rfMRI Netmats & Dual Regression

We will use data from the HCP500 Parcellation + node Timeseries + Netmats (PTN) release, which includes data on the R468 subjects with complete rfMRI datasets, but loading in node-timeseries from only 10 subjects, so that things aren't too slow to run.

Overview

For most of these analyses we will use the FSLNets toolbox for carrying out basic network modelling from HCP rfMRI "node" timeseries data. FSLNets currently requires MATLAB (or Octave - a free alternative).

To estimate networks and perform a statistical analysis on this requires the following steps:

1. Process the data to get subject-specific timecourses relating to a given set of (e.g., group-ICA) spatial maps - for example, using dual regression stage 1 outputs

2. Look for outliers and remove "bad" group-ICA components (nodes)

3. Calculate correlations (full and partial) between all pairs of timeseries

4. View correlation matrix (network) weights/connections - looking at consistency and hierarchy

5. Perform statistical analysis over all the network, for a group of subjects

In part I you will learn how to generate network matrices from the output of step 1 of dual regression after looking at the timecourses' spectra and after detecting and removing bad components. In part II you will look at the network properties of your average group network, while in part III you will be running randomise to see if you can find group differences in network connectivity.

<u>Part I – Networks estimation</u>

We have already run the group-ICA "Parcellation" and the dual regression using 50 components (as part of the HCP R500 "PTN" data release). For more information about the PTN release, see the documentation at: humanconnectome.org/documentation/S500/HCP500_GroupICA+NodeTS+Netmats_Summary_28aug2014.pdf

The main thing you will feed into FSLNets network modelling is N timecourses of tp time points from S subjects' datasets - i.e., timeseries from N network nodes.

We are going to use the output of the first stage of dual regression as timeseries (S=20, tp=4800) and the 50 independent components as network nodes (N=50)

Now cd into the practical folder, start Octave, add the folder containing the scripts we are going to use to the path, and setup a few variables:

- From the terminal:

```
cd /home/hcpcourse/day2-tuesday/rfMRI_Practical_2
octave
```

- Then in Octave type:

```
addpath /usr/local/FSLNets
addpath(sprintf('%s/etc/matlab',getenv('FSLDIR')))
pkg load statistics
```

The above tells octave where to look for certain commands. Now set the group-ICA spatial maps directory (this is copied from the PTN data release, a subset of which we have provided in this practical's folder):

```
group_maps='PTN/groupICA/groupICA_3T_Q1-Q6related468_MSMsulc_d50.ica/melodic_IC_sum';
```

Set (previously-run) dual regression's output directory (this is where the node timeseries live):

```
ts_dir='CourseNodes';
```

You are now ready to load in all subjects' timeseries data files from the dual-regression output directory:

```
ts=nets_load(ts_dir,0.72,1,4);
```

The second parameter is the TR (temporal resolution) of the rfMRI data, which some later analyses may need to know.

The third parameter controls the normalisation applied to the standard deviation of each timeseries loaded:
0 means no normalisation
1 means normalise the overall standard deviation of all data points for each subject to be 1
2 means normalise each timeseries of each subject to have a standard deviation of 1.
If you only intend to do correlation-based netmat analysis, it makes no difference which option you choose here, because correlation is not affected by the overall amplitude of the timeseries being correlated. However, some analyses, such as simply studying different timeseries' amplitudes, would depend on this option.

The fourth parameter says that each subject's set of node timeseries is actually 4 runs already concatenated together. The reason for telling FSLNets this is that we have found that it works better to estimate netmats from the 4 runs separately and then average them – to get the best netmat estimate for

a given subject. Putting the “4” in the command tells FSLNets to keep the 4 runs separate from each other for now.

The loaded variable ts is a MATLAB "structure" - meaning that it contains several sub-variables, such as the TR, number of subjects loaded, and the raw timeseries data.

We will now have a quick look at the temporal spectra of the RSNs, as a quick check that these look reasonable (they should fall off smoothly with increasing frequency, and fall to zero at the very lowest frequencies, due to the highpass temporal filtering applied in the preprocessing of the raw rfMRI data):

```
ts_spectra=nets_spectra(ts);
```

The left part of the plot shows one spectrum per group-ICA component, each averaged across all subjects. The right part shows the same thing, but with all spectra shown on the same scale, and normalised to have the same peak value (to help showing outlier spectra). The median spectrum is also shown with a thick black line (and as a thin grey line under every plot on the left).

There is now the option to remove components' (nodes') timeseries that correspond to artefacts rather than plausible RSNs. This is done typically on the basis either of looking at both the spectra (e.g., having a strong higher-frequency tail, indicative of certain artefacts), and also on the basis of the group-ICA spatial maps. To view the spatial maps,

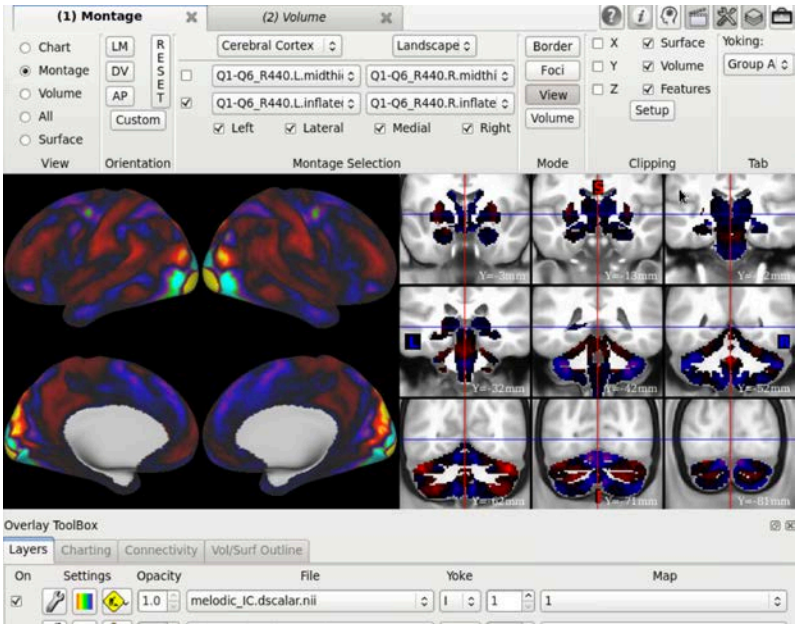
- Open a second terminal window.

```
cd /home/hcpcourse/day2-tuesday/rfMRI_Practical_2
```

- In this new terminal window, enter:

```
wb_view GroupAverage_ICA.scene &
```

- In the wb_view Scenes window, press **Show**.
- This will display the first group-average ICA component on inflated surface views plus coronal volume slices (see Figure). This component represents visual cortex, particularly central visual fields. (Note that there are only a few files loaded, but it would take many steps of instruction to not only load them but to configure each tab



appropriately. Scenes save time!)

- Click in the Map scroll box to activate, then use the arrow key to scroll through some of the ICA components.

Most components are largely or exclusively cortical, but many (e.g., #22) include a mixture of subcortical (e.g., cerebellar) as well as cerebral cortical regions. Some are predominantly subcortical (e.g., #30 is mainly cerebellar; #47 is mainly basal ganglia)

Examine components 49 and 50, which have a very different spatial pattern that is predominantly in the brainstem.

If we want to exclude bad components from further time series analysis, we list the "good" components (starting from 1), and then apply the removal/cleanup function. In this data it's a bit hard to tell if 49 and 50 are valid or artefactual - they might be valid - though they have very high levels of white noise in the spectra, and spatially are in the brainstem. For now let's say they are artefact, and exclude them. Back in the the terminal running octave, enter:

```
ts.DD=[1:48];

ts=nets_tsclean(ts,1);
```

The "1" tells FSLNets to remove the bad timeseries "aggressively", meaning that they are first regressed out of all the others, before being excluded. This is probably the right thing to do if the netmats of most interest are partial correlation netmats, which is generally the case here.

Now you are ready to compute a network matrix for each subject, which in general will be an NxN matrix of connection strengths (or N-[number of removed components]).

The simplest approach is just to use "full" correlation, giving an NxN matrix of correlation coefficients. Or, you might want to estimate the partial correlation matrix, which should do a better job of only estimating the direct network connections than the full correlation does.

We will compute 2 different versions of these "netmats" for each subject, so that we can compare their different properties. We will compute simple full correlation ('corr'), and partial correlation that has been "regularised" a very small amount (using ridge regression) in order to potentially improve the mathematical robustness of the estimation ('ridgep'). The extra parameter in the 'ridgep' case (0.01 in this case) controls the strength of the regularisation. Run in the octave terminal:

```
netmats1=nets_netmats(ts,1,'corr');

netmats2=nets_netmats(ts,1,'ridgep',0.01);
```

(Each of these will take a couple of minutes to run.)

For each of these lines, first, an NxN correlation matrix (aka network matrix or "netmat") is estimated for each subject, using the estimation method chosen (in the units of "r", or correlation coefficients). Then it is converted into a netmat of z-statistics (using the Fisher r to z transform). That is what the 1 controls (if 0 had been used instead, this conversion would not have taken place). The conversion takes into account the temporal smoothness of the data, to try to make sure that the resulting "z" values are valid.

We will now average the 4 netmats estimated for each subject and replace those 4 with the average – doing that separately for each subject. Don't worry if the following matlab commands are confusing – it is just taking each set of 4 rows in the netmats matrices, and replacing them with a single (average) row. Run, again in the octave terminal,:

```
tmp1=[]; tmp2=[];

for i=1:ts.Nsubjects/4

  tmp1=[tmp1; 2*mean(netmats1((i-1)*4+1:i*4,:))];

  tmp2=[tmp2; 2*mean(netmats2((i-1)*4+1:i*4,:))];

end;

netmats1=tmp1; netmats2=tmp2;
```

What's inside each of these new variables? For example, in the matrix netmats1, each row is a different subject's network matrix (netmat), with the NxN matrix entries "unwrapped" into a single line. So here we have 10 rows, each containing up to 50x50=2500 values, depending on how many timeseries we removed in the cleaning process above. You might argue that this matrix, which contains the parcellated functional connectomes from multiple subjects, represents the "complete" output of the (functional connectivity part of) the entire Human Connectome Project!

Part II - Group-average netmat summaries

We will now compute group-average netmat summaries - averaging netmats across all subjects (Mnet) and doing one-group t-tests across them (Znet).

A group-level netmat tells you about the group-average relationships between all N components. This is run separately for the 2 types of netmat (correlation and partial correlation). In each plot, the left part shows the output of the one-group t-test (a z-stat group netmat). On the right is a scatterplot of all individual subjects' netmat values against the group average - another way of judging cross-subject consistency.

```
[Znet1,Mnet1]=nets_groupmean(netmats1,1);

[Znet2,Mnet2]=nets_groupmean(netmats2,1);
```

It's important to be clear what each network matrix represents. An NxN netmat derived using full correlation of a single subject's set of N timeseries simply represents the correlation between every one of the N timeseries and every other. Hence each row - and each column - corresponds to the correlations between one particular timeseries and all others.

For example, the matrix entry in row 3 and column 6 tells you about the correlation between timeseries 3 and 6 - and these timeseries relate to the group-level spatial map numbers 3 and 6. A group-level netmat (e.g., either Mnet1 - the average of all subjects' individual netmats, or Znet1 - the z-statistic netmat derived from a one-group t-test across all subjects) likewise tells you about the group-level relationships between all N components. These N components (originally derived by the N-dimensional group-ICA, or alternatively any group-level parcellation or ROI-atlas) are also referred to as network "nodes". The elements in the network matrix are also referred to as network "edges" - for example, if you threshold a netmat and binarise the result, you have a set of edges that connect the nodes.

We can now view a "network hierarchy" visualisation of these group-level netmats.

```
nets_hierarchy(Znet1,Znet2,ts.DD,group_maps);
```

Don’t worry if running this command returns information like this in the terminal:

```
ans = 99th% abs value below diagonal is 5.165505

ans = 99th% abs value above diagonal is 5.364357

warning: your version of GraphicsMagick limits images to 8 bits per pixel
```

The command starts by taking the first parameter (here the group-level z-stat netmat from the full correlation) and reorders the nodes in order to bring together clusters of nodes that are highly correlated with each other (i.e. applies hierarchical clustering). This results in groupings of nodes, which form larger-scale "networks".

Enlarge the figure window (even bigger than the screen size might help) to see the details. You may be able to associate the different branches of the hierarchical tree to the main larger-scale networks described in the literature.

The first parameter in the command above (full correlation group netmat) is shown below the diagonal (such netmats are symmetric about the diagonal, so we only need to show one half of them). Above the diagonal is shown the second parameter - here the partial correlation group netmat. You should be able to see that the group-level netmats is "sparser" for partial correlation than full correlation – where sparser means a larger number of edges with values close to zero. This is because partial correlation is aiming to estimate only the direct connections strengths, and not just any (potentially indirect) correlation value.

The last parameter tells the command the location of a set of "thumbnail" images previously created from the original group-ICA spatial maps; and the parameter before that gives the list of “good” nodes remaining after the bad ones had been deleted earlier.

Finally, there is a command for producing a more interactive web-based version of the group netmats. Unfortunately, it runs in Matlab but not in Octave (due to a bug in Octave) – so we have already run this for you – we ran:

```
nets_netweb(Znet1,Znet2,ts.DD,group_maps,'netweb');   % don't run this
```

You can open the result from this in a web browser. For example, to do that from inside Octave, type:

```
system('firefox netweb/index.html &')
```

You might want to make the web browser window as large as possible and then reload the page. You can click on individual nodes to see all supra-threshold connections to that node (try changing the threshold). You can change various aspects of the display to being driven by full or partial correlation – switch between these to see many clear differences. You can view the “whole” network (for a given set threshold) by turning on “Highlight network”. You can separate out a sub-view of just those nodes connected to a selected node by turning on “Show sub-network”.

Part III - Cross-subject stats with netmats

Univariate cross-subject netmat modelling

We are now in the position to test how the netmats vary across subjects.

This is a "univariate" regression, meaning that we will test each network matrix element separately for whether it correlates with our cross-subject design (covariate) - and then we will estimate p-values for these tests, correcting for the multiple comparisons across all netmat elements (network edges). By analogy to high-level task-fMRI analyses: you can think of each subject's netmat as being an NxN image of voxels, and the univariate testing as modelling each voxel (in isolation from each other) across subjects.

We have created a design matrix (covariate of interest, plus a column of 1s to model the group mean connection strength), as well as a contrast vector [1 0] which tells the modelling to only care about the covariate of interest. These are saved in files “design.mat” and “design.con”. Let’s assume that the covariate of interest is IQ – an example non-imaging measure that we want to test for association with network connections in the brain. (Note that in fact this design is artificially created, in order to allow us to find a statistically significant effect in this data despite the very low number of subjects.) In general you can create such designs with tools such as the “Glm” GUI in FSL.

The command to run (which calls FSL’s randomise from within matlab, with 5000 permutations for each contrast), is the following:

```
[p_uncorrected,p_corrected]=nets_glm(netmats2,'design.mat','design.con',1);
```

The first parameter passed into this command is the set of subjects' netmats - here netmats2 is the set of all subjects' regularized partial correlation netmats.

Once randomise has finished, you will see a figure showing a "netmat" containing p-values (corrected for multiple comparisons across all netmat elements tested). For convenience of display we are actually showing 1-p values – so displayed values very close to 1 are significant. Below the diagonal (imaginary diagonal across the plot) are the (corrected) 1-p values, and above the diagonal we are showing the netmat elements where the test is significant, at corrected-p<0.05. You will probably see that only one network edge significantly correlates with IQ.

We will now run a command that displays the 6 netmat elements (edges) which have the most significant p-values, ordered in decreasing strength of association:

```
nets_edgepics(ts,group_maps,Znet2,reshape(p_corrected(1,:),ts.Nnodes,ts.Nnodes),6);
```

Each pair of thumbnails corresponds to one position in the NxN network matrix, and the node numbers are listed in the text captions. The coloured bar joining each pair of nodes tells you what the overall group-average connection (full, partial or regularized partial correlation, depending on the 3rd parameter passed to the command) strength is: thicker means a stronger connection; red means it's positive, and blue means that the connection is "negative" (meaning that the two nodes tend to anti-correlate on average). The "value" numbers tell you the 1-p-values - so the higher these are, the more significantly the correlation between IQ and this edge strength across the population. Anything less than 0.95 is not significant, after correcting for multiple comparisons. Note the node numbers for the strongest edge and look at these node maps in Workbench, to visualise the nodes' maps in more detail.

Note that your interpretation of a significant positive correlation between IQ and edge strengths (across subjects) is likely to depend on the sign and size of the group-average connection (shown with the coloured bars). For example, if the group mean is negative (blue), then a positive IQ correlation means that higher IQ is associated with smaller negative connection strengths. Finally, we will explain later in the week how it is important to take into account family structure when doing such permutation testing – and describe tools for doing so.

Part IV – Optional extra 1 – Creating and viewing parcellated timeseries and netmats for Workbench

Note: This section relies on having loaded in the node-timeseries and calculated single subject netmats in Octave, as done above.

We now show how to create files containing node-timeseries and netmats, in a form that can be viewed directly in Workbench. To create such files, containing data from inside your Matlab session, first involves creating "dummy" versions of these files, then reading them into Matlab (including all their header information), overwriting the data (as opposed to header) parts, and then writing the files back out to disk.

First, you can run the following to setup various variables that you'll need:

Note: for all the commands in this practical do not copy/paste anything after the %. These are just explanatory comments.

```
WBC='/usr/local/workbench/bin_rh_linux64/wb_command'; % wb_command binary

PARCEL_LABELS='PTN/groupICA/groupICA_3T_Q1-Q6related468_MSMsulc_d50.ica/melodic_IC_ftb.dlabel.nii';
          % group-ICA "labels" file
```

Now, you can create a dummy parcel-timeseries file, read it into Matlab, insert the actual parcel timeseries data from one run of one subject, and save it back out. Note that we have to deal with the fact that we deleted some timeseries.

```
system(sprintf('%s -cifti-parcellate rfMRI_REST1_LR_Atlas_hp2000_clean.dtseries.nii %s COLUMN Sub1Run1.ptseries.nii',WBC,PARCEL_LABELS));
    % create dummy parcel timeseries file

grot=ciftiopen('Sub1Run1.ptseries.nii',WBC);

grot.cdata=zeros(ts.NnodesOrig,1200);
    % zero all the data inside this

grot.cdata(ts.DD,:)=ts.ts(1:1200,:)';
    % write the real parcel-timeseries data into it

ciftisave(grot,'Sub1Run1.ptseries.nii',WBC);
    % save back out to file
```

Now, you can do the same thing with single-subject netmats, saving out separate files for full and partial correlation:

```
system(sprintf('%s -cifti-correlation Sub1Run1.ptseries.nii Sub1Run1_fullcorrelation.pconn.nii',WBC))

grot=ciftiopen('Sub1Run1_fullcorrelation.pconn.nii',WBC);  % read netmat file
into Matlab

grot.cdata=zeros(ts.NnodesOrig);  % zero all the data in it

grot.cdata(ts.DD,ts.DD)=reshape(netmats1(1,:),ts.Nnodes,ts.Nnodes);
    % write the correlation netmat into it

ciftisave(grot,'Sub1Run1_fullcorrelation.pconn.nii',WBC);  % save the file out

grot.cdata=zeros(ts.NnodesOrig);  % zero the data again

grot.cdata(ts.DD,ts.DD)=reshape(netmats2(1,:),ts.Nnodes,ts.Nnodes);
    % write the partial correlation netmat

ciftisave(grot,'Sub1Run1_partialcorrelation.pconn.nii',WBC);
    % save the file out to a second file
```

- In wb_view, double-click the second scene ("Full correlation pconn (Sub1Run1). If you already closed wb_view, re-launch it in a terminal window: wb_view GroupAverage_ICA.scene &

This will show a parcellated connectivity map displayed on surfaces and volume slices for a left parietal parcel (indicated by the white outline form, not colored).

- Click around on various locations on surfaces or volume slices to see the functional connectivity maps for different parcels.

- Double-click the third scene ("Full correlation surf, volume; full corr, partial corr matrices"). You will now see two additional tabs displayed in the montage, showing the full correlation matrix (lower left) and the partial correlation matrix (lower right).

- Click around on various locations on surfaces or volume slices to see the functional connectivity maps for different parcels, along with the corresponding highlighted rows of the two connectivity matrices, which are all yoked together.

Setting up a montage display like this takes a bit of time if you don't have a pre-existing scene like the ones you created in rfMRI Practical 1.

Part V – Optional extra 2 – Heavy netmat regularisation

Here is an example of what happens if you use stronger partial correlation regularisation, with the "L1-norm" regularisation method - which seems to perform the best in the case of datasets of lower quality than HCP. We start by loading a much smaller dataset (just two subjects), because this netmat method is much slower than the ones used above. We then run the same netmat methods as above, as well as a third method – the L1-norm, applied with a high regularisation parameter:

```
tsB=nets_load('CourseNodes_2subjects',0.72,1,4);

netmats1B=nets_netmats(tsB,1,'corr');

netmats2B=nets_netmats(tsB,1,'ridgep',0.01);

netmats3B=nets_netmats(tsB,1,'icov',100); % takes a few minutes
```

The command below lets us look at the estimated netmat values for just the top row of each netmat, for the 2 subjects and 4 runs. Each row in the display is a different run/subject, and each column is the connection strength between different nodes and node 1.

```
imagesc(netmats3B(:,1:tsB.Nnodes),[-5 5]); colorbar;
```

Note how there are a lot of empty (zero) values. This network estimation method has estimated that there is no "edge" (connection) at all between two nodes at that point in the netmat. Note, however, how some zeros vary between the subjects – and even between different runs for a given subject! That is clearly a

limitation for using this method on this data – it can't be ideal to combine across these different estimated netmats (across runs and/or subjects) when the sets of network edges aren't properly compatible. Have a look at the same thing for the other two sets of netmats that we just estimated above (1 and 2).

As a further illustration of the effect of the sparsification with this method, the following command displays a scatterplot (across all netmat elements) of the full correlation vs L1-regularised partial correlation, after averaging each across the 4 runs. You can see that a lot of values that are non-zero in the full correlation have been zeroed in the partial correlation.

```
scatter(mean(netmats1B),mean(netmats3B));
```

Authors:

The author of this practical was Steve Smith, with contributions from Jennifer Elam, Erin Reid, and David Van Essen.