OKACOGNET Survival Kit

Tools used in the project.

Eclipse/workspace/OKACOGNET/Resources/April-2013/ Okacognet23subjHOLM-YZ-p01 and Okacognet19-elderHOLM-YZ-p01.m

Data for Elder in OKACOGNET/Resources/Elder-19/ FunImgFC\_AALTC

Data for Young **in** OKACOGNET/Resources/FunImgFC\_AALTC

**Pajek for Graph Visualization.**

It is windows native software (.exe) to run in Mac OS, download

Wine (an icon with a half empty glass shows in top bar)

Wine-Task File Manager Pajek Pajek.exe

Open Network (.net) click on draw icon on the right, first row

**Brain connectivity**

[**Brain Connectivity Toolbox**](http://sites.google.com/a/brain-connectivity-toolbox.net/bct/) **(BCT)**

VisualConnectome (VisualConnectome is a Matlab toolbox for brain connectivity analysis and visualization. It aims to make it easier for analysis of brain connectivity which is constructed from structural MRI, DTI and functional MRI data, and to visualize the brain connectivity to obtain an intuitive insight of brain network.)

[SurfStat](http://www.math.mcgill.ca/keith/surfstat/) and [Brain Connectivity Toolbox](http://sites.google.com/a/brain-connectivity-toolbox.net/bct/) is used as plugins to import brain surface file and calculate network topological measures in this toolbox.

**Measures related with distance (network vulnerability)**

The geodesic path between two nodes i.e., the shortest path using Floyd’s algorithm (ojo para weighted directed graph) (OKACOGNET/software/MITNetwork/FloydSPR.m)

Use

|  |  |
| --- | --- |
| [graphshortestpath](http://www.mathworks.es/es/help/bioinfo/ref/graphshortestpath.html) | Solve shortest path problem in graph |

**From** [**http://www.mathworks.es/es/help/bioinfo/network-analysis-and-visualization.html**](http://www.mathworks.es/es/help/bioinfo/network-analysis-and-visualization.html)Apply basic graph theory algorithms to Protein-Protein Interactions (PPI) and other gene networks; view network relationships using interactive maps, hierarchy plots, and pathways (Bioinformatics toolbox STD of Matlab)

*spRZcor=sparse(RZcor);*

*n\_effic=0;*

*for i= 1:90*

*for j=1:90*

*if i ==j*

*else*

*[mydist,mypath,mypred]= graphshortestpath(spRZcor,i,j,'directed',false);*

*n\_effic=n\_effic+(1/mydist);*

*end*

*end*

*end*

*glob\_n\_effic= (n\_effic)/(90\*(90-1));*

(glob\_n\_effic = 0.3795)

I created graph\_efficiency.m

*%% Jaime algorithm*

*function [eff\_value]=graph\_efficiency(AdjMax)*

*% \*INPUT:\**

*% AdjMax: Adjacent matrix that represents a sparse graph*

*%*

*% \*OUTPUT:\**

*% S: eficiency value calculated based on Latora-Melchiori genios*

Network vulnerability is then:

%graphshortestpath(spRZcor,90,'directed',false);

%Calculate Network vulnerability

%glob\_n\_vul=max(node\_vul)

spRredu=sparse(RZcor);

node\_eff=zeros(1,90);

node\_vul=zeros(1,90);

for i=1:90

%buld new matrix deleting the node and all its links

spRredu(i,:)=[];

spRredu(:,i)=[];

%call to efficiency function

spRredu=sparse(spRredu);

node\_eff(1,i)= graph\_efficiency(spRredu);

node\_vul(1,i)= (glob\_n\_effic-node\_eff(1,i))/glob\_n\_effic;

spRredu=sparse(RZcor);

end

glob\_n\_vul=max(node\_vul)

%glob\_n\_vul =0.0201 (quite robust network , because young subjects? Check with older ones?)

We can, furthermore, calculate the ranking of vulnerability of each node: (max vertex 60, min vertex 89)

(As suggested by Gol’dshtein et al. [69], the ordered distribution of vertices with respect to their

vulnerability **V**i is related to the network hierarchy, thus the most vulnerable (critical)

vertex occupies the highest position in the network hierarchy.)

NB: We should study Improvability (http://arxiv.org/abs/cond-mat/0407491)

>> [B,IX]=sort(node\_vul)

B =

Columns 1 through 12

-0.0227 -0.0111 -0.0099 -0.0090 -0.0068 -0.0068 -0.0053 -0.0053 -0.0050 -0.0047 -0.0038 -0.0037

Columns 13 through 24

-0.0028 -0.0028 -0.0026 -0.0025 -0.0023 -0.0022 -0.0017 -0.0015 -0.0014 -0.0013 -0.0012 -0.0011

Columns 25 through 36

-0.0006 -0.0003 0.0001 0.0002 0.0002 0.0004 0.0005 0.0006 0.0006 0.0016 0.0017 0.0018

Columns 37 through 48

0.0019 0.0022 0.0023 0.0025 0.0026 0.0027 0.0027 0.0028 0.0029 0.0031 0.0032 0.0035

Columns 49 through 60

0.0039 0.0041 0.0041 0.0042 0.0043 0.0044 0.0044 0.0044 0.0044 0.0046 0.0050 0.0051

Columns 61 through 72

0.0054 0.0054 0.0056 0.0056 0.0056 0.0057 0.0059 0.0060 0.0060 0.0061 0.0068 0.0068

Columns 73 through 84

0.0068 0.0069 0.0070 0.0072 0.0075 0.0079 0.0082 0.0083 0.0089 0.0090 0.0092 0.0099

Columns 85 through 90

0.0106 0.0123 0.0136 0.0143 0.0173 0.0201

IX =

Columns 1 through 20

89 35 72 86 85 39 38 37 3 69 70 78 21 4 24 58 77 25 6 9

Columns 21 through 40

41 76 88 22 82 57 55 59 1 53 28 2 7 81 80 54 66 79 67 44

Columns 41 through 60

18 56 5 36 17 42 10 26 12 27 68 11 33 75 16 8 83 64 87 63

Columns 61 through 80

43 40 34 61 45 51 29 65 47 52 49 71 48 23 15 62 84 13 20 90

Columns 81 through 90

74 32 19 14 30 73 46 50 31 60

**Characterisation of Triangles**

%Transitivity: The transitivity is the ratio of triangles to triplets in the network and is an alternative to the clustering coefficient.

transitivity\_bu(RZcor) %BCT toolbox

ans =

0.6104

Using BCT there are several measures like:

***Assortativity:***The assortativity coefficient is a correlation coefficient between the degrees of all nodes on two opposite ends of a link. A positive assortativity coefficient indicates that nodes tend to link to other nodes with the same or similar degree.

assortativity\_bin(RZcor,0)

ans =

0.4358

[assortativity\_bin](https://sites.google.com/site/bctnet/Home/functions/assortativity_bin.m?attredirects=0)

***Rich club coefficient:*** The rich club coefficient at level k is the fraction of edges that connect nodes of degree k or higher out of the maximum number of edges that such nodes might share.

**Centrality Measurements**

***Betweenness centrality:*** Node betweenness centrality is the fraction of all shortest paths in the network that contain a given node. Nodes with high values of betweenness centrality participate in a large number of shortest paths.

??? Error using ==> mtimes

Logical inputs must be scalar.

Error in ==> betweenness\_bin at 34

NPd=NPd\*G;

***Edge betweenness centrality:*** Edge betweenness centrality is the fraction of all shortest paths in the network that contain a given edge. Edges with high values of betweenness centrality participate in a large number of shortest paths.

**Hypothesis Testing**

Statistics Toolbox offers a number of hypothesis tests

*normplot(degree\_vec)* shows match of normal distribution for the observable debree\_vec

The Kolmogorov-Smirnov test states that degree distribution is normal

*kstest(degree\_vec)*

*ans =*

*1*

*>> lillietest(degree\_vec)*

*ans =*

*1*

*We find a contradiction here, because while kstest and lillietest refute the bull hypothesis, the* Jarque-Bera test (The test is specifically designed for alternatives in [Generating Data Using the Pearson System](http://www.mathworks.es/es/help/stats/generating-data-using-flexible-families-of-distributions.html#br5k833-4) of distributions. The test returns the value h = 1 if it rejects the null hypothesis at the 5% significance level, and h = 0 if it cannot.) returns 0 therefore accepts the null hypothesis that data come from a normal distribution with unknown mean and variance. BUT *the* Jarque-Bera is inappropriate here because is for large value samples, eferring to the Lilliefors test (see [lillietest](http://www.mathworks.es/es/help/stats/lillietest.html)) for small samples.

**Network identification**

Calculating the Randomized counterpart of a given complex network

There are several algorithms to calculate the randomized counterpart in order to identify the topological properties that are under or over represented in the real network compared to the

In order to assess the significance of a given motif, we a real or canonical network, which has been obtained through empirical work to be compared with a population of randomly generated networks. Thus, we need the baseline or null hypothesis given by the randomly generated networks. The objective is to compare (metrics) the real network with its equivalent synthetic (randomly generated) networks (Olaf Sporns. Networks of the Brain . The MIT Press, 1 edition, October 2010.).

i. MIT Toolbox

Software:

<http://strategic.mit.edu/downloads.php?page=matlab_networks>

 [random\_graph.m](http://strategic.mit.edu/docs/matlab_networks/random_graph.m) - generate a random graph adjacency matrix using various models;

random\_graph(n) - Erdos-Renyi graph with **n** nodes and probability of attachment 0.5;  
random\_graph(n,p) - Erdos-Renyi graph with **n** nodes and probability of attachment **p**;   
random\_graph(n,[ ],E) - random graph with E number of edges;  
random\_graph(n,[ ],[ ],distribution) - nodal degrees are drawn from a particular distribution (uniform, normal, binomial, exponential);  
random\_graph(n,[ ],[ ],'sequence',deg\_seq) - the degrees match a given sequence;

For that, I need to compute number of edges and notes, n and e, n=90 and edges is given by (numnodes.m)

 [getNodes.m](http://strategic.mit.edu/docs/matlab_networks/getNodes.m) - return the list of nodes for varying graph representations;

 [getEdges.m](http://strategic.mit.edu/docs/matlab_networks/getEdges.m) - return the list of edges for varying graph representations;

 [numnodes.m](http://strategic.mit.edu/docs/matlab_networks/numnodes.m) - number of vertices/nodes in the network;

 [numedges.m](http://strategic.mit.edu/docs/matlab_networks/numedges.m) - number of edges/links in the network;

There are also other basic functions to test properties of the graph

issymmetric(RZcor), numnodes(RZcor), numedges(RZcor), isequal(RZcor,RZcor)

Get a population of Random graph with n , E (using Erdos Renyi model?)

To get one random graph:

*RRandcor=random\_graph(numnodes(RZcor),[ ],numedges(RZcor))*

We use a loop to create a population of random graphs with n nodes and e edges.

*randomgraphs=zeros(90,90,90)*

*M=100% number of random matrix to be generated*

*for i= 1:M*

*randomgraphs(i,:,:)=random\_graph(RZnumnodes,[ ],RZnumedges)*

*end*

ii. Sergei Maslov

Sergei Maslov and Kim Sneppen. Specificity and stability in topology

of protein networks. arXiv:cond-mat/0205380 , May 2002. Science, 296,

910-913 (2002).

(note Maslov matlab library (4 .m’s) are in my computer in same dir. Than the MIT lib /OKACOGNETsoftware/MITNetwork)

Here, randomization is preserving the degrees of each nod, that is, each node in the generated random networks will have the same number of immediate neighbors.

To generate a randomized null-model network in which the degrees of all

nodes are strictly preserved, for undirected graph :

*RanMaslov = sym\_generate\_srand(RZcor)*

%Syntax:

%[R\_12,Z\_12,n\_12,nr\_12, nsr\_12]=sym\_corr\_profile(s1,Nstat,edge\_bins);

% INPUT:

% srand=sym\_generate\_srand(s1)

% s1 - the adjacency matrix of an undirected network

% Nstat - (optional) the number of randomized networks in the ensemble. Default: 3

% edge\_bins - (otional) the array to bin degrees. Default: [1,3,10,30,100...]

% OUTPUT:

% n\_12 - number of edges connecting different bins to each other

% nr\_12 - same averaged over Nstat realizations of a randomized network

% nsr\_12 - square of nr\_12 averaged over Nstat realizations of a randomized network

% R\_12 - correlation profile ratio: R\_12=n\_12./nr\_12;

% Z\_12 - correlation profile Z-score: Z\_12=(n\_12-nr\_12)./sqrt(nsr\_12-nr\_12.^2);

*Maslovresults=sym\_correlation\_profile(RZcor,100)*

**Matlab Code**

To build the adjacency matrix that will be later visualized with Pajek:

1. “load .mat file for each subject, .mat is the putput of SPM for R-fMRI”

*load('/Users/jagomez/Eclipse/workspace/OKACOGNET/Resources/FunImgFC\_AALTC/sub\_001\_AALTC.mat')*

2. “declare a vector with the activity information for each of the 90 areas”

*Msub1=[AAL01TC, AAL02TC,AAL03TC,AAL04TC,AAL05TC,AAL06TC,AAL07TC,AAL08TC,AAL09TC,AAL10TC,AAL11TC, AAL12TC,AAL13TC,AAL14TC,AAL15TC,AAL16TC,AAL17TC,AAL18TC,AAL19TC,AAL20TC,AAL21TC,AAL22TC,AAL23TC,AAL24TC, AAL25TC,AAL26TC,AAL27TC,AAL28TC,AAL29TC,AAL30TC,AAL31TC,AAL32TC,AAL33TC,AAL34TC, AAL35TC,AAL36TC,AAL37TC,AAL38TC,AAL39TC,AAL40TC,AAL41TC,AAL42TC,AAL43TC,AAL44TC,AAL45TC,AAL46TC,AAL47TC, AAL48TC,AAL49TC,AAL50TC,AAL51TC,AAL52TC,AAL53TC,AAL54TC,AAL55TC,AAL56TC,AAL57TC, AAL58TC,AAL59TC,AAL60TC,AAL61TC,AAL62TC,AAL63TC,AAL64TC,AAL65TC,AAL66TC,AAL67TC,AAL68TC,AAL69TC,AAL70TC, AAL71TC,AAL72TC,AAL73TC,AAL74TC,AAL75TC,AAL76TC,AAL77TC,AAL78TC,AAL79TC,AAL80TC, AAL81TC,AAL82TC,AAL83TC,AAL84TC,AAL85TC,AAL86TC,AAL87TC,AAL88TC,AAL89TC,AAL90TC]*

3. Create MsubAll matrix, a 3D matrix (172,90,23) = (time series vector, areas, subjects)

*MsubAll=zeros(172,90,23);*

*MsubAll(:,:,1)= Msub1;*

*…*

*MsubAll(:,:,23)= Msub23;*

4. %Calculate correlation coefficient matrix for each subject

*RsubAll=zeros(90,90,23)*

*for i=1:23*

*RsubAll(:,:,i)= corrcoef(MsubAll(:,:,i));*

*end*

5. Calculate z transform

6. t-test for p = 0.001…

7. Correction with bonferroni-holmes

*[PScor, RZcor]=bonf\_holm(PS,0.001);*

*8.Save data and export to Pajek*

savefile= 'MsubTotalSubjects.mat'

coords = rand(90,3)

write\_matrix\_to\_pajek(RZ,'/Users/jagomez/Eclipse/workspace/OKACOGNET/Resources/MatLabLib/RZ23sub.net','weighted',false,'directed',false,'coords',coords );

xlswrite('/Users/jagomez/Eclipse/workspace/OKACOGNET/Resources/MatLabLib/FinaleRZ23-HOLM-YL.xls',RZcor, 'A1:CL90')

9.Use libraries to analyze the graph **RZcor (calculate local and global properties like clustering, char path and others)**

% Add plfit to Matlab <http://tuvalu.santafe.edu/~aaronc/powerlaws/plfit.m>

% PLFIT fits a power-law distributional model to data.

The output 'alpha' is the maximum likelihood estimate of the scaling exponent, 'xmin' is the estimate of the lower bound of the power-law behavior, and L is the log-likelihood of the data x>=xmin under the fitted power law.

PLFIT(x) estimates x\_min and alpha according to the goodness-of-fit based method described in Clauset, Shalizi, Newman (2007). x is a vector of observations of some quantity to which we wish to fit the power-law distribution p(x) ~ x^-alpha for x >= xmin.  
[alpha, xmin, L] = plfit(degree)

%Plplot plots (on log-log axes) the empirical distribution along with the fitted power-law distributio

h = plplot(degree,xmin,alpha);

%(equiv. the fraction of node\_s neighbors that neighbors of each other).

%Input: A, binary undirected connection matrix

%Output: C, clustering coefficient vector

Cpsw= clustering\_coef\_bu(RZcor);

cpsw= mean(Cpsw);

%Transitivity: The transitivity is the ratio of triangles to triplets in the network and is an alternative to the clustering coefficient.

(How diff is from Categories, colimits?)

transitivity\_bu(RZcor)

Lp=charpath(RZcor)

10. Network identification

Compare real network chars with randomly generated equivalent network

Ratio Cp/Cr >1 then small world etc.