Atlas-based Imaging Data Analysis pipeline for functional and structural MRI Data **AIDAmri**

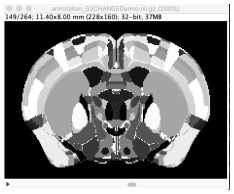
Leon Scharwächter, Niklas Pallast and Markus Aswendt
Department of Neurology
University Hospital Cologne

v1.1
November 2020

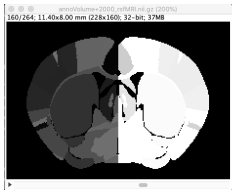
Contents

1 Introduction

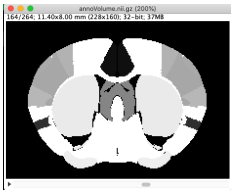
The Atlas-based Processing Pipeline for functional and structural MRI data (AIDAmri) was developed for automated processing of mouse brain MRI. AIDAmri works with T2-weighted MRI (T2w), diffusion weighted MRI or diffusion tensor imaging (DTI) and resting-state functional MRI (fMRI). The Allen Mouse Brain Reference Atlas (ARA, CCF v3) is registered on each of these MRI data sets and is used to analyse regions of interest. Furthermore, the regions of the ARA are used as seed-points for the connectivity and activity matrices. The lib User-defined ROIs and masks can be generated separately and used for analysis, e.g. stroke lesion masks and peri-infarct regions. AIDAmri comes with different atlas and template versions, which are necessary for the registration to work (Figure 1): a) annotation 50 changed anno: original ARA CCF v3 labels (regions with grey values¿100k changed to new values starting with 2000), b) anno volume 2000 rsfMRI: atlas from a) with reduced number of atlas regions by selective region fusion, 96 regions in total, split between hemispheres (right side +2000), c) same as in b) but no split, d) same as in a) but split, e) original ARA template with 50 um isotropic resolution, f) custom-made MRI template. For the complete list of atlas labels see: `annoVolume+2000_rsfMRI.nii.txt`



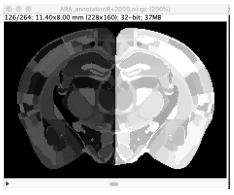
(a) Annotation 50 Changed Anno



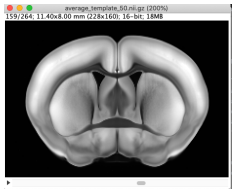
(b) Anno Volume 2000 rsfMRI



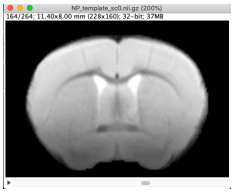
(c) Anno Volume



(d) ARA Annotation R2000



(e) Average Template 50



(f) NP Template sc0

Figure 1: Atlases included in the `/lib` folder

2 Installation

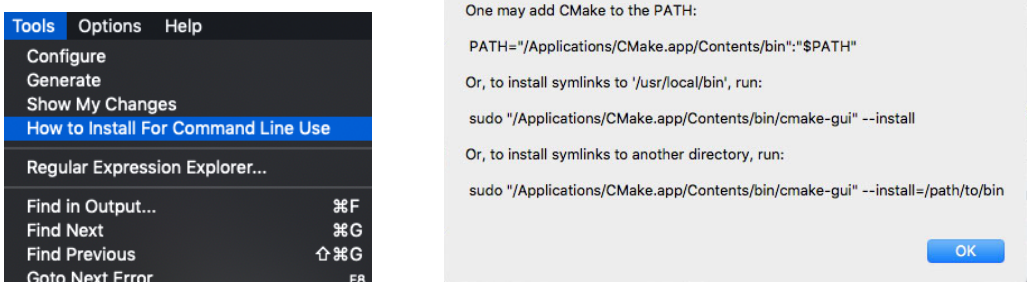
1. Download the folders `/bin` and `/lib` by using this link
   `/bin` and `/lib` should be located in the same directory

   The folder `/bin` contains the python scripts necessary for all preprocessing steps. The folder `/lib` contains several information about six atlases. Please do not conduct any changes to these files as it could influence the access from the python scripts.

2. Download & Install DSI-Studio and copy the install path into `.../bin/3.2_DTIConnectivity/dsi_studioPath.txt`

3. Download & Install FSL 5.0.1

4. Download & Install Cmake, open the software and click in the upper menu on *Tools* → *How To Install For Command Line Use* like shown in Figure 2 and follow the steps



(a) Menu of Cmake software (b) Install for command line use

Figure 2: Installation instructions to install Cmake

5. Download & Install Python 3.6 or higher using Anaconda and enter the following command to install necessary packages
   `pip install nipype==1.1.2 lmfit==0.9.11 progressbar2==3.38.0 nibabel shutil` (without line break!)
   Anaconda will tell you if additional packages are necessary. We recommend to install AIDAmri in a separate Anaconda environment (see the related documentation).

6. Install NiftyReg by conducting the following steps:

   a. Generate your source folder `.../NiftyReg/niftyreg_source`

   b. Download NiftyReg from the git by replacing `<path>` by your personal path and enter the following command
   `git clone`
   `git://git.code.sf.net/p/niftyreg/git <path>/niftyreg_source`
   (without line break!)

   c. Change the folder by typing in the terminal:
   `cd <path>/niftyreg_source`

   d. Type in the command line
   `git reset --hard 83d8d1182ed4c227ce4764f1fdab3b1797eecd8d`

   e. Follow the steps described here

3 Functions

List of functions:

- **1_PV2NIfTiConverter**: Bruker to NIfTy converter
- **2.1_T2PreProcessing:** T2w MRI pre-processing including brain extraction, bias field correction and atlas registration
- **2.2_DTIPreprocessing:** DTI pre-processing including brain extraction, bias field correction and atlas registration
- **2.3_fMRIPreProcessing**: DTI pre-processing including brain extraction, bias field correction and atlas registration
- **3.1_T2Processing**: Stroke mask calculations across all subjects per group (Incidence mapping), SNR calculations.
- **3.2_DTIConnectivity**: Whole brain fiber tracking using DSI Studio and calculation of diffusion measures (FA, AD, RD, MD) for every brain region.

- **3.2.1_DTIdata_extract**: Write a txt file with DTI values from all brain regions.
- **3.3_fMRIActivity**: functional connectivity analysis for all atlas regions
- **4.1_ROI_analysis**: Analysis of T2w, DTI, and rs-fMRI with user-defined atlas regions, e.g.: peri-infarct regions around the stroke lesion.

Attention: All program examples are only listed with the mandatory input parameters. For more details/help, call python `.../python <command> -h`. The command line examples are given with the identifier `testData<No.>.nii.gz` and can be identically applied to other data. The test data is freely available: https://doi.org/10.12751/g-node.70e11f.

After a successful download, you can choose to either process single files manually or automate the processing for the whole dataset. In both cases, processing includes file conversion from the raw Bruker format into the NIfTI format, several preprocessing steps and registration with the Allen Brain Reference Atlas. The functions in `/bin` are named accoring to the MRI sequence to be processed (T2, DTI, fMR) and labeled with 1-4.1 in the order of processing: for example `3.1_T2Processing` requires `1_PV2nIfTiConverter` and `2.1_T2PreProcessing` to be executed sequentially.

4 Batch processing

To process the whole project folder at once, only two scripts are necessary: 1) creates a new project folder and converts all files to the NIfTI format, and 2) The second script applies pre-processing steps and the registration with the atlas. Remember that our test data set already is converted into the NIfTI format. In this case only the second script needs to be applied. In General, the raw (Bruker format) data needs to be in the following structure: projectfolder/days/subjects/data/

To convert the whole project folder into the NIfTI format, open the terminal and change the directory to the `/bin` folder of the AIDAmri installation using `cd <path to AIDAmri>/bin`

Enter a similar line as in the example below to start the the first script (Bruker to NIfTI conversion). For the first script to work, the table `groupMapping.csv` within `/bin` needs to be adjusted beforehand: specify the group name of every subject within the project folder (e.g. using Excel). This is necessary for the script to properly generate the new processed project folder structure. Remember that the algorithm does not read the first row of the table containing the titles (Subject, Group).

Example:
```
python conv2Nifti_auto.py -f /path/to/raw_dataset
-d Baseline P1 P7 P14 -g Group1 Group2
```
(without line break!)

As you can see, this script computes the conversion either for all data in the raw project folder or for certain days and/or groups specified through the optional arguments -d and -g. During the conversion a new folder called proc_data is being created in the same directory where the raw data folder is located.

After a successful Bruker to NIfTI conversion, the second script can be applied to the new project folder proc_data.

Example:
```
python batchProc.py -f /path/to/proc_data -g Group1 Group2
-d Baseline P1 P7 P14 -t T2w DTI fMRI
```
(without line break!)

This script runs every necessary script for all (pre-)processing and registration steps. The data needs to be ordered like after the Bruker2NIfTI conversion: projectfolder/days/groups/subjects/data/. Again, you can specify which days -d and groups -g to compute and which dataformat -t (T2w, DTI, fMRI) to process. Please keep in mind that the hereby executed scripts are related to each other and therefore T2w always needs to be specified before DTI.

Dependent on the size of your project, this process may take a while. After finishing, your project folder is ready for network graph analysis, e.g. using AIDAconnect.

5 Processing single files step-by-step

Open the `AIDA_gui.py` in the `/bin` folder. You can use the GUI to process the test data set, which represents one subject on one day of measurement. Type the following command line

```
python AIDA_gui.py
```

to open the window shown in Figure 3.

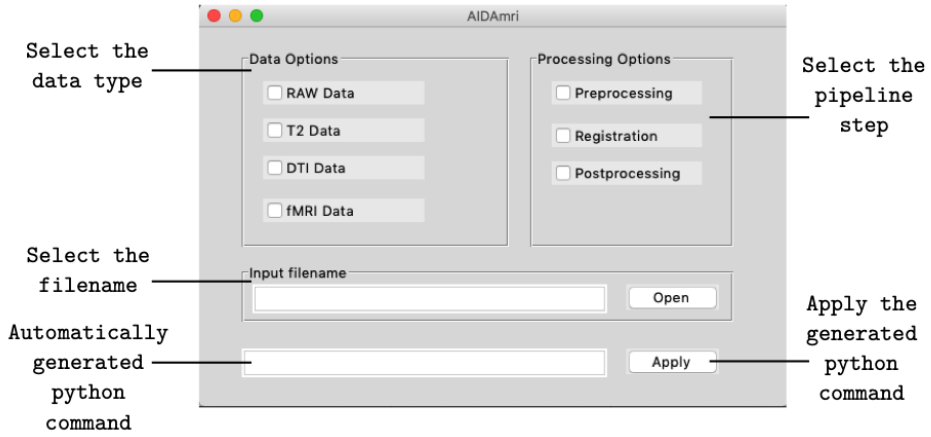
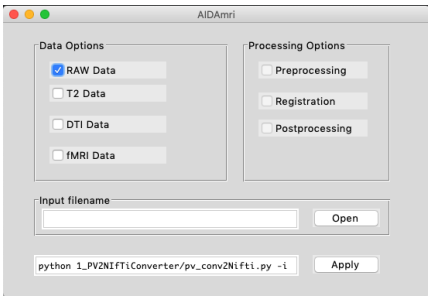


Figure 3: Description of the user interface to process the data step by step.

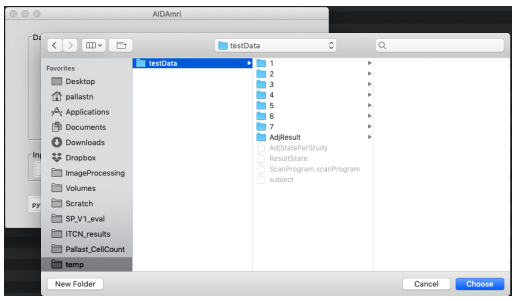
5.1 Convert raw data

Convert Bruker raw data to NIfTI files by specifying the folder containing all raw folders of each scan (see Figure 4). A file with exactly the same name is created in the given input folder. It contains all sorted NIfTI files. The raw data should have the same orientation as the example dataset.

```
python pv_conv2Nifti.py -i .../testData
```



(a) Options and modes



(b) Selected file

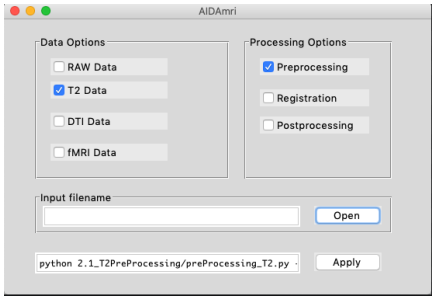
Figure 4: Convert raw data to NIfTI files by specifying the folder containing all raw folders of each scan

Remember to move the new generated file to a new project folder, if you want to separate the raw Bruker files with the processed NIfTI files. We recommend to build a folder structure as follows: projectfolder/days/groups/subjects/data/ especially if you want to use AIDAconnect for graph analysis.

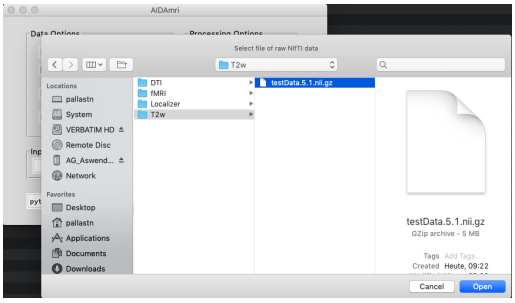
5.2 Processing of T2w & T2mapping data

Apply the reorientation, bias field correction and brain extraction to the T2w data set. The automatically attached endings of the processed filenames indicate which steps have been performed. Brain extraction should be of good quality and must be manually checked or corrected by adapting the default parameter (see Figure 5).

```
python preProcessing_T2.py -i .../testData/T2w/testData.5.1.nii.gz
```



(a) Options and modes



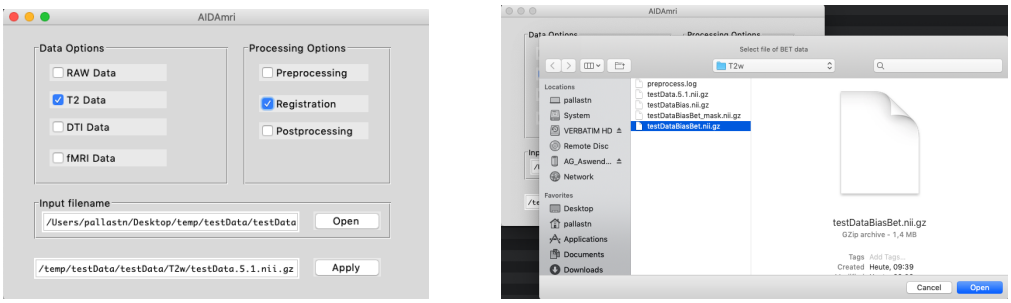
(b) Selected file

Figure 5: Apply the preprocessing to the T2 data by selecting the correct filename and options

The registration will also work without the following step: You can segment a region by taking the brain extracted dataset as reference (ends with `...BET.nii.gz`). We recommend to conduct this step with itk-SNAP. The saved file should end with the extension `...Stroke_mask.nii.gz`

The next step includes the registration of the Allen Brain Reference Atlas with the brain extracted T2 dataset. The result is a variety of files. An impression of the registration can be obtained by superimposing the brain extracted file with the annotations of the Allen Brain (ends with `..._Anno.nii.gz`) (see Figure 6)

```
python registration_T2.py -i .../testData/T2w/testDataBiasBet.nii
```

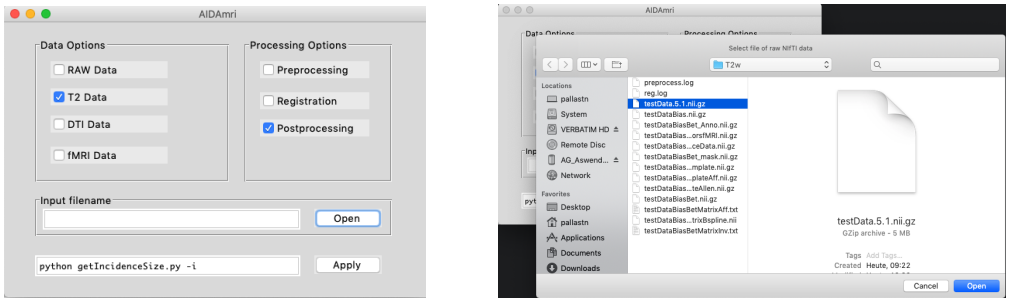


(a) Options and modes (b) Selected file

Figure 6: Apply the registration between the T2 data and the ARA by selecting the correct filename and options

If you previously defined a region of interest, the region size, segmented parental ARA regions and segmented original ARA regions can be determined in that step. Here, the segmented region `.../Stroke_mask.nii.gz` is overplayed with the Allen Brain Reference Atlas and saved in the file `...Anno_mask.nii.gz`. You do not need to enter single files, but the path to the `.../T2w` folders (see Figure 7)

```
python getIncidenceSize_par.py -i .../testData/T2w
python getIncidenceSize.py -i .../testData/T2w
```



(a) Options and modes (b) Selected file

Figure 7: Get the previously defined region of interest, the region size, segmented parental regions and segmented original regions by selecting the correct filename and options

The results, such as affected regions and ROI volume are stored in the folder `.../T2w` in the following files:

```
affectedRegions.txt
affectedRegions.nii.gz
affectedRegions_Parental.txt
affectedRegions_Parental.nii.gz
```

5.3 Processing of T2 data

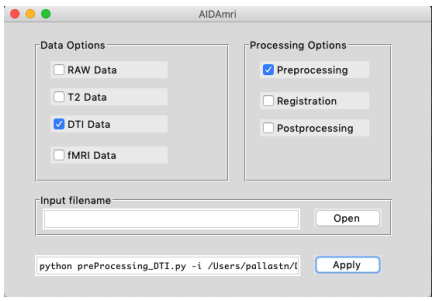
From the masks drawn on the T2-weighted images, it is possible to determine both the incidence map and the size of affective regions. For example, if a `day1` folder contains multiple `Mouse_1-Mouse_15` folders and the processed T2 data is in those folders, the command would be as follows

```
python getIncidenceMap.py -i .../day1 -s Mouse*
```

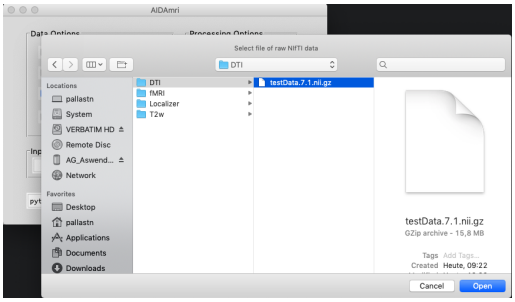
5.4 Processing of DTI data

The DTI processing procedure includes a dimension reduction, bias correction, a threshold application, and the subsequent brain extraction. The endings on the filenames indicate which steps have been performed (see Figure 8).

```
python preProcessing_DTI.py -i .../DTI/testData.7.1.nii.gz
```



(a) Options and modes

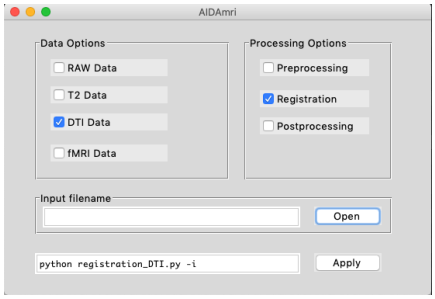


(b) Selected file

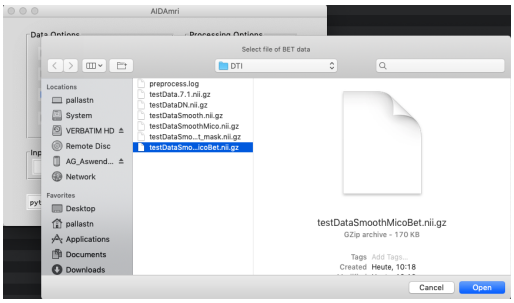
Figure 8: Apply the preprocessing to the DTI data by selecting the correct filename and options

The next step includes the registration of the Allen Brain Reference Atlas with the brain extracted DTI dataset. For processing a reference (stroke) mask, two options are available: a) Registration of a reference mask that is related to another dataset/day, e.g. to always use the same mask, append command `-r <filename of ref>` b) else, the algorithm will automatically use the corresponding reference mask from the respective subject folder (see Figure 9). If no mask is defined, the registration will proceed without.

```
python registration_DTI.py -i .../DTI/testDataSmoothMicoBet.nii.gz
```



(a) Options and modes

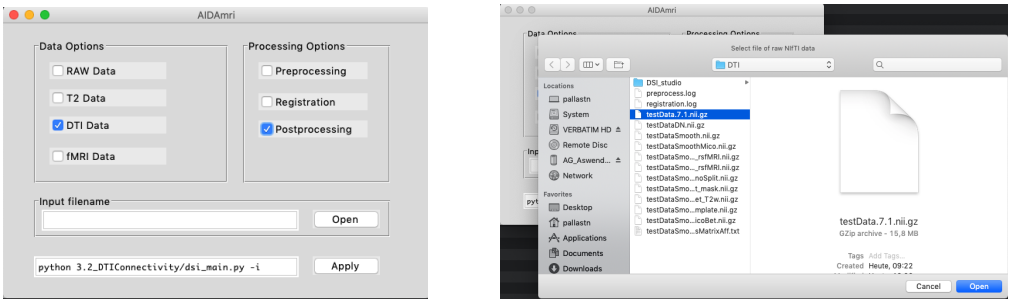


(b) Selected file

Figure 9: Apply the registration between the DTI data and the ARA by selecting the correct filename and options

The connectivity is finally calculated using DSI-Studio. All connectivity matrices are based on the reference atlas (see Figure 10).

```
python dsi_main.py -i .../DTI/testData.7.1.nii.gz
```



(a) Options and modes (b) Selected file

Figure 10: Process the DTI data with respect to the ARA regions by selecting the correct filename and options

The connectivity matrices of the parental ARA, the original ARA and the related ROI are stored in the folder `.../DTI/connectivity` as `.txt` and `.mat`. DSI-Studio differentiates between matrices that count how many fibers pass and end in each region. The adjacency matrices can be visualised using the related plot function:

```
python plotDTI_mat.py -i .../testData/fMRI/connectivity/testData*
.connectivity.mat
```
(without line break!)

The folder `.../DTI/connectivity` also contains the diffusion value maps (e.g. FA map) registered with the atlas. This data can be extracted and saved as `.txt` with the region name and the corresponding FA/RD/MD/AD using the function:

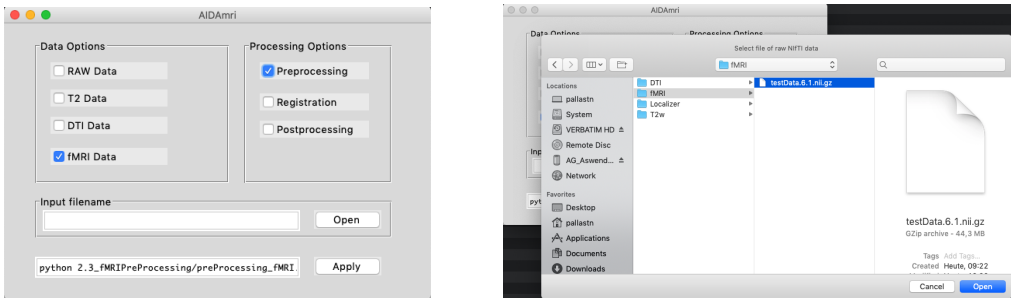
```
python DTIdata_extract.py image_file roi_file
```

in the 3.2.1_DTIdata_extract folder. To iteratively process all subjects use the `iterativeRun.py` function.

5.5 Processing of fMRI data

The fMRI processing is roughly comparable to the preprocessing of the DTI datasets. Brain extraction should be of good quality and must be manually checked or corrected by adapting the given parameters (see Figure 11).

```
python preProcessing_fMRI.py -i .../fMRI/testData.6.1.nii.gz
```

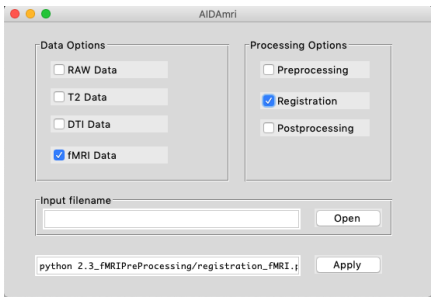


(a) Options and modes (b) Selected file

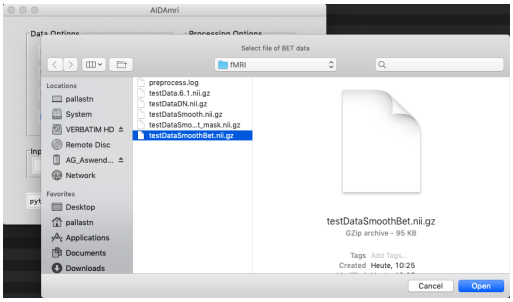
Figure 11: Apply the preprocessing to the rsfMRI data by selecting the correct filename and options

The step includes the registration of the Allen Brain Reference Atlas with the brain extracted fMRI dataset. The result is a variety of files. An impression of the registration can be obtained by superimposing the brain extracted file with the annotations of the Allen Brain (ends with `..._Anno.nii.gz`) (see Figure 12)

```
python registration_fMRI.py -i .../testData/fMRI/testSmoothBet.nii
```



(a) Options and modes

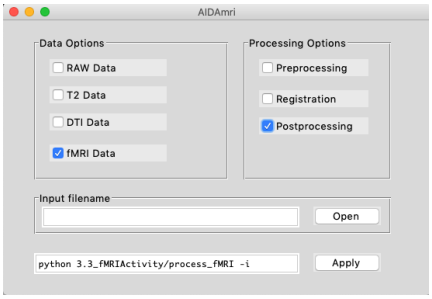


(b) Selected file

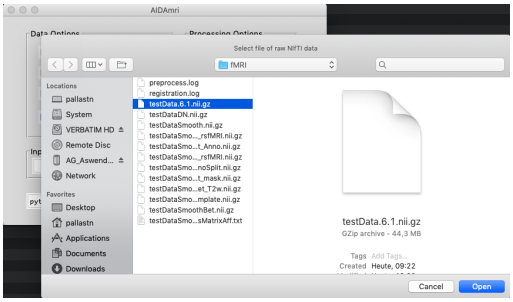
Figure 12: Apply the registration between the rsfMRI data and the ARA by selecting the correct filename and options

If physiological data are not available, the step will be conducted without the included regression. All activity matrices are based on the reference atlas (see Figure 13).

```
python process_fMRI -i .../fMRI/testData.6.1.nii.gz
```



(a) Options and modes



(b) Selected file

Figure 13: Process the rsfMRI data with respect to the ARA regions by selecting the correct filename and options

The activity matrices of the parental Atlas (original Atlas) are stored in the folder `.../fMRI/regr` as `.txt` and `.mat` with the prefix `MasksTCs.` and `MasksTCsSplit.`. The related adjacency matrices can be visualised using the related plot function:

```
python plotfMRI_mat.py -i .../testData/fMRI/regr/MasksTCsSplit*.mat
```

5.6 Peri-infarct ROI analysis

You can create custom peri-infarct masks to further analyze stroke related regions. Go to the folder `bin/4.1_ROI_analysis` and open the file `proc_tools.py` with an arbitrary editor that can open python files. Adjust all directories, paths and further specifications as described in the script. To decide which regions to include in the peri-infarct region, modify:

`cortex_labels_1.txt` and `cortex_labels_2.txt`

(for the full list of atlas labels, see `../lib/annoVolume+2000_rsfMRI.nii.txt`)

Proceed with the scripts in the order 1 to 4. The first script creates peri-infarct masks for all time points:

```
python 01_dilate_mask_process.py
```

The second script aligns the peri-infarct masks in the rsfMRI and DTI space:

```
python 02_apply_xfm_process.py
```

The result of the third script depends on the imaging type: In case of rsfMRI, a MATLAB file which contains two text files is being created. 1) for each region one column with the averaged rsfMRI time series and 2) the atlas labels names. In case of DTI, a modified atlas labels file which includes individually shaped peri-infarct brain regions is being created. These new generated regions replace the original regions in the file:

```
python 03_create_seed_rois_process.py
```

The fourth script is not mandatory, but a helper tool to compare the number of voxels included in the peri-infarct region for each subject.

```
python 04_examine_rois.py
```

Attention: The scripts for peri-infarct ROI analysis are provided for analysis of time point 7 only (e.g. 7 days post stroke). For other time points, manual modifications are necessary.