Introduction to Computational Medicine - Final Report

Table of Contents

Joe and the Joes

Abstract:

In this report, we will pose the question of whether the three rat brains in the Brain Architecture Management System Rat Connectome Project database are similar enough in terms of associativity to be described as similar brains. To do this, we intend to iterate through different cluster-sized stochastic block models (SBM), and at each step, determine if any of the three graphs are sampled from that sized SBM. If the three graphs all conform to the same set of SBMs, we will conclude that they are of similar associativity.

Question:

Informal:

The problem we're trying to solve is whether all three rat brains can be modeled by the same size (number of clusters) stochastic block model, or whether they cannot. This will indicate the overall similarity in the brain associativity.

Formal:

Given Gi, i ? {1,2,3} where Gi ?Gis the space of all graphs that are directed and loopy and have 503 vertices and unweighted edges, and Gi denotes the adjacency graphs for three rat brains, H0: Gi ~ SBMk Ha: Gi ~ SBMj j ? k

Where k,h ? {1…5} denote number of blocks in the stochastic block model (SBM).

Test Statistic:

Where ? is a maximum likelihood ratio between the null and alternative hypotheses.

We will generate the null and alternative distributions using a Monte Carlo simulation on the respective SBM block model sizes.

For each k, we will produce a result for a given $p < 0.05$ whether each graph is distributed as a size k SBM. Each graph will have a set of k sized SBMs where they match, and we will match those sets of k between graphs. If there are elements in common, we will conclude that the rat brains are of the same associativity.

Pseudocode:

Inputs: rat brain files. Outputs: singificance levels for each combination of cluster sizes Extract rat matrices from database For each rat, create an adjacency matrix for their graphs For clusters of size n and m, Use spectral clustering algorithm to create two vertex sets for each rat, sized n and m Find maximum likelihoods for each edge in each cluster Use to create a random matrix sampled from SBM of each size Compute the likelihood ratio of these random graphs being sampled from SBMs of each size Apply heuristic penalty for larger block size: subtract ~ number of blocks^2 Evaluate when likelihood ratio of rat graphs is greater or less than this ratio If all rats are distributed from one sized SBMs statistically, conclude they are similarly associative.

Spectral Clustering Pseudocode Input: the adjacency matrix (W), the number of clusters (k), Type of normalization (Type) Output: Clustercell First iterate through the adjacency matrix to convert any element greater than or equal to 1 to 1 Then calculate the degree matrix from the sum of the simplified matrix From the degree matrix and the simplified matrix, compute the normalized Laplacian If specified, normalization Laplacian using algorithms specified in Shi and Malik (2000) (Type = 2) or Jordan and Weiss (2002) (Type = 3) Compute the eigenvectors based off of the k smallest eigenvalues Use k-means to cluster the eigenvectors Place each cluster in a separate array and place all arrays into Clustercell

Data:

We will use the three rat graphs given in the Brain Architecture Management System Rat Connectome Project. This data set is the best available for comparing individuals from a population of individuals and compare properties between them.

Model

We will be modeling the different graphs as Stochastic Block Model random graphs. In order to do this, given the fact that the rat brains are weighted random graphs, we would need to apply some thresholding to make each edge's connection binary instead of an integer. Hypothetically, this is not optimal for biological data such as brain connectomes, as other non-bernoulli type connection schemes generally are better models (scale-free networks, etc.) However, we chose to use SBMs for simplicity.

Results:

Rats 1,2,3: Significance levels for difference in block sizes. Since no size (1-5) is consistently the best fit for each rat, there is no overlap and therefore we cannot conclude that the rats are of similar associativity. For instance, rat #1 had H0 cluster size of 2 and significantly not 3, 4, or 5 (using the figure, and disregarding cluster size 1), however this was not observed for rats #2 or #3, (with rats #2 and #3, Ha also worked) etc.

Discussion:

Although our sample size is low, if we take our results to indicative of a larger phenomena, clustering is not similar across species. This fact provokes further study into the significance of these clustering differences and how they translate to individual rat development differences. However, the conclusion that the rat brains are not similar is not accessible, since we only compared them using a very specific metric. In addition, we made several assumptions about the distribtion of the graph itself, including the statement that all the edge probabilities were independent, and identically distributed within a cluster and between clusters. However, in biological settings, independence is rare and perhaps a more scale-free graph structure would be a more accurate description.

Significance:

Determining whether connectivity clustering is similar across species will provide a basis for further research into the implications of connectivity on organisms function and even possibly allow for the quantification of the effect of environmental differences on cognitive processes.

Sources:

Shi and Malek (2000) http://www.cs.berkeley.edu/~malik/papers/SM-ncut.pdf Jordan and Weiss (2002) http://ai.stanford.edu/~ang/papers/nips01-spectral.pdf Spectral Clustering Code (Ingo Buerk 2011/2012) http://www.mathworks.com/matlabcentral/fileexchange/34412-fast-and-efficient-spectral-clustering

%%Code

```
function []=ICM

warning('off','all') %optional
load('D:\DownloadsD\rats (1).mat')
ratGraphs = {rat1,rat2,rat3};

maxCluster=7;
nmc=50;

alphaList=ones(maxCluster-1,maxCluster-1,3);

for n = 2:maxCluster %number of clusters in SBM for H0
    for m = 2:maxCluster %number of clusters in SBM for HA
        if(n ~= m)
            clusterList0={[],[],[]};
            clusterList1={[],[],[]};

            for g=1:3 %for each rat, form stochastic block models in
 h0 and h1


 clusterList0{g}=SpectralClustering(ratGraphs{g},n,2); %theta_hat_0

 clusterList1{g}=SpectralClustering(ratGraphs{g},m,2); %theta_hat_1
            end


            for g=1:3

                %Likelihoods are using ln(likelihood) because matlab
 does
                %not handle numbers of such low magnitude
                [LH_0real,pHatMat0] =
 LikelihoodEstimation(ratGraphs{g},clusterList0{g},n);
```

```
                [LH_1real,pHatMat1] =
 LikelihoodEstimation(ratGraphs{g},clusterList1{g},m);

                h0count = 0; %count of how many times L_h0 > L_h1

                for mc=1:nmc
                    %fprintf('%d,',mc)
                    A=zeros(size(ratGraphs{g})); %Generate a random
 graph based on h0

                    for i = 1:n
                        for j=1:n

                            for a=(clusterList0{g}{i})
                                for b=(clusterList0{g}{j})
                                    pHat=pHatMat0(i,j);

                                    A(a,b)=(rand<pHat);
                                end
                            end


                        end
                    end



 [LH_0,~]=LikelihoodEstimation(A,clusterList0{g},n);

 [LH_1,~]=LikelihoodEstimation(A,clusterList1{g},m);

                    LH_0=abs(LH_0);
                    LH_1=abs(LH_1)-n*m*50;%heuristic measure of
 penalty applied

                    h0count = h0count + (LH_0 > LH_1);
                end

                alpha = (h0count+1)/(nmc+1);
                alphaList(n,m,g)=alpha;



                %disp(['LH_0: ' num2str(LH_0) ' \tLH_1: '
 num2str(LH_1)]);
                disp([num2str(alpha) ' alpha, for H0 = SBM_'
 num2str(n) ', HA = SBM_' num2str(m)])
            end
        end
    end
end

B=alphaList(2:end,2:end,:);
```

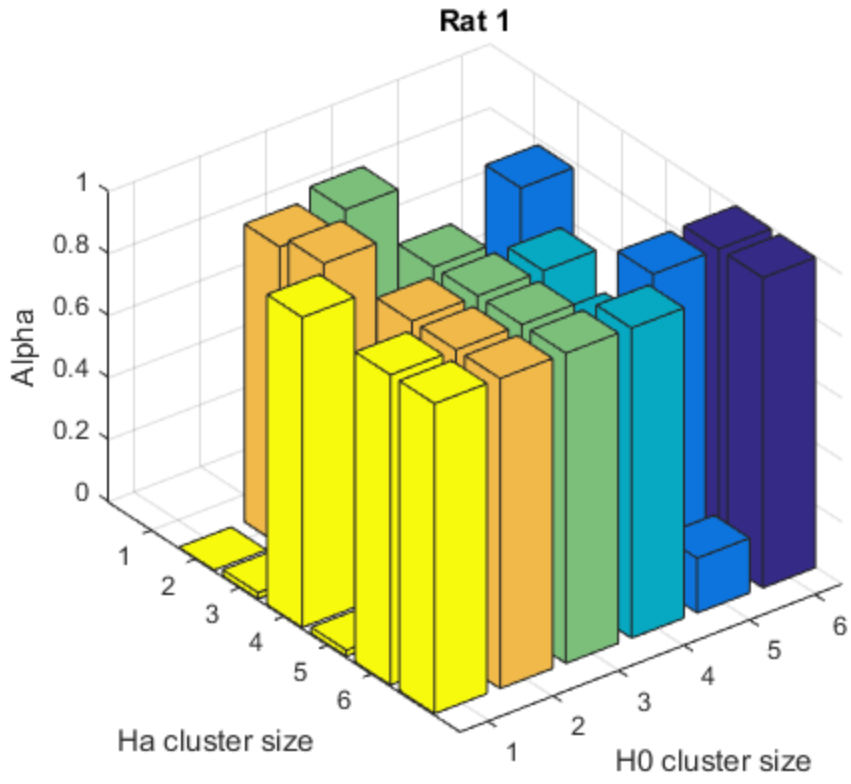
```
figure
f=bar3(B(:,:,1));
xlabel('H0 cluster size')
ylabel('Ha cluster size')
zlabel('Alpha')
title('Rat 1')
rotate(f,[0,0,1],180)

figure
f=bar3(B(:,:,2));
xlabel('H0 cluster size')
ylabel('Ha cluster size')
zlabel('Alpha')
title('Rat 2')
rotate(f,[0,0,1],180)

figure
f=bar3(B(:,:,3));
xlabel('H0 cluster size')
ylabel('Ha cluster size')
zlabel('Alpha')
title('Rat 3')
rotate(f,[0,0,1],180)

end

function [lnLH,pHatMat] =
 LikelihoodEstimation(adjacency,clusters,numClusters)
m=numClusters;
pHatMat=zeros(m,m);
lnLH = 0;
for i = 1:m
    for j=1:m

        sum = 0;

        for a=(clusters{i})
            for b=(clusters{j})
                sum=sum+adjacency(a,b);
            end
        end
        blockSize=length(clusters{i}) * length(clusters{j});

        pHat=sum/blockSize;

        pHatMat(i,j)=pHat;

        %to avoid taking the ln(0), avoid pHat = 0 or 1
        if(pHat ~= 0)
            lnLH = lnLH + log(pHat)*(sum);
        end
        if(pHat ~= 1)
            lnLH = lnLH + log(1-pHat)*(blockSize-sum);
        end
```

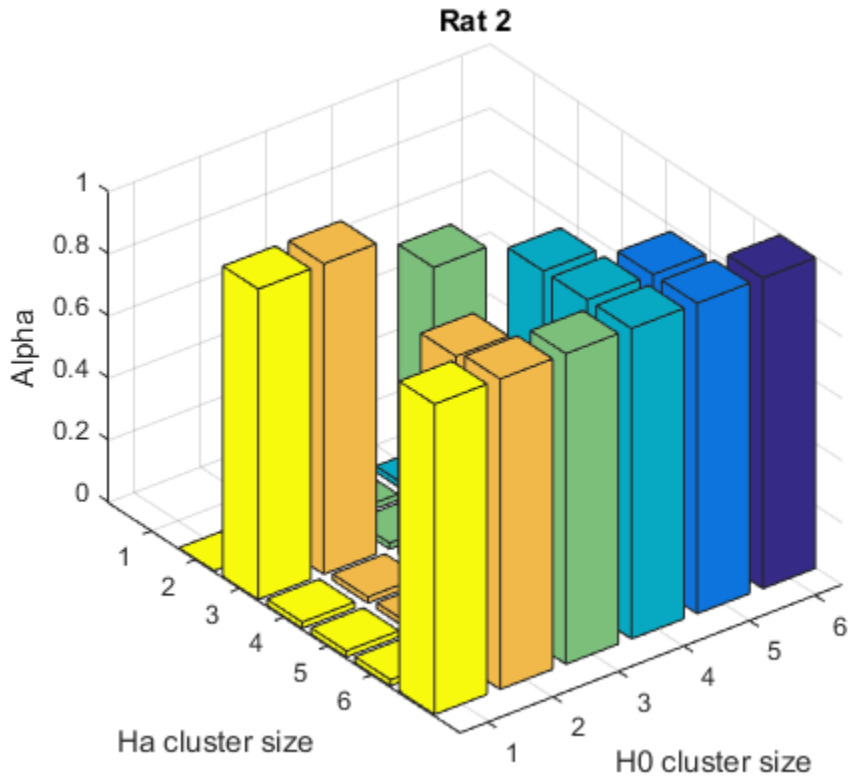
```
    end
end


end

function [clustercell, L, U] = SpectralClustering(W, k, Type)
%SPECTRALCLUSTERING Executes spectral clustering algorithm
%   Executes the spectral clustering algorithm defined by
%   Type on the adjacency matrix W and returns the k cluster
%   indicator vectors as columns in C.
%   If L and U are also called, the (normalized) Laplacian and
%   eigenvectors will also be returned.
%
%   'W' - Adjacency matrix, needs to be square
%   'k' - Number of clusters to look for
%   'Type' - Defines the type of spectral clustering algorithm
%            that should be used. Choices are:
%      1 - Unnormalized
%      2 - Normalized according to Shi and Malik (2000)
%      3 - Normalized according to Jordan and Weiss (2002)
%
%   References:
%   - Ulrike von Luxburg, "A Tutorial on Spectral Clustering",
%     Statistics and Computing 17 (4), 2007
%
%   Author: Ingo Buerk
%   Year  : 2011/2012
%   Bachelor Thesis
%Convert to 1 to 0 - Chris
for i = 1:length(W)
    for j = 1:length(W)
        if W(i,j) > 0
            W(i,j) = 1;
        end
    end
end
% calculate degree matrix
degs = sum(W, 2);
D    = sparse(1:size(W, 1), 1:size(W, 2), degs);
% compute unnormalized Laplacian
L = D - W;
% compute normalized Laplacian if needed
switch Type
    case 2
        % avoid dividing by zero
        degs(degs == 0) = eps;
        % calculate inverse of D
        D = spdiags(1./degs, 0, size(D, 1), size(D, 2));

        % calculate normalized Laplacian
        L = D * L;
    case 3
```

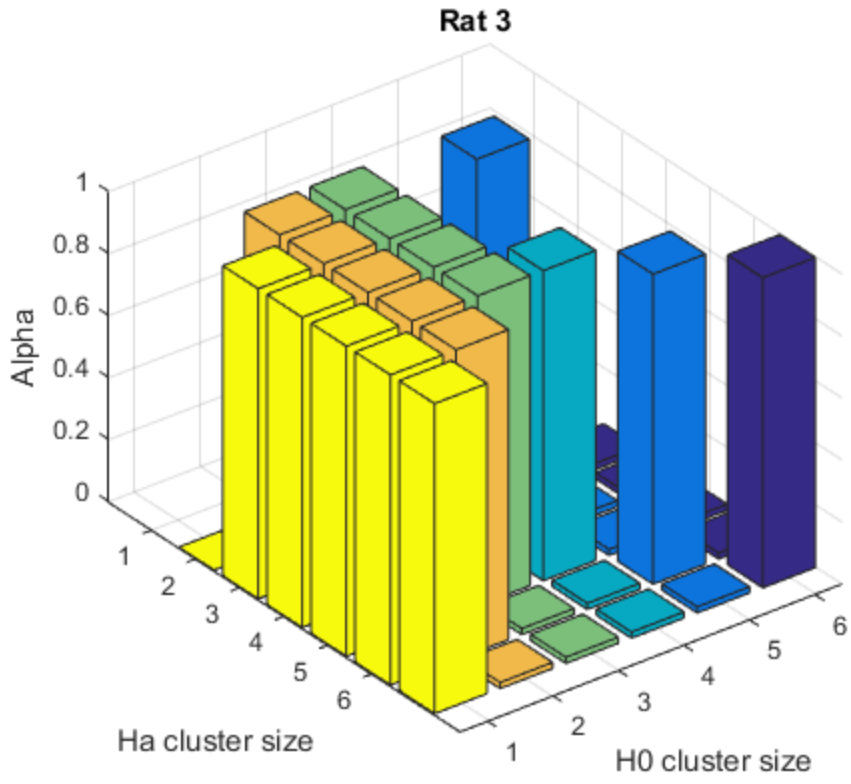
```
        % avoid dividing by zero
        degs(degs == 0) = eps;
        % calculate D^(-1/2)
        D = spdiags(1./(degs.^0.5), 0, size(D, 1), size(D, 2));

        % calculate normalized Laplacian
        L = D * L * D;
end
% compute the eigenvectors corresponding to the k smallest
% eigenvalues
diff   = eps;
[U, ~] = eigs(L, k, diff);
% in case of the Jordan-Weiss algorithm, we need to normalize
% the eigenvectors row-wise
if Type == 3
    U = bsxfun(@rdivide, U, sqrt(sum(U.^2, 2)));
end
% now use the k-means algorithm to cluster U row-wise
% C will be a n-by-1 matrix containing the cluster number for
% each data point
try
    C = kmeans(U, k, 'start', 'cluster', ...
        'EmptyAction', 'singleton');
catch ME
    clustercell = SpectralClustering(W,k,Type);
    return
end



% now convert C to a n-by-k matrix containing the k indicator
% vectors as columns
C = sparse(1:size(D, 1), C, 1);

C = full(C);
clustercell = cell(1,k); %generate and populate clustercell with
 placeholders

for i = 1:size(C,1)
    for j = 1:k
        if C(i,j) == 1
            clustercell{1,j} = [clustercell{1,j}, i];
        end
    end
end

end

0.17647 alpha, for H0 = SBM_2, HA = SBM_3
1 alpha, for H0 = SBM_2, HA = SBM_3
0.019608 alpha, for H0 = SBM_2, HA = SBM_3
1 alpha, for H0 = SBM_2, HA = SBM_4
1 alpha, for H0 = SBM_2, HA = SBM_4
0.019608 alpha, for H0 = SBM_2, HA = SBM_4
```

```
1 alpha, for H0 = SBM_2, HA = SBM_5
1 alpha, for H0 = SBM_2, HA = SBM_5
0.019608 alpha, for H0 = SBM_2, HA = SBM_5
1 alpha, for H0 = SBM_2, HA = SBM_6
1 alpha, for H0 = SBM_2, HA = SBM_6
0.019608 alpha, for H0 = SBM_2, HA = SBM_6
1 alpha, for H0 = SBM_2, HA = SBM_7
1 alpha, for H0 = SBM_2, HA = SBM_7
1 alpha, for H0 = SBM_2, HA = SBM_7
1 alpha, for H0 = SBM_3, HA = SBM_2
0.019608 alpha, for H0 = SBM_3, HA = SBM_2
0.019608 alpha, for H0 = SBM_3, HA = SBM_2
0.88235 alpha, for H0 = SBM_3, HA = SBM_4
1 alpha, for H0 = SBM_3, HA = SBM_4
0.019608 alpha, for H0 = SBM_3, HA = SBM_4
1 alpha, for H0 = SBM_3, HA = SBM_5
0.019608 alpha, for H0 = SBM_3, HA = SBM_5
0.019608 alpha, for H0 = SBM_3, HA = SBM_5
1 alpha, for H0 = SBM_3, HA = SBM_6
0.98039 alpha, for H0 = SBM_3, HA = SBM_6
1 alpha, for H0 = SBM_3, HA = SBM_6
1 alpha, for H0 = SBM_3, HA = SBM_7
0.019608 alpha, for H0 = SBM_3, HA = SBM_7
1 alpha, for H0 = SBM_3, HA = SBM_7
0.019608 alpha, for H0 = SBM_4, HA = SBM_2
0.019608 alpha, for H0 = SBM_4, HA = SBM_2
0.019608 alpha, for H0 = SBM_4, HA = SBM_2
0.019608 alpha, for H0 = SBM_4, HA = SBM_3
0.019608 alpha, for H0 = SBM_4, HA = SBM_3
0.019608 alpha, for H0 = SBM_4, HA = SBM_3
1 alpha, for H0 = SBM_4, HA = SBM_5
0.019608 alpha, for H0 = SBM_4, HA = SBM_5
1 alpha, for H0 = SBM_4, HA = SBM_5
1 alpha, for H0 = SBM_4, HA = SBM_6
0.019608 alpha, for H0 = SBM_4, HA = SBM_6
1 alpha, for H0 = SBM_4, HA = SBM_6
0.019608 alpha, for H0 = SBM_4, HA = SBM_7
0.019608 alpha, for H0 = SBM_4, HA = SBM_7
1 alpha, for H0 = SBM_4, HA = SBM_7
0.019608 alpha, for H0 = SBM_5, HA = SBM_2
0.019608 alpha, for H0 = SBM_5, HA = SBM_2
0.019608 alpha, for H0 = SBM_5, HA = SBM_2
0.019608 alpha, for H0 = SBM_5, HA = SBM_3
0.019608 alpha, for H0 = SBM_5, HA = SBM_3
0.019608 alpha, for H0 = SBM_5, HA = SBM_3
0.019608 alpha, for H0 = SBM_5, HA = SBM_4
0.019608 alpha, for H0 = SBM_5, HA = SBM_4
0.019608 alpha, for H0 = SBM_5, HA = SBM_4
0.019608 alpha, for H0 = SBM_5, HA = SBM_6
0.019608 alpha, for H0 = SBM_5, HA = SBM_6
1 alpha, for H0 = SBM_5, HA = SBM_6
1 alpha, for H0 = SBM_5, HA = SBM_7
0.019608 alpha, for H0 = SBM_5, HA = SBM_7
1 alpha, for H0 = SBM_5, HA = SBM_7
```

```
0.019608 alpha, for H0 = SBM_6, HA = SBM_2
0.019608 alpha, for H0 = SBM_6, HA = SBM_2
0.019608 alpha, for H0 = SBM_6, HA = SBM_2
1 alpha, for H0 = SBM_6, HA = SBM_3
0.019608 alpha, for H0 = SBM_6, HA = SBM_3
0.019608 alpha, for H0 = SBM_6, HA = SBM_3
0.019608 alpha, for H0 = SBM_6, HA = SBM_4
0.019608 alpha, for H0 = SBM_6, HA = SBM_4
0.019608 alpha, for H0 = SBM_6, HA = SBM_4
0.019608 alpha, for H0 = SBM_6, HA = SBM_5
0.019608 alpha, for H0 = SBM_6, HA = SBM_5
1 alpha, for H0 = SBM_6, HA = SBM_5
0.019608 alpha, for H0 = SBM_6, HA = SBM_7
1 alpha, for H0 = SBM_6, HA = SBM_7
1 alpha, for H0 = SBM_6, HA = SBM_7
0.019608 alpha, for H0 = SBM_7, HA = SBM_2
0.019608 alpha, for H0 = SBM_7, HA = SBM_2
0.019608 alpha, for H0 = SBM_7, HA = SBM_2
0.019608 alpha, for H0 = SBM_7, HA = SBM_3
0.019608 alpha, for H0 = SBM_7, HA = SBM_3
1 alpha, for H0 = SBM_7, HA = SBM_3
0.019608 alpha, for H0 = SBM_7, HA = SBM_4
0.019608 alpha, for H0 = SBM_7, HA = SBM_4
0.019608 alpha, for H0 = SBM_7, HA = SBM_4
1 alpha, for H0 = SBM_7, HA = SBM_5
0.019608 alpha, for H0 = SBM_7, HA = SBM_5
1 alpha, for H0 = SBM_7, HA = SBM_5
0.96078 alpha, for H0 = SBM_7, HA = SBM_6
0.019608 alpha, for H0 = SBM_7, HA = SBM_6
1 alpha, for H0 = SBM_7, HA = SBM_6
```


Rat 1
Alpha
0
0.2
0.4
0.6
0.8
1
Ha cluster size
1
2
3
4
5
6
H0 cluster size
1
2
3
4
5
6


Rat 2
Alpha
0
0.2
0.4
0.6
0.8
1
Ha cluster size
1
2
3
4
5
6
H0 cluster size
1
2
3
4
5
6



Published with MATLAB® R2015a