RABIES Documentation

Release 0.5.1

CoBrALab

Nov 12, 2023

CONTENT

1 What you can do with RABIES **3**
- 1.1 Preprocessing . . . 3
- 1.2 Confound correction . . . 3
- 1.3 Analysis . . . 4

2 Notes on software design **5**

3 Citation **7**

4 License **9**

5 Acknowledgements **11**

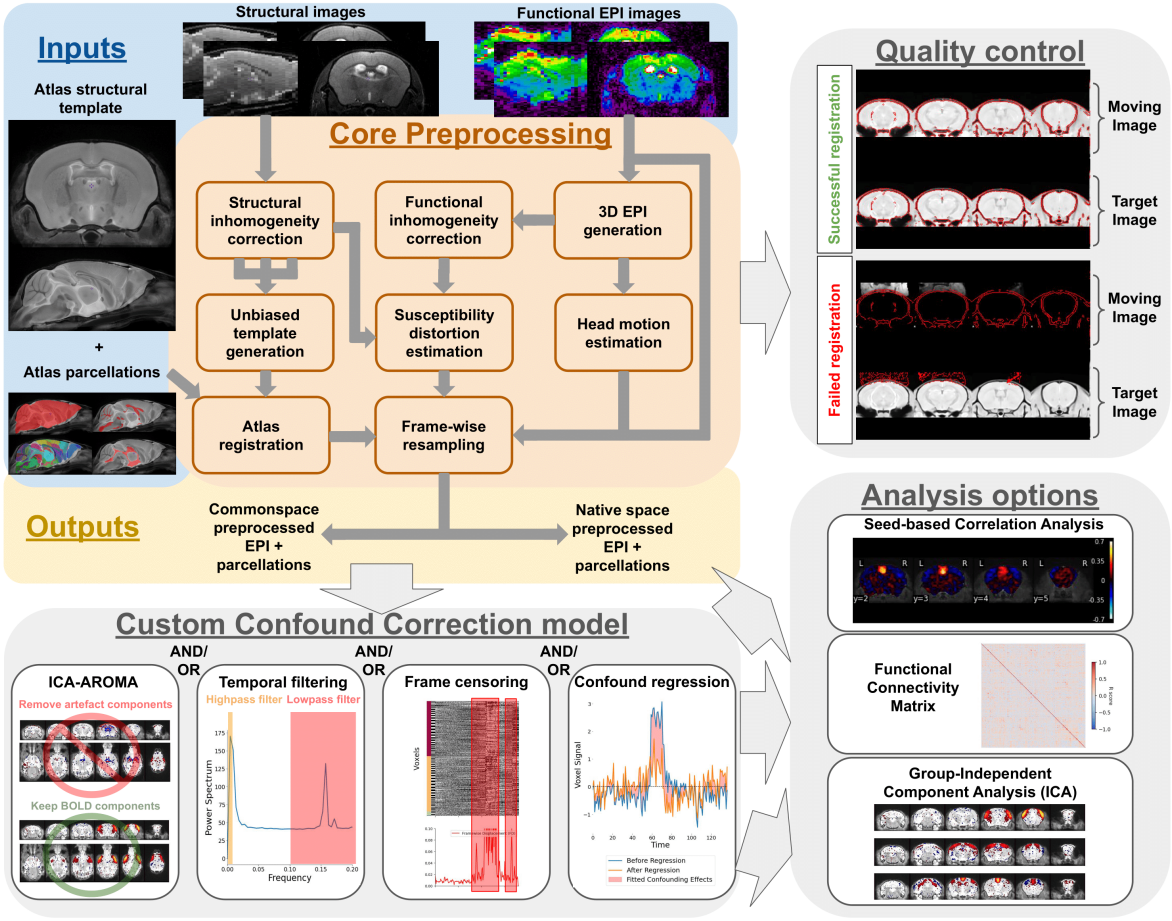
6 Ask for help **13**

7 Contributing to RABIES **15**
- 7.1 Installation . . . 15
  - 7.1.1 PyPi . . . 15
  - 7.1.2 Container (Singularity/Docker) . . . 15
  - 7.1.3 Neurodesk . . . 16
- 7.2 Running The Software . . . 16
  - 7.2.1 Input data in BIDS standard . . . 16
  - 7.2.2 Command Line Interface . . . 16
  - 7.2.3 Example execution syntax . . . 36
  - 7.2.4 Execution syntax with containerized installation (Singularity and Docker) . . . 37
  - 7.2.5 Additional Resources . . . 38
- 7.3 Preprocessing Pipeline . . . 39
  - 7.3.1 Structural inhomogeneity correction . . . 40
  - 7.3.2 Common space alignment (i.e. Unbiased template generation + Atlas registration) . . . 43
  - 7.3.3 3D EPI generation . . . 46
  - 7.3.4 Head motion estimation . . . 47
  - 7.3.5 Functional inhomogeneity correction . . . 49
  - 7.3.6 Susceptibility distortion estimation . . . 49
  - 7.3.7 Frame-wise resampling . . . 50
  - 7.3.8 Adapted workflow without structural scans (i.e. –bold_only) . . . 52
- 7.4 Preprocessing quality control (QC) . . . 52
  - 7.4.1 Recommendations for registration troubleshooting . . . 54
- 7.5 Confound Correction pipeline . . . 55
  - 7.5.1 rabies.confound_correction_pkg.confound_correction.init_confound_correction_wf [source code] . . . 57
- 7.6 Connectivity Analysis . . . 58

7.6.1 Correlation-based connectivity . . . 58
7.6.2 ICA-based connectivity . . . 59
7.7 Data quality assessment . . . 59
7.7.1 Scan diagnosis report . . . 59
7.7.2 Distribution plot . . . 65
7.7.3 Group stats . . . 67
7.7.4 Optimization of confound correction strategy . . . 67
7.7.5 Generating the reports . . . 68
7.7.6 Guidelines for analysis quality control . . . 70
7.8 Understanding the Outputs . . . 71
7.8.1 Preprocessing Outputs . . . 71
7.8.2 Confound Correction Outputs . . . 73
7.8.3 Analysis Outputs . . . 73
7.9 Metric definitions . . . 75
7.9.1 Nuisance regressors for confound regression . . . 75
7.9.2 Temporal scan diagnosis . . . 76
7.9.3 Spatial scan diagnosis . . . 77
7.9.4 Distribution plot . . . 77
7.9.5 Group statistical QC report . . . 77
7.10 Contributing to RABIES . . . 78
7.10.1 Dev environment . . . 78
7.10.2 Instructions to create a pull request . . . 79
7.10.3 Interactive debugging with Spyder and debug_workflow.py . . . 79
7.10.4 Creation of a new module and integration within a Nipype workflow . . . 79
7.11 Bibliography . . . 81

Bibliography **83**

RABIES is an open source image processing pipeline for rodent fMRI. It conducts state-of-the-art preprocessing and confound correction, and supplies standard resting-state functional connectivity analyses. Visit our documentation at https://rabies.readthedocs.io/en/stable/.



CHAPTER

ONE

WHAT YOU CAN DO WITH RABIES

The primary purpose of RABIES is to provide rodent fMRI research with a standard, flexible, and reliable image processing platform. The package is complemented with informative data diagnostic features for confound characterization and encourages best practices for quality control and reproducibility. The RABIES software is structured into three main processing stages: **preprocessing**, **confound correction** and **analysis**.

1.1 Preprocessing

The preprocessing workflow regroups essential fMRI preprocessing steps prior to analysis. It includes a robust registration workflow with automatically-adapting parameters allowing to succesfully process diverse acquisition types (i.e. rodent species, scan field strength, coil type, ...), and can conduct the following preprocessing steps:

- head motion correction
- susceptibility distortion correction
- resampling to native or common space
- brain parcellation
- slice timing correction (optional)
- despiking (optional)
- visual assessment of registration for quality control

1.2 Confound correction

Following preprocessing, a range of strategies to correct fMRI confounds (e.g. motion) can then be conducted within RABIES:

- linear detrending
- confound regression (with several options for nuisance regressors)
- frequency filtering (highpass, lowpass, bandpass)
- frame censoring (or scrubbing)
- ICA-AROMA
- spatial smoothing

1.3 Analysis

Simple resting-state connectivity analyses are made available after preprocessing and confound correction. RABIES also provides a 'data diagnosis' workflow, which generates several indices of data quality and potential confounds, and conversaly, aims to improve the correction of confounds and transparency with regards to data quality:

- seed-based functional connectivity
- whole-brain connectivity matrix
- group-ICA
- dual regression
- data diagnosis

CHAPTER

TWO

NOTES ON SOFTWARE DESIGN

Nipype workflows: The image processing pipelines are structured using the Nipype library, which allows to build dynamic workflows in the form of a computational graph. Each node in the graph consists of a processing step, and the required inputs/outputs define the links between nodes. In addition to supporting code organization, Nipype workflows also handle several plugin architectures for parallel execution as well as memory management. The computational time to run the entire RABIES pipeline will vary substantially depending on data size, but for most uses, it will range **from a few hours to a day** when using proper computational resources and parallel execution.

Reproducible and transparent research: RABIES aims to follow best practices for reproducible and transparent research, including the following:

- open source code https://github.com/CoBrALab/RABIES
- standardized input data format with BIDS
- easily shared, automatically-generated visual outputs for quality control
- containerized distribution of the software hosted on Docker Hub which can be downloaded via Docker and Singularity platforms

CHAPTER

THREE

CITATION

Citing RABIES: A preprint introducing the software is available on bioXriv, which should be cited accordingly when using RABIES.

Boilerplate: a boilerplate summarizing the preprocessing and confound correction operations is automatically generated in the output folder. You can use the boilerplate to help describe your methods in a paper.

CHAPTER

FOUR

LICENSE

The RABIES license allows for uses in academic and educational environments only. Commercial use requires a commercial license from CoBrALab contact@cobralab.ca, http://cobralab.ca

CHAPTER

FIVE

ACKNOWLEDGEMENTS

This software was developped by the CoBrALab, located at the Cerebral Imaging Center of the Douglas Mental Health University Institute, Montreal, Canada, in affiliation with McGill University, Montreal, Canada. This work was supported by funding from Healthy Brains, Healthy Lives (HBHL), the Fonds de recherche du Québec - Santé (FRQS) and - Nature et technologies (FRQNT), and the Natural Sciences and Engineering Research Council (NSERC) of Canada. fMRIPrep was an important inspirational source for this project, in particular with regards to best practices for software reproducibility and code design using Nipype. We also thank the organizers of BrainHack School Montreal, which guided the initial steps of this project in 2018.

CHAPTER

SIX

ASK FOR HELP

If you need support in using the software or experience issues that are not documented, we'll provide support on the Github discussion.

CHAPTER

SEVEN

CONTRIBUTING TO RABIES

Read our dedicated documentation

7.1 Installation

7.1.1 PyPi

The software is available on PyPi, which makes the rabies python package widely accessible with

```
pip install rabies
```

However, this does not account for non-python dependencies found in `dependencies.txt`.

7.1.2 Container (Singularity/Docker)

For most uses, we recommend instead using a containerized installation with Singularity or Docker. Containers allow to build entire computing environments, grouping all dependencies required to run the software. This in turn reduces the burden of installing dependencies manually and ensures reproducible behavior of the software. Singularity is generally preferred over Docker since it requires less permission, and can thus be imported from most computing environment (e.g. high performance computing clusters such as Compute Canada.)

A containerized version of RABIES is available from Github. After installing Singularity or Docker, the following command will pull and build the container:

- Install Singularity .sif file:

```
singularity build rabies.sif docker://ghcr.io/cobralab/rabies:latest
```

- Install Docker image:

```
docker pull ghcr.io/cobralab/rabies:latest
```

A specific tag version can be selected (instead of `latest`) from the list online. Versions prior to 0.5.0 are found on Docker Hub.

7.1.3 Neurodesk

RABIES is also made available on the Neurodesk platform, as part of the built-in tools for neuroimaging. The Neurodesk platform allows for an entirely browser-based neuroimaging computing environment, with pre-built neuroimaging tools from the community, and aims at reducing needs for manual development of computing environments and at improving reproducible neuroimaging. More details on Neurodesk here https://neurodesk.github.io/docs/.

7.2 Running The Software

7.2.1 Input data in BIDS standard

The input dataset must be organized according to the BIDS data structure[GAC+16]. RABIES will iterate through all subjects found to contain a functional file (see section on BIDS filters below), and will also iterate according to sessions and runs found within each subject if available. If anatomical scans are used for preprocessing (i.e. not using `--bold_only`), each functional scan will be matched to one corresponding anatomical scan of the same subject/session (using BIDS filters for the anatomical image, see below).

Directory structure for an example dataset

- Our example dataset has the following BIDS structure:

8 directories, 8 files

BIDS filters to identify functional and structural images

By default, RABIES will use the 'bold' or 'cbv' suffix to identify functional scans and the 'T1w' or 'T2w' suffix for structural scans. Files which don't match the BIDS filters are ignored. However, the BIDS filters can also be customized with the `--bids_filter` parameter during the preprocessing stage. This can be useful for instance if the default is not enough to find the right set of scans. The custom BIDS filter must be formated into a JSON file with the functional filter under 'func' and structural filter under 'anat' (see example below for the default parameters):

```
{
    "func": {
        "suffix":["bold","cbv"]
    },
    "anat": {
        "suffix":["T1w","T2w"]
    }
}
```

7.2.2 Command Line Interface

RABIES is executed using a command line interface, within a terminal. The software is divided into three main processing stages: preprocessing, confound correction and analysis. Accordingly, the command line interface allows for three different mode of execution, corresponding to the processing stages. So first, when executing the software, one of the processing stage must be selected. Below you can find the general –help message printed with `rabies --help`, which provides a summary of each processing stage together with options for parallel processing and memory management. Then, the –help associated to each processing stage, i.e. `preprocess`, `confound_correction` and `analysis`, describes in more detail the various parameters available to adapt image processing according to the user needs. Click on the corresponding –help to expand:

```
usage: rabies [-h] [--inclusion_ids [INCLUSION_IDS ...]]
              [--exclusion_ids [EXCLUSION_IDS ...]]
              [-p {Linear,MultiProc,SGE,SGEGraph,PBS,LSF,SLURM,SLURMGraph}]
              [--local_threads LOCAL_THREADS]
              [--scale_min_memory SCALE_MIN_MEMORY] [--min_proc MIN_PROC]
              [--figure_format {png,svg}] [--verbose VERBOSE] [-f]
              Processing stage ...

RABIES performs multiple stages of rodent fMRI image processing, including preprocessing,
confound correction, simple analyses and data quality assessment.

optional arguments:
  -h, --help            show this help message and exit

Processing options:
  The RABIES workflow is seperated into three main processing stages: preprocessing,
  confound correction and analysis. Outputs from the preprocessing provide the inputs for
  the subsequent confound correction, and finally analysis.

  Processing stage      Description
    preprocess
                        Conducts preprocessing on an input dataset in BIDS format.␣
↪Preprocessing includes
                        motion realignment, susceptibility distortions correction␣
↪through non-linear
                        registration, alignment to commonspace, anatomical parcellation␣
↪and evaluation of
                        nuisance timecourses.

    confound_correction

                        Flexible options for confound correction are applied directly on␣
↪preprocessing outputs
                        from RABIES to derive cleaned timeseries. Various correction␣
↪strategies, if selected, are
                        applied in the following order, following best practices from␣
↪human litterature:
                           #1 - Compute and apply frame censoring mask (from FD and/or␣
↪DVARS thresholds)
                           #2 - If --match_number_timepoints is selected, each scan is␣
↪matched to the
                                defined minimum_timepoint number of frames.
                           #3 - Linear/Quadratic detrending of fMRI timeseries and␣
↪nuisance regressors
                           #4 - Apply ICA-AROMA.
                           #5 - If frequency filtering and frame censoring are applied,␣
↪simulate data in censored
                                timepoints using the Lomb-Scargle periodogram, as␣
↪suggested in Power et al. (2014,
                                Neuroimage), for both the fMRI timeseries and nuisance␣
↪regressors prior to filtering.
                           #6 - As recommended in Lindquist et al. (2019, Human brain␣
↪mapping), make the nuisance
```

(continues on next page)

(continued from previous page)

```
                              regressors orthogonal to the temporal frequency filter.
                           #7 - Apply highpass and/or lowpass filtering on the fMRI␣
↪timeseries (with simulated
                              timepoints).
                           #8 - Re-apply the frame censoring mask onto filtered fMRI␣
↪timeseries and nuisance
                              regressors, taking out the simulated timepoints. Edge␣
↪artefacts from frequency
                              filtering can also be removed as recommended in Power et␣
↪al. (2014, Neuroimage).
                           #9 - Apply confound regression using the selected nuisance␣
↪regressors (see --conf_list
                              options).
                           #10 - Scaling of timeseries variance
                           #11 - Apply Gaussian spatial smoothing.

    analysis
                        Conduct simple resting-state functional connectivity (FC)␣
↪analysis, or data quality
                        diagnosis, on cleaned timeseries after confound correction.␣
↪Analysis options include
                        seed-based FC, whole-brain FC matrix, group-ICA and dual␣
↪regression. --data_diagnosis
                        computes features of data quality at the individual scan and␣
↪group levels, as in
                        Desrosiers-Gregoire et al. (in prep)


Execution Options:
  Options for parallel execution and memory management.

  --inclusion_ids [INCLUSION_IDS ...]
                        Define a list of BOLD scan to include, i.e. run the pipeline on␣
↪a subset of the data.
                        To do so, provide the full path to the corresponding BOLD file␣
↪in the input BIDS folder. The list
                        of scan can be specified manually as a list of file name '--scan_
↪list scan1.nii.gz
                        scan2.nii.gz ...' or the files can be imbedded into a .txt file␣
↪with one filename per row.
                        By default, 'all' the scans found in the input BIDS directory or␣
↪from the previous
                        processing step. This can be provided at any processing stage.
                        ***NOTE: do not enter this parameter right before the processing␣
↪stage (preprocess, etc...), this will cause
                        parsing errors. Instead, provide another parameter after --
↪inclusion_ids (e.g. --verbose or -p).
                        (default: ['all'])

  --exclusion_ids [EXCLUSION_IDS ...]
                        Instead of providing a list of scans to include, this argument␣
↪provides a list of scans to exclude (while
```

(continues on next page)

(continued from previous page)

```
                        keeping all other scans). This argument follows the same syntax␣
↪rules as --includion_ids. --exclusion_ids
                        and --inclusion_ids cannot be used simultaneously.
                        (default: ['none'])

  -p {Linear,MultiProc,SGE,SGEGraph,PBS,LSF,SLURM,SLURMGraph}, --plugin {Linear,
↪MultiProc,SGE,SGEGraph,PBS,LSF,SLURM,SLURMGraph}
                        Specify the nipype plugin for workflow execution.
                        Consult https://nipype.readthedocs.io/en/0.11.0/users/plugins.
↪html for details.
                        (default: Linear)

  --local_threads LOCAL_THREADS
                        For --plugin MultiProc, set the maximum number of processors run␣
↪in parallel.
                        Defaults to number of CPUs.
                        (default: 2)

  --scale_min_memory SCALE_MIN_MEMORY
                        For --plugin MultiProc, set the memory scaling factor attributed␣
↪to nodes during
                        execution. Increase the scaling if memory crashes are reported.
                        (default: 1.0)

  --min_proc MIN_PROC   For --plugin SGE/SGEGraph, scale the number of nodes attributed␣
↪to jobs to
                        avoid memory crashes.
                        (default: 1)

  --figure_format {png,svg}
                        Select the file format for figures generated by RABIES.
                        (default: png)

  --verbose VERBOSE     Set the verbose level. 0=WARNING, 1=INFO, 2 or above=DEBUG.
                        (default: 1)

  -f, --force           The pipeline will not stop if previous outputs are encountered.
                        Previous outputs will be overwritten.
                        (default: False)
```

```
usage: rabies preprocess [-h] [--bids_filter BIDS_FILTER] [--bold_only]
                         [--anat_autobox] [--bold_autobox]
                         [--oblique2card {none,affine,3dWarp}]
                         [--apply_despiking]
                         [--HMC_option {intraSubjectBOLD,0,1,2,3,optim}]
                         [--isotropic_HMC] [--voxelwise_motion]
                         [--apply_slice_mc] [--detect_dummy]
                         [--data_type {int16,int32,float32,float64}]
                         [--anat_inho_cor ANAT_INHO_COR]
                         [--anat_robust_inho_cor ANAT_ROBUST_INHO_COR]
                         [--bold_inho_cor BOLD_INHO_COR]
                         [--bold_robust_inho_cor BOLD_ROBUST_INHO_COR]
```

(continues on next page)

(continued from previous page)

```
                          [--commonspace_reg COMMONSPACE_REG]
                          [--inherit_unbiased_template INHERIT_UNBIASED_TEMPLATE]
                          [--bold2anat_coreg BOLD2ANAT_COREG]
                          [--nativespace_resampling NATIVESPACE_RESAMPLING]
                          [--commonspace_resampling COMMONSPACE_RESAMPLING]
                          [--anatomical_resampling ANATOMICAL_RESAMPLING]
                          [--apply_STC] [--TR TR]
                          [--tpattern {alt-z,alt-z2,seq-z,alt+z,alt+z2,seq+z}]
                          [--stc_axis {X,Y,Z}]
                          [--interp_method {linear,cubic,quintic,heptic,wsinc5,wsinc9,
↪fourier}]
                          [--anat_template ANAT_TEMPLATE]
                          [--brain_mask BRAIN_MASK] [--WM_mask WM_MASK]
                          [--CSF_mask CSF_MASK] [--vascular_mask VASCULAR_MASK]
                          [--labels LABELS]
                          bids_dir output_dir

positional arguments:
  bids_dir              The root folder of the BIDS-formated input data directory.

  output_dir            the output path to drop outputs from major preprocessing steps.


optional arguments:
  -h, --help            show this help message and exit
  --bids_filter BIDS_FILTER
                        Allows to provide additional BIDS specifications (found within␣
↪the input BIDS directory)
                        for selected a subset of functional and/or anatomical images.␣
↪Takes as input a JSON file
                        containing the set of parameters for functional image under 'func
↪' and under 'anat' for the
                        anatomical image. See online documentation for an example.
                        (default: {'func': {'suffix': ['bold', 'cbv']}, 'anat': {'suffix
↪': ['T1w', 'T2w']}})

  --bold_only           Apply preprocessing with only EPI scans. Commonspace␣
↪registration is executed directly using
                        the corrected EPI 3D reference images. The commonspace␣
↪registration simultaneously applies
                        distortion correction, this option will produce only commonspace␣
↪outputs.
                        (default: False)

  --anat_autobox        Crops out extra space around the brain on the structural image␣
↪using AFNI's 3dAutobox
                        https://afni.nimh.nih.gov/pub/dist/doc/program_help/3dAutobox.
↪html.
                        (default: False)

  --bold_autobox        Crops out extra space around the brain on the EPI image using␣
↪AFNI's 3dAutobox
```

(continues on next page)

(continued from previous page)

```
                        https://afni.nimh.nih.gov/pub/dist/doc/program_help/3dAutobox.
↪html.
                        (default: False)

  --oblique2card {none,affine,3dWarp}
                        Correct for oblique coordinates on all structural and functional␣
↪data.
                           WARNING: these corrections are suboptimal, and may alter the␣
↪data. Only apply if necessary.

                        affine: only the affine matrix is changed to cardinal axes.

                        3dWarp: Applies AFNI's 3dWarp -oblique2card. This involves␣
↪resampling the
                        data on a new isotropic grid.
                        (default: none)

  --apply_despiking     Applies AFNI's 3dDespike https://afni.nimh.nih.gov/pub/dist/doc/
↪program_help/3dDespike.html.
                        (default: False)

  --HMC_option {intraSubjectBOLD,0,1,2,3,optim}
                        Select a pre-built option for registration during head motion␣
↪realignment. 'optim' was customized
                        as documented in https://github.com/CoBrALab/RABIES/discussions/
↪259. Other options were taken from
                        https://github.com/ANTsX/ANTsR/blob/master/R/ants_motion_
↪estimation.R.
                        (default: optim)

  --isotropic_HMC       Whether to resample the EPI to isotropic resolution (taking the␣
↪size of the axis with highest
                        resolution) for the estimation of motion parameters. This should␣
↪greatly mitigating registration
                        'noise' which arise from partial volume effects, or poor image␣
↪resolution (see online post on
                        this topic https://github.com/CoBrALab/RABIES/discussions/288).␣
↪This option will increase
                        computational time, given the higher image resolution.
                        (default: False)

  --voxelwise_motion    Whether to output estimates of absolute displacement and␣
↪framewise displacement at each voxel.
                        This will generate 4D nifti files representing motion timeseries␣
↪derived from the 6 motion
                        parameters. This is handled by antsMotionCorrStats.
                        (default: False)

  --apply_slice_mc      Whether to apply a slice-specific motion correction after␣
↪initial volumetric HMC. This can
                        correct for interslice misalignment resulting from within-TR␣
↪motion. With this option,
```

(continues on next page)

(continued from previous page)

```
                        motion corrections and the subsequent resampling from␣
↪registration are applied sequentially
                        since the 2D slice registrations cannot be concatenate with 3D␣
↪transforms.
                        (default: False)

  --detect_dummy        Detect and remove initial dummy volumes from the EPI, and␣
↪generate a reference EPI based on
                        these volumes if detected. Dummy volumes will be removed from␣
↪the output preprocessed EPI.
                        (default: False)

  --data_type {int16,int32,float32,float64}
                        Specify data format outputs to control for file size.
                        (default: float32)


Registration Options:
  Customize registration operations and troubleshoot registration failures.

  --anat_inho_cor ANAT_INHO_COR
                        Select options for the inhomogeneity correction of the␣
↪structural image.
                        * method: specify which registration strategy is employed for␣
↪providing a brain mask.
                        *** Rigid: conducts only rigid registration.
                        *** Affine: conducts Rigid then Affine registration.
                        *** SyN: conducts Rigid, Affine then non-linear registration.
                        *** no_reg: skip registration.
                        *** N4_reg: previous correction script prior to version 0.3.1.
                        *** disable: disables the inhomogeneity correction.
                        * otsu_thresh: The inhomogeneity correction script necessitates␣
↪an initial correction with a
                         Otsu masking strategy (prior to registration of an anatomical␣
↪mask). This option sets the
                         Otsu threshold level to capture the right intensity␣
↪distribution.
                        *** Specify an integer among [0,1,2,3,4].
                        * multiotsu: Select this option to perform a staged␣
↪inhomogeneity correction, where only
                         lower intensities are initially corrected, then higher␣
↪intensities are iteratively
                         included to eventually correct the whole image. This technique␣
↪may help with images with
                         particularly strong inhomogeneity gradients and very low␣
↪intensities.
                        *** Specify 'true' or 'false'.
                        (default: method=SyN,otsu_thresh=2,multiotsu=false)

  --anat_robust_inho_cor ANAT_ROBUST_INHO_COR
                        When selecting this option, inhomogeneity correction is executed␣
↪twice to optimize
```

(continues on next page)

(continued from previous page)

```
                        outcomes. After completing an initial inhomogeneity correction␣
↪step, the corrected outputs
                        are co-registered to generate an unbiased template, using the␣
↪same method as the commonspace
                        registration. This template is then masked, and is used as a new␣
↪target for masking during a
                        second iteration of inhomogeneity correction. Using this dataset-
↪specific template should
                        improve the robustness of masking for inhomogeneity correction.
                        * apply: select 'true' to apply this option.
                         *** Specify 'true' or 'false'.
                        * masking: Combine masks derived from the inhomogeneity␣
↪correction step to support
                         registration during the generation of the unbiased template,␣
↪and then during template
                         registration.
                         *** Specify 'true' or 'false'.
                        * brain_extraction: conducts brain extraction prior to template␣
↪registration based on the
                         combined masks from inhomogeneity correction. This will enhance␣
↪brain edge-matching, but
                         requires good quality masks. This must be selected along the
↪'masking' option.
                         *** Specify 'true' or 'false'.
                        * template_registration: Specify a registration script for the␣
↪alignment of the
                         dataset-generated unbiased template to a reference template for␣
↪masking.
                        *** Rigid: conducts only rigid registration.
                        *** Affine: conducts Rigid then Affine registration.
                        *** SyN: conducts Rigid, Affine then non-linear registration.
                        *** no_reg: skip registration.
                        (default: apply=false,masking=false,brain_extraction=false,
↪template_registration=SyN)

  --bold_inho_cor BOLD_INHO_COR
                        Same as --anat_inho_cor, but for the EPI images.
                        (default: method=Rigid,otsu_thresh=2,multiotsu=false)

  --bold_robust_inho_cor BOLD_ROBUST_INHO_COR
                        Same as --anat_robust_inho_cor, but for the EPI images.
                        (default: apply=false,masking=false,brain_extraction=false,
↪template_registration=SyN)

  --commonspace_reg COMMONSPACE_REG
                        Specify registration options for the commonspace registration.
                        * masking: Combine masks derived from the inhomogeneity␣
↪correction step to support
                         registration during the generation of the unbiased template,␣
↪and then during template
                         registration.
                        *** Specify 'true' or 'false'.
```

(continues on next page)

(continued from previous page)

```
                        * brain_extraction: conducts brain extraction prior to template␣
↪registration based on the
                         combined masks from inhomogeneity correction. This will enhance␣
↪brain edge-matching, but
                         requires good quality masks. This must be selected along the
↪'masking' option.
                        *** Specify 'true' or 'false'.
                        * template_registration: Specify a registration script for the␣
↪alignment of the
                         dataset-generated unbiased template to the commonspace atlas.
                        *** Rigid: conducts only rigid registration.
                        *** Affine: conducts Rigid then Affine registration.
                        *** SyN: conducts Rigid, Affine then non-linear registration.
                        *** no_reg: skip registration.
                        * fast_commonspace: Skip the generation of a dataset-generated␣
↪unbiased template, and
                         instead, register each scan independently directly onto the␣
↪commonspace atlas, using the
                         template_registration. This option can be faster, but may␣
↪decrease the quality of
                         alignment between subjects.
                        *** Specify 'true' or 'false'.
                        (default: masking=false,brain_extraction=false,template_
↪registration=SyN,fast_commonspace=false)

  --inherit_unbiased_template INHERIT_UNBIASED_TEMPLATE
                        Provide a path to a previous RABIES preprocessing output folder␣
↪to inherit the unbiased template
                        generated in that previous run, as well as the registration to␣
↪the external atlas. In place of
                        conducting unbiased template generation, each scan is registered␣
↪to this pre-generated template
                        with registration parameters that are consistent with that of␣
↪the previous run. The atlas registration
                        is also inherited, and won't be conducted again.
                        By selecting this option, the following preprocessing parameters␣
↪will be overriden to enforce
                        consistency with the previous run: --anatomical_resampling, --
↪commonspace_reg, --anat_template,
                        --brain_mask, --WM_mask, --CSF_mask, --vascular_mask, --labels
                        (default: none)

  --bold2anat_coreg BOLD2ANAT_COREG
                        Specify the registration script for cross-modal alignment␣
↪between the EPI and structural
                        images. This operation is responsible for correcting EPI␣
↪susceptibility distortions.
                        * masking: With this option, the brain masks obtained from the␣
↪EPI inhomogeneity correction
                         step are used to support registration.
                        *** Specify 'true' or 'false'.
                        * brain_extraction: conducts brain extraction prior to␣
↪registration using the EPI masks from
```

(continues on next page)

(continued from previous page)

```
                        inhomogeneity correction. This will enhance brain edge-matching,
↪ but requires good quality
                        masks. This must be selected along the 'masking' option.
                       *** Specify 'true' or 'false'.
                       * registration: Specify a registration script.
                       *** Rigid: conducts only rigid registration.
                       *** Affine: conducts Rigid then Affine registration.
                       *** SyN: conducts Rigid, Affine then non-linear registration.
                       *** no_reg: skip registration.
                       (default: masking=false,brain_extraction=false,registration=SyN)


Resampling Options:
  The following options allow to resample the voxel dimensions for the preprocessed EPIs
  or for the anatomical images during registration.
  The resampling syntax must be 'dim1xdim2xdim3' (in mm), follwing the RAS axis␣
↪convention
  (dim1=Right-Left, dim2=Anterior-Posterior, dim3=Superior-Inferior). If 'inputs_defined'
  is provided instead of axis dimensions, the original dimensions are preserved.

  --nativespace_resampling NATIVESPACE_RESAMPLING
                       Can specify a resampling dimension for the nativespace fMRI␣
↪outputs.
                       (default: inputs_defined)

  --commonspace_resampling COMMONSPACE_RESAMPLING
                       Can specify a resampling dimension for the commonspace fMRI␣
↪outputs.
                       (default: inputs_defined)

  --anatomical_resampling ANATOMICAL_RESAMPLING

                       This specifies resampling dimensions for the anatomical␣
↪registration targets. By
                       default, images are resampled to isotropic resolution based on␣
↪the smallest dimension
                       among the provided anatomical images (EPI images instead if --
↪bold_only is True).
                       Increasing voxel resampling size will increase registration␣
↪speed at the cost of
                       accuracy.
                       (default: inputs_defined)


STC Options:
  Specify Slice Timing Correction (STC) info that is fed to AFNI's 3dTshift
  (https://afni.nimh.nih.gov/pub/dist/doc/program_help/3dTshift.html). The STC is applied
  in the anterior-posterior orientation, and thus RABIES assumes slices were acquired in
  this direction.

  --apply_STC          Select this option to apply the STC step.
                       (default: False)
```

(continues on next page)

(continued from previous page)

```
  --TR TR               Specify repetition time (TR) in seconds. (e.g. --TR 1.2). 'auto'␣
↪will read the TR from
                        the nifti header.
                        (default: auto)

  --tpattern {alt-z,alt-z2,seq-z,alt+z,alt+z2,seq+z}
                        Specify if interleaved ('alt') or sequential ('seq') acquisition,
↪ and specify in which
                        direction (- or +) to apply the correction. If slices were␣
↪acquired from front to back,
                        the correction should be in the negative (-) direction. If␣
↪slices were collected in an interleaved
                        order starting with the second or (second-to-last) slice, use
↪'alt+z2' or 'alt-z2'. Refer to this discussion on the
                        topic for more information https://github.com/CoBrALab/RABIES/
↪discussions/217.
                        (default: alt-z)

  --stc_axis {X,Y,Z}    Can specify over which axis of the image the STC must be applied.
↪ Generally, the correction
                        should be over the Y axis, which corresponds to the␣
↪anteroposterior axis in RAS convention.
                        (default: Y)

  --interp_method {linear,cubic,quintic,heptic,wsinc5,wsinc9,fourier}
                        Can specify the interpolation method used for STC. Polynomial␣
↪methods (e.g., linear, cubic, etc.)
                        will introduce greater autocorrelation to the interpolated␣
↪timeseries, while wsinc and fourier methods
                        will introduce less (or none). Refer to this discussion on the␣
↪topic for more information
                        https://github.com/CoBrALab/RABIES/discussions/267.
                        (default: fourier)


Template Files:
  Specify commonspace template and associated mask/label files. By default, RABIES
  provides the mouse DSURQE atlas
  https://wiki.mouseimaging.ca/display/MICePub/Mouse+Brain+Atlases.

  --anat_template ANAT_TEMPLATE
                        Anatomical file for the commonspace atlas.
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_average.
↪nii.gz)

  --brain_mask BRAIN_MASK
                        Brain mask aligned with the template.
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_mask.
↪nii.gz)

  --WM_mask WM_MASK     White matter mask aligned with the template.
```

(continues on next page)

(continued from previous page)

```
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_eroded_
↪WM_mask.nii.gz)

  --CSF_mask CSF_MASK   CSF mask aligned with the template.
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_eroded_
↪CSF_mask.nii.gz)

  --vascular_mask VASCULAR_MASK
                        Can provide a mask of major blood vessels to compute associated␣
↪nuisance timeseries.
                        The default mask was generated by applying MELODIC ICA and␣
↪selecting the resulting
                        component mapping onto major brain vessels.
                        (default: /home/docs/.local/share/rabies/vascular_mask.nii.gz)

  --labels LABELS       Labels file providing the atlas anatomical annotations.
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_labels.
↪nii.gz)
```

```
usage: rabies confound_correction [-h] [--nativespace_analysis]
                                  [--image_scaling {None,global_variance,voxelwise_
↪standardization,grand_mean_scaling,voxelwise_mean}]
                                  [--scale_variance_voxelwise]
                                  [--detrending_order {linear,quadratic}]
                                  [--conf_list [{WM_signal,CSF_signal,vascular_signal,
↪global_signal,aCompCor_percent,aCompCor_5,mot_6,mot_24} ...]]
                                  [--frame_censoring FRAME_CENSORING]
                                  [--TR TR] [--highpass HIGHPASS]
                                  [--lowpass LOWPASS]
                                  [--edge_cutoff EDGE_CUTOFF]
                                  [--smoothing_filter SMOOTHING_FILTER]
                                  [--match_number_timepoints]
                                  [--ica_aroma ICA_AROMA] [--read_datasink]
                                  [--timeseries_interval TIMESERIES_INTERVAL]
                                  [--generate_CR_null]
                                  preprocess_out output_dir

positional arguments:
  preprocess_out        path to RABIES preprocessing output directory.

  output_dir            path for confound correction output directory.


optional arguments:
  -h, --help            show this help message and exit
  --nativespace_analysis
                        Conduct confound correction and analysis in native space.
                        (default: False)

  --image_scaling {None,global_variance,voxelwise_standardization,grand_mean_scaling,
↪voxelwise_mean}
                        Select an option for scaling the image variance to match the␣
↪intensity profile of
```

(continues on next page)

(continued from previous page)

```
                        different scans and avoid biases in data variance and amplitude␣
↪estimation during analysis.
                        The variance explained from confound regression is also scaled␣
↪accordingly for later use with
                        --data_diagnosis.
                        *** None: No scaling is applied, only detrending.
                        *** global_variance: After applying confound correction, the␣
↪cleaned timeseries are scaled
                            according to the total standard deviation of all voxels, to␣
↪scale total variance to 1.
                        *** voxelwise_standardization: After applying confound␣
↪correction, each voxel is separately
                            scaled to unit variance (z-scoring).
                        *** grand_mean_scaling: Timeseries are divided by the mean␣
↪signal across voxel, and
                            multiplied by 100 to obtain percent BOLD fluctuations.
                        *** voxelwise_mean: each voxel is seperataly scaled according to␣
↪its mean intensity,
                            a method suggested with AFNI https://afni.nimh.nih.gov/afni/
↪community/board/read.php?1,161862,161864.
                            Timeseries are then multiplied by 100 to obtain percent BOLD␣
↪fluctuations.
                        (default: grand_mean_scaling)

  --scale_variance_voxelwise

                        If scaling was not applied voxelwise with voxelwise_
↪standardization or voxelwise_mean, this option
                        standardize the variance at each voxel to be equal, while␣
↪preserving to total variance of the
                        4D timeseries (i.e. voxels have same variance, but not unit␣
↪variance).
                        (default: False)

  --detrending_order {linear,quadratic}
                        Select between linear or quadratic (second-order) detrending of␣
↪voxel timeseries.
                        (default: linear)

  --conf_list [{WM_signal,CSF_signal,vascular_signal,global_signal,aCompCor_percent,
↪aCompCor_5,mot_6,mot_24} ...]
                        Select list of nuisance regressors that will be applied on voxel␣
↪timeseries, i.e., confound
                        regression.
                        *** WM/CSF/vascular/global_signal: correspond to mean signal␣
↪from WM/CSF/vascular/brain
                            masks.
                        *** mot_6: 6 rigid head motion correction parameters.
                        *** mot_24: mot_6 + their temporal derivative, then all 12␣
↪parameters squared, as in
                            Friston et al. (1996, Magnetic Resonance in Medicine).
                        *** aCompCor_percent: method from Muschelli et al. (2014,␣
↪Neuroimage), where component timeseries
```

(continues on next page)

(continued from previous page)

```
                        are obtained using PCA, conducted on the combined WM and CSF␣
↪masks voxel timeseries.
                        Components adding up to 50 percent of the variance are␣
↪included.
                      *** aCompCor_5: aCompCor method, but taking 5 first principal␣
↪components.
                      (default: [])

  --frame_censoring FRAME_CENSORING
                      Censor frames that are highly corrupted (i.e. 'scrubbing').
                      * FD_censoring: Apply frame censoring based on a framewise␣
↪displacement threshold. The frames
                       that exceed the given threshold, together with 1 back and 2␣
↪forward frames will be masked
                       out, as in Power et al. (2012, Neuroimage).
                      *** Specify 'true' or 'false'.
                      * FD_threshold: the FD threshold in mm.
                      * DVARS_censoring: Will remove timepoints that present outlier␣
↪values on the DVARS metric
                       (temporal derivative of global signal). This method will censor␣
↪timepoints until the
                       distribution of DVARS values across time does not contain␣
↪outliers values above or below 2.5
                       standard deviations.
                      *** Specify 'true' or 'false'.
                      * minimum_timepoint: Can set a minimum number of timepoints␣
↪remaining after frame censoring.
                       If the threshold is not met, an empty file is generated and the␣
↪scan is not considered in
                       further steps.
                      (default: FD_censoring=false,FD_threshold=0.05,DVARS_
↪censoring=false,minimum_timepoint=3)

  --TR TR             Specify repetition time (TR) in seconds. (e.g. --TR 1.2). 'auto'␣
↪will read the TR from
                      the nifti header.
                      (default: auto)

  --highpass HIGHPASS Specify highpass filter frequency.
                      (default: None)

  --lowpass LOWPASS   Specify lowpass filter frequency.
                      (default: None)

  --edge_cutoff EDGE_CUTOFF
                      Specify the number of seconds to cut at beginning and end of␣
↪acquisition if applying a
                      frequency filter. Highpass filters generate edge effects at␣
↪begining and end of the
                      timeseries. We recommend to cut those timepoints (around 30sec␣
↪at both end for 0.01Hz
                      highpass.).
```

(continues on next page)

(continued from previous page)

```
                        (default: 0)

  --smoothing_filter SMOOTHING_FILTER
                        Specify filter size in mm for spatial smoothing. Will apply␣
↪nilearn's function
                        https://nilearn.github.io/modules/generated/nilearn.image.smooth_
↪img.html
                        (default: None)

  --match_number_timepoints
                        With this option, only a subset of the timepoints are kept post-
↪censoring to match the
                        --minimum_timepoint number for all scans. This can be conducted␣
↪to avoid inconsistent
                        temporal degrees of freedom (tDOF) between scans during␣
↪downstream analysis. We recommend
                        selecting this option if a significant confounding effect of␣
↪tDOF is detected during --data_diagnosis.
                        The extra timepoints removed are randomly selected among the set␣
↪available post-censoring.
                        (default: False)

  --ica_aroma ICA_AROMA
                        Apply ICA-AROMA denoising (Pruim et al. 2015). The original␣
↪classifier was modified to incorporate
                        rodent-adapted masks and classification hyperparameters.
                        * apply: apply the denoising.
                        *** Specify 'true' or 'false'.
                        * dim: Specify a pre-determined number of MELODIC components to␣
↪derive. '0' will use an automatic
                         estimator.
                        * random_seed: For reproducibility, this option sets a fixed␣
↪random seed for MELODIC.
                        (default: apply=false,dim=0,random_seed=1)

  --read_datasink
                        Choose this option to read preprocessing outputs from datasinks␣
↪instead of the saved
                        preprocessing workflow graph. This allows to run confound␣
↪correction without having
                        available RABIES preprocessing folders, but the targetted␣
↪datasink folders must follow the
                        structure of RABIES preprocessing.
                        (default: False)

  --timeseries_interval TIMESERIES_INTERVAL
                        Before confound correction, can crop the timeseries within a␣
↪specific interval.
                        e.g. '0,80' for timepoint 0 to 80.
                        (default: all)

  --generate_CR_null    Estimate overfitting from confound regression by generating␣
↪phase randomised regressors,
```

(continues on next page)

(continued from previous page)

```
                        following the method by Bright and Murphy (2015), NeuroImage. By␣
↪selecting this option,
                        an additional figure will be generated to display the variance␣
↪explained by the real
                        regressors VS the randomized regressors to assess overfitting.
                        (default: False)
```

```
usage: rabies analysis [-h] [--prior_maps PRIOR_MAPS]
                       [--prior_bold_idx [PRIOR_BOLD_IDX ...]]
                       [--prior_confound_idx [PRIOR_CONFOUND_IDX ...]]
                       [--data_diagnosis]
                       [--scan_QC_thresholds SCAN_QC_THRESHOLDS]
                       [--outlier_threshold OUTLIER_THRESHOLD] [--extended_QC]
                       [--seed_list [SEED_LIST ...]]
                       [--seed_prior_list [SEED_PRIOR_LIST ...]] [--FC_matrix]
                       [--ROI_type {parcellated,voxelwise}]
                       [--ROI_csv ROI_CSV] [--group_ica GROUP_ICA] [--DR_ICA]
                       [--NPR_temporal_comp NPR_TEMPORAL_COMP]
                       [--NPR_spatial_comp NPR_SPATIAL_COMP]
                       [--optimize_NPR OPTIMIZE_NPR]
                       [--network_weighting {absolute,relative}]
                       confound_correction_out output_dir

positional arguments:
  confound_correction_out
                        path to RABIES confound correction output directory.

  output_dir            path for analysis outputs.


optional arguments:
  -h, --help            show this help message and exit
  --prior_maps PRIOR_MAPS
                        Provide a 4D nifti image with a series of spatial priors␣
↪representing common sources of
                        signal (e.g. ICA components from a group-ICA run). This 4D prior␣
↪map file will be used for
                        Dual regression, Dual ICA and --data_diagnosis. The RABIES␣
↪default corresponds to a MELODIC
                        run on a combined group of anesthetized-ventilated and awake␣
↪mice. Confound correction
                        consisted of highpass at 0.01 Hz, FD censoring at 0.03mm, DVARS␣
↪censoring, and
                        mot_6,WM_signal,CSF_signal as regressors.
                        (default: /home/docs/.local/share/rabies/melodic_IC.nii.gz)

  --prior_bold_idx [PRIOR_BOLD_IDX ...]
                        Specify the indices for the priors corresponding to BOLD sources␣
↪of interest from --prior_maps.
                        This will determine the set of networks analyzed for --data_
↪diagnosis.
                        IMPORTANT: index counting starts at 0 (i.e. the first component␣
↪is selected with 0, not 1)
```

(continues on next page)

(continued from previous page)

```
                        (default: [5, 12, 19])

  --prior_confound_idx [PRIOR_CONFOUND_IDX ...]
                        Specify the indices for the confound components from --prior_
↪maps. This is pertinent for
                        evaluating features in the --data_diagnosis outputs.
                        IMPORTANT: index counting starts at 0 (i.e. the first component␣
↪is selected with 0, not 1)
                        (default: [0, 2, 6, 7, 8, 9, 10, 11, 13, 14, 21, 22, 24, 26, 28,␣
↪29])

  --data_diagnosis      Generates a set of data quality assessment reports as described␣
↪in Desrosiers-Gregoire et al. 2023.
                        These reports aim to support interpreting connectivity results␣
↪and account for data quality issues,
                        and include: 1-a scan-level qualitative diagnosis report, 2-a␣
↪quantitative distribution report, 3-a
                        group-level statistical report for network analysis.
                        (default: False)

  --scan_QC_thresholds SCAN_QC_THRESHOLDS
                        Option to specify scan-level thresholds to remove scans from the␣
↪dataset QC report.
                        This can be specified for a given set of network analyses among␣
↪DR (dual regression), SBC (seed
                        connectivity), or NPR. For each analysis, the following QC␣
↪parameters can be specified:
                        * Dice: Threshold for the minimum network detectability computed␣
↪as Dice overlap with the prior.
                        *** Specify a list of thresholds between 0 and 1. The order of␣
↪thresholds provided within the list
                            will be matched to the list of networks for the␣
↪corresponding analysis (for DR/NPR, this
                            will be matched to the --prior_bold_idx list, and for SBC it␣
↪will be matched to --seed_list).
                            If the list is empty, no thresholding is applied, otherwise,␣
↪the length of the lists for the
                            thresholds and networks must match.
                        * Conf: Threshold for the maximum temporal correlation with DR␣
↪confound timecourses.
                        *** Specify a list of thresholds between 0 and 1. The same rules␣
↪as Dice are followed for specifying
                            the order of thresholds within the list.
                        * Amp: Whether to automatically remove outliers from the network␣
↪amplitude measure.
                        *** Specify 'true' or 'false'.
                        The expression for the parameters must follow a dictionary␣
↪syntax, as with this example:
                        '{DR:{Dice:[0.3],Conf:[0.25],Amp:false},SBC:{Dice:[0.3]}}'.
                        Note that the expression must be written within ' '.
                        (default: {})
```

(continues on next page)

(continued from previous page)

```
  --outlier_threshold OUTLIER_THRESHOLD
                        The modified Z-score threshold for detecting outliers during␣
↪dataset QC when using
                        --data_diagnosis. The default of 3.5 is recommended in https://
↪www.itl.nist.gov/div898/handbook/eda/section3/eda35h.htm.
                        (default: 3.5)

  --extended_QC         Select this option to output the network correlation with the␣
↪original image intensity (calculated
                        during detrending) and the BOLDsd (signal variability). These␣
↪two additional features allow to
                        further inspect potential issues of signal amplitude scaling␣
↪between scans. However, it is unclear
                        whether signal of interest (i.e. neural metabolism) contribute␣
↪to each measure, so these outputs should
                        be interpreted with caution.
                        (default: False)

  --seed_list [SEED_LIST ...]
                        Can provide a list of Nifti files providing a mask for an␣
↪anatomical seed, which will be used
                        to evaluate seed-based connectivity maps using on Pearson's r.␣
↪Each seed must consist of
                        a binary mask representing the ROI in commonspace.
                        (default: [])

  --seed_prior_list [SEED_PRIOR_LIST ...]
                        For analysis QC of seed-based FC during --data_diagnosis, prior␣
↪network maps are required for
                        each seed provided in --seed_list. Provide the list of prior␣
↪files in matching order of the
                        --seed_list arguments to match corresponding seed maps.
                        (default: [])

  --FC_matrix           Compute whole-brain connectivity matrices using Pearson's r␣
↪between ROI timeseries.
                        (default: False)

  --ROI_type {parcellated,voxelwise}
                        Define ROIs for --FC_matrix between 'parcellated' from the␣
↪provided atlas during preprocessing,
                        or 'voxelwise' to derive the correlations between every voxel.
↪(default: parcellated)

  --ROI_csv ROI_CSV     A CSV file with the ROI names matching the ROI index numbers in␣
↪the atlas labels Nifti file.
                        A copy of this file is provided along the FC matrix generated␣
↪for each subject.
                        (default: /home/docs/.local/share/rabies/DSURQE_40micron_labels.
↪nii.gz)

  --group_ica GROUP_ICA
```

(continues on next page)

(continued from previous page)

```
                        Perform group-ICA using FSL's MELODIC on the whole dataset's␣
↪cleaned timeseries.
                        Note that confound correction must have been conducted on␣
↪commonspace outputs.
                        * apply: compute group-ICA.
                        *** Specify 'true' or 'false'.
                        * dim: Specify a pre-determined number of MELODIC components to␣
↪derive. '0' will use an automatic
                         estimator.
                        * random_seed: For reproducibility, this option sets a fixed␣
↪random seed for MELODIC.
                        (default: apply=false,dim=0,random_seed=1)

  --DR_ICA              Conduct dual regression on each subject timeseries, using the␣
↪priors from --prior_maps. The
                        linear coefficients from both the first and second regressions␣
↪will be provided as outputs.
                        Requires that confound correction was conducted on commonspace␣
↪outputs.
                        (default: False)

  --NPR_temporal_comp NPR_TEMPORAL_COMP
                        Option for performing Neural Prior Recovery (NPR). Specify with␣
↪this option how many extra
                        subject-specific sources will be computed to account for non-
↪prior confounds. This options
                        specifies the number of temporal components to compute. After␣
↪computing
                        these sources, NPR will provide a fit for each prior in --prior_
↪maps indexed by --prior_bold_idx.
                        Specify at least 0 extra sources to run NPR.
                        (default: -1)

  --NPR_spatial_comp NPR_SPATIAL_COMP
                        Same as --NPR_temporal_comp, but specify how many spatial␣
↪components to compute (which are
                        additioned to the temporal components).
                        (default: -1)

  --optimize_NPR OPTIMIZE_NPR
                        This option handles the automated dimensionality estimation when␣
↪carrying out NPR. NPR will be
                        carried out iteratively while incrementing the number of non-
↪prior components fitted, until
                        convergence criteria are met (see below). A convergence report␣
↪is generated to visualize the
                        results across iterations.

                        Convergence criterion 1: Iterations continue until the␣
↪correlation between the fitted component
                        and the prior does not reach the specified minimum.
```

(continues on next page)

(continued from previous page)

```
                        Convergence Criterion 2: At each iteration, the difference␣
↪between the previous and new output
                        is evaluated (0=perfectly correlated; 1=uncorrelated). The␣
↪forming set of successive iterations
                        (within a certain window length) is evaluated, and when a set␣
↪respects the convergence threshold
                        for each iteration within the window, the iteration preceding␣
↪that window is selected as optimal
                        output. We take the iteration preceding the  window, as this␣
↪corresponds to the last iteration
                        which generated changes above threshold. The sliding-window␣
↪approach is employed to prevent
                        falling within a local minima, when further ameliorations may be␣
↪possible with further iterations.

                        When multiple priors are fitted, they are all simultaneously␣
↪subjected to the evaluation of
                        convergence, and as long as one prior fit does not meet the␣
↪thresholds, iterations continue.

                        * apply: select 'true' to apply this option. If selected, this␣
↪option overrides --NPR_spatial_comp
                         and --NPR_spatial_comp.
                        *** Specify 'true' or 'false'.
                        * window_size: Window size for criterion 2.
                        *** Must provide an integer.
                        * min_prior_corr: Threshold for criterion 1.
                        *** Must provide a float.
                        * diff_thresh: Threshold for criterion 2.
                        *** Must provide a float.
                        * max_iter: Maximum number of iterations.
                        *** Must provide an integer.
                        * compute_max: select 'true' to visualize all iterations until␣
↪max_iter in the report.
                        *** Specify 'true' or 'false'.
                        (default: apply=false,window_size=5,min_prior_corr=0.5,diff_
↪thresh=0.03,max_iter=20,compute_max=false)

  --network_weighting {absolute,relative}
                        Whether to derive absolute or relative (variance-normalized)␣
↪network maps, representing
                        respectively network amplitude + shape or network shape only.␣
↪This option applies to both
                        dual regression (DR) and Neural Prior Recovery (NPR) analyses.
                        (default: absolute)
```

7.2.3 Example execution syntax

The following section provides examples describing the basic syntax for running the RABIES command line interface.

preprocess

```
rabies -p MultiProc preprocess input_BIDS/ preprocess_outputs/ --apply_STC --TR 1.2 --
↪commonspace_reg masking=true,brain_extraction=false,template_registration=SyN,fast_
↪commonspace=false
```

First, we have to preprocess the dataset before it can be analyzed. In this example, we are running the RABIES preprocessing on the dataset found in the `input_BIDS/` folder, formatted according to the BIDS standard, and the outputs from RABIES are stored in the `preprocess_outputs/` folder. Additional execution parameters were specified:

- `-p MultiProc` will execute the pipeline in parallel using the local threads available. Notice that this parameter is specified before the processing stage, because it is one of the `Execution Options` affiliated to the `rabies --help`.
- `--apply_STC` is a boolean variable which, when selected, will apply slice timing correction during preprocessing, which is not applied by default in RABIES.
- `--TR 1.2` specifies the repetition time (TR) of the fMRI images that are processed, which must be defined to apply slice timing correction appropriately. Notice that this parameter must be provided with an argument, here `1.2` for TR = 1.2sec, and this is done by writing down the argument with a space dividing the associated parameter.
- `--commonspace_reg masking=true,brain_extraction=false,template_registration=SyN, fast_commonspace=false` this argument manages the options for the commonspace registration step. Some arguments, including `--commonspace_reg`, take multiple parameters as input, where each parameter-value pairs follow the syntax of `parameter=value`. In this case, we are using the masking optino with `masking=true`, which use available brain masks from inhomogeneity correction to drive the registration operations, and we specify a non-linear registration to the commonspace template with `template_registration=SyN`.

confound_correction

```
rabies -p MultiProc confound_correction preprocess_outputs/ confound_correction_outputs/␣
↪--conf_list WM_signal CSF_signal vascular_signal mot_6 --smoothing_filter 0.3
```

Next, after completing preprocessing, in most cases the data should be corrected for potential confounds prior to analysis. This is done in the confound correction stage, where confounds are modelled and regressed from the data. In this example we correct the preprocessed data found in the `preprocess_outputs/` folder and store the cleaned outputs in the `confound_correction_outputs/` folder. Among the range of options available for confound correction, we define in this example three parameters:

- `--conf_list` is the option to regress nuisance timeseries from the data, i.e., confound regression. This parameter takes a list as input, where each argument in the list is seperated by a space as follow `WM_signal CSF_signal mot_6`. This list defines which nuisance timeseries are going to model confounds during confound regression, in this case, the WM and CSF mean signals together with the 6 rigid realignment parameters from head motion realignment.
- `--smoothing_filter` will additionally apply Gaussian spatial smoothing, where in this case, a filter size of `0.3` mm is specified.

analysis

```
rabies -p MultiProc analysis confound_correction_outputs analysis_outputs/ --group_ica␣
↪apply=true,dim=30,random_seed=1
```

Finally, after conducting preprocessing and confound correction, certain analyses can be run within RABIES. In this case, the cleaned outputs found in `confound_correction_outputs/` are going to be analyzed, with analysis outputs found in `analysis_outputs/`. We perform a group independent component analysis (ICA) with 30 components by providing `--group_ica apply=true,dim=30,random_seed=1` to the command.

7.2.4 Execution syntax with containerized installation (Singularity and Docker)

Containers are independent computing environments which have their own dependencies installed to ensure consistent and reliable execution of the software across computing platforms. Singularity containers, as opposed to Docker, can be exported to remote high-performance computing platforms (e.g. computecanada). The main difference in execution syntax when running a container, as opposed to the examples above, is that the paths between the local environment where the data is stored must be 'linked' to the container's internal paths. All relevant directories containing data that will be used by RABIES must be related to a container internal path, and this is done using `-B` for Singularity and `-v` for Docker. See below for examples:

Singularity execution

preprocess

```
singularity run -B $PWD/input_BIDS:/input_BIDS:ro \
-B $PWD/preprocess_outputs:/preprocess_outputs/ \
/path_to_singularity_image/rabies.sif -p MultiProc preprocess /input_BIDS/ /preprocess_
↪outputs/ --apply_STC --TR 1.2 --commonspace_reg masking=true,brain_extraction=false,
↪template_registration=SyN,fast_commonspace=false
```

Singularity containers are stored in image files, for instance `rabies.sif`. `singularity run /path_to_singularity_image/rabies.sif` will execute the image, in this case the RABIES pipeline, and the same rules for the command line interface then apply as previously demonstrated. However, the container must gain access to the relevant folders for running RABIES, in this case an input folder and an output folder, and this is done with `-B`:

- `-B $PWD/input_BIDS:/input_BIDS:ro`: this argument relates the BIDS input folder found in `$PWD/input_BIDS` to an internal path to the container, which we call `/input_BIDS`. The inputs are thus accessed according to this path in the RABIES arguments with `/input_BIDS/`. the `:ro` means that the container is only provided reading permissions at this location.
- `-B $PWD/preprocess_outputs:/preprocess_outputs/`: same as with the `/input_BIDS/`, but now we are relating a desired output directory `$PWD/preprocess_outputs` to `/preprocess_outputs`, and the container has writing permissions at this path since `:ro` is not present.

confound_correction

```
singularity run -B $PWD/input_BIDS:/input_BIDS:ro \
-B $PWD/preprocess_outputs:/preprocess_outputs/ \
-B $PWD/confound_correction_outputs:/confound_correction_outputs/ \
/path_to_singularity_image/rabies.sif -p MultiProc confound_correction /preprocess_
↪outputs/ /confound_correction_outputs/ --conf_list WM_signal CSF_signal vascular_
↪signal mot_6 --smoothing_filter 0.3
```

The required paths are similarly provided for the confound correction stage. Note here that the path to `$PWD/input_BIDS` is still linked to the container, even though it is not explicitely part of the arguments during the confound correction call. This is necessary since the paths used in the preprocessing steps still need to be accessed at later stages, and there will be an error if the paths are not kept consistent across processing steps.

analysis

```
singularity run -B $PWD/input_BIDS:/input_BIDS:ro \
-B $PWD/preprocess_outputs:/preprocess_outputs/ \
-B $PWD/confound_correction_outputs:/confound_correction_outputs/ \
-B $PWD/analysis_outputs:/analysis_outputs/ \
/path_to_singularity_image/rabies.sif -p MultiProc analysis /confound_correction_outputs␣
↪/analysis_outputs/ --group_ica apply=true,dim=30,random_seed=1
```

The same logic applies at the analysis stage.

Docker execution

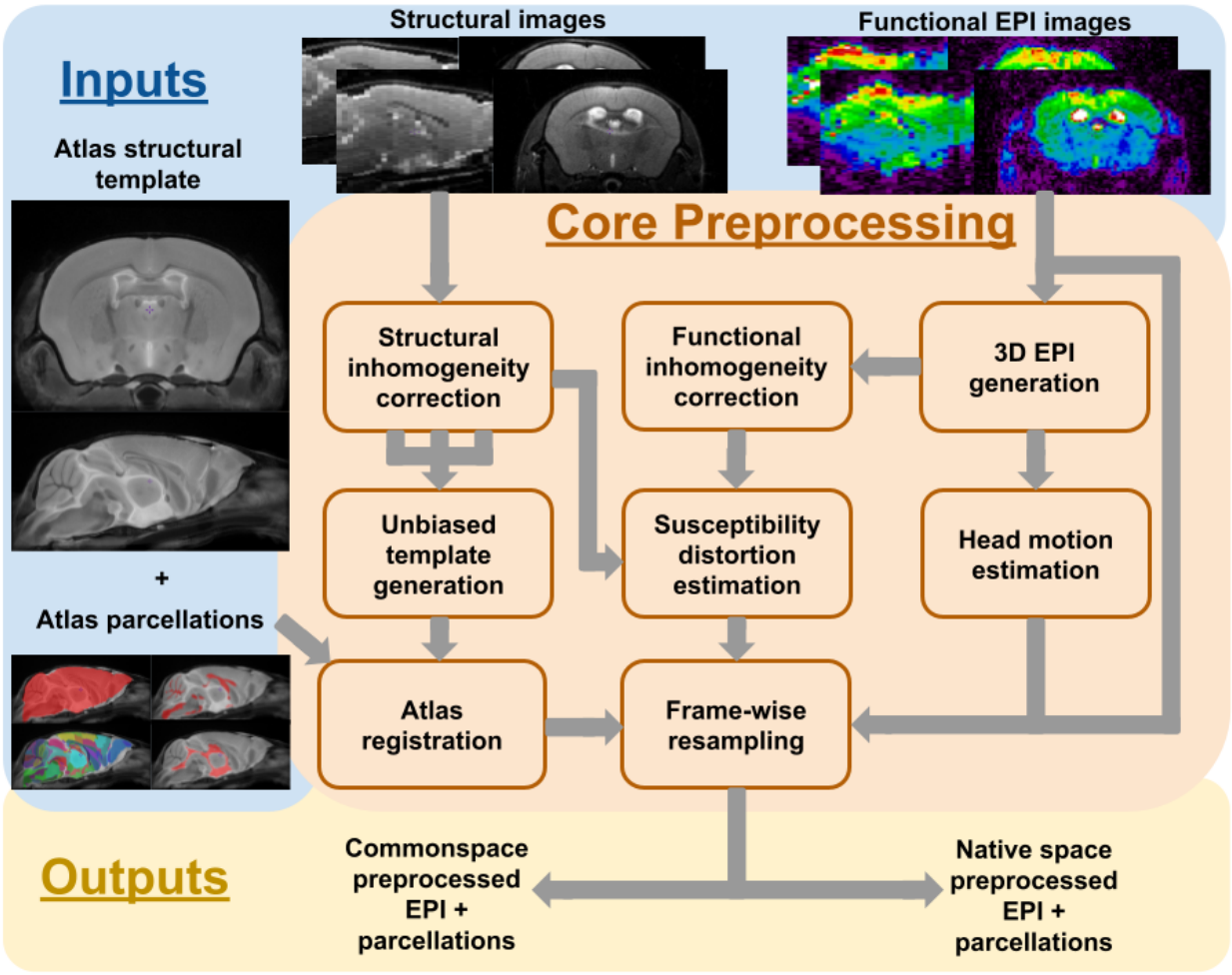
```
docker run -it --rm --user $(id -u) \
-v $PWD/input_BIDS:/input_BIDS:ro \
-v $PWD/preprocess_outputs:/preprocess_outputs/ \
gabdesgreg/rabies:tagname -p MultiProc preprocess /input_BIDS/ /preprocess_outputs/ --
↪apply_STC --TR 1.2 --commonspace_reg masking=true,brain_extraction=false,template_
↪registration=SyN,fast_commonspace=false
```

The syntax in Docker is very similar to Singularity, except that `-B` is replaced by `-v`, and further parameters may be needed (e.g. `-it`, `--rm`). `--user $(id -u)` can be added to mitigate writing permission issues when using Docker. Note that 'tagname' should be replaced by the proper RABIES version you are using (e.g. 0.4.8).

7.2.5 Additional Resources

- **Workshop and tutorial for RABIES:**
  - Hands-on tutorial on RABIES (originally developed for and presented at the INCF Neuroinformatics Assembly 2023).
  - A workshop providing a complete software overview was recorded and posted online on February 2023.
- Conversion from Bruker raw to Nifti formats can be handled with BrkRaw (consult associated documentation from the CoBrALab)
- CoBrALab recommendations for using compute canada.

7.3 Preprocessing Pipeline



The preprocessing of fMRI scans prior to analysis consists of, at minimum, the anatomical alignment of scans to a common space, head realignment to correct for motion, and the correction of susceptibility distortions arising from the echo-planar imaging (EPI) acquisition of functional scans. The core preprocessing pipeline in RABIES carries each of these steps with state-of-the-art processing tools and techniques.

To conduct common space alignment, structural images, which were acquired along the EPI scans, are initially corrected for inhomogeneities (**Structural inhomogeneity correction**) and then registered together to allow the alignment of different MRI acquisitions. This registration is conducted by generating an unbiased data-driven template (**Unbiased template generation**) through the iterative non-linear registration of each image to the dataset consensus average, where the average gets updated at each iteration to provide an increasingly representative dataset template (https://github.com/CoBrALab/optimized_antsMultivariateTemplateConstruction; [ATS+11]). The finalized template after the last iteration provides a representative alignment of each MRI session to a template that shares the acquisition properties of the dataset (e.g. brain shape, FOV, anatomical contrast, ...), making it a stable registration target for cross-subject alignment. This newly-generated unbiased template is then itself registered to an external reference atlas to provide both an anatomical segmentation and a common space comparable across studies defined from the provided reference atlas (**Atlas registration**).

The remaining preprocessing involves the EPI image. A volumetric EPI image is first derived using a trimmed mean across the EPI frames, after an initial motion realignment step (**3D EPI generation**). Using this volumetric EPI as a target, the head motion parameters are estimated by realigning each EPI frame to the target using a rigid registration

(**Head motion estimation**). To correct for EPI susceptibility distortions, the volumetric EPI is first subjected to an inhomogeneity correction step (**Functional inhomogeneity correction**), and then registered non-linearly to the anatomical scan from the same MRI session, which allows to calculate the required geometrical transforms for recovering brain anatomy [WPG+17] (**Susceptibility distortion estimation**). Finally, after calculating the transformations required to correct for head motion and susceptibility distortions, both transforms are concatenated into a single resampling operation (avoiding multiple resampling) which is applied at each EPI frame, generating the preprocessed EPI timeseries in native space [EMB+19] (**Frame-wise resampling**). Preprocessed timeseries in common space are also generated by further concatenating the transforms allowing resampling to the reference atlas.

The workflow of the RABIES preprocessing pipeline is summarized in the diagram above, and each preprocessing module is further described below.

7.3.1 Structural inhomogeneity correction

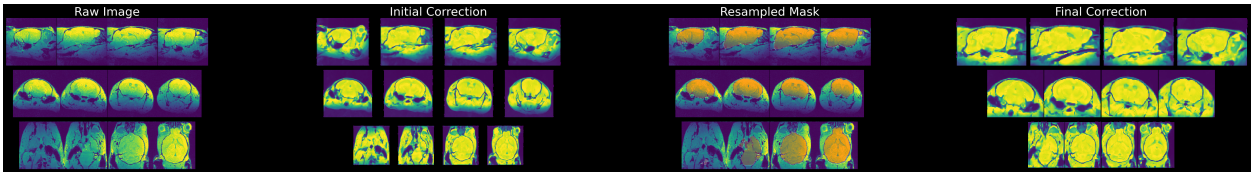


Figure: displays steps of inhomogeneity correction for the structural image.

rabies.preprocess_pkg.inho_correction.init_inho_correction_wf [source code]

```
    """
    Corrects an input 3D image for intensity inhomogeneities. The image is denoised with␣
↪non-local mean
    denoising (Manjón et al., 2010) (for EPIs, denoising was carried beforehand during␣
↪3D EPI generation)
    followed by iterative correction for intensity inhomogeneities (Sled et al., 1998).␣
↪Initial masking
    is achieved via intensity thresholding, giving an initial correction of the image,␣
↪and a registration is
    then conducted to register a brain mask for a final round of correction.

    References:
        Manjón, J. V., Coupé, P., Martí-Bonmatí, L., Collins, D. L., & Robles, M. (2010).
↪ Adaptive non-local means
            denoising of MR images with spatially varying noise levels. Journal of␣
↪Magnetic Resonance Imaging:
            JMRI, 31(1), 192-203.
        Sled, J. G., Zijdenbos, A. P., & Evans, A. C. (1998). A nonparametric method for␣
↪automatic correction of
            intensity nonuniformity in MRI data. IEEE Transactions on Medical Imaging,␣
↪17(1), 87-97.

    Command line interface parameters:
        --anat_inho_cor ANAT_INHO_COR
                                Select options for the inhomogeneity correction of the␣
↪structural image.
                                * method: specify which registration strategy is␣
↪employed for providing a brain mask.
```

(continues on next page)

(continued from previous page)

```
                                *** Rigid: conducts only rigid registration.
                                *** Affine: conducts Rigid then Affine registration.
                                *** SyN: conducts Rigid, Affine then non-linear␣
↪registration.
                                *** no_reg: skip registration.
                                *** N4_reg: previous correction script prior to version␣
↪0.3.1.
                                *** disable: disables the inhomogeneity correction.
                                * otsu_thresh: The inhomogeneity correction script␣
↪necessitates an initial correction with a
                                Otsu masking strategy (prior to registration of an␣
↪anatomical mask). This option sets the
                                Otsu threshold level to capture the right intensity␣
↪distribution.
                                *** Specify an integer among [0,1,2,3,4].
                                * multiotsu: Select this option to perform a staged␣
↪inhomogeneity correction, where only
                                lower intensities are initially corrected, then higher␣
↪intensities are iteratively
                                included to eventually correct the whole image. This␣
↪technique may help with images with
                                particularly strong inhomogeneity gradients and very low␣
↪intensities.
                                *** Specify 'true' or 'false'.
                                (default: method=SyN,otsu_thresh=2,multiotsu=false)

       --anat_robust_inho_cor ANAT_ROBUST_INHO_COR
                                When selecting this option, inhomogeneity correction is␣
↪executed twice to optimize
                                outcomes. After completing an initial inhomogeneity␣
↪correction step, the corrected outputs
                                are co-registered to generate an unbiased template,␣
↪using the same method as the commonspace
                                registration. This template is then masked, and is used␣
↪as a new target for masking during a
                                second iteration of inhomogeneity correction. Using this␣
↪dataset-specific template should
                                improve the robustness of masking for inhomogeneity␣
↪correction.
                                * apply: select 'true' to apply this option.
                                *** Specify 'true' or 'false'.
                                * masking: Combine masks derived from the inhomogeneity␣
↪correction step to support
                                registration during the generation of the unbiased␣
↪template, and then during template
                                registration.
                                *** Specify 'true' or 'false'.
                                * brain_extraction: conducts brain extraction prior to␣
↪template registration based on the
                                combined masks from inhomogeneity correction. This will␣
↪enhance brain edge-matching, but
                                requires good quality masks. This should be selected␣
↪along the 'masking' option.
```

(continues on next page)

(continued from previous page)

```
                                *** Specify 'true' or 'false'.
                                * template_registration: Specify a registration script␣
↪for the alignment of the
                                dataset-generated unbiased template to a reference␣
↪template for masking.
                                *** Rigid: conducts only rigid registration.
                                *** Affine: conducts Rigid then Affine registration.
                                *** SyN: conducts Rigid, Affine then non-linear␣
↪registration.
                                *** no_reg: skip registration.
                                (default: apply=false,masking=false,brain_
↪extraction=false,template_registration=SyN)

        --bold_inho_cor BOLD_INHO_COR
                                Same as --anat_inho_cor, but for the EPI images.
                                (default: method=Rigid,otsu_thresh=2,multiotsu=false)

        --bold_robust_inho_cor BOLD_ROBUST_INHO_COR
                                Same as --anat_robust_inho_cor, but for the EPI images.
                                (default: apply=false,masking=false,brain_
↪extraction=false,template_registration=SyN)

    Workflow:
        parameters
            opts: command line interface parameters
            image_type: between 'EPI' and 'structural'. Defines which script to run␣
↪depending on
                image type
            output_folder: specify a folder to execute the unbiased template generation␣
↪and store important outputs
            num_procs: set the maximum number of parallel threads to launch

        inputs
            target_img: the image to correct
            anat_ref: the registration target with a brain mask
            anat_mask: the brain mask of the registration target
            name_source: reference file for naming purpose
            template_anat: the structural template in for robust inhomogeneity correction
            template_mask: the brain mask for robust inhomogeneity correction

        outputs
            corrected: the output image after the final correction
            denoise_mask: the brain mask resampled on the corrected image
            init_denoise: the image after a first round of correction
    """
```

7.3.2 Common space alignment (i.e. Unbiased template generation + Atlas registration)

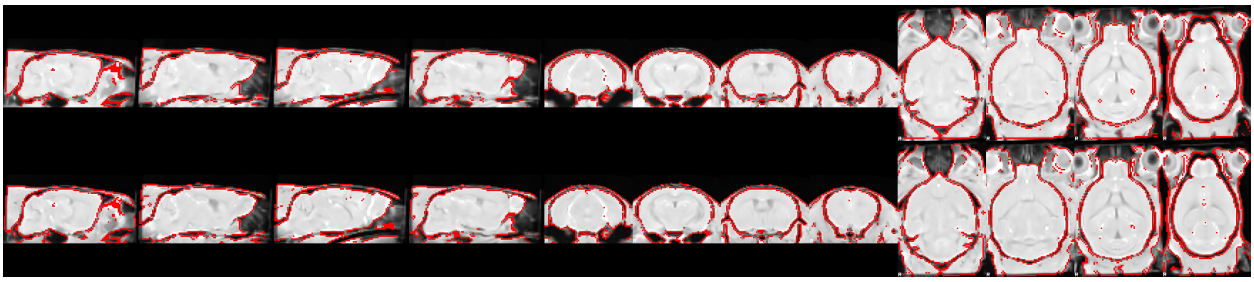


Figure: displays the overlap between a structural scan (top) and the dataset-generated unbiased template (bottom).

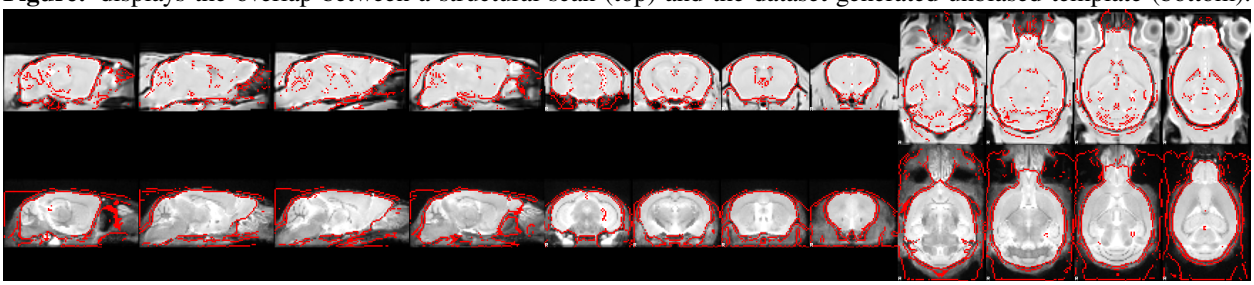


Figure: displays the overlap between the unbiased template (top) and the reference atlas template (bottom).

rabies.preprocess_pkg.commonspace_reg.init_commonspace_reg_wf [source code]

```
    """
    This workflow handles the alignment of all MRI sessions to a common space. This is␣
↪conducted first by generating
    a dataset-specific unbiased template from the input structural images, thereby␣
↪aligning the different MRI
    sessions. Through a set of iterations, images are registered to a consensus average␣
↪generated from the overlap
    of all scans at the previous iteration. Registrations are increasingly stringent,␣
↪executing 2 iterations each
    for a rigid, then affine and finally non-linear template generation. The final␣
↪iteration provides the individual
    transforms to align each scan to the unbiased template. This template generation␣
↪process creates a robust target
    for the alignment of MRI sessions sharing the same acquisition properties, which␣
↪will minimize registration
    inconsistencies between sessions, as opposed to the direct registration to an␣
↪external template. The algorithm is
    implemented in https://github.com/CoBrALab/optimized_
↪antsMultivariateTemplateConstruction.
    After generating the unbiased template, the template itself is registered with a non-
↪linear registration to the
    reference atlas in common space, providing transforms to common space and the␣
↪associated brain parcellations.


    References:
```

(continues on next page)

(continued from previous page)

```
        Avants, B. B., Tustison, N. J., Song, G., Cook, P. A., Klein, A., & Gee, J. C.␣
↪(2011). A reproducible evaluation
        of ANTs similarity metric performance in brain image registration. NeuroImage,␣
↪54(3), 2033-2044.

    Command line interface parameters:
        --commonspace_reg COMMONSPACE_REG
                              Specify registration options for the commonspace␣
↪registration.
                              * masking: Combine masks derived from the inhomogeneity␣
↪correction step to support
                              registration during the generation of the unbiased␣
↪template, and then during template
                              registration.
                              *** Specify 'true' or 'false'.
                              * brain_extraction: conducts brain extraction prior to␣
↪template registration based on the
                              combined masks from inhomogeneity correction. This will␣
↪enhance brain edge-matching, but
                              requires good quality masks. This should be selected␣
↪along the 'masking' option.
                              *** Specify 'true' or 'false'.
                              * template_registration: Specify a registration script␣
↪for the alignment of the
                              dataset-generated unbiased template to the commonspace␣
↪atlas.
                              *** Rigid: conducts only rigid registration.
                              *** Affine: conducts Rigid then Affine registration.
                              *** SyN: conducts Rigid, Affine then non-linear␣
↪registration.
                              *** no_reg: skip registration.
                              * fast_commonspace: Skip the generation of a dataset-
↪generated unbiased template, and
                              instead, register each scan independently directly onto␣
↪the commonspace atlas, using the
                              template_registration. This option can be faster, but␣
↪may decrease the quality of
                              alignment between subjects.
                              *** Specify 'true' or 'false'.
                              (default: masking=false,brain_extraction=false,template_
↪registration=SyN,fast_commonspace=false)

    Workflow:
        parameters
            opts: command line interface parameters
            commonspace_masking: whether masking is applied during template generation␣
↪and registration
            brain_extraction: whether brain extraction is applied for template␣
↪registration
            template_reg: registration method
            fast_commonspace: whether the template generation step is skipped and␣
↪instead each scan is registered directly in commonspace
```

(continues on next page)

(continued from previous page)

```
            output_folder: specify a folder to execute the workflow and store important␣
↪outputs
            transforms_datasink: datasink node where the transforms are stored
            num_procs: set the maximum number of parallel threads to launch
            output_datasinks: whether to generate a datasink from the outputs of the␣
↪workflow
            joinsource_list: names for the iterable nodes to join before unbiased␣
↪template generation

        inputs
            moving_image_list: list of files corresponding to the images from different␣
↪MRI sessions
            moving_mask_list: mask files overlapping with the moving images, inherited␣
↪from the inhomogeneity
                correction step. These masks are used for --commonspace_masking and --
↪brain_extraction
            template_anat: the target structural template to register the unbiased␣
↪template
            template_mask: the brain mask of the structural template

        outputs
            unbiased_template: the generated unbiased template
            unbiased_mask: brain mask resampled over the unbiased template
            native_mask: the atlas brain mask resampled to an associated MRI session in␣
↪native space
            to_atlas_affine: affine transform for registration to the atlas
            to_atlas_warp: non-linear transform for registration to the atlas
            to_atlas_inverse_warp: inverse of the non-linear transform for registration␣
↪to the atlas
            native_to_unbiased_affine: affine transform from native space to the␣
↪unbiased template
            native_to_unbiased_warp: non-linear transform from native space to the␣
↪unbiased template
            native_to_unbiased_inverse_warp: inverse of the non-linear transform from␣
↪native space to the unbiased template
            native_to_commonspace_transform_list: ordered list of the transforms to␣
↪apply to move from the
                native space to the common space
            native_to_commonspace_inverse_list: list defining whether the inverse of␣
↪affine transforms should
                be applied for native_to_commonspace_transform_list
            commonspace_to_native_transform_list: ordered list of the transforms to␣
↪apply to move from the
                common space to the natiev space
            commonspace_to_native_inverse_list: list defining whether the inverse of␣
↪affine transforms should
                be applied for commonspace_to_native_transform_list
    """
```

7.3.3 3D EPI generation

rabies.preprocess_pkg.bold_ref.init_bold_reference_wf [source code]

```
    """
    The 4D raw EPI file is used to generate a representative volumetric 3D EPI. This␣
↪volume later becomes the target for
    motion realignment and the estimation of susceptibility distortions through␣
↪registration to the structural image.
    Two iterations of motion realignment to an initial median of the volumes are␣
↪conducted, then a trimmed mean is
    computed on the realignment volumes, ignoring 5% extreme, and this average becomes␣
↪the reference image. The final
    image is then corrected using non-local means denoising (Manjón et al., 2010).

    References:
        Manjón, J. V., Coupé, P., Martí-Bonmatí, L., Collins, D. L., & Robles, M. (2010).
↪ Adaptive non-local means
            denoising of MR images with spatially varying noise levels. Journal of␣
↪Magnetic Resonance Imaging:
            JMRI, 31(1), 192-203.

    Command line interface parameters:
        --detect_dummy        Detect and remove initial dummy volumes from the EPI, and␣
↪generate a reference EPI based on
                              these volumes if detected. Dummy volumes will be removed␣
↪from the output preprocessed EPI.
                              (default: False)

    Workflow:
        parameters
            opts: command line interface parameters

        inputs
            bold_file: Nifti file with EPI timeseries

        outputs
            ref_image: the reference EPI volume
            bold_file: the input EPI timeseries, but after removing dummy volumes if --
↪detect_dummy is selected
    """
```

7.3.4 Head motion estimation

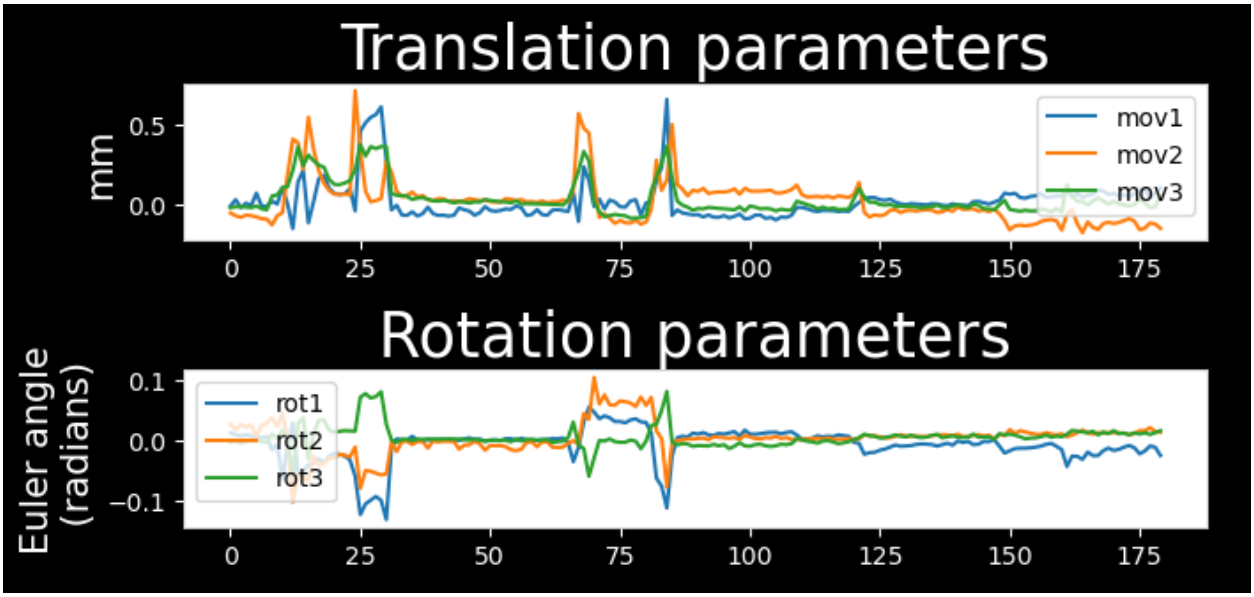


Figure: example of the 6 motion parameters.

rabies.preprocess_pkg.hmc.init_bold_hmc_wf [source code]

```
    """
    This workflow estimates motion during fMRI acquisition. To do so, each EPI frame is␣
↪registered to a volumetric
    target reference image with a rigid registration using ANTs' antsMotionCorr algorithm␣
↪(Avants et al., 2009).
    This results in the measurement of 3 Euler angles in radians and 3 translations in␣
↪mm (from ITK's
    Euler3DTransform https://itk.org/Doxygen/html/classitk_1_1Euler3DTransform.html) at␣
↪each time frame, which are
    then stored into an output CSV file.

    References:
        Avants, B. B., Tustison, N., & Song, G. (2009). Advanced normalization tools␣
↪(ANTS). The Insight Journal, 2, 1-35.

    Command line interface parameters:
        --HMC_option {intraSubjectBOLD,0,1,2,3}
                              Select an option for head motion realignment among the␣
↪pre-built options from
                              https://github.com/ANTsX/ANTsR/blob/master/R/ants_motion_
↪estimation.R.
                              (default: intraSubjectBOLD)

        --apply_slice_mc      Whether to apply a slice-specific motion correction after␣
↪initial volumetric HMC. This can
                              correct for interslice misalignment resulting from␣
↪within-TR motion. With this option,
```

(continues on next page)

(continued from previous page)

```
                                motion corrections and the subsequent resampling from␣
↪registration are applied sequentially
                                since the 2D slice registrations cannot be concatenate␣
↪with 3D transforms.
                                (default: False)

    Workflow:
        parameters
            opts: command line interface parameters

        inputs
            bold_file: Nifti file with EPI timeseries to realign
            ref_image: the 3D image target for realignment

        outputs
            motcorr_params: CSV file which contains all translation and rotation␣
↪parameters
            slice_corrected_bold: if using the experimental method --apply_slice_mc,␣
↪these are the EPI frames
                after both rigid and then slice-specific realignment
    """
```

rabies.preprocess_pkg.hmc.EstimateMotionParams [source code]

```
    """
    This interface generates estimations of absolute displacement and framewise␣
↪displacement, together with
    the expansion of the 6 motion parameters to include derivatives and squared␣
↪parameters (Friston 24).
    Absolute and framewise displacement are computed within antsMotionCorrStats as␣
↪follows:
        1. For each timepoint, the 3 Euler rotations and translations are converted to␣
↪an affine matrix
        2. For each voxel within a brain mask representing the referential space post-
↪motion realignment,
            the inverse transform is applied to generate a point pre-motion realignment.
        3. Absolute displacement is computed as the distance between the referential␣
↪point post-correction
            and the point pre-correction generated from the affine. For framewise␣
↪displacement, the
            distance is measured between the pre-correction points generated from the␣
↪current and the
            next timeframes. Distance is measured in mm with the Euclidean distance.
        4. From the distance measurements, voxelwise 4D timeseries are generated, and␣
↪for framewise
            displacement, the mean and max displacement at each timeframe is stored in a␣
↪CSV file.
    """
```

7.3.5 Functional inhomogeneity correction

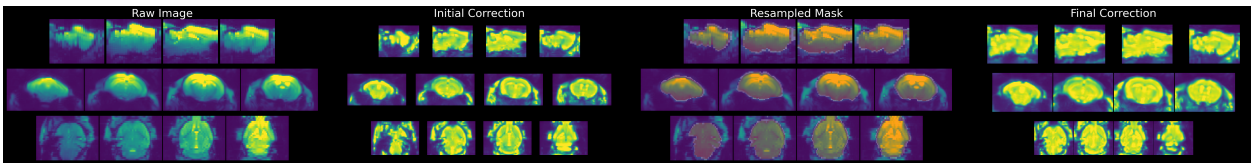


Figure: displays steps of inhomogeneity correction for the volumetric EPI.

The workflow is the same as the **structural inhomogeneity correction**.

7.3.6 Susceptibility distortion estimation

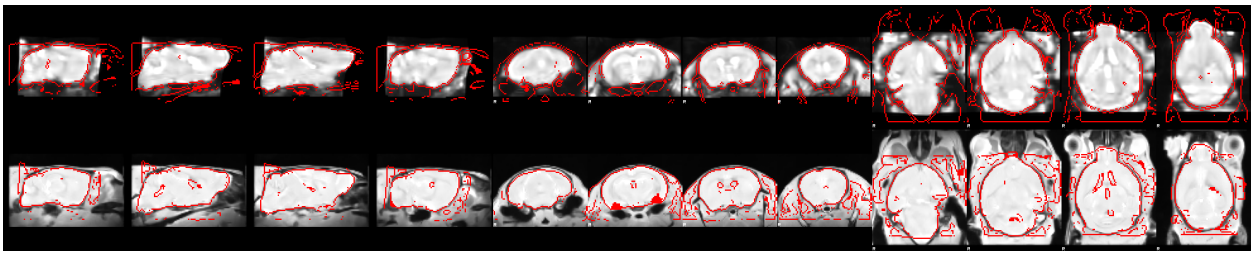


Figure: displays the overlap between the volumetric EPI (top) and structural image (bottom).

rabies.preprocess_pkg.registration.init_cross_modal_reg_wf [source code]

```
    """
    The input volumetric EPI image is registered non-linearly to an associated␣
↪structural MRI image.
    The non-linear transform estimates the correction for EPI susceptibility distortions␣
↪(Wang et al., 2017).

    References:
        Wang, S., Peterson, D. J., Gatenby, J. C., Li, W., Grabowski, T. J., &␣
↪Madhyastha, T. M. (2017).
           Evaluation of Field Map and Nonlinear Registration Methods for Correction of␣
↪Susceptibility Artifacts
           in Diffusion MRI. Frontiers in Neuroinformatics, 11, 17.

    Command line interface parameters:
        --bold2anat_coreg BOLD2ANAT_COREG
                           Specify the registration script for cross-modal alignment␣
↪between the EPI and structural
                           images. This operation is responsible for correcting EPI␣
↪susceptibility distortions.
                           * masking: With this option, the brain masks obtained from␣
↪the EPI inhomogeneity correction
                           step are used to support registration.
                           *** Specify 'true' or 'false'.
                           * brain_extraction: conducts brain extraction prior to␣
↪registration using the EPI masks from
                           inhomogeneity correction. This will enhance brain edge-
↪matching, but requires good quality
```

(continues on next page)

(continued from previous page)

```
                            masks. This should be selected along the 'masking' option.
                            *** Specify 'true' or 'false'.
                            * registration: Specify a registration script.
                            *** Rigid: conducts only rigid registration.
                            *** Affine: conducts Rigid then Affine registration.
                            *** SyN: conducts Rigid, Affine then non-linear registration.
                            *** no_reg: skip registration.
                            (default: masking=false,brain_extraction=false,
↪registration=SyN)

    Workflow:
        parameters
            opts: command line interface parameters

        inputs
            ref_bold_brain: volumetric EPI image to register
            anat_ref: the target structural image
            anat_mask: the brain mask of the structural image
            moving_mask: a EPI mask inherited from inhomogeneity correction

        outputs
            bold_to_anat_affine: affine transform from the EPI to the anatomical image
            bold_to_anat_warp: non-linear transform from the EPI to the anatomical image
            bold_to_anat_inverse_warp: inverse non-linear transform from the EPI to the␣
↪anatomical image
            output_warped_bold: the EPI image warped onto the structural image
    """
```

7.3.7 Frame-wise resampling

rabies.preprocess_pkg.resampling.init_bold_preproc_trans_wf [source code]

```
    """
    This workflow carries out the resampling of the original EPI timeseries into␣
↪preprocessed timeseries.
    This is accomplished by applying at each frame a combined transform which accounts␣
↪for previously estimated
    motion correction and susceptibility distortion correction, together with the␣
↪alignment to common space if
    the outputs are desired in common space. All transforms are concatenated into a␣
↪single resampling operation
    to mitigate interpolation effects from repeated resampling.
    This workflow also carries the resampling of brain masks and labels from the␣
↪reference atlas onto the
    preprocessed EPI timeseries.

    Command line interface parameters:
        Resampling Options:
            The following options allow to resample the voxel dimensions for the␣
↪preprocessed EPIs
```

(continues on next page)

(continued from previous page)

```
            or for the anatomical images during registration.
            The resampling syntax must be 'dim1xdim2xdim3' (in mm), follwing the RAS axis␣
↪convention
            (dim1=Right-Left, dim2=Anterior-Posterior, dim3=Superior-Inferior). If
↪'inputs_defined'
            is provided instead of axis dimensions, the original dimensions are␣
↪preserved.

        --nativespace_resampling NATIVESPACE_RESAMPLING
                            Can specify a resampling dimension for the nativespace fMRI␣
↪outputs.
                            (default: inputs_defined)

        --commonspace_resampling COMMONSPACE_RESAMPLING
                            Can specify a resampling dimension for the commonspace fMRI␣
↪outputs.
                            (default: inputs_defined)

    Workflow:
        parameters
            opts: command line interface parameters
            resampling_dim: specify the desired output voxel dimensions after resampling

        inputs
            name_source: a reference file for naming the output
            bold_file: the EPI timeseries to resample
            motcorr_params: the motion correction parameters
            transforms_list: a list of transforms to apply onto EPI timeseries,␣
↪including
                susceptibility distortion correction and resampling to common space
            inverses: a list specifying whether the inverse affine transforms should be
                applied in transforms_list
            ref_file: a reference image in the targetted space for resampling. Should be␣
↪the structural
                image from the same session if outputs are in native space, or the atlas␣
↪template for
                outputs in common space
            mask_transforms_list: the list of transforms to apply onto the atlas␣
↪parcellations
                to overlap with the EPI
            mask_inverses: a list specifying whether the inverse affine transforms␣
↪should be
                applied in mask_transforms_list

        outputs
            bold: the preprocessed EPI timeseries
            bold_ref: a volumetric 3D EPI generated from the preprocessed timeseries
            brain_mask: the brain mask resampled onto preprocessed EPI timeseries
            WM_mask: the WM mask resampled onto preprocessed EPI timeseries
            CSF_mask: the CSF mask resampled onto preprocessed EPI timeseries
            vascular_mask: the vascular mask resampled onto preprocessed EPI timeseries
            labels: the atlas labels resampled onto preprocessed EPI timeseries
```

(continues on next page)

(continued from previous page)

```
    """
```

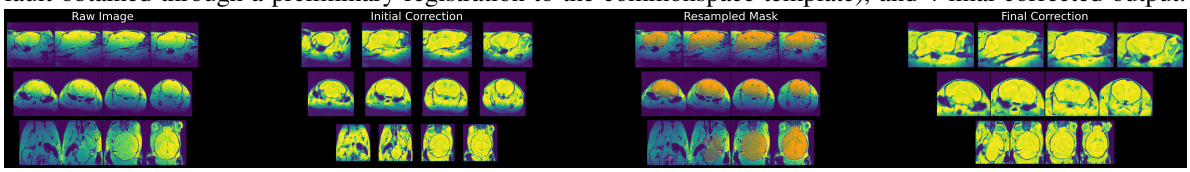
7.3.8 Adapted workflow without structural scans (i.e. –bold_only)

Structural scans are recommended, but not required to complete preprocessing with RABIES. An alternative workflow is also implemented to preprocess a input dataset which contains only EPI functional images, and can be selected with the `--bold_only` option. In this alternative workflow, the volumetric EPI corrected for inhomogeneity during **Functional inhomogeneity correction** replaces the structural image for the purpose of common space alignment, and is thus used for generating the unbiased template, in turn, this template is registered to the reference atlas. This final registration to the atlas accounts for estimation of susceptibility distortions instead of the registration to a structural image from the same MRI session. Given that this atlas registration must be applied to account for susceptibility distortions, only preprocessed timeseries in common space are provided when running this workflow. If using the RABIES default mouse atlas, the default template is changed to a EPI reference template, which offers a more robust target for EPI registration than a structural image as reference template.

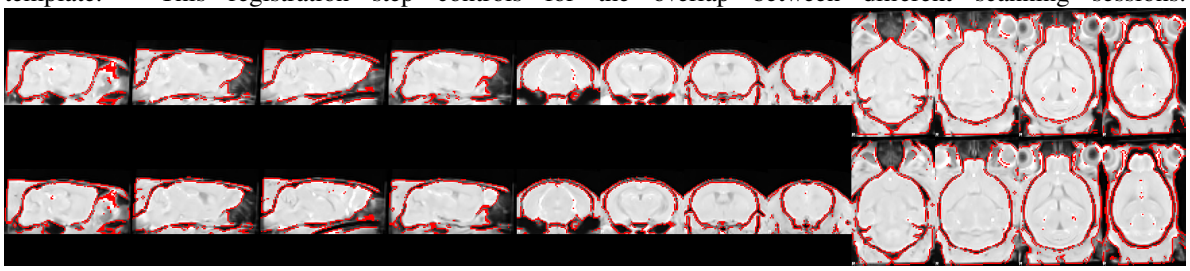
7.4 Preprocessing quality control (QC)

Several registration operations during preprocessing are prone to fail in accurately aligning images, and it is thus necessary to visually inspect the quality of registration to prevent errors arising from failed alignment, or biased analyses downstream. For this purpose, RABIES generates automatically a set PNG images allowing for efficient visual assessment of key registration steps. These are found in the `{output_folder}/preprocess_QC_report/` folder, which contains several subfolders belonging to different registration step in the pipeline or providing supportive information about the files:

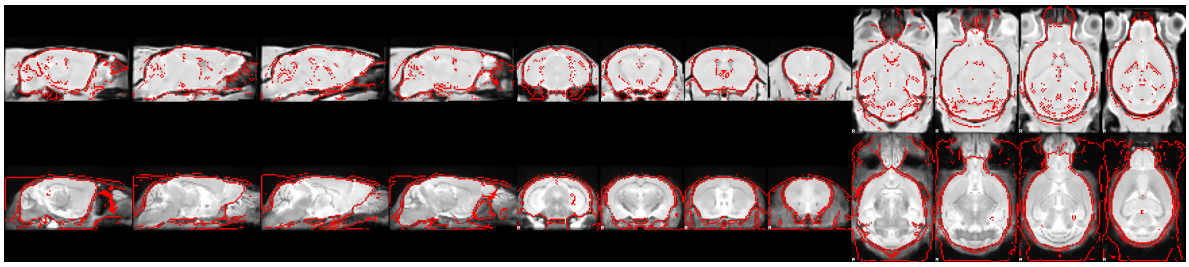
- `anat_inho_cor/`: intensity inhomogeneities are corrected for prior to important registration operations. This folder allows to assess the quality of the inhomogeneity correction, which is crucial for the performance of downstream registration. The figure is divided in 4 columns, showing 1-the raw image, 2-an initial correction of the image, 3-an overlay of the anatomical mask used to conduct a final correction (by default obtained through a preliminary registration to the commonspace template), and 4-final corrected output.



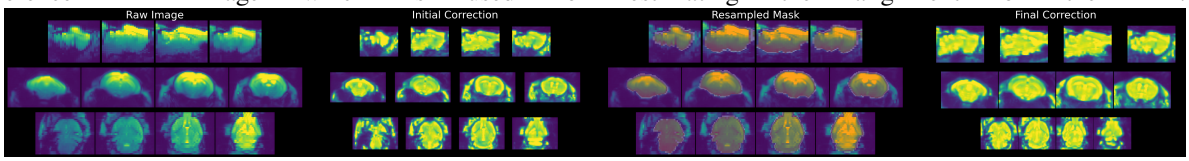
- `Native2Unbiased/`: alignment between each anatomical image and the generated unbiased template. This registration step controls for the overlap between different scanning sessions.



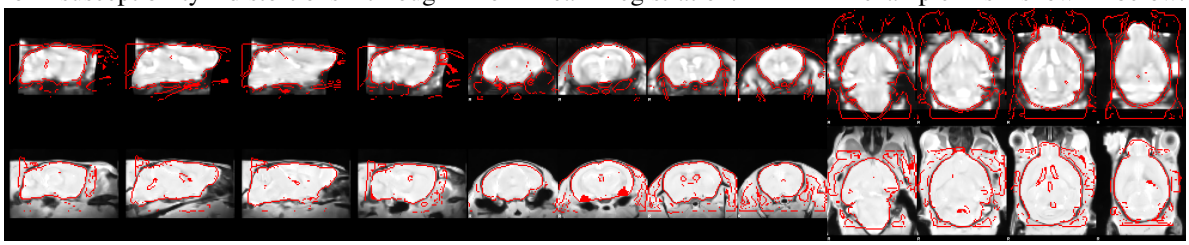
- `Unbiased2Atlas/`: alignment of the generated unbiased template to the external anatomical template in commonspace. This step ensures proper alignment with the commonspace and the associated brain parcellation.



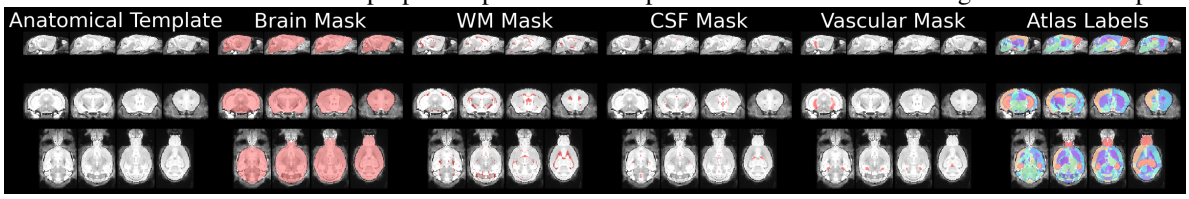
- `bold_inho_cor/`: same as `anat_inho_cor/`, but conducted on the 3D reference EPI image which is used for estimating the alignment of the EPI.



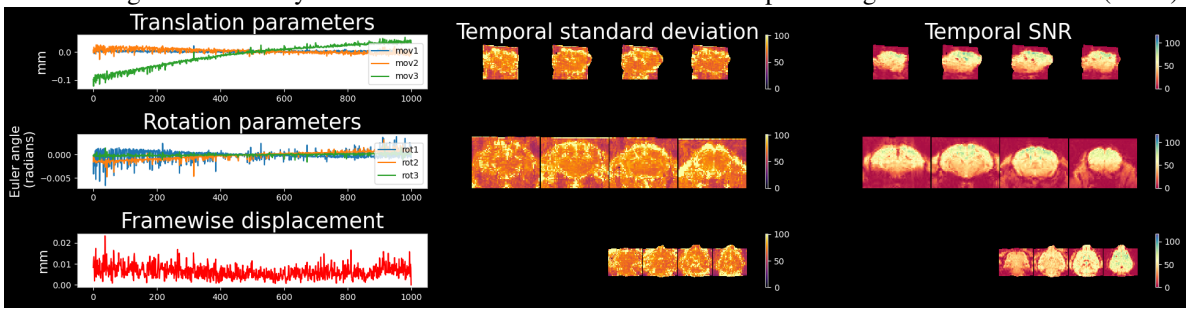
- `EPI2Anat/`: shows the alignment of the EPI image to its associated anatomical image of the same scanning session. This step resamples the EPI into native space, and corrects for susceptibility distortions through non-linear registration. An example is shown below:



- `template_files/`: displays the overlap of the provided external anatomical template with it's associated masks and labels. Allows to validate that proper template files were provided and share those along the RABIES report.



- `temporal_features/`: includes the timecourse of the head motion realignment parameters together with framewise displacement, to observe subject motion. Also includes a spatial map of the signal variability at each voxel and then the temporal signal-to-noise ratio (tSNR).



7.4.1 Recommendations for registration troubleshooting

When first attemting preprocessing with RABIES, we recommend following the default parameters as they involve less stringent modifications of the images and mostly rely on the original quality of the MR images at acquisition. However, the default parameters do not offer a generalizable robust workflow for every datasets, and to reach ideal outcomes, the workflow parameters may require tuning. We provide below recommendations for common types of registration failures that may be found from the QC report.

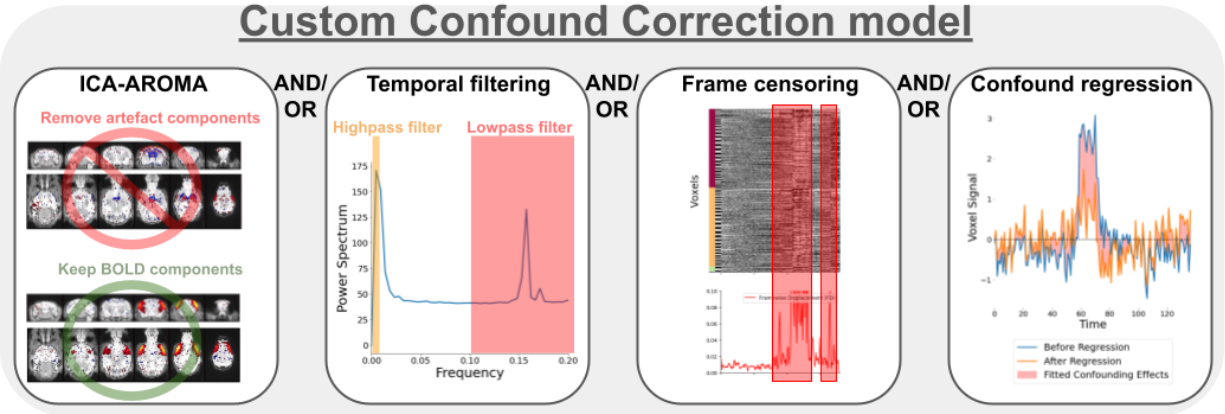
Inhomogeneity correction (anat or BOLD) `--anat_inho_cor`, `--bold_inho_cor`, `--anat_robust_inho_cor`, `--bold_robust_inho_cor`

- **Only a subset of the scans have failed masking, or the mask is partially misregistered:** Consider using the `--anat_robust_inho_cor`/`--bold_robust_inho_cor` option, which will register all corrected images to generate a temporary template representing the average of all scans, and this template is then itself masked, and becomes the new target for masking during a second iteration of inhomogeneity correction. This should provide a more robust registration target for masking. The parameters for handling this setp are the same as `--commonspace_reg` below.
- **The inhomogeneity biases are not completely corrected:** if you observe that drops in signal are still present after the connection, you should consider applying `multiotsu=true`. This option will better correct low intensities in an image with important signal drops.
- **Tissue outside the brain is provoking registration failures:** if the intensity of tissue outside the brain was enhanced during the initial inhomogeneity correction and leads to masking failures, you can consider using `--anat_autobox`/`--bold_autobox` which can automatically crop out extra tissue. You can also modify the `otsu_thresh` to set the threshold for the automatic masking during the initial correction, and attempt to select a threshold that is more specific to the brain tissue.
- **There are still a large proportion of masking failures (mismatched brain sizes or non-linear wraps, or mask outside of the brain):** Consider applying a less stringent registration `method`, going down from `SyN` -> `Affine` -> `Rigid` -> `no_reg` . If `no_reg` is selected, you may have to also adjust the `otsu_thresh` to obtain an automatically-generated brain mask covering only the brain tissues.

Commonspace registration `--commonspace_reg` or susceptibility distortion correction `--bold2anat_coreg`

- **Many scans are misregistered, or brain edges are not well-matched:** First, inspect the quality of inhomogeneity correction for those scans, and refer to instructions above if the correction or brain masking was poor. If good quality masks were obtained during inhomogeneity correction, they can be used to improve registration quality by using `masking=true`. If registration errors persist, in particular if brain edges are not well-matched, `brain_extraction=true` can be used to further constrain the matching of brain edges after removing tissue outside the brain. However, the quality of brain edge delineation depends on masks derived during inhomogeneity correction, so this option depends on high quality masking during this previous step.
- **Scans have incomplete brain coverage (e.g. cerebellum/olfactory bulbs), and surrounding brain tissue is streched to fill in missing regions:** The non-linear registration assumes corresponding brain anatomy between the moving image and the target. If brain regions are missing, the surrounding tissue may be improperly stretched to fill missing areas. Using the `brain_extraction=true` can largely mitigate this issue.

7.5 Confound Correction pipeline



The workflow for confound correction regroups a broad set of standard tools from the human litterature. The implementation of each step is structured to follow best practices and prevent re-introduction of confounds, as recommended in [PML+14] and [LGWC19]. Importantly, each operation is optional (except detrending), and a set of operations can be selected to design a customized workflow. Optimal correction strategy can be dataset-specific, and ideally, should be tuned to address relevant quality issues identified within the dataset (see section on *data quality assessment*).

1. **Frame censoring** (`--frame_censoring`): Frame censoring temporal masks are derived from FD and/or DVARS thresholds, and applied first on both BOLD timeseries before any other correction step to exclude signal spikes which may bias downstream corrections, in particular, detrending, frequency filtering and confound regression[PML+14].

    - Censoring with framewise displacement (see *definition*): Apply frame censoring based on a framewise displacement threshold. The frames that exceed the given threshold, together with 1 back and 2 forward frames will be masked out[PBS+12].
    - Censoring with DVARS (see *definition*): The DVARS values are z-scored ($DVARS_Z = \frac{DVARS - \mu}{\sigma}$, where μ is the mean DVARS across time, and σ the standard deviation), and frames with $|DVARS_Z| > 2.5$ (i.e. above 2.5 standard deviations from the mean) are removed. Z-scoring and outlier detection is repeated within the remaining frames, iteratively, until no more outlier is detected, to obtained a final set of frames post-censoring.
    - `--match_number_timepoints` : This option can be selected to constrain each scan to retain the same final number of frames, to account for downstream impacts from unequal temporal degrees of freedom (tDOF) on analysis. To do so, a pre-set final number of frames is defined with `minimum_timepoint`, and a number of extra frames remaining post-censoring (taking into account edge removal in 5)) is randomly selected and removed from the set.

2. **Detrending** (`--detrending_order`): Linear (or quadratic) trends are removed from timeseries. Detrended timeseries $\hat{Y}$ are obtained by performing ordinary-least square (OLS) linear regression,

$$\beta = OLS(X, Y)$$

$$\hat{Y} = Y - X\beta$$

    where Y is the timeseries and the predictors are $X = [intercept, time, time^2]$ ($time^2$ is included if removing quadratic trends).

3. **ICA-AROMA** (`--ica_aroma`): Cleaning of motion-related sources using the ICA-AROMA[PMvR+15] classifier. The hard-coded human priors for anatomical masking and the linear coefficients for classification were adapted from the original code to function with rodent images. ICA-AROMA is applied prior to frequency filtering to remove further effects of motion than can result in ringing after filtering[Car13, PMvR+15].

4. **Frequency filtering** (`--TR`/`--highpass`/`--lowpass`/`--edge_cutoff`):

    1. Simulating censored timepoints: frequency filtering requires particular considerations when applied after frame censoring, since conventional filters cannot handle missing data (censoring results in missing timepoints). To address this issue, we implemented a method described in [PML+14] allowing the simulation of data points while preserving the frequency composition of the data. This method relies on an adaptation of the Lomb-Scargle periodogram, which allows estimating the frequency composition of the timeseries despite missing data points, and from that estimation, missing timepoints can be simulated while preserving the frequency profile [MGG+04].

    2. Butterworth filter: Following the simulation, frequency filtering (highpass and/or lowpass) is applied using a 3rd-order Butterworth filter (scipy.signal.butter). If applying highpass, it is recommended to remove 30 seconds at each end of the timeseries using `--edge_cutoff` to account for edge artefacts following filtering[PML+14]. After frequency filtering, the temporal mask from censoring is re-applied to remove simulated timepoints.

5. **Confound regression** (`--conf_list`): For each voxel timeseries, a selected set of nuissance regressors (see *regressor options*) are modelled using OLS linear regression and their modelled contribution to the signal is removed. Regressed timeseries $\hat{Y}$ are obtained with

$$\beta = OLS(X, Y)$$

$$Y_{CR} = X\beta$$

$$\hat{Y} = Y - Y_{CR}$$

    where Y is the timeseries, X is the set of nuisance timecourses and Y_{CR} is the confound timeseries predicted from the model at each voxel (Y_{CR} is a time by voxel 2D matrix).

6. **Intensity scaling** (`--image_scaling`): Voxel intensity values should be scaled to improve comparability between scans/datasets. The following options are provided:

    - Grand mean (**recommended**): Timeseries are divided by the mean intensity across the brain, and then multiplied by 100 to obtain percent BOLD deviations from the mean. The mean intensity of each voxel is derived from the β coefficient from the intercept computed during **Detrending**.
    - Voxelwise mean: Same as grand mean, but each voxel is independently scaled by its own mean signal.
    - Global standard deviation: Timeseries are divided by the total standard deviation across all voxel timeseries.
    - Voxelwise standardization: Each voxel is divided by its standard deviation.
    - Homogenize variance voxelwise: if no scaling was already applied voxelwise (voxelwise mean or standardization), by selecting the option `--scale_variance_voxelwise`, timeseries are first scaled voxelwise by their standard deviation (yielding homogeneous variance distribution across voxels), and then re-scaled to preserve the original total standard deviation of the entire 4D timeseries (i.e. the global standard deviation does not change). Inhomogeneous variability distribution can be a *confound signature*, thus this option may downscale their impact. `--scale_variance_voxelwise` can be applied in combination with grand mean scaling.

7. **Smoothing** (`--smoothing_filter`): Timeseries are spatially smoothed using a Gaussian smoothing filter (nilearn.image.smooth_img).

7.5.1 rabies.confound_correction_pkg.confound_correction.init_confound_correction_wf [source code]

```
    """
    This workflow applies the RABIES confound correction pipeline to preprocessed EPI␣
↪timeseries. The correction steps are
    orchestrated in line with recommendations from human litterature:
    #1 - Compute and apply frame censoring mask (from FD and/or DVARS thresholds)
    #2 - If --match_number_timepoints is selected, each scan is matched to the defined␣
↪minimum_timepoint number of frames.
    #4 - Linear/Quadratic detrending of fMRI timeseries and nuisance regressors
    #4 - Apply ICA-AROMA.
    #5 - If frequency filtering and frame censoring are applied, simulate data in␣
↪censored timepoints using the Lomb-Scargle periodogram,
         as suggested in Power et al. (2014, Neuroimage), for both the fMRI timeseries␣
↪and nuisance regressors prior to filtering.
    #6 - As recommended in Lindquist et al. (2019, Human brain mapping), make the␣
↪nuisance regressors orthogonal
         to the temporal frequency filter.
    #7 - Apply highpass and/or lowpass filtering on the fMRI timeseries (with simulated␣
↪timepoints).
    #8 - Re-apply the frame censoring mask onto filtered fMRI timeseries and nuisance␣
↪regressors, taking out the
         simulated timepoints. Edge artefacts from frequency filtering can also be␣
↪removed as recommended in Power et al. (2014, Neuroimage).
    #9 - Apply confound regression using the selected nuisance regressors.
    #10 - Scaling of timeseries variance.
    #11 - Apply Gaussian spatial smoothing.

    References:
        Power, J. D., Barnes, K. A., Snyder, A. Z., Schlaggar, B. L., & Petersen, S. E.␣
↪(2012). Spurious but systematic
            correlations in functional connectivity MRI networks arise from subject␣
↪motion. Neuroimage, 59(3), 2142-2154.
        Power, J. D., Mitra, A., Laumann, T. O., Snyder, A. Z., Schlaggar, B. L., &␣
↪Petersen, S. E. (2014). Methods to detect,
            characterize, and remove motion artifact in resting state fMRI. Neuroimage,␣
↪84, 320-341.
        Lindquist, M. A., Geuter, S., Wager, T. D., & Caffo, B. S. (2019). Modular␣
↪preprocessing pipelines can reintroduce
            artifacts into fMRI data. Human brain mapping, 40(8), 2358-2376.

    Workflow:
        parameters
            cr_opts: command line interface parameters from confound_correction

        inputs
            bold_file: preprocessed EPI timeseries
            brain_mask: brain mask overlapping with EPI timeseries
            csf_mask: CSF mask overlapping with EPI timeseries
            motion_params_csv: CSV file with motion regressors
            FD_file: CSV file with the framewise displacement
```

(continues on next page)

(continued from previous page)

```
        outputs
            cleaned_path: the cleaned EPI timeseries
            aroma_out: folder with outputs from ICA-AROMA
            VE_file: variance explained (R^2) from confound regression at each voxel
            STD_file: standard deviation on the cleaned EPI timeseries
            CR_STD_file: standard deviation on the confound timeseries modelled during␣
↪confound regression
            random_CR_STD_file_path: variance fitted by random regressors during␣
↪confound regression
            corrected_CR_STD_file_path: CR_STD_file after substracting the variance␣
↪fitted by random
                regressors.
            frame_mask_file: CSV file which records which frame were censored
            CR_data_dict: dictionary object storing extra data computed during confound␣
↪correction
    """
```

7.6 Connectivity Analysis

Following the completion of the confound correction workflow, RABIES allows the estimation of resting-state connectivity using standard analyses: seed-based connectivity, whole-brain connectivity, group independent component analysis (ICA) and dual regression (DR). For each analysis (except for group-ICA), RABIES will compute individualized connectivity maps for each scan separately, which can then be exported for relevant statistical analyses (e.g. group comparison) conducted outside of RABIES.

7.6.1 Correlation-based connectivity

Correlation-based analyses rely on computing a temporal correlation between different brain regions' BOLD fluctuations to estimate their functional coupling. The assessment of connectivity using correlation requires careful a priori cleaning of confounds (see *confound correction* and *data quality assessment*), as various fMRI confounds introduce spurious correlations that won't be distinguished from neural activity.

- **Seed-based connectivity** (`--seed_list`): Seed-based connectivity is the first technique developped for the mapping of connectivity during resting state [BZYHH95]. The mean timecourse is first extracted from an anatomical seed of interest, and the correlation (Pearson's r in RABIES) between this timecourse and every other voxel is computed to obtain a correlation map, representing the 'connectivity strength' between the seed and every other brain regions.
- **Whole-brain connectivity** (`--FC_matrix`/`--ROI_type`): This technique is an extension of the seed-based connectivity technique to encompass every brain region. That is, using the anatomical parcellation provided along with the atlas during preprocessing, the seed timecourse for every parcel is first extracted, and then the cross-correlation (Pearson's r) is measured between every region pair. The correlation values are then re-organized into a whole-brain matrix representing the connectivity between every corresponding region pair.

7.6.2 ICA-based connectivity

The second analysis approach available within RABIES rely on the spatial decomposition of BOLD timeseries using ICA, which models the data as a linear combination of independent sources. In contrast with correlation-based connectivity, which models a single linear relationship between regions, the ICA framework accounts for multiple potentially overlapping sources of BOLD fluctuations, which may further separate confound contributions from connectivity estimates. To obtain individualized connectivity estimates, this analysis framework consists of first deriving ICA components at the group level to define sources, and then recovering individual-specific versions of the sources with dual regression [NSOngurB17].

- **Group ICA** (`--group_ica`): RABIES uses FSL's MELODIC ICA algorithm [BS04] to derive ICA components. For group-ICA, timeseries for all scans aligned in commonspace are concatenated to group all data, before computing the ICA decomposition, yielding

$$Y_{concat} = A\hat{S}$$

  where Y_{concat} are the concatenated timeseries, $\hat{S}$ are the set of spatial maps defining the independent sources, and A is the mixing matrix storing timecourses associated to each component.

- **Dual regression (DR)** (`--prior_maps`/`--DR_ICA`) : DR builds on the group ICA decomposition to model scan-specific versions of the group-level components, thus allowing to estimate individualized connectivity for a given brain network first identified through group-ICA [BMFS09, NSOngurB17]. DR consists of two consecutive linear regression steps, where scan-specific timecourses are first derived for each ICA component, and then a scan-specific spatial map is obtained for each component timecourse. Using multivariate linear regression (with Ordinary least square (OLS)), component timecourses are obtained with

$$\beta_{TC} = OLS(\hat{S}, Y)$$

  describing $Y = \hat{S}\beta_{TC} + \epsilon$ where Y are the scan timeseries, $\hat{S}$ are the ICA components and β_{TC} corresponds to the estimated timecourses for each component. To accurately measure connectivity amplitude in the spatial maps derived from DR, the timecourses from the first regression step must be standardized prior to the second regression[NSOngurB17]. In RABIES, timecourses are thus variance-normalized using an *L2-norm*

$$\beta^*_{TC} = \frac{\beta_{TC}}{||\beta_{TC}||_2}$$

  where $||x||_2 = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$ is the *L2-norm* of x. The normalized timecourses β^*_{TC} are then inputed into a second regression step to derive the spatial maps β_{SM} with

$$\beta_{SM} = OLS(\beta^*_{TC}, Y^T)$$

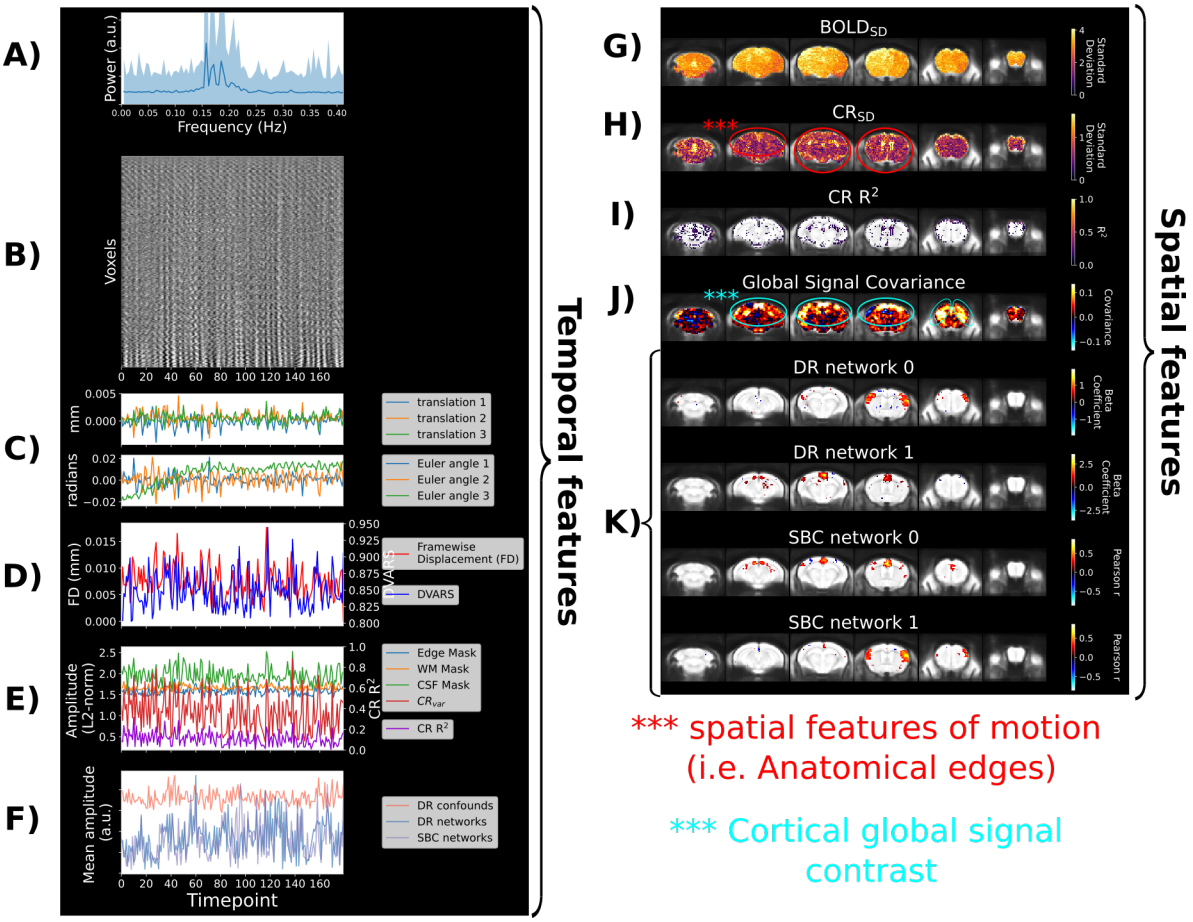
  where $Y = \beta^*_{TC}\beta_{SM} + \epsilon$, thus completing the linear model of the timeseries. The resulting scan-specific spatial maps β_{SM} will comprise information about network amplitude and shape, which may be compared across subjects or groups with further statistical tests[NSOngurB17].

7.7 Data quality assessment

7.7.1 Scan diagnosis report

By executing `--data_diagnosis` at the analysis stage of the pipeline, a set of visual reports are generated to support data quality assessment in relationship to connectivity analysis. Here, the *spatiotemporal diagnosis* report is described. The diagnosis is a visual report generated for each scan independently after conducting dual regression or seed-based connectivity analysis. It will display a large set of temporal and spatial features for the scan supporting the assessment of potential data quality issues, and whether network connectivity is impacted. Unless specified otherwise, all metrics are computed from fMRI timeseries after the confound correction stage. This page first covers an example of the report with the description for the set of features, and second provides guidance for interpreting the report.

Spatiotemporal diagnosis



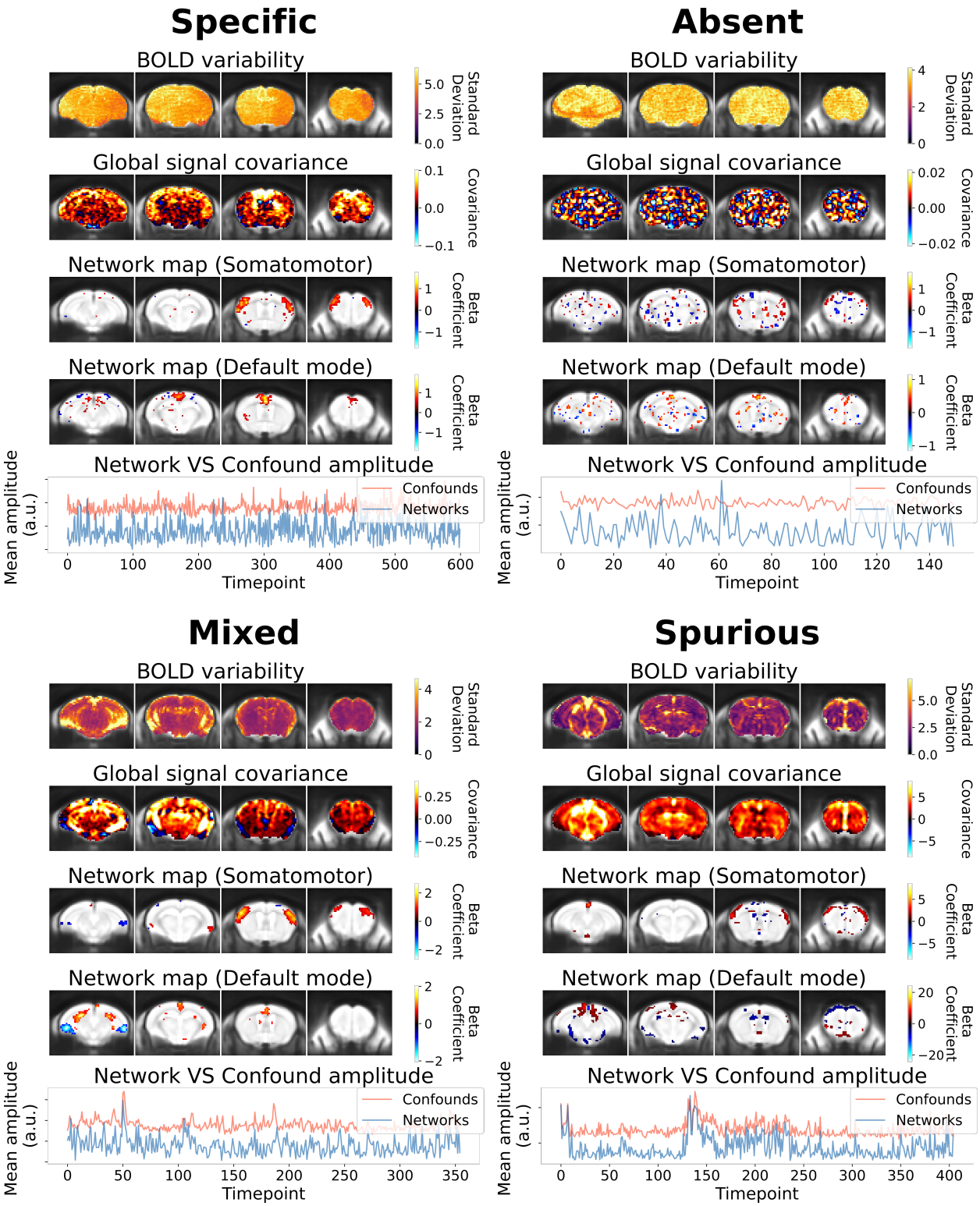
Above is an example of the report (files generated into the `data_diagnosis_datasink/figure_temporal_diagnosis/` and `data_diagnosis_datasink/figure_spatial_diagnosis/` folders) for a scan with little confound signatures and clear network connectivity. Each spatial map is represented along 6 cortical slices, overlapped onto the anatomical template in common space. The network maps from dual regression (DR) or seed-based connectivity (SBC) are thresholded to include the top 4% of the voxels with the highest values. In this example, both dual regression and seed-based connectivity was conducted, where DR network 0 and SBC network 1 correspond to analysis of the somatomotor network, whereas DR network 1 and SBC network 0 correspond to the default mode network. Below we detail the interpretation of each feature included in the diagnosis (whereas the detailed computations for each metric are further described in the *Metric definitions* page):

- **A) Power spectrum:** the frequency power spectrum (averaged across voxels) is displayed to assess the dominant frequency profile.
- **B) Carpet plot:** the entire fMRI timeseries are displayed in a time by voxel 2D matrix. This allows to visualize global fluctuations in signal intensity, which can be a proxy for various global artefacts [PPLM17].
- **C) The translation and rotation head motion parameters:** those are the 6 rigid body parameters estimated during preprocessing, and allow tracking of head position across scan duration.
- **D) The framewise displacement and the temporal shifts in global signal from the root-mean-square of the timeseries' temporal derivative (DVARS) [PBS+12]:** Framewise displacement quantifies movement between consecutive frames, which reveals the timing and amplitude of spontaneous motion, whereas DVARS reveals shifts in global fMRI signal intensities (which can also indicate suspicious spikes in signal).
- **E) Markers of confounded fMRI signal fluctuation with anatomical masks and confound regression:** A

representative timecourse is derived within a set of anatomical masks (edge, white matter and CSF masks) using the L2-norm across voxels. Each of these anatomical regions is susceptible to motion [PMvR+15], and the white matter and CSF signal can reveal physiological confounds, thus offering broader insights into potential confound sources. Furthermore, the diagnosis leverages the confound regression step, where nuisance timecourses (e.g. the 6 realignment parameters) are fitted against the fMRI signal at each voxel to obtain a modelled timecourse representing the confounded portion of the fMRI signal. To obtain an average confound timecourse across the brain, we compute the L2-norm across voxels. The proportion of variance explained by confound regression is also provided. These features allow both to visualize confound effects, and evaluate whether confound regression appropriately modelled confounds detected from other temporal features.

- **F) Mean amplitude of network VS confound timecourses:** The averaged timecourse between network analyses and confound sources are compared to assess whether network amplitude is spurious (i.e. correlated with confound timecourse). To model confound timecourses, dual regression analysis is conducted with a complete set of components from Independent Component Analysis representing a mixture of networks and confounds from various origins, and the timecourses from confound components are compiled to summarize a broad set of potential confounds (by default, RABIES this set of ICA components for mice).
- **G) Spatial distribution in signal variability (BOLDSD):** The first spatial feature of the diagnosis is the signal variability (standard deviation) at each voxel. This map offers an index of whether significant confounds are contributing to the signal (see other examples in **Interpretation of the report and main features to inspect**). Without the influence from confounds, as in this example, signal variability is largely homogeneous.
- **H) Confound regression variance explained (CRSD):** The variance explained from confound regression is quantified at each voxel by taking the standard deviation from the modelled confound timecourse. This allows to contrast spatially the amplitude of confound effects. This feature can specifically delineate the presence of confounds and identify the type of confound. In this example, minor motion signatures are identified.
- **I) Confound regression variance explained proportion:** Similar to CRSD, but showing instead the proportion of variance explained (R2).
- **J) Global signal covariance:** This map displays the covariance of each voxel with the global signal. The contrast from this map allows to evaluate the predominant source of global signal fluctuation, which can take various forms depending on the contributions from neural network and confounds (see examples below in **Interpretation of the report and main features to inspect**). In the ideal case, there is predominant contrast found in gray matter, with a shape reminescent of brain network, as in the example shown above.
- **K) Network spatial maps:** Finally, the diagnosis shows the spatial network maps fitted using dual regression (or seed-based analysis) from the selected set of brain networks of interest (in this case the somatomotor and default mode networks). These fits provide insights into the quality of network analysis, and how they may affect downstream statistical analyses.
- Note that if frame censoring was applied, the time axis is discontinued (i.e. there are holes that are not shown.)
- Note that CR_{SD} and CR_{R^2} are computing according to the specified list of regressors `--conf_list` during confound correction. If no regressor was specified, CR_{SD} and CR_{R^2} are still estimated with a regression of the 6 motion parameters, but the regression isn't applied to remove signal from the timeseries.

Interpretation of the report and main features to inspect

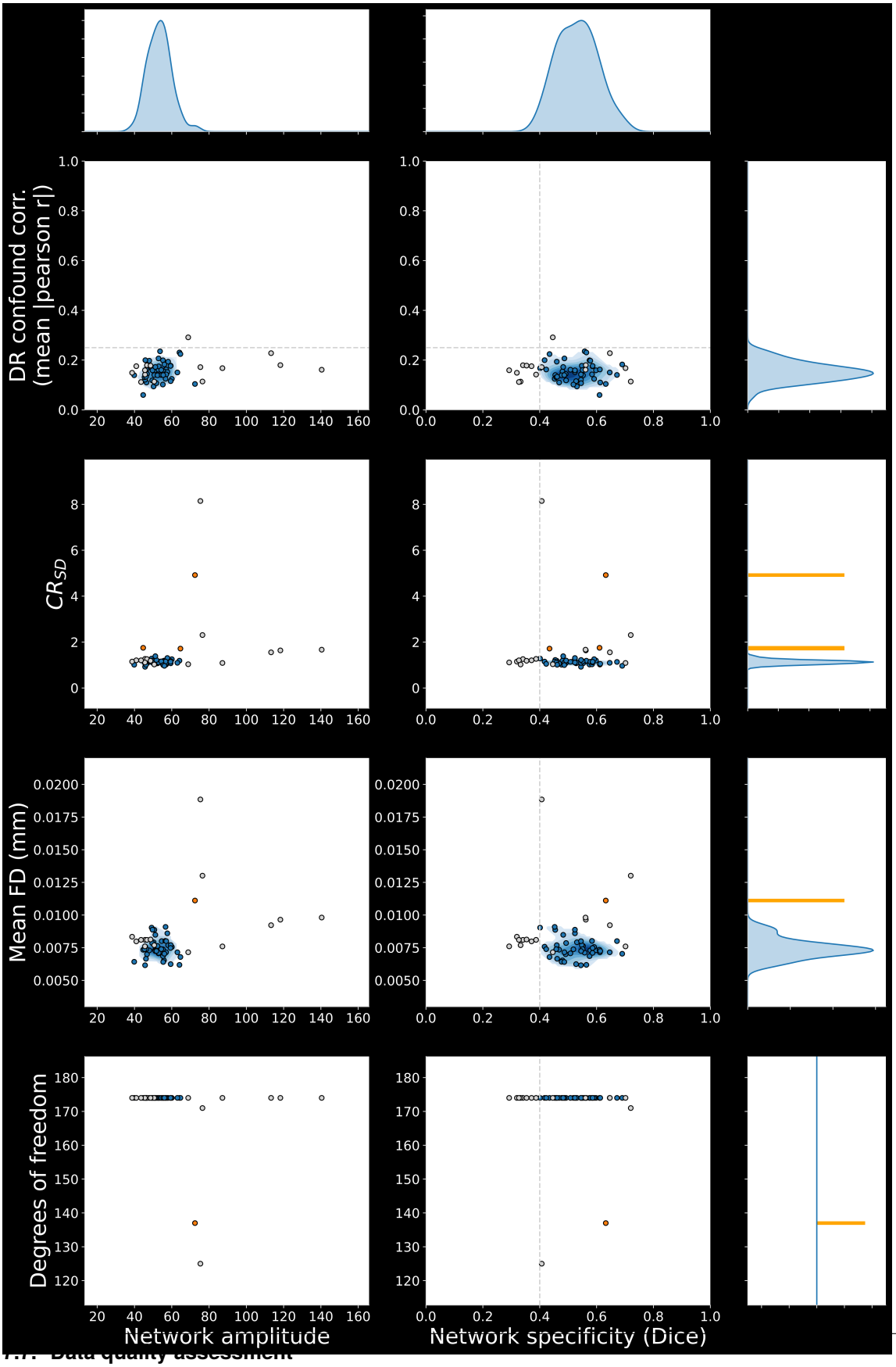


A subset of the features in the spatiotemporal diagnosis are most crucial in determining scan quality in relationship to connectivity analysis, and are displayed above across 4 main categories of scan quality. Below we describe the key role of these four features in relationship to those 4 scan categories:

- **BOLD variability:** The resulting BOLD variability map presents an homogeneous contrast in uncorrupted scans, and can otherwise reveal the anatomical signature of confounds, thus allowing to identify the type of confound.
- **Global signal covariance:** The global signal covariance map is sensitive to both non-neural confounds (e.g. the spurious category) and network signatures (e.g. the specific category). The global signal covariance thus reflects whether network or confound sources dominate coordinated fluctuations, and can delineate the most likely contributors to downstream connectivity measures.
- **Network map:** Allows inspecting whether the expected anatomical features of the network are effectively captured (i.e. network specificity). This is most crucial in ensuring that the network is not absent (see the absent category), or to ensure that the network shape is not distorded with spurious features (see spurious category).
- **Network and confound timecourses:** Finally, the respective timecourses for networks and confounds can be compared to reveal direct relationships between network amplitude and confounds in the temporal domain. Although this metric does not describe the type of confound, it is the most direct indicator of spurious connectivity. It is an important complement to the inspection of network shape, since spurious effects may only affect amplitude with minimal impact on shape.

These 4 features are sufficient to capture the essential characteristics of network detectability and spurious connectivity at the single scan level. The remaining features from the spatiotemporal diagnosis provide additional details regarding timeseries properties, the motion parameters, or confound regression, and can further support characterizing the specific origin of confounds (e.g. determining that a correlation between network and confound timecourse is originating from framewise displacement (i.e. motion)).

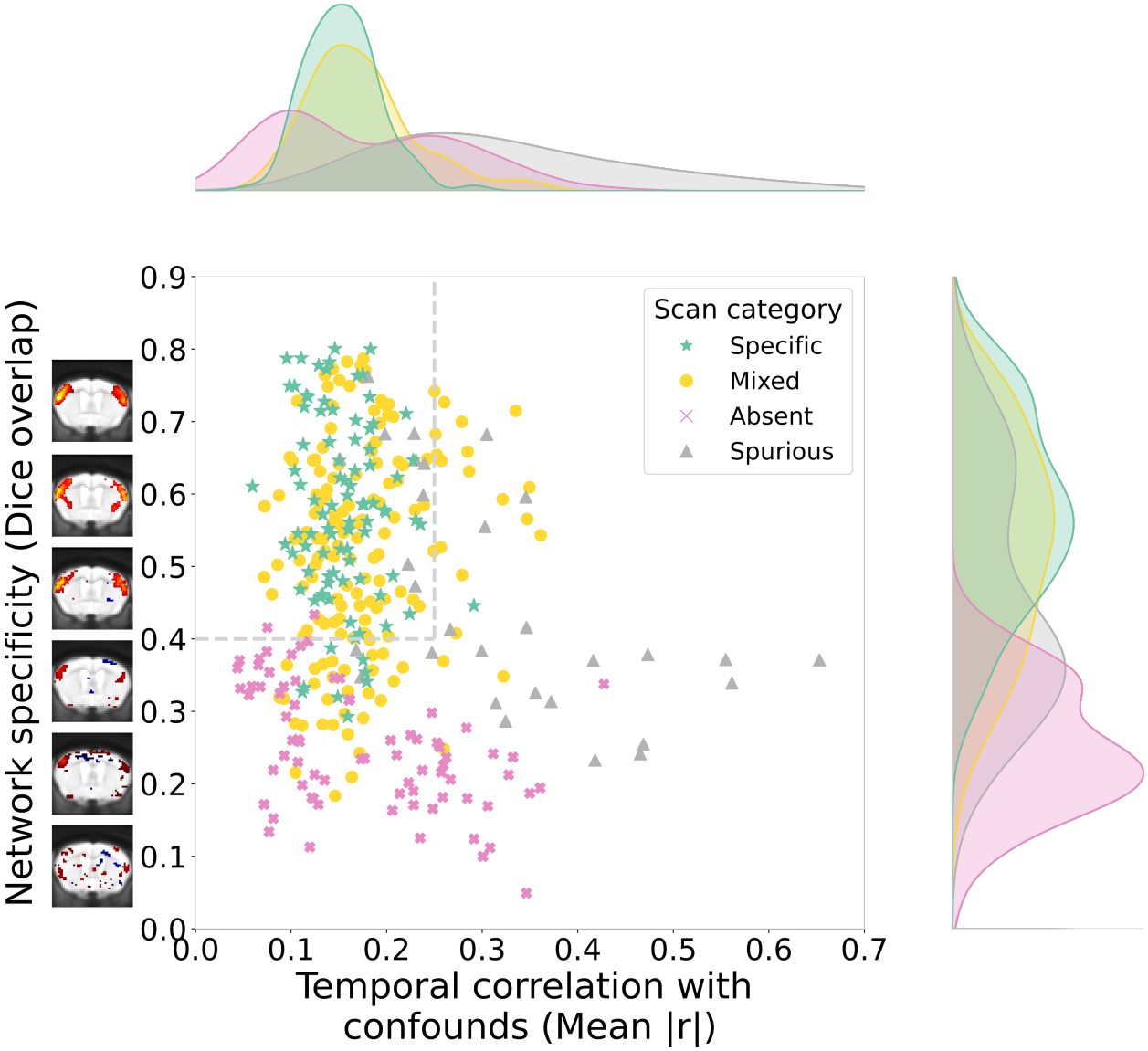
7.7.2 Distribution plot



The distribution plot allows visualizing the distribution of data quality measures across the dataset, where measures of network connectivity (specificity and amplitude) are contrasted with measures of confounds across samples (each point in the plot is a scan). Data points labelled in gray were removed using `--scan_QC_thresholds`, where the gray dotted lines correspond to the QC thresholds selected for network specificity (Dice overlap) and DR confound correlation. Among the remaining samples and for each metric separately, scan presenting outlier values were detected based on a modified Z-score threshold (set with `--outlier_threshold`, 3.5 by default) and labelled in orange. The derivation of the quality metrics is described in details on the *metrics documentation*.

The report was designed to subserve two main functions: 1. Inspect that network specificity is sufficient and the temporal correlation with confounds (i.e. DR confound corr.) minimal, and set thresholds for scan inclusion using `--scan_QC_thresholds` (top right subplot, more details on this below), and 2. complement the group statistical report to visualize the association between connectivity and the three confound measures included in the report (CR_{SD}, mean FD and tDOF). In the later case, it can be possible for instance to determine whether a group-wise correlation in statistical report is driven by outliers.

Scan-level thresholds based on network specificity and confound temporal correlation



The measures of network specificity (using Dice overlap) and temporal correlation with confounds (where confound timecourses are extracted using confound components specified with `--conf_prior_idx` and measured through dual regression) were defined in [DGDGMC23] for conducting scan-level QC (the figure above is reproduced from the study). They were selected as ideal measures for quantifying issues of network detectability and spurious connectivity (the figure above demonstrate how *categories of scan quality outcomes* can be distinguished with these metrics), and applying inclusion thresholds to select scans which respect assumptions for network detectability and minimal effects from confounds.

7.7.3 Group stats

Inspecting scan-level features is insufficient to conclude that inter-scan variability in connectivity isn't itself impacted (which is of primary interest for group analysis). This final report is aimed at inspecting features of connectivity variability at the group level, and focuses on two aspects:

1. **Specificity of network variability:** the standard deviation in connectivity across scan is computed voxelwise. This allows to visualize the spatial contrast of network variability. If primarily driven by network connectivity, the contrast should reflect the anatomical extent of the network of interest (as in the example above for the mouse somatomotor network), or otherwise may display spurious or absent features. For more details on the development of this metric, consult [DGDGMC23].
    - **Relationship to sample size**: [DGDGMC23] demonstrate that the contrast of the network variability map depends on sample size. If network connectivity is observed in individual scans, but not in this statistical report, increasing sample size may improve this contrast.
2. **Correlation with confounds:** Connectivity is correlated across subject, for each voxel, with either of the three measures of confound included: variance explained from confound correction at a given voxel (CR_{SD}, see *predicted confound timeseries* Y_{CR}), mean framewise displacement (FD), or temporal degrees of freedom. This allows establishing the importance of the association with potential confounds. What constitute a 'concerning' correlation may depend on the study, and the effect size of interest (i.e. is the effect size of interest much higher or similar to the effect size of confounds?).

Quantitative CSV report: A CSV file is also automatically generated along the figure, which records a quantitative assessment of these two aspects. More specifically, the overlap between the network variability map and the reference network map is measuring using Dice overlap, and for confound measures, the mean correlation is measured within the area of the network (consult the *metric details elsewhere*). These measures can be referred to for a quantitative summary instead (although visualization is preferred, as the Dice overlap for network variability may not perfectly distinguish network and spurious features).

IMPORTANT: the validity of this report is dependent on whether *scan-level assumptions* of network detectability and minimal confound effects are met. This is because either the lack of network activity or spurious effects in a subset of scan can drive 'apparent' network variability, since there will be differences in the presence VS absence of the network across scans, but these differences may be actually driven by data quality divergences.

7.7.4 Optimization of confound correction strategy

On this page is a procedure for improving confound correction design based on observations from the data quality assessment reports. These recommendations were originally developed in [DGDGMC23], and consist of a stepwise protocol where confound correction is improved incrementally while referring to data quality reports and the table found on this page, relating data quality features to corresponding corrections. The protocol is as follows:

1. Initiate a **minimal** confound correction, and generate data quality reports at the analysis stage. Correction should be minimal at first to mitigate potential issues of over-correction, where network activity itself can be removed by excessive correction. A minimal correction can consist of applying frame censoring using framewise displacement and the regression of 6 motion parameters together with spatial smoothing.

2. Evaluation of the data quality reports (as described in the *guidelines on the main page*)
3. The most sensible additional correction is selected based on the observations and using the table below.
4. The confound correction pipeline stage is re-run with **one** additional correction at a time, and the data quality reports are re-evaluated. Only a single correction is tested at a time so its impact can be evaluated, and the correction is only kept if there were beneficial impacts.
5. Repeat 3. and 4. until desirable quality outcomes are met, or no further options are left for confound correction.

The table below offers guidance for prioritizing additional corrections based on observations from the data quality reports. The confound correction workflow and the various strategies available are described elsewhere in the *confound correction pipeline*.

Data quality can have serious impacts on analysis outcomes, leading to false findings. Rodent imaging can suffer from spurious effects on connectivity measures if potential confounds are not well accounted for, or acquisition factors, such as anesthesia levels, can influence network activity [DGDGMC23, GCA+20]. To support interpretability, troubleshooting and reproducible research, RABIES includes a set of reports for conducting data quality assessment in individual scans and conducting quality control prior to network analysis at the group level. The reports are designed most specifically to evaluate the detectability of canonical brain networks and the impact of potential confounds (motion, physiological instabilities, and more).

This page describes how to generate the reports, our guidelines for conducting quality network of network analysis, and how to include those reports in a publication.

7.7.5 Generating the reports

At the analysis stage of the pipeline, the `--data_diagnosis` option can be selected to generate the data quality reports. To generate the report, ICA components must also be provided with `--prior_maps` and a set of components corresponding to confounds must be selected using `--conf_prior_idx` (see further details below). Connectivity can be evaluated for both dual regression and seed-based connectivity:

- **For *dual regression***: dual regression is always conducted using the set components from `--prior_maps`, since certain features are derived from confound components defined in `--conf_prior_idx`. On the other hand, connectivity will be evaluated in the reports for each network included in `--bold_prior_idx`.
- **For *seed-based connectivity***: reports will be generated for each seed provided to `--seed_list`. However, each seed needs to be supplemented with a reference network map (a 3D Nifti file for each seed, provided with `--seed_prior_list`) which should represent the expected connectivity for the canonical network corresponding to that seed.

The set of reports are generated in the `data_diagnosis_datasink/` (details *here*). The interpretation of each report is described within its dedicated documentation page, and include:

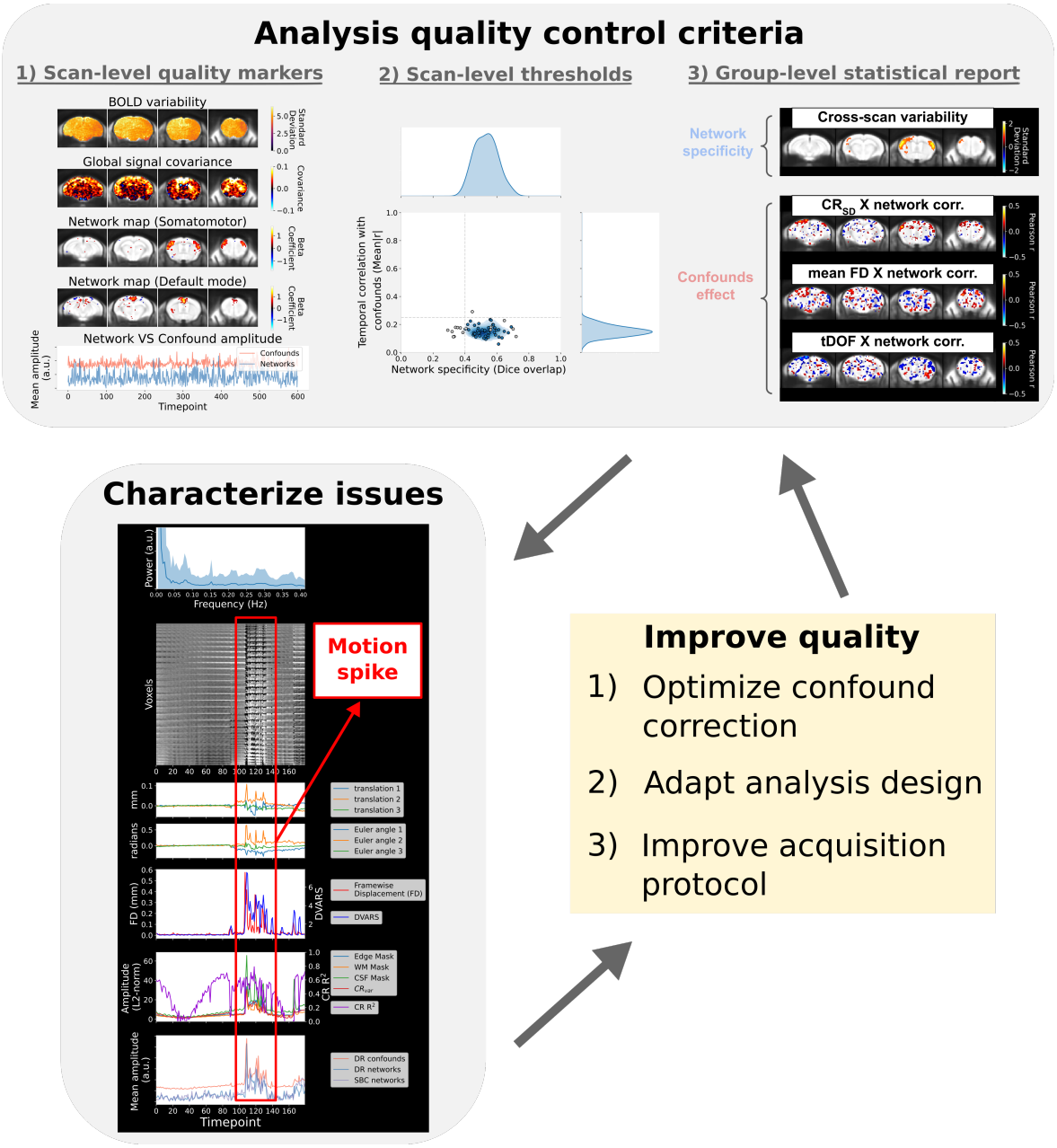
- *Spatiotemporal diagnosis*: this qualitative report generated for each scan regroups a set of temporal and spatial features allowing to characterize the specific origin of data quality issues.
- *Distribution plots*: quantitative report displaying the distribution of scans along measures of: specificity of network connectivity, network amplitude (for dual regression), and confound measures. Visualizing the dataset distribution can help identify outliers.
- *Group statistics*: for a given network, this group-level report regroups brain maps for visualizing cross-scan variability in connectivity and the group-wise correlation between connectivity and confounds.

Classification of group ICA components

Ideally, the ICA components should be derived directly from the dataset analyzed by using *group ICA*, although a pre-computed set is available by default. Newly-generated components must be visually inspected to identify the set of components corresponding to confound sources (which is inputted with `--conf_prior_idx`). This can be done by visualizing the group_melodic.ica/melodic_IC.nii.gz file, or using the automatically-generated FSL report in group_melodic.ica/report. Similarly, components corresponding to networks of interest can be identified and inputted with `--bold_prior_idx`.

Classifying components requires careful considerations, and we recommend a conservative inclusion (i.e. not every components need to be classified, only include components which have clear feature delineating a network or a confound). Consult [ZGRW15] or [DGDGMC23] for more information on classifying ICA components in rodents, or the pre-computed set can be consulted as reference (the defaults for `--bold_prior_idx` and `--conf_prior_idx` correspond to the classification of these components).

7.7.6 Guidelines for analysis quality control



Below are our recommendations for how the set of quality reports can be used identify and control for the impact of data quality issues on downstream group analysis. Although the reports may be used for a breadth of applications, these guidelines are formulated most specifically for a standard resting-state fMRI design aiming to compare network connectivity between subjects or groups. In particular, the following guidelines aim to identify features of spurious or absent connectivity, remove scans where these features are prominent to avoid false results (e.g. connectivity difference is driven by motion), and determine whether these issues may confound group statistics.

1. Inspect the *spatiotemporal diagnosis* for each scan. Particular attention should be given to the 4 main quality markers defining *categories of scan quality*, and whether features of spurious or absent connectivity are prominent.

2. If spurious or absent connectivity is prominent in a subset of scans, these scans should be detected and removed

to mitigate false results. This is done by setting thresholds using `--scan_QC_thresholds` for scan-level measures of network specificity and confound correlation. These measures are documented in the *distribution plots*, and the specific measures for each scan ID can be consulted in the CSV file accompanying the plot. Using this CSV file, sensible threshold values can be selected for delineating scans with spurious or absent connectivity. Additionally, for dual regression analysis, `--scan_QC_thresholds` can be used to automatically detect and remove scans which present outlier values in network amplitude, which can be an indicator of spurious connectivity [NSOngurB17]. By applying `--scan_QC_thresholds`, these scans won't be included for generating the group statistical report (thus the reports must re-generated after defining `--scan_QC_thresholds`).

3. Finally, the *group statistical report* can be consulted to identify the main driven of variability in connectivity across scans, and whether it relates primarily to network activity or to confounds.

If significant issues are found from this evaluation, the design of the confound correction stage may be revisited to improve quality outcomes (see *dedicated documentation*).

Disclaimer: Although these guidelines are meant to support identifying analysis pitfalls and improve research transparency, they are not meant to be prescriptive. The judgement of the experimenter is paramount in the adopting adequate practices (e.g. network detectability may not always be expected, if studying the impact of anesthesia or inspecting a visual network in blind subjects), and the conversation surrounding what should constitute proper standards for resting-state fMRI is evolving.

Reporting in a publication

All figures from the report are generated in PNG (or SVG) format, and can be shared along a publication for data transparency. Ideally, a version of the spatiotemporal diagnosis can be shared for each scan used in deriving connectivity results, together with a group statistical report and its affiliated distribution plot for each groups/datasets if the analysis involves comparing connectivity differences across subjects and/or group.

The set of ICA components classified as networks and confounds should be reported appropriately (e.g. melodic_IC.nii.gz file can be shared with its associated component classification). If certain scan inclusion/exclusion criteria were selected based on the quality control guidelines described above, it is particularly important to describe the observations motivating these criteria and make the associated reports readily accessible for consultation (e.g. the set of spatiotemporal diagnosis files for scans displaying spurious/absent connectivity and motivated setting a particular QC threshold with `--scan_QC_thresholds`). If the design of confound correction was defined using these tools, this should also be appropriately reported.

7.8 Understanding the Outputs

In this section, there is a description for all the output files provided at each processing stage. Important outputs from RABIES are stored into `datasink/` folders, which will be generated in the output folder specified at execution.

7.8.1 Preprocessing Outputs

Multiple datasink folders are generated during preprocessing for different output types: `anat_datasink/`, `bold_datasink/`, `unbiased_template_datasink/`, `transforms_datasink/` and `confounds_datasink/`.

- `anat_datasink/`: Includes the inhomogeneity-correction anatomical scans.
  - `anat_preproc/`: anatomical scans after inhomogeneity correction
- `bold_datasink/`: Includes all outputs related to the functional scans, where files are either resampled onto the native or commonspace of the EPI. The native space outputs are resampled over the anatomical scan from each corresponding MRI session, whereas the commonspace outputs are resampled over the reference atlas (the original EPI voxel resolution is unchanged during resampling unless specified otherwise in the RABIES command).

  - native_bold/: preprocessed EPI timeseries resampled to nativespace
  - native_brain_mask/: brain mask in nativespace
  - native_WM_mask/: WM mask in nativespace
  - native_CSF_mask/: CSF mask in nativespace
  - native_labels/: atlas labels in nativespace
  - native_bold_ref/: a volumetric 3D EPI average generated from the 4D native_bold/
  - commonspace_bold/: preprocessed EPI timeseries resampled to commonspace
  - commonspace_mask/: brain mask in commonspace
  - commonspace_WM_mask/: WM mask in commonspace
  - commonspace_CSF_mask/: CSF mask in commonspace
  - commonspace_vascular_mask/: vascular mask in commonspace
  - commonspace_labels/: atlas labels in commonspace
  - commonspace_resampled_template/: the commonspace anatomical template, resampled to the EPI's dimensions
  - input_bold/: the raw EPI scans provided as inputs in the BIDS data folder
  - initial_bold_ref/: the initial volumetric 3D EPI average generated from the 4D input_bold/
  - raw_brain_mask/: brain mask resampled onto the 4D input_bold/
  - inho_cor_bold/: the volumetric 3D EPI (initial_bold_ref/) after inhomogeneity correction, which is later used for registration of the EPI
  - inho_cor_bold_warped2anat/: inho_cor_bold after co-registration to the associated anatomical image (anat_preproc/)
  - std_map_preprocess/: the temporal standard deviation at each voxel on the commonspace_bold/
  - tSNR_map_preprocess/: the temporal signal-to-noise ratio (tSNR) of the commonspace_bold/

- unbiased_template_datasink/: Outputs related to the generation of the unbiased template using https://github.com/CoBrALab/optimized_antsMultivariateTemplateConstruction. The unbiased template corresponds to the average of all anatomical (or functional with --bold_only) scans after their alignment.
  - unbiased_template/: the unbiased template generated from the input dataset scans
  - warped_unbiased_template/: the unbiased template, registered to the reference atlas in commonspace
- transforms_datasink/: datasink for all the relevant transform files resampling between the different spaces. The bold_to_anat registration transformed the raw EPI to overlap with the anatomical image, correcting for susceptibility distortions, which corresponds to the native space. The native_to_unbiased registration overlaps every scans to the generated unbiased template, and then the unbiased_to_atlas corresponds to the registration of the unbiased template with the reference atlas, which defines the commonspace.
  - bold_to_anat_affine/: affine transforms from the EPI co-registration to the anatomical image
  - bold_to_anat_warp/: non-linear transforms from the EPI co-registration to the anatomical image
  - bold_to_anat_inverse_warp/: inverse of the non-linear transforms from the EPI co-registration to the anatomical image
  - native_to_unbiased_affine/: affine transforms for the alignment between native space and the unbiased template

  - native_to_unbiased_warp/: non-linear transforms for the alignment between native space and the unbiased template
  - native_to_unbiased_inverse_warp/: inverse of the non-linear transforms for the alignment between native space and the unbiased template
  - unbiased_to_atlas_affine/: affine transforms for the alignment between unbiased template and the atlas in commonspace
  - unbiased_to_atlas_warp/: non-linear transforms for the alignment between unbiased template and the atlas in commonspace
  - unbiased_to_atlas_inverse_warp/: inverse of the non-linear transforms for the alignment between unbiased template and the atlas in commonspace
- motion_datasink/: files derivated from motion estimation
  - motion_params_csv/: contains the 24 motion parameters which can be used as nuisance regressors at the confound correction pipeline stage.
  - FD_csv/: a CSV file with timescourses for either the mean or maximal framewise displacement (FD) estimations.
  - FD_voxelwise/: a Nifti image which contains framewise displacement evaluated at each voxel
  - pos_voxelwise/: a Nifti image which tracks the displacement (derived from the head motion realignment parameters) of each voxel across time

7.8.2 Confound Correction Outputs

Important outputs from confound correction will be found in the confound_correction_datasink/:

- confound_correction_datasink/:
  - cleaned_timeseries/: cleaned timeseries after the application of confound correction
  - frame_censoring_mask/: contains CSV files each recording as a boolean vector which timepoints were censored if frame censoring was applied.
  - aroma_out/: if --ica_aroma is applied, this folder contains outputs from running ICA-AROMA, which includes the MELODIC ICA outputs and the component classification results
  - plot_CR_overfit/: will contain figures illustrating the variance explained by random regressors during confound correction, and the variance explained by the real regressors after substrating the variance from random regressors.
  - background_masking_fig/: will illustrate the image background masks automatically generated if using --image_scaling background_noise

7.8.3 Analysis Outputs

Outputs from analyses will be found in the analysis_datasink/, whereas outputs relevant to the --data_diagnosis are found in data_diagnosis_datasink/:

- analysis_datasink/:
  - group_ICA_dir/: complete output from MELODIC ICA, which the melodic_IC.nii.gz Nifti which gives all spatial components, and report/ folder which includes a HTML visualization.

  - matrix_data_file/: .pkl file which contains a 2D numpy array representing the whole-brain correlation matrix. If --ROI_type parcellated is selected, the row/column indices of the array are matched in increasing order of the atlas ROI label number.
  - matrix_fig/: .png file which displays the correlation matrix
  - seed_correlation_maps/: nifti files for seed-based connectivity analysis, where each seed provided in --seed_list has an associated voxelwise correlation maps
  - dual_regression_nii/: the spatial maps from dual regression, which correspond to the linear coefficients from the second regression. The list of 3D spatial maps obtained are concatenated into a 4D Nifti file, where the order of component is consistent with the priors provided in --prior_maps.
  - dual_regression_timecourse_csv/: a CSV file which stores the outputs from the first linear regression during dual regression. This corresponds to a timecourse associated to each prior component from --prior_maps.
  - NPR_prior_filename/: spatial components fitted during NPR
  - NPR_prior_timecourse_csv/: timecourses associated to each components from NPR_prior_filename
  - NPR_extra_filename/: the extra spatial components fitted during NPR which were not part of priors
  - NPR_extra_timecourse_csv/: timecourses associated to each components from NPR_extra_filename

- data_diagnosis_datasink/:
  - figure_temporal_diagnosis/: figure which displays scan-level temporal features from the *spatiotemporal diagnosis*
  - figure_spatial_diagnosis/: figure which displays scan-level spatial features from the *spatiotemporal diagnosis*
  - analysis_QC/: group-level features of data quality from --data_diagnosis
    - sample_distributions/: contains the *distribution plots*
      - {analysis}_sample_distribution.png: the distribution plot for a given network analysis
      - {analysis}_outlier_detection.csv: a CSV which associates the measures displayed in the distribution plot with corresponding scan IDs
    - parametric_stats/: *group statistical report* for analysis quality control (using parametric measures)
      - DR{component #}_QC_maps.png: The _QC_maps.png files are displaying statistical maps relevant to analysis quality control. The DR refers to dual regression analysis, and the {component #} is relating the file to one of the BOLD components specified in --prior_bold_idx
      - DR{component #}_QC_stats.csv: a follow-up to _QC_maps.png which allows for the quantitative categorization of data quality outcomes in Desrosiers-Gregoire et al. (in prep.)
      - seed_FC{seed #}_QC_maps.png: same statistical maps as with DR{component #}_QC_maps.png, but for seed-based connectivity analysis
      - seed_FC{seed #}_QC_stats.csv: same measures as with DR{component #}_QC_maps.png, but for seed-based connectivity analysis
    - non_parametric_stats/: same as parametric_stats/, but using non-parametric measures
  - temporal_info_csv/: CSV file containing the data plotted with figure_temporal_diagnosis/
  - spatial_VE_nii/: Nifti file with the confound regression percentage variance explained (R^2) at each voxel
  - CR_prediction_std_nii/: Nifti file with the confound regression variance explained at each voxel

- `random_CR_std_nii/`: Nifti file with the variance explained from random regressors at each voxel
- `corrected_CR_std_nii/`: Nifti file with the confound regression variance explained at each voxel after removing the variance explained by random regressors
- `temporal_std_nii/`: the standard deviation at each voxel after confound correction
- `GS_cov_nii/`: the covariance of each voxel with the global signal

7.9 Metric definitions

On this page, the L2-norm corresponds to $||x||_2 = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$

7.9.1 Nuisance regressors for confound regression

- **mot_6**: Corresponds to the head motion translation and rotation parameters. Prior to the regression, the motion regressors are also subjected to the same frame censoring, detrending and frequency filtering which were applied to the BOLD timeseries to avoid the re-introduction of previously corrected confounds, as recommend in [PML+14] and [LGWC19].
- **mot_24**: Corresponds to the 6 motion parameters together with their temporal derivatives, and 12 additional parameters are obtained by taking the squared terms (i.e. Friston 24 parameters [FWH+96])

$$mot24_t = [mot6_t, (mot6_t - mot6_{t-1}), (mot6_t)^2, (mot6_t - mot6_{t-1})^2]$$

  with $mot24_t$ representing the list of 24 regressors for timepoint t. As with mot_6, the 24 regressors are additionally subjected to censoring, detrending and frequency filtering if applied on BOLD.
- **WM/CSF/vascular/global signal**: The mean signal is computed within the corresponding brain mask (WM,CSF,vascular or whole-brain mask) from the partially cleaned timeseries (i.e. after the confound correction steps 1-4 up to frequency filtering).
- **aCompCor_percent**: Principal component timecourses are derived from timeseries within the combined WM and CSF masks (aCompCor technique [MNC+14]). From the timeseries within the WM/CSF masks $Y_{WM/CSF}$, a principal component analysis (PCA) decomposition is conducted to derive

$$Y_{WM/CSF} = W_{aCompCor} C^T$$

  with C corresponding to a set of spatial principal components, and W to their associated loadings across time. The set of first components explaining 50% of the variance are kept, and their loadings $W_{aCompCor}$ provide the set of aCompCor nuisance regressors. PCA is conducted on the partially cleaned timeseries (i.e. after the confound correction steps 1-4 up to frequency filtering).
- **aCompCor_5**: Same as **aCompCor_percent**, but the first 5 components are kept instead of a set explaining 50% of the variance.

7.9.2 Temporal scan diagnosis

- **Head motion translation and rotation parameters**: Corresponds to 3 rotations (Euler angles in radians) and 3 translations (in mm) measured for head motion realignment at each timeframe.
- **Framewise displacement**: For each timepoint, corresponds to the displacement (mean across the brain voxels) between the current and the next frame. For each brain voxel within the referential space for head realignment (i.e. the *3D EPI* which was provided as reference for realignment) and for each timepoint, the inverse transform of the head motion parameters (from the corresponding timepoint) is applied to obtain the voxel position pre-motion correction. Framewise displacement can then be computed for each voxel by computing the Euclidean distance between the positions pre-motion correction for the current and next timepoints. Thus, the mean framewise displacement FD_t at timepoint t is computed as

$$FD_t = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x_{i,t+1}-x_{i,t})^2+(y_{i,t+1}-y_{i,t})^2+(z_{i,t+1}-z_{i,t})^2}$$

  using the 3D x,y and z spatial coordinates in mm for timepoints t and $t+1$ and for each voxel indices i. Framewise displacement for the last frame (which has no future timepoint) is set to 0.

- **DVARS**: represents the estimation of temporal shifts in global signal at each timepoint, measured as the root-mean-square of the timeseries' temporal derivative

$$DVARS_t = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_{i,t}-Y_{i,t-1})^2}$$

  where $Y_{i,t}$ corresponds to the BOLD signal in brain voxel i at timepoint t. The first timepoint is set to 0 (has no previous timepoint).

- **Edge/WM/CSF mask**: The L2-norm across voxels within a mask, at each timepoint.
- CR_{var}: The variance estimated by confound regression is computed for each timepoint. This is done by taking the L2-norm $CR_{var} = ||Y_{CR}||_2$ across voxels at each timepoints, where Y_{CR} is the *predicted confound timeseries*.
- **CR** R^2: Represents the proportion of variance explained (and removed) by confound regression. This is obtained with $CR_{R^2} = 1 - \frac{Var(\hat{Y})}{Var(Y)}$ at each timepoint, where Y and $\hat{Y}$ are the timeseries pre- and post-regression respectively, and $Var(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu_x)^2$ calculates the variance, with μ as the mean.
- **Mean amplitude**: A set of timecourse are averaged as $\frac{1}{n}\sum_{i=1}^{n}|X_i|$, where X_i is the timecourse i. Timecourses can correspond to either of the following sets:
  - DR confounds: timecourses from the first stage of dual regression, using confound components provided to `--prior_confound_idx`.
  - DR networks: network timecourses from the first stage of dual regression as specified with `--prior_bold_idx`.
  - SBC networks: network timecourses derived from the set of seeds provided in `--seed_list`.

7.9.3 Spatial scan diagnosis

- **BOLDSD**: The temporal standard deviation is computed for each voxel from the BOLD timeseries.
- **CRSD**: The temporal standard deviation computed on each voxel from the predicted confound timeseries during confound regression (i.e. Y_{CR}).
- **CR R2**: The proportion of variance explained by confound regression at each voxel. This is obtained with $CR_{R^2} = 1 - \frac{Var(\hat{Y})}{Var(Y)}$ at each voxel, where Y and $\hat{Y}$ are the timeseries pre- and post-regression respectively, and $Var(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)^2$ is variance of x, with μ as the mean.
- **Global signal covariance (GScov)**: The covariance between the global signal and the timeseries at each voxel is measured as $GS_{cov} = \frac{1}{n}\sum_{t=1}^{n} Y_t \times GS_t$, where $GS_t = \frac{1}{n}\sum_{i=1}^{n} Y_i$, i.e. the mean across all brain voxels for a given timepoint.
- **DR network X**: The linear coefficients resulting from the *second regression with dual regression*, corresponding to a network amplitude map (for the Xth network specified for analysis with `--prior_bold_idx`).
- **SBC network X**: The voxelwise correlation coefficients (pearson's r) estimated with seed-based connectivity (for the Xth seed provided for analysis with `--seed_list`).

7.9.4 Distribution plot

- **Network amplitude (not computed for seed-based connectivity)**: The overall network amplitude is summarized by computing the L2-norm across the network connectivity map outputed from dual regression (i.e. linear coefficients from the *second regression* β_{SM})
- **Network specificity**: The network map (seed-based or dual regression) and the corresponding canonical network map are thresholded to include the top 4% of voxels with highest connectivity, and the overlap of the thresholded area is computed using Dice overlap. For dual regression, the 'canonical network' map will consist of the original ICA component corresponding to that network provided with `--prior_maps`, and for seed-based connectivity, the reference network maps are provided using the `--seed_prior_list` parameter.
- **Dual regression confound correlation**: The timecourse for a single network (from a seed or dual regression) is correlated with the timecourse from each confound component (provided using `--prior_confound_idx`) modelled through dual regression, then the absolute mean correlation is computed to obtain the average amplitude of confound correlations for this specific network analysis.
- **Total** CR_{SD}: The total standard deviation across the *predicted confound timeseries* Y_{CR}.
- **Mean framewise displacement**: The mean framewise displacement computed across time (only including frames after censoring applied for confound correction).
- **Temporal degrees of freedom**: The remaining degrees of freedom post-confound correction are calculated as `tDOF = Original number of timepoints - Number of censored timepoints - Number of AROMA components removed - Number of nuisance regressors`.

7.9.5 Group statistical QC report

- **Specificity of network variability**: similarly to network specificity in the distribution plot, the network variability map and the corresponding canonical network map are thresholded to include the top 4% of voxels, and then the overlap is estimated using Dice overlap.
- **Mean confound correlation**: for each confound correlation map (either CR_{SD}, mean FD or tDOF), the mean is computed across voxels included within the thresholded area of the canonical network map, to obtain a mean correlation within the network's core region.

7.10 Contributing to RABIES

RABIES aims to provide an accessible tool responding to growing needs across the preclinical fMRI community. This effort should be community-driven, and community involvement will be paramount in achieving this goal in several respects:

- Adapting and maintaining **accessibility** for users across the broader community
- **Reproducibility and transparency**, as well as scientific scrutiny and rigor
- Defining and incorporating **best practices** across the different aspects of image processing and analysis, as well as quality control
- Leveraging appropriate expertise for the **integration of new tools**

Suggestions for improvements can be shared using the Github issues system and discussion board. Contributions from developers are welcomed and encouraged. This page provides preliminary guidelines for getting started as a RABIES developer, and covers: setting a developer environment, submitting a pull request, testing and debugging, and basic instructions for adding a new module to the pipeline. We recommend discussing your proposed updates on the Github discussion board or issues prior to creating a pull request. Thank you for your support!

7.10.1 Dev environment

For development, it is recommend to install RABIES locally, as this will make the testing and debugging process smoother. This requires installing the dependencies listed in dependencies.txt, and then installing RABIES in an appropriate python environment (e.g. using anaconda) from the Github repository. This can be done by cloning the repository, and then running `python setup.py install`.

...using a container

It is possible to run operations using a container to avoid installing dependencies manually (however, it won't be possible to use an interface for debugging (e.g. Spyder)). This can be with `docker exec`. First, an instance of the container must be opened, which can be done by including `-d --entrypoint sh --name mycontainer` when calling `docker run`. The paths to access from the container must be set with `-v`. Here's an example:

```
docker run -it -v $PWD:/work_dir -v /path_to_local_RABIES_package:/RABIES:ro \
--rm --entrypoint sh -d --name mycontainer rabies:local_testing
```

You can then execute commands from inside the container as follows (`mycontainer` corresponds to the name set above):

```
docker exec mycontainer micromamba run $COMMAND
```

To test for error, `$COMMAND` can correspond to `error_check_rabies.py --complete`.

Upgrading the RABIES package: to test your updates, you must re-install RABIES inside the container. Below is a method to do so:

```
mkdir -p /tmp/RABIES
# copy all files from your upgraded package inside the container (/RABIES must be␣
↪related to your local package with -v)
rsync -avz /RABIES/* /tmp/RABIES/.
# re-install package
cd /tmp/RABIES
python setup.py install
```

These can be compiled into a .sh script to execute in place of `$COMMAND` above.

7.10.2 Instructions to create a pull request

1. On github, fork the RABIES repository to have your own copy.
2. Clone your repository to carry out local modifications and testing. Use the `--recursive` option to download the submodules together with the main RABIES package.
3. Create and checkout into a new branch with `git checkout -b my_new_branch` (provide a sensible name for the branch). You are ready to make modifications to the code.
4. Testing and debugging: install your updated version of the package with `python setup.py install`, using a proper dev environment (see above). Your can test the workflow with specific parameters by editing the `debug_workflow.py` script, and executing in debug mode with Spyder (see below). Before commiting changes, make sure that running `error_check_rabies.py --complete` completes with no error.
5. Commit and push your modifications to Github, and create a pull request from your forked repo to the original.

7.10.3 Interactive debugging with Spyder and debug_workflow.py

Here are some recommendations for debugging using Spyder:

1. open the debug_workflow.py file in Spyder
2. find the scrips with your local installation to add breakpoints for debugging. Using `import rabies; os.path.abspath(rabies.__file__)` will provide the path to the **init**.py file of your installed package, and from there you can find file of interest and add a breakpoint where desired.
3. execute debug_workflow.py in debug mode, and run until it finds the breakpoint, and debug from there.

7.10.4 Creation of a new module and integration within a Nipype workflow

RABIES' workflow is structured using Nipype (for more info on Nipype, see online documentation and tutorial). Preferably, a new function should be created as a Nipype interface, which has the following syntax:

```
from nipype.interfaces.base import (
    traits, TraitedSpec, BaseInterfaceInputSpec,
    File, BaseInterface
)
class NewInterfaceInputSpec(BaseInterfaceInputSpec):
    # you must select an appropriate input type with traits.type (can be Dict, File, Int,
↪ ...)
    input_str = traits.Str(exists=True, mandatory=True,
                           desc="An input string.")

class NewInterfaceOutputSpec(TraitedSpec):
    out_file = File(
        exists=True, desc="An output file.")


class NewInterface(BaseInterface):
    """
    Describe your module.
```

(continues on next page)

(continued from previous page)

```
    """

    input_spec = NewInterfaceInputSpec
    output_spec = NewInterfaceOutputSpec

    def _run_interface(self, runtime):
        input_str = self.inputs.input_str

        '''
        YOUR CODE
        '''

        setattr(self, 'out_file', out_file)

        return runtime

    def _list_outputs(self):
        return {'out_file': getattr(self, 'out_file')}
```

You can then create a Nipype node for your interface:

```
from .other_script import NewInterface # import your interface if from a different script
from nipype.pipeline import engine as pe

new_interface_node = pe.Node(NewInterface(),
                             name='new_interface')
```

Instead of an interface, it is also possible to create a Nipype node from any python function:

```
from nipype.pipeline import engine as pe
from nipype.interfaces.utility import Function

new_function_node = pe.Node(Function(input_names=['input_1', 'input_2', ...],
                                                output_names=['output_1', 'output_2', ...
↪],
                                                function=NewFunction),
                                        name='new_function')
```

After creating a node which can carry the desired operation, it must be integrated within a workflow by linking up the inputs and outputs with other nodes. Below is an example of a simple workflow which conducts slice-timing correction:

```
from nipype.pipeline import engine as pe
from nipype.interfaces.utility import Function
from nipype.interfaces import utility as niu

# this function creates and return a Nipype workflow which conducts slice timing␣
↪correction
def init_bold_stc_wf(name='bold_stc_wf'):

    workflow = pe.Workflow(name=name) # creating a new Nipype workflow
    # creating an intermediate node for storing inputs to the workflow
```

(continues on next page)

(continued from previous page)

```
    inputnode = pe.Node(niu.IdentityInterface(
        fields=['bold_file']), name='inputnode')
    # creating an intermediate node for storing outputs to the workflow
    outputnode = pe.Node(niu.IdentityInterface(
        fields=['stc_file']), name='outputnode')

    # preparing the node conducting STC
    slice_timing_correction_node = pe.Node(Function(input_names=['in_file', 'tr',
↪'tpattern', 'stc_axis',
                                                                 'interp_method', 'rabies_data_
↪type'],
                                                    output_names=[
                                                        'out_file'],
                                                    function=slice_timing_correction),
                                           name='slice_timing_correction', mem_gb=1.
↪5*opts.scale_min_memory)

    # linking up the inputnode to provide inputs to the STC node, and outputs from STC␣
↪to the outputnode of the workflow
    workflow.connect([
        (inputnode, slice_timing_correction_node, [('bold_file', 'in_file')]),
        (slice_timing_correction_node,
            outputnode, [('out_file', 'stc_file')]),
    ])
    return workflow
```

This example demonstrates the basic syntax of a Nipype workflow. Most likely, a new interface will be integrated as part of a pre-existing workflow (instead of creating a new one), in which case the right nodes must be linked up with the new interface.

7.11 Bibliography

BIBLIOGRAPHY

[ATS+11] Brian B Avants, Nicholas J Tustison, Gang Song, Philip A Cook, Arno Klein, and James C Gee. A reproducible evaluation of ANTs similarity metric performance in brain image registration. *Neuroimage*, 54(3):2033–2044, February 2011.

[BMFS09] Christian F Beckmann, Clare E Mackay, Nicola Filippini, and Stephen M Smith. Group comparison of resting-state FMRI data using multi-subject ICA and dual regression. *Neuroimage*, 47(Suppl 1):S148, 2009.

[BS04] Christian F Beckmann and Stephen M Smith. Probabilistic independent component analysis for functional magnetic resonance imaging. *IEEE Trans. Med. Imaging*, 23(2):137–152, February 2004.

[BZYHH95] Bharat Biswal, F Zerrin Yetkin, Victor M Haughton, and James S Hyde. Functional connectivity in the motor cortex of resting human brain using echo-planar MRI. *Magn. Reson. Med.*, 34(4):537–541, 1995.

[Car13] Joshua Carp. Optimizing the order of operations for movement scrubbing: comment on power et al. *Neuroimage*, 76:436–438, August 2013.

[DGDGMC23] Gabriel Desrosiers-Gregoire, Gabriel A Devenyi, Joanes Grandjean, and M Mallar Chakravarty. Rodent automated bold improvement of EPI sequences (RABIES): a standardized image processing and data quality platform for rodent fMRI. *status needed for bibtex rendering with Sphinx*, September 2023.

[EMB+19] Oscar Esteban, Christopher J Markiewicz, Ross W Blair, Craig A Moodie, A Ilkay Isik, Asier Erramuzpe, James D Kent, Mathias Goncalves, Elizabeth DuPre, Madeleine Snyder, Hiroyuki Oya, Satrajit S Ghosh, Jessey Wright, Joke Durnez, Russell A Poldrack, and Krzysztof J Gorgolewski. fMRIPrep: a robust preprocessing pipeline for functional MRI. *Nat. Methods*, 16(1):111–116, January 2019.

[FWH+96] Karl J Friston, Steven Williams, Robert Howard, Richard S J Frackowiak, and Robert Turner. Movement-Related effects in fMRI time-series. 1996.

[GAC+16] Krzysztof J Gorgolewski, Tibor Auer, Vince D Calhoun, R Cameron Craddock, Samir Das, Eugene P Duff, Guillaume Flandin, Satrajit S Ghosh, Tristan Glatard, Yaroslav O Halchenko, Daniel A Handwerker, Michael Hanke, David Keator, Xiangrui Li, Zachary Michael, Camille Maumet, B Nolan Nichols, Thomas E Nichols, John Pellman, Jean-Baptiste Poline, Ariel Rokem, Gunnar Schaefer, Vanessa Sochat, William Triplett, Jessica A Turner, Gaël Varoquaux, and Russell A Poldrack. The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments. *Sci Data*, 3:160044, June 2016.

[GCA+20] Joanes Grandjean, Carola Canella, Cynthia Anckaerts, Gülebru Ayrancı, Salma Bougacha, Thomas Bienert, David Buehlmann, Ludovico Coletta, Daniel Gallino, Natalia Gass, Clément M Garin, Nachiket Abhay Nadkarni, Neele S Hübner, Meltem Karatas, Yuji Komaki, Silke Kreitz, Francesca Mandino, Anna E Mechling, Chika Sato, Katja Sauer, Disha Shah, Sandra Strobelt, Norio Takata, Isabel Wank, Tong Wu, Noriaki Yahata, Ling Yun Yeow, Yohan Yee, Ichio Aoki, M Mallar Chakravarty, Wei-Tang Chang, Marc Dhenain, Dominik von Elverfeldt, Laura-Adela Harsan, Andreas Hess, Tianzi

Jiang, Georgios A Keliris, Jason P Lerch, Andreas Meyer-Lindenberg, Hideyuki Okano, Markus Rudin, Alexander Sartorius, Annemie Van der Linden, Marleen Verhoye, Wolfgang Weber-Fahr, Nicole Wenderoth, Valerio Zerbi, and Alessandro Gozzi. Common functional networks in the mouse brain revealed by multi-centre resting-state fMRI analysis. *Neuroimage*, 205:116278, January 2020.

[LGWC19] Martin A Lindquist, Stephan Geuter, Tor D Wager, and Brian S Caffo. Modular preprocessing pipelines can reintroduce artifacts into fMRI data. *Hum. Brain Mapp.*, 40(8):2358–2376, June 2019.

[MGG+04] Adolf Mathias, Florian Grond, Ramon Guardans, Detlef Seese, Miguel Canela, and Hans H Diebner. Algorithms for spectral analysis of irregularly sampled time series. *J. Stat. Softw.*, 11(1):1–27, 2004.

[MNC+14] John Muschelli, Mary Beth Nebel, Brian S Caffo, Anita D Barber, James J Pekar, and Stewart H Mostofsky. Reduction of motion-related artifacts in resting state fMRI using aCompCor. *Neuroimage*, 96:22–35, August 2014.

[NSOngurB17] Lisa D Nickerson, Stephen M Smith, Döst Öngür, and Christian F Beckmann. Using dual regression to investigate network shape and amplitude in functional connectivity analyses. *Front. Neurosci.*, 11:115, 2017.

[PBS+12] Jonathan D Power, Kelly A Barnes, Abraham Z Snyder, Bradley L Schlaggar, and Steven E Petersen. Spurious but systematic correlations in functional connectivity MRI networks arise from subject motion. *Neuroimage*, 59(3):2142–2154, February 2012.

[PML+14] Jonathan D Power, Anish Mitra, Timothy O Laumann, Abraham Z Snyder, Bradley L Schlaggar, and Steven E Petersen. Methods to detect, characterize, and remove motion artifact in resting state fMRI. *Neuroimage*, 84:320–341, January 2014.

[PPLM17] Jonathan D Power, Mark Plitt, Timothy O Laumann, and Alex Martin. Sources and implications of whole-brain fMRI signals in humans. *Neuroimage*, 146:609–625, February 2017.

[PMvR+15] Raimon H R Pruim, Maarten Mennes, Daan van Rooij, Alberto Llera, Jan K Buitelaar, and Christian F Beckmann. ICA-AROMA: a robust ICA-based strategy for removing motion artifacts from fMRI data. *Neuroimage*, 112:267–277, May 2015.

[WPG+17] Sijia Wang, Daniel J Peterson, J C Gatenby, Wenbin Li, Thomas J Grabowski, and Tara M Madhyastha. Evaluation of field map and nonlinear registration methods for correction of susceptibility artifacts in diffusion MRI. *Front. Neuroinform.*, 11:17, February 2017.

[ZGRW15] Valerio Zerbi, Joanes Grandjean, Markus Rudin, and Nicole Wenderoth. Mapping the mouse brain with rs-fMRI: an optimized pipeline for functional network identification. *Neuroimage*, 123:11–21, December 2015.