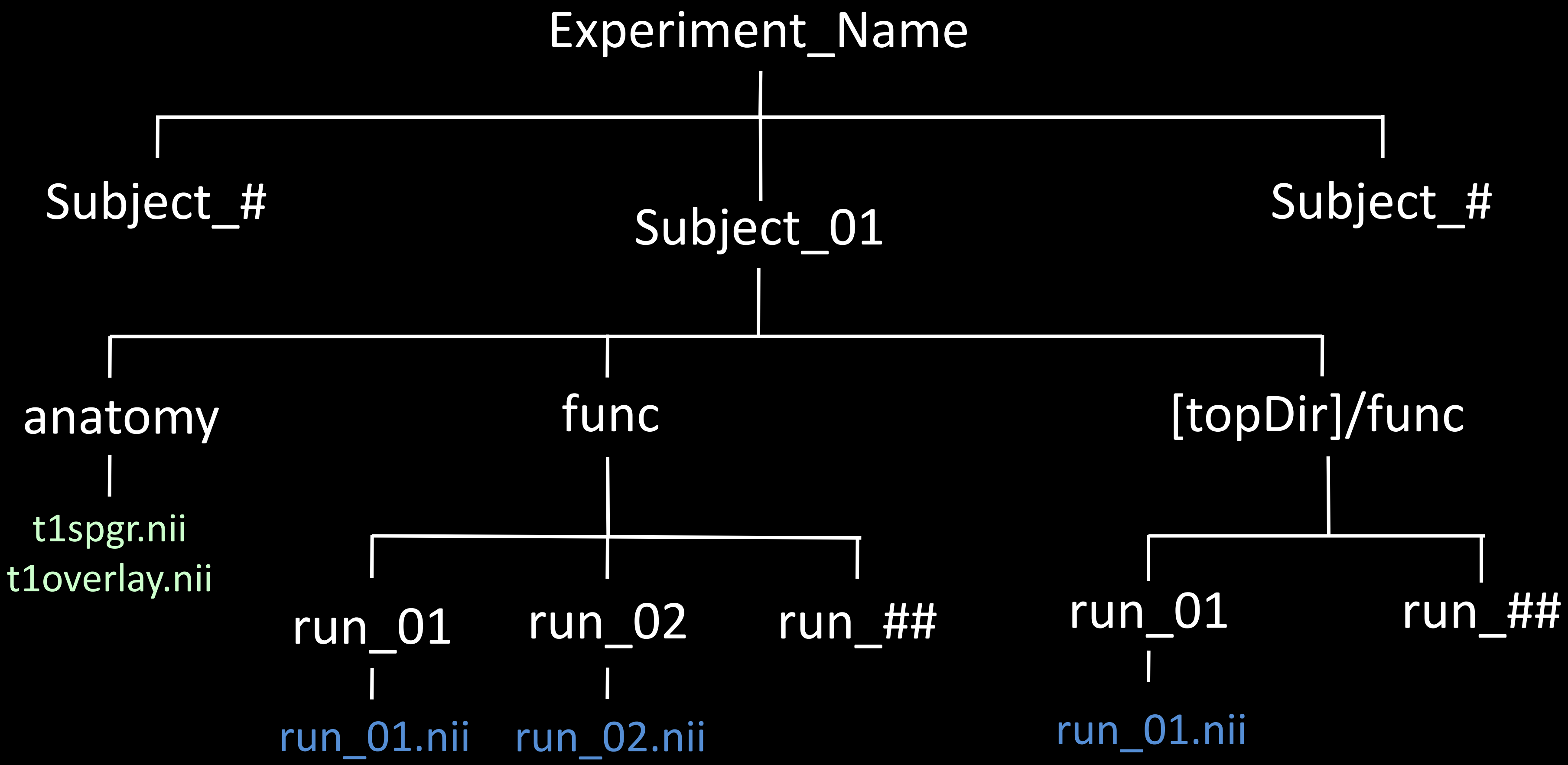
Methods Core

Preprocessing Documentation

Requirements

- SPM8 with VBM8 toolbox
- FSL 4.1.7 or higher
- Bash
- 4D NIFTI images (.nii)
- Standard directory structure

Expected Directory Structure

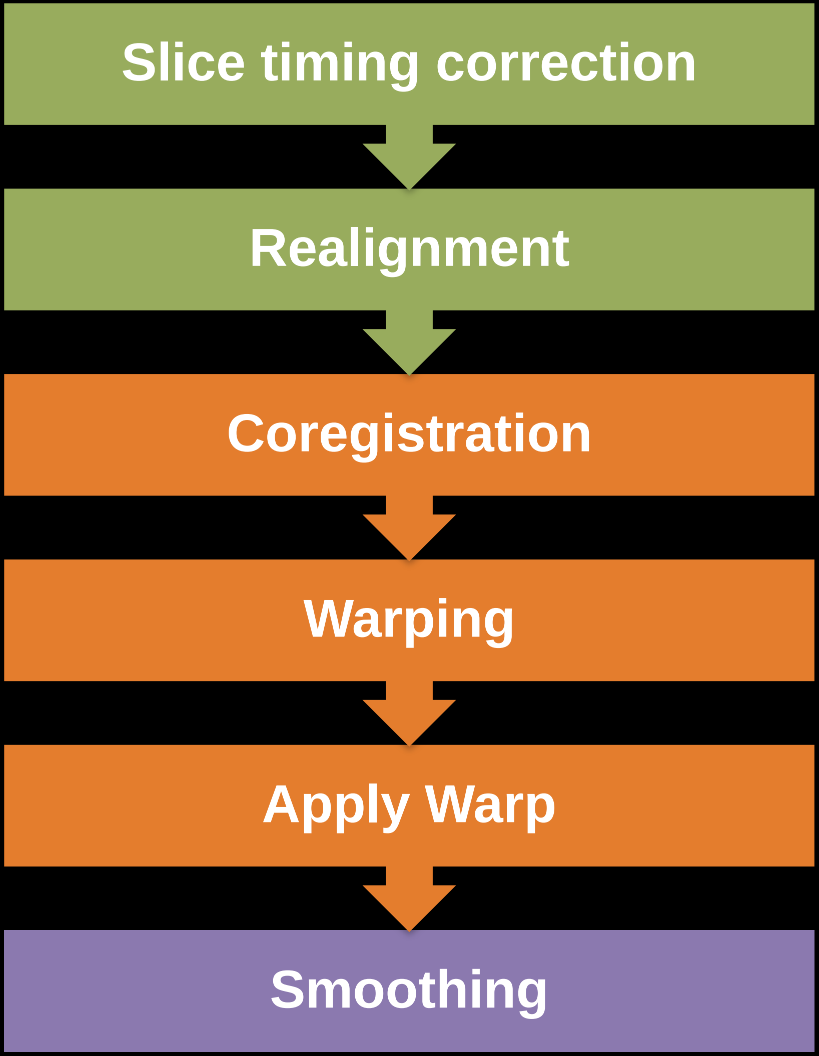


Note: All example commands will use this directory structure and are executed while at the directory Experiment_Name in a Unix shell.

Command Features

- Minimize user script writing
- Extensive logging of all processing
- Allows quality control between commands
- Allows for large scale data throughput
- Built-in help (just type command name at shell)
- Launches to background
- Email/text message notification
- Optimized performance

Processing Stream



Slice timing correction

Realignment

Coregistration

Warping

Apply Warp

Smoothing

sliceTime

realignfMRI

coregOverlay

coregHiRes

vbm8HiRes

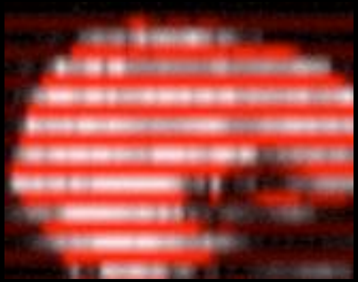
vbm8Check

warpfMRI

vbm8Check

smoothfMRI

sliceTime



This script uses FSL’s slicetimer
TR=2

TYPE THIS COMMAND BY ITSELF:

```
sliceTime
```

View information & available options. All commands have this feature.

sliceTime

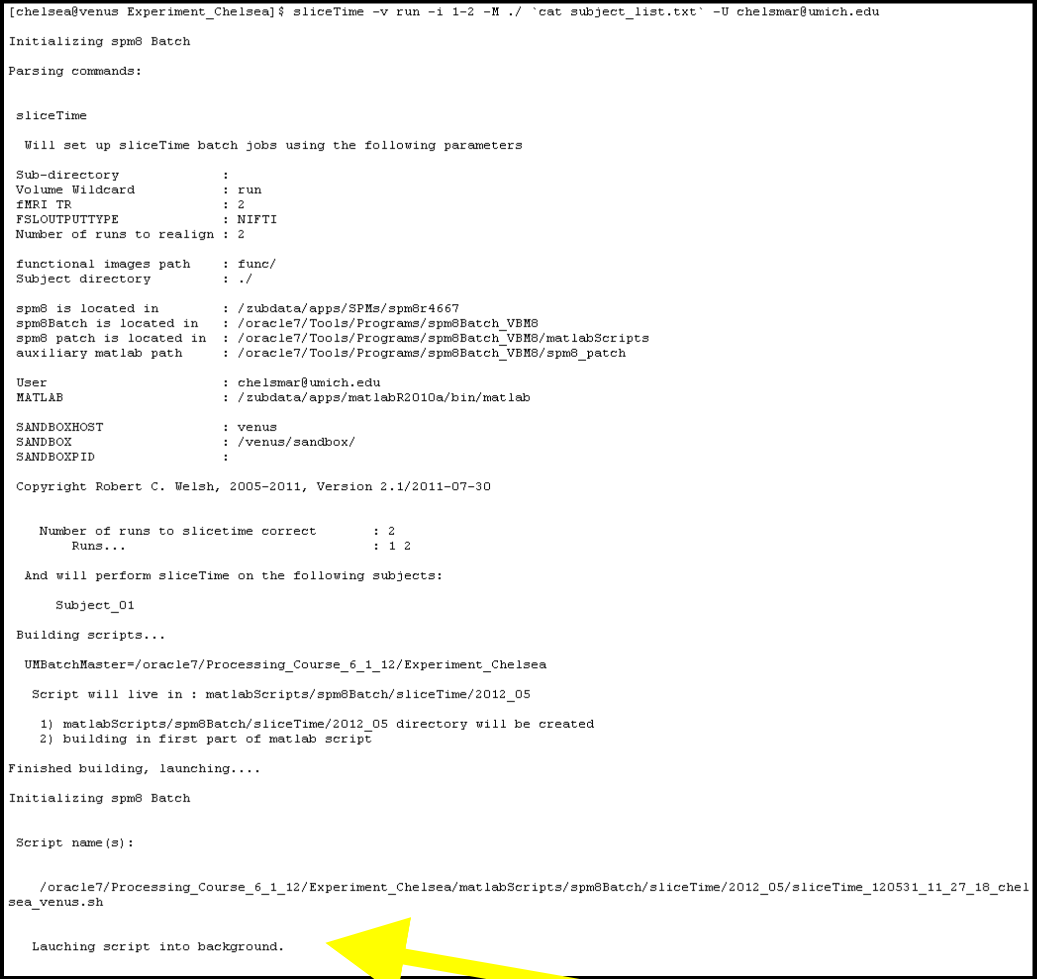
```
sliceTime –v run –i 1-2 –M ./ <Subjects>
-U youremail@umich.edu
```

sliceTime Options

-A		all runs present
-D		enable super debug flag
-d		enable debug flag
-f	[directory]	functional directory e.g. connect/func
-i	[#-#]	inclusive run list
-M	[directory]	master subject directory
-n	[name]	name prepend
-t		test flag
-U	[unique]	user email name/txt msg address
-v	[name]	volume name (name of functional file)
-#	[run number]	include this run number

sliceTime

This is what you should see after typing command:



```
[chelsea@venus Experiment_Chelsea]$ sliceTime -v run -i 1-2 -M ./ `cat subject_list.txt` -U chelsmar@umich.edu

Initializing spm8 Batch

Parsing commands:


 sliceTime

  Will set up sliceTime batch jobs using the following parameters

 Sub-directory              :
 Volume Wildcard            : run
 fMRI TR                    : 2
 FSLOUTPUTTYPE              : NIFTI
 Number of runs to realign : 2

 functional images path     : func/
 Subject directory          : ./

 spm8 is located in         : /zubdata/apps/SPMs/spm8r4667
 spm8Batch is located in    : /oracle7/Tools/Programs/spm8Batch_VBM8
 spm8 patch is located in   : /oracle7/Tools/Programs/spm8Batch_VBM8/matlabScripts
 auxiliary matlab path      : /oracle7/Tools/Programs/spm8Batch_VBM8/spm8_patch

 User                       : chelsmar@umich.edu
 MATLAB                     : /zubdata/apps/matlabR2010a/bin/matlab

 SANDBOXHOST                : venus
 SANDBOX                    : /venus/sandbox/
 SANDBOXPID                 :

 Copyright Robert C. Welsh, 2005-2011, Version 2.1/2011-07-30


    Number of runs to slicetime correct       : 2
        Runs...                               : 1 2

  And will perform sliceTime on the following subjects:

      Subject_01

 Building scripts...

  UMBatchMaster=/oracle7/Processing_Course_6_1_12/Experiment_Chelsea

   Script will live in : matlabScripts/spm8Batch/sliceTime/2012_05

    1) matlabScripts/spm8Batch/sliceTime/2012_05 directory will be created
    2) building in first part of matlab script

Finished building, launching....

Initializing spm8 Batch


 Script name(s):


    /oracle7/Processing_Course_6_1_12/Experiment_Chelsea/matlabScripts/spm8Batch/sliceTime/2012_05/sliceTime_120531_11_27_18_chel
sea_venus.sh


   Lauching script into background.
```

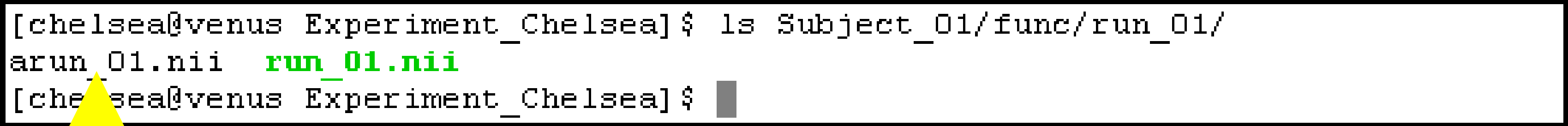
Tells you where the script will look for files, how many runs it's going to do, etc… Also where the logging files will be stored. If you see errors, this is a good place to start checking for what you did wrong.

Launching script into background = YAY!

sliceTime

If you typed everything correctly, you will get an email when sliceTime is complete.

List the contents of Subject_01's run_01 directory:

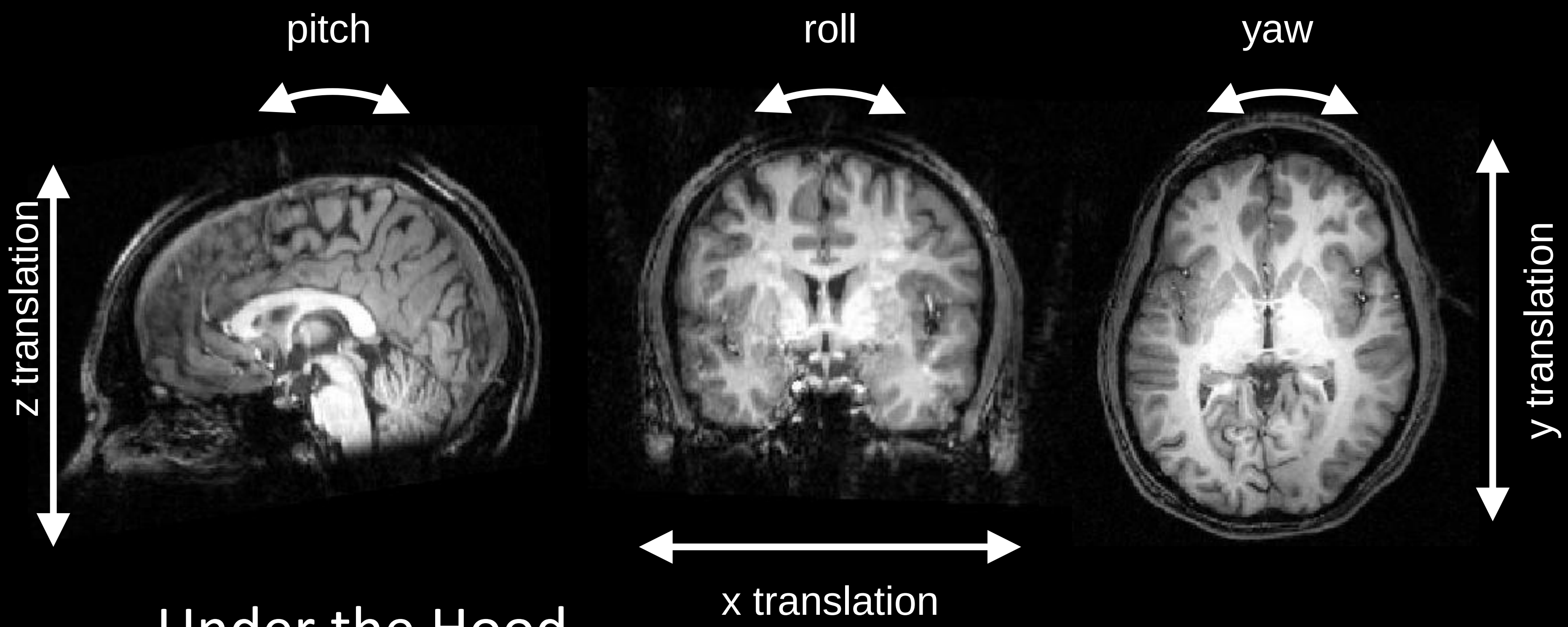


```
[chelsea@venus Experiment_Chelsea]$ ls Subject_01/func/run_01/
arun_01.nii  run_01.nii
[chelsea@venus Experiment_Chelsea]$ 
```

You should now see a sliceTime corrected run – arun_01.nii

Realignment

Remember: This step is to align each volume of the brain to a target volume using six parameters: three translations and three rotations



Under the Hood

- Builds shell script that calls FSL's *mcflirt*
- Default mcflirt options: "-cost normcorr -stats -plots"
- Default template is middle volume

Realignment

```
realignfMRI –v arun –i 1-2 –M ./ <Subjects>
-U youremail@umich.edu
```

realignfMRI Options

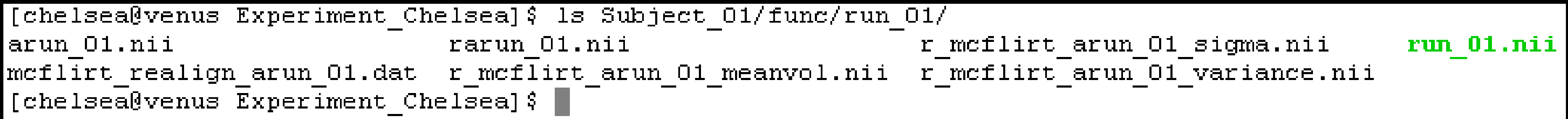
-A		all runs present
-D		enable super debug flag
-d		enable debug flag
-f	[directory]	functional directory e.g. connect/func
-i	[#-#]	inclusive run list
-M	[directory]	master subject directory
-m	[“options”]	mcflirt options
-n	[name]	name prepend
-S	[#]	standard volume number for mcflirt
-t		test flag
-U	[unique]	user email name/txt msg address
-v	[name]	volume name wild card
-#	[run number]	include this run number

Realignment

Launching Script into Background = YAY!

Another email

List the contents of the run directories again

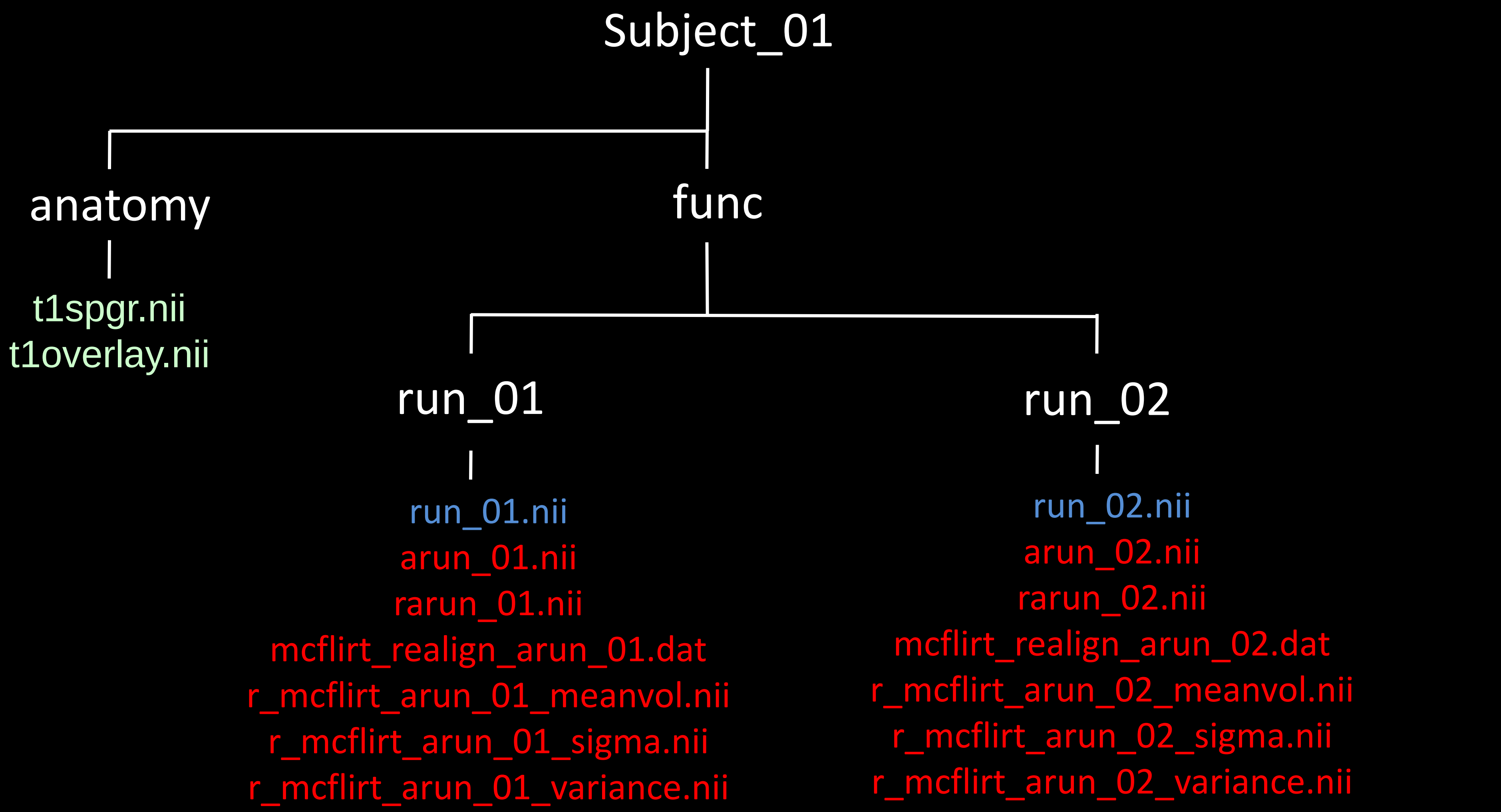


```
[chelsea@venus Experiment_Chelsea]$ ls Subject_01/func/run_01/
arun_01.nii                  rarun_01.nii                   r_mcflirt_arun_01_sigma.nii     run_01.nii
mcflirt_realign_arun_01.dat  r_mcflirt_arun_01_meanvol.nii  r_mcflirt_arun_01_variance.nii
[chelsea@venus Experiment_Chelsea]$ 
```

Output

- rarun_##.nii
- mcflirt_realign_arun.dat - transformation parameters
- r_mcflirt_arun_meanvol.nii - mean image after realign
- r_mcflirt_arun_sigma.nii - std over time
- r_mcflrit_arun_variance.nii - variance over time

Pause – A look at what happened so far



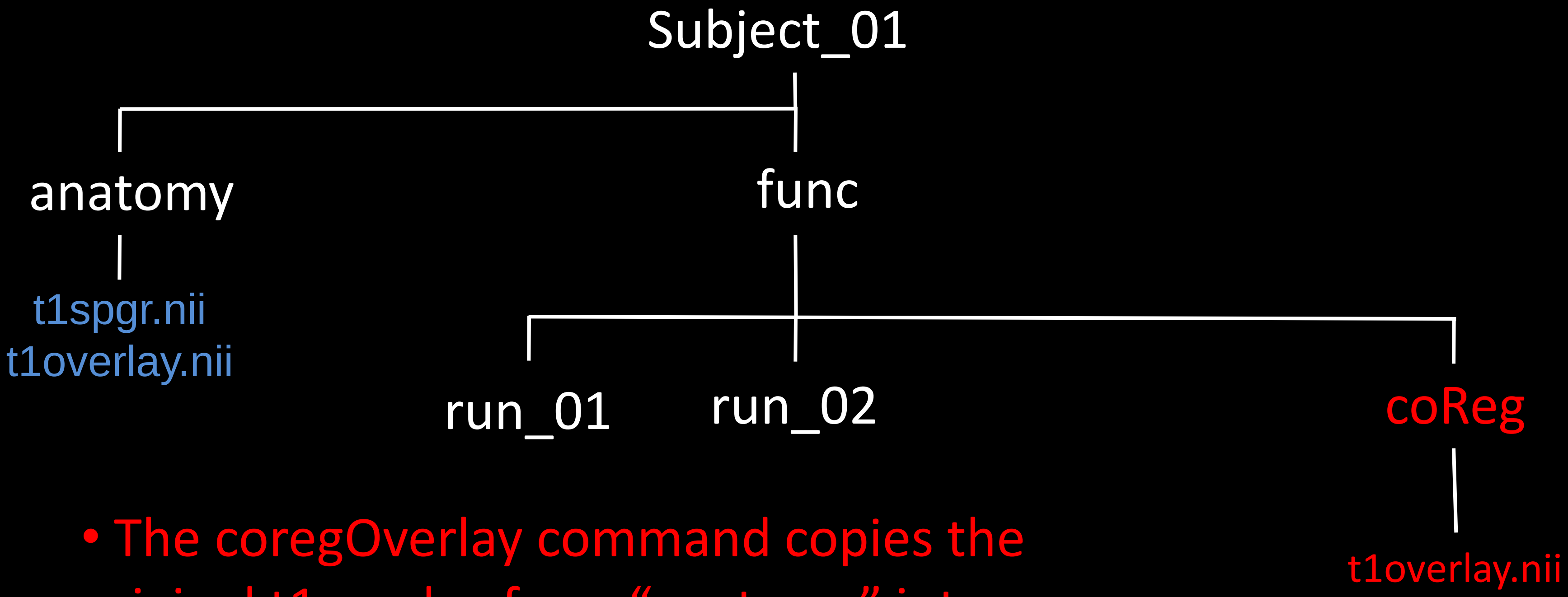
- Subject_01
  - anatomy
    - t1spgr.nii
    - t1overlay.nii
  - func
    - run_01
      - run_01.nii
      - arun_01.nii
      - rarun_01.nii
      - mcflirt_realign_arun_01.dat
      - r_mcflirt_arun_01_meanvol.nii
      - r_mcflirt_arun_01_sigma.nii
      - r_mcflirt_arun_01_variance.nii
    - run_02
      - run_02.nii
      - arun_02.nii
      - rarun_02.nii
      - mcflirt_realign_arun_02.dat
      - r_mcflirt_arun_02_meanvol.nii
      - r_mcflirt_arun_02_sigma.nii
      - r_mcflirt_arun_02_variance.nii

coregOverlay

```
coregOverlay –o t1overlay –v rarun –M ./
<Subjects> -U your_email@umich.edu
```

-a [directory]	set path for anatomy directory
-D	enable super debug flag
-d	enable debug flag
-f [directory]	functional directory e.g. connect/func
-M [directory]	master subject directory
-n [name]	name to add to output file name
-O	name of other files to process
-o [name]	name of overlay file
-r	set reslice flag
-s [directory]	set sub-directory under ‘run_##’
-t	test flag
-U [unique]	user email name/txt msg address
-v [name]	name of functional image
-w [directory]	set output path

coregOverlay



Subject_01

anatomy

t1spgr.nii
t1overlay.nii

func

run_01

run_02

coReg

t1overlay.nii

- The coregOverlay command copies the original t1overlay from “anatomy” into a new directory “coReg” located under “func”

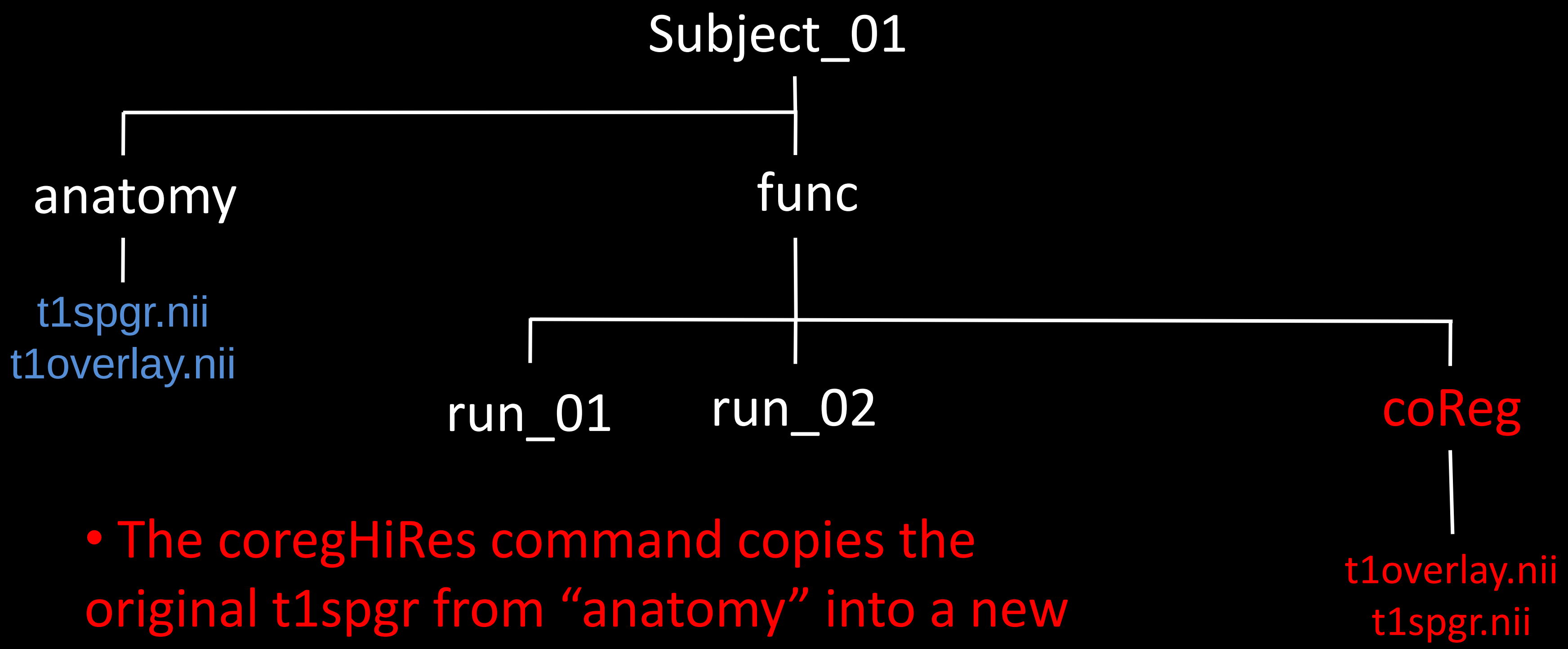
- The copied t1overlay is the one that is coregistered

coregHiRes

```
coregHiRes –h t1spgr –o t1overlay –M ./
<Subjects> -U your_email@umich.edu
```

-a	[directory]	set path for anatomy directory
-D		enable super debug flag
-d		enable debug flag
-f	[directory]	functional directory e.g. connect/func
-h	[name]	name of high resolution anatomical (t1spgr)
-M	[directory]	master subject directory
-n	[name]	name to add to output file name
-O		name of other files to process
-o	[name]	name of overlay file
-r		set reslice flag
-t		test flag
-U	[unique]	user email name/txt msg address
-w	[directory]	set output path

coregHiRes



- The coregHiRes command copies the original t1spgr from “anatomy” into a new directory “coReg” located under “func”

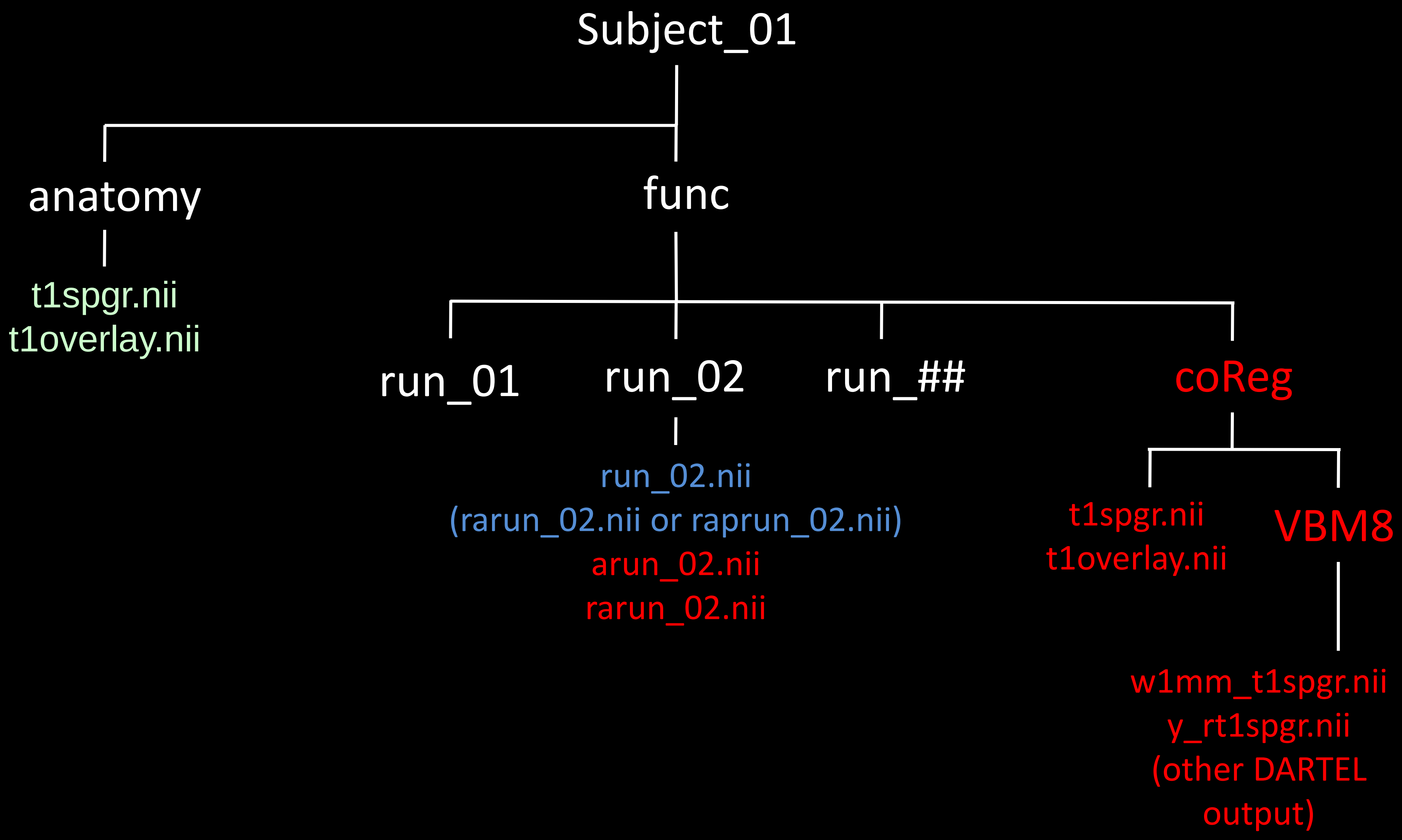
- The copied t1spgr is the one that is coregistered to the t1overlay in coReg (the one that is already in functional space)

DARTEL Warping – vbm8HiRes

```
vbm8HiRes   –h t1spgr   –a func/coReg
–w func/coReg/VBM8   –I r3mm_avg152T1_BET –n w3mm_
–M ./   <Subjects>   -U your_email@umich.edu
```

-a	[directory]	set path for anatomy directory
-D		enable super debug flag
-d		enable debug flag
-h	[name]	name of high resolution anatomical (t1spgr)
-I	[Ref name]	set the reference image to use for VBM8
-M	[directory]	master subject directory
-n	[name]	name to add to output file name
-O		name of other files to process
-t		test flag
-U	[unique]	user email name/txt msg address
-w	[directory]	set output path
-z		set the voxel size for resampling

Directory Structure – After vbm8HiRes



Subject_01

anatomy

t1spgr.nii
t1overlay.nii

func

run_01

run_02

run_02.nii
(rarun_02.nii or raprun_02.nii)
arun_02.nii
rarun_02.nii

run_##

coReg

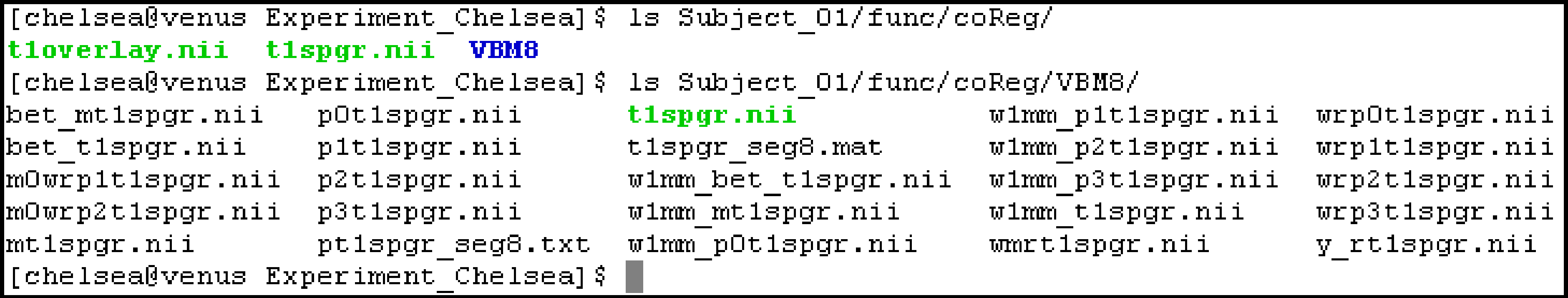
t1spgr.nii
t1overlay.nii

VBM8

w1mm_t1spgr.nii
y_rt1spgr.nii
(other DARTEL output)

DARTEL Warping – vbm8HiRes

List the contents of func/coReg to see new VBM8 directory.
List the contents of the VBM8 directory to see DARTEL output



```
[chelsea@venus Experiment_Chelsea]$ ls Subject_01/func/coReg/
t1overlay.nii  t1spgr.nii  VBM8
[chelsea@venus Experiment_Chelsea]$ ls Subject_01/func/coReg/VBM8/
bet_mt1spgr.nii   p0t1spgr.nii      t1spgr.nii           w1mm_p1t1spgr.nii  wrp0t1spgr.nii
bet_t1spgr.nii    p1t1spgr.nii      t1spgr_seg8.mat      w1mm_p2t1spgr.nii  wrp1t1spgr.nii
m0wrp1t1spgr.nii  p2t1spgr.nii      w1mm_bet_t1spgr.nii  w1mm_p3t1spgr.nii  wrp2t1spgr.nii
m0wrp2t1spgr.nii  p3t1spgr.nii      w1mm_mt1spgr.nii     w1mm_t1spgr.nii    wrp3t1spgr.nii
mt1spgr.nii       pt1spgr_seg8.txt  w1mm_p0t1spgr.nii    wmrt1spgr.nii      y_rt1spgr.nii
[chelsea@venus Experiment_Chelsea]$
```

Quality Checks – vbm8Check

A quality check tool with FSL:

vbm8Check – 3 steps

0. check skull stripping
1. check registration between spgr & template
2. (later)

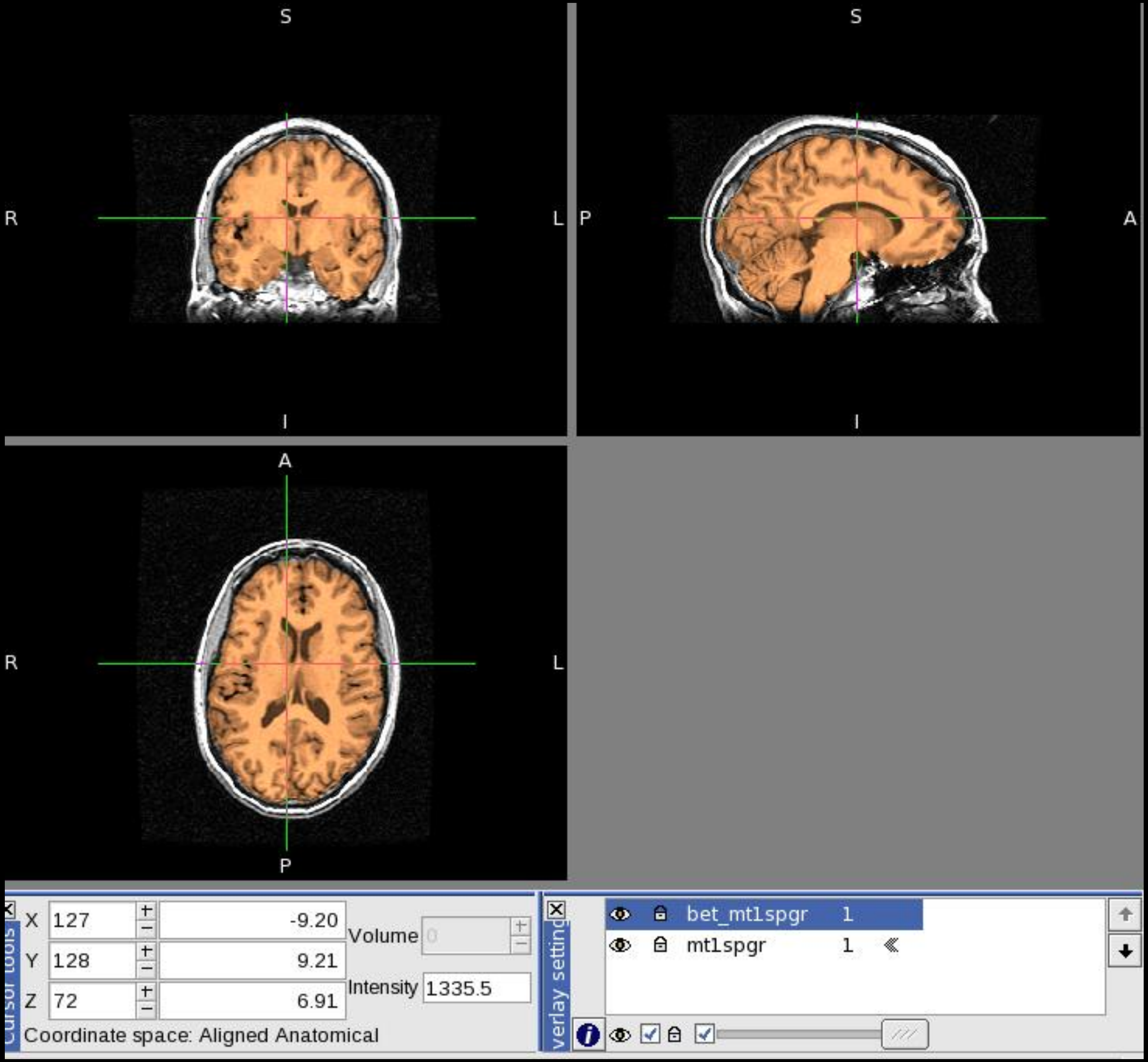
Quality Checks – vbm8Check

0. skull stripping of spgr

TYPE:

```
vbm8Check -a func/coReg/VBM8 –h t1spgr –M ./ <Subjects>
```

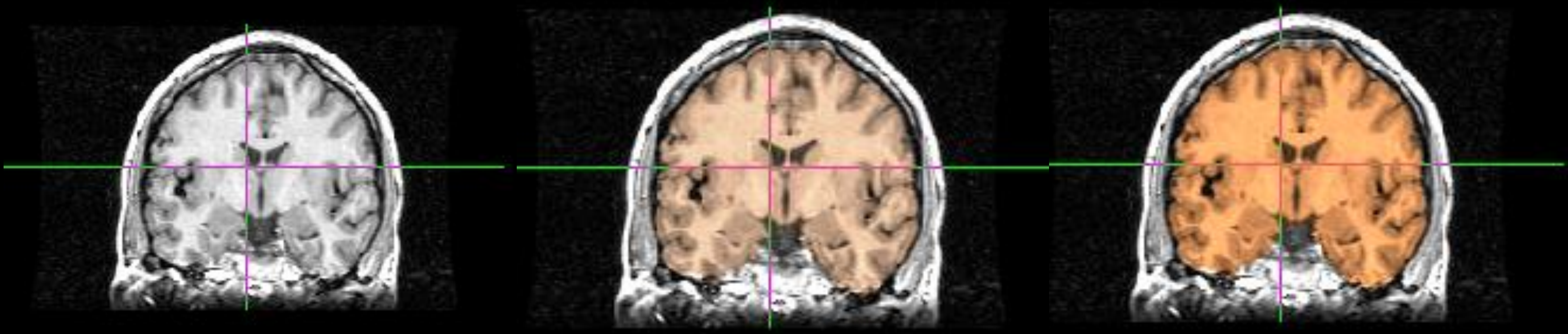
Quality Checks – vbm8Check

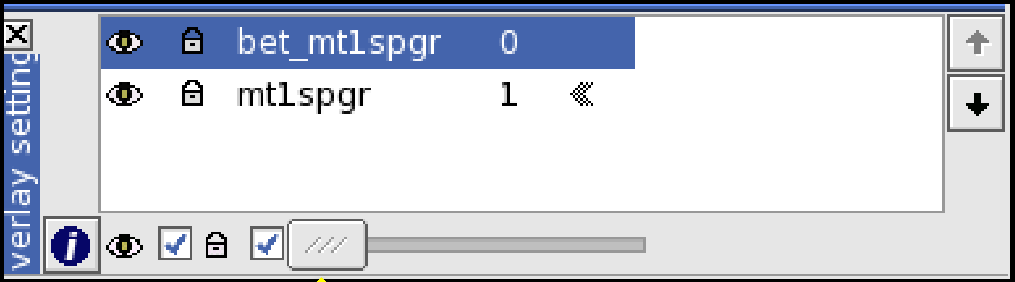


X 127 -9.20
Y 128 9.21
Z 72 6.91
Volume 0
Intensity 1335.5
Coordinate space: Aligned Anatomical

bet_mt1spgr 1
mt1spgr 1

Quality Checks – vbm8Check





verlay setting

👁	🔒	bet_mt1spgr	0
👁	🔒	mt1spgr	1

Scroll this button back and forth to fade between spgr and skull-stripped spgr. Click around the brain and look at different views. Close FSL

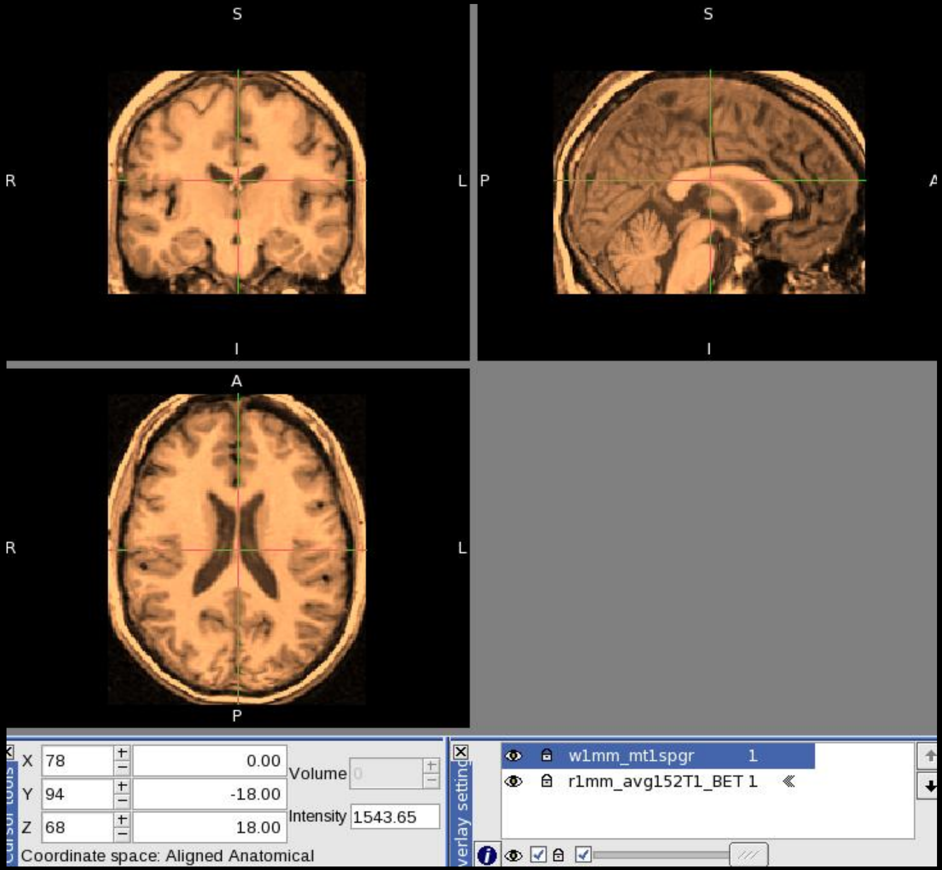
Quality Checks – vbm8Check

1. Registration between spgr & template

TYPE:

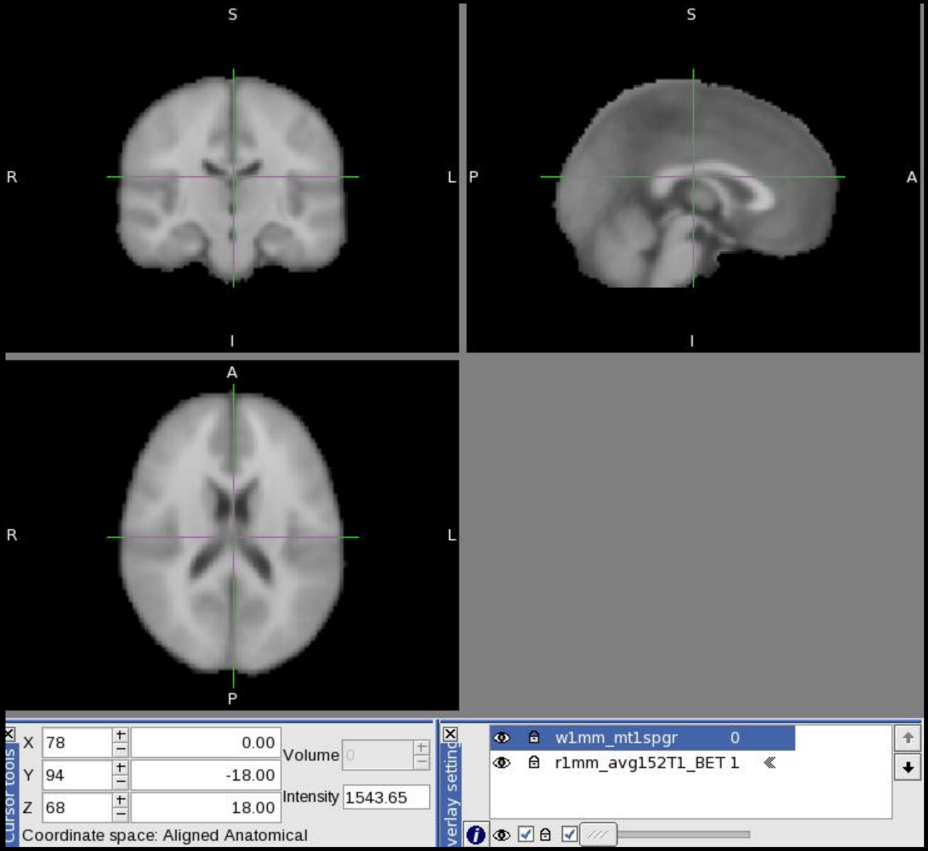
```
vbm8Check -a func/coReg/VBM8
–h w3mm_mt1spgr  -1
–l r3mm_avg152T1_BET
–M ./ <Subjects>
```

Quality Checks – vbm8Check



X 78 0.00
Y 94 -18.00
Z 68 18.00
Volume 0
Intensity 1543.65
Coordinate space: Aligned Anatomical

w1mm_mt1spgr 1
r1mm_avg152T1_BET 1



X 78 0.00
Y 94 -18.00
Z 68 18.00
Volume 0
Intensity 1543.65
Coordinate space: Aligned Anatomical

w1mm_mt1spgr 0
r1mm_avg152T1_BET 1

Use the little button again to fade back and forth between the spgr and the template checking that they match up. Close FSL

Apply Warp – warpfMRI

```
warpfMRI   –h t1spgr   –w coReg/VBM8
–I r3mm_avg152T1_BET   –n w3mm_ -v rarun   –W   –M ./
<Subjects>   -U your_email@umich.edu
```

-f	[directory]	set path to functional directory
-h	[name]	name of high resolution anatomical (t1spgr)
-I	[Ref name]	set the reference image to use for VBM8
-M	[directory]	master subject directory
-n	[name]	name to add to output file name
-t		test flag
-U	[unique]	user email name/txt msg address
-v	[name]	name of functional volume
-W		Enable VBM8 (DARTEL) warping for fMRI
-w	[directory]	set output path
-z		set the voxel size for resampling

Quality Checks –vbm8Check

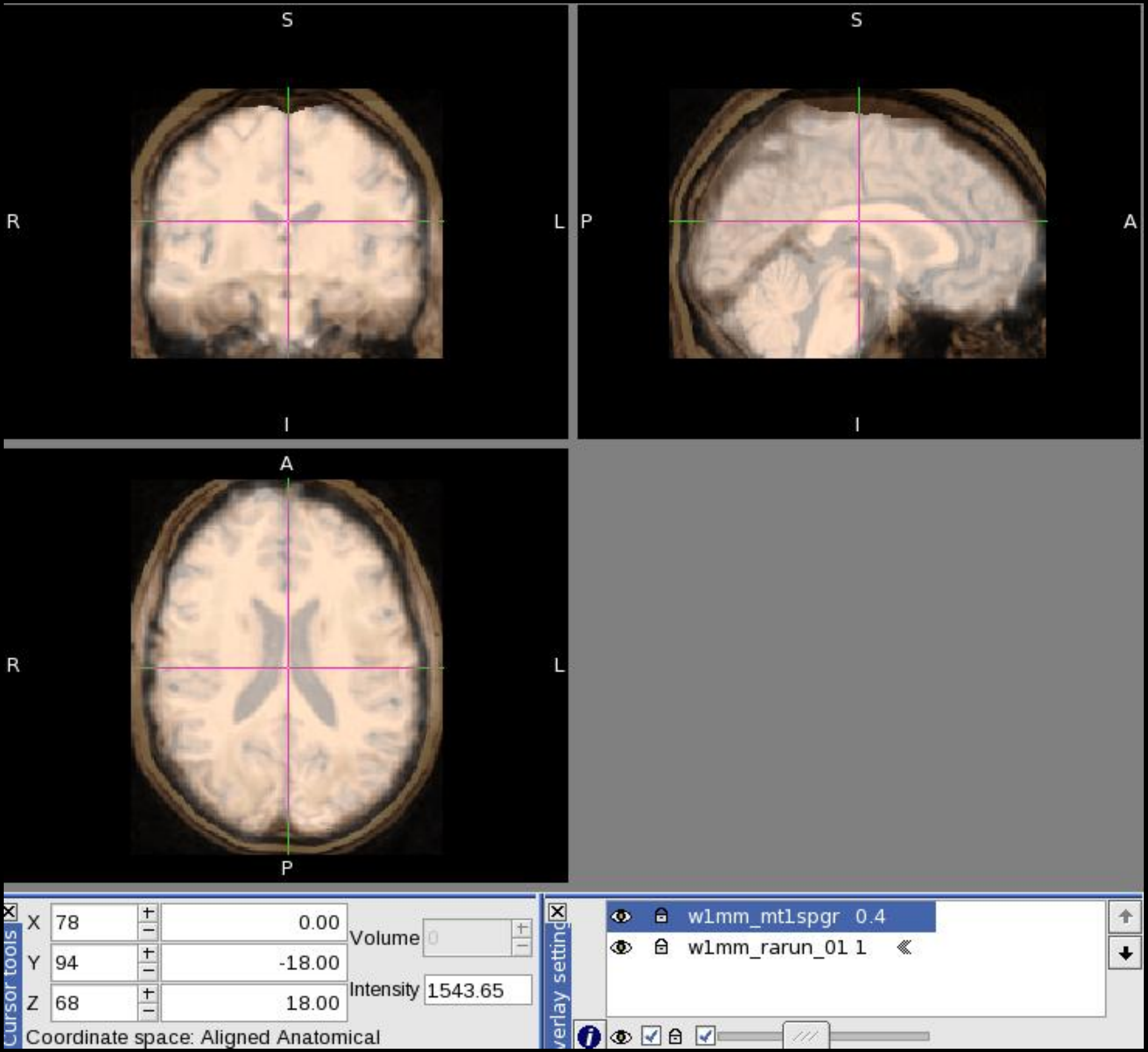
vbm8Check

2. Registration between spgr & functionals

TYPE:

```
vbm8Check   -a func/coReg/VBM8
–h w3mm_mt1spgr    -2   –v w3mm_rarun
–M ./ <Subjects>
```

Quality Checks – vbm8Check



X 78 0.00
Y 94 -18.00
Z 68 18.00

Volume 0

Intensity 1543.65

Coordinate space: Aligned Anatomical

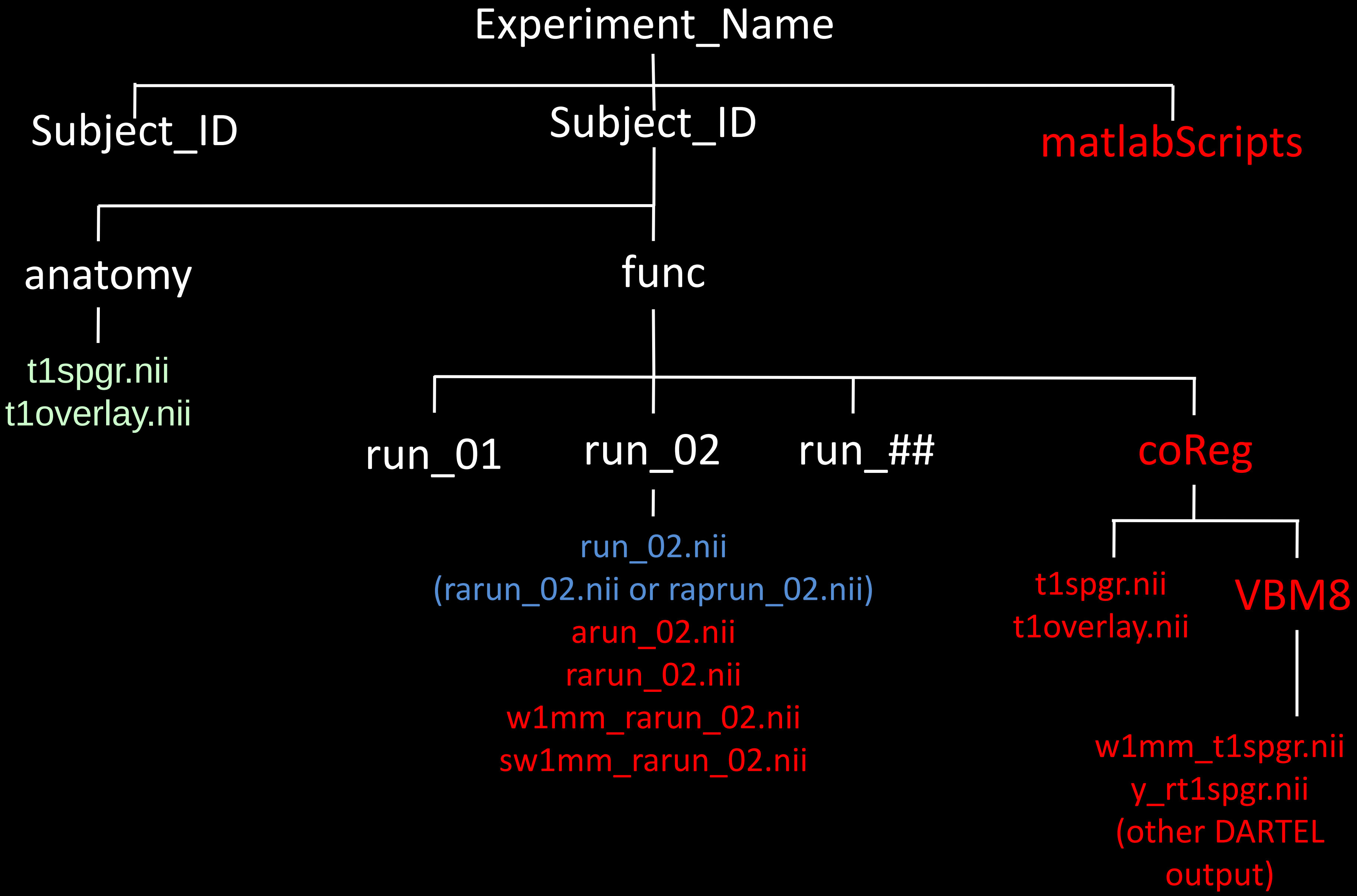
w1mm_mt1spgr 0.4
w1mm_rarun_01 1

Smoothing - smoothfMRI

```
smoothfMRI   5 5 5   –v w1mm_    –M ./
<Subjects>   -U your_email@umich.edu
```

-D	enable super debug flag
-d	enable debug flag
-f [directory]	set path to functional directory
-M [directory]	master subject directory
-n [name]	name to add to output file name
-t	test flag
-U [unique]	user email name/txt msg address
-v [name]	name of functional volume

Directory Structure – After Processing



Experiment_Name

Subject_ID

Subject_ID

matlabScripts

anatomy

t1spgr.nii
t1overlay.nii

func

run_01

run_02

run_02.nii
(rarun_02.nii or raprun_02.nii)
arun_02.nii
rarun_02.nii
w1mm_rarun_02.nii
sw1mm_rarun_02.nii

run_##

coReg

t1spgr.nii
t1overlay.nii

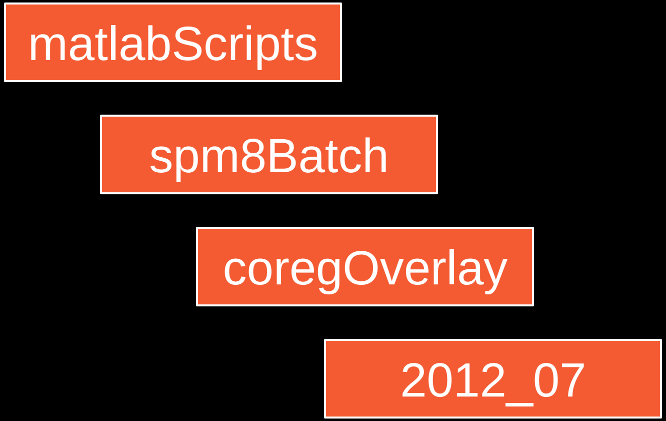
VBM8

w1mm_t1spgr.nii
y_rt1spgr.nii
(other DARTEL output)

Logging

When a process is run, the command will automatically configure a “job” by writing a shellscript file and as needed a matlab script file. These files will be created in a directory specified by the name of the command being issued and then further segregated by the year and month. The “job” files have names that are based on the command issued, the date/time, the user and the computer node. Once the command completes building the process it will launch into the background for execution, releasing the interactive terminal.

Example:



matlabScripts

spm8Batch

coregOverlay

2012_07

coregOverlay_120722_09_37_18_heffjos.sh
coregOverlay_120722_09_37_18_heffjos.m
coregOverlay_120722_09_37_18_heffjos.log