# wb_command and HCP Pipelines I

## Part 1.  wb_command

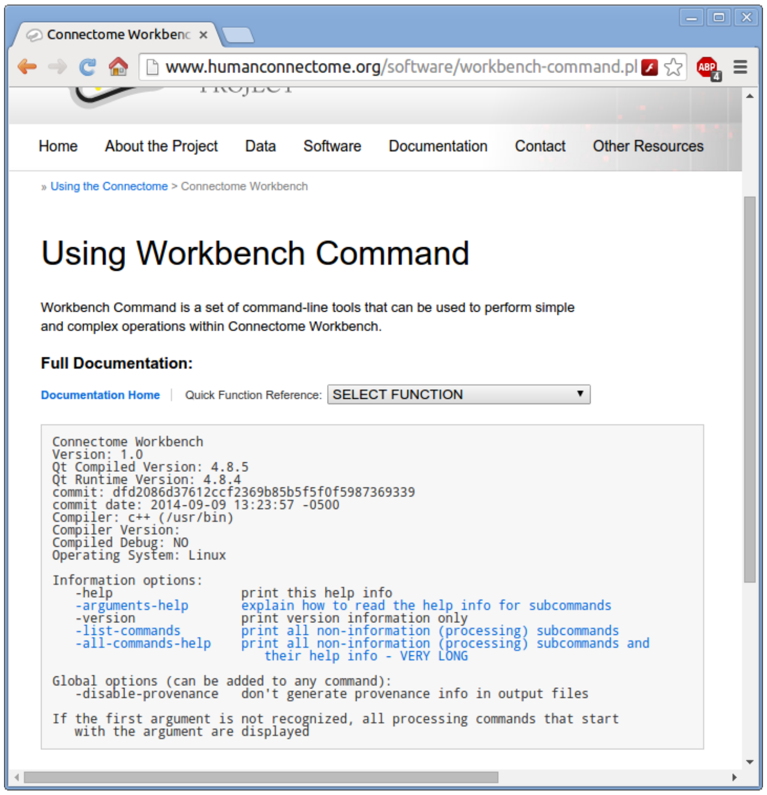
### Basic use of the command line

- "pwd" - print working directory, shows what folder you are currently in
- "ls" - list files and folders in the current directory
- "cd <directory>" - change directory, moves into the specified folder - use the special folder name ".." to move to the parent folder
- "cp <file> <new file>" - copy a file
- "mv <file> <new location>" - move, renames a file and/or puts it in a different folder
- "man <command>" - get information on options and usage of most command line utilities
- You can scroll in the terminal to view text that has passed the top of the terminal window.

### Introduction to wb_command

There are several ways to access the help info for wb_command:

1) wb_command -help
2) [http://humanconnectome.org/software/workbench-command.php](http://humanconnectome.org/software/workbench-command.php) - tracks the current released version (see screen capture below)
3) wb_view, Help menu, Workbench Help..., wb_command - reflects the release in use, but not additional changes in a development version


Connectome Workbenc ×
www.humanconnectome.org/software/workbench-command.pl
Home
About the Project
Data
Software
Documentation
Contact
Other Resources
» Using the Connectome > Connectome Workbench

Using Workbench Command

Workbench Command is a set of command-line tools that can be used to perform simple and complex operations within Connectome Workbench.

**Full Documentation:**

Documentation Home | Quick Function Reference: SELECT FUNCTION

```
Connectome Workbench
Version: 1.0
Qt Compiled Version: 4.8.5
Qt Runtime Version: 4.8.4
commit: dfd2086d37612ccf2369b85b5f5f0f5987369339
commit date: 2014-09-09 13:23:57 -0500
Compiler: c++ (/usr/bin)
Compiler Version:
Compiled Debug: NO
Operating System: Linux

Information options:
   -help                print this help info
   -arguments-help      explain how to read the help info for subcommands
   -version             print version information only
   -list-commands       print all non-information (processing) subcommands
   -all-commands-help   print all non-information (processing) subcommands and
                           their help info - VERY LONG

Global options (can be added to any command):
   -disable-provenance   don't generate provenance info in output files

If the first argument is not recognized, all processing commands that start
   with the argument are displayed
```

To get the help information for a processing command, run it with no arguments:

- In your terminal window, enter

  ```
  wb_command -cifti-all-labels-to-rois
  ```

  This command has short and simple instructions (even if the particulars are not yet clear).

- Enter

  ```
  wb_command -cifti-math
  ```

  This is a more complex command, with many options and explanatory comments. You are not expected to figure out what these options mean yet. They illustrate the complexity you may encounter in command arguments – just recognize that this method of displaying the syntax for a command is a go-to resource when you are trying to implement particular commands.

  The options of a command use a tree structure, denoted by the indentation of the parameter in the help information. You must finish dealing with a “branch” of this tree, including all options you want in it, before starting a new branch at the same level, or supplying more options or parameters to an earlier level. An easy way to get this right is to supply the arguments and options you want in the order that they are listed in the help information.

- Enter

  ```
  wb_command -arguments-help
  ```

  The example it contains should help with the first exercise below.

Processing commands are generally ordered by the type of files that are used as data input. Thus, -cifti-math, -metric-math, and -volume-math, and many other operations are “duplicated” for different input types. Part of the reason for this is that for operations that use spatial information (smoothing, gradient, dilation, etc), the different data file types require different additional inputs (metric file inputs need a single matching surface file, volume file inputs already have spatial information, and cifti files generally need 2 surface files, one for each cortical hemisphere).

Commands do not try to ‘guess’ the intended file extension on input or output filenames. You must specify the filename with the extension on it (e.g., if you want to use a file named “some_vol.nii.gz”, specify “some_vol.nii.gz” on the command line, not “some_vol”).

If you specify “.nii.gz” on an output volume file, it will compress the file, whereas specifying an output with just “.nii” will not compress it. Note that there is no similar behavior for GIFTI files (you cannot do “.func.gii.gz”, etc). Also, CIFTI files should never be compressed while in use; you will get an error if you specify “.gz” on the end of a CIFTI output filename.

The features of the CIFTI file type make the -cifti-* commands operate somewhat differently. Notably, CIFTI allows any mapping type to be used on any dimension, so many -cifti-* commands will take an argument of “direction”, specifying which dimension of a file the command should operate on. Specifying “ROW” means to operate on the mapping that is oriented along the rows of the CIFTI file. In dtseries, dscalar, and dlabel files, the spatial mapping (“dense”, “brain models”) is along the columns, so for these file types you will want to specify COLUMN for gradient, smoothing, and other spatial operations. In dconn files, both dimensions have a dense mapping, so it should be noted that when viewing a dconn, clicking a point loads a single row from the dconn file.

## Exercises

Use the given commands to conduct some processing operations, comparing them to the help information of the command in order to understand how the parts of the command line map to the inputs, outputs, and options. You will need to use the exact output filenames given, so that the scene for displaying the results will work.

In the following exercises you will use 3 commands to process exemplar group-average task-fMRI data in ways that will be informative relative to subsequent lectures and practicals.

- In a terminal window, go the day1-monday/wb_command_intro directory.
- Type “ls" to see the nine files needed for these exercises.

1) *Threshold a z-statistic task-fMRI dscalar file using -cifti-math.*

Your first objective is to use -cifti-math to threshold the z-statistic image “Related210_AllTasks_zstat.dscalar.nii” at (+/-)5.0088, which is the two-tailed Bonferroni corrected (for number of grayordinates) z threshold at p=0.05. You also want to assign the positive and negative passing regions values of 1 and 2, respectively, and to save the result as “Related210_AllTasks_signif.dscalar.nii”

Since it will take most students more time than we have available to learn how to use this command from scratch, we provide you with the precise command:

- Cut and paste the following command into the terminal window, with only spaces separating the arguments, ignoring the line wrapping done to fit the line on the page (that is why the “\”s are added:

```
wb_command -cifti-math '(x > 5.0088) + 2 * (x < -5.0088)' \
Related210_AllTasks_signif.dscalar.nii -var x \
Related210_AllTasks_zstat.dscalar.nii
```

The expression '(x > 5.0088) + 2 * (x < -5.0088)' (with the single quotes) achieves the desired Bonferroni correction given the number of grayordinates (91282) involved. The command will output a message to the terminal, indicating how it parsed the expression. This can help identify problems caused by order of operations or missing quotes.

2) *Convert the thresholded z-statistic map into a dlabel file for localizing its boundaries.*

Your second objective is to apply -cifti-label-import on this new file, with the pre-made label definition file “posneg.txt” (you can view this file with “less posneg.txt” or “gedit posneg.txt”). Save the result to “Related210_AllTasks_signif.dlabel.nii”. The main reason for making it a label file is because wb_view has a display mode for label data that only shows the outline of the labeled areas.

- Cut and paste the following command into the terminal window:

```
wb_command -cifti-label-import Related210_AllTasks_signif.dscalar.nii \
posneg.txt Related210_AllTasks_signif.dlabel.nii
```

3) *Generate a spatial gradient on a task-fMRI ‘beta map’, showing the spatial locations where the task effect size changes most rapidly.*

Your third objective is to run -cifti-gradient on the “Related210_AllTasks_beta.dscalar.nii” file, saving the result to “Related210_AllTasks_beta_grad.dscalar.nii”. To calculate the spatial gradients accurately, it is important to use information available in the cortical midthickness surfaces and the vertex area metric files (.func.gii) that are provided. You are asked to use 1mm sigma presmoothing on surface and in the volume. (Note that higher amounts of smoothing make the command take longer to run, and the course machines have limited computing capability).

- Cut and paste the following command into the terminal window:

```
wb_command -cifti-gradient Related210_AllTasks_beta.dscalar.nii COLUMN \
Related210_AllTasks_beta_grad.dscalar.nii -left-surface \
Related210.L.midthickness.32k_fs_LR.surf.gii -left-corrected-areas \
Related210_AvgVertArea.L.midthickness.32k_fs_LR.func.gii -right-surface \
Related210.R.midthickness.32k_fs_LR.surf.gii -right-corrected-areas \
Related210_AvgVertArea.R.midthickness.32k_fs_LR.func.gii -surface-presmooth \
1 -volume-presmooth 1
```

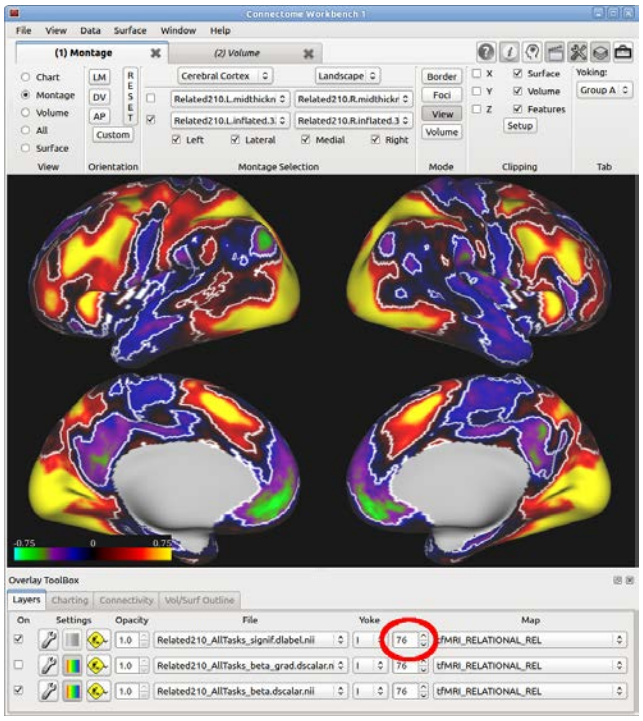
- Once these steps have completed, run

```
wb_view results.scene
```

- Load the first scene to see the results:

[*If an error message appears, consult a course instructor, as one or more of the 3 files created in the exercise above may have been improperly named or generated.*]

This scene shows results for a particular task contrast (#76, the “tfMRI_RELATIONAL_REL” pattern matching task) from a group-average analysis of 210 HCP subjects. It shows the effect size (beta) map, which represents the % change of the BOLD fMRI signal related to the task



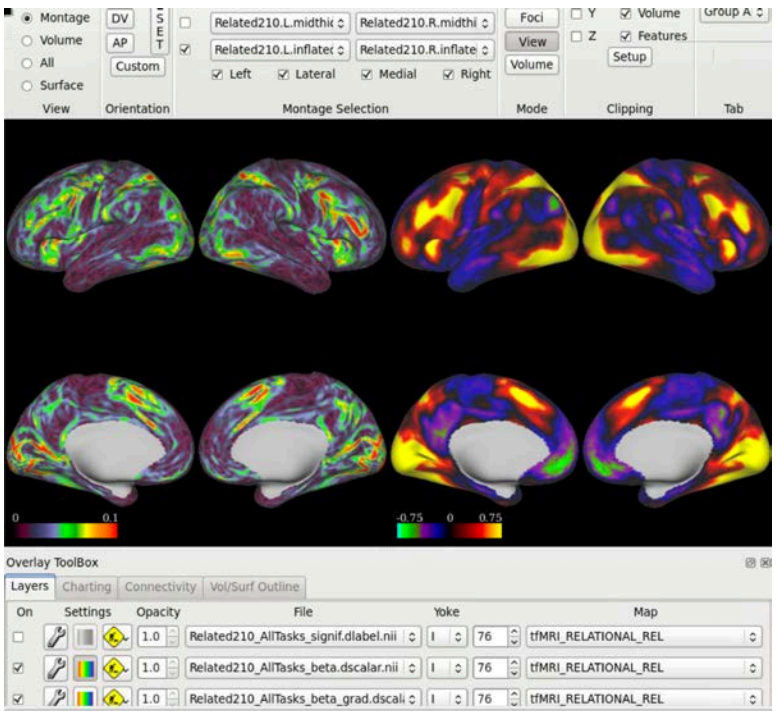
| On | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|
| ✓ | 1.0 | Related210_AllTasks_signif.dlabel.nii | I | 76 | tfMRI_RELATIONAL_REL |
| | 1.0 | Related210_AllTasks_beta_grad.dscalar.n | I | 76 | tfMRI_RELATIONAL_REL |
| ✓ | 1.0 | Related210_AllTasks_beta.dscalar.nii | I | 76 | tfMRI_RELATIONAL_REL |

contrast. Additionally, white outlines enclose regions that are statistically significant relative to chance (significantly positive activations have bright yellows and reds inside them; significantly negative "de-activations" enclose bright blue and green regions; non-significant regions are darker). These white outlines are from the dlabel file you created by thresholding the statistical significance map (z-statistic) in steps 1 and 2 above; they appear as outlines rather than solidly filled regions thanks to a wb_view display feature you will learn about in a later practical.

- To view a different task contrast from the same analysis, *type* "64" into the circled box and *press enter*. This shows group-average results for the "LANGUAGE_STORY" task, which entails listening to stories in the scanner).
- *Do the same* with "22", in order to compare the effect sizes of a third task contrast ("TOOL-AVG", which shows responses to viewing tools specifically vs the average of four object categories).
- *Switch* to tile tabs view (CTRL 'M') to view the subcortical results alongside the surface maps.

For analyses involving different numbers of subjects, the effect size required to achieve statistical significance will increase or decrease, causing the regions above the significance threshold to grow or shrink, but the values of the effect size map should remain comparable across studies. Note that the first two task contrasts (#1 and #64) have relatively strong effect sizes, whereas the third (#22) is weaker. The analysis and interpretation of such results will be further discussed in subsequent sessions.

- *Double-click* the second scene (gradients plus beta maps)) to show the spatial gradients of the effect size maps that you created in step 3 above (tab 1) next to the effect size (beta) maps (tab 2) in tile tabs view.



| On | Settings | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|---|
| ☐ | | 1.0 | Related210_AllTasks_signif.dlabel.nii | I | 76 | tfMRI_RELATIONAL_REL |
| ✓ | | 1.0 | Related210_AllTasks_beta.dscalar.nii | I | 76 | tfMRI_RELATIONAL_REL |
| ✓ | | 1.0 | Related210_AllTasks_beta_grad.dscali | I | 76 | tfMRI_RELATIONAL_REL |

The gradient provides an objective measure of how much the task activity changes as you move across the surface from one location to another. Reds, yellows, greens, and even light blues are regions of the gradient map where large, rapid changes in task activity occur; dark blues, purples, and blacks show regions of relatively stable task activation (whether the absolute level is high or low). Sharp changes in task activity (or connectivity, architecture, etc) are thought to often occur at the boundaries of brain areas. Gradients and effect size maps can be used together to see if different studies found the same level of activity with the same boundaries, a particularly stringent test of cross-study comparison and reproducibility. These issues will also be discussed further in subsequent sessions.

Both scenes have the same files set up in the layers, with some toggled off, so you can toggle the top two layers to view the outlines, gradient, and/or beta map as desired.

### Commonly used wb_command options.

Some commonly used workbench commands are:

- -*-math: does arbitrary elementwise math expressions on data files
- -cifti-separate: creates metric, label, and/or volume files from a CIFTI file
- -cifti-create-*, -cifti-replace-structure: takes data from metric/label/volume files, and puts it into a CIFTI file
- -*-merge: concatenate data files, or select columns/subvolumes from them
- -cifti-convert: dump the cifti data matrix into other formats, for elementwise use in programs that don't understand CIFTI (and don't need spatial information)
- -*-label-import: make a label file from a metric, volume, or CIFTI file that contains integer-value data
- -file-information: summary information of a data file

## Part 2. HCP Pipelines I

### Introduction

- The goal of this session is to get participants familiar with the *HCP Pipeline Scripts* including:
  - Where and how they can be obtained and installed
  - What prerequisites are there for running them, both software and hardware
  - What are the data organization assumptions made
  - How to modify the example batch scripts provided as part of the HCP Pipeline Scripts in order to run the scripts on your own data
- The HCP Pipeline Scripts are open source tools that are available in a public source code

repository

- They are described in Glasser et al. 2013[1]

## Flow and Dependencies

- The flow and dependencies for the HCP Pipelines are shown in Figure 1.

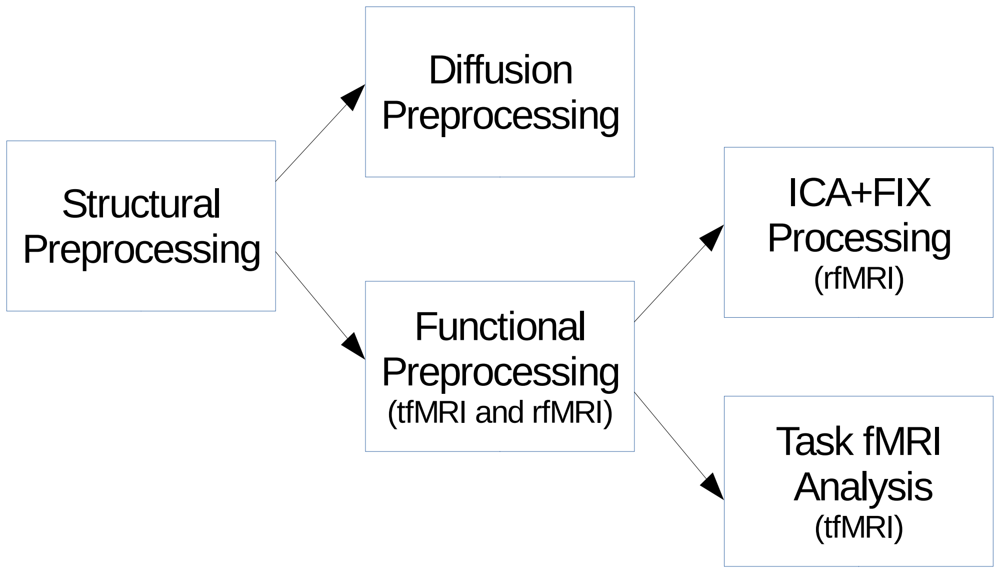

Structural Preprocessing
Diffusion Preprocessing
Functional Preprocessing (tfMRI and rfMRI)
ICA+FIX Processing (rfMRI)
Task fMRI Analysis (tfMRI)

*Figure 1: HCP Pipelines Flow and Dependencies*

## The HCP Pipeline Scripts GitHub Repository

- The repository containing the HCP Pipeline Scripts is located at: https://github.com/Washington-University/Pipelines
- The best way to get a copy of the scripts is to download a **Release** from the repository.
- Releases are available at https://github.com/Washington-University/Pipelines/releases
    - Be careful to note whether you are getting an actual "Release" version of the code or a "Pre-release" version
    - Pre-release versions have an orange Pre-release marker to the left of them
    - Pre-release versions are NOT FULLY TESTED. They are generally made available for Alpha

1 http://www.ncbi.nlm.nih.gov/pubmed/23668970

and Beta testing

- If you want the version of the HCP Pipeline Scripts that were used (and are still being used) for processing the standard HCP Phase II 3T Skyra data, get version 3.4.0
- Selecting a release link (e.g. *v3.4.0 – First Public Release*[2]) will take you to a page where you can choose whether to download a ZIP compressed file or a GZIP compressed tar archive.

- In general, downloading and unpacking the compressed file to a location from which you will run the scripts is all the installation that is necessary.
  - You may want to put them someplace that is accessible to multiple users on your system
  - You may also want to rename the root directory (e.g. Pipelines-3.4.0 to just Pipelines)

## The HCP Pipeline Scripts Wiki

- Another useful resource for setting up and running the pipelines is the HCP Pipelines Wiki at https://github.com/Washington-University/Pipelines/wiki
- There you will file the *Release Notes, Installation, and Usage Guide.*
- Also found there are a FAQ, a version numbering policy, and some information about versions that existed prior to the first public release of the scripts.
- Contributions to the wiki from the user community are welcome. It's not currently publicly editable, but contributions can be made by submitting them to Tim Brown at tbbrown@wustl.edu or by submitting them to the hcp-users discussion list. You can sign up for the hcp-users discussion list at http://humanconnectome.org/contact/#subscribe.There is already a user contributed guide to setting up the pipelines to run under Mac OSX.

## Prerequisite Software

- It is important to take note of the Prerequisites listed in the *Release Notes, Installation, and Usage Guide*
- In general, those prerequisites are:
  - A 64-bit Linux Operating System (CentOS, Ubuntu, etc.)
  - The FMRIB Software Library (a.k.a. FSL)
    - Version 5.0.6 (not later, not earlier) is required to run version 3.4.0 of the pipelines. Version 3.4.0 of the pipelines is being used for all standard releases of 3T data for the HCP project starting with the S500 release.

2 https://github.com/Washington-University/Pipelines/releases/tag/v3.4.0

    - The repository contains newer (pre-release) versions of the scripts that should work with version 5.0.7 and above.
  - FreeSurfer
    - There is an HCP customized version of FreeSurfer 5.3.0 that is required (5.3.0-HCP).
    - The changes in the 5.3.0-HCP are expected to be incorporated into any future releases of FreeSurfer
    - When you install FreeSurfer to use on your own systems, you will need to remember to create and install a license file for FreeSurfer by visiting and submitting the FreeSurfer registration form.
  - Connectome Workbench version 1.0 or above
  - The HCP version of *gradunwarp* is necessary for gradient nonlinearity correction
    - Version 1.0.2 at https://github.com/Washington-University/gradunwarp/releases
    - gradunwarp also has its own set of prerequisites, which include Python >= v2.7 (but not version 3.x), Numpy, Scipy, nibabel, and PyDICOM
    - To perform gradient nonlinearity correction you will also need a gradient coefficients file for your specific scanner

## Prerequisite Hardware

- The memory and processing time requirements for running the *HCP Pipeline Scripts* are relatively high. We do not have specific processor type requirements, but the following reference points should help.
- The HCP generally runs these script by submitting them to a grid engine managed by a Portable Batch System (PBS) job scheduler which schedules the jobs on compute nodes in a cluster available at the Washington University Center for High Performance Computing (CHPC)[3].
- Even with these relatively high-powered compute nodes, processing can take quite a while as is illustrated below.
- Note that the term “Walltime” used below refers to time that passes on the “clock on your wall”, which is distinct from actual “CPU time” - time when the job is actually using CPU cycles. CPU time is necessarily less than walltime, but insofar as the HCP compute nodes are mostly dedicated to HCP jobs when they are running and not splitting time with other jobs, the walltime and CPU time are generally pretty close.

3 CHPC hardware resources are described at http://chpc.wustl.edu/hardware.html

- Structural Preprocessing – one subject/session – Walltime: 24-96 hours, Memory: 12G-24GB
- Functional Preprocessing – one time-series – varies with duration of scanning session
  - rfMRI (~1200 volumes) – Walltime: 36-48 hours; Memory: 20GB-24GB
  - tfMRI (~175 volumes to ~400 volumes) – Walltime: 12-24 hours; Memory: 10GB-12GB
- Diffusion Preprocessing – significantly depends on whether you are running the eddy portion using a GPU
  - Walltime: as high as 36 hours when not using a GPU, closer to 4 hours when using a GPU
  - Memory: 24GB-50GB
- Task fMRI Analysis – Walltime: 4-24 hours; Memory: 12GB
- The "take away" from this is that you will find it difficult to complete runs of the pipelines with relatively low powered machines

## For You To Do[4]

### Step 1 : Become familiar with the Pipeline Scripts Organization

- Visit `/home/hcpcourse/tools/Pipelines` and examine the Pipeline Scripts directory structure

```
Pipelines
|
+- DiffusionPreprocessing
+- Examples
|  +- Scripts
+- fMRISurface
+- fMRIVolume
+- FreeSurfer
+- global
+- PostFreeSurfer
+- PreFreeSurfer
+- TaskfMRIAnalysis
```

- Most of the directories correspond to one of the pipelines (e.g. DiffusionPreprocessing, TaskfMRIAnalysis) or one of the stages of the pipelines (e.g. PreFreeSurfer, fMRISurface, etc)

4 Note that this is the shorter, action-oriented version of the steps for you to carry out. There is another, longer version of these steps not printed here, but in the electronic version at the end titled *For You To Do - Long Form.* The long form contains the same steps, but also includes supplemental information giving more details to provide better understanding.

- The Examples/Scripts directory is going to be your best place to start for learning how to run the various pipeline scripts on your own data.

## Step 2: Become familiar with the Data Organization

- Visit `/home/hcpcourse/data` and examine the data organization for a study
- One subject's unprocessed data have been installed for you in: `/home/hcpcourse/data`

```
/home/hcpcourse/data
|
+- 100307
   +- unprocessed
      +- 3T
         +- Diffusion                              Diffusion Scans
         +- rfMRI_REST<#>_[RL|LR|PA|AP]            Resting State fMRI Scans
         +- tfMRI_<task-name>_[RL|LR|PA|AP]        Task fMRI Scans
         +- T1w_MPR<#>                             T1w Structural Scans
         +- T2w_SPC<#>                             T2w Structural Scans
```

`<#>` = a scan number

`<task-name>` = a task name (e.g. EMOTION, GAMBLING, WM, etc.)

`[RL|LR|PA|AP]` = phase encoding direction

- The scan files in the `unprocessed/3T/T1w*` and `unprocessed/3T/T2w*` directories are the inputs to Structural Preprocessing.

## Step 3: Modifying an Example Batch Script

- We'll illustrate the process of running processing by having you prepare to run the PreFreeSurfer portion of Structural Preprocessing.
- Making a copy of the script
  `Examples/Scripts/PreFreeSurferPipelineBatch.sh`
  for your own modification, use:

```
$ cd ~/tools/Pipelines/Examples/Scripts
$ cp PreFreeSurferPipelineBatch.sh PreFreeSurferPipelineBatch.mine.sh
```

- Edit the batch script. You should be able to use the command `gedit` to fire up a fairly standard text editor.
- Near the top of the script (right after `get_batch_options $@`) are lines that set 3 primary variables that determine the general setup of your run of the pipelines. (The following is what you should see. Next, you will change some of these values.)

```
StudyFolder=”${HOME}/projects/Pipelines_ExampleData
Subjlist=”100307”
EnvironmentScript=
```

*The EnvironmentScript setting is all on one line.*

- Change the `StudyFolder` variable to point to the root of your “study”.

  ```
  StudyFolder=”${HOME}/data”
  ```

- Since our example subject is 100307, you can leave the `Subjlist` variable as is.
- Give the `EnvironmentScript` variable a new value that points it to the environment setup script you will create in the next step.

  ```
  EnvironmentScript= ”${HOME}/tools/Pipelines/Examples/Scripts/
     SetUpHCPPipeline.mine.sh”
  ```

  *This setting of the EnvironmentScript should all be on one line, with no spaces.*

- *Save* your changes and *exit* from the editor.

## Step 4: Modifying a Pipeline Environment Script

- Make a copy of the example environment script (`SetUpHCPPipeline.sh`)

```
$ cd ~/tools/Pipelines/Examples/Scripts
$ cp SetUpHCPPipeline.sh SetUpHCPPipeline.mine.sh
```

- Use an editor to *change or add* the following environment variable settings in your copy of the environment setup script. The existing lines in the original `SetUpHCPPipeline.sh` script for setting up FSLDIR and setting up FREESURFER are commented out (prefaced with a #) and do not include the export keyword. *Remove* the comment markers, *include* the export keyword, and *change* the paths assigned as shown below.

```
export FSLDIR=”/usr/local/fsl”
. ${FSLDIR}/etc/fslconf/fsl.sh

export FREESURFER_HOME=”/usr/local/freesurfer”
. ${FREESURFER_HOME}/SetUpFreeSurfer.sh

export HCPPIPEDIR=${HOME}/tools/Pipelines
echo “HCPPIPEDIR: ${HCPPIPEDIR}”

export CARET7DIR=/usr/local/workbench/bin_rh_linux64
echo “CARET7DIR: ${CARET7DIR}”
```

- *Save* your changes and *exit* from the editor.

## Step 5: Test Running the PreFreeSurfer Pipeline

- Once you have edited both the batch script for running PreFreeSurfer and the environment setup script, you can do a test invocation of the batch script as follows (assuming you've renamed your PreFreeSurfer batch script as: `PreFreeSurferPipelineBatch.mine.sh`).

```
$ ./PreFreeSurferPipelineBatch.mine.sh --runlocal
```

Note that there are 2 (two) hyphens in front of `runlocal`.

You should see a number of logging messages indicating that `PreFreeSurferPipeline.sh` is running and showing you the study folder, subject being processed, input images, templates, field maps, etc.

- Then you should see a log message on the screen that looks something like:

```
 START: ACPCAlignment
Final FOV is:
0.000000 …
```

- This is pretty good confirmation that you've set things up correctly and are successfully running the PreFreeSurfer portion of the Structural Preprocessing Pipeline.
- We will not have time to allow the PreFreeSurfer portion of the pipeline or the other portions to complete. So you should press `Ctrl-C` to cancel this processing.

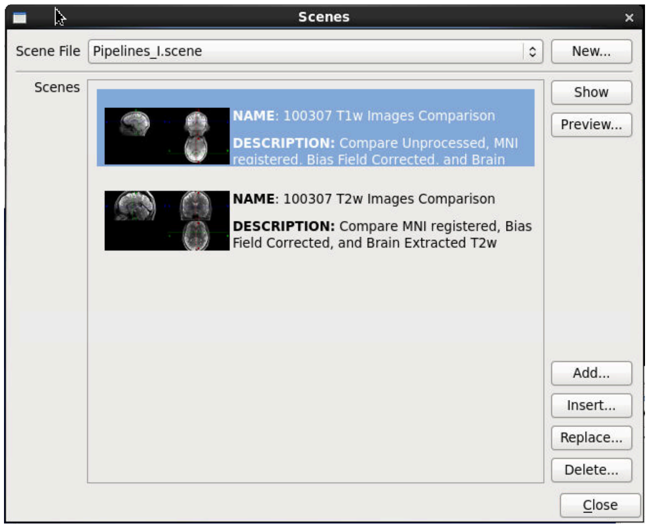
## Step 6: Review already structurally preprocessed data

- In your home directory (`/home/hcpcourse`), there is a subdirectory named `processed_data` (parallel to the subdirectory named `data`). The `processed_data` directory has a set of fully processed data from the same example subject (`100307`).
- Follow the instructions below to compare various versions of the T1w image, T2w image, and surfaces.

### 100307 T1w Images Comparison

```
$ cd ~/day1-monday/pipelines_I
$ wb_view &
```

- On the Workbench splash screen, *double click* on **Pipelines_I.spec**
- *Click* on **Load scenes**. The **Scenes box** will open.



**NAME:** 100307 T1w Images Comparison
**DESCRIPTION:** Compare Unprocessed, MNI registered, Bias Field Corrected, and Brain

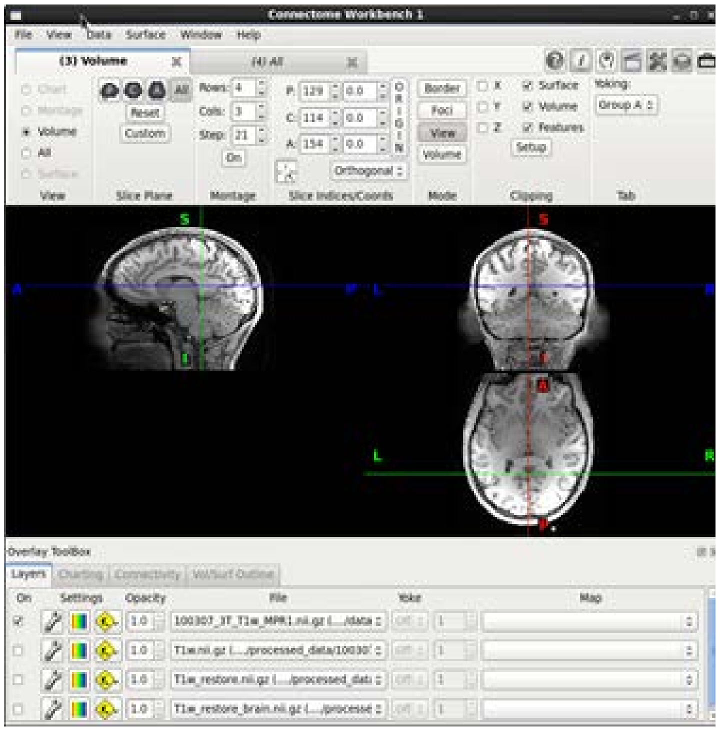
**NAME:** 100307 T2w Images Comparison
**DESCRIPTION:** Compare MNI registered, Bias Field Corrected, and Brain Extracted T2w

- *Double click* on the first scene: **100307 T1w Images Comparison**

You are now viewing an All axes view of the unprocessed T1-weighted MPRAGE NIFTI volume for subject 100307. This volume has *not* been gradient nonlinearity-corrected.

- *Click* the "**On**" checkboxes for layers 2-4 in the Overlay Toolbox (below the Viewing Area).

- *Click* the checkbox beside layer 1 to toggle that layer off.


Connectome Workbench 1
File View Data Surface Window Help
(3) Volume
(4) All
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline
On Settings Opacity File Yoke Map
100307_3T_T1w_MPR1.nii.gz
T1w.nii.gz (.../processed_data/10030
T1w_restore.nii.gz (.../processed_dat
T1w_restore_brain.nii.gz (.../processe

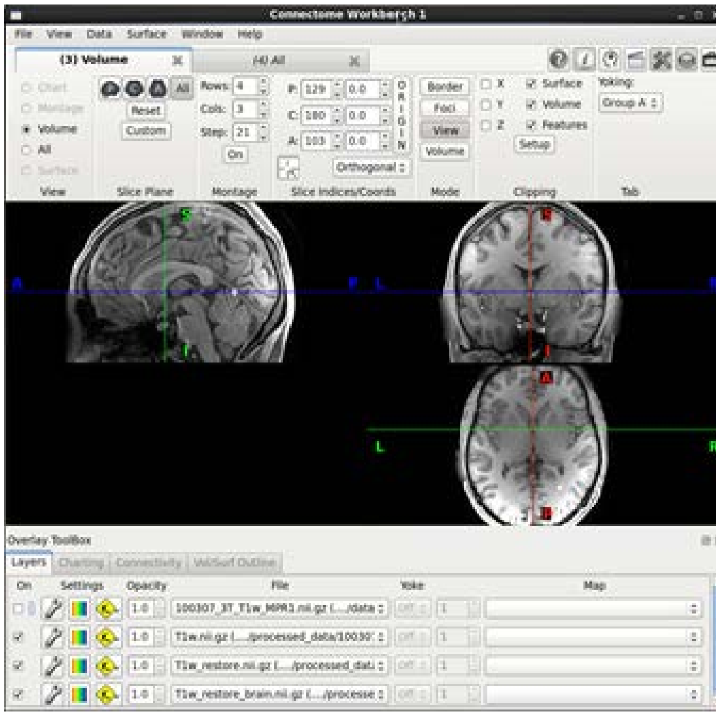
**Unprocessed T1w MPRAGE**

This reveals the data in the 2nd layer: **T1w.nii.gz**, which is the T1w volume after gradient nonlinearity-correction and nonlinear registration into MNI space (path: processed_data/100307/MNINonLinear/ T1w.nii.gz).

- *Toggle* layer 1 *off and on* to compare the unprocessed T1w volume with the MNI-registered volume.

Note the differences in the origin, the rotation, and the scaling between the two volumes.

- *Toggle off* both layers 1 and 2.


Connectome Workbench 1
File View Data Surface Window Help
(3) Volume
(4) All
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline
On Settings Opacity File Yoke Map
100307_3T_T1w_MPR1.nii.gz
T1w.nii.gz (.../processed_data/10030
T1w_restore.nii.gz (.../processed_dat
T1w_restore_brain.nii.gz (.../processe

**MNI-registered T1w**

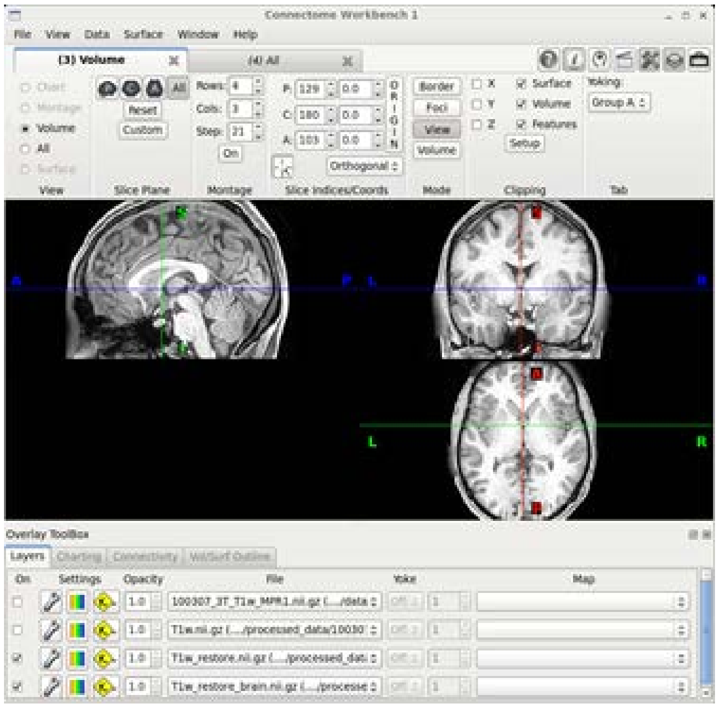
This reveals the data in the 3rd layer: **T1w_restore.nii.gz**, which is the MNI-registered T1w volume after bias field correction has been applied, see top image next page (full path: processed_data/100307/MNINonLinear/T1w_restore.nii.gz).

- *Toggle layer 2 off and on* to compare the uncorrected T1w volume with the cleaner, higher contrast bias-field corrected version.
- *Toggle off* both layers 2 and 3. (layer 1 should also still be off).

This reveals the data in the 4th layer: **T1w_restore_brain.nii.gz**, which is the MNI-registered T1w volume after the bias field correction and brain extraction processing steps

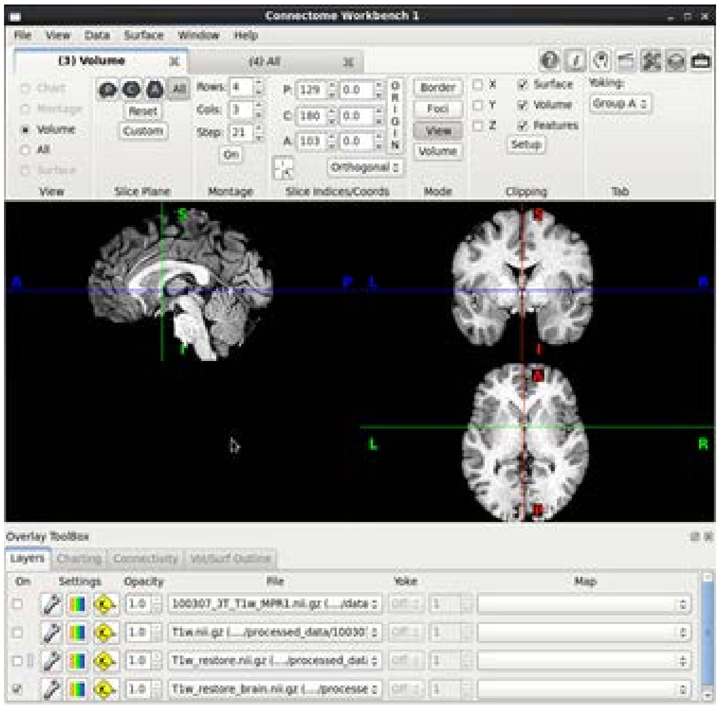
(full path: processed_data/100307/MNINonLinear/T1w_restore_brain.nii.gz).

- *Toggle* layer 3 *off and on* to compare the volumes with and without the skull and other non-brain tissue.



| On | Opacity | File | Yoke |
|---|---|---|---|
| ☐ | 1.0 | 100307_3T_T1w_MPR1.nii.gz (.../data | Off |
| ☐ | 1.0 | T1w.nii.gz (.../processed_data/10030 | Off |
| ☑ | 1.0 | T1w_restore.nii.gz (.../processed_dat | Off |
| ☑ | 1.0 | T1w_restore_brain.nii.gz (.../processe | Off |

**Bias field-corrected T1w**



| On | Opacity | File | Yoke |
|---|---|---|---|
| ☐ | 1.0 | 100307_3T_T1w_MPR1.nii.gz (.../data | Off |
| ☐ | 1.0 | T1w.nii.gz (.../processed_data/10030 | Off |
| ☐ | 1.0 | T1w_restore.nii.gz (.../processed_dat | Off |
| ☑ | 1.0 | T1w_restore_brain.nii.gz (.../processe | Off |

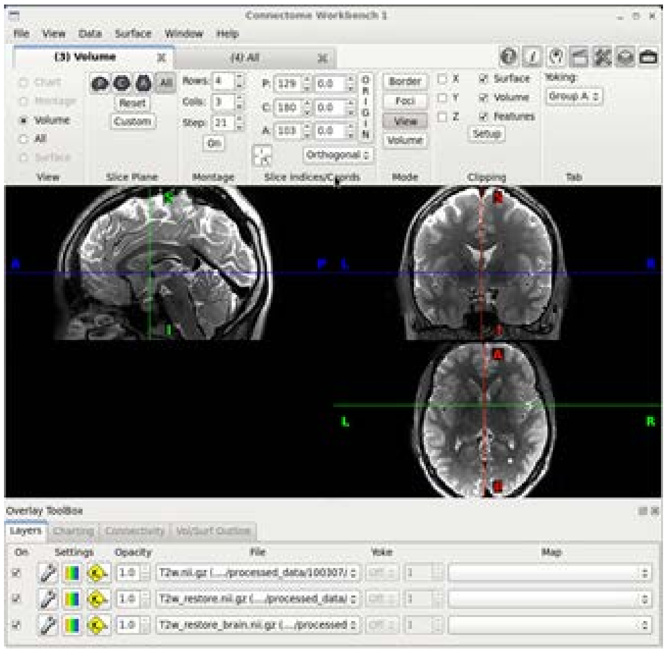
**After Brain extraction T1w**

## 100307 T2w Images Comparison

- If you closed the Scenes box, reopen it with the 🎬 button in the top right corner of the Workbench Window.
- In the scenes box, double click on the second scene: **100307 T2w Images Comparison**

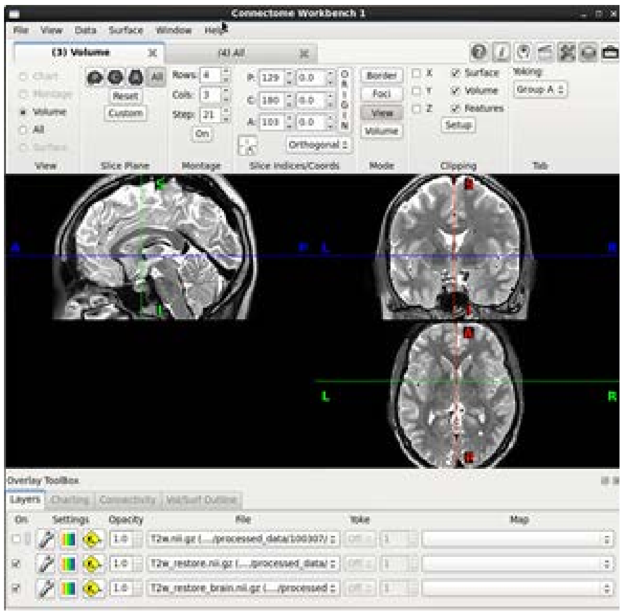
Here, you are viewing an All axes view of the T2-weighted NIFTI volume (T2w.nii.gz) for subject 100307 after nonlinear registration to MNI space.

The scene is set up similarly to the T1w scene above (except without the unprocessed T2w image). T2w.nii.gz is the 1st layer, T2w_restore.nii.gz (registered T2w volume after bias field correction) is the 2nd layer, and T2w_restore_brain.nii.gz (registered T2w volume after bias field correction and brain extraction) is the 3rd layer (full paths: processed_data/100307/MNINonLinear/T2w*.nii.gz). Images of these 3 layers are shown on the following pages.

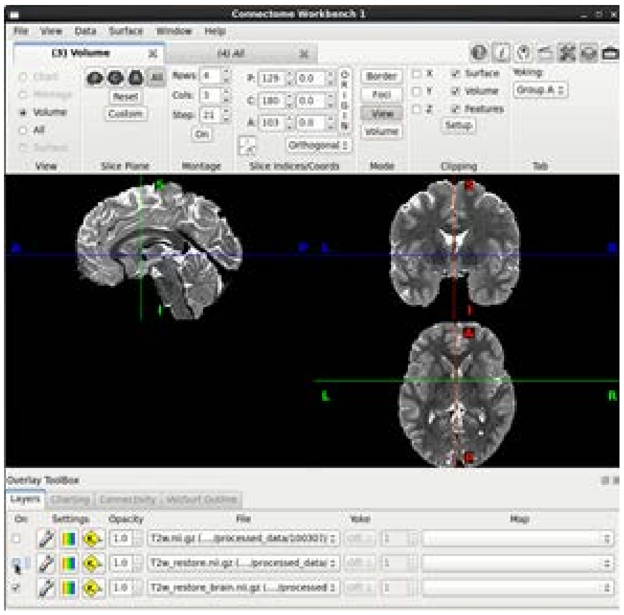


**MNI-registered T2w Volume.**

- *Toggle layer 1 off and on* to compare the uncorrected T2w volume with the cleaner, higher contrast bias-field corrected version.
- *Toggle off* layer 1.
- Toggle layer 2 off and on to compare the T2w volumes with and without the skull and other non-brain tissue.



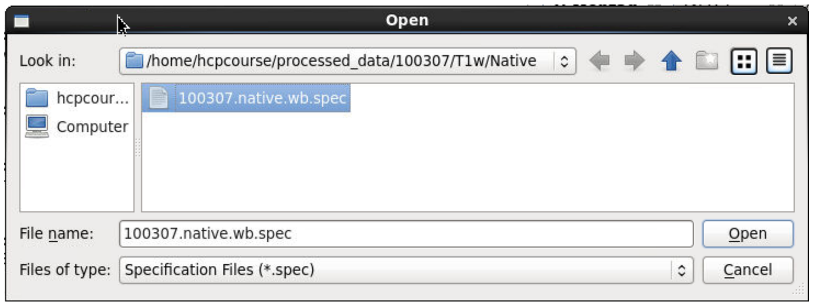
**MNI-registered T2w after Bias Field Correction.**



**MNI-registered T2w after Bias Field Correction and Brain Extraction.**

## Native Space Surfaces

- *Click* **File > Open File…**
- *Navigate to* **/home/hcpcourse/processed_data/100307/T1w/Native/100307.native.wb.spec**
- *Click* **Open**.
- *Click* the **Load** button on the **Open Spec Files** Dialog.



Look in: /home/hcpcourse/processed_data/100307/T1w/Native

100307.native.wb.spec

File name: 100307.native.wb.spec

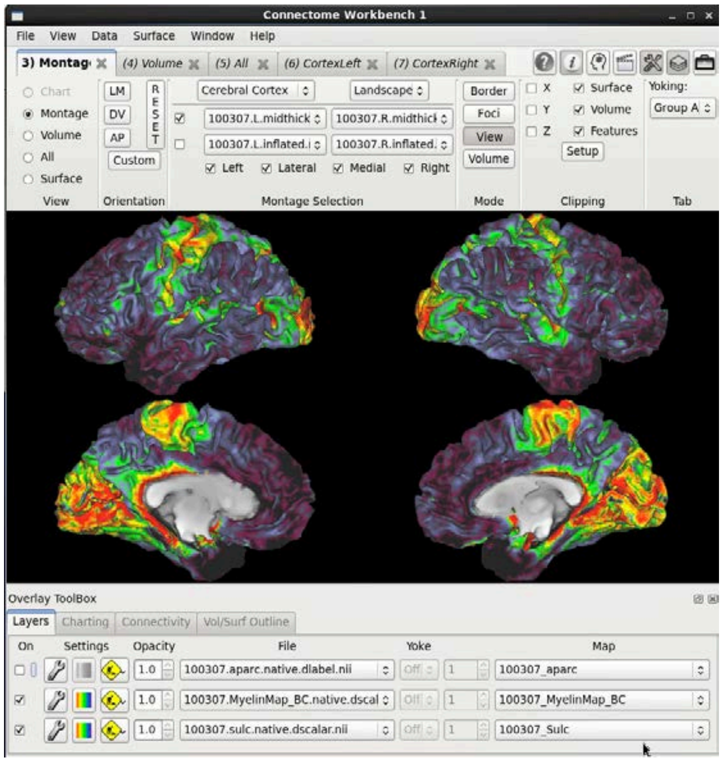
Files of type: Specification Files (*.spec)

The spec file loads many files that are automatically populate the Viewing tabs. The spec files produced by the HCP structural pipeline make it easy for you to quickly load structural data on any HCP subject into Workbench.

The first tab is a montage of left and right hemisphere 100307 midthickness surfaces in Native (subject) space, with various data as layers displayed on the surfaces. The aparc.native "geographic" (gyri and sulci) parcellation generated by FreeSurfer is on top: 100307.aparc.native.dlabel.nii. The native myelin map is on the 2$^{nd}$ layer: 100307.MyelinMap_BC.native.dscalar.nii. The FreeSurfer-generated map of sulcal depth is in the bottom layer: 100307.sulc.native.dscalar.nii. The native myelin map is shown in the image below.

In the **Overlay Toolbox** (under the viewing area) under **File**, the file names are listed, one row per layer. Beside each file name is an up/down arrow.

- *Click the up/down arrow* for any of the layers to explore the other surface data that has been loaded by the spec file.
- *Use* the on/off checkboxes to compare layers.
- *Click* on the **Volume** tab.
- Similarly, *click* on the dropdowns for any of the layers to explore the other voliume data that has been loaded by the spec file.



Montage Selection: ☑ 100307.L.midthick | 100307.R.midthick; ☐ 100307.L.inflated | 100307.R.inflated

| On | Settings | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|---|
| ☐ | | 1.0 | 100307.aparc.native.dlabel.nii | Off | 1 | 100307_aparc |
| ☑ | | 1.0 | 100307.MyelinMap_BC.native.dscal | Off | 1 | 100307_MyelinMap_BC |
| ☑ | | 1.0 | 100307.sulc.native.dscalar.nii | Off | 1 | 100307_Sulc |

- *Click* on the **All** tab and note that this view allows one to explore data on the surface and in the volume together.

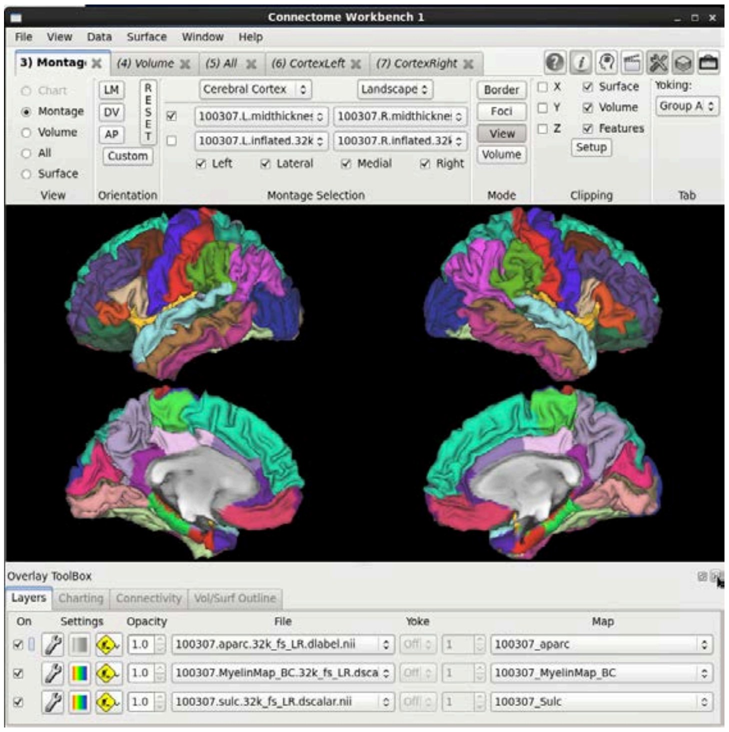
## MNI Space Surfaces

- As above, *click* **File > Open File…**
- *Navigate to* **/home/hcpcourse/processed_data/100307/MNINonLinear/fsaverage_LR32k/100307.32k_fs_LR.wb.spec**
- *Click* **Open**.
- *Click* the **Load** button on the **Open Spec Files** Dialog.

As before, this spec file loads many files that populate the Viewing tabs.

This time, the loaded surfaces and volumes are in MNI-Space, the surface mesh has been resampled to 32,000 vertices (32K mesh), and the surfaces aligned to the fs_LR atlas.

- In the **Overlay Toolbox**, *explore* the **Files** loaded and the **Maps** within each file.
- *Click* on the in the **Volume** tab and *explore* the files available.
- *Click* on the **All** tab and note that both surface and volume files can be displayed as layers in this view.
- *Explore* the **Toolbar** options to change the view in each of the Viewing Tabs.



| On | Settings | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|---|
| ✓ | | 1.0 | 100307.aparc.32k_fs_LR.dlabel.nii | Off | 1 | 100307_aparc |
| ✓ | | 1.0 | 100307.MyelinMap_BC.32k_fs_LR.dsca | Off | 1 | 100307_MyelinMap_BC |
| ✓ | | 1.0 | 100307.sulc.32k_fs_LR.dscalar.nii | Off | 1 | 100307_Sulc |

*End of Practical.*

### Authors:

The authors of this practical were Tim Coalson, Tim Brown, and Jennifer Elam with contributions from Matt Glasser and David Van Essen.

## For You To Do – Long Form

## Step 1 : Become familiar with the Pipeline Scripts Organization

- Visit `/home/hcpcourse/tools/Pipelines` and examine the Pipeline Scripts directory structure

```
Pipelines
|
+- DiffusionPreprocessing
+- Examples
|  +- Scripts
+- fMRISurface
+- fMRIVolume
+- FreeSurfer
+- global
+- PostFreeSurfer
+- PreFreeSurfer
+- TaskfMRIAnalysis
```

- Most of the directories correspond to one of the Pipelines (e.g. DiffusionPreprocessing, TaskfMRIAnalysis) or one of the stages of the pipelines (e.g. PreFreeSurfer, fMRISurface, etc)
- The Examples/Scripts directory is going to be your best place to start for learning how to run the various pipeline scripts on your own data.

## Step 1: Supplemental Information

- *The Pipelines Scripts have already been installed for you on the lab computer systems in the directory*
  `${HOME}/tools/Pipelines (a.k.a. /home/hcpcourse/tools/Pipelines)`
- *The Structural Preprocessing Pipeline is implemented in 3 stages (PreFreeSurfer, FreeSurfer, and PostFreeSurfer). Each of the stages is implemented in a script in the respective subdirectory (`PreFreeSurfer`, `FreeSurfer`, and `PostFreeSurfer`)*
- *The Functional Preprocessing Pipeline is implemented in 2 stages (Volume-based Preprocessing and Surface-based Preprocessing). These stages are implemented in scripts in the `fMRIVolume` and `fMRISurface` subdirectories.*
- *Diffusion Preprocessing and Task fMRI Analysis are implemented in scripts in the `DiffusionPreprocessing` and `TaskfMRIAnalysis` subdirectories respectively.*
- *The ICA+FIX processing is maintained by the FMRIB group at Oxford and is available separately at www.fmrib.ox.ac.uk/~steve/ftp/fix.tar.gz*

## Step 2: Become familiar with the Data Organization

- Visit `/home/hcpcourse/data` and examine the data organization for a study
- One subject's unprocessed data have been installed for you in: `/home/hcpcourse/data`

```
/home/hcpcourse/data
|
+- 100307
   +- unprocessed
      +- 3T
         +- Diffusion                              Diffusion Scans
         +- rfMRI_REST<#>_[RL|LR|PA|AP]            Resting State fMRI Scans
         +- tfMRI_<task-name>_[RL|LR|PA|AP]        Task fMRI Scans
         +- T1w_MPR<#>                             T1w Structural Scans
         +- T2w_SPC<#>                             T2w Structural Scans
```

`<#>` = a scan number
`<task-name>` = a task name (e.g. EMOTION, GAMBLING, WM, etc.)
`[RL|LR|PA|AP]` = phase encoding direction

- The scan files in the `unprocessed/3T/T1w*` and `unprocessed/3T/T2w*` directories are the inputs to Structural Preprocessing

## *Step 2: Supplemental Information*

- *The example batch scripts assume that your data is arranged such that there is a root directory for your study which contains a subdirectory for each subject (or session) in your study. The name of each subject subdirectory is assumed to be the subject id.*
- *This data organization is assumed by the example batch scripts, but not by the actual pipeline scripts that are invoked by the batch scripts. The actual pipeline scripts generally take full path specifications to input and output files or output directories. Therefore, the input parameters to the actual pipeline scripts can be adapted to a different data organization.*
- *For example, if your main study directory is something like `/home/tbb/data/AlzheimersStages`[5] and you have 4 subjects in the study with subject ids: `Subj1243`, `Subj1253`, `Subj1263`, and `Subj1273`, then to process your data using the example batch scripts, your data organization should look like:*

```
/home/tbb/data/AlzheimersStages
/
+- Subj1243
|  +- unprocessed
|     +- 3T
|        +- Diffusion
|        +- rfMRI_REST<#>_[RL|LR|PA|AP]
|        +- tfMRI_<task-name>_[RL|LR|PA|AP]
|        +- T1w_MPR<#>
|        +- T2w_SPC<#>
+- Subj1253
+- Subj1263
+- Subj1273
```

- *If there were 7T data collected for this subject, there would be another directory named `7T` at the same level as the `3T` directory.*
- *Depending upon how your data is collected, you may choose to use other indentifying information instead of the `7T` or `3T` designators. For example, if a longitudinal study*

5 Note that this `AlzheimersStages` study is just an example name for a study. You will not find an `AlzheimersStages` directory on course computers.

*needs to share the same structural scans across the temporal conditions for each subject, but needs to keep other "early" and "late" Task fMRI scans separated, you might choose something as simple as `early` and `late` as subdirectory names under the unprocessed directory.*

## Step 3: Modifying an Example Batch Script

- We'll illustrate the process of running processing by having you prepare to run the PreFreeSurfer portion of Structural Preprocessing.
- Making a copy of the script

  ```
        Examples/Scripts/PreFreeSurferPipelineBatch.sh
  ```

  for your own modification and use

  ```
  $ cd ~/tools/Pipelines/Examples/Scripts
  $ cp PreFreeSurferPipelineBatch.sh ¥
       PreFreeSurferPipelineBatch.mine.sh
  ```

- Edit the batch script. You should be able to use the command `gedit` to fire up a fairly standard text editor.
- Near the top of the script (right after `get_batch_options $@`) are lines that set 3 primary variables that determine the general setup of your run of the pipelines. (The following is what you should see. Next, you will change some of these values.)

  ```
  StudyFolder=" ${HOME}/projects/Pipelines_ExampleData
  Subjlist=" 100307"
  EnvironmentScript=
    " ${HOME}/projects/Pipelines/Examples/Scripts/
     SetUpHCPPipeline.sh"
  ```

  *The EnvironmentScript setting is all on one line.*

- Change the `StudyFolder` variable to point to the root of your "study"

  ```
  StudyFolder=" ${HOME}/data"
  ```

- Since our example subject is 100307, you can leave the `Subjlist` variable as is.
- Give the `EnvironmentScript` variable a new value that points it to the environment setup script you will create in the next step.

  ```
  EnvironmentScript=
    " ${HOME}/tools/Pipelines/Examples/Scripts/
     SetUpHCPPipeline.mine.sh"
  ```

  *This setting of the EnvironmentScript should all be on one line, with no spaces.*

- Save your changes and exit from the editor.

## *Step 3: Supplemental Information*

- *The `StudyFolder` variable establishes the root directory for your study data, like the `/home/hcpcourse/data/AlzheimersStages` directory or the `${HOME}/data` directory.*
- *The `Subjlist` variable is a space delimited list of the subject IDs for the subjects you would like to process in the current batch run. (Based on our discussion above about data organization, it should be clear that each item listed in the `Subjlist` should be the name of a subdirectory found in the `StudyFolder`.)*
- *The `Subjlist` variable is already set to 100307, which is appropriate for this example.*
- *The `EnvironmentScript` variable points to another script file that is used to set environment variables that will be used to determine where to find the other tools that will be used (eg.g the HCP Pipeline Scripts themselves, FreeSurfer, FSL, etc.)*
- *Variables that indicate how your data was collected or how you want it processed*
  - *There are a number of other variables set in the `PreFreeSurferPipelineBatch.sh` script that either provide information about how your data was collected or provide guidance as to how you want the data processed.*
  - *After the section of comments that are prefaced with `###...### INPUTS ###...###`, there is a `for` loop that cycles through each of the listed subjects to process. The variable settings which inform or guide your processing are made in the `for` loop.*
  - *For each variable, there are number of comments that (hopefully) explain what the possible values are and what they mean.*
  - *The following is a sampling of those variables to give you a feel for what can be done by setting the variables.*
    - `AvgrdcSTRING`
      - *The `AvgrdcSTRING` variable is used to determine what type of averaging is done across similar structural scans (T1 and T2) and what type of readout distortion correction is done.*
      - *Avgrdc stands for Averaging and readout distortion correction.*
      - *Note that Readout Distortion is different from Gradient Distortion and these are corrected separately.*

        - *Supported values for* `AvgrdcSTRING` *are:*
            - `NONE` → *average any repeated structural scans (T1w scans averaged together and T2w scans averaged together), but do no readout distortion correction.*
            - `FIELDMAP | SiemensFieldMap` → *these two values are equivalent – average any repeated structural scans and use Siemens specific Gradient Echo Field Maps for readout distortion correction*
            - `TOPUP` *– average any repeated structural scans and use Spin Echo Feield Maps for readout distortion correction*
            - `GeneralElectricFieldMap` → *average any repeated structural scans and use General Electric specific Gradient Echo Field Maps for readout distortion correction*
    - `MagnitudeInputName`
        - *Only applies when using Siemens specific Gradient Echo Field Maps for readout distortion correction*
        - *Set to "point to" the subject specific 4D Gradient Echo Field Map magnitude volume*
    - `PhaseInputName`
        - *Only applies when using Siemens specific Gradient Echo Field Maps for readout distortion correction*
        - *Set to "point to" the subject specific 3D Gradient Echo Field Map phase difference volume*
    - `SpinEchoPhaseEncodePositive` and `SpinEchoPhaseEncodeNegative`
        - *Only applies when using Spin Echo Field Maps for readout distortion correction*
        - *Positive and Negative phase encoded spin echo field maps respectively.*
- *Determining whether your jobs will be queued to a grid engine*
    - *Most of the example batch scripts provide a mechanism for you to specify that you would like the scripts to run "locally". Running "locally" means to simply execute the script command within the terminal/shell from which you invoked the script.*

*Adding the `--runlocal` command line option to your invocation of the batch script will cause the script to be run locally.*

- *Without that command line option specified, an attempt will be made to submit the actual processing jobs to a grid engine using the `fsl_sub` utility that is part of FSL.*
- *You can change the command used to queue jobs to grid enging (using the `queuing_command` variable) and the queue to which jobs are submitted (using the `QUEUE` variable).*

## Step 4: Modifying a Pipeline Environment Script

- Make a copy of the example environment script (`SetUpHCPPipeline.sh`)

```
$ cd ~/tools/Pipelines/Examples/Scripts
$ cp SetUpHCPPipeline.sh SetUpHCPPipeline.mine.sh
```

- Use an editor to change or add the following environment variable settings in your copy of the environment setup script. The existing lines in the original `SetUpHCPPipeline.sh` script for setting up FSLDIR and setting up FREESURFER are commented out (prefaced with a #) and do not include the `export` keyword. Remove the comment markers, include the `export` keyword, and change the values assigned as shown below.

```
export FSLDIR=” /usr/local/fsl”
. ${FSLDIR}/etc/fslconf/fsl.sh

export FREESURFER_HOME=” /usr/local/freesurfer”
. ${FREESURFER_HOME}/SetUpFreeSurfer.sh

export HCPPIPEDIR=${HOME}/tools/Pipelines
echo “HCPPIPEDIR: ${HCPPIPEDIR}”

export CARET7DIR=/usr/local/workbench/bin_rh_linux64
echo “CARET7DIR: ${CARET7DIR}”
```

- Save your changes and exit from the editor.

## Step 4: Supplemental Information

- *The environment setup script primarily consists of settings of environment variables and invocation of tool-specific setup scripts.*
- *The environmental settings in this script will be used to run all your pipelines, not just the PreFreeSurfer pipeline or even just the Structural Preprocessing Pipelines.*
- *Environment Variables that determine the tools to be used*
  - *HCPPIPEDIR*
    - *Should be set to point to where we have installed the HCP Pipeline Scripts that you want to use.*
    - *For this course, it should be set to ${HOME}/tools/Pipelines*
    - *Notice that quite a lot of the other environment variables set in the setup script are set relative to the HCPPIPEDIR variable. This means that for most of the others, you will not have to change the given values.*
  - *CARET7DIR*
    - *Should be set to point to the binary executables directory within the directory where we have installed Workbench (formerly known as CARET.)*
    - *For this course, we have installed Workbench in /user/local/workbench, and we are using a 64-bit version of Linux. Therefore, the variable should be set to /usr/local/workbench/bin_linux64.*
  - *FSLDIR*
    - *Should be set to point to the installation location for FSL.*
    - *The FSL setup script, ${FSLDIR}/etc/fslconf/fsl.sh, must also be invoked to make sure the FSL environment is fully configured.*
    - *For this course, we have installed FSL in /usr/local/fsl*
  - *FREESURFER_HOME*
    - *Should be set to point to the installation location for FreeSurfer.*
    - *The FreeSurfer setup script, ${FREESURFER_HOME}/SetUpFreeSurfer.sh, must also be invoked to make sure FreeSurfer is fully configured.*
  - *Gradient Unwarping*

- *The gradient unwarping Python script (gradient_unwarp.py) is found by use of the PATH environment variable.*
- *For this course, your PATH has already been configured such that gradient_unwarp.py is found appropriately.*

## Step 5: Test Running the PreFreeSurfer Pipeline

- Once you have edited both the batch script for running PreFreeSurfer and the environment setup script, you can do a test invocation of the batch script as follows (assuming you've renamed your PreFreeSurfer batch script as: `PreFreeSurferPipelineBatch.mine.sh`.)

```
$ ./PreFreeSurferPipelineBatch.mine.sh --runlocal
```

**Note that there are 2 (two) hyphens in front of `runlocal`.**

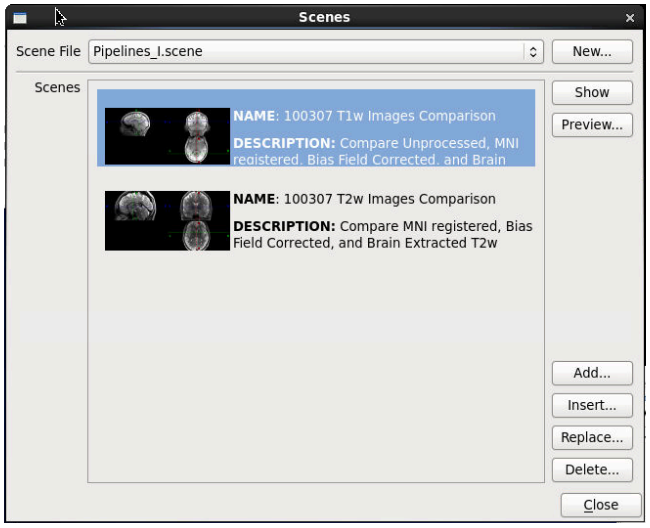
- You should see a number of logging messages indicating that `PreFreeSurferPipeline.sh` is running and showing you the study folder, subject being processed, input images, templates, field maps, etc.
- Then you should see a log message on the screen that looks something like:

```
 START: ACPCAlignment
Final FOV is:
0.000000 …
```

- This is pretty good confirmation that you've set things up correctly and are successfully running the PreFreeSurfer portion of the Structural Preprocessing Pipeline.
- We will not have time to allow the PreFreeSurfer portion of the pipeline or the other portions to complete. So you should press `Ctrl-C` to cancel this processing.

## Step 6: Review already structurally preprocessed data

- In your home directory (`/home/hcpcourse`), there is a subdirectory named `processed_data` (parallel to the subdirectory named `data`). The `processed_data` directory has a set of fully processed data from the same example subject (`100307`).
- Follow the instructions below to compare various versions of the T1w image, T2w image, and surfaces.



**NAME**: 100307 T1w Images Comparison
**DESCRIPTION:** Compare Unprocessed, MNI registered, Bias Field Corrected, and Brain

**NAME**: 100307 T2w Images Comparison
**DESCRIPTION:** Compare MNI registered, Bias Field Corrected, and Brain Extracted T2w

## 100307 T1w Images Comparison

```
$ cd ~/day1-monday/pipelines_I
$ wb_view &
```

- On the Workbench splash screen, *double click* on **Pipelines_I.spec**
- *Click* on **Load scenes**. The **Scenes box** will open.
- *Double click* on the first scene: **100307 T1w Images Comparison**

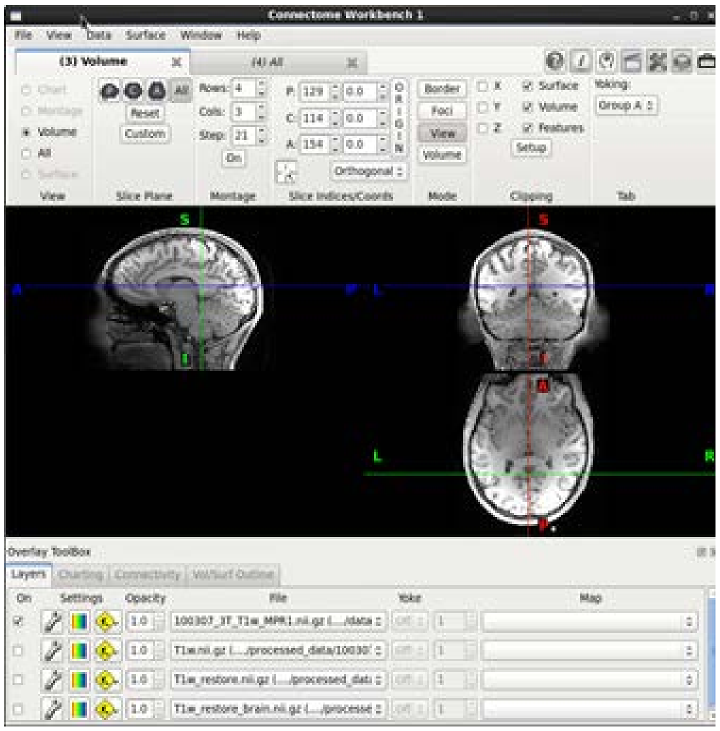
You are now viewing an All axes view of the unprocessed T1-weighted MPRAGE NIFTI volume for subject 100307. This volume has *not* been gradient nonlinearity-corrected.

- *Click* the "**On**" checkboxes for layers 2-4 in the Overlay Toolbox (below the Viewing Area).
- *Click* the checkbox beside layer 1 to toggle that layer off.

This reveals the data in the 2nd layer: **T1w.nii.gz**, which is the T1w volume after gradient nonlinearity-correction and nonlinear registration into MNI space (path: processed_data/100307/MNINonLinear/T1w.nii.gz).

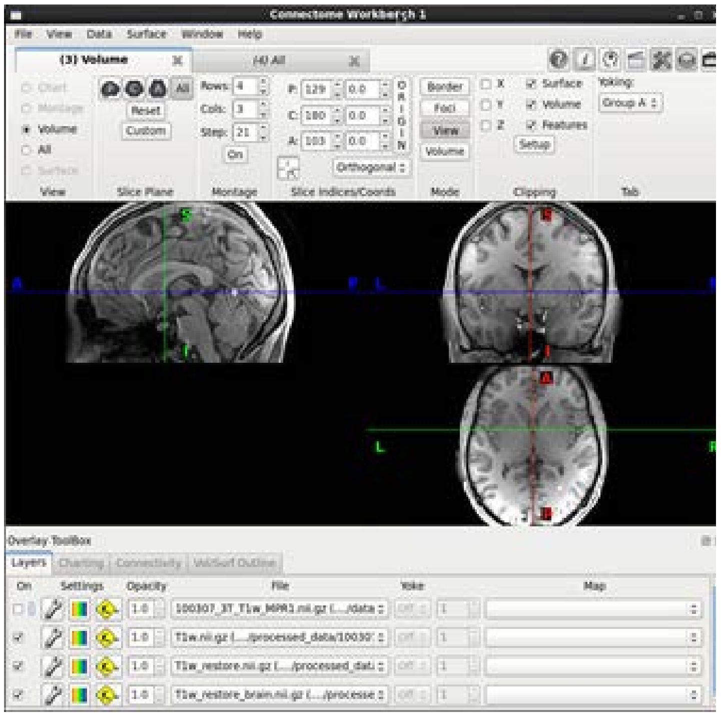
- *Toggle* layer 1 *off and on* to compare the unprocessed T1w volume with the MNI-registered volume.

Note the differences in the origin, the rotation, and the scaling between the two volumes.


Connectome Workbench 1
File View Data Surface Window Help
(3) Volume
(4) All
Chart Montage Volume All Surface
Reset Custom
Rows: 4 Cols: 3 Step: 21 On
P: 129 C: 114 A: 154
Orthogonal
Border Foci View Volume
X Y Z
Surface Volume Features Setup
Yoking: Group A
View Slice Plane Montage Slice Indices/Coords Mode Clipping Tab
S I A P L R
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline

| On | Opacity | File | Yoke |
|---|---|---|---|
| ✓ | 1.0 | 100307_3T_T1w_MPR1.nii.gz (.../data | Off |
|  | 1.0 | T1w.nii.gz (.../processed_data/10030 | Off |
|  | 1.0 | T1w_restore.nii.gz (.../processed_dat | Off |
|  | 1.0 | T1w_restore_brain.nii.gz (.../processe | Off |

**Unprocessed T1w MPRAGE**


Connectome Workbench 1
File View Data Surface Window Help
(3) Volume
(4) All
Chart Montage Volume All Surface
Reset Custom
Rows: 4 Cols: 3 Step: 21 On
P: 129 C: 180 A: 103
Orthogonal
Border Foci View Volume
X Y Z
Surface Volume Features Setup
Yoking: Group A
View Slice Plane Montage Slice Indices/Coords Mode Clipping Tab
S I A P L R
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline

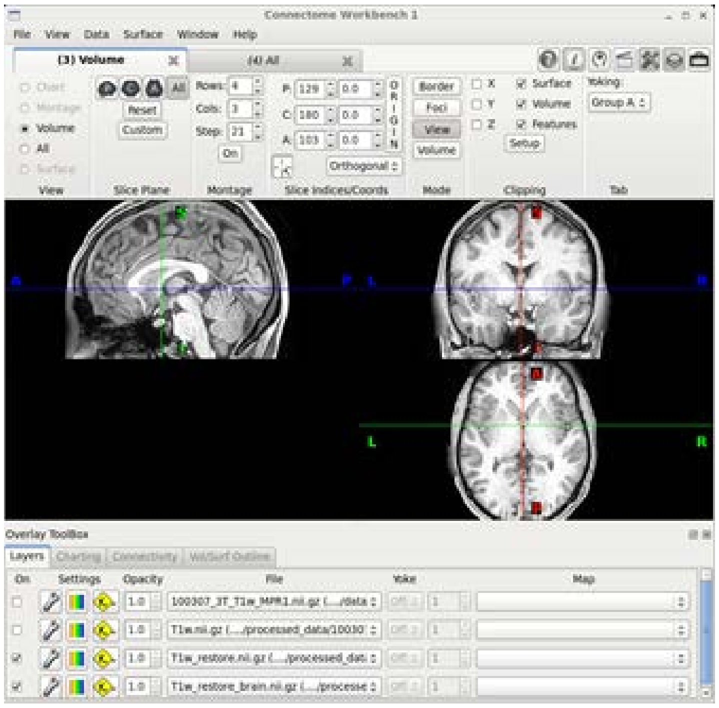
| On | Opacity | File | Yoke |
|---|---|---|---|
|  | 1.0 | 100307_3T_T1w_MPR1.nii.gz (.../data | Off |
| ✓ | 1.0 | T1w.nii.gz (.../processed_data/10030 | Off |
| ✓ | 1.0 | T1w_restore.nii.gz (.../processed_dat | Off |
| ✓ | 1.0 | T1w_restore_brain.nii.gz (.../processe | Off |

**MNI-registered T1w**

- *Toggle off* both layers 1 and 2.

This reveals the data in the 3rd layer: **T1w_restore.nii.gz**, which is the MNI-registered T1w volume after bias field correction has been applied, see top image next page (full path: processed_data/100307/MNINonLinear/T1w_restore.nii.gz).

- *Toggle layer 2 off and on* to compare the uncorrected T1w volume with the cleaner, higher contrast bias-field corrected version.
- *Toggle off* both layers 2 and 3. (layer 1 should also still be off).

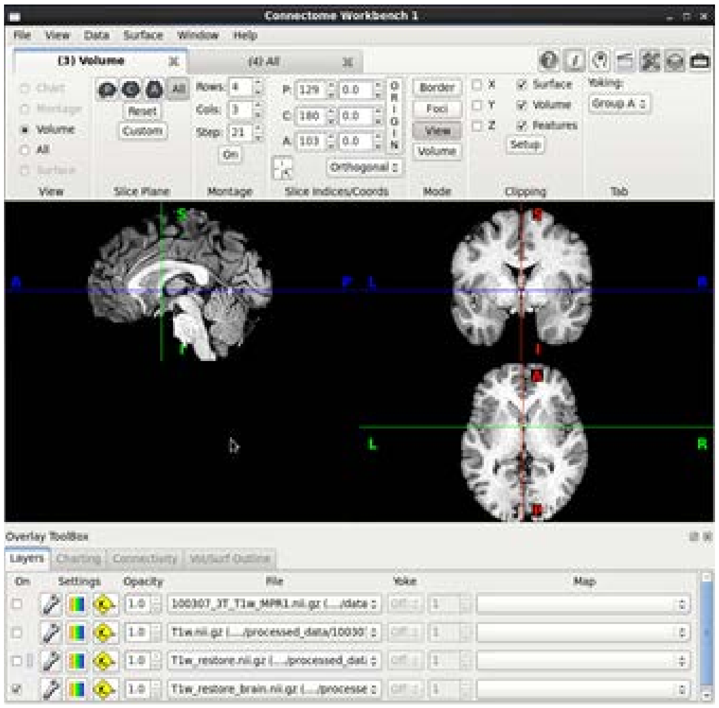


| On | Settings | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|---|
| ☐ | | 1.0 | 100307_3T_T1w_MPR1.nii.gz (..../data | Off | 1 | |
| ☐ | | 1.0 | T1w.nii.gz (..../processed_data/10030 | Off | 1 | |
| ☑ | | 1.0 | T1w_restore.nii.gz (..../processed_dat | Off | 1 | |
| ☑ | | 1.0 | T1w_restore_brain.nii.gz (..../processe | Off | 1 | |

**Bias field-corrected T1w**

This reveals the data in the 4th layer: **T1w_restore_brain.nii.gz**, which is the MNI-registered T1w volume after the bias field correction and brain extraction processing steps (full path: processed_data/100307/MNINonLinear/T1w_restore_brain.nii.gz).

- *Toggle* layer 3 *off and on* to compare the volumes with and without the skull and other non-brain tissue.



| On | Settings | Opacity | File | Yoke | | Map |
|---|---|---|---|---|---|---|
| ☐ | | 1.0 | 100307_3T_T1w_MPR1.nii.gz (..../data | Off | 1 | |
| ☐ | | 1.0 | T1w.nii.gz (..../processed_data/10030 | Off | 1 | |
| ☐ | | 1.0 | T1w_restore.nii.gz (..../processed_dat | Off | 1 | |
| ☑ | | 1.0 | T1w_restore_brain.nii.gz (..../processe | Off | 1 | |

**After Brain extraction T1w**

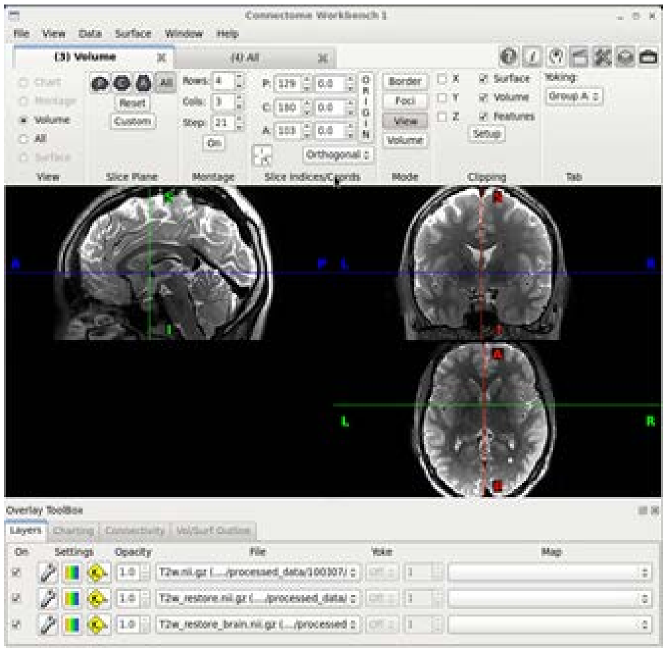
### 100307 T2w Images Comparison

- If you closed the Scenes box, reopen it with the button in the top right corner of the Workbench Window.
- In the scenes box, double click on the second scene: **100307 T2w Images Comparison**

Here, you are viewing an All axes view of the T2-weighted NIFTI volume (T2w.nii.gz) for subject 100307 after nonlinear registration to MNI space.

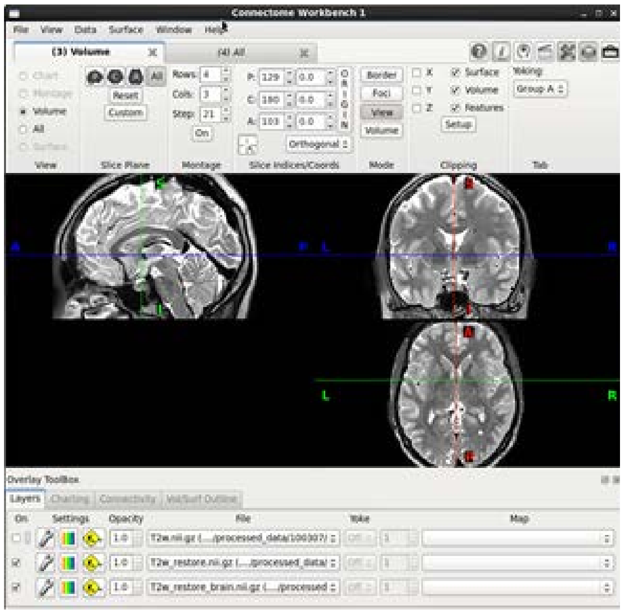
The scene is set up similarly to the T1w scene above (except without the unprocessed T2w image). T2w.nii.gz is the 1st layer, T2w_restore.nii.gz (registered T2w volume after bias field correction) is the 2nd layer, and T2w_restore_brain.nii.gz (registered T2w volume after bias field correction and brain extraction) is the 3rd layer (full paths: processed_data/100307/MNINonLinear/T2w*.nii.gz). Images of these 3 layers are shown on the following pages.

- *Toggle layer 1 off and on* to compare the uncorrected T2w volume with the cleaner, higher contrast bias-field corrected version.
- *Toggle off* layer 1.
- Toggle layer 2 off and on to compare the T2w volumes with and without the skull and other non-brain tissue.



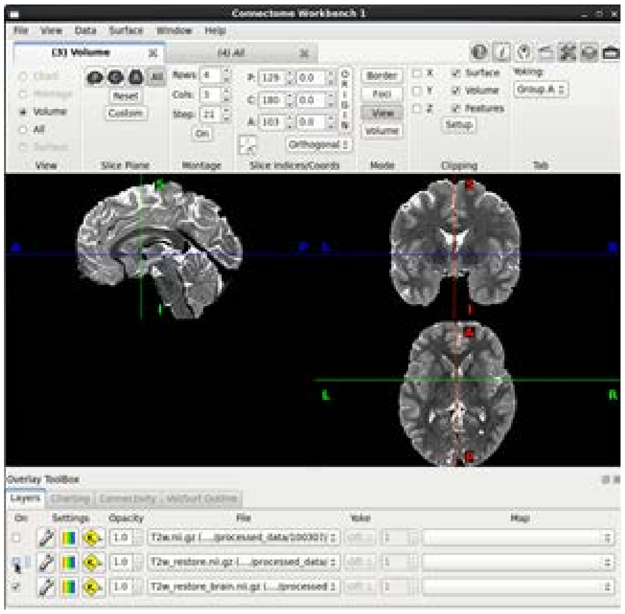
| On | Opacity | File |
|---|---|---|
| ✓ | 1.0 | T2w.nii.gz (.../processed_data/100307/ |
| ✓ | 1.0 | T2w_restore.nii.gz (.../processed_data/ |
| ✓ | 1.0 | T2w_restore_brain.nii.gz (.../processed |

**MNI-registered T2w Volume.**



| On | Opacity | File |
|---|---|---|
|  | 1.0 | T2w.nii.gz (.../processed_data/100307/ |
| ✓ | 1.0 | T2w_restore.nii.gz (.../processed_data/ |
| ✓ | 1.0 | T2w_restore_brain.nii.gz (.../processed |

**MNI-registered T2w after Bias Field Correction.**

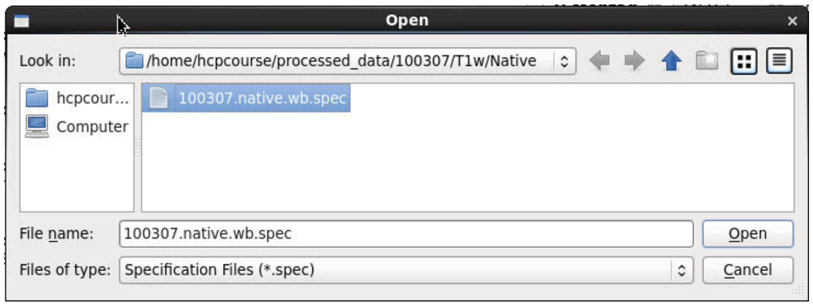


| On | Opacity | File |
|---|---|---|
|  | 1.0 | T2w.nii.gz (.../processed_data/100307/ |
|  | 1.0 | T2w_restore.nii.gz (.../processed_data/ |
| ✓ | 1.0 | T2w_restore_brain.nii.gz (.../processed |

**MNI-registered T2w after Bias Field Correction and Brain Extraction.**

## Native Space Surfaces

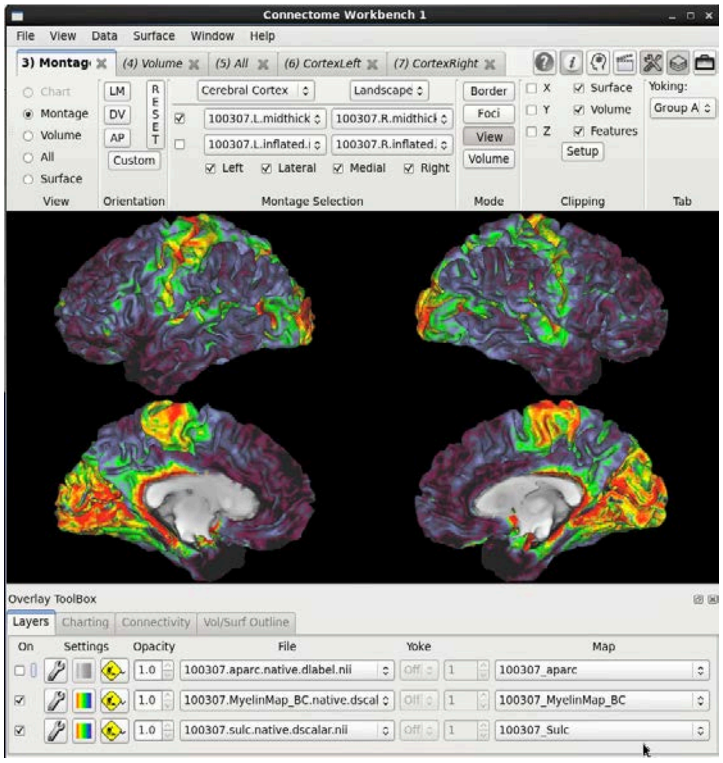
- *Click* **File > Open File…**
- *Navigate to* **/home/hcpcourse/processed_data/100307/T1w/Native/100307.native.wb.spec**
- *Click* **Open**.
- *Click* the **Load** button on the **Open Spec Files** Dialog.



Look in: /home/hcpcourse/processed_data/100307/T1w/Native
File name: 100307.native.wb.spec
Files of type: Specification Files (*.spec)

The spec file loads many files that are automatically populate the Viewing tabs. The spec files produced by the HCP structural pipeline make it easy for you to quickly load structural data on any HCP subject into Workbench.

The first tab is a montage of left and right hemisphere 100307 midthickness surfaces in Native (subject) space, with various data as layers displayed on the surfaces. The aparc.native "geographic" (gyri and sulci) parcellation generated by FreeSurfer is on top: 100307.aparc.native.dlabel.nii. The native myelin map is on the 2nd layer: 100307.MyelinMap_BC.native.dscalar.nii. The FreeSurfer-generated map of sulcal depth is in the bottom layer: 100307.sulc.native.dscalar.nii. The native myelin map is shown in the image below.


Connectome Workbench 1
File View Data Surface Window Help
3) Montag (4) Volume (5) All (6) CortexLeft (7) CortexRight
Chart Montage Volume All Surface
View
LM DV AP Custom
Orientation
RESET
Cerebral Cortex Landscape
100307.L.midthick 100307.R.midthick
100307.L.inflated 100307.R.inflated
Left Lateral Medial Right
Montage Selection
Border Foci View Volume
Mode
X Y Z Surface Volume Features Setup
Clipping
Yoking: Group A
Tab
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline
On Settings Opacity File Yoke Map

| On | Opacity | File | Yoke | Map |
|---|---|---|---|---|
| ☐ | 1.0 | 100307.aparc.native.dlabel.nii | Off | 100307_aparc |
| ☑ | 1.0 | 100307.MyelinMap_BC.native.dscal | Off | 100307_MyelinMap_BC |
| ☑ | 1.0 | 100307.sulc.native.dscalar.nii | Off | 100307_Sulc |

In the **Overlay Toolbox** (under the viewing area) under **File**, the file names are listed, one row per layer. Beside each file name is an up/down arrow.

- *Click the up/down arrow* for any of the layers to explore the other surface data that has been loaded by the spec file.
- *Use* the on/off checkboxes to compare layers.
- *Click* on the **Volume** tab.
- Similarly, *click* on the dropdowns for any of the layers to explore the other voliume data that has been loaded by the spec file.

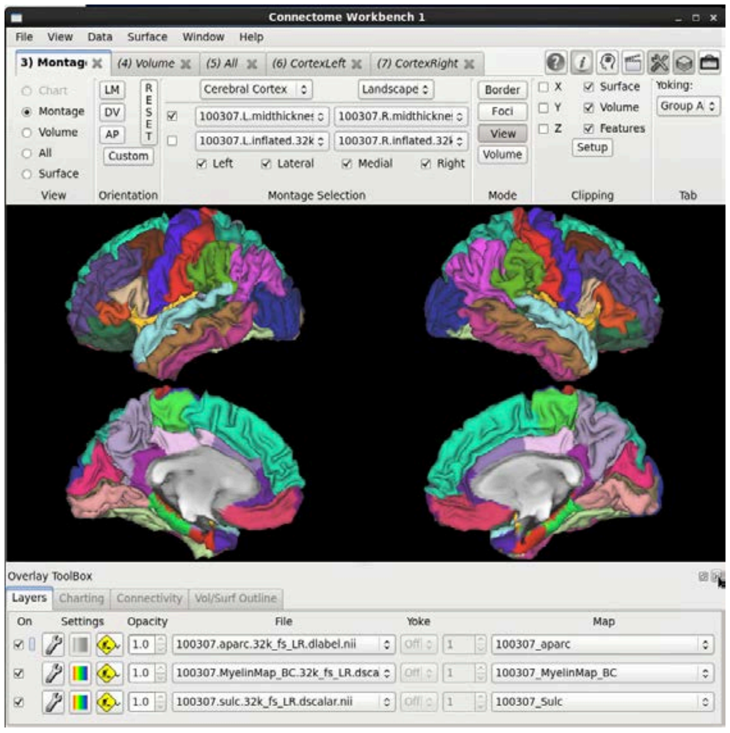
- *Click* on the **All** tab and note that this view allows one to explore data on the surface and in the volume together.

## MNI Space Surfaces

- As above, *click* **File > Open File…**
- *Navigate to* **/home/hcpcourse/processed_data/100307/MNINonLinear/fsaverage_LR32k/100307.32k_fs_LR.wb.spec**
- *Click* **Open**.
- *Click* the **Load** button on the **Open Spec Files** Dialog.

As before, this spec file loads many files that populate the Viewing tabs.

This time, the loaded surfaces and volumes are in MNI-Space, the surface mesh has been resampled to 32,000 vertices (32K mesh), and the surfaces aligned to the fs_LR atlas.


Connectome Workbench 1
File View Data Surface Window Help
3) Montage (4) Volume (5) All (6) CortexLeft (7) CortexRight
Chart Montage Volume All Surface
View
LM DV AP Custom
RESET
Orientation
Cerebral Cortex
Landscape
100307.L.midthickness 100307.R.midthickness
100307.L.inflated.32k 100307.R.inflated.32k
Left Lateral Medial Right
Montage Selection
Border Foci View Volume
Mode
X Y Z
Surface Volume Features
Setup
Clipping
Yoking: Group A
Tab
Overlay ToolBox
Layers Charting Connectivity Vol/Surf Outline

| On | Opacity | File | Yoke | Map |
|---|---|---|---|---|
| ✓ | 1.0 | 100307.aparc.32k_fs_LR.dlabel.nii | Off | 100307_aparc |
| ✓ | 1.0 | 100307.MyelinMap_BC.32k_fs_LR.dsca | Off | 100307_MyelinMap_BC |
| ✓ | 1.0 | 100307.sulc.32k_fs_LR.dscalar.nii | Off | 100307_Sulc |

- In the **Overlay Toolbox**, *explore* the **Files** loaded and the **Maps** within each file.
- *Click* on the in the **Volume** tab and *explore* the files available.
- *Click* on the **All** tab and note that both surface and volume files can be displayed as layers in this view.
- *Explore* the **Toolbar** options to change the view in each of the Viewing Tabs.

*End of Practical.*