Laminar fMRI pipeline

Documentation

Tommy Clausner

last updated on March 11, 2018

Contents

List of Figures

List of Tables

1 Pre- requisites

- MRIcron (https://www.nitrc.org/projects/mricron)
- FSL (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki)
- FreeSurfer (https://surfer.nmr.mgh.harvard.edu/)
- MrVista (https://web.stanford.edu/group/vista/cgi-bin/wiki/index.php/MrVista)
- SPM (http://www.fil.ion.ucl.ac.uk/spm/)
- analysePRF (http://kendrickkay.net/analyzePRF/)
- knkutils (https://github.com/kendrickkay/knkutils/)
- Open fMRI Analysis (https://github.com/TimVanMourik/OpenFmriAnalysis)
- MATLAB R2015a or later (https://nl.mathworks.com/products/matlab.html)
- MATLAB Robotic Systems Toolbox (https://nl.mathworks.com/products/robotics.html - can hopefully be omitted in future releases)

Use e.g. MRIcron's dicom2nii to convert raw MRI data to NifTi files.

It might be necessary to modify initMrVista.m such that the first two lines point towards the mrVista and SPM master directory.

2 Folder structure

The general folder structure is set up by a call to setupfolders.sh (see also Figure 4)

- Functional data must be located in niftis/**functionals**/
- Anatomical data must be located in niftis/**t1**/
- Inverted functional data must be located in niftis/**inverted**/
- Proton density data must be located in niftis/**pd**/
- Inverted proton density data must be located in niftis/**pdinverted**/

3 How things work

Parts of the analysis run in Bash shell scripts. Those can be called from their root directory using "sh scriptname.sh" (excl. quotation marks). Each step within the analysis has a different subfolder containing the respective results of this step. Leading numbers indicate in which logical order the corresponding shell scripts should be executed. Note that all files created by each respective step is stored in the coresponding folder. However files that need to be preset (e.g. config files) must be located in A_helperfiles.

All functions with the prefix *do_* will affect the data.

4 Suggested order of operations

4.1 Pre-processing

- makenewsubject.sh (double click)
- copy NifTi files to the correct location (see above)
- open terminal (the following are terminal commands)
- cd PATH/SUBJECTFOLDER/B_scripts
- sh runonqsub.sh 32gb do_preparefunctionals.sh
- sh runonqsub.sh 16gb do_fsrecon.sh
- sh runonqsub.sh 16gb do_makemasksandlabels.sh
- sh runonqsub.sh 32gb do_realignment.sh
- sh runonqsub.sh 16gb do_preparecoregistration.sh
- sh do_correctavgdiff.sh 0.975
- sh do_coregistration.sh
- sh runonqsub.sh 32gb do_distcorr.sh
- sh do_prepareapplytopup.sh
- sh runonqsub.sh 64gb do_applydistcorr.sh

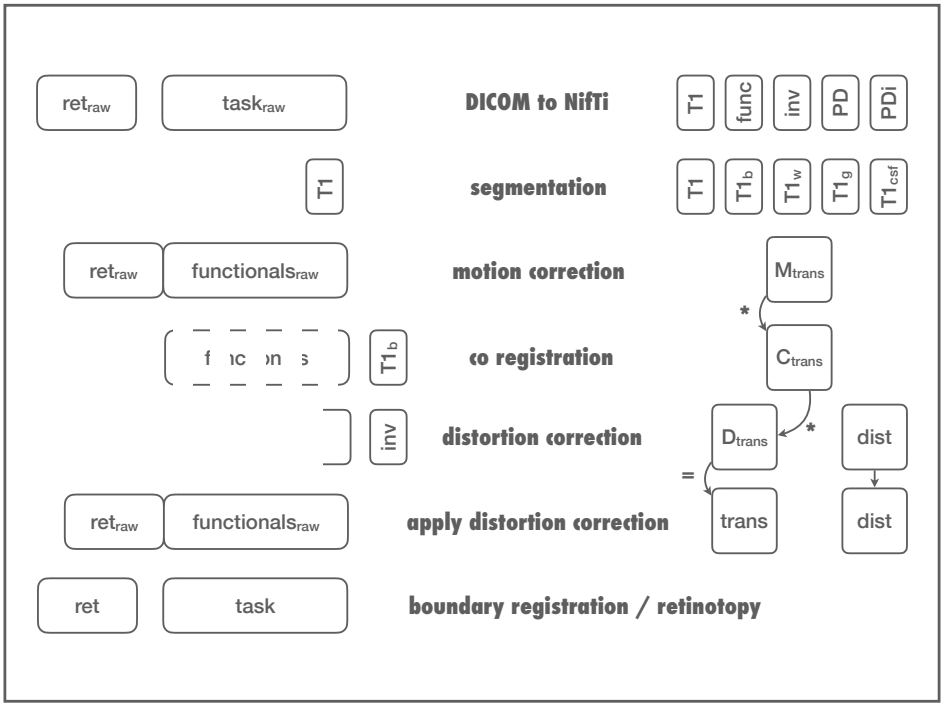

Figure 1: Overview Pre-processing fMRI

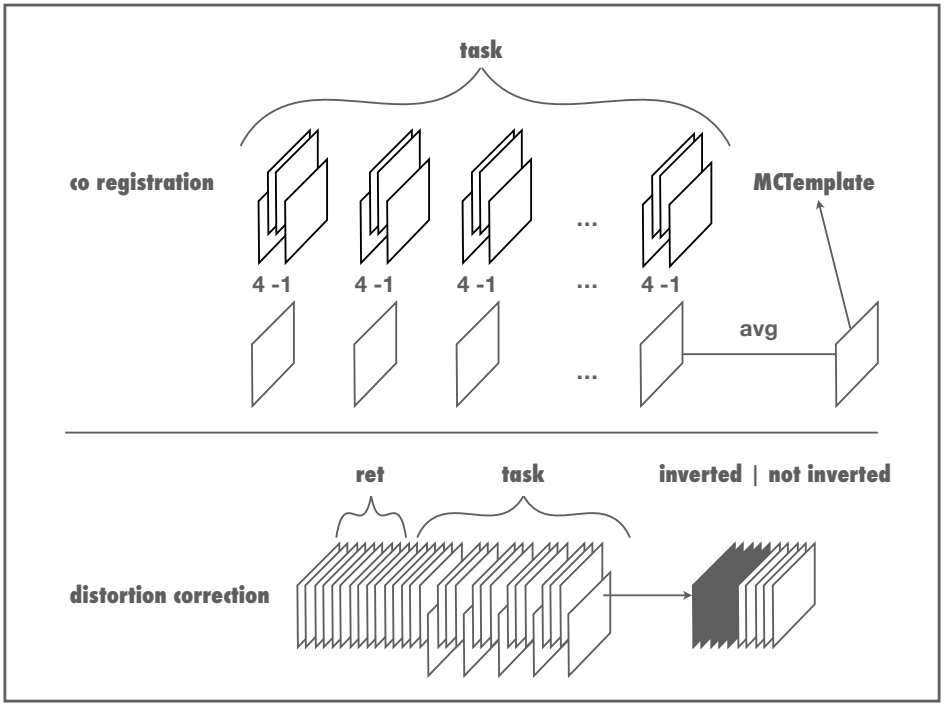

Figure 2: Coregistration and distortion correction visual depiction

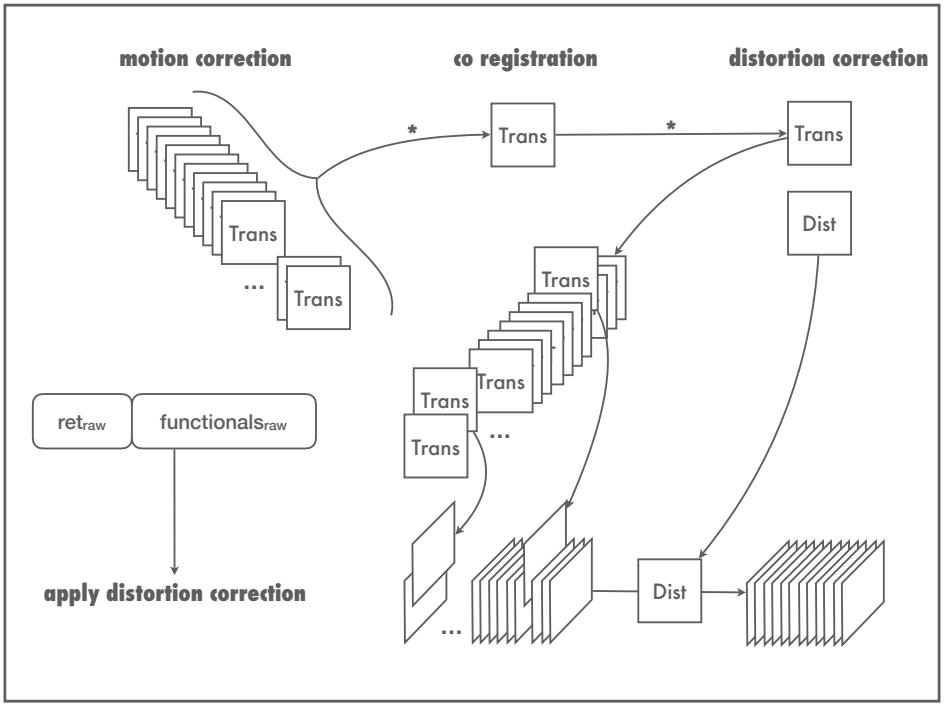

Figure 3: Apply corrections from all steps at once

4.2 Retinotopy

- sh runonqsub.sh 32gb do_prepareretinotopy.sh
- sh do_tseriesinterpolation.sh 180gb
- sh do_analyzePRF.sh 180gb

4.3 Functional laminar analysis

5 qsub

You can run the respective scripts directly in a qsub instance:
When running the computation on the cluster use the wrapper function runonqsub.sh

Important:
runonqsub.sh does not work with do_finecoregistration.sh (this function sets up a qsub session by itself). Please find more information below. The reason is that do_finecoregistration.sh initiates a MATLAB session with the correct requirements and forwards this to qsub

the following requirements on memory and computation time can be expected:

script	memory required	time required
do_preparefunctionals.sh	32GB	$\approx 10min$
do_fsrecon.sh	16GB	$\approx 6h$
do_makemasksandlabels.sh	16GB	$\approx 1min$
do_realignment.sh	32GB	$\approx 30min$
do_preparecoregistration.sh	16GB	$\approx 4.5h$
do_correctavgdiff.sh	4GB	$\approx 3sec$
do_coregistration.sh	8GB	$\approx 30min$
do_distcorr.sh	32GB	$\approx 15min$
do_applydistcorr.sh	64GB	$\approx 30min$
do_prepareretinotopy.sh	32GB	$\approx 4min$
do_tseriesinterpolation.sh	180GB	$\approx 7h$
do_analyzePRF.sh	250GB	$\approx 7h$
meanT.sh	16GB	$\approx 2min$

Table 1: Approximated time and memory requirements for the respective script to run. Note that everything that is more than 4GB must be run on the cluster. Use runonqsub.sh for that purpose.

Figure 4: Initial folder structure as required for the analysis

6 Helper files

There are several helper files that are needed in order to perform the analysis:

6.1 acquisition_parameters.txt

indicating the respective acquisition parameters per volume needed in order to perform the distortion correction. One line indicates the respective parameter setting for the respective volume (1st line corresponds to 1st volume, etc.)

e.g.:
0 1 0 0.042
0 -1 0 0.042

The first 3 columns indicate the respective phase coding direction the last column some weird value, that is only important if it changes. Otherwise it's fine to use any value as long as it is the same.

6.2 b02b0.cnf

Settings for distortion correction.

Needed in order to set the parameters for the distortion correction (example provided by FSL).

6.3 RetStim.mat

Stimuli used for retinotopy scans. Stimuli file must be such that each frame is represented as a binary image. The matrix must be of shape $Y \times X \times T$ where X and Y are the image dimensions in pixel and T is the respective number of frames.

7 Scripts explained (support for runonqsub.sh: Q)

7.1 runonqsub.sh

Initiates a qsub session and sends it to the cluster. This is necessary, since scripts running on the cluster are copied to a different directory, but within scripts all directories are relative. runonqsub.sh must hence be run in a "normal" session not using qsub, giving the respective script that was intended to run as an argument. runonqsub will then set everything up correctly. Further you have to specify the amount of memory needed.

example: sh runonqsub.sh 32gb myscript.sh

Additional arguments can be passed after the script.

example: sh runonqsub.sh 16gb meanT.sh csfmask

Under the hood:

- a qsub session is initiated like so: qsub -l walltime=12:00:00,mem=$1 -F "$DIR ${@:3}" $DIR/$2
- $DIR is the current absolute path to B_scripts
- $1 is the amount of memory used (e.g. 32gb)
- $2 is the respective script to run (e.g. myscript.sh)
- ${@:3} is all following arguments that are parsed to the script
- every time runonqsub.sh is called the absolute path is forwarded to the script (-F $DIR) in order to keep the relative dependencies clear

7.2 cleanscriptsfolder.sh

Since qsub puts a output + error log into /B_ scripts cleanscriptsfolder.sh can be used to move all qsub outputs to /B_scripts/qsuboutput

7.3 makenewsubject.sh

executable script (double click to execute)

Creates a new subject (S#) folder structure in order to get all folder dependencies right. It automatically detects folders called "S*number*" and creates a new folder incrementing *number* by one

7.4 setupfolders.sh (Q)

Sets up the necessary folder structure for the analysis (within a subject folder). This function is automatically called when using makenewsubject.sh

example: sh setupfolders.sh

7.5 checkhelperfiles.sh (Q)

Checks whether all necessary helper files are existing needed in order to perform the analysis.

example: sh checkhelperfiles.sh

7.6 do_preparefunctionals.sh (Q)

Prepares functional data, i.e. removes the first 4 and the last x volumes of every set of functionals, where x is the number of volumes acquired after the stimulation ended. If files were modified (i.e. if volumes were deleted), the original file will remain in /niftis/functionals/old

example: sh do_preparefunctionals.sh

7.7 do_fsrecon.sh (Q)

Performs segmentation using Freesurfer.

If running on OSX the script assumes:
FREESURFER_HOME=/Applications/freesurfer

If running on linux the script assumes:
FREESURFER_HOME=/opt/freesurfer/version

The target image that is used for the segmentation must be located in /niftis/t1

example: sh do_fsrecon.sh

Under the hood:

- calls recon-all -i /niftis/t1/* -subjid 0_freesurfer -all
- all results are stored in /0_freesurfer

7.8 do_makemasksandlabels.sh (Q)

example: sh do_makemasksandlabels.sh

Uses FreeSurfer reconstruction to create masks for CSF, gray matter and white matter and creates labeled volume for use within retinotopy containing of left / right + gray / white matter.

If the orientation needs to be changed, parse the respective FreeSurfer compatible orientation parameter (https://surfer.nmr.mgh.harvard.edu/pub/docs/html/mri_convert.help.xml.html)

example: sh do_makemasksandlabels.sh RAS

7.9 do_realignment.sh (Q)

example: sh do_realignment.sh

Merges and motion corrects all files in /niftis/functionals using the average volume.

numberofvolumes.txt is written to /A_helperfiles giving information about which sets had how many volumes. The first column is coresponds to dim4 from fslinfo

Opens an instance of fsleyes to show the results.

Under the hood:

- fslmerge is called to create a combined .nii for all files in niftis/functionals/ - files are merged allong the 4th dimension
- a .txt file is created saving which files were merged and how many volumes were in there
- mcflirt is used on the combined data doing the motion correction based on the mean volume
- all results are stored in /1_realignment

7.10 do_preparecoregistration.sh (Q)

sets up mrVista session requirements, that is prepareing all files to be used with mrVista (defining Inplane anatomy, 3D Anatomy, functionals)

example: sh do_preparecoregistration.sh

7.11 do_correctavgdiff.sh (Q)

Due to the subtraction of the 1st from the 4th volume the area outside the brain has the highest value. In order to correct this the lowest value will be shifted to zero the a certain part of the higher values are cut (e.g. 0.975).

example: sh do_correctavgdiff.sh 0.975

The result should be checked using "fsleyes ../2_coregistration/Inplane/MCTemplate.nii.gz" to ensure that the area outside the brain is nulled. Note that it can happen that some areas within the brain are nulled as well. If they are not too wide spread they do not affect later coregistration.

7.12 do_coregistration.sh

It will submit a qsub session using MATLAB, that in turn will be using FreeSurfer (bbregister) and OpenFmriAnalysis to perform a linear and non-linear boundary registration. Movie files of the respective co-registration are stored in /C_miscResults
The file used to execute MATLAB commands is /B_scripts/do_coregistration.m Note that it will be modified using the correct folder settings, which yields tmp.m that will be called and removed after MATLAB was closed.

example: do_coregistration.sh

7.13 do_coregistration.m

Sets up environment and initializes mrVista session. The function is used within do_finecoregistration.sh

Resulting transformation matrices (normal and inverted) are stored in /2_coregistration

7.14 do_distcorr.sh (Q)

example: sh do_distcorr.sh

Uses the averge volume of the inverted set of images in /niftis/inverted as reference for *inverted*

Uses as many volumes n as there were in /niftis/inverted. Respective volumes are selected starting from the last going in $4\ x\ n$ steps backward. Hence the resulting average of this set was computed over the 4th volume of each of n last sets in the normal images. Simplified (all 1 are selected):

for n=5

0,0,0,0,...,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1

Both (normalavg and invertedavg) will be used to esimate the distortion correction.

Under the hood:

- reference volumes (normal + inverted) are selected in order to do the field distortion estimate (selection: see above).
- selected reference volumes are merged into a single file (/3_distcorrection/all_b0.nii.gz)
- topup is called using the merged (normal + inverted) .nii and performs field mapping accoriding to the specifications in /A_helperfiles/b02b0.cnf using the acquisition parameters specified in acquisition_parameters.txt

- all results are stored in /3_distcorrection

7.15 do_prepareapplytopup.sh (Q)

Sets up files for use with do_applydistcorr.sh

Motion correction and co-registration matrices are transformed into vector format (calling transmats2topup.sh).
To each of the motion correction files the transformation from the coregistration and distortion correction is added. The actual addition is done in mergetransforms.sh

The result is a topup compatible file (first row zeros, second row actual transformation) for each volume.

The result will look like this:

[0 0 0 0 0 0
$Trans_x$ $Trans_y$ $Trans_z$ Rot_x Rot_y Rot_z]

Individual matrices for use with do_applydistcorr.sh are stored in:

/2_coregistration/transmatstoapply/####.txt

7.16 transmats2topup.sh

Converts all transformation matrices contained within a specified folder to a 6 element vector having the following properties:

[$Trans_x$ $Trans_y$ $Trans_z$ Rot_x Rot_y Rot_z]

example: transmats2topup.sh <path relative to function>

The result of this transformation is stored in:

/<initial data folder>/topupformat/*

The wrapping is done using a MATLAB script (transmats2topup.m) running in the background. So since the function will load an instance of MATLAB in the background, inputting as many matrices as possible might be advisable.

Note that the function is called within do_prepareapplytopup.sh

7.17 transmats2topup.m

Wrapper function to call tc_transmat2vec.m for all files specified in the input. Since MATLAB is run in the background each call to a new function or script would initialize a new instance. By using this wrapper function MATLAB only has to be initialized once and processes all files in the respective input folder.

7.18 tc_transmat2vec.m

A hack of the MATLAB function rotm2axang.m (Robotics toolbox), which per default transforms a set of 3x3 rotation matrices into single rotations around x,y and z. Additional functionality in tc_transmat2vec.m includes readability for files compatible with dlmread.m Further the function outputs the translation. However it is necessary to supply a 4x4 transformation matrix or a string containing the filename of a file that is 4x4 transformation matrix in dlm readable format. The respective outcome is [axang,trans] with axang being the rotations around x,y,z in radians and trans being the translation along x,y,z

7.19 mergetransforms.sh

Adds input vector one element wise to input vector 2.

example: mergetransforms.sh file1.txt file2.txt

Note that the function is called within do_prepareapplytopup.sh

7.20 do_applydistcorr.sh (Q)

Applies distortion correction to all functional data.

example: sh do_applydistcorr.sh

Under the hood:

- distortion correction is applied to the full set of functionals using the OUTPUT of topup (topup [...] –out=OUTPUT) as well as acquisition_parameters.txt (Using –method=jac)
- all results are stored in /3_distcorrection

7.21 do_prepareretinotopy.sh (Q)

Prepares files for use with "do_retinotopy.sh". The function copies all functionals to /4_retinotopy and extracts the retinotopy volumes. A separate file containing those (ret.nii.gz) will be created in the respective folder. Afterwards a unzipped copy of

"ret.nii.gz" will be created.

example: sh do_prepareretinotopy.sh

7.22 do_retinotopy.sh

Wrapper function to call a MATLAB script doing the retinotopic analysis (do_retinotopy.m).

The functions submits a qsub MATLAB session to the cluster and runs the analysis. It can be specified how much memory to use.

example: sh do_retinotopy.sh 32gb

7.23 do_retinotopy.m

Computes retinotopic mapping. It uses /A_helperfiles/RetStim.mat to obtain stimulus information and /4_retinotopy/ret.nii as functional data file.

7.24 meanT.sh (Q)

example: sh meanT.sh my_mask

Computes the averge activation per volume and outputs the respective result as "avgvolumeovertime.txt" in /C_miscResults according to the region specified within the mask. If the function is called without pre-defined mask, the average over the full volume will be computed.

Masks can be csfmask, graymattermask, whitemattermask

7.25 matmul.sh

Matrix multiplication from files.

example: sh matmul.sh file1 file2

7.26 multiply_all_M_in_A_with_B.sh

Wrapper function that uses matmul.sh to multiply all matrices in the folder given in the first argument (path relative to function) and multiplies them with the matrix given in the second argument (path relative to function).

Note: the first argument (e.g. A) is a folder the second a file (e.g. B.txt).

example: sh multiply_all_M_in_A_with_B.sh A B.txt

The result of this transformation is stored in:

/<A>/matmulResults/*

7.27 file2plot.sh

Can be used to plot data from files.

example: sh file2plot.sh /C_miscResults/avgvolumeovertimefullmask.txt

7.28 liveupdateqsub.sh

Submits a qstat -u user request every second.

example: sh liveupdateqsub.sh tomcla

exit command by hitting ctrl+c