This documentation describes the process for creating the *MNI\_Glasser\_HCP* atlas. It is based on the parcellation described in the 2016 paper by Glasser, et al [1]. As described more thoroughly in that paper, this parcellation was created from clustering multi-modal data from the Human Connectome Project. Importantly (at least, importantly in the scheme of describing this atlas), that parcellation was defined in a standardized space utilizing multimodal surface matching (MSM) algorithm [2], a method for registering participants which leverages multiple modalities simultaneously. Here, we have mapped that parcellation onto the volumetric MNI152 template brain, utilizing only the T1-weighted anatomical image and the cortical folding patterns. This was done in a three-step transformation (MSM template -> surface template -> MNI152 surface -> MNI152 volume), with each step introducing some uncertainty for the precise locations of the parcellations. Of course, there are times when a volumetric atlas is useful and necessary, but any individual utilizing such an atlas should first ask themselves if one of the other intermediate (that is, closer to the space where the parcellation was originally created) transformations would be more appropriate to their purposes.

*For instance, a better way to leverage the work of Glasser, et al in your own study, assuming you don’t have the images you need to do their multi-modal registration, would be to run each subject through FreeSurfer, and use mri\_surf2surf to convert the fsaverage version to each subject’s space and average within parcellations on the surface. This would entirely sidestep the use of this atlas altogether. I’m OK with that.*

Of course, sometimes you can’t do that. So, then, you can use this atlas.

Scapegoating aside, we can now move on to handwaving. Here’s what took place to produce the atlas now before you. First (er… second, after Glasser and colleagues made the atlas in CIFTI standard space [1]), Kathryn Mills [3] took the trouble of converting the parcellation to fsaverage space. fsaverage [4] is a surface-based template distributed with Freesurfer [5]. Brains can be registered to fsaverage using cortical folding patterns, generally with the help of Freesurfer’s *recon-all* pipeline. To take the parcellation from fsaverage to MNI152, the volumetric template (the non-linearly warped version, which has clearer grey-white contrast) was submitted to *recon-all*, so that we had a cortical reconstruction. Freesurfer’s *mri\_surf2surf* tool was used to convert from the fsaverage surface to the MNI152 surface. Then, the cortical ribbon was sampled back to the volume.

Coordinates were also generated for each parcellation. This is trickier than it sounds on the surface (*get it?)* because the parcellations are oddly shaped. First, because they exist in the grey matter ribbon, they are folded such that the (volumetric) center of mass of the parcellation is often a point that is not included in the parcellation. Then, even when defined on the surface, many parcellations are C-shaped such that the centroid vertex has the same problem. To get around this, Freesurfer’s *mris\_divide\_parcellation* was used to cut each parcellation into equal pieces “perpendicular to the long axis.” Each division was set to be 75mm2 in size. This size was chosen arbitrarily but set to be small enough that each division was essentially a thin strip (and thus regularly shaped). The middle division (or in the union of the two central divisions) was identified. This region will have equal area on either side of it. The centroid vertex of this region was calculated to determine the coordinates for the parcellation. To ensure that these were, in fact, still in the volumetric grey matter ribbon, a 4mm sphere was calculated around the calculated coordinate, the union of this and the parcellation was taken, and the center of mass (volumetric) of that was kept as the final coordinate. All of these steps, while they create a reasonable coordinate for each parcellation, have considerable uncertainty attached, and no efforts whatsoever have been taken to characterize how much of an effect that may have on, for instance, seed placement for functional connectivity analyses. Use at your own risk, and don’t come crying to me if it doesn’t work. (Alternatively, if it works really well for you, I’ll take payment in chocolate.)

Regions on the left are numbered 1-180. Regions on the right are numbered 1001-11180. If you want to view the atlas, use the command

afni -XXXnpane 1180 MNI\_Glasser\_HCP+tlrc

To use the atlas…

* Make a folder somewhere (I used ~/CustomAFNI)
* Copy (attached) CustomAtlases.niml, .BRIK.gz, and .HEAD into that folder
* Run @AfniEnv -set AFNI\_SUPP\_ATLAS\_DIR ~/CustomAFNI/
* Run @AfniEnv -set AFNI\_ ATLAS\_LIST MNI\_Glasser\_HCP
* Run @AfniEnv -set AFNI\_ATLAS\_COLORS MNI\_Glasser\_HCP

Now you should be able to go to atlas locations, use the whereami function, and atlas colors based on the atlas.

If you want more information on the nitty-gritty of how the atlas was created, see the notes below.

[1] Glasser, M. F., Coalson, T. S., Robinson, E. C., Hacker, C. D., Harwell, J., Yacoub, E., ... & Smith, S. M. (2016). A multi-modal parcellation of human cerebral cortex. *Nature*, *536*(7615), 171-178.

[2] Robinson, E. C., Jbabdi, S., Glasser, M. F., Andersson, J., Burgess, G. C., Harms, M. P., ... & Jenkinson, M. (2014). MSM: a new flexible framework for multimodal surface matching. *Neuroimage*, *100*, 414-426.

[3] <https://figshare.com/articles/HCP-MMP1_0_projected_on_fsaverage/3498446>

[4] Fischl, B., Sereno, M. I., & Dale, A. M. (1999). Cortical surface-based analysis: II: inflation, flattening, and a surface-based coordinate system. *Neuroimage*, *9*(2), 195-207.

[5] Dale, A. M., Fischl, B., & Sereno, M. I. (1999). Cortical surface-based analysis: I. Segmentation and surface reconstruction. *Neuroimage*, *9*(2), 179-194.

*The nitty-gritty*

— Start with new version of Glasser atlas in fsaverage space. Downloaded from here:

https://osf.io/rne5b/

filenames ?h.Glasser2016.mgz

— convert to nii

mri\_convert ?h.Glasser2016.mgz ?h.Glasser2016.nii

— get a surface reconstruction of the MNI Atlas

3dZeropad -prefix ./MNI\_Atlas\_zp.nii -RL 256 -AP 256 -IS 256 /Applications/AFNI/MNI152\_T1\_2009c+tlrc

recon-all -all -sid MNI\_Atlas -i MNI\_Atlas\_zp.nii

@SUMA\_Make\_Spec\_FS -NIFTI -sid MNI\_Atlas

— use Freesurfer to move from fsaverage to MNI

mri\_surf2surf —srcsubject fsaverage —trgsubject MNI\_Atlas —sval atlasmgz/?h.Glasser2016.nii —tval MNI\_Atlas/SUMA/?h.Glasser\_HCP.gii —mapmethod nnf —hemi ?h

cd MNI\_Atlas/SUMA

— use SUMA to resample the surface to std.141

SurfToSurf -i\_gii std.141.?h.smoothwm.gii -i\_gii ?h.smoothwm.gii -prefix res.141.lh -mapfile std.141.MNI\_Atlas\_?h.niml.M2M -dset ?h.Glasser\_HCP.gii -output\_params NearestNode

mv res.141.?h.?h.Glasser\_HCP.niml.dset res.141.?h.Glasser\_HCP.niml.dset <- $hemi is in both prefix and dset from above, otherwise it will protest to overwriting the 1D file. Rename now so it doesn’t look silly, and so SUMA will load this later.

— make a fatter ribbon so that all of the grey voxels get a label (I guess that’s a judgement call. The method for \*not\* doing that should be obvious)

3dcalc -a ?h.ribbon.nii -datum float -prefix ?h.fl.ribbon.nii -expr 'a'

3dmerge -1blur\_fwhm 2 -doall -datum float -prefix ?h.blur.ribbon.nii ?h.fl.ribbon.nii

— Use the new fat-ribbon

@surf\_to\_vol\_spackle -spec std.141.MNI\_Atlas\_?h.spec -surfA smoothwm -surfB pial -maskset ?h.blur.ribbon.nii’<0.25..2>’ -surfset res.141.?h.Glasser\_HCP.niml.dset -mode -prefix ?h.Glasser\_HCP.fill

- Pad numbers so that left and right are separable

3dcalc -a lh.Glasser\_HCP.fill.nii.gz -expr 'a' -prefix lh.MNI\_Glasser\_HCP.nii

3dcalc -a rh.Glasser\_HCP.fill.nii.gz -expr 'a+(step(a)\*1000)' -prefix rh.MNI\_Glasser\_HCP.nii

- Combine into a single volume

3dcalc -a lh.MNI\_Glasser\_HCP+orig. -b rh.MNI\_Glasser\_HCP+orig. -expr 'max(a,b)' -prefix MNI\_Glasser\_HCP.nii

- smooth

3dLocalstat -stat mode -nbhd 'SPHERE(2)' -prefix hcp\_mode2 Glasser\_HCP\_labels.nii

3dcalc -a mode.MNI\_Glasser\_HCP.nii -b MNI\_Glasser\_HCP.nii -expr '(step(b)-step(a))\*b + a' -prefix smooth.MNI\_Glasser\_HCP.nii

- Add Atlas-y things

3drefit -space MNI smooth.MNI\_Glasser\_HCP.nii

3dcopy smooth.MNI\_Glasser\_HCP.nii MNI\_Glasser\_HCP+tlrc

3drefit -labeltable label\_tables.txt MNI\_Glasser\_HCP+tlrc

(open .HEAD file in a text editor)

\* atlas points were generated separately (see below)

\* copy paste Atlas points, adjusted to new numbers, fix character count

\* fixed a few stragglers

\* erased history

=============

=============

\*\*\* making atlas locations \*\*\*

=============

=============

(rh.HCP-MNI.annot is taken from here: https://figshare.com/articles/HCP-MMP1\_0\_projected\_on\_fsaverage/3498446)

mris\_divide\_parcellation MNI\_Atlas rh ./rh.HCP-MNI.annot 75 rh.75\_div.annot

mris\_divide\_parcellation MNI\_Atlas lh ./lh.HCP-MNI.annot 75 lh.75\_div.annot

this gives us a new annotation with each of the original parcellations divided up into equal pieces about 75 mm^2 in area

mri\_annotation2label --subject MNI\_Atlas --hemi ?h --ctab ./?h.75\_ctab.txt --annotation 75\_div

export the annotation information into a text file

(in matlab)

[lh\_temp.vol, lh\_temp.M, lh\_temp.mr\_parms, lh\_temp.volsz] = load\_mgh('lh.w-g.pct.mgh');

[rh\_temp.vol, rh\_temp.M, rh\_temp.mr\_parms, rh\_temp.volsz] = load\_mgh('rh.w-g.pct.mgh');

% get template .mgz information from another dataset from this same subject

[vertices lh\_temp.vol ctab] = read\_annotation('../label/lh.75\_div.annot');

[vertices rh\_temp.vol ctab] = read\_annotation('../label/rh.75\_div.annot');

% read in the annotation created above

save\_mgh(lh\_temp.vol, 'lh.75\_div.mgh', lh\_temp.M, lh\_temp.mr\_parms);

save\_mgh(rh\_temp.vol, 'rh.75\_div.mgh', rh\_temp.M, rh\_temp.mr\_parms);

% save the annotation as a .mgh file

!mri\_convert lh.75\_div.mgh lh.75\_div.nii

!mri\_convert rh.75\_div.mgh rh.75\_div.nii

% convert to .nii

still\_to\_do = [1:180];

for n = 1:180

a=num2str(n);

while length(a) < 3

a=['0' a];

end

[~,roi\_name] = system(['n=' a '; cat rh.75\_ctab.txt | awk -v N=$n ''$1==N{print $2}'' ']);

roi\_name = roi\_name(1:end-1);

% for each ROI, find the right lines in the annotation output file

[~,num\_divs] = system(['n=' a '; cat rh.75\_ctab.txt | grep ' roi\_name ' | tail -n 1 | awk -F \_div ''{print $2}'' | awk ''{print $1}'' ']);

num\_divs = str2num(num\_divs);

% figure out how many divisions this ROI was split into

if num\_divs/2 == round(num\_divs/2)

% even

div1 = (num\_divs/2);

[~,div\_label1] = system(['cat rh.75\_ctab.txt | grep ' roi\_name '\_div' num2str(div1) ' | awk ''{print $1}'' ']);

div\_label1 = div\_label1(1:end-1);

div2 = (num\_divs/2)+1;

[~,div\_label2] = system(['cat rh.75\_ctab.txt | grep ' roi\_name '\_div' num2str(div2) ' | awk ''{print $1}'' ']);

div\_label2 = div\_label2(1:end-1);

if div\_label1 + 1 ~= div\_label2

continue

end

% find the middle two divisions, check that they make sense

system(['mri\_surfcluster --in rh.75\_div.nii --subject MNI\_Atlas2 --hemi rh --surf pial --centroid --thmin ' div\_label1 ' --thmax ' div\_label2 ' --sum surfclust\_output\_div/rh.p.' a '.txt --nofixmni'])

system(['mri\_surfcluster --in rh.75\_div.nii --subject MNI\_Atlas2 --hemi rh --surf white --centroid --thmin ' div\_label1 ' --thmax ' div\_label2 ' --sum surfclust\_output\_div/rh.w.' a '.txt --nofixmni'])

% on both the pial surface and the white surface, find the location of the centroid voxel within the two central divisions

still\_to\_do(n) = 0;

% note that this ROI completed successfully

else

% odd

div = (num\_divs/2)+.5;

[~,div\_label] = system(['cat rh.75\_ctab.txt | grep ' roi\_name '\_div' num2str(div) ' | awk ''{print $1}'' ']);

div\_label = div\_label(1:end-1);

system(['mri\_surfcluster --in rh.75\_div.nii --subject MNI\_Atlas2 --hemi rh --surf pial --centroid --thmin ' div\_label ' --thmax ' div\_label ' --sum surfclust\_output\_div/rh.p.' a '.txt --nofixmni'])

system(['mri\_surfcluster --in rh.75\_div.nii --subject MNI\_Atlas2 --hemi rh --surf white --centroid --thmin ' div\_label ' --thmax ' div\_label ' --sum surfclust\_output\_div/rh.w.' a '.txt --nofixmni'])

still\_to\_do(n) = 0;

% same as above, but slightly simpler when there's an odd number of divisions

end

end

for n = 1:180

a=num2str(n);

while length(a) < 3

a=['0' a];

end

% read in the output files created above, parsing for the centroid location

[A,B] = system(['head -n 36 surfclust\_output\_div/lh.p.' a '.txt | tail -n 1 | awk ''{print $4}'' ']);

disp(B)

% I think this might be here to make sure there are the right number of lines in the file.

[A,B] = system(['head -n 35 surfclust\_output\_div/lh.p.' a '.txt | tail -n 1 | awk ''{print $5}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/lh.w.' a '.txt | tail -n 1 | awk ''{print $5}'' ']);

Coords\_lh(n,1) = -mean([str2num(B) str2num(C)]);

[A,B] = system(['head -n 35 surfclust\_output\_div/lh.p.' a '.txt | tail -n 1 | awk ''{print $7}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/lh.w.' a '.txt | tail -n 1 | awk ''{print $7}'' ']);

Coords\_lh(n,2) = mean([str2num(B) str2num(C)]);

[A,B] = system(['head -n 35 surfclust\_output\_div/lh.p.' a '.txt | tail -n 1 | awk ''{print $6}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/lh.w.' a '.txt | tail -n 1 | awk ''{print $6}'' ']);

Coords\_lh(n,3) = -mean([str2num(B) str2num(C)]);

% find each coordinate as the average of the x, y, and z locations between calculating the centroid vertex

% on the pial and the grey matter. It should (ideally) be the same vertex, so this should end up

% giving us the mid-cortical location of that vertex.

expr=['step(4-(x-' num2str(Coords\_lh(n,1)) ')\*(x-' num2str(Coords\_lh(n,1)) ')-(y-' num2str(Coords\_lh(n,2)) ')\*(y-' num2str(Coords\_lh(n,2)) ')-(z-' num2str(Coords\_lh(n,3)) ')\*(z-' num2str(Coords\_lh(n,3)) '))'];

pref = ['coord\_sph/out.lh.' a '.nii'];

p\_list = findstr(expr, '--');

for p = length(p\_list):-1:1

expr = [expr(1:p\_list(p)-1) '+' expr(p\_list(p)+2:end)];

end

% build the pieces of the 3dcalc command to make a sphere centered around the coordinate location I found

system(['3dcalc -a MNI\_Glasser\_HCP+tlrc -expr "' expr '" -prefix ' pref ' -overwrite'])

% and execute that 3dcalc command

clear expr pref p\_list Coords\_lh

[A,B] = system(['head -n 35 surfclust\_output\_div/rh.p.' a '.txt | tail -n 1 | awk ''{print $5}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/rh.w.' a '.txt | tail -n 1 | awk ''{print $5}'' ']);

Coords\_rh(n,1) = -mean([str2num(B) str2num(C)]);

[A,B] = system(['head -n 35 surfclust\_output\_div/rh.p.' a '.txt | tail -n 1 | awk ''{print $7}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/rh.w.' a '.txt | tail -n 1 | awk ''{print $7}'' ']);

Coords\_rh(n,2) = mean([str2num(B) str2num(C)]);

[A,B] = system(['head -n 35 surfclust\_output\_div/rh.p.' a '.txt | tail -n 1 | awk ''{print $6}'' ']);

[A,C] = system(['head -n 35 surfclust\_output\_div/rh.w.' a '.txt | tail -n 1 | awk ''{print $6}'' ']);

Coords\_rh(n,3) = -mean([str2num(B) str2num(C)]);

expr=['step(4-(x-' num2str(Coords\_rh(n,1)) ')\*(x-' num2str(Coords\_rh(n,1)) ')-(y-' num2str(Coords\_rh(n,2)) ')\*(y-' num2str(Coords\_rh(n,2)) ')-(z-' num2str(Coords\_rh(n,3)) ')\*(z-' num2str(Coords\_rh(n,3)) '))'];

pref = ['coord\_sph/out.rh.' a '.nii'];

p\_list = findstr(expr, '--');

for p = length(p\_list):-1:1

expr = [expr(1:p\_list(p)-1) '+' expr(p\_list(p)+2:end)];

end

system(['3dcalc -a MNI\_Glasser\_HCP+tlrc -expr "' expr '" -prefix ' pref ' -overwrite']);

clear expr pref p\_list Coords\_lh

% same thing for the other hemisphere

end

(back in bash)

for n in `count -dig 3 1 180`; do for hemi in 'lh' 'rh'; do

coord=`3dCM -mask MNI\_Glasser\_HCP+tlrc out.$hemi.$n.nii`

echo $hemi $n $coord >> coord\_list.txt

done

done

(aaaaaand back to matlab)

fid=fopen('atlas\_points.txt', 'w');

fid\_c = fopen('coord\_sph/coord\_list.txt', 'r');

%for n = 1:180

while 1

L = fgetl(fid\_c)

if L == -1

break

end

if strcmp(L(1:2), 'lh')

hemi\_add=0;

H='L\_';

else % rh

hemi\_add=1000;

H='R\_';

end

L\_nums = str2num(L(3:end));

n=L\_nums(1);

line = find(cell2mat(Names(:,1)) == n);

% I think I copied and pasted these into the workspace by hand

long\_parcel\_name = Names{line, 2};

parcel\_name = Names{line, 2};

parcel\_num = n+hemi\_add;

a=num2str(n);

while length(a) < 3

a=['0' a];

end

fprintf(fid, ['<ATLAS\_POINT\n data\_type="atlas\_point" \n STRUCT="%s%s"\n VAL="%d"\n OKEY="%d"\n ' ...

' GYoAR="0"\n COG = "%f %f %f"\n SB\_LABEL="%s%s" />\n'], ...

H, parcel\_name, parcel\_num, parcel\_num, L\_nums(2), L\_nums(3), L\_nums(4), H, long\_parcel\_name)

end

fclose(fid)

fclose(fid\_c)

=============

=============

\*\*\* done making atlas locations \*\*\*

=============

Yeah, I know. This was probably more work than it was worth, and probably not worth the wait. Oh, well. It's done now.

=============

# make some pretty surfaces

mkdir surfs\_mode2

cd surfs\_mode2

IsoSurface -isorois+dsets -o hcp\_mode2.gii -input ../hcp\_mode2\_ribbon+tlrc. -Tsmooth 0.3 30

# show these surfaces

echo "Use this command to see surfaces"

echo "suma -onestate -i surfs\_mode2/hcp\_mode2\*.gii &"