Statistical modeling of visual cortical neurons

Cristian Bargiacchi, Christian Mancini

Preliminary steps

```r
library(igraph)
library(ergm)
```

We first need to load our data using `igraph`. Data can be found here

```r
neurons_g <- read_graph("Data/mouse_visual.cortex_2.graphml","graphml")
Y = as_adjacency_matrix(neurons_g, sparse = F)
diag(Y) = NA
```

For starting the modeling we first have to convert an igraph object to a network one.

The conversion retains the order of the nodes but we also have to pass the attributes.

```r
neurons = network(Y, directed = T)
neurons %v% "type1" = vertex_attr(neurons_g,"type1",V(neurons_g))
neurons %v% "type2" = vertex_attr(neurons_g, "type2",V(neurons_g))
```

Now we are good to go!

Homogeneous Simple Random Graph

Let's start with the simplest model.

Assumptions:

- The probability of forming a tie is the **same** for every pair.

```r
srg_homo = ergm(neurons ~ edges)
```

```r
summary(srg_homo)
```

```
Call:
ergm(formula = neurons ~ edges)

Maximum Likelihood Results:

      Estimate Std. Error MCMC % z value Pr(>|z|)
edges -5.16921    0.06854      0  -75.42   <1e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 52444  on 37830  degrees of freedom
 Residual Deviance:  2642  on 37829  degrees of freedom

AIC: 2644  BIC: 2652  (Smaller is better. MC Std. Err. = 0)
```

This model corresponds to a logistic regression, so we can interpret the result as odds:

> The odds of observing a relation between two randomly selected nodes is about 99.43% lower than that of not observing it.

Non-Homogeneous Simple Random Graph

Assumptions:

- The same probability of forming a tie is relaxed.
- Takes in consideration sender and receiver effect

> Since some node do not have in or out degree, those parameter will be set to `-inf`, so the model can't be used, but it can be extimeted.

```
srg_no_homo = ergm(neurons ~ edges + sender + receiver,
                   control = control)
```

Dyad independence model

Assumptions:

- Dyads are independents and follows a *Multinomial* distribution
- We take in consideration the reciprocity parameter γ_{ij} (*mutual*)

> Since some estimations took to long we switched to a *Stochastic Approximation*.

Classic p1 model

Assumptions:

- the reciprocity parameter is $\gamma = \gamma_{ij}$
- μ_{ij} depends additively from on the sender and receiver effect of node i and j involved.

```
p1_classic = ergm(neurons ~ edges + sender + receiver + mutual,
                  control = control.ergm(seed = 1, main.method = "Stochastic-Approximation"))
```

Sender and reciver independency assumption

We now construct 3 p1 model with the following assumptions:

1. Sender effect independent
2. Receiver effect independent (can't be estimated)
3. Sender and receiver effect independent

```
p1_sender_ind = ergm(neurons ~ edges + receiver + mutual,
                     control = control.ergm(seed = 1, main.method = "Stochastic-Approximation"))
```

```
Warning in mple.existence(pl): The MPLE does not exist!
```

```
Warning: Approximate Hessian matrix is singular. Standard errors due to MCMC approximation of the
likelihood cannot be evaluated. This is likely due to insufficient MCMC sample size or highly correlate
model terms.
```

> This model below can **not** be estimated

```
p1_receiver_ind = ergm(neurons ~ edges + sender + mutual,
                       control = control.ergm(seed = 1, main.method = "Stochastic-Approximation"))
```

```r
p1_mutual_only = ergm(neurons ~ edges + mutual,
                      control = control.ergm(seed = 1))
```

```r
summary(p1_mutual_only)
```

```
Call:
ergm(formula = neurons ~ edges + mutual, control = control.ergm(seed = 1))

Monte Carlo Maximum Likelihood Results:

       Estimate Std. Error MCMC % z value Pr(>|z|)
edges  -5.16710    0.06824      0  -75.72   <1e-04 ***
mutual     -Inf    0.00000      0    -Inf   <1e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


 Warning: The following terms have infinite coefficient estimates:
  mutual
```

The following is a small table that compares BIC of usable p1 models.

```r
print(BIC(p1_classic,p1_sender_ind))
```

Nodal attributes

In this part of the analysis we include nodal attributes to explore *homophily* and *main* effects.

- **Main effect**: Nodes of a specific type have more chance to form ties
- **Homophily effect** Nodes of a specific type have more chance to form ties between nodes of the same type

  As saw in the descriptive analysis we expect to observe dissortative mixing.

Since we only have categorical attributes we will use:

- `nodefactor()` to include main effect
- `nodematch()` to include homophily effect

Let's explore "*type1*" nodal attribute.

```r
main_homo_type_one = ergm(neurons ~ edges + nodefactor("type1") + nodematch("type1"),
            control = control.ergm(seed = 1))
```

```r
summary(main_homo_type_one)
```

```
Call:
ergm(formula = neurons ~ edges + nodefactor("type1") + nodematch("type1"),
    control = control.ergm(seed = 1))

Maximum Likelihood Results:

                                               Estimate Std. Error MCMC % z value Pr(>|z|)
edges                                          -7.24825    0.37136      0 -19.518   <1e-04 ***
nodefactor.type1.Characterized pyramidal neuron  4.12940    0.33609      0  12.287   <1e-04 ***
nodefactor.type1.Dendritic fragment            -0.01284    0.16781      0  -0.077    0.939
nodematch.type1                                -4.10482    0.51257      0  -8.008   <1e-04 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 52444  on 37830  degrees of freedom
 Residual Deviance:  1850  on 37826  degrees of freedom

AIC: 1858  BIC: 1892  (Smaller is better. MC Std. Err. = 0)
```

As we expect, the *Dissortative mixing* is captured and "*Characterized pyramidal neuron*" have more chances to form a ties (which is true because they start the synapses). More precisely, the probability is 62.14 higher respect to a "*Cell body in EM volume*".

Now we use *"type2"* attribute, but only as main effect, and we remove `mutual` since it is estimated as `-inf`.

```r
main_type_two = ergm(neurons ~ edges + nodefactor("type2"),
            control = control.ergm(seed = 1))
```

```r
summary(main_type_two)
```

```
Call:
ergm(formula = neurons ~ edges + nodefactor("type2"), control = control.ergm(seed = 1))

Maximum Likelihood Results:

                                                Estimate Std. Error MCMC % z value Pr(>|z|)
edges                                            -0.9258     0.1354      0  -6.836   <1e-04 ***
nodefactor.type2.Postsynaptic excitatory target  -2.9271     0.1270      0 -23.047   <1e-04 ***
nodefactor.type2.Postsynaptic inhibitory target  -2.6742     0.1349      0 -19.824   <1e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 52444  on 37830  degrees of freedom
 Residual Deviance:  2016  on 37827  degrees of freedom

AIC: 2022  BIC: 2048  (Smaller is better. MC Std. Err. = 0)
```

We basically got the same conclusions. Since the reference is "*NA*" which can only be "*Characterized pyramidal neuron*", the odds of form a tie are smaller if a node is "*excitatory*" or "*inhibitory*".

Let's now put all together:

```r
main_homo = ergm(neurons ~ edges + nodefactor("type1") + nodefactor("type2")
                 + nodematch("type1"),
            control = control.ergm(seed = 1))
```

and compare BIC for these models including nodal attributes.

Model	BIC
Main Homo Type One	1891.98
Main Type Two	2047.53
Main Homo	1890.25

According to `BIC` the full model is better.

Nodal attributes (Binary)

Let's repeat the above analysis but using new attributes which are the binarization of the real ones.

As reference category we choose "*Characterized pyramidal neuron*" and "*Postsynaptic excitatory target*".

Generate the new attributes:

```
type1.new = rep(0, 195)
type1.new[vertex_attr(neurons_g,"type1") == "Characterized pyramidal neuron"] = 1
type1.new
neurons %v% "type1.new" = type1.new

type2.new = rep(0, 195)
type2.new[vertex_attr(neurons_g,"type2") == "Postsynaptic excitatory target"] = 1
type2.new
neurons %v% "type2.new" = type2.new
```

Estimate all the models again:

```
main_homo_type_one_binary = ergm(neurons ~ edges + nodefactor("type1.new")
                                 + nodematch("type1.new"),
            control = control.ergm(seed = 1))

main_type_two_binary = ergm(neurons ~ edges + nodefactor("type2.new"),
            control = control.ergm(seed = 1))

main_homo_binary = ergm(neurons ~ edges + nodefactor("type1.new")
                        + nodefactor("type2.new") + nodematch("type1.new"),
            control = control.ergm(seed = 1))
```

Model	BIC
main_homo_type_one_binary	1853.40
main_type_two_binary	2502.18
main_homo_binary	1858.36

According to `BIC` the new model with just "*type1.new*" is the best for now with a score of 1853.40.

Let's explore the summary of the model

```
summary(main_homo_type_one_binary)
```

```
Call:
ergm(formula = neurons ~ edges + nodefactor("type1.new") + nodematch("type1.new"),
    control = control.ergm(seed = 1))

Maximum Likelihood Results:

                       Estimate Std. Error MCMC % z value Pr(>|z|)
edges                   -5.5203     0.3742      0 -14.752   <1e-04 ***
nodefactor.type1.new.1   2.4305     0.3674      0   6.616   <1e-04 ***
nodematch.type1.new     -3.2727     0.3742      0  -8.745   <1e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 52444  on 37830  degrees of freedom
```

```
 Residual Deviance:  1822  on 37827  degrees of freedom

AIC: 1828  BIC: 1853  (Smaller is better. MC Std. Err. = 0)
```

As we can see every parameter is significant. The interpretation is the same as before.