Stat4DS / Homework 02

Pierpaolo Brutti

Due Thursday, December 19, 2019, 23:59 PM on Moodle

General Instructions

I expect you to upload your solutions on Moodle as a **single running** `R Markdown` file (`.rmd`) + its `html` output, named with your surnames.

R Markdown Test

To be sure that everything is working fine, start `RStudio` and create an empty project called `HW1`. Now open a new `R Markdown` file (`File > New File > R Markdown...`); set the output to `HTML mode`, press `OK` and then click on `Knit HTML`. This should produce a web page with the knitting procedure executing the default code blocks. You can now start editing this file to produce your homework submission.

Please Notice

- For more info on `R Markdown`, check the support webpage that explains the main steps and ingredients: R Markdown from RStudio.
- For more info on how to write math formulas in LaTex: Wikibooks.
- Remember our **policy on collaboration**: *Collaboration on homework assignments with fellow students is* ***encouraged****. However, such collaboration should be clearly acknowledged, by listing the names of the students with whom you have had discussions concerning your solution. You may* ***not****, however, share written work or code after discussing a problem with others. The solutions should be written by* ***you****.*

Exercise: Connect your brain

1. Background: MRI and fMRI

Since its invention in the early 70s by Lauterbur and Mansfield (2003 Nobel prize in Physiology and Medicine), **magnetic resonance imaging** (MRI) has evolved into a versatile tool for the in vivo examination of tissue. Unlike X-ray computed tomography (CT) and positron emission tomography (PET), it does not rely on high energetic radiation but on the nuclear magnetic resonance phenomenon. Consequently, it does in principle not harm the examined tissue and can be applied also in healthy subjects. Thus, MRI is a perfect tool for the examination of the **living brain** in neuroimaging.

Functional magnetic resonance imaging (fMRI) is a technique to examine the human (or animal) brain "at work". fMRI is used to analyze (neuro-)scientific questions, e.g., on the localization of neural capabilities, on the consequences of neuronal diseases on brain function or to study the **functional connectivity** of the brain. For this, in fMRI, a **time series** of MRI volumes is acquired, while the subject in the scanner is typically performing some cognitive task.

What fMRI images visualize is the so called blood oxygenation level–dependent (BOLD) contrast: as active neurons rely on increased oxygen supply, the neural activity is related to a local change in support of blood oxygenation. Thus, fMRI can be used as a natural, yet indirect, contrast for detecting neural activity. In order to achieve a sufficient temporal resolution the spatial resolution of fMRI is typically limited. An fMRI dataset then consists of more than 100 image volumes with a spatial voxel dimension of about 2-4 mm.

Data from fMRI experiments suffer from several artifacts of different origins that require special preprocessing ahead of the statistical analysis like *slice time correction*, *motion correction*, *registration*, *normalization*, *brain masking* and *brain tissue segmentation*. For the sake of this exercise, I'll provide you with a clean, pre-processed dataset, but be aware that these early data analytic stages are crucial and not at all trivial.

2. The Task & The Data: Functional Connectivity for Resting State fMRI

Functional connectivity addresses the interaction between cortical brain regions, and it is usually quantified by measuring the level of dependency between the observed resting state time series in these regions.

The raw data were recorded within a longitudinal study for long-term neural and physiological phenotyping (Poldrack et al. 2015). It consists of recordings of a single human subject in 107 sessions over a period of 532 days. Here we will only use the resting state fMRI data recorded in session 105 and including ten runs of 10 min duration with closed eye. The data are available from the openNeuro archive and can be downloaded from Amazon Web Services S3 storage using the AWS CLI tool.

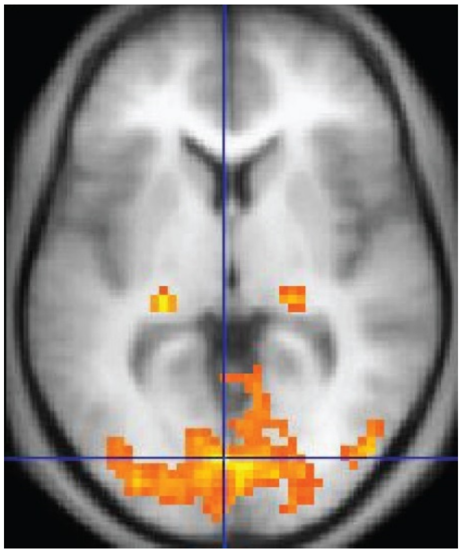
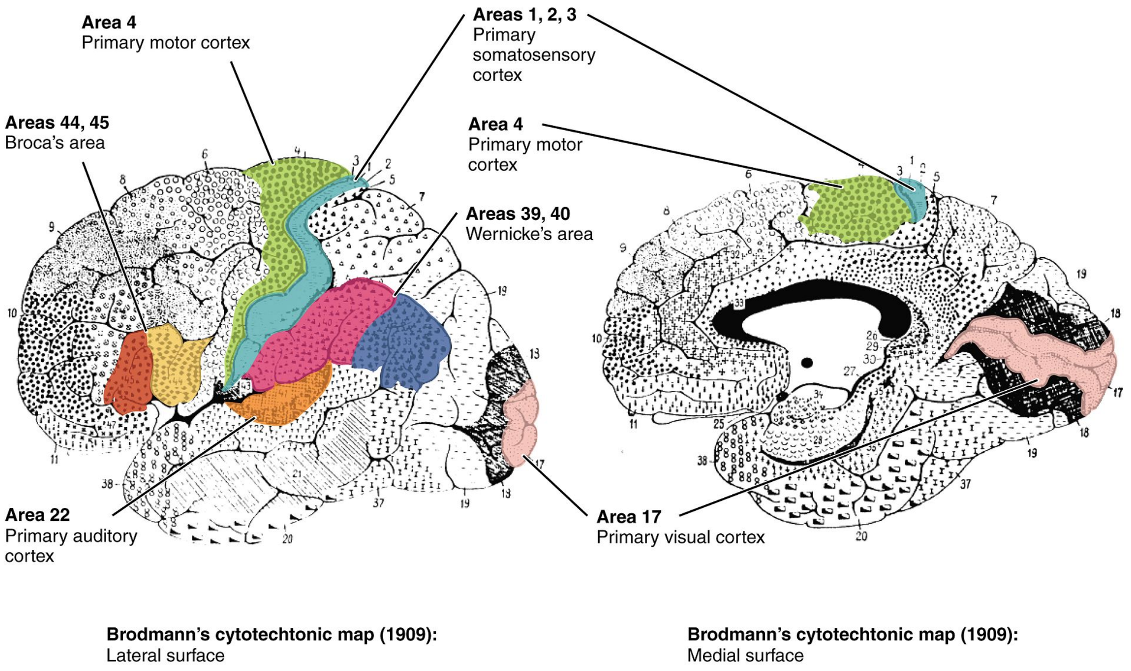


Figure 1: An fMRI image with yellow areas showing increased activity compared with a control condition

The other important piece of the data–analytic puzzle we are facing comes from the so called **functional brain atlas**: they provide information about the spatial location of functional brain regions aggregating knowledge on brain functionality and anatomy accumulated over more that 100 year of brain research. We use them essentially to tag fMRI voxels with specific cortical brain regions. The oldest atlas system dates back to the German anatomist Brodmann who defined 52 cortical areas based on the cytoarchitectural organization of neurons.



This is all nice and good, but to attach an observed fMRI voxels to a specific area of your functional atlas of choice we first need to *normalize* each individual brain or, in other words, we need to map it onto a "standard brain" in order to then be able to identify the corresponding brain regions. Talairach coordinates, also known as *Talairach space*, is one famous 3-dimensional coordinate system (atlas) that uses Brodmann areas as the labels for brain regions.

3. The Toolkit: Association Graphs

As already mentioned, the problem is one of evaluating dependecy between cortical regions: time to put the **association graph** machinery introduced in class to good use.

Let $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ be random vector IID from some joint distribution $f_{\boldsymbol{X}}(\cdot)$ where $\boldsymbol{X}_i = \left[X_i(1), \ldots, X_i(\mathrm{D})\right]^{\mathsf{T}} \in \mathbb{R}^{\mathrm{D}}$.

PLEASE NOTICE:

1. In our application D will be the number of cortical regions of interest from the functional atlas of choice.
2. n will be equal to the length of the recorded time series: yes, for the sake of this exercise we will drop the time dependecy and treat them as IID data.

The **vertices** (nodes) of the graph refer to the D features: each node corresponds to a single feature. **Edges** instead represent *relationships* between the features.

The graph is represented by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{V_1, ..., V_{\mathrm{D}}\}$ is the vertex–set and $\mathcal{E}$ the edge–set. We can regard the edge–set $\mathcal{E}$ as a $(\mathrm{D} \times \mathrm{D})$ matrix $\mathbf{E}$ where $\mathbf{E}(j,k) = 1$ if there is an edge between feature j and feature k and 0 otherwise. Alternatively, you can regard $\mathcal{E}$ as a list of *unordered* pairs where $\{j,k\} \in \mathcal{E}$ if there is an edge between j and k.

In an **association graph** we put an edge between V_j and V_k if

$$\big|\,\rho(j,k)\,\big| \geqslant \epsilon,$$

where $\rho(j,k)$ is *some* measure of *association* like *Pearson's correlation*, *Kendall's* τ, *partial correlation*, etc. Often we set $\epsilon = 0$ in which case there is an edge if and only if $\rho(j,k) \neq 0$. We also write $\rho_{j,k}$ or $\rho(X_j, X_k)$ to mean the same as $\rho(j,k)$.

The parameter $\rho(j,k)$ is required to have the following property:

$$X \perp Y \quad \Rightarrow \quad \rho(X,Y) = 0.$$

In general, the reverse may **not** be true. We will say that ρ is *strong* if

$$X \perp Y \quad \Leftrightarrow \quad \rho(X,Y) = 0.$$

Let's mention, for example, the recent work by Bergsma and Dassios (2014) – TauStar package – who extended the Kendall's τ into a *strong* correlation, and also the now well–known distance covariance – energy package – defined in Szekely et al. (2007).

PLEASE NOTICE:
In practice, if we have a confidence interval $\mathrm{C}_{n,\alpha}$ then we can put an edge whenever $[-\epsilon, +\epsilon] \cap \mathrm{C}_{n,\alpha} = \emptyset$.

For the **Pearson correlation**, for example, we know we can build these intervals via nonparametric bootstrap or going for the asymptotic Fisher Z-transform: of course the bootstrap is a viable alternative for many other association measures.

This idea is perfectly fine in case we have only a small number of edges/cortical areas to check. In general though, the graphs are characterized by a large number of nodes and, consequently, a huge amount of edges: $m = \binom{\mathrm{D}}{2}$ to be precise.

TAKE HOME MESSAGE:
We necessarily need to control for multiplicity in order to avoid a ridiculous overflow of false edges!

The easiest procedure to implement – although quite conservative – is the so called Bonferroni correction that simply asks for adjusting the nominal level of the intervals from α to α/m where $m = \binom{\mathrm{D}}{2}$ is the number of intervals we are building.

But, a **simultaneous** confidence set for **all** the correlations at once can be obtained using the **bootstrap** (Wasserman et al., 2013). This is especially important if we want to put an edge when $\big|\rho(j,k)\big| \geqslant \epsilon$.

SIMULTANEOUS BOOTSTRAPPED CI'S FOR A GENERIC ASSOCIATION MEASURE ρ

1. Let $\mathbf{R}$ be the $(\mathrm{D} \times \mathrm{D})$ matrix of **true** correlations and let $\widehat{\mathbf{R}}$ be the $(\mathrm{D} \times \mathrm{D})$ matrix of plug-in estimates.
2. Now let $\{\boldsymbol{X}_1^\star, \dots, \boldsymbol{X}_n^\star\}$ be a **bootstrap** sample and $\widehat{\mathbf{R}}^\star$ be the $(\mathrm{D} \times \mathrm{D})$ matrix of bootstrapped correlations.
3. After taking B bootstrap samples we have $\{\widehat{\mathbf{R}}_1^\star, \dots, \widehat{\mathbf{R}}_B^\star\}$.
4. For each bootstrap sample $b \in \{1, \dots, B\}$, define the following bootstrapped replicate of a **simultaneous** statistics

$$\Delta_b = \sqrt{n} \max_{j,k} \Big|\, \widehat{\mathbf{R}}_b^\star[j,k] - \widehat{\mathbf{R}}[j,k] \,\Big|,$$

with its associated boostrapped ECDF:

$$\widehat{F}_n(t) = \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}\big(\Delta_b \leqslant t\big).$$

Within the usual bootstrap analogy, for large n and B, $\widehat{F}_n(t)$ should be a good approximation to

$$F_n(t) = \mathbb{P}\Big(\sqrt{n}\,\max_{j,k}\Big|\widehat{\mathbf{R}}[j,k] - \mathbf{R}[j,k]\Big| \leqslant t\Big).$$

5. Finally, to build our simultaneous confidence set, consider the sample quantile at level $1-\alpha$ of the bootstrapped distribution $\widehat{F}_n(t)$, say $t_\alpha = \widehat{F}_n^{-1}(1-\alpha)$, and set

$$\mathrm{C}_{j,k}(\alpha) = \left[\widehat{\mathbf{R}}[j,k] - \frac{t_\alpha}{\sqrt{n}}, \widehat{\mathbf{R}}[j,k] + \frac{t_\alpha}{\sqrt{n}}\right].$$

Property: $\mathbb{P}\big(\mathbf{R}[j,k] \in \mathrm{C}_{j,k}(\alpha)$ **for all** $(j,k)\big) \overset{n}{\to} 1-\alpha$

As an alternative, **partial correlations** can be considered. In our setup, they effectively measure the association of two resting state series with the (linear) effect of series in all other regions **removed**. As highlighed in class, partial correlations can be effectively computed from inverse covariance or precision matrices.

Now let's go back to graphs. Let $\boldsymbol{X} = \big[X(1), \dots, X(\mathrm{D})\big]^{\mathsf{T}}$ and let $\rho^{(p)}_{j,k}$ denote this time the partial correlation between $X(j)$ and $X(k)$ **given all** the other variables. Let $\mathbf{R}_{(p)} = \big[\,\rho^{(p)}_{j,k}\,\big]_{j,k}$ be the $(\mathrm{D}\times\mathrm{D})$ matrix of partial correlations. Then we have:

Result. *The matrix* $\mathbf{R}_{(p)}$ *is given by*

$$\mathbf{R}_{(p)}[j,k] = -\frac{\Lambda_{j,k}}{\sqrt{\Lambda_{j,j}\cdot\Lambda_{k,k}}},$$

where $\Lambda = \Sigma^{-1}$ *is the precision matrix associated to the covariance matrix* Σ.

The partical correlation graph $\mathcal{G}$ has an edge between j and k when $\rho^{(p)}_{j,k} \neq 0$.

In the **low–dimensional** setting ($\mathrm{D} << n$), we can estimate $\mathbf{R}_{(p)}$ as follows. Let $\widehat{\Sigma}_n$ be the sample covariance, and $\widehat{\Lambda} = \widehat{\Sigma}_n^{-1}$ its inverse. Then define

$$\widehat{\mathbf{R}}_{(p)}[j,k] = \widehat{\rho}^{(p)}_{j,k} = -\frac{\widehat{\Lambda}_{j,k}}{\sqrt{\widehat{\Lambda}_{j,j}\cdot\widehat{\Lambda}_{k,k}}}.$$

The easiest and reliable way to construct the graph is then to get **simultaneous** confidence intervals $\mathrm{C}_{j,k}(\alpha)$ using **boostrap** (again), and put in an edge $\{j,k\}$ if $0 \notin \mathrm{C}_{j,k}(\alpha)$.

There is also a *Normal approximation* similar to linear correlations. Define

$$Z_{j,k} = h\big(\widehat{\rho}^{(p)}_{j,k}\big) = \frac{1}{2}\log\left(\frac{1+\widehat{\rho}^{(p)}_{j,k}}{1-\widehat{\rho}^{(p)}_{j,k}}\right) \quad \text{then} \quad Z_{j,k} \mathrel{\dot\sim} \mathrm{N}_1\left(\theta_{j,k}, \frac{1}{n-g-3}\right)$$

where $\theta_{j,k} = h\big(\rho^{(p)}_{j,k}\big)$ and $g = (\mathrm{D}-2)$.

As before, via a Bonferroni correction we put an edge between node j and node k if $|Z_{j,k}| > z_{\frac{\alpha}{2m}} \cdot \frac{1}{\sqrt{n-g-3}}$ where $m = \binom{\mathrm{D}}{2}$.

An implementation of this method is provided by the function `sinUG()` in the package `R` package `SIN`. The only difference is that, instead of Bonferroni, it adopts a finer correction for multiplicity.

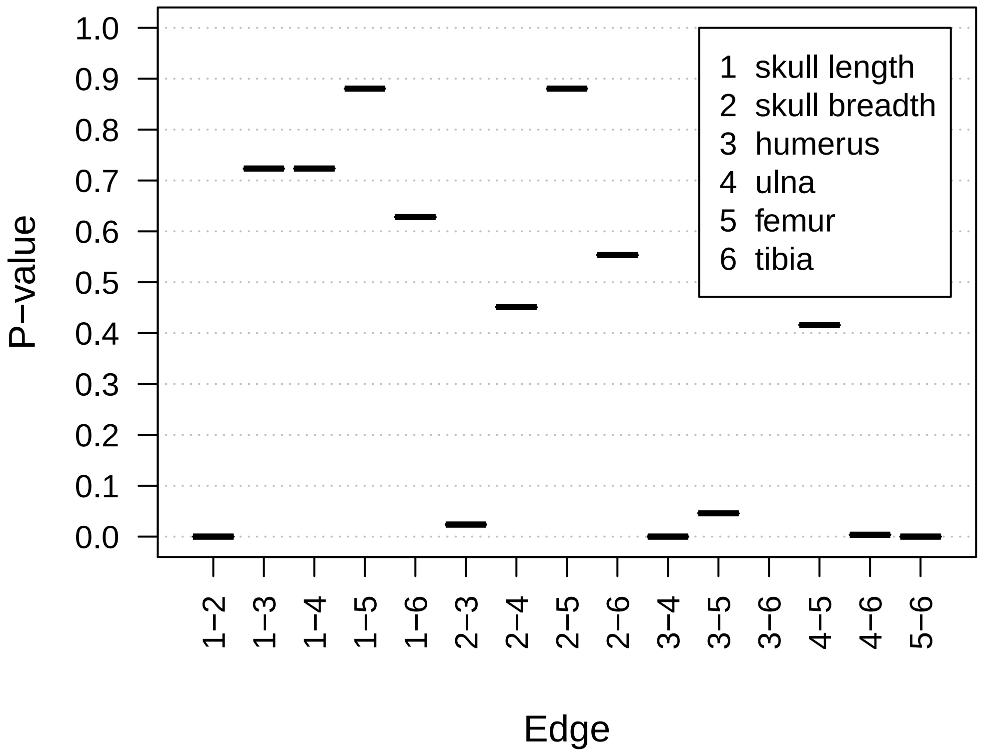
```r
# Package
library(SIN); library(help = SIN); ?sinUG

# Some data: are there dependencies between biometric measurements?
data(fowlbones)
?fowlbones

# Run & take a look
out <- sinUG(fowlbones$corr,fowlbones$n)
round(out,3)          # estimated partial correlation matrix
```

```
##                skull length skull breadth humerus  ulna femur tibia
## skull length             NA         0.000   0.723 0.723 0.880 0.628
## skull breadth         0.000            NA   0.024 0.451 0.880 0.553
## humerus               0.723         0.024      NA 0.000 0.046 0.723
## ulna                  0.723         0.451   0.000    NA 0.416 0.004
## femur                 0.880         0.880   0.046 0.416    NA 0.000
## tibia                 0.628         0.553   0.723 0.004 0.000    NA
```

```
plotUGpvalues(out)    # take a look at the p-values
```



```
# Get the graph associated to a specific confidence level
library(igraph)

par(mfrow = c(1,2))
alpha = 0.05
E.SIN = getgraph(out, alpha)
print(E.SIN)
```

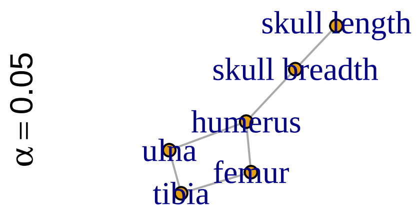
```
##                skull length skull breadth humerus ulna femur tibia
## skull length              0             1       0    0     0     0
## skull breadth             1             0       1    0     0     0
## humerus                   0             1       0    1     1     0
## ulna                      0             0       1    0     0     1
## femur                     0             0       1    0     0     1
## tibia                     0             0       0    1     1     0
```

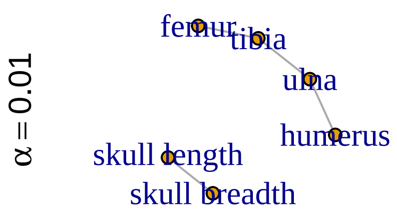
```
G.SIN = graph.adjacency(E.SIN, mode = "undirected")
plot(G.SIN, ylab = expression(alpha == 0.05))

alpha = 0.01
E.SIN = getgraph(out, alpha)
print(E.SIN)
```

```
##                skull length skull breadth humerus ulna femur tibia
## skull length              0             1       0    0     0     0
## skull breadth             1             0       0    0     0     0
## humerus                   0             0       0    1     0     0
## ulna                      0             0       1    0     0     1
## femur                     0             0       0    0     0     1
## tibia                     0             0       0    1     1     0
```

```
G.SIN = graph.adjacency(E.SIN, mode = "undirected")
plot(G.SIN, ylab = expression(alpha == 0.01))
```





PLEASE NOTICE:

In high–dimensions, this protocol will **not** work since $\widehat{\Sigma}_n$ is **not** invertible. In fact one can show that

$$\mathbb{V}\text{ar}\left(\widehat{\rho}^{(p)}_{j,k}\right) \approx \frac{1}{n - \mathrm{D}},$$

and this blows up when D is close to, or even larger than, n.

In such a bad situation, we have at least two exit strategies:

1. Apply some **shrinkage** (statistical lingo) or **regularization** (more math inclined) scheme to "fix" the ill–conditioned matrix $\widehat{\Sigma}_n$ and make it non–singular + bootstrap on the entries of the resulting matrix. See, for example, Schager and Strimmer (2005) and Ledoit and Wolf (2004).
2. Use the *graphical lasso* – `glasso` package.
   Warning! The reliability of the graphical lasso depends on lots of non-trivial, uncheckable assumptions.

⇝ Your job ⇜

1. Take a look at basic tools to deal with graphs in `R` such as the `igraph` and `ggraph` packages.
2. Load the pre–processed data matrix $\mathbb{X} = \left[x_{t,j}\right]_{t,j}$ contained in the file `hw2_data.RData`. The resulting object named `mts` is a (240×81) numerical matrix. The 81 columns are related to different Brodmann cortical areas labeled with an integer plus a prefix `L` or `R` depending on the hemisphere they belong. The rows instead are the observation times (again, here we will drop the temporal dependency). Notice that, for each cortical area, the time series we are working on is obtained by averaging those associated to voxels belonging to the same Brodmann area.
3. With this data, consider any association measure you want (but partial correlation, see the last point), and implement the **bootstrap procedure** described in the box above entitled "SIMULTANEOUS BOOTSTRAPPED CI'S FOR A GENERIC ASSOCIATION MEASURE ρ".
4. Graphically represent the estimated graph but try to: 1. visualize its dynamic as ϵ varies, 2. visualize the strengh of the dependency adopting a suitable color-scale for the edges of the graph.
   Draw some conclusion: what are the areas that show the highest/lowest degree of connectivity?
5. Repeat the analysis using this time the linear partial correlation coefficient as implemented in `SIN`.
   Compare the results...even better if "visually"...