Network Flexibility and Reinforcement Learning

Raphael Gerraty, 2015-2016

Example scripts for running network preprocessing and analysis functions contained here. See paper for details when it comes out.

Preprocessing

Need to add

Running nonlinear registration with FNIRT

After nuisance regression has been run, the residual timeseries needs to be transformed into standard space (in this case, MNI). Make sure fsl_anat has been run on each structural image first. The following bash code was used to perform these transformations:

```bash
#fnirt has already been run, just applying transformation
for i in /data/engine/rgerraty/learn_dyncon/4*/Learn*;
    do
    #run linear registration on example functional image
    flirt -ref $i/../structural/mprage.anat/T1_biascorr_brain.nii.gz\
     -in $i/reg/example_func.nii.gz\
     -omat $i/reg/example_func2highres.mat;

     echo warping $i;

    #apply warp from FNIRT to preprocessed 4D data
    applywarp --ref=$FSLDIR/data/standard/MNI152_T1_2mm.nii.gz\
      --in=$i/36par+spikes.feat/stats/res4d.nii.gz\
      --out=$i/36par+spikes.feat/stats/res4d_std.nii.gz\
      --warp=$i/../structural/mprage.anat/T1_to_MNI_nonlin_field.nii.gz\
      --premat=$i/reg/example_func2highres.mat;
done
```

Extracting time courses

Once the preprocessed images have been registered, we extract summary timecourses (using the 1st eigenvector) for each Harvard-Oxford ROI, using the function extract_ROIs.sh. The output of this function is a timecourse for each ROI in the specified input folder, as well as a .txt file containing all of the ROIs. The bash code used to run this function on each learning block for each subject is below:

```
for i in /data/engine/rgerraty/learn_dyncon/4*/Learn?_PEprior.feat/36par+spikes.feat/;
    do
    #extract timeseries (1st eigenvector) data from each ROI in ~/Harvard-Oxford_ROIs/
    ~/GitHub/rl_flexibility/extract_ROIs.sh $i/stats/res4d_std.nii.gz ~/Harvard-Oxford_ROIs/
done
```

Calculate coherence matrices for each time window

```
addpath ~/GitHub/rl_flexibility
%read in all subject/run ROI timeseries directories
[a,b]=system('ls -d /data/engine/rgerraty/learn_dyncon/4*/Learn?_PEprior.feat/36par+spikes.f
c=strread(b,'%s');

for i=1:size(c,1)

    %calculate coherence per time window from concatenated ROI file
    filename=char(strcat(c(i),'/all_rois.txt'))
    %need to specify filename, window length in TR, sampling rate, bandpass
    conn_cell=coherence_by_block(filename,25,.5,.06,.12);
    save(char(strcat(c(i),'/conn_cells')),'conn_cell')

end
```

Run multi-slice community detection and flexibility statistics

Input coherence matrix for each block. Also need number of blocks, resolution and coupling parameters. In Matlab

```
%need multi-slice, flexibility codes not yet on GitHub for network_diags to run
addpath ~/GitHub/rl_flexibility
addpath ~/scripts/MATLAB/GenLouvain_for_Raphael/
addpath ~/scripts/MATLAB/Bassett_Code/

%read in data
[a,b]=system('ls -d /data/engine/rgerraty/learn_dyncon/4*/Learn?_PEprior.feat/36par+spikes.f
c=strread(b,'%s');

%concatenate runs for each subject
numruns=4
k=1;
for j=1:size(c,1)/numruns
    c(k)
    conn_cell_cat=[];
    for i=1:numruns
        load(strcat(char(c(k-1+i)),'/conn_cells'))
        conn_cell_cat=cat(3,conn_cell_cat,conn_cell)
    end

    %network_diags code:
    %runs multi-slice community detection
    %gives flexibility for each run
    %also allegiance matrix (not using yet)
    %need to specify number of blocks, simulations, coupling, resolution
    [a_mat,flex]=network_diags(conn_cell_cat,4,100,1,1.1813)
    save(char(strcat(c(k),'/../../../../a_mat')),'a_mat')
    save(char(strcat(c(k),'/../../../../flex')),'flex')
    k=k+numruns;
end
```

Pull flexibility statistics

For plotting and preparing for heirarchical models. Matlab.

```
%load data and concatenate flexibility statistics
[a,b]=system('ls -d /data/engine/rgerraty/learn_dyncon/4*/flex.mat');
c=strread(b,'%s');
flex_cat=[];
for j=1:size(c,1)
    load(char(c(j)))
    flex_cat=cat(3,flex_cat,flex)
end
plot(squeeze(mean(flex_cat)))

block=repmat([1:4]',22,1);
sub=repmat([1:22]',1,4)'
sub=sub(:);

%reshape whole-brain average flexibility
meanflex=squeeze(mean(flex_cat));
meanflex=meanflex(:);

%get striatal average flexibility
str_ind=[49,51,54,104,106,109];
strflex=squeeze(mean(flex_cat(str_ind,:,:)));
strflex=strflex(:);

plot(squeeze(mean(flex_cat(str_ind,:,:))))

%write out csv for modeling in R
```

```
flexdata=[sub block meanflex strflex]
dlmwrite('/data/engine/rgerraty/learn_dyncon/flexdata.csv',flexdata)
```

```
data<-read.delim('~/DynLearn/mem_lrn_mixmod.txt',header=1)
flexdat<-read.delim('/data/engine/rgerraty/learn_dyncon/flexdata.csv',header=F)
names(flexdat)<-c('subject,'block','wb_flex','str_flex')
```