

**Task Force 2.2: ASL Functionalities-Test Report**

* Motion correction

**Function name:** batch\_realign.m

**Description:** realigns raw EPI images using SPM realignment tools

**Inputs:** 4D NIfTI

**Outputs:** This script seems to be one code from a whole pipeline. Therefore, there is dependency on passing the file path and subjects.

**Syntax:** DNA

**Function name:** ASL\_realign.m

**Description: motion correction**

**Inputs: path\_temp:** the path of the ASL image**, name\_asl:name of the ASL image**

**Outputs:** Generates realigned images from the time series

**Syntax:** P = ASL\_realign(path\_temp, name\_asl)

* Brain extraction/skull stripping

**Function name:** ASLtbx\_createbrainmask.m

**Description:** use a single threshold to extract the brain, can manually change the threshold by the user

**Inputs:** full path to the input image as P='C:\Users\...\M02.nii';

**Outputs:** a brain mask file named 'brainmask.nii' with the same path as input

**Notes:** This file needs modification: It seems spm\_write\_vol cannot write the mask well when the mask is only 0 and 1. Added several lines for modification

**Syntax:** maskimg=ASLtbx\_createbrainmask(P)

**Function name:** batch\_create\_mask.m

**Description:** Use a single threshold to extract the brain region, can manually change the threshold by the user

**Inputs:** full path to the input image as PF='C:\Users\...\M02.nii';

**Outputs:** a brain mask file named 'mask\_perf\_cbf.img' with the same path as input

**Notes:** This script seems to be one code from a whole pipeline. Therefore, there is dependency on passing the file path and subjects. But if we only took the core code for the test, this function is the same as the previous function . And need the same modification as in previous script

**Syntax:**

**Function name:** ASL\_getBrainMask.m

**Description: motion correction**

**Inputs:** imgtpm,imgfile,flag\_addrealignmsk

**Outputs: two masks are generated:** brnmsk\_dspl (datatype, logical), used for cbf map display;brnmsk\_clcu (datatype, logical), used for glo\_cbf calculation.

**Syntax:** [brnmsk\_dspl, brnmsk\_clcu] = ASL\_getBrainMask(imgtpm,imgfile,flag\_addrealignmsk)

* Segmentation of high-resolution images (Grey Matter and White Matter)

**Function name:** ASLtbx\_spmnsegment

**Description:** Probabilistically segment the high resolution T1 images

**Inputs:** High resolution T1 image

**Outputs:** Probabilistic segmentations of GM, WM and CSF (file names starting with c1, c2 and c3)

**Syntax:** ASLtbx\_spmnsegment(full\_path\_of\_T1\_Image)

* Co-registration of ASL images to high resolution T1-weighted, T2-weighted, and/or FLAIR images

**Function name:** ASLtbx\_spmcoreg.m

**Description:** using spm for registering source and allimgs to target.

**Inputs:** target, source, allimgs

**Outputs:** co-registered images

**Syntax:** [output]=ASLtbx\_spmcoreg(target, source, allimgs)

**Function name:** ASL\_coreg

**Description:** coregister source to target and apply transform matrix to source and varargin

**Inputs:** target,source,varargin

**Outputs:** co-registered images and transformation matrix

**Syntax:** ASL\_coreg(target,source,varargin)

**Function name:** spm\_reslice\_yli

**Description:** similar to spm\_reslice, does co-registration

**Inputs:** P - matrix or cell array of filenames flags - a structure containing various options.

**Outputs:** co-registered image files to the same subdirectory with a prefix.

**Syntax:** spm\_reslice\_yli(P,flags)

* Cerebral Blood Flow (CBF) quantification (please specify the model used)

**Function name:** asl\_perf\_subtract.m / asl\_perf\_subtractMB.m

**Description:** For CASL and PASL,generate the difference and calculate the perfusion images from control/label and M0 images

**Inputs:** Please see the description in the MATLAB code for detailed information.

**Outputs:** Will output CBF maps into the path you defined

**Syntax:** One example for PASL quantification. Both P and M0img are paths for control/label and M0 images: FAIR='C:\Users\...\FAIR\_CL.nii'.

perfnum = asl\_perf\_subtract(FAIR,1, 0, 0, [0 0 1 0 0 1 0 1 0 0], ...

0.5, 0, 0.9, 1, 2, 0.8, 45, 20, M0img);

perfnum = asl\_perf\_subtractMB(FAIR,1, 0, 0, [0 0 1 0 0 1 0 1 0 0], ...

0.5, 0, 0.9, 1, 2, 0.8, 45, 20, M0img);

**Function name:** ASLQuantificationPCASL.m

**Description:** For pCASL, calculate the perfusion maps with difference images, M0 and related parameters according to the 2015 ASL white paper

**Inputs:** data, M0 and mask should be the matrix for pCASL difference, proton density and brain mask images

**Outputs:** Will generate CBF maps in MATLAB workspace

**Notes:** This script needs small modification as shown in the script.

**Syntax:** [output, output\_nantmean, NANCOUNTMASK] = ASLQuantificationPCASL(data, M0, 1.8, 1.8,1.6, 0.9, 0.6,'no','',14, 1, mask, 'yes', 'yesfigure');

**Function name:** batch\_calc\_rcbf.m

**Description:** Calculate the normalized perfusion maps divided by the mean flow

**Inputs:** Choose perfusion maps and mask in the pop out windows

**Outputs:** Will generate normalized CBF maps in same path as original CBF maps

**Notes:** This script seems to be one code from a whole pipeline. Therefore, there is dependency on passing the file path and subjects with spm batch system. But can use the core code lines as a separate function.

**Syntax:** Just run the whole script, no function to call.

**Function name:** ASLDeltaM.m

**Description:** Calculate the difference images from the control and label images

**Inputs:** CTRL and TAG are matrix values from the control and label images of

ASL images, method can be any value as shown in the script

**Outputs:** Output will be a matrix for difference images in MATLAB workspace

**Syntax:** Output=ASLDeltaM(CTRL,TAG,'PairWise');

**Function name:** batch\_perf\_subtract.m

**Description:** batch file for the same function as asl\_perf\_subtract.m

**Function name:** ASL\_calculateCBFmap.m

**Description:** This function obtains CBF value in the unit of ml/100g/min

**Inputs:** path\_temp: the paths of the ASL images, asl\_paras: the parameters of the ASL images

**Outputs:** a CBF map in .img format is created in the specified path

**Notes:** the images are in .img format, instead of dicom or nifti

**Syntax:** ASL\_calculateCBFmap(path\_temp,asl\_paras)

* Outlier removal or Denoising

**Function name:** ASLtbx\_aoc

**Description:** It adaptively removes CBF volumes from the CBF time series and averages the remaining volume to obtain the meanCBF map. The removal is based on estimated motion, and correlation between individual CBF volumes and mean CBF map. Algorithm based on

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3777751/

**Inputs:** CBF time series and optionally: text file with the estimated motion parameters realigning the raw ASL time series, brain mask in the ASL space, an algorithm specific threshold parameter (default 0.17) and number of iterations (default 2)

**Outputs:** Outlier volume numbers.Also generates the estimated CBF map obtained after outlier removal.

**Syntax:**

ASLtbx\_aoc(cbfimgs, motionfile, maskimg, th, iter)

See the help section of the code for details.

Use the **batch\_outlier\_clean** to call the function

**Function name:** ASLtbx\_aocSL

**Description:** It adaptively removes slices from CBF volumes in the CBF time series and averages the remaining slices and volumes to obtain the mean CBF map. Algorithm is based on

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3777751/

**Inputs:** CBF time series, c1img and c2img (GM and WM tissue probability maps obtained from SPM segmentation of high resolution T1 image, coregistered and downsampled to the ASL space), masking (brain mask in the ASL space), th (a threshold parameter, default =0.17, better to not change), motionfile (test file with the estimated motion parameters realigning the raw ASL time series)

**Outputs:** oidx: identified outliers. Also generates the estimated CBF map obtained after outlier removal.

**Syntax:** ASLtbx\_aocSL(cbfimgs, c1img, c2img, maskimg, th, motionfile)

**Function name:** batch\_SCORE\_clean.m

**Description:** Calculate

**Inputs:** The path of perfusion maps, GM, WM and CSFProbability maps.

**Outputs:** Will

**Notes:** This script seems to be one code from a whole pipeline. Therefore, there is dependency on passing the file path and subjects with spm batch system. But can use the core code lines as a separate function.

**Syntax:** Just run the whole script, no function to call. But does need to include the SCORE\_denoising.m function.

* Normalization to common template (e.g., MNI space)

**Function name:** ASLtbx\_spm12normest

**Description:** Normalize the structural image to the standard template (TPM.nii) using spm12 functions.

**Inputs:** strucimg should be paths of the target structural images

**Outputs:** normalized structural images in the assigned folder

**Notes:** The whole script runs fine, however, the normalized image is black and white.

**Syntax:** ASLtbx\_spm12normest(strucimg)

**Function name:** ASLtbx\_spm12normest

**Description:** Normalize the structural image to the standard template (TPM.nii) using spm12 functions.

**Inputs:** strucimg should be paths of the target structural images

**Outputs:** normalized structural images in the assigned folder

**Notes:** The whole script runs fine, however, the normalized image is black and white.

**Syntax:** ASLtbx\_spm12normest(strucimg)

* Deep learning-based method (Any application relevant to ASL)

**File name:** test\_ADNI\_w\_ft\_0630.py

Description: Denoise ASL data in ADNI dataset with fine-tuning model presented in “Improving Sensitivity of Arterial Spin Labeling Perfusion MRI in Alzheimer's Disease Using Transfer Learning of Deep Learning-Based ASL Denoising”.

**Inputs:**

To keep the script easy to run we just hard coded the input within the script. There are three inputs you will need to run the script.

1. A “subjectlist\_valid.txt” file with subject name.
2. A directory containing all the subjects that need to be denoised. Each folder in this directory is one subject. In each subject folder, there are three subfolders i.e., “ASL”, “ANAT”, and “ANAT\_DL\_mask”. “ASL” folder contains ASL data in native space. “ANAT” folder contains the structural images for preprocessing the data e.g., registration or normalization. “ANAT\_DL\_mask” contains the gray matter, white matter, and the cerebrospinal fluid.
3. A trained deep learning model.

**Outputs:**

Denoised ASL data in each subject’s “ASL” folder.

**Syntax:** python test\_ADNI\_w\_ft\_0630.py

**File name:** test\_ADNI\_w\_Direct\_FT\_0630.py

**Description:** Denoise ASL data in ADNI dataset with direct transfer learning model presented in “Improving Sensitivity of Arterial Spin Labeling Perfusion MRI in Alzheimer's Disease Using Transfer Learning of Deep Learning-Based ASL Denoising”. The main difference between this script and “test\_ADNI\_w\_ft\_0630.py” is that the models used for denoising are different.

**Inputs:**

To keep the script easy to run we just hard coded the input within the script. There are three inputs you will need to run the script.

1. A “subjectlist\_valid.txt” file with subject name.
2. A directory containing all the subjects that need to be denoised. Each folder in this directory is one subject. In each subject folder, there are three subfolders i.e., “ASL”, “ANAT”, and “ANAT\_DL\_mask”. “ASL” folder contains ASL data in native space. “ANAT” folder contains the structural images for preprocessing the data e.g., registration or normalization. “ANAT\_DL\_mask” contains the gray matter, white matter, and the cerebrospinal fluid.
3. A trained deep learning model.

**Outputs:** Denoised ASL data in each subject’s “ASL” folder.

**Syntax:** python test\_ADNI\_w\_Direct\_FT\_0630.py

**Function name:** read\_subject\_list

**Description:** This is a helper function used in “test\_ADNI\_w\_Direct\_FT\_0630.py” and “test\_ADNI\_w\_ft\_0630.py”. The function is used to read the subject list in a “.txt” file.

**Inputs:** A “.txt” file with each line denotes a subject name.

**Outputs:** A list of all subjects.

**Syntax:** content = read\_subject\_list('subjectlist\_valid.txt')

**Function name:** get\_subj\_c123

**Description:** This is a helper function used in “test\_ADNI\_w\_Direct\_FT\_0630.py” and “test\_ADNI\_w\_ft\_0630.py”. The function is used to get the gray matter, white matter, and the cerebrospinal fluid of each subject.

**Inputs:** A subject name and a folder containing the subject name.

**Outputs:** A mask with the gray matter, the white matter, and the cerebrospinal fluid.

**Syntax:** mask = get\_subj\_c123(subject\_name, folder\_name)

**Function name:** parse\_args

**Description:** Parser for command-line options, arguments and sub-commands. The function is used to set up the deep learning model’s folder and name.

**Inputs:** A directory of the deep learning model and the model’s file name.

**Outputs:** An instance of the parser.

**Syntax:** args = parse\_args()

**Function name:** predict\_nii\_w\_masks\_rbk

**Description:** The function is used to denoise all the subjects’ ASL data within a folder by using a deep learning model.

**Inputs:** A deep learning model, the root folder of all subjects, and the name of the denoised results.

**Outputs:** Denoised results.

**Syntax:** predict\_nii\_w\_masks\_rbk(model,root\_folder,save\_name)

* Create ROIs

**Function name:** roipoly.m

**Description:** selects a polygonal region of interest within an image, this function should be similar to MATLAB’s built-in function “roipoly”.

**Inputs:** A 2D image file name

**Outputs:**  returns a binary image that can be used as a mask for masked filtering, returns the polygon coordinates

**Syntax:** [BW, xi, yi] = roipoly(‘image.jpeg’);

* Smoothing

**Function name:** ASLtbx\_smoothing

**Description:** smoothes an image, returns the smoothed image

**Inputs:** image path, and a kernel size

**Outputs:** smoothed image, written as a different file

**Syntax:** ASLtbx\_smoothing(‘image.nii’, 2)

**Function name:** batch\_smooth\_wconfile and batch\_smooth

**Description:** A code from pipeline. Needs dependency from the spm batch. It uses spm\_smooth to smooth a number of files.

**Inputs:** path of the images

**Outputs:** smoothed images

**Syntax:** As this is part of the pipeline,just run the script, no function to call

* Any additional functionalities with appropriate description

**Function name:** ASL\_calculateDiffMap

**Description:** Calculate difference map and write out the .img/.hdr files

**Inputs:** path\_temp, asl\_paras

**Outputs: difference image and header files**

**Syntax:** ASL\_calculateDiffMap(path\_temp,asl\_paras)