpass** : functional band-pass filtering

**functional\_center** : centers functional data to origin (0,0,0) coordinates

**functional\_centertostruct** : centers functional data to approximate structural-volume coordinates

**functional\_coregister\_affine\_reslice** : functional affine coregistration to structural volumes

**functional\_coregister\_affine\_noreslice** : functional affine coregistration to structural volumes without reslicing (applies transformation to source header files)

**functional\_coregister\_nonlinear** : functional non-linear coregistration to structural volumes

**functional\_label** : labels current functional files as part of list of Secondary Datasets

**functional\_manualorient** : applies user-defined affine transformation to functional data

**functional\_manualspatialdef** : applies user-defined spatial deformation to functional data

**functional\_motionmask** : creates functional motion masks (mean BOLD signal spatial derivatives wrt motion parameters)

**functional\_normalize\_direct** : functional direct normalization

**functional\_normalize\_indirect** : functional indirect normalization (coregister to structural; normalize structural; apply same transformation to functionals)

**functional\_normalize\_indirect\_preservemasks** : functional indirect normalization with user-defined Grey/White/CSF masks (coregister to structural; normalize structural; apply same transformation to functionals as well as to Grey/White/CSF masks)

**functional\_realign** : functional realignment

**functional\_realign\_noreslice** : functional realignment without reslicing (applies transformation to source header files)

**functional\_realign&unwarp** : functional realignment & unwarp (motion-by-inhomogeneity interactions)

**functional\_realign&unwarp&fieldmap** : functional realignment & unwarp & inhomogeneity correction (from vdm/phasesmap files)

**functional\_regression** : removal of user-defined temporal components from BOLD timeseries (keeping residuals of linear regression model)

**functional\_remove\_scans** : removes user-defined number of initial scans from functional data

**functional\_segment** : functional segmentation (Gray/White/CSF tissue classes)

**functional\_segment&normalize\_direct** : functional direct unified normalization and segmentation

**functional\_segment&normalize\_indirect** : functional indirect unified normalization and segmentation (coregister to structural; normalize and segment structural; apply same transformation to functionals)

**functional\_slicetime** : functional slice-timing correction

**functional\_smooth** : functional spatial smoothing (spatial convolution with Gaussian kernel)

**functional\_smooth\_masked** : functional spatial masked-smoothing (spatial convolution with Gaussian kernel restricted to voxels within Grey Matter mask)

**functional\_surface\_coreg&resample** : coregister&resample functional data at the location of FreeSurfer subject-specific structural cortical surface

**functional\_surface\_resample** : resample functional data at the location of FreeSurfer subject-specific structural cortical surface

**functional\_surface\_smooth** : functional spatial diffusion of surface data

**functional\_vdm\_create** : creation of vdm (voxel-displacement-map) from fieldmap dataset (reads 'fmap' secondary functional dataset containing magnitude and phase difference images and creates 'vdm' secondary functional dataset containing voxel-displacement map)

**structural\_center** : centers structural data to origin (0,0,0) coordinates

**structural\_manualorient** : applies user-defined affine transformation to structural data

**structural\_manualspatialdef** : applies user-defined spatial deformation to structural data

**structural\_segment&normalize** : structural unified normalization and segmentation

**structural\_normalize** : structural normalization to MNI space (without segmentation)

**structural\_normalize\_preservemasks** : structural normalization to MNI space with user-defined Grey/White/CSF masks (normalizes structural data and applies same transformation to user-defined Grey/White/CSF mask ROIs)

**structural\_segment** : structural segmentation (Grey/White/CSF tissue classes)

(additional preprocessing-step options are required only when including associated preprocessing steps, default values are specified below between brackets)

**#affreg** : (normalization) affine registration before normalization ['mni']

**#art\_thresholds** : (functional\_art) ART thresholds for identifying outlier scans

- art\_thresholds(1): threshold value for global-signal (z-value; default 5)
- art\_thresholds(2): threshold value for subject-motion (mm; default .9)

additional options: art\_thresholds(3): 1/0 global-signal threshold based on scan-to-scan changes in global-BOLD measure (default 1)

- art\_thresholds(4): 1/0 subject-motion threshold based on scan-to-scan changes in subject-motion measure (default 1)
- art\_thresholds(5): 1/0 subject-motion threshold based on composite-movement measure (default 1)
- art\_thresholds(6): 1/0 force interactive mode (ART gui) (default 0)
- art\_thresholds(7): [only when art\_threshold(5)=0] subject-motion threshold based on rotation measure
- art\_thresholds(8): N number of initial scans to be flagged for removal (default 0)

note: when art\_threshold(5)=0, art\_threshold(2) defines the threshold based on the translation measure, and art\_threshold(7) defines the threshold based on the rotation measure; otherwise art\_threshold(2) defines the (single) threshold based on the composite-motion measure

note: the default art\_thresholds(1:2) [5 .9] values correspond to the "intermediate" (97th percentile) settings; to use the "conservative" (95th percentile) settings use [3 .5]; to use the "liberal" (99th percentile) settings use [9 2] values instead

note: art needs subject-motion files to estimate possible outliers. If a 'realignment' first-level covariate exists it will load the subject-motion parameters from that first-level covariate; otherwise it will look for a rp\_\*.txt file (SPM format) in the same folder as the functional data

note: subject-motion files can be in any of the following formats: a) \*.txt file (SPM format; three translation parameters in mm followed by pitch/roll/yaw in radians); b) \*.par (FSL format; three Euler angles in radians followed by translation parameters in mm); c) \*.siemens.txt (Siemens MotionDetectionParameter.txt format); d) \*.deg.txt (same as SPM format but rotations in degrees instead of radians)

**#boundingbox** : (normalization) target bounding box for resliced volumes (mm) [-90,-126,-72;90,90,108]

**#bp\_filter** : (functional\_bandpass) Low- and High- frequency thresholds (in Hz)

**#bp\_keep0** : (functional\_bandpass) 0: removes average BOLD signal (freq=0Hz component); 1: keeps average BOLD signal in output independent of band-pass filter values; [1]

**#coregtomean** : (functional\_coregister/segment/normalize) 0: use first volume; 1: use mean volume (computed during realignment); 2: use user-defined source volume (see Setup.coregsource\_functionals field) [1]

**#diffusionsteps** : (surface\_smooth) number of diffusion steps

**#fwhm** : (functional\_smooth) Smoothing factor (mm) [8]

**#interp** : (normalization) target voxel interpolation method (0:nearest neighbor; 1:trilinear; 2 or higher:n-order spline) [4]

**#label** : (functional\_label) label of secondary dataset [datestr(now)]

**#reg\_names** : (functional\_regression) list of first-level covariates to use as model regressors / design matrix

**#reg\_dimensions** : (functional\_regression) list of maximum number of dimensions (one value for each model regressor)

**#reg\_deriv** : (functional\_regression) list of 0/1/2 values (one value for each model regressor); add first- or second- order

derivatives to each model regressor

**#reg\_skip** : (functional\_regression) 1: does not create output functional files, only creates session-specific dp\_\*.txt files with covariate timeseries to be included later in an arbitrary first-level model [0]

**#removescans** : (functional\_remove\_scans) number of initial scans to remove

**#reorient** : (functional\_structural\_manualorient) 3x3 or 4x4 transformation matrix or filename containing corresponding matrix

**#respatialdef** : (functional\_structural\_manualspatialdef) nifti deformation file (e.g. y\_\*.nii or \*seg\_sn.mat files)

**#rtm** : (functional\_realign) 0: use first volume; 1: use mean volume [0]

**#sliceorder** : (functional\_slicetime) acquisition order (vector of indexes; 1=first slice in image; note: use cell array for subject-specific vectors)  
alternatively sliceorder may also be defined as one of the following strings: 'ascending', 'descending', 'interleaved (middle-top)', 'interleaved (bottom-up)', 'interleaved (top-down)', 'interleaved (Siemens)', 'interleaved (Philips)', 'BIDS' (this option reads slice timing information from .json files)  
alternatively sliceorder may also be defined as a vector containing the acquisition time in milliseconds for each slice (e.g. for multi-band sequences)

**#ta** : (functional\_slicetime) acquisition time (TA) in seconds (used to determine slice times when sliceorder is defined by a vector of slice indexes; note: use vector for subject-specific values). Defaults to (1-1/nslices)\*TR where nslices is the number of slices

**#template\_structural**: (structural\_normalize SPM8 only) anatomical template file for approximate coregistration [spm/template/T1.nii]

**#template\_functional**: (functional\_normalize SPM8 only) functional template file for normalization [spm/template/EPI.nii]

**#tpm\_template** : (structural\_segment, structural\_segment&normalize in SPM8, and any segment/normalize option in SPM12) tissue probability map [spm/tpm/TPM.nii]

**#tpm\_ngaus** : (structural\_segment, structural\_segment&normalize in SPM8&SPM12) number of gaussians for each tissue probability map

**#vdm\_et1** : (functional\_vdm\_create) ET1 (Echo Time first echo in fieldmap sequence) (default [] : read from .json file / BIDS)

**#vdm\_et2** : (functional\_vdm\_create) ET2 (Echo Time second echo in fieldmap sequence) (default [] : read from .json file / BIDS)

**#vdm\_ert** : (functional\_vdm\_create) ERT (Effective Readout Time in functional data) (default [] : read from .json file / BIDS)

**#vdm\_blip** : (functional\_vdm\_create) k-space traversal blip direction (+1 or -1; default -1)

**#vdm\_type** : (functional\_vdm\_create only) type of fieldmap sequence files ([]: automatically detect; 1: magnitude+phasediff (or magnitude1+magnitude2+phasediff); 2: real1+imag1+real2+imag2; 3: fieldmapHz)

**#vdm\_fmap** : (functional\_vdm\_create only) location of fieldmap sequence files (secondary functional dataset number or label containing fieldmap sequence files) ['fmap']

**#voxelsize\_anat** : (structural normalization) target voxel size for resliced volumes (mm) [2]

**#voxelsize\_func** : (functional normalization) target voxel size for resliced volumes (mm) [2]

**#sets** : defines functional dataset to preprocess (0 for Primary Dataset; [1-N] or labels for Secondary Datasets) [0]

# Running first-level analyses (*fl FIRSTLEVEL*)

## General information

Example syntax:

```
fl FIRSTLEVEL EXAMPLE01 mnispace speech
```

Runs 'speech' firstlevel analyses for subject EXAMPLE01 (data preprocessed using the 'mnispace' pipeline)

- Expects imported dataset in  $\$/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01$  (created in Step 2)
- Expects  $\$/EXAMPLE/config/firstlevel\_speech.cfg$  file describing first-level model estimation details
- Expects experimental design information to: a) have been specified in the EXAMPLE01.cfg file in step1; or b) be explicitly defined in a analysis-specific *firstlevel\\_EXAMPLE01\\_speech.cfg* file (see below)
- Creates  $\$/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech$  folder with first-level analyses

Note: if a file *firstlevel\\_EXAMPLE01\\_speech.cfg* exists within the config directory the design information (i.e. #design field) within this file will be used. Otherwise, the design information in the file *EXAMPLE01.cfg* will be used instead. This option may be used in cases where first-level contrasts or model estimation options may be different among different subjects.

General command syntax:

```
fl FIRSTLEVEL [SubjectID] [PipelineID] [ModelID] ...
```

*[SubjectID]* String identifying individual experiment/subject/session

*[PipelineID]* String identifying an individual preprocessing pipeline

*[ModelID]* String identifying an individual first-level analysis, and associated with an existing *firstlevel\\_ModelID.cfg* file defining first-level model estimation options. Optionally, also associated with an existing *firstlevel\\_SubjectID\\_ModelID.cfg* file (for multiple experimental designs)

Optional additional inputs: fieldName and fieldValue pairs to FL (see *help FL*)

See also: *fl FIRSTLEVEL.PLOT*, *fl FIRSTLEVEL.CONTRAST*

# Configuration file defining first-level model estimation options

Example (firstlevel\_speech.cfg)

```
%% DEFAULT .CFG FILE DEFINING FIRST-LEVEL MODEL OPTIONS
%% see "help fl" for details
%%

#contrasts                                     % contrast definitions : contrast-name condition1 weight1 condition2 wegith2 ...
Speech-Baseline Speech 1 Baseline -1
Baseline-Speech Baseline 1 Speech -1

#model_basis                                   % temporal basis
hrf+deriv

#model_covariates                             % first-level covariates of no interest
motion
linear
art

#model_serial                                 % serial correlations
AR(1)

#hpf                                           % high-pass filter (sec)
128

#functional_label                             % label of input functional data
smoothed
```

## Definition details

**#contrasts** : contrast definition strings. Cell array (with one line per contrast, fields are separated by whitespaces):  
CONTRAST\_NAME CONDITION\_NAME1 CONDITION\_WEIGHT1 CONDITION\_NAME2 CONDITION\_WEIGHT2 ...  
e.g. S-N S 1 N -1

note: valid condition names are:

- 1) (in all cases) those defined in the #names field in design\_\*.cfg files
- 2) (for temporal modulations) [conditionname]xtime, where [conditionname] is a condition name (#names field in design\_\*.cfg files)
- 3) (for parametric modulations including condition-by-covariate interactions; pmod\_interaction=1) [conditionname]x[covariatename], where [conditionname] is a condition name #names field in design\_\*.cfg files and [covariatename] is a parametric covariate name (#pmod\_names field in design\_\*.cfg files)
- 4) (for parametric modulations not including condition-by-covariate interactions; pmod\_interaction=0) [covariatename], where [covariatename] is a parametric covariate name (#pmod\_names field in design\_\*.cfg files)
- 5) (for non-parametric modulations) [conditionname]\_EVENT## where [conditionname] is a condition name (#names field in design\_\*.cfg files) and ## is the event number

(additional data- and model- definition options)

**#functional\_label**: choose version of functional data to enter in first-level analysis: only when used in combination with "functional\_label" preprocessing steps: enter Secondary Dataset label identifying the desired functional dataset (default: Primary Dataset, i.e. fully preprocessed functional data)

**#functional\_smoothinglevel**: choose version of functional data to enter in first-level analysis: only when used in combination with preprocessing pipelines which implement multiple smoothing steps: enter 0/1/2: 0 unsmoothed data (remove any 's' prefix from filename); 1 minimally-smoothed data (remove extra 's' prefixes from filename, keeping at most just one); 2 fully-smoothed data (keep filename unchanged) (default: 1)

**#model\_basis** : hrf / hrf+deriv / hrf+derivs / none : response function: enter hrf for hemodynamic response function only; hrf+deriv to add temporal derivative; hrf+derivs to add temporal and dispersion derivatives; none for no hrf convolution (default: hrf+deriv)

**#model\_covariates**: list of additional covariates; possible values: motion, motion+deriv, motion+deriv+square, art, linear, or <filename>.mat (one file per run) (default: motion / art)

**#model\_serial** : none / AR(1) : serial correlation modeling (default: AR(1))

**#model\_session** : 1/0 : estimates session-specific task effects (default value: 0; when set to 0 SPM estimates session-invariant task effects -it assumes same effect across all sessions-; alternatively set to cell array of task names in order to specify individual task effects that will be model as session-invariant)

**#mthresh** : implicit masking threshold (default value: 0.8)

**#explicitmask** : optional explicit mask (filename)

**#hpf** : high-pass filter threshold -in seconds- (default value: 128s)

**#lpf** : low-pass filter threshold -in seconds- (default value: 0s = no low-pass filter)

**#contrastonly** : 1/0 : skips first-level model definition/estimation step and performs only contrast definition/estimation [0]

**#contrast\_addsession:** 0/1 for each contrast defined in #contrasts field above, create additional SESSION##\_[contrastname] including only the n-th run data (default 0)

**#contrast\_addcv** : 0/1 for each contrast defined in #contrasts field above, create additional SESSION##\_[contrastname] and ORTH\_TO\_SESSION##\_[contrastname] including only the n-th run and all but the n-th run data, respectively (default 0)

**#contrast\_addoddeven:** 0/1 for each contrast defined in #contrasts field above, create additional ODD\_[contrastname] and EVEN\_[contrastname] including only odd and even numbered runs, respectively (default 0)

**#contrast\_removenonestimablecols:** 0/1 When defining contrast vectors skip design-matrix columns that are not fully estimable individually (default 0)



# Running second-level analyses (*fl SECONDLEVEL*)

## General information

Example syntax:

```
fl SECONDLEVEL EXAMPLE mnispace speech average
```

Runs 'average' second-level analyses for subjects in **EXAMPLE** experiment (data preprocessed using the 'mnispace' pipeline, and using the first-level contrasts from the 'speech' analyses)

- Expects firstlevel results in **\$/EXAMPLE/derivatives/mnispace/sub-EXAMPLE\*/results/firstlevel/speech** (created in Step 3)
- Expects **\$/EXAMPLE/config/secondlevel\_average.cfg** file describing second-level model definition (note: additional covariates may be defined in a *participants.tsv* file following BIDS convention and located in the **\$/EXAMPLE/config**)
- Creates **\$/EXAMPLE/derivatives/mnispace/results/secondlevel/speech/average** folder with second-level analyses

General command syntax:

```
fl SECONDLEVEL [StudyID] [PipelineID] [ModelID] [ResultsID] ...
```

**[StudyID]** String identifying individual experiment

**[PipelineID]** String identifying an individual preprocessing pipeline

**[ModelID]** String identifying an individual first-level analysis

**[ResultsID]** String identifying an individual second-level analysis, and associated with an existing **secondlevel\_[ResultsID].cfg** file defining second-level model definition.

Optional additional inputs: additional fieldName and fieldValue pairs for secondlevel configuration (e.g. *mask* field)

Input .cfg file format example: **secondlevel\_correlation.cfg** (see **HELP FL** for additional details):

```
#contrast_names % list of first-level contrasts of interest
Speech-Baseline
#design % second-level design matrix
AllSubjects
age
#contrast_between % between-subjects contrast
0 1
```

See also: *fl SECONDLEVEL.PLOT*, *fl SECONDLEVEL.ROI*

# Configuration file defining second-level analysis

Example (secondlevel\_average.cfg)

```
#contrast_names          % input data (first-level contrasts to be entered in this second-level analysis)
Speech-Baseline

#design                   % design matrix (columns from participants.tsv to be entered in this second-level analysis)
AllSubjects
```

## Example (secondlevel\_correlation.cfg)

```
#contrast_names          % input data (first-level contrasts to be entered in this second-level analysis)
Speech-Baseline

#design                   % design matrix (columns from participants.tsv to be entered in this second-level analysis)
AllSubjects
age

#contrast_between        % between-subjects contrast (select/compare elements in design matrix)
0 1
```

## Example (secondlevel\_anova.cfg)

```
#contrast_names          % input data (first-level contrasts to be entered in this second-level analysis)
SpeechFast-Baseline
SpeechSlow-Baseline

#contrast_within          % within-subjects contrast (select/compare elements in input data)
1 -1

#design                   % design matrix (columns from participants.tsv to be entered in this second-level analysis)
AllSubjects
group

#contrast_between         % between-subjects contrast (select/compare elements in design matrix)
0 1
```

## Definition details

**#contrast\_names:** list of first-level contrasts to enter into second-level analyses (1 x Nmeasures)  
contrast names should match those defined during first-level analyses (i.e. those defined defined in the **#contrasts** field in the firstlevel\_\*.cfg model estimation file)

**#design** : design matrix (Neffects x Nsubjects)  
enter one row for each modeled effect (across subjects)  
each row should contain either the keyword *AllSubjects* (indicating a 'constant term' effect) or a subject/participant covariate name as defined in the participants file (data columns in \$/EXAMPLE/participants.tsv; see below)  
alternatively, each row should contain a vector of values characterizing the modeled effect (with one value/number per subject)

**#contrast\_between:** between-subjects contrast vector/matrix (Nc1 x Neffects)  
when entering a vector, *contrast\_between* should contain as many values as effects entered in the **#design** field above  
when entering a matrix, *contrast\_between* should contain as many columns as effects entered in the **#design** field above, and any arbitrary number of rows  
when not defined, *contrast\_between* defaults to *eye(Neffects)*

**#contrast\_within** : within-subjects contrast vector/matrix (Nc2 x Nmeasures)  
when entering a vector, *contrast\_within* should contain as many values as names entered in the **#contrast\_names** field above  
when entering a matrix, *contrast\_within* should contain as many columns as names entered in the **#contrast\_names** field above, and any arbitrary number of rows  
when not defined, *contrast\_within* defaults to *eye(Nmeasures)*

(optional additional fields)

**#mask** : (optional) analysis mask file

**#analysistype** : (optional, analysis type) 1: include both parametric and non-parametric stats;  
2: include only parametric stats (Random Field Theory assumptions); 3: include only non-parametric stats (permutation/randomization analyses)

**#subjects** : (optional, not recommended) explicit list of subject ids (1 x Nsubjects)  
enter an explicit list of subjectID's (e.g. EXAMPLE01 EXAMPLE02 ...)  
alternatively, enter single key with wildcards to indicate multiple subjects (e.g. sub-\*)  
when not defined, *#subjects* defaults to all subjects in study

**#data** : (optional, not recommended) explicit list of nifti files entered into second-level analysis (Nsubjects x Nmeasures)  
when not defined, *#data* is determined by the choice of **#contrast\_names** values, and the information within each subject first-level estimation SPM.mat files

**#folder** : (optional, not recommended) folder where analysis are stored  
when not defined, *#folder* defaults to \$/[StudyID]/derivatives/[PipelineID]/results/secondlevel/[AnalysisID]/[ResultsID]

## Storing subject-level information

The keyword '*AllSubjects*' used in **#design** field in some of the examples above is a second-level covariate defined implicitly by FL and modelling constant terms in second-level analyses (i.e. a column of all 1's in the design matrix). For any other effect across subjects that we may wish to model we need to define the corresponding subject-level values characterizing this effect. In FL all covariates (other than *AllSubjects*) listed by name in the **#design** field must be explicitly defined in a separate file named "participants.tsv". This file format follows BIDS convention for subject/participant information

- all entries in participants.tsv file are separated by tabs (tsv = Tab-Separated-Values)
- first line details field names and it *must* include a field named "participant\_id" listing valid subjectIDs
- non-numeric covariates (e.g. 'control' and 'patient' in example below) will be converted to numeric values (alphabetical-order ranks; e.g. control=1; patient=2) when entered in second-level analysis design matrix
- for additional details about the format of a participants.tsv file below see BIDS documentation ([https://bids.neuroimaging.io/bids\\_spec1.0.1-rc1.pdf](https://bids.neuroimaging.io/bids_spec1.0.1-rc1.pdf))

An example of a valid *participants.tsv* file encoding subjects gender, age, and group information would be the following:

### Participants.tsv :

```
participant_id age male female group valid
sub-EXAMPLE01 34 1 0 control 1
sub-EXAMPLE02 12 0 1 control 1
sub-EXAMPLE03 na 0 1 patient 1
sub-EXAMPLE04 37 1 0 patient n/a
```

# Running second-level ROI-based analyses (*f1 SECONDLEVEL.ROI*)

## General information

Example syntax:

```
f1 SECONDLEVEL.ROI EXAMPLE mnispace speech average atlas
```

Runs ROI-based analyses using ROIs defined in 'atlas', for new or already computed 'average' second-level analyses in **EXAMPLE** experiment (data preprocessed using the 'mnispace' pipeline, and using the first-level contrasts from the 'speech' analyses)

- Expects **\$/EXAMPLE/config/roi\_atlas.cfg** file describing atlas ROIs (note: multiple RoiID files can be entered to include ROIs from multiple sources)
- Expects already-computed **\$/EXAMPLE/derivatives/mnispace/results/secondlevel/speech/average** folder with second-level analyses (see *f1 SECONDLEVEL*)
- Creates **REX\_atlas.mat** file in **\$/EXAMPLE/derivatives/mnispace/results/secondlevel/speech/average** with ROI-based analysis results

General command syntax:

```
f1 SECONDLEVEL.ROI [StudyID] [PipelineID] [ModelID] [ResultsID] [RoiID] ...
```

**[StudyID]** String identifying individual experiment

**[PipelineID]** String identifying an individual preprocessing pipeline

**[ModelID]** String identifying an individual first-level analysis

**[ResultsID]** String identifying an individual second-level analysis, and associated with an existing secondlevel\_**[ResultsID]**.cfg file defining second-level model definition.

**[RoiID]** String identifying an individual set of regions of interest

Optional additional inputs: fieldName and fieldValue pairs for roi configuration (e.g. *roi\_labels* field)

Input .cfg file format example: **roi\_atlas.cfg** (see HELP FL for additional details):

```
#rois          % list of roi files
/project/busplab/software/conn/rois/atlas.nii
```

See also: *f1 SECONDLEVEL.ROI.PLOT*, *f1 SECONDLEVEL*

## Configuration file defining ROI

Example (roi\_atlas.cfg)

```
#rois                % list of ROI files  
/project/busplab/software/conn/rois/atlas.nii
```

Example (roi\_parcel.cfg; using subject-specific ROI files)

```
#rois                % list of ROI files  
/projectnb/busplab/rois/SEQPDS/parcels_<subjectid>.nii
```

```
#roi_ss              % subject-specific ROIs  
1
```



## Definition details

**#rois** : list of ROI files  
note: for subject-specific files:  
a) for subject-specific ROIs that have been already defined during the “fl IMPORT” step, **#rois** may instead contain a list of ROI names (see fl IMPORT **#rois.files** and **#rois.names** fields)  
b) alternatively, for arbitrary subject-specific ROIs, file names or file paths listed in the **#rois** field may include the keyword “<SUBJECTID>” (without quotes) to indicate subject-specific files. The key <SUBJECTID> key will then be automatically replaced by the actual *SubjectID* (e.g. same subject IDs used in the “fl IMPORT *SubjectID*” syntax) of each individual subject entered into the second-level analysis (e.g. “/sub-<SUBJECTID>/mri/roi.nii” will be replaced by “/sub-EXAMPLE01/mri/roi.nii”; similarly, “/mri/roi\_<SUBJECTID>.nii” will be replaced by “/mri/roi\_EXAMPLE01.nii”, etc.)

**#roi\_measure** : (optional) measure computed within each ROI {'mean', 'eigenvariate', 'weighted mean', 'weighted', 'eigenvariate', 'median', 'sum', 'weighted sum', 'count', 'max', 'min'}

**#roi\_threshold**: (optional) absolute-threshold defining voxels within each ROI [0]

**#roi\_labels** : (optional) list of ROI indexes or ROI labels to be included in analysis (by default when entering atlas files defining multiple ROIs all ROIs in the atlas are included in the analyses) ({})

**#roi\_groups** : (optional) list of ROIs to group together as a single ROI in second-level analysis (by default each ROI is analyzed separately) ({})  
note: For multiple groups separate the list of ROI indexes/labels by the label of each new group-ROI (e.g. enter “group1 roi1 roi2 group2 roi3 roi4 roi5” to define two groups of rois)

**#roi\_ss** : (optional) 1/0 indicating whether ROIs are subject-specific [0]

## Optional/advanced commands

To launch GUI displaying imported subject data:

```
f1 OPEN [SubjectID]
```

To launch GUI displaying imported&preprocessed subject data:

```
f1 OPEN [SubjectID] [PipelineID]
```

To create QA plots displaying preprocessing results:

```
f1 QA.CREATE [SubjectID] [PipelineID]
```

To create QA plots displaying firstlevel results:

```
f1 QA.CREATE [SubjectID] [PipelineID] [ModelID]
```

To display QA plots for individual subject:

```
f1 QA.PLOT [SubjectID] [PipelineID]
```

To display QA plots for all subjects in a given experiment:

```
f1 QA.PLOTS [StudyID] [PipelineID]
```

To convert all original DICOM files to nifti format without yet importing them:

```
f1 CONVERT.DICOM [SubjectID]
```

Note: for this use dicom files are expected to be located in the sub-[SubjectID]/scanner/ folder, output run-#.nii/json files will be stored in same folder (and optionally organized into anat, func, fmap, and dwi subfolders)

To run *additional* preprocessing steps to an existing (already-preprocessed) subject:

```
f1 PREPROCESSING.APPEND [SubjectID] [PipelineID] [Pipeline2ID]
```

Note: preprocessing output will be stored in the same derivatives/[PipelineID]/sub-[SubjectID] folder as the original data

Note: expects pipeline\_[Pipeline2ID].cfg file (or pipeline\_[SubjectID]\_[Pipeline2ID].cfg, for subject-specific preprocessing steps)

To run *additional* preprocessing steps to an existing (already-preprocessed) subject, and create a new preprocessing pipeline branch with the results:

```
f1 PREPROCESSING.BRANCH [SubjectID] [PipelineID] [Pipeline2ID]
```

Note: preprocessing output will be stored in a new derivatives/[Pipeline2ID]/sub-[SubjectID] folder

To run preprocessing steps in parallel (submitting a job to the SCC cluster):

```
f1 PREPROCESSING [SubjectID] [PipelineID] parallel
```

To check status of jobs submitted to SCC cluster :

```
f1 PREPROCESSING.REPORT [SubjectID] [PipelineID]
```

To delete jobs submitted to SCC cluster :

```
f1 PREPROCESSING.DELETE [SubjectID] [PipelineID]
```

To display first-level results:

```
f1 FIRSTLEVEL.PLOT [SubjectID] [PipelineID] [ModelID]
```

To run contrast-estimation step (skip first-level model definition and estimation):

```
f1 FIRSTLEVEL.CONTRAST [SubjectID] [PipelineID] [ModelID]
```

To display second-level results:

```
f1 SECONDLEVEL.PLOT [StudyID] [PipelineID] [ModelID] [ResultsID]
```

To display second-level results within ROIs:

```
f1 SECONDLEVEL.ROI [StudyID] [PipelineID] [ModelID] [ResultsID] [RoiID]
```

To import multiple subjects simultaneously use wildcards or explicit lists, e.g.

```
f1 IMPORT EXAMPLE*
```

or

```
f1( 'IMPORT', { 'EXAMPLE01', 'EXAMPLE03' } )
```

To preprocess multiple subjects simultaneously use wildcards or explicit lists, e.g.

```
f1 PREPROCESSING EXAMPLE* mnispace
```

or

```
f1( 'PREPROCESSING', { 'EXAMPLE01', 'EXAMPLE03' }, 'mnispace' )
```

To run first-level analyses for multiple subjects simultaneously use wildcards or explicit lists, e.g.

```
fl FIRSTLEVEL EXAMPLE* mnispace speech
```

or

```
fl('FIRSTLEVEL',{'EXAMPLE01','EXAMPLE03'},'mnispace','speech')
```

To list subjects defined in an experiment:

```
fl LIST [StudyID]
```

To list all preprocessing pipelines run on a subject:

```
fl LIST [SubjectID]
```

To list all first-level analyses run using data from a preprocessing pipeline:

```
fl LIST [SubjectID] [PipelineID]
```

To list all effect-names defined in an existing first-level analysis:

```
fl LIST [SubjectID] [PipelineID] [ModelID]
```

To grant read&write permissions to other members of your user group:

```
fl FIXPERMISSIONS [SubjectID] [PipelineID]
```

Note: needed for back-compatibility only; fl commands do this automatically

To initialize FL and check your SPM/CONN software versions:

```
fl INIT
```

Note: needed for compatibility with 3rd-party user-created functions/scripts; fl commands do this automatically

See *HELP FL* for additional information

# Example dataset directory structure

## *Generated by user (input data and configuration files):*

/projectnb/busplab/FL/EXAMPLE/config/EXAMPLE01.cfg  
/projectnb/busplab/FL/EXAMPLE/config/SessionDesignA.cfg  
/projectnb/busplab/FL/EXAMPLE/config/SessionDesignB.cfg  
/projectnb/busplab/FL/EXAMPLE/config/SessionDesignC.cfg  
/projectnb/busplab/FL/EXAMPLE/config/SessionDesignD.cfg  
/projectnb/busplab/FL/EXAMPLE/config/firstlevel\_speech.cfg  
/projectnb/busplab/FL/EXAMPLE/config/preprocessing\_mnispace.cfg  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/scanner/run00\_09.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/scanner/run00\_61.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/scanner/run01\_63.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/scanner/run02\_65.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/scanner/run03\_67.nii

## *Generated by script Step 1:*

/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/anat/sub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-01\_bold.json  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-01\_bold.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-02\_bold.json  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-02\_bold.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-03\_bold.json  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-03\_bold.nii  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-04\_bold.json  
/projectnb/busplab/FL/EXAMPLE/sub-EXAMPLE01/func/sub-EXAMPLE01\_run-04\_bold.nii

## *Generated by script Step 2:*

/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/c1csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/c2csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/c3csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/c4csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/c5csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/centering\_sub-EXAMPLE01\_run-01\_T1w.mat  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/csub-EXAMPLE01\_run-01\_T1w\_seg8.mat  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/icentering\_sub-EXAMPLE01\_run-01\_T1w.mat  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/iy\_csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/sub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/wc0csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/wc1csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/wc2csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/wc3csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/wcsub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/anat/y\_csub-EXAMPLE01\_run-01\_T1w.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/art\_mask\_rsub-EXAMPLE01\_run-01\_bold.nii  
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/art\_mean\_rsub-EXAMPLE01\_run-01\_bold.nii

[illegible]

```

/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/swrsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/swrsub-EXAMPLE01_run-02_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/swrsub-EXAMPLE01_run-03_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/swrsub-EXAMPLE01_run-04_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wart_mean_rsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wc1art_mean_rsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wc2art_mean_rsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wc3art_mean_rsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wrsub-EXAMPLE01_run-01_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wrsub-EXAMPLE01_run-02_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wrsub-EXAMPLE01_run-03_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/wrsub-EXAMPLE01_run-04_bold.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/func/y_art_mean_rsub-EXAMPLE01_run-01_bold.nii

```

*Generated by script Step 3:*

[illegible]

[illegible]



```
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/beta_0078.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/con_0001.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/con_0002.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/contrastdefinitions.stderr
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/mask.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/spmT_0001.nii
/projectnb/busplab/FL/EXAMPLE/derivatives/mnispace/sub-EXAMPLE01/results/firstlevel/speech/spmT_0002.nii
```

Generated by script QA plots:

[illegible]