Supplementary Information for

Comparing meso-scale maps given by community detection algorithms on the Emmons laboratory Caenorhabditis elegans connectome data

George Vernon

George Vernon
E-mail: george.vernon@warwick.ac.uk

This PDF file includes:

Supplementary text

Supporting Information Text

Subhead. This SI provides the code used to generate and manipulate stochastic block models on the C. elegans connectome. Some data must be imported first for the script to run properly.

Codes

ERMM R script. This script performs 1,000 iterations at a time for the specified `qmin` and `qmax`.

```
library(mixer)
NeuronLabels <- read.table("NeuronLabels.csv", sep=",")
connectomesymbin <- read.table("StrucConnectomeSymBinLabels.csv", header = TRUE, sep = ",")
bestcriterion <- -100000

numsamples = 1000
mtrx <- matrix(nrow = 3, ncol = numsamples)
bestbestcriterion <- bestbestm$criterion

start_time <- Sys.time()
for (i in 1:numsamples) {
  mtrx[1,i] <- i
  setSeed(unclass(Sys.time()))
  mixer(connectomesymbin, qmin=5, qmin=15, nbiter=80, fpnbiter=40) -> ERMMout
  m <- getModel(ERMMout)
  m$criterion -> criterion
  if (criterion > bestcriterion) {
    bestcriterion <- criterion
    bestm <- m
    bestERMMout <- ERMMout
  }
  mtrx[2,i] <- criterion
  mtrx[3,i] <- m$q
  print("Done")
}

if (bestcriterion > bestbestcriterion) {
  bestbestcriterion <- bestcriterion
  bestbestm <- bestm
  bestbestERMMout <- bestERMMout
}

sortedmtrx <- mtrx[, order(mtrx[2,],decreasing=TRUE)]
write.table(sortedmtrx, "filename.txt")

end_time <- Sys.time()

time <- end_time - start_time

print(time)
```

ERMM map script. This script extracts the group mapping from the MixeR `getModel` function output.

```
# R script for finding highest tau for each node

groupmap <- matrix(nrow = 1, ncol = 300)

for (i in 1:300) {
  groupmap[i] <- which.max(bestbesttaus[,i])
}

write.table(groupmap, "groupmap.txt")
\end{lstlistin}

\subsection*{MATLAB Louvain Block Model Script}

This script performs 20,000 iterations of the Louvain algorithm, takes the best one and produces a plot.

\begin{lstlisting}
GapJunctionsSym = GapJunctionsSym - diag(diag(GapJunctionsSym));
GapJunctionsSymBin = GapJunctionsSym ~= 0;

ConnectomeSymBin = GapJunctionsSymBin + ChemSynapsBinSym;
ConnectomeSymBin = ConnectomeSymBin ~= 0;

```

```
ConnectomeSymBin = double(ConnectomeSymBin);
ConnectomeSymBinLabels = [NeuronLabels; ConnectomeSymBin];

writematrix(ConnectomeSymBinLabels,'StrucConnectomeSymBinLabels.csv');
writematrix(NeuronLabels, 'NeuronLabels.csv');

Q = 0;
for i = 1:20000
    [Wi, Qi] = community_louvain(ConnectomeSymBin);
    if Qi > Q
        W = Wi;
    end
end

ConnectomeSymBinCom = [W.'; NeuronLabels; ConnectomeSymBin] ;

[ConnectomeSymBinComSort, index] = sortrows(ConnectomeSymBinCom.', [1,2]);

ConnectomeSymBinSort = ConnectomeSymBin(index,index);

D = diff(sortrows(W));
S = find(D>0); % index where each group begins
S = S - 0.5; %centering the ticks

L = size(W)
T = [0;S;L(1)] % grid for the labels
U = zeros(5,1);
for i = 1:5
    U(i) = (T(i) + T(i+1))/2
end

spy(ConnectomeSymBinSort,'k',2)
xlabel('')
grid on
ax = gca;
xticks(S)
yticks(S)
xticklabels([])
yticklabels([])
ax.GridColor = '#7E2F8E'
ax.GridLineStyle = '-';
ax.GridAlpha = 0.5;
ax.Layer = 'top';
ax.LineWidth = 1;
i = 0
grouplabels = cellstr(string(1:5));
text(zeros(1,5) - 10, U, grouplabels,'Color','black','FontSize',18,'HorizontalAlignment', 'right');
text(U - 10, zeros(1,5) + 315, grouplabels,'Color','black','FontSize',18);
```

MATLAB Spectral Algorithm Script. This script performs 20,000 iterations of the Spectral algorithm, takes the best one and produces a plot.

```
GapJunctionsSym = GapJunctions - diag(diag(GapJunctions));
GapJunctionsSymBin = GapJunctionsSym ~= 0;

ConnectomeSymBin = GapJunctionsSymBin + ChemSynapsBinSym;
ConnectomeSymBin = ConnectomeSymBin ~= 0;

ConnectomeSymBin = double(ConnectomeSymBin);

Q = 0;
for i = 1:20000
    [Wi, Qi] = modularity_und(ConnectomeSymBin);
    if Qi > Q
        W = Wi;
    end
end

%relabelling so my groups match between Spectral and Louvain

for i = 1:300
    if W(i) == 1
        W(i) = 4;
    elseif W(i) == 2
```

```
        W(i) = 3;
    elseif W(i) == 3
        W(i) = 1;
    elseif W(i) == 4
        W(i) = 2;
    end
end

ConnectomeSymBinCom = [W.'; NeuronLabels; ConnectomeSymBin] ;

[ConnectomeSymBinComSort, index] = sortrows(ConnectomeSymBinCom.',[1,2]);

ConnectomeSymBinSort = ConnectomeSymBin(index,index);

D = diff(sortrows(W));
S = find(D>0); % index where each group begins
S = S - 0.5; %centering the ticks

L = size(W);
T = [0;S;L(1)]; % grid for the labels
U = zeros(5,1);
for i = 1:5
    U(i) = (T(i) + T(i+1))/2;
end

spy(ConnectomeSymBinSort,'k',2)
xlabel('')
grid on
ax = gca;
xticks(S)
yticks(S)
xticklabels([])
yticklabels([])
ax.GridColor = '#7E2F8E';
ax.GridLineStyle = '-';
ax.GridAlpha = 0.5;
ax.Layer = 'top';
ax.LineWidth = 1;
grouplabels = cellstr(string(1:5));
text(zeros(1,5) - 10, U, grouplabels,'Color','black','FontSize',18,'HorizontalAlignment', 'right');
text(U - 10, zeros(1,5) + 315, grouplabels,'Color','black','FontSize',18);
```

Tikz script for producing the spring diagram. This script is executable with the LuaLaTeX compiler to generate spring plots visualising the group sizes and connection probabilities.

```
\RequirePackage{luatex85}
\documentclass[tikz,border=5]{standalone}
\usetikzlibrary{graphs,graphdrawing}
\usegdlibrary{force,trees}
\usepackage{luacode}
\begin{luacode*}

linewidthparameter = 0.1

function parseNetwork(nodeWeights, adjacencyMatrix)
  local i, j, n, v, w, str
  n = 0
  weights = {}
  for str in string.gmatch(nodeWeights, "[^%s]+") do
    w = tonumber(str)
        tex.print("n" .. n + 1 .. "/[minimum size=" .. w .. ", label=above:Group " .. n + 1 .. ", label=center: " .. w
             .." ];")
      n = n + 1
  end
  i = 0
  j = 0
  for str in string.gmatch(adjacencyMatrix, "[^%s]+") do
    v = tonumber(str)
      if v > 0 then
        if j > i then
        tex.print("n" .. i + 1 .. " --[line width=" .. (v / linewidthparameter) .. "]" .. "n" .. j + 1 .. ";")
        end
      end
        j = (j + 1) % n
```

```
        if j == 0 then i = i + 1 end
  end
end
\end{luacode*}
\tikzgraphsset{%
  node weights/.store in=\nodeweights,
  adjacency matrix/.store in=\adjacencymatrix,
  declare={network}{
  [/utils/exec={%
    \edef\networkspec{\directlua{parseNetwork("\nodeweights","\adjacencymatrix")}}},
    parse/.expanded=\networkspec]
}}
\begin{document}
\begin{tikzpicture}
\graph [spring electrical layout,
  edges={draw=blue!40!cyan, shorten >=-1em, shorten <=-1em},
  nodes={circle, fill=blue!30!cyan!50},
  electric charge=15]{
    network [
      node weights={
    20   26   29   25   32   28   42   18   31   37   12
    },
      adjacency matrix={
   0.58 0    0    0    0    0    0    0    0    0    0
   0    0.38 0    0.21 0    0    0    0    0    0    0
   0    0    0.22 0    0    0    0    0    0    0    0.29
   0    0.21 0    0.36 0.1  0    0    0.18 0    0.13 0.33
   0    0    0    0.1  0.21 0    0    0.10 0    0.15 0.26
   0    0    0    0    0    0.29 0    0.25 0    0    0.29
   0    0    0    0    0    0    0.24 0.09 0    0    0.33
   0    0    0    0.18 0.10 0.25 0.09 0.58 0    0    0.46
   0    0    0    0    0    0    0    0    0.26 0.16 0.08
   0    0    0    0.13 0.15 0    0    0    0.16 0.39 0.16
   0    0    0.29 0.33 0.26 0.29 0.33 0.46 0.08 0.16 0.83
}];
};
\end{tikzpicture}
\end{document}

% https://tex.stackexchange.com/questions/362451/how-to-produce-this-matlab-social-network-graph-from-an-adjacency-
    matrix-in-tikz This code modified from StackExchange and used with permission under the MIT license
```