Task fMRI Analyses

This tutorial will guide you through key concepts for running and interpreting task fMRI analyses on CIFTI data. In **Part 1,** you will learn how to run scan-level (*aka* lower-level) analyses and subject-level (*aka* mid-level or second-level) analyses using publically available HCP Pipelines. You will also learn how to run group-level (*aka* higher-level) analyses and correlations with customized scripts utilizing wb_command and FSL. In **Part 2,** you will implement Bonferroni and FDR correction for multiple comparisons.

In both parts, we will use the EMOTION task as example data. We focus primarily on running "parcellated" analyses [although an optional section in Part 2 includes exploration of the "dense" (all grayordinates) data for the same task].

Part 1: Key concepts for running task fMRI analyses

What tools are able to run task fMRI analyses on CIFTI data?

Currently, tools required to analyze MRI data in CIFTI format are being actively created and updated. CIFTI-compliant tools for specific analyses that you want to conduct may be available. For an updated list of CIFTI-compliant tools, visit:

https://wiki.humanconnectome.org/display/PublicData/List+of+CIFTI-compliant+tools

But, what if tools for my analyses aren't CIFTI-compliant yet?

One option, which is used by the HCP Pipelines for task fMRI analyses, is to convert the CIFTI-format file into a supported format for other tools. For instance, you can use "wb_command -cifti-separate" to break the CIFTI file into two GIFTI cortical surfaces and one NIFTI subcortical volume or use "wb_command -cifti-convert -to-nifti" to wrap the CIFTI matrix into a "Fake NIFTI" file. At the end of this practical, there is an optional section with more details on these conversions, so that you can understand the procedure and outputs.

In the HCP Pipelines, we utilize FSL tools to run analyses on GIFTI surfaces, NIFTI subcortical volumes, and "fake NIFTI" files described above. An extensive discussion regarding how to set up GLMs for analysis by FSL is beyond the scope of this practical. However, we will walk through some details that are relevant for running task fMRI analyses using the HCP Pipelines. Many of these details (and more!) can be found in the Pipelines documentation:

https://github.com/Washington-University/Pipelines/wiki/v3.4.0-Release-Notes,-Installation,-and-Usage#task-analysis

Setting up files before running "lower-level" and "mid-level" analyses

First, let's go to the directory for this practical session:

```
cd /home/hcpcourse/day4-thursday/practical1-morning
```

FSL tools (i.e., FILM and FLAME) expect design files to be present. For lower-level, this is an .fsf file and a set of explanatory variables (*aka* EVs) for each scan run. Mid-level and group-level analyses can each be captured with an .fsf file.

The necessary .fsf and EV files are already set up in this directory. However, if you ever start processing from "unproc" resources, you will need to copy the EVs into the "preproc" resources once the fMRIVolume and fMRISurface Pipelines have completed. There are two scripts that help do this. Take just a couple minutes to review the main() function near the end of the following scripts to understand how this is done.

- To copy EVs for each participant, review the main() function:

  `gedit Pipelines/Examples/Scripts/copy_evs_into_results.sh &`

- To prepare Level 1 .fsf (in Pipelines/Examples/fsf_templates), review the main() function:

  `gedit Pipelines/Examples/Scripts/generate_level1_fsf.sh &`

- Level 2 (mid-level) .fsf files also need to be put in place before the next step, but there is no script to do this. The following command will make the necessary subdirectories and copy the .fsf file in the new directory:

```
install -D \
Pipelines/Examples/fsf_templates/tfMRI_EMOTION_hp200_s4_level2.fsf \
DATA/100307/MNINonLinear/Results/\
tfMRI_EMOTION/tfMRI_EMOTION_hp200_s4_level2.fsf
```

What does the lower-level model actually look like?

Let's review a lower-level design, so that you can understand what predictors are being fit to the timeseries. HCP used relative paths in the template .fsf files, so they would be easily extensible across subjects. However, the GUIs for Feat and Glm do not handle these relative paths well. Paths are relative to the .feat directory, which will not exist until *after* you run the analysis! To view the design, we will "nest" a copy of the .fsf file one level deeper so that we can view it.

- *Go* to the lower-level data directory:

  `cd DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL`

- *Run* the following:

  `install -D tfMRI_EMOTION_RL_hp200_s4_level1.fsf ViewDesign/design.fsf`

- Let's use FSL's Glm GUI to view the design. We need to be in the ViewDesign directory for this to work correctly. *Run*:

  `cd ViewDesign; Glm &`

- In the GLM Setup window, *click* **Load** and open the **design.fsf** file.
- In the General Linear Model window that appears, *click* **View Design**.

- In the **Model** window that opens, note the main blocked predictors, as well as the corresponding temporal derivatives. The bottom portion of the window contains the weighting used to create each of the contrasts included in the design.
- In the **GLM Setup** window, *click* **Exit** to quit the Glm GUI.

Using the TaskfMRIAnalysisBatch.sh launch script

Let's go back to the practical session directory:

```
cd /home/hcpcourse/day4-thursday/practical1-morning
```

Once the EVs and .fsf files for the Level 1 analyses are in place, you are ready to launch the TaskfMRIAnalysis Pipeline scripts.

> N.B. The Pipeline scripts currently assume two scan runs for each task. If you are customizing these Pipelines to utilize any number of runs other than two, you will need to make edits to the scripts as well as the Level 2 template .fsf files.

The first script, TaskfMRIAnalysisBatch.sh, is essentially a "wrapper" script that loops over the list of participants and tasks that you wish to analyze. It also sets some variables that are submitted as options to the subsequent scripts. Typically, this is the only thing that you will need to edit in order to start your analysis.

Let's edit TaskfMRIAnalysisBatch.sh to run a parcellated analysis for the EMOTION task for one participant (100307)!

```
cp Pipelines/Examples/Scripts/TaskfMRIAnalysisBatch.sh \
./TaskfMRIAnalysisBatch_MINE.sh

gedit TaskfMRIAnalysisBatch_MINE.sh
```

- *Search* in the script, at the section starting with "StudyFolder=", for the variables *italicized* here:
  - *StudyFolder* sets the location of the HCP data. *Change it* to the full path of the DATA folder in this directory: **/home/hcpcourse/day4-thursday/practical1-morning/DATA**
  - *SubjList* identifies the participants to run through task analysis. For time considerations, we will run just a single subject. Set it to **100307**.
  - *EnvironmentScript* will export global variables used throughout the HCP Pipeline scripts. *Set* to: **/home/hcpcourse/day4-thursday/practical1-morning/SetUpHCPPipeline.sh** (N.B. A version of SetUpHCPPipeline.sh already exists at that location that has been set up correctly for setting the environment for the Practical computers).
- *Search* in the script, at the section starting with TaskNameList="" for the variables italicized here:
  - *TaskNameList* can be modified to run a shorter (or different) list of tasks; Comment out all tasks other than EMOTION (by adding '#' at front of line). (Do not comment out TaskNameList="")

- *SmoothingList* can be modified if you want to perform additional spatial smoothing level(s) on the timeseries data. For a parcellated analysis, smoothing is not performed.
- *TemporalFilter* can be modified to change the high pass filter cutoff applied to the timeseries data; 200 (seconds) is an appropriate setting for this blocked design.
- *VolumeBasedProcessing* can be set to YES if you would also like analyses conducted on the whole-brain volumes registered to MNI space; Leave as "NO" for parcellated analyses.
- *ParcellationList* and *ParcellationFileList* can be set to conduct parcellated analyses on grayordinate parcellations. For this demonstration, *set* **ParcellationList** to "Gordon" and *set* the **ParcellationFileList** to the full path to the following dlabel.nii file:

  /home/hcpcourse/day4-thursday/practical1-morning/Gordon_333_Parcel+FS_Anatomical_Parcel.dlabel.nii

  Later, we will see what this parcellation looks like.

- To understand what happens next, find the section containing **About to run**
  - *queuing_command* can be set to utilize some scheduler, such as SGE via fsl_sub. It is controlled by the --runlocal option when launching the script.
  - Ultimately, the script calls *TaskfMRIAnalysis.sh* with the appropriate options set or determined above for each subject and for each task.

Save your file! Now that your variables have been edited, let's launch the script. Execute it with the following command:

```
bash TaskfMRIAnalysisBatch_MINE.sh --runlocal | tee MINE.log
```

If you set up the script variables correctly, output should now be appearing in your terminal. (The "tee MINE.log" portion at the end of the command allows the output to scroll to the terminal, while also saving a copy to a file called MINE.log). If not, try to debug why it didn't run from the output messages.

e.g., If you set StudyFolder to "DAT" instead of "DATA", you would get the following message:

```
ERROR: failed to open file '/home/hcpcourse/day4-thursday/practical1-morning/DAT/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/tfMRI_EMOTION_RL_Atlas.dtseries.nii'
```

It should only take about 2 minutes to complete this Level 1/2 parcellated analysis on a single subject.

While **TaskfMRIAnalysisBatch.sh** is running, let's learn a bit about the remaining scripts in the Task fMRI Analysis Pipeline. At the end of this practical, there is an optional section where you can review some of these scripts

TaskfMRIAnalysis.sh: Determines which version of FSL is installed and calls the compatible version of the pipeline scripts. On these machines, FSL 5.0.8 is installed, so the v2.0 scripts will run.

TaskfMRIAnalysis.v2.0.sh: Defines necessary inputs based on subject ID and other options provided. For each of the scan runs within the given task, it launches the **TaskfMRILevel1.v2.0.sh** script to run the Level 1 (lower-level) analyses. After both lower-level analyses complete, it launches the **TaskfMRILevel2.v2.0.sh** script – a Level 2 (mid-level) fixed-effects analysis to combine the activity estimates from each lower-level scan run into subject-level activity estimates.

Before proceeding to group-level analyses, let's make sure that the lower- and mid-level analyses you started earlier have finished. First, we'll do two quick checks.

- First, back in the terminal window, *review* the end of **MINE.log** – the last line should have a time stamp, followed by the phrase "TaskfMRIAnalysis.sh - Completed".
- Second, *check* that the Level 2 parcellated zstat file exists. It gets created in the subject folder tree within the StudyFolder that you set earlier. Specifically:

```
cd ~/day4-thursday/practical1-morning/DATA/100307/MNINonLinear/Results
cd tfMRI_EMOTION/tfMRI_EMOTION_hp200_s2_level2_Gordon.feat
```

Inside that directory are a number of files, including individual files for each contrast of the EMOTION task. Conveniently, the script also creates a single "summary" file containing the results for all the contrasts. That file is **100307_tfMRI_EMOTION_level2_hp200_s2_Gordon.pscalar.nii**

(Note that the file name lacks a contrast name string such as "FACES-SHAPES", which is how you can know that it is the "summary" file containing the results of all contrasts).

The EMOTION task contains 6 contrasts, so that summary pscalar.nii file should contain 6 different "maps". *Verify that* using the following command:

```
wb_command -file-information \
100307_tfMRI_EMOTION_level2_hp200_s2_Gordon.pscalar.nii \
-only-number-of-maps
```

You could take a look at this pscalar.nii now using Workbench (wb_view), but we are going revisit and take a look at this group-level parcellated zstat file in Part 2 of the practical, so let's continue onward.

After Level 2 analyses are completed for each participant, we can run Group-level analyses (Level 3) using another customized script. For convenience, we have already pre-computed the same Level 1 and 2 parcellated analysis that we just did for subject 100307 on 37 more subjects.

The first step is to create the .fsf file for a group-level analysis using FSL's Glm Gui. Let's go back to the practicals directory:

```
cd ~/day4-thursday/practical1-morning
```

Launch:

```
Glm &
```

- In the **GLM Setup** window, *change* the pull down menu to **Higher-level, Non-timeseries design**
- *Move* the **General Linear Model** window that pops up over to the side.
- In the already existing **GLM Setup** window, enter **38** into the **# inputs** field (and *hit enter*).
- *Click* the **Wizard** button, *select* **single group average** and *click* **Process**.

- You can look at the EVs and Contrasts if you'd like, but they're very simple (just a column of 1's to model a group mean).
- Back in the **GLM Setup** window, *click* **Save**. Save the file in the fsf directory as **group_mean_U38.fsf**
- *Exit* the Glm GUI

For additional information and examples about the types of GLMs that can be set up, feel free later to visit the following pages on the FSL wiki:

- For more detail about Level 1: http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FEAT/UserGuide
- For more detail about higher-level: http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/GLM

The next step to running the Group Level analyses is to set up a copy of the TaskfMRILevel3Batch.sh script with the options necessary for your analysis. Ultimately, the TaskfMRILevel3Batch.sh script is a wrapper script for the TaskfMRILevel3.sh script. Similar to the Level 2 script, it uses a fake NIFTI approach to set up the analysis using FSL's flameo. We've set everything for you, so you don't need to change anything at this time.

Let's try to run the script as is, so that it can run while you're working.

```
bash TaskfMRILevel3Batch.sh | tee MINE_Level3.log
```

That will only take about 4 minutes to complete.

While that is running, you can review this brief description of the most common settings that you would need to change in other instances:

- *EnvironmentScript*: the path to the environment script to get global variables for running HCP Pipelines. This is preset correctly for you.
- *HCPPIPEDIR*: the path to the Level 3 scripts directory. This is preset correctly for you.
- *StudyFolder*: The path to your CinaB, or wherever you have stored the input data for your analysis. This is preset correctly for you.
- *SubjList*: needs to be a space-delimited list of subject IDs. Typically, this can be read in from a file, but your file needs to have linux end-of-line characters to work correctly.
- *GroupAverageName*: This will become the name of the directory where your outputs are saved. We are calling this analysis "U38".
- *DesignFolder* and *LevelThreeFsf*: together specify the location of your group-level fsf design file.
- *LevelTwoTaskList*: the base name of your Level 2 task directory
- *SmoothingList*: a space-delimited list of spatial smoothing parameters from your Level 2 analyses
- *TemporalFilter*: the temporal filter cutoff parameter from your Level 2 analyses
- *ParcellationList*: the value of ParcellationList set in your Level 2 analyses

You probably want to view the results from this Level 3 parcellated analysis. But hold off for a moment, since we'll already be viewing the results shortly as part of thresholding for multiple comparisons.

Part 2: Bonferroni for multiple comparison correction

We are going to implement Bonferroni and FDR correction for multiple comparisons using the Level3 parcellated analysis generated above. A very similar sort of correction could be done on a dense analysis across all grayordinates (and that is included as an optional part later in this Practical), but a parcellated analysis has drastically fewer comparisons to correct for.

For a Bonferroni correction, we treat each test as if it was independent of the others, which means to control the overall Type 1 error rate, we want

$p_{Bonferroni\text{-}threshold} < 0.05/(\#\ tests)$

How many parcels (ROIs) were used in the parcellated analysis?

Use wb_command -file-information on a parcellated scalar (pscalar.nii) file from the Level 3 analysis to find out:

```
cd ~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear/Results

cd tfMRI_EMOTION/tfMRI_EMOTION_hp200_s2_level3_Gordon.gfeat

wb_command -file-information \
tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii | less
```

Sidebar on Linux less tool: The operation "| less" at the end of the command "pipes" the output of wb_command to the Linux less tool, which gives you the ability to navigate through the output within the terminal. 'Space-bar' or 'f' will advance the output by one page; 'b' will go back by one page; 'Enter/return' will advance the output by one line. 'G' (shift + g) will advance you to the end of the file, and lower case 'g' will return you to the beginning of the file. 'q' will exit from the less tool. The information we will be looking for initially is on the first page.

- *Look* for "Number of Rows" or "CIFTI Dim[1]" to identify number of parcels.

BTW: Do you know what "Number of Columns" and "CIFTI Dim[0]" represents in this pscalar.nii?

- While we're at it, *identify* which Contrast ("Map") is FACES-SHAPES, by looking at the end of the -file-information output. (Or, re-run the command, adding the -only-map-names optional argument to limit the output to just the map names).

So, we want $p_{Bonferroni\text{-}threshold} < 0.05/(\#\ tests) = 0.05/352 = 1.42e\text{-}4$

What would be the equivalent z-statistic value for a one-sided test?

- For that, launch Octave in a new terminal:

```
$ octave
```

- The following will convert the p-value threshold to the z-stat equivalent (for a one-sided test):

```
zthr_bonferroni = -norminv(0.05/352)
```

This yields 3.63.

- Let's stay with Octave for the time being, and *load* the actual pscalar data using a function 'ciftiopen.m' that we've placed into the ~/matlab directory.

```
addpath('~/matlab')
cd ~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear/Results
cd tfMRI_EMOTION/tfMRI_EMOTION_hp200_s2_level3_Gordon.gfeat
file = 'tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii';
C = ciftiopen(file,'wb_command');
```

The actual data values will initially be in the C.cdata field. Let's pull them into a new variable called just 'cdata', and check the size (i.e., dimensions):

```
cdata = C.cdata;
size(cdata)
```

Do you understand why the dimensions of this data are 352 x 6?

We identified the index of the FACES-SHAPES contrast above. Which column of data in Octave represents this contrast?

- Next, let's generate a histogram of the parcel z-stats for FACES-SHAPES contrast, and find how many parcels from that contrast have a z-stat that exceeds the Bonferroni adjusted threshold:

```
hist(cdata(:,3),100)
%Use 100 bins in 'hist' command, because that generates a nice
%match to the Workbench histogram display (to follow shortly)
I = find(cdata(:,3) > zthr_bonferroni);
length(I)
```

So, 55 parcels from the FACES-SHAPES contrast survive Bonferroni correction.

To see which parcels those were, let's look at these parcellated results in Workbench (wb_view). Using what you learned about using Workbench from the previous days:

- In a new terminal window, enter:
  `wb_view ~/data/HCP_Q1-Q6_GroupAvg_Related440_Unrelated100_v1/\`
  `HCP_Q1-Q6_R440_U100_DATA.32k_fs_LR.wb.spec &`
- *Press* **Load**. This "spec" file will conveniently give you the necessary surface models and volume templates for Workbench.
- *Load* the pscalar.nii from the parcellated Level3 analysis:

Under the **File:Open File:Files of type: "Any File (*)"** or **Connectivity - Parcel Scalar Files (*.pscalar.nii)**

Then, navigate to the following file, and open it. This will add it to the set of files loaded in Workbench.

~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear/Results/tfMRI_EMOTION/tfMRI_EMOTION_hp200_s2_level3_Gordon.gfeat/tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii

- In the **Montage** tab, *set* the **pscalar.nii** that you just loaded into Workbench to display as the top layer. *Set* the **Yoke** group to **I**. *Toggle off* other layers that you don't want displayed at the same time.

- In the **Volume** tab, similarly *set* the **tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii** as the top layer and *set* the **Yoke** group to **I**. *Toggle off* other layers that you don't want displayed at the same time.

- *Explore* the Level 3 parcellated results for the various contrasts a little since this is the first time you are viewing them in Workbench.

- Display the FACES-SHAPES contrast in both the Montage and Volume tabs by selecting the 3rd "map" within that pscalar. (Under **Maps** *click* the dropdown and *select* FACES-SHAPES).

- *Click* on the wrench icon next to the first layer.

- In the **Overlay and Map Settings** box, *set* the **high Threshold** to 3.63 and *click* **Show Data Outside Thresholds**. This will only display the parcels having z > 3.63.

Those are then the parcels that have Bonferroni corrected significant group activation in this "U38" group of subjects. (Because we used a Bonferroni correction corresponding to a one-sided analysis, be sure to display only the "Positive" results – i.e., uncheck the "Negative" box).

Note if you did not set the **tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii** file to the same Yoke group in both the Montage and Volume tabs, you would need to set the threshold in both tabs separately.

- In **Overlay and Map Settings**, *set* **Data Normalization** to **Selected Map in File** and then *compare* the histogram for the FACES-SHAPES contrast in Workbench to the one you generated above in Octave – they should match! ***N.B****. If Data Normalization is set to "All Maps in File" (the default setting), then the histogram represents the composite across all the contrasts/maps and it will not match the Octave-generated histogram.*

Find the Left Amygdala anatomically in the **Volume** tab of Workbench (approx. Coronal slice 170 at -7 mm). *Click* on it to get its z-stat value in the "Information" window.

What if you want to know which numerical row index contains the Amydala_L data for comparison to the data loaded in Octave? The following combination of wb_command with grep (issued from the terminal window; not inside Octave) will give you the numerical index for the Amydala_L parcel (type in the linux / bash terminal, not the octave terminal):

```
wb_command -nifti-information \
tfMRI_EMOTION_level3_cope1_hp200_s2.pscalar.nii -print-xml | \
grep "Parcel Name" | grep -n "Amygdala_L"
```

Then, in the Octave window:

```
cdata(334,3)  %row indexes parcel; column indexes contrast/map
```

That value should match what you found using Workbench (i.e., 8.0650)

Thus ends our Bonferroni correction section!

FDR for multiple comparison correction

Next: What if we use FDR (false-discovery-rate) correction instead? FDR controls for the expected number of false-positive results, rather than the "family-wise error" (as Bonferroni does), and thus is a little less "conservative".

First, we need to convert the parcel z-stats for FACES-SHAPES into (one-sided) p-values. Then using the FDR.m function (which we've also installed into ~/matlab), we'll compute the p-value threshold, and associated z-stat threshold that would yield an expected false-positive rate of 5%.

In Octave:

```
pvals = 1 - normcdf(cdata(:,3));

pthr = FDR(pvals,0.05);

zthr_fdr = -norminv(pthr)   %See A1 below for result
```

How does this compare to the Bonferroni correction value

At this point, you can repeat what we did above for Bonferroni, but using this new FDR-based threshold.

```
I = find(cdata(:,3) > zthr_fdr);

length(I)
```

1) How many parcels survive FDR correction? (See A2 below for result)
2) Threshold the FACES-SHAPES contrast in Workbench at the FDR threshold, to see visually what parcels survive.

Do you know what you would do differently if we wanted to do FDR on the negative tail of the FACES-SHAPES results instead? [Hint: The negative of FACES-SHAPES is the SHAPES-FACES contrast]. Time (and/or interest) permitting: How many parcels have significantly greater activation to SHAPES rather than FACES? [See A3 below for result -- a lot fewer than FACES vs. SHAPES].

A1: 2.11; A2: 121; A3: 36

Optional/ Time Permitting

A) Dense vs. parcellated analysis

We have pre-computed the "dense" (i.e., per grayordinate) group ("Level 3") maps that one would get using the same 38 subjects for the contrasts of the EMOTION task, using 4 mm of smoothing.

Those are available at:

cd ~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear/Results/tfMRI_EMOTION

cd tfMRI_EMOTION_hp200_s4_level3.gfeat

Using the Workbench window that you already have open:

1) Under the **File:Open File:Files of type** *change* to **Any File (*)** or **Connectivity - Dense Scalar Files (*.dscalar.nii)**. Then, *navigate* to the directory listed above and load **tfMRI_EMOTION_level3_zstat1_hp200_s4.dscalar.nii** into Workbench.

2) *Select* the dscalar that you just loaded as one of the layers and explore the dense maps for EMOTION a little bit. Compare to the parcellated results by *toggling* the displayed Layers on and off in Workbench. Note: To do this "fairly" you'll want to harmonize the color Palette of the dense and parcellated maps:
   a) For the dense map, first *set* a **Threshold** in **Overlay and Map Settings** (e.g., 3.63)
   b) *Change* to a **Fixed** mapping with **Pos Max** = 8 and **Neg Max** = -8
   c) *Click* **Apply to Files…** button and *select* both the dense (**tfMRI_EMOTION_level3_zstat1_hp200_s4.dscalar.nii**) and parcellated (**tfMRI_EMOTION_level3_zstat1_hp200_s2.pscalar.nii**) maps so that these Palette settings will be applied in common to both sets of maps.
   d) As a sanity check, *display* the color bar (rainbow box in main Workbench window) for both maps, and confirm that they now have the same color scale!

3) Also load the parcellation itself. In the **File:Open File:Files of type** *change* to **Dense Label Files (*dlabel.nii)**, then *navigate* to and *load*

   ~/day4-thursday/practical1-morning/Gordon_333_Parcel+FS_Anatomical_Parcel.dlabel.nii

   Let's see if the borders of the task activation in the dense map correspond reasonably to the parcel edges. To facilitate this comparison:

   a) *Click* on the wrench icon for the **Gordon_333_Parcel+FS_Anatomical_Parcel.dlabel.nii** layer.
   b) In **Overlay and Map Settings**, go to **Labels** tab for the dense label file.
   c) Set **Drawing Type** = **Outline Color**
   d) The parcel edges will now be outlined, making it easy to compare the parcel locations to the location of activation in the dense task map.

How to you feel this particular parcellation scheme does at capturing the dense activation pattern of the FACES-SHAPES contrast?

At this point, if you wished, you could do the same sort of Bonferroni/FDR calculations and corrections that we did above. The only conceptual difference is that the # of tests is now 91282 (the number of grayordinates) rather 352 (the number of parcels).

If you did that, you would find that 2814 grayordinates (out of 91282) survive Bonferroni correction (zthr_bonferroni = 4.87) and 15223 grayordinates survive FDR correction (zthr_fdr = 2.39) for the FACES-SHAPES contrast. (In both cases, the regions include DLPFC, visual cortex, fusiform gyrus, and amygdala).

However, more interesting might be the following question:

How do the z-stats of the parcellated analysis (in which the Level1/2 analyses themselves started with the average time course within each parcel) compare to averaging the z-stats within parcels of the dense maps? Do you understand the key difference?

> If all the grayordinates in a parcel respond the same way to a task, then computing an average time course reduces noise (increases SNR), without losing any spatial specificity. The reduced noise increases z-stats. Conceptually, parcellation is just a form of spatial smoothing, except the smoothing is implemented by averaging all the grayordinates within the parcel!

To get the average z-stats within parcels of the dense maps, we use wb_command -cifti-parcellate. *Check the syntax* of -cifti-parcellate yourself first as practice. See if you can figure out what the desired command would be.

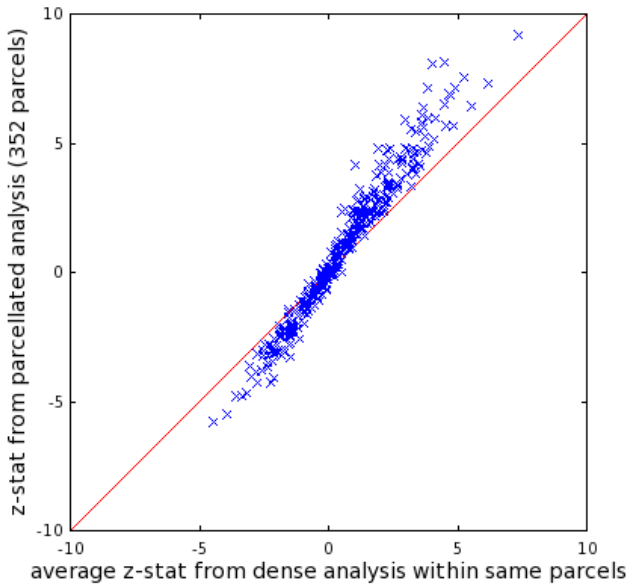
The desired commands in this case would be (in the linux / bash terminal):

```
cd ~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear
cd Results/tfMRI_EMOTION/tfMRI_EMOTION_hp200_s4_level3.gfeat
wb_command -cifti-parcellate \
tfMRI_EMOTION_level3_zstat1_hp200_s4.dscalar.nii \
~/day4-thursday/practical1-morning/\
Gordon_333_Parcel+FS_Anatomical_Parcel.dlabel.nii \
COLUMN tfMRI_EMOTION_level3_zstat1_hp200_s4.pscalar.nii
```

Load that data into Octave and create a scatterplot comparing the two approaches:

```
cd ~/day4-thursday/practical1-morning/DATA/U38/MNINonLinear/Results/\
tfMRI_EMOTION/tfMRI_EMOTION_hp200_s4_level3.gfeat;

file2 = 'tfMRI_EMOTION_level3_zstat1_hp200_s4.pscalar.nii';

C2 = ciftiopen(file2,'wb_command');

cdata2 = C2.cdata;

plot(cdata2(:,3),cdata(:,3),'x')

hold on

plot([-10 10],[-10 10],'r') %Create line of unity

xlabel('average z-stat from dense analysis within same
parcels','FontSize',14)

ylabel('z-stat from parcellated analysis (352 parcels)','FontSize',14)

axis square
```

You should get the following:



This figure illustrates the extent to which z-stats are increased by the parcellated analysis in this example. An important caveat to keep in mind is that the parcellation should be appropriate for the task.

B) Converting CIFTI into other formats

Let's remind ourselves what a CIFTI timeseries looks like…

In a new terminal window, enter:

```
cd /home/hcpcourse/day4-thursday/practical1-morning
```

```
wb_view ~/data/HCP_Q1-Q6_GroupAvg_Related440_Unrelated100_v1/\
HCP_Q1-Q6_R440_U100_DATA.32k_fs_LR.wb.spec &
```

- *Click* **Load** spec file
- Under the **File:Open File:Files of type:(*.dtseries.nii)**, navigate to and open the following file:

  /home/hcpcourse/day4-thursday/practical1-morning/DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/tfMRI_EMOTION_RL_Atlas.dtseries.nii
- *Set* the dtseries file that you just loaded as one of the displayed layers.
- Notice L and R cortical representation, and parcel-constrained grayordinates in subcortical volume
- *Click* through time points in timeseries.
- Back in the terminal window, *look at the dimensions and header contents* of the dtseries:

```
wb_command -file-information \
DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/\
tfMRI_EMOTION_RL_Atlas.dtseries.nii -no-map-info
```

Workbench can break this CIFTI-format timeseries into two GIFTI surfaces (one for each hemisphere) and a NIFTI subcortical volume. Let's do this.

- Separate CIFTI file into two GIFTI surfaces and one subcortical NIFTI volume:

```
wb_command -cifti-separate-all \
DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/\
tfMRI_EMOTION_RL_Atlas.dtseries.nii \
-volume 100307_tfMRI_EMOTION_RL_AtlasSubcortical.nii.gz -left \
100307_tfMRI_EMOTION_RL_atlasroi.L.32k_fs_LR.func.gii -right \
100307_tfMRI_EMOTION_RL_atlasroi.R.32k_fs_LR.func.gii
```

Now, let's take a look at the files that we just created.

- First the left hemisphere GIFTI file:

```
wb_command -file-information \
100307_tfMRI_EMOTION_RL_atlasroi.L.32k_fs_LR.func.gii -no-map-info
```

- Next, the volume NIFTI file:

```
wb_command -file-information \
100307_tfMRI_EMOTION_RL_AtlasSubcortical.nii.gz -no-map-info
```

The GIFTI and NIFTI files output by wb_command -cifti-separate-all maintain the spatial information contained in the original CIFTI file. However, these structures are now contained in three separate files, in different formats, that need to loaded and handled separately.

If you want to analyze surfaces using tools that support NIFTI but not GIFTI or CIFTI, the situation changes. The first step is to "wrap" the CIFTI data matrix into the X, Y, and Z dimensions of the NIFTI file. This is accomplished using the "wb_command -cifti-convert" command:

```
wb_command -cifti-convert -to-nifti \
DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/\
tfMRI_EMOTION_RL_Atlas.dtseries.nii \
100307_tfMRI_EMOTION_RL_fakeNIFTI.nii.gz
```

How exactly is the CIFTI matrix wrapped into a NIFTI file? The CIFTI file contains 91282 rows (grayordinates), but a NIFTI-1 file is limited to 32767 elements in each dimension. So, how are the grayordinates wrapped into the available space in the NIFTI file?

```
wb_command -file-information 100307_tfMRI_EMOTION_RL_fakeNIFTI.nii.gz \
-no-map-info
```

Note that the Dimensions of this "fakeNIFTI" file are 32767, 3, 1, 176

We've called this a "fake NIFTI" approach, since clearly the spatial information from surfaces or from subcortical volume is not maintained in the output NIFTI file. However, by referencing a template CIFTI file with the same spatial information as the original, the fake NIFTI can be unwrapped back into the CIFTI format.

```
wb_command -cifti-convert -from-nifti \
100307_tfMRI_EMOTION_RL_fakeNIFTI.nii.gz \
DATA/100307/MNINonLinear/Results/tfMRI_EMOTION_RL/\
tfMRI_EMOTION_RL_Atlas.dtseries.nii \
100307_tfMRI_EMOTION_RL_Atlas_unwrapped.dtseries.nii
```

WARNING: Analytic approaches that require spatial neighbor information are not appropriate using a "fake NIFTI" approach. For instance, procedures such as spatial smoothing should be done in the native CIFTI space using "wb_command -cifti-smoothing" prior to "wb_command -cifti-convert -to-nifti". Similarly, interpretation of "clusters" of activation should be done after converting fake NIFTI files back to CIFTI using "wb_command -cifti-convert -from-nifti". However, univariate statistics and processing that do not require information about the spatial neighborhood of the grayordinates (e.g., GLM analyses and temporal filtering) can be done using the fake NIFTI approach.

C) Reviewing scripts for Level 1 and Level 2 analyses

Let's review these scripts to identify the essential steps being done to conduct these analyses on CIFTI files.

Let's review the TaskfMRILevel1.v2.0.sh. First open the file:

```
gedit Pipelines/TaskfMRIAnalysis/scripts/TaskfMRILevel1.v2.0.sh &
```

See if you can determine where the following occur in the script: (Hint: search for the *italicized* items)

- Creating a *.ptseries* file when you're doing a parcellated analysis

- Editing the template *fsf* file using *sed* to utilize the temporal filter, spatial smoothing, and number of timepoints for this specific analysis
- Creating the design.mat, design.con, and design.fts using *feat_model*
- Using wb_command *-cifti-smoothing* to apply surface smoothing to the left and right cortex and parcel-constrained volumetric smoothing to subcortex
- Applying the temporal filter by doing a *FAKENIFTI* conversion, followed by fslmaths
- Doing a Level 1 dense grayordinates analysis using film_gls after using *-cifti-separate* to break the CIFTI into two surface timeseries (func.gii) and one subcortical volume (nii.gz)
- Doing a Level 1 *parcellated* analysis using film_gls after converting parcellated timeseries to a FAKENIFTI
- Running a Level 1 analysis using film_gls on the *Standard NIFTI Volume* data

Let's also review the TaskfMRILevel2.v2.0.sh. Again, locate or answer the following:

```
gedit Pipelines/TaskfMRIAnalysis/scripts/TaskfMRILevel2.v2.0.sh &
```

- Where is the output directory name modified in the fsf file (using *sed* to change *hp200_s4* to a name specific to your settings)?
- How are the ContrastNames determined from the Level 1 design.con?
- How does the script determine whether to run Grayordinates, Standard Volume, or Parcellated analysis? How does it run both Grayordinates and StandardVolumeStats?
- For Level 2 Grayordinate analyses, does the analysis (flameo) run on the original CIFTI, the separated GIFTI and NIFTI, or the fake NIFTI?
- What gets output into Contrasts.txt?

Authors:

The authors of this practical were Greg Burgess, Mike Harms, Matt Gasser, and Jennifer Elam, contributions from Erin Reid and Donna Dierker.