

afni\_proc.py  
is your *friend*  
*or it will be soon*

Example scripts from

**AFNI\_data6/FT\_analysis**

Also see <https://arxiv.org/abs/1709.07471>

Appendix has processing scripts

# What the hell is afni\_proc.py?

- It is a Python program that
  - Takes as input a series of “options” describing processing steps to use to analyze datasets *from one subject*
  - Produces as output a Unix tcsh script file that runs all the **AFNI** programs to do the processing
- Reasons to use afni\_proc.py
  - It is flexible and compact, to produce a long script
  - The output script not only does the data analysis, but also saves various diagnostic tools and files
  - All intermediate output datasets are saved to help diagnose things when results are confusing or just plain wrong
  - You can get help from us on the **AFNI** message board

# Where do afni\_proc.py command lines/scripts come from?

- **Method #1:**
  - take an existing script (from yourself or a friend), and modify it to meet your needs
- **Method #2:**
  - find an approximate fit to what you want in examples from afni\_proc.py's help, and modify to meet your needs
- **Method # $\sqrt{-1}$ :**
  - use GUI uber\_subject.py
- **Method #666:**
  - beg for help on the **AFNI** message board

# Starting Simple - 1

```
afni_proc.py -subj_id FT \
  -dsets FT/FT_epi_r?+orig.HEAD \
  -copy_anat FT/FT_anat+orig \
  -tcats_remove_first_trs 2 \
  -regress_stim_times FT/AV*.txt \
  -regress_stim_labels Vrel Arel \
  -regress_basis 'BLOCK(20,1)' \
  -regress_est_blur_errts \
  -regress_opts_3dD \
  -gltsym 'SYM: Vrel -Arel' \
  -glt_label 1 V-A
```

Script file  
s01.ap.simple

# Starting Simple - 2

`afni_proc.py` `-subj_id FT`

ID will label  
output files

```
-dsets FT/FT_epi_r?+orig.HEAD \
-copy_anat FT/FT_anat+orig \
-tcat_remove_first_trs 2 \
-regress_stim_times FT/AV*.txt \
-regress_stim_labels Vrel Arel \
-regress_basis 'BLOCK(20,1)' \
-regress_est_blur_errts \
-regress_opts_3dD \
      -gltsym 'SYM: Vrel -Arel' \
      -glt_label 1 V-A
```

Script file  
s01.ap.simple

# Starting Simple - 3

```
afni_proc.py -subj_id FT
```

EPI time  
series  
datasets  
to analyze

```
-dsets FT/FT_epi_r?+orig.HEAD
```

```
-copy_anat FT/FT_anat+orig
```

```
-tcat_remove_first_trs 2
```

```
-regress_stim_times FT/AV*.txt
```

```
-regress_stim_labels Vrel Arel
```

```
-regress_basis 'BLOCK(20,1)'
```

```
-regress_est_blur_errts
```

```
-regress_opts_3dD
```

```
    -gltsym 'SYM: Vrel -Arel'
```

```
    -glt_label 1 V-A
```

Script file  
s01.ap.simple

# Starting Simple - 4

```
afni_proc.py -subj_id FT \
  -dsets FT/FT_epi_r?+orig.HEAD \
  -copy_anat FT/FT_anat+orig \
  -tcat_remove_first_trs 2 \
  -regress_stim_times FT/AV*.txt \
  -regress_stim_labels Vrel Arel \
  -regress_basis 'BLOCK(20,1)' \
  -regress_est_blur_errts \
  -regress_opts_3dD \
  -gltsym 'SYM: Vrel -Arel' \
  -glt_label 1 V-A
```

T1-weighted  
anatomical  
dataset for  
alignment to  
EPI datasets

Script file  
s01.ap.simple

# Starting Simple - 5

```
afni_proc.py -subj_id FT \
  -dsets FT/FT_epi_r?+orig.HEAD \
  -copy_anat FT/FT_anat+orig \
```

```
-tcat_remove_first_trs 2 \
  -regress_stim_times FT/AV*.txt \
  -regress_stim_labels Vrel Arel \
  -regress_basis 'BLOCK(20,1)' \
  -regress_est_blur_errts \
  -regress_opts_3dD \
  -gltsym 'SYM: Vrel -Arel' \
  -glt_label 1 V-A
```

Stimulus timing files, labels, and HRF model;  
**Note:** timing files have *start* times for each task iteration

Script file  
s01.ap.simple



# Starting Simple - 6

```
afni_proc.py -subj_id FT \
  -dsets FT/FT_epi_r?+orig.HEAD \
  -copy_anat FT/FT_anat+orig \
  -tcats_remove_first_trs 2 \
  -regress_stim_times FT/AV*.txt \
  -regress_stim_labels Vrel Arel \
  -regress_basis 'BLOCK(20,1)' \
  -regress_est_blur_errts \
  -regress_opts_3dD \
  -gltsym 'SYM: Vrel -Arel' \
  -glt_label 1 V-A
```

Estimate  
smoothness  
of EPI noise  
for group  
analysis

Script file  
s01.ap.simple

# Starting Simple - 7

```
afni_proc.py -subj_id FT \
  -dsets FT/FT_epi_r?+orig.HEAD \
  -copy_anat FT/FT_anat+orig \
  -tcat_remove_first_trs 2 \
  -regress_stim_times FT/AV*.txt \
  -regress_stim_labels Vrel Arel \
  -regress_basis 'BLOCK(20,1)' \
  -regress_est_blur_errts \
  -regress_opts_3dD \
  -gltsym 'SYM: Vrel -Arel' \
  -glt_label 1 V-A \
```

Set up  
General  
Linear  
Test  
between 2  
conditions

Script file  
s01.ap.simple

# A Real Case - 1

```
#!/usr/bin/env tcsh
```

```
# creation date: Thu Sep 10 14:27:59 2015
```

```
# set data directories
```

```
set top_dir      = FT
```

```
# set subject and group identifiers
```

```
set subj         = FT
```

```
set group_id     = horses
```

Not actually  
used here

Code subject level information  
into shell variables:

Makes it easier to re-use this afni\_proc.py command

# A Real Case - 2

Script file  
s05.ap.uber

```
afni_proc.py -subj_id $subj \
  -script proc.$subj -scr_overwrite \
  -blocks tshift align tlrc volreg blur mask scale regress \
  -copy_anat $top_dir/FT_anat+orig \
  -dsets \
    $top_dir/FT_epi_r1+orig.HEAD \
    $top_dir/FT_epi_r2+orig.HEAD \
    $top_dir/FT_epi_r3+orig.HEAD \
  -volreg_align_to MIN_OUTLIER \
  -volreg_align_e2a \
  -volreg_tlrc_warp \
  -blur_size 4.0 \
  -tcat_remove_first_trs 2 \
  -regress_stim_times \
    $top_dir/AV1_vis.txt \
    $top_dir/AV2_aud.txt \
  -regress_stim_labels \
    vis aud \
  -regress_basis 'BLOCK(20,1)' \
  -regress_censor_motion 0.3 \
  -regress_opts_3dD \
    -jobs 2 \
    -gltsym 'SYM: vis -aud' -glt_label 1 V-A \
    -gltsym 'SYM: 0.5*vis +0.5*aud' -glt_label 2 mean.VA \
  -regress_compute_fitts \
  -regress_make_ideal_sum sum_ideal.1D \
  -regress_est_blur_epits \
  -regress_est_blur_errts \
  -regress_run_clustsim yes \
  -execute
```

The entire afni\_proc.py  
command:  
Font size will be bigger  
on following slides!

# A Real Case – 3a

Script file  
s05.ap.uber

```
afni_proc.py -subj_id $subj  
-script proc.$subj -scr_overwrite  
-blocks tshift align tlrc volreg blur mask scale regress  
-copy_anat $top_dir/FT_anat+orig  
-dsets  
    $top_dir/FT_epi_r1+orig.HEAD  
    $top_dir/FT_epi_r2+orig.HEAD  
    $top_dir/FT_epi_r3+orig.HEAD  
-volreg_align_to MIN_OUTLIER  
-volreg_align_e2a  
-volreg_tlrc_warp  
-blur_size 4.0
```

Set up which  
processing  
"blocks"  
will be run

\  
\  
\  
\  
\  
\  
\  
\  
\  
\  
\  
\

# A Real Case – 3b

Script file  
s05.ap.uber

```
afni_proc.py -subj_id $subj \
  -script proc.$subj -scr_overwrite \
  -blocks tshift align tlrc volreg blur mask scale regress \
  -copy_anat $top_dir/FT_anat+orig \
  -dsets \
    $top_dir/FT_epi_r1+orig.HEAD \
    $top_dir/FT_epi_r2+orig.HEAD \
    $top_dir/FT_epi_r3+orig.HEAD \
  -volreg_align_to MIN_OUTLIER \
  -volreg_align_e2a \
  -volreg_tlrc_warp \
  -blur_size 4.0 \
```

Select input  
datasets  
(anat and EPI)

# A Real Case – 3c

Script file  
s05.ap.uber

```
afni_proc.py -subj_id $subj \
  -script proc.$subj -scr_overwrite \
  -blocks tshift align tlrc volreg blur mask scale regress \
  -copy_anat $top_dir/FT_anat+orig \
  -dsets \
    $top_dir/FT_epi_r1+orig.HEAD \
    $top_dir/FT_epi_r2+orig.HEAD \
    $top_dir/FT_epi_r3+orig.HEAD \
  -volreg_align_to MIN_OUTLIER \
  -volreg_align_e2a \
  -volreg_tlrc_warp \
  -blur_size 4.0
```

Specify how  
“volreg” step  
will operate

# A Real Case – 3d

Script file  
s05.ap.uber

```
afni_proc.py -subj_id $subj \
  -script proc.$subj -scr_overwrite \
  -blocks tshift align tlrc volreg blur mask scale regress \
  -copy_anat $top_dir/FT_anat+orig \
  -dsets \
    $top_dir/FT_epi_r1+orig.HEAD \
    $top_dir/FT_epi_r2+orig.HEAD \
    $top_dir/FT_epi_r3+orig.HEAD \
  -volreg_align_to MIN_OUTLIER \
  -volreg_align_e2a \
  -volreg_tlrc_warp \
  -blur_size 4.0 \
```

Specify how  
much spatial  
blurring will  
be used  
(FWHM mm)









# A Real Case – 4d

Script file  
s05.ap.uber

```
-tcat_remove_first_trs 2
-regress_stim_times
    $stop_dir/AV1_vis.txt
    $stop_dir/AV2_aud.txt
-regress_stim_labels
    vis aud
-regress_basis 'BLOCK(20,1)'
-regress_censor_motion 0.3
-regress_opts_3dD
    -jobs 2
    -gltsym 'SYM: vis -aud' -glt_label 1 V-A
    -gltsym 'SYM: 0.5*vis +0.5*aud' -glt_label 2 mean.VA
-regress_compute_fitts
-regress_make_ideal_sum sum_ideal.1D
-regress_est_blur_epits
-regress_est_blur_errts
-regress_run_clustsim yes
-execute
```

Other regression options:  
Compute fitted model  
(best fit to data);  
Create sum of task ideal  
response time series  
(for display purposes)

# A Real Case – 4e

Script file  
s05.ap.uber

```
-tcat_remove_first_trs 2
-regress_stim_times
    $stop_dir/AV1_vis.txt
    $stop_dir/AV2_aud.txt
-regress_stim_labels
    vis aud
-regress_basis 'BLOCK(20,1)'
-regress_censor_motion 0.3
-regress_opts_3dD
    -jobs 2
    -gltsym 'SYM: vis -aud' -glt_label 1 V-A
    -gltsym 'SYM: 0.5*vis +0.5*aud' -glt_label 2 mean.VA
-regress_compute_fitts
-regress_make_ideal_sum sum_ideal.1D
-regress_est_blur_epits
-regress_est_blur_errts
-regress_run_clustsim yes
-execute
```

Estimate smoothness  
of *noise* in the data:  
From the dataset itself,  
From the *residuals*  
(=data-model fit).

And estimate cluster-  
size thresholds from  
smoothness estimates

# A Real Case – 4f

Script file  
s05.ap.uber

```
-tcat_remove_first_trs 2
-regress_stim_times
    $stop_dir/AV1_vis.txt
    $stop_dir/AV2_aud.txt
-regress_stim_labels
    vis aud
-regress_basis 'BLOCK(20,1)'
-regress_censor_motion 0.3
-regress_opts_3dD
    -jobs 2
    -gltsym 'SYM: vis -aud' -glt_label 1 V-A
    -gltsym 'SYM: 0.5*vis +0.5*aud' -glt_label 2 mean.VA
-regress_compute_fitts
-regress_make_ideal_sum sum_ideal.1D
-regress_est_blur_epits
-regress_est_blur_errts
-regress_run_clustsim yes
-execute
```

Run script after  
it is created

# Summarizing Results

- Each afni\_proc.py **results** directory has a file with a name like **out.ss\_review.SUBJECT-ID.txt**
  - Each line give some information about the data and the processing results, such as number of time points censored
- A command like this will generate a table with all these summary results from all subjects:

```
gen_ss_review_table.py  
-infile data_orig/sub*/*.results/out.ss_review.*.txt  
-tablefile UCLA.xls
```

- Next slide: what part of this output looks like in Excel
  - On Linux, you can use LibreOffice <http://www.libreoffice.org/>

# Excel Snapshot Image of UCLA.xls

num TRs per run (censored)

O	P	Q	R	S	T	U	V
max censored	outlier limit	average outlier	num TRs above	num runs found	num TRs per	num TRs per	num TRs per
value_1	value_1	value_1	value_1	value_1	value_1	value_1	value_1
1.86277	0.02	0.0015669	3	1	242	222	20
0.542716	0.02	0.00133806	5	1	242	236	6
1.01995	0.02	0.00101045	2	1	242	231	11
0.970727	0.02	0.00113016	2	1	242	239	3
1.2554	0.02	0.00493115	17	1	242	220	22

Number of TRs with too many outlier values for each subject

Number of TRs found for each subject

Number of TRs censored for each subject



# Another Valuable Summary: TSNR

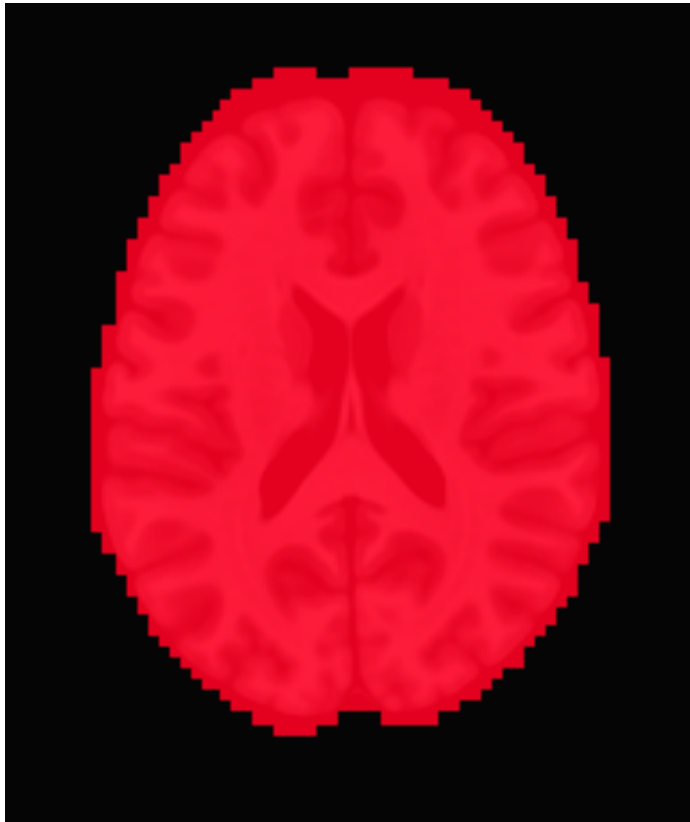
TSNR average
value_1
183.74
218.72
190.983
223.144
194.404

- Measures magnitude of EPI signal strength divided by standard deviation of noise
- For 3 Tesla data, values around 200 are usual – with “standard” scanning parameters (TR about 2-3s, voxel size about 2-3mm)
- If some subject’s TSNR is much lower than others, examine the data to find the problem!

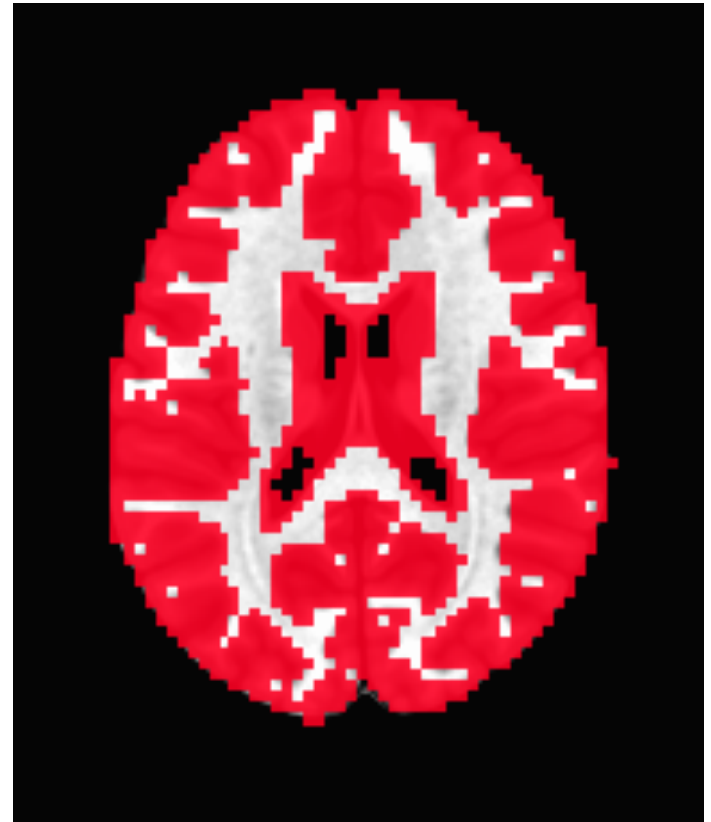
# Masking for Group Analysis

- In each **results** directory, the output dataset **full\_mask.SUBJECT-ID+tlrc.HEAD** is the 0-1 brain mask of the EPI dataset in the template space
- Combine all these masks into one mask dataset:  
**3dmask\_tool**  
-input data\_orig/sub-\*/\*.results/full\_mask.\*.HEAD  
-prefix mask\_all.nii -frac 0.8
- Another way: use a gray matter plus CSF mask from MNI template (*if* you have used nonlinear alignment to that template):  
**3dresample -master mask\_all.nii -prefix mask\_GC.nii**  
-rmode NN  
-input ~/abin/MNI152\_2009\_template.nii.gz '[4]'

# Whole Brain and GM+CSF Masks



73517 voxels



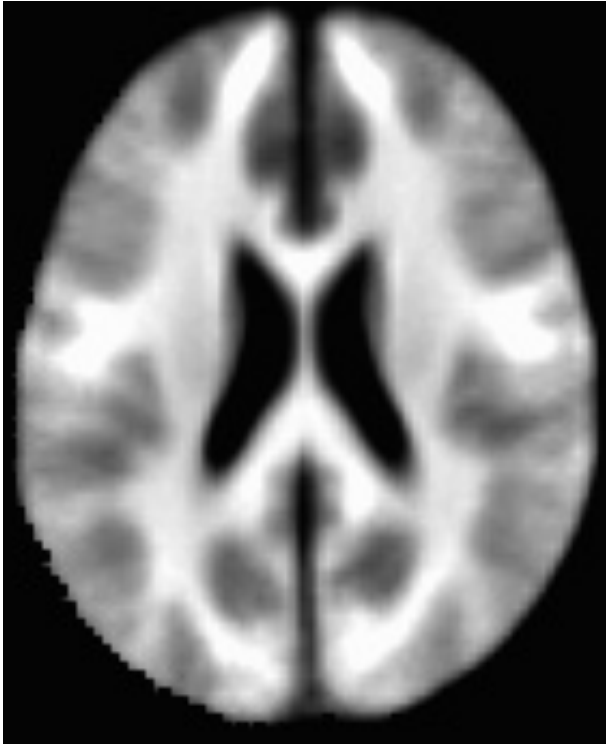
3 mm<sup>3</sup> voxels

53104 voxels

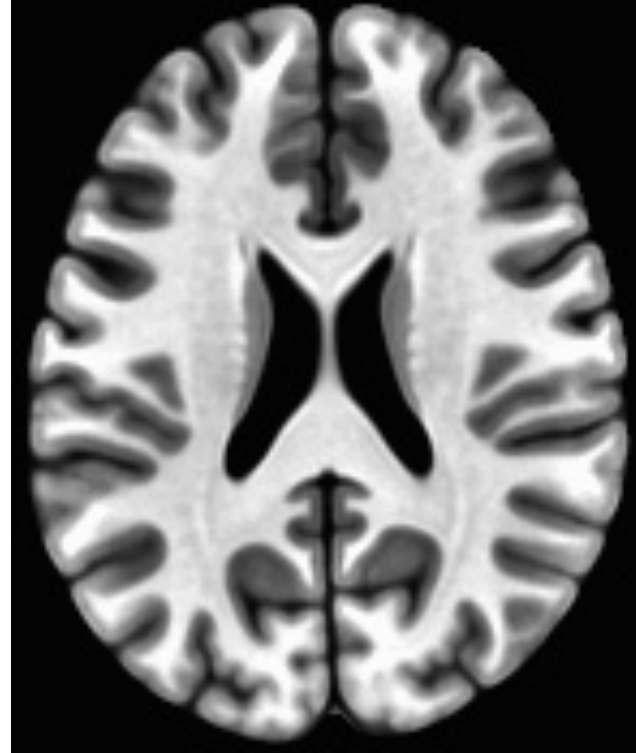
# Nonlinear Warping to MNI Template

- afni\_proc.py *can* do the nonlinear warping for you
  - But, nonlinear warping is slow (in fact, slowly slow)
  - If you need to re-rerun subject analysis, nonlinear warping will slow the re-run script down *a lot*
- Solution: do the nonlinear warping *before* using afni\_proc.py, then supply the warping results so that afni\_proc.py will skip doing the warping itself
- Mechanism: the **@SSwarper** script (tcsh)
  - Does Skull Stripping ("SS") and nonlinear warping
  - Base dataset is **MNI152\_2009\_template.nii.gz**
    - Nonlinearly warped, not too blurry

## Two MNI Templates



**MNI152\_1mm\_uni+tlrc**  
Affine alignments

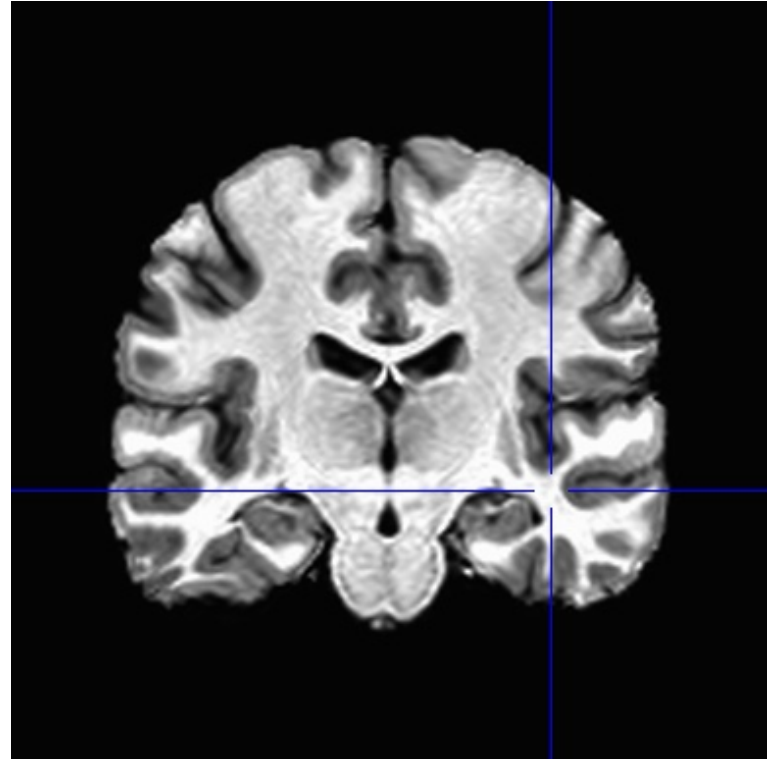
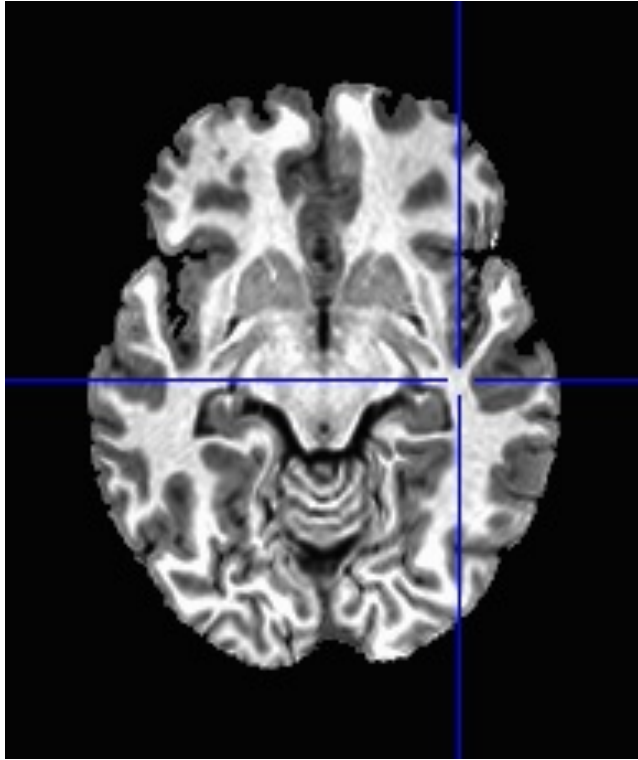


**MNI152\_2009\_template.nii.gz**  
Nonlinear alignments

# What @SSwarper Reads and Writes

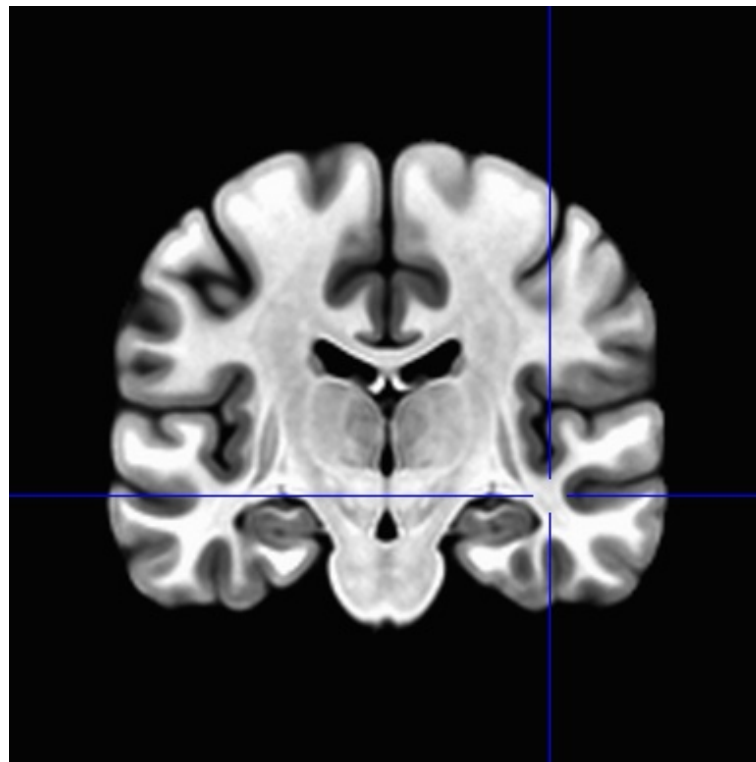
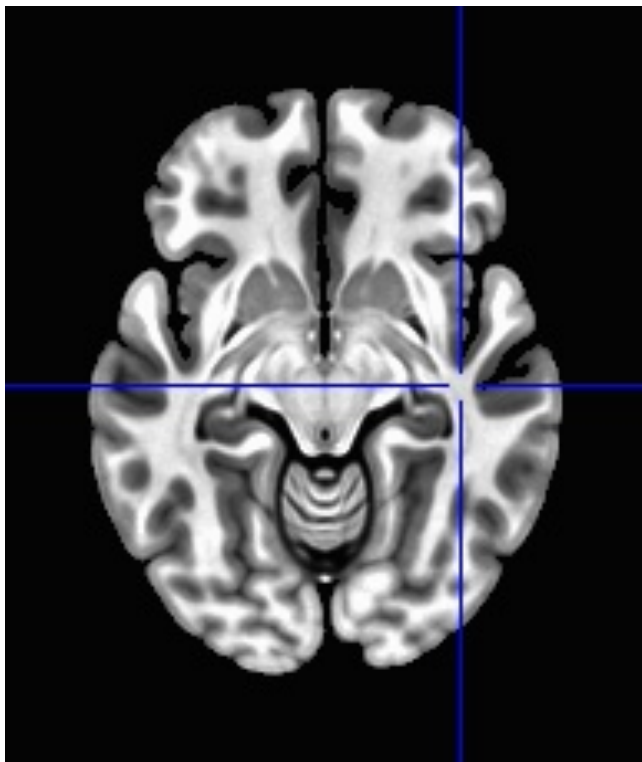
- Inputs:
  - T1-weighted anatomical image of subject (skull-on)
  - Subject ID code, for names of output files
- Outputs (subject ID = **sub007**):
  - **anatSS.sub007.nii**
    - skull-stripped dataset in original coordinates
  - **anatQQ.sub007.nii**
    - skull-stripped dataset, nonlinearly warped to MNI template
  - **anatQQ.sub007.aff12.1D**
    - affine matrix to transform original dataset to MNI template
  - **anatQQ.sub007\_WARP.nii**
    - incremental warp from affine transformation to nonlinearly aligned dataset
- These files are needed for later use in `afni_proc.py`

# @SSwarper Results



sub00440 from Beijing-Zang  
in the FCON-1000 collection

# MNI Template Slices



For comparison



# Nonlinear Registration Script

- What follows is a script for doing nonlinear warping (registration) of *one* anatomical dataset to an MNI template
- In a real study, this script is run once for each subject
- Takes a long time, so the script should be submitted to a multi-node cluster

# Nonlinear Registration - 1

```
#!/bin/tcsh
### This script nonlinear warps one anatomical dataset,
### taken from the anat_orig directory, to the MNI 2009
### nonlinear template (supplied with AFNI binaries), and
### puts the resulting files into anat_warped directory.
### The only command line argument is the subject ID
```

```
set subj = $argv[1]
```

```
set tempdir = .
```

```
# don't log AFNI programs in ~/.afni.log
```

```
# don't try any version checks
```

```
# don't auto-compress output files
```

```
setenv AFNI_DONT_LOGFILE YES
```

```
setenv AFNI_VERSION_CHECK NO
```

```
setenv AFNI_COMPRESSOR NONE
```

# Nonlinear Registration - 2

```
### go to data directory
```

```
# topdir = directory above this Scripts directory
```

```
set topdir = `dirname $cwd`
```

```
cd $topdir/anat_orig
```

```
### create final output directories
```

```
mkdir -p $topdir/anat_warped
```

```
mkdir -p $topdir/anat_warped/snapshots
```

```
### create temp directory to hold work, and copy anat there
```

```
mkdir -p temp_$subj
```

```
cp anat_$subj.nii.gz temp_$subj
```

```
cd temp_$subj
```

# Nonlinear Registration - 3

```
### process the anat dataset, using the AFNI script
### that does the warping and skull-stripping
@SSwarper anat_${subj}.nii.gz $subj
# compress the output datasets
gzip -lv *.nii
### move the results to where they belong
# skull-stripped original, Q-warped dataset, and the warps
\mv -f anatSS.${subj}.nii.gz      anatQQ.${subj}.nii.gz      \
    anatQQ.${subj}.aff12.1D anatQQ.${subj}_WARP.nii.gz \
    $topdir/anat_warped
# snapshots for visual inspection
\mv -f *.jpg $topdir/anat_warped/snapshots
# delete the temporary directory
cd ..
\rm -rf temp_${subj}
exit 0
```

# Nonlinear Registration - 4

*Add these lines above `afni_proc.py` command:*

```
set basedset = MNI152_2009_template.nii.gz
set tpath = `@FindAfnIDsetPath $basedset`
if( "$tpath" == '' ) then
    echo "***** @SSwarper -- Failed to find $basedset :("
    exit 1
endif
set basedset = $tpath/$basedset
```

*Add these options to `afni_proc.py` command:*

```
-copy_anat anat_warped/anatSS.${subj}.nii          \  
-tlrc_base $basedset                               \  
-tlrc_NL_warp                                       \  
-tlrc_NL_warped_dsets                               \  
$warpdir/anatQQ.${subj}.nii.gz                     \  
$warpdir/anatQQ.${subj}.aff12.1D                   \  
$warpdir/anatQQ.${subj}_WARP.nii.gz                \  

```