**Overall file structure**

**Run\_pipelines\_noSGE (or python wrapper)**

**Pipeline\_PART1:**

\* **Input**: input textfiles (subject inputs, pipeline list, task design)

**\*** performs all preprocessing steps (directly, or by calling other functions)

\* **Outputs**: files in “intermediate\_processed” and “intermediate\_metrics” folders

\* “intermediate\_metrics” contains NPAIRS metrics and SPMs of ***all*** tested pipelines, for all runs

**Pipeline\_PART2:**

\* **Input**: textfiles and “intermediate\_metrics” data from Pipeline\_PART1

\* identifies 3 pipelines: conservative (CONS), optimal fixed (FIX) and individual (IND)

\* re-generates SPMs and preprocessed data for these optimal pipelines, for each subject

\* **Output**: optimal SPMs and processed .nii files in “optimization\_results” folder

**spatial\_normalization:**

\* **Input1**: a “template” brain map, subjects’ 3D T1 images, “intermediate\_processed” data

\* generates affine transformation from functional to common template space

\* **Input2**: SPMs and preprocessed data data in “optimization\_results” folder

\* applies transformation to optimally processed data

\* **Output**: all .nii data in “optimization\_results” are now spatially normalized (\*\_sNorm suffix)

**Output structure:**  final directory structure for optimized pipelines:

<path>/intermediate\_processed

afni\_processed 🡪 files produced by combinations of AFNI steps (**MOTCOR**,**CENSOR**,**TIMECOR**,**RETROICOR**,**SMOOTH**)

diagnostic 🡪 files of diagnostic data for quality control (also used in **CENSOR**)

masks 🡪 subject brain masks (produced by *3dAutomask*)

mpe 🡪 rigid-body motion parameter estimates (produced by **MOTCOR**, used in **MOTREG**)

split\_info 🡪 data from subject input files, reformatted as .mat files

<path>/intermediate\_metrics

regressors 🡪 contains all design matrices used for different processing pipelines (e.g. task+noise covariates)

res0\_params 🡪 contains some parameters of task design

res1\_spms 🡪 contains SPMs produced of ***all*** tested pipelines

res2\_temp 🡪 contains activation timeseries of ***all*** tested pipelines

res3\_stats 🡪 contains NPAIRS metrics (P,R) of ***all*** tested pipelines

<path>/optimization\_results

matfiles 🡪 .mat files containing summary info. from optimized subject pipelines

processed 🡪 optimally preprocessed subject datasets

spms 🡪 analysis SPMs from optimally preprocessed subject data

**Pipeline\_PART1**(InputStruct, input\_pipeset, analysis\_model, modelparam, niiout, contrast\_list\_str, dospnormfirst, DEOBLIQUE, TPATTERN)

%% this script applies all pipeline steps to subject data, generating NPAIRS metrics and SPMs for every tested pipeline.

These results are then fed into Pipeline\_PART2 to perform optimization

%% Pipeline steps (MOTCOR, CENSOR, TIMECOR, RETROICOR, SMOOTH) run by calling **Pipeline\_PART1\_afni\_steps**.m

%% Pipeline steps (DETREND, MOTREG, GSPC1, DETREND, PHYPLUS, LOWPASS) and analysis of processed data is run

within the script, by calling the function **apply\_regression\_step**.

%% main results (in “intermediate\_metrics” folder) are stored in .mat files, organized as:

.res0\_params folder, containing parameters of task design

.res1\_spms folder, containing activation maps produced by all tested pipelines, for each subject

.res2\_temp folder, containing activation timeseries produced by all tested pipelines, for each subject

.res3\_stats folder, containing NPAIRS metrics (P,R) produced by all tested pipelines, for each subject

--> InputStruct = list of subject input/output names (textfile or .mat format)

--> input\_pipeset = list of all tested pipelines (textfile)

--> analysis\_model = string specifying analysis model used to generate SPMs, metrics (e.g. ‘GLM’, ‘GNB’, ‘LDA’)

--> modelparam = string giving special arguments for running analyses, created by running python wrapper

--> niiout = flag to output all pipeline SPMs as .nii files (0=off, 1=on) [**TURN OFF UNLESS YOU ARE ONLY**

**RUNNING A FEW PIPELINES. OTHERWISE IT WILL USE A LOT OF STORAGE SPACE**]

--> contrast\_list\_str = string, specifying which task conditions should be analyzed

--> dospnormfirst = flag if spatial normalization done early in pipeline (not used right now)

--> DEOBLIQUE = flag to correct for oblique scans, convert to cardinal axes via "3dWarp" (0=off, 1=on)

--> TPATTERN = if slice-timing not available, user can supply timing info

(string, fed as argument into "3dTshift")

%% lines 181-196 parse the input files (and check for errors), in preparation for running pipelines

(181) .reformat "InputStruct" textfile into matlab struct file (call: Read\_Input\_File.m)

(184) .take list of pipeline steps in "input\_pipeset", reformat into matlab objects

\*\*pipeset\_half = 5-column design matrix, that determine which AFNI steps off/on (i.e. each row is a different

pipeline) this is the first half of tested pipelines, run by Pipeline\_PART1\_afni\_steps.m

\*\*other steps (detSet, mprSet, tskSet, ...) determine if regression steps are turned off/on

(196) .check if any errors in input files (call: check\_input\_file\_integrity.m)

%% this step runs the first half of all pipeline steps, mainly calls to AFNI functions (creates.nii data in afni\_processed folders)

(200) .run the first half of all pipeline steps, which call AFNI functions (call: **Pipeline\_PART1\_afni\_steps**.m)

(204) .take input arguments, and append a list of specific task contrasts to be analyzed

\*\*appends arguments to InputStruct file

\*\*takes modelparam, analysis\_model (analysis approach), contrast\_list\_str (specific task contrasts to analyze)

(211-217) .reformat, copy task design files into new /intermediate\_processed/split\_info folder (for efficiency purposes)

(221) .for each subject in InputStruct...

%% lines 245-377 do (a) some formatting of motion parameter estimates (mpes) and

%% (b) generate spatial priors for QC later, including motion artifact, vascular noise and white matter

(245-246) .load subject brain mask into matlab

(258). for each run of subject data (can be 1 run, or multi-run)...

%% lines 299-377 (a) generate components of motion parameter estimates for MOTREG

%% and (b) spatial maps for QC, including edge artifact, vascular regions and white matter

(299) .take motion parameter estimates and get principal components (PCs) for MOTREG

(306-309) .load minimally processed \*baseproc.nii file for generating some spatial priors

(315-326) .compute spatial derivatives on mean \*baseproc.nii volume, as spatial prior for

detecting motion artifacts

(331-339) .remove mean+linear trends for all \*baseproc.nii files, before measuring physio. noise priors

(347-364) .estimate vascular and white matter spatial priors (for quality checks, and running PHYPLUS)

(369-370) .define signal/noise regressors

(372-375) .store masks and noise priors in "split\_info" structure

(377) .store all spatial noise maps and signal/noise regressors

%% lines 398-475 run the second half of all pipeline steps, generating pipeline SPMs and NPAIRS metrics

(398) .for each pipeline combo. produced by "**Pipeline\_PART1\_afni\_steps**.m", apply pipeline steps...

(411-437) .for each run of subject data (can be 1 run, or multi-run), load+store .nii files in cell array

(443-463) .test all combinations of steps turned on/off (DETREND/MOTREG/TASK/GSPC1/LOWPASS)...

(453-457) .apply noise regressions and run analysis for the chosen combination of pipeline steps

\*\* this calls functions: apply\_regression\_step (for 1 run/subject) or

\*\* apply\_regression\_step\_group (multiple runs/subject)

\*\* outputs include: SPMs in IMAGE\_set, pipeline metrics in METRIC\_set, and BOLD

timeseries in TEMP\_set. (\*\_set\_0= without PHYPLUS step, \*\_set\_y=with PHYPLUS)

%% all subsequent lines store results of pipeline analyses

(476) .if an analysis model was chosen (i.e. producing metrics and SPMs)...

(504) .for every pipeline step...

(509-513) .measure SPM correlation with spatial motion artifact (test for bad pipelines)

(517-523) .measure amount of white matter signal in SPMs

(525) .test for presence of global signal effect in SPMs (fraction of positive voxels)

(531-539) .store all pipeline SPMs (/res1\_spms/spms\*.mat), associated BOLD timeseries

(/res2\_temp/tmp\*.mat), pipeline metrics (/res3\_stats/stats\*.mat)

(544-589) .if niiout>0, convert all pipeline SPMs into .nii files -- **not usually recommended unless only a**

**few pipelines are tested**

**Pipeline\_PART1\_afni\_steps**( InputStruct, input\_pipeset\_half, dospnormfirst, DEOBLIQUE, TPATTERN )

%% This script is called by Pipeline\_PART1, and runs a subset of early pipeline steps, mainly calls to AFNI functions (MOTCOR = 3dvolreg, CENSOR = interpolate\_fmri, RETROICOR = 3dretroicor, TIMECOR = 3dTshift, SMOOTH = 3dmerge)

%% main results (in “intermediate\_processed” folder):

.afni\_processed folder, contains intermediate processed files e.g. \*\_m1c0p1t0s6.nii

.masks folder, containing \*\_mask.nii files

.mpe folder, containing \*\_mpe and \*\_maxdisp files (used in MOTREG step)

.diagnostic folder, containing \* \_QC\_output.mat files (for visual assessment of data quality, also used in CENSOR step)

--> InputStruct = list of subject input/output names (textfile or .mat format)

--> input\_pipeset\_half = 5-column design matrix, indicating which AFNI steps turned off/on

(i.e. each row is a different pipeline)

--> dospnormfirst = flag if spatial normalization done early in pipeline (not used right now)

--> DEOBLIQUE = flag to correct for oblique scans, convert to cardinal axes via "3dWarp" (0=off, 1=on)

--> TPATTERN = if slice-timing not available, user can supply timing info

(string, fed as argument into "3dTshift")

(66) for each subject in InputStruct...

(68) for each of subject's runs...

( 83-105) .preparation steps (create output folders, etc)

(107-113) .fix scan coordinate axes (call: 3dWarp) if DEOBLIQUE turned on [OUT: \*\_deob.nii file]

.drop scans from start/end of run, based on "DROP" argument in InpuStruct [OUT: \*\_drop.nii file]

%% lines 115-133 create (a) outputs for data with/without motion correction, for diagnostic purposes

%% diagnostic plots also include (b) list of outlier volumes for CENSOR step

%% also (c) motion parameter values for MOTREG and corrected data (MOTCOR) for upcoming pipeline steps

(115-119) .fixed spatial smoothing of unprocessed data 6mm FWHM (call: 3dmerge) [OUT: \*\_tempsmo.nii]

.coarse mask of \*\_tempsmo.nii data (call: 3dAutomask) [OUT: \*\_mask\_nomc.nii]

.identify scan volume of minimum displacement "bricknum", as future reference for MOTCOR

using unprocessed \*\_tempsmo.nii and \*\_mask\_nomc.nii (call: min\_displace\_brick.m)

(121-123) .motion correction of raw data, relative to volume "bricknum" (call: 3dvolreg)

[OUT: /mpe/\*\_mpe (file of rigid-body motion parameters), for MOTREG pipeline step later]

[OUT: /mpe/\*\_maxdisp (file of maximum displacement per volume, for diagnostic purposes only)]

[OUT: \*\_drop+mc.nii]

.spatial smoothing 6mm FWHM (call: 3dmerge) [OUT: \*\_tempsmo\_mc.nii]

.masking of data (call: 3dAutomask) [OUT: /masks/\*\_mask.nii]

(128) .diagnostic testing on data \*\_smo.nii without motion correction (call: diagnostic\_fmri\_pca.m)

[OUT: /diagnostic/\*\_smo\_QC\_output.mat]

(a .mat file listing outlier volumes for CENSOR step, plus some other data)

(133) .diagnostic testing on data \*\_tempsmo\_mc.nii WITH motion correction (call: diagnostic\_fmri\_pca.m)

[OUT: /diagnostic/\*\_mc+smo\_QC\_output.mat]

(a .mat file listing outlier volumes for CENSOR step, plus some other data)

(134) .copy \*\_tempsmo\_mc.nii into \*\_baseproc.nii file, for later use in spatial normalization

%% all steps from (137-341) are now applying pipeline steps in order. At each step, we take .nii data from last

%% step, and generate new .nii files with the step turned ON/OFF

%% data naming indicates which steps were turned off/on, e.g. \*\_m1c0p1t0s6.nii indicates a pipeline with %% (MOTCOR=ON, CENSOR=OFF, RETROICOR=ON, TIMECOR=OFF, SMOOTH=6)

(137-213) .if pipeline with **MOTCOR** OFF is being tested...

(138-143) .if pipeline with **CENSOR** OFF is tested...

(140). rename unprocessed file \*\_drop.nii [OUT: \*\_m0c0.nii]

(144-150) .if pipeline with **CENSOR**=1 is tested

(147). identify and discard outlier timepoints in unprocessed data \*\_drop.nii

(call: interpolate\_fmri.m) [OUT: \*\_m0c1.nii]

(151-213) .if pipeline with **CENSOR**=2 or is tested

\*\*\* removes outliers from \*\_drop.nii and performs extra PCA/ICA cleanup -- not validated

(214-290) .if pipeline with **MOTCOR** ON is being tested...

(215-220) .if pipeline with **CENSOR** OFF is tested...

(217). rename unprocessed file \*\_drop+mc.nii [OUT: \*\_m1c0.nii]

(221-227) .if pipeline with **CENSOR**=1 is tested

(224). identify and discard outlier timepoints in unprocessed data \*\_drop+mc.nii

(call: interpolate\_fmri.m) [OUT: \*\_m1c1.nii]

(228-290) .if pipeline with **CENSOR**=2 or is tested

\*\*\* removes outliers and performs extra PCA/ICA cleanup -- not fully validated yet

(214-290) .if pipeline with **RETROICOR** OFF is being tested...

(295) for all previously created pipeline files, append "p0" [OUT: \*\_m0c0p0.nii, \*\_m0c1p0.nii, etc.]

(300-307) .if pipeline with **RETROICOR** ON is being tested...

(303) .for all previously generated pipeline files, perform physiological correction (call: 3dretroicor)

[OUT: \*\_m0c0p1.nii, \*\_m0c1p1.nii, etc.]

(309-316) .if pipeline with **TIMECOR** OFF is being tested...

(312) for all previously created pipeline files, append "t0" [OUT: \*\_m0c0p0t0.nii, \*\_m0c0p1t0.nii …]

(317-328) .if pipeline with **TIMECOR** ON is being tested...

(295) .for all previously generated pipeline files, perform slice-timing correction (call: 3dTshift)

[OUT: \*\_m0c0p0t1.nii, \*\_m0c0p1t1.nii, etc.]

(330-341) .for all chosen smoothing kernels (**SMOOTH**)...

(334) .if smoothing kernel =0, just append "t0" [OUT: \*\_m0c0p0t0s0.nii, etc.]

(336) .for all previously generated pipeline files, do spatial smoothing step SMOOTH (call: 3dmerge)

e.g. for a FWHM=6mm kernel, [OUT: \*\_m0c0p0t0s6.nii, \*\_m0c0p0t1s6.nii, etc.]

(343-350) .now delete all intermediate processed .nii files, that did not have smoothing applied

(e.g. \*\_m0.nii, \*\_m0c0.nii, \*\_m0c0p0.nii, \*\_m0c0p0t0.nii, etc.)

(353) .also delete coarse mask /masks/\*\_mask\_nomc.nii

(358-384). some extra steps if dospnormfirst turned on

\*\* (382) also delete intermediate \*\_tempsmo.nii files

(387-402). more extra steps if dospnormfirst turned on + multiple runs per subject

(404-449). more extra steps if multiple runs per subject (and dospnormfirst=off)

**Pipeline\_PART2**( InputStruct, optimize\_metric, mot\_gs\_control, process\_out, keepmean)

%% This script takes the results of pipeline analyses in **Pipeline\_PART1**, and identifies the set of optimal pipelines

%% results are included for “conservative” pipeline (CON), optimal “fixed” pipeline (FIX) and individual pipelines (IND)

%% main results (in “optimization\_results” folder):

.spms folder, contains optimal z-scored SPMs per subject, denoted rSPM\_\*\_CON\_FIX\_IND.nii

.processed folder, contains optimally processed 4D fMRI data, denoted Proc\_\*.nii

.matfiles folder, contains some summary results of optimized subjects, in .mat files

--> InputStruct = list of subject input/output names (textfile or .mat format)

--> optimize\_metric = string, specifying NPAIRS metric used for optimization (conventionally ‘dPR’)

--> mot\_gs\_control = 2D vector, specifying if we exclude pipelines with motion artifact and/or global signal

artifact in their SPMs, when doing optimization

--> process\_out = flag, determining if optimally processed 4D .nii data are output (otherwise just SPMs)

--> keepmean = flag, re-adds voxel means to optimally processed data before outputting (for compatibility with

SPM and PLS software packages)

(191). if more than one pipeline is being run, do full optimization...

(198). for each subject ...

(201) .load subject's pipeline metrics

(203-213) .reformat metrics -- if more than one task contrast, average pipeline metrics over contrasts

(240-250) .store all pipeline and artifact metrics in matrices

(267-282) .find the "most conservative" pipeline, which best matches:

{MOTCOR=1,CENSOR=1,TIMECOR=1,RETROICOR=1,SMOOTH=6,DET=<recommended by AFNI heuristic>,MOTREG=1}

(287-355) .find the optimal fixed pipeline across all subjects...

\*\* first rank pipelines for each subject, and find pipeline with highest median rank

\*\* find pipelines not significantly different from "highest ranked" (95% CI based on friedman test)

\*\* look for highest-ranked pipeline in this set, with MOTCOR and PHYPLUS

(forcing physio.+motion correction avoids choosing "bad" pipelines)

\*\* if no pipeline in the set has MOTCOR/PHYPLUS and these steps are available, choose highest-

ranked pipeline with them turned on

(360-520) .find optimal individual pipelines

\*\* if mot\_gs\_control(1)>0, we reject IND pipelines if SPMs have motion artifact

\*\* reject if pipeline without MOTCOR have significantly more artifact than pipelines with MOTCOR

\*\* if mot\_gs\_control(1)>0, we reject IND pipelines if SPMs have global signal artifact

\*\* reject if pipeline without GSPC1 have significantly more artifact than pipelines with GSPC1

(530-588) .store optimal pipeline sets

(592) .if process\_out>0, optimization results output as .nii files...

(596) .for each subject...

(599-625) .load intermediate\_metrics data (Pipeline\_PART1), and save the optimal rSPMs

(653-713) .regenerate optimally processed datasets, save as .nii files (for CON, FIX, IND)

(718) .otherwise, if only 1 pipeline was run, just generate SPM and processed data for single pipeline