Estimating and simulating multiple related functional connectivity networks via the `MNS` Package

Ricardo Pio Monti, Christoforos Anagnostopoulos and Giovanni Montana

December 7, 2015

Abstract

In many neuroimaging studies functional connectivity measurements are collected across a cohort of multiple subjects. Given such data, a fundamental problem corresponds to accurately quantifying variability across subjects. In this vignette we provide a brief illustration of the recently proposed Mixed Neighborhood Selection (MNS) algorithm which is able to simultaneously estimate connectivity networks at the population and subject level as well as quantify inter-subject variability. The `MNS` package includes parallel implementations the MNS algorithm as well as cross-validation functions; thereby providing computationally efficient methods through which to select regularization parameters and perform model estimation.

Moreover, this vignette also introduces an algorithm from which to simulate functional connectivity networks for a cohort of related individuals. It is well documented that functional connectivity displays reproducible activation patterns across subjects while simultaneously exhibiting high inter-subject variability. To our surprise, we found there to be limited algorithms through which to simulate connectivity networks for a cohort of subjects which display these widely accepted properties. To address this, we present a simple and efficient method through which to simulate functional connectivity networks for multiple subjects.

1 Introduction

A cornerstone in the understanding brain connectivity is the notion that connectivity can be represented as a graph or network composed of a set of nodes interconnected by a set of edges [Bullmore and Sporns, 2009]. In the context of functional connectivity, edges represent statistical dependencies across spatially remote brain regions [Friston, 2011]. Understanding and quantifying variability in functional connectivity is a fundamental problem in modern neuroscience [Kelly et al., 2012, Mueller et al., 2013]. Standard approaches within the neuroimaging literature tend to either ignore this variability by estimating a single network across all subjects or proceed too cautiously by estimating a network for each subject independently[1]. The former strategy makes implicit assumptions regarding the exchangeability of the data which are difficult to justify in practice. Conversely, the latter approach does not exploit the shared network structure across subjects leading to detrimental effects on the quality of the estimated networks.

In order to partially address this issue, the mixed neighborhood selection (MNS) algorithm was recently proposed [Monti et al., 2015a]. Briefly, the MNS algorithm looks to decompose the network structure into two classes: reproducible edges which are present across the majority of the cohort and edges which represent subject-specific idiosyncrasies. It follows that the latter captures the inter-subject variation.

In order to empirically validate the proposed MNS algorithm we require a method through which to produce synthetic data where the true underlying covariance structure is known. Moreover, it would be desirable for such data to showcase the many hallmarks of functional imaging data. To our surprise, we found that while there is a wide array of algorithms for simulating connectivity on a subject-specific level, limited attention has been given to generating simulated data for a cohort of related subjects. As a result, in this vignette we present and implement a novel algorithm through which to simulate realistic functional connectivity networks for a cohort of subjects.

The objective of this vignette is two-fold. First, we introduce the aforementioned algorithm through which to simulate functional connectivity across a cohort of subjects. The proposed algorithm was designed to display several of the well-document properties of functional connectivity networks as well as properties reported during an exploratory data analysis of the ABIDE data set [Di Martino et al., 2014]. This is presented in Section 2. Second, we introduce the MNS algorithm together with the corresponding `MNS` package. We highlight the strengths of the MNS algorithm through the use of examples with simulated data.

2 Simulating functional connectivity networks for a cohort of subjects

There is a wealth of literature and algorithms for simulating a functional connectivity network for a single subject. However, there are limited methods through which to simulate connectivity networks across a cohort of subjects. While it would be possible to assume the networks are identical across all subjects, this corresponds to a tenuous assumption which can rarely be validated in practice.

[1] More sophisticated methods (e.g., those proposed by Varoquaux et al. [2010]) have also been suggested but we do not discuss this further in this vignette. See Monti et al. [2015a] for a detailed discussion

In this section we present a novel algorithm through which to simulate functional connectivity networks across a cohort of subjects. The resulting networks are shown to display some of the well-documented properties of connectivity. These properties include significant inter-subject variability [Mueller et al., 2013] together with a subset of highly reproducible edges.

The remainder of this section is organized as follows: we briefly review some of the properties of functional connectivity networks in Section 2.1. The proposed algorithm is then introduced in Section 2.2 together with an alternative method in Section 2.3. We conclude by presenting some examples in Section 2.4.

2.1 Properties of functional connectivity networks

There are several well-documented properties of functional connectivity networks, chief among which is their modular structure and the presence of hub nodes [Bullmore and Sporns, 2009]. In addition to this, high inter-subject variability is often reported across subjects. A hallmark of this variability is that it does not occur uniformly but instead tends to display certain characteristics: for example the pre-frontal region is often reported to display high inter-subject variability [Finn et al., 2015] and the distance between regions is also hypothesized to play a role [Van Dijk et al., 2012].

It is often the case that the these characteristics are quantified and measured via the use of graph theoretic techniques [Rubinov and Sporns, 2010]. For example, the clustering coefficient is often employed as a measure of functional segregation or modularity while the degree distribution is often employed to see if the network follows a power-law distribution.

In the remainder of this section we present a new algorithm through which to simulate multiple related functional connectivity networks and subsequently employ graph theoretic measures to demonstrate that the proposed algorithm is able to recreate many of the properties typically observed in neuroimaging data. In particular we focus on recreating the following properties:

1. Networks should display a scale-free organization [Bullmore and Sporns, 2009]. This implies that node degrees should follow a power-law distribution resulting in the presence of highly connected hub nodes.

2. Significant inter-subject variability should be present. Following from reports in the literature, we assume there is a subset of edges which demonstrate the highest variability (e.g., these could correspond to edges in the pre-frontal region or between spatially remote regions as discussed previously).

3. Finally, based on an exploratory data analysis of the ABIDE data (documented in Monti et al. [2015a]) we found that the clustering coefficient, a measure of network cohesiveness [Barrat et al., 2004], was significantly higher within a population network as opposed to subject-specific networks We therefore look to recreate this property as well.

2.2 Proposed algorithm

The proposed algorithm proceeds as follows: first a population network is simulated according to the preferential attachment model of Barabási and Albert [1999]. This corresponds to the set of reproducible edges which are shared throughout the entire cohort of subjects, denoted

by E^{pop}. In order to obtain the corresponding precision matrix, Θ^{pop}, we follow Danaher et al. [2014] and uniformly sample the edge strengths.

In order to introduce inter-subject variability a set of variable edges, $\tilde{E}$, is randomly selected according to the Erdős and Rényi [1959] model. The cardinality of $\tilde{E}$ is specified, such that only $e_{ran} = |\tilde{E}|$ random edges are selected. The choice of e_{ran} directly affects the extent of inter-subject variability in the simulated cohort. We write $E^{(i)}$ to denote the subject-specific idiosyncrasies associated with the ith subject. Thus, for each subject edges in $\tilde{E}$ are added to the edge structure, $E^{(i)}$, with probability $\tau \in [0, 1]$. If variable edges are present their strength is sampled uniformly at random independently for each subject. We note that setting $\tau = 0$ results in an identical network for all subjects. At the other end of the spectrum, setting $\tau = 1$ results in all subjects having identical edge support (i.e., the same edges will be present or absent across all subjects). However, edge weights for edges within $\tilde{E}$ are randomly sampled thereby introducing variability across subjects.

Pseudo-code for the proposed method is provided in Algorithm 1. It is important to note that the proposed algorithm returns simulated network structure for both the population connectivity network as well as the subject-specific networks. Moreover, we also obtain a simulated network of highly variable edges captured in $\tilde{E}$.

Algorithm 1: Generate population and subject-specific random networks

Input: Number of nodes p, number of subjects N, size of random effects network $e_{ran} = |\tilde{E}|$, a random effects edge probability $\tau \in [0, 1]$ and connectivity strength $r \in \mathbb{R}_+$

Result: Population network, Θ^{pop}, subject-specific networks, $\{\Theta^{(i)}\}$, random effects edges $\tilde{E}$

```
begin
    Simulate E^pop according to Barabási and Albert [1999] model
    Build Θ^pop by randomly selecting edge weights from the interval [−r, −r/2] ∪ [r/2, r]
    Simulate Ẽ according to Erdős and Rényi [1959] model with e_ran edges
    for i ∈ {1, ..., N} do
        for each edge (j, k) do
            if (j, k) ∈ Ẽ then
                (j, k) ∈ E^(i) with probability τ
        Randomly select edge weights and signs for Θ^(i)
return E^pop, Ẽ, {E^(i)} and Θ^pop, {Θ^(i)}
```

2.2.1 Data generation

While Algorithm 1 can be employed to simulate functional connectivity networks across a cohort of subjects, care must be taken when looking to simulate the corresponding data. In this package, data is generated for each subject according to the following multivariate normal distribution:

$$X^{(i)} \sim \mathcal{N}\left(0, \left(PD\left(\Theta^{pop} + \Theta^{(i)}\right)\right)^{-1}\right), \tag{1}$$

where $PD(\cdot)$ is a function applied in order to ensure the resulting matrix is positive definite. In this work we follow Danaher et al. [2014] and ensure $\Theta^{pop} + \Theta^{(i)}$ is positive definite by rescaling the off-diagonal entries. This involves dividing each off-diagonal entry by the sum of the absolute values of all off-diagonal elements in its corresponding row. This results in a non-symmetric matrix which we average with its transpose in order to obtain a symmetric matrix.

2.3 Alternative simulation methods

In addition to the Algorithm described above, the `MNS` package also implements the network simulation method described in Danaher et al. [2014]. Briefly, this method simulates related networks for a three-subject problem. Nodes are divided into ten equally sized and unconnected sub-networks. Within each sub-network the connectivity structure is simulated according to the preferential attachment model of Barabási and Albert [1999], thus nodes display a power-law degree distribution. Of the ten networks, eight are present across all three subjects. Of the remaining two sub-networks, one is present in two of the three subjects while the final sub-network is present only in one subject.

While such an approach shares several similarities with the proposed method (e.g., node degree follow a power-law distribution in both), there are several key differences. First and foremost, this method is only able to simulate subject-specific networks. As a result, there is no clear method from which to obtain population networks. Moreover, it can be argued that this method of network simulation is unrealistic as nodes are divided into equally sized and unconnected components. Finally, in its current implementation this method is only able to simulate data for $N = 3$ subjects and the degree of inter-subject variability is fixed. This is in contrast to the proposed method where networks can be simulated for any number of subjects and the degree of inter-subject variability can be varied by changing parameters e_{ran} and τ.

2.4 Implementation and examples

In this section we provide various examples to highlight the network simulation methods within the `MNS` package and give example code.

Within the `MNS` package, random networks are simulated via the `gen.Network` function. This function implements both the proposed method for network simulation, described in Section 2.2, as well as the method of Danaher et al. [2014], described in Section 2.3.

The `gen.Network` function takes as input a number of parameters, the most important of which is the `method` parameter which defines the algorithm through which to simulate random networks. This parameter can take one of two values; the default setting of `method="cohort"` simulates random networks according to algorithm 1 while setting `method="danaher"` employs the algorithm described in Section 2.3.

The number of nodes is specified by parameter `p`, while the parameter `Nobs` specifies the number of observations to simulate per subject. If this parameter is not provided then only the random networks are simulated and returned (i.e., random data for each subject is not simulated). The remaining parameters only affect the `"cohort"` simulation method; `Nsub` is the number of subjects[2], `sparsity` is the sparsity of the population network[3], `REsize` and

[2] note this is fixed at three for `method="danaher"`

[3] the number of edges added in each step of the Barabási and Albert [1999] algorithm is altered to obtain the desired sparsity

REprob correspond to e_{ran} and τ above and REnoise determines the variability across edges in $\tilde{E}$. Some examples are described below.

In the following code networks are simulated using Algorithm 1 for $N = 3$ subjects. The resulting object, Net, is a list which contains three entries. The first, called Networks, is a list of length N where the ith entry corresponds the precision matrix for the ith subject. Note that is will contain both the population edges, E^{pop}, as well as the subject-specific edges, $E^{(i)}$. The second entry, called PopNet, is the population network while the third entry, RanNet, indicates which edges are highly variable. The resulting simulating networks can then be plotted using the plot.MNS function. This is an S3 method for objects of class MNS and is discussed further in Section 3.

```
> library("MNS")
> set.seed(1)
> N=3
> Net = gen.Network(method = "cohort", p = 20,
+                           Nsub = N, sparsity = .2,
+                           REsize = 20, REprob = .5,
+                           REnoise = 1)
> # plot simulated networks:
> plot(Net, view="sub")
```

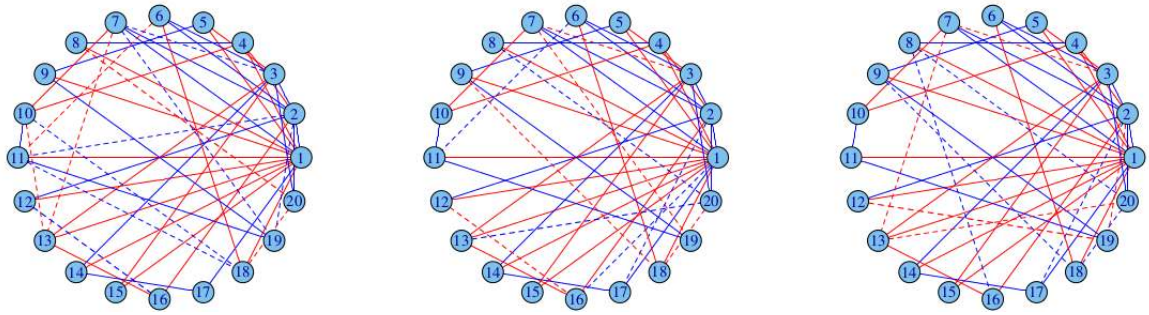

Figure 1: Random networks for $N = 3$ subjects simulated using Algorithm 1. Solid edges indicate reproducible population edges present across all subjects while dashed edges represent subject-specific idiosyncrasies and encode inter-subject variability. Edge color is indicative of the nature of the partial correlation.

The above code results in the networks shown in Figure 1, one per subject. Solid edges are population edges which are present across all subjects while dashed edges represent subject-specific idiosyncrasies which are only present across some of the subjects. Finally, line color is indicative of the nature of the relation between nodes; blue edges indicating a positive partial correlation while negative partial correlations are indicated by red edges. We note there is clear inter-subject variability introduced by the additional edges. Returning to Algorithm 1, the blue edges are generated in step 2. While the set of red variable edges, $\tilde{E}$, is selected in step four and these edges are randomly added in steps 7 and 8.

We note that it is possible to obtain networks varying edge densities by appropriately adjusting the sparsity parameter Moreover, additional inter-subject variability can be introduced by increasing REsize or REprob.

In contrast, we may also simulate networks according to the model proposed in Section 2.3. Note that this only requires the number of nodes, p, to be specified.

```
> library("MNS")
> set.seed(1)
> Net = gen.Network(method = "Danaher", p = 20)
> # plot simulated networks:
> plot(Net, view="sub")
```

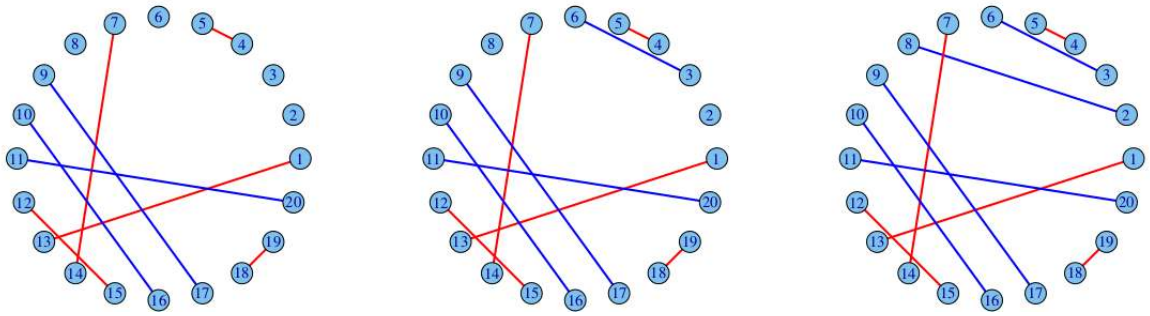

Figure 2: Random networks as simulated using method described in Section 2.3.

Finally, we note that if the `Nobs` argument is passed then multivariate data will be simulated as discussion in Section 2.2.1. This can be achieved as follows, with an example plotted in Figure 3.

```
> library("MNS")
> set.seed(1)
> N=3
> Net = gen.Network(method = "cohort", p = 20,
+                         Nobs = 500,
+                         Nsub = N, sparsity = .2,
+                         REsize = 20, REprob = .5,
+                         REnoise = 1)
> # plot simulated networks:
> plot.ts(Net$Data[[1]][, c(1,2,3,4,5, 6)], main="")
```

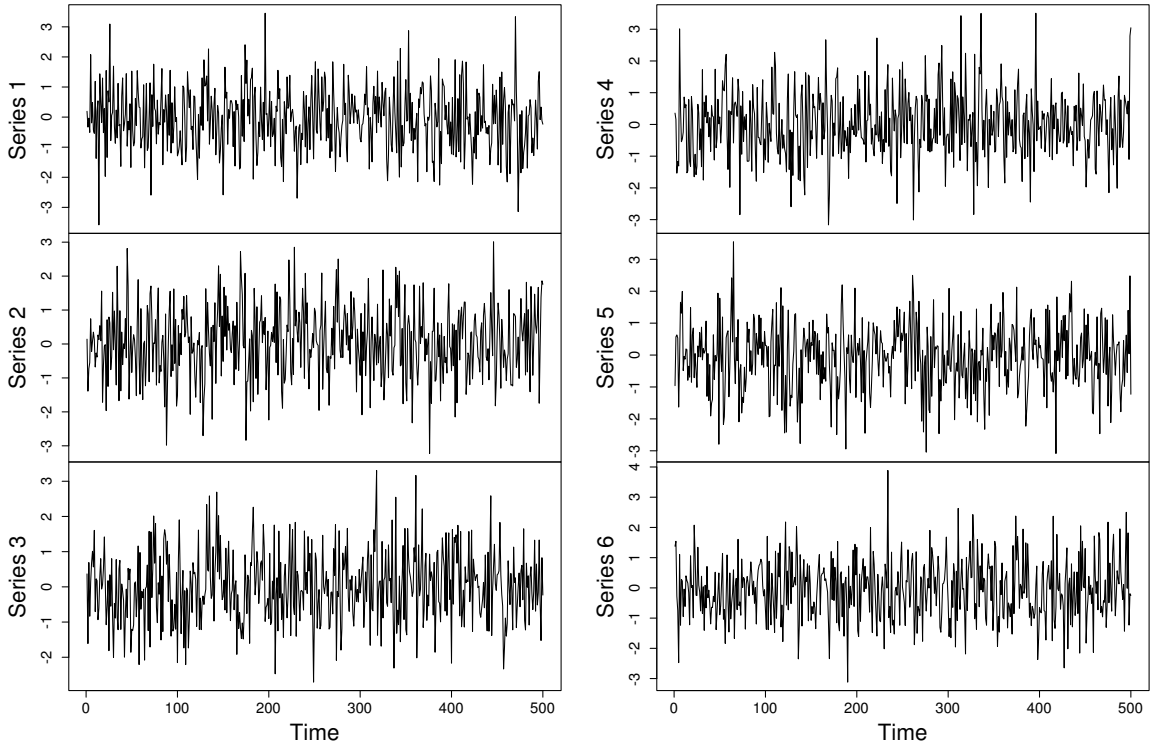

Figure 3: Simulated data for for six nodes across a single subject.

3 Mixed Neighborhood Selection

In this section we briefly overview the Mixed Neighborhood Selection (MNS) algorithm and presents its implementation in the `MNS` package. For a more thorough discussion please see Monti et al. [2015a]. The MNS algorithm looks to model functional connectivity networks as Gaussian graphical models (GGMs). This is a popular approach within the neuroimaging community [Varoquaux and Craddock, 2013] and is partially motivated by the large number of highly efficient algorithms through which to estimate the corresponding graphical models.

Throughout this work we focus on neighborhood selection algorithms, first introduced by Meinshausen and Bühlmann [2006]. The intuition behind neighborhood selection is that if the edge structure at each node can be accurately inferred, then the overall edge structure can also be inferred. Moreover, neighborhood selection methods are particularly appealing since it can be shown that the neighborhood (i.e., the local edge structure) of any node, v, can be inferred be considering the optimal linear prediction of observations at that node given all other nodes. In such models, nodes that are not in the neighborhood of v will be omitted from the set of optimal predictors. As a result, covariance selection is reformulated as a subset selection problem. The latter problem can be efficiently solved via the use of the Lasso [Tibshirani, 1996]. Thus neighborhood selection allows us to recast covariance selection as a series of Lasso regression problems, each of which is solved independently.

The MNS algorithm extends neighborhood selection to the scenario where data from multi-

ple subjects is available. This is achieved by introducing an additional mixed effect component. The objective of the additional random effect is to capture subject-specific idiosyncrasies. This serves to yield a more accurate model of population covariance structure (as inter-subject variability is recognized and dealt with adequately) as well as accurate subject-specific networks (as information is only leveraged across subjects when appropriate). Furthermore, we are also able to obtain an estimate of inter-subject variability on an edge-by-edge basis. This is a crucial advantage of the MNS algorithm when compared to alternative methods. By providing an estimate of variability across edges we are able to obtain a clear intuition as to which regions drive inter-subject variability.

In the remainder of this section we discuss some of the details of the MNS algorithm together with the corresponding functions. In Section 3.1 we discuss parameter selection. The visualization methods of the `MNS` package are discussed in Section 3.2.

3.1 Parameter selection

The MNS algorithm requires the input of two regularization parameters. The first parameter, λ_1, dictates the severity of regularization applied on the population network. Thus large values of λ_1 result in sparse population networks. The second parameter, λ_2, penalizes subject-specific deviations from the population network. Thus large values of λ_2 will lead to identical networks for all subjects. In the `MNS` function these two regularization parameters are specified by the `lambda_pop` and `lambda_random` arguments respectively. These parameters must be selected with care as they are closely related; for example placing a high regularization on the population network (i.e., a high λ_1) may lead the model to over-estimate subject-specific edges as compensation.

In the `MNS` package the regularization parameters are selected via K-fold cross-validation. While alternative approaches based on information theoretic criteria could be employed we found cross-validation to perform better in practice.

The `cv.MNS` function implements K-fold cross-validation. The `dat` argument contains the data for all subjects. This should be in the form a list where the ith entry is the data for the ith subject. The `l1range` and `alpharange` parameters specify the grid of regularization parameters. Note that the sparsity penalty has been re-parameterized as follows:

$$\alpha \cdot \lambda \, ||\beta||_1 + (1 - \alpha) \cdot \lambda \, ||\sigma||_1, \tag{2}$$

where parameters λ are specified by the argument `lambda_range` and parameters α by `alpha_range`.

Cross-validation methods are renown for their high computational cost. In order to reduce some of the computational burden, the `cv.MNS` function can also be run in parallel by appropriately setting the `parallel` argument. Further, the `cores` argument allows users to specify the number of cores to employ.

```
> set.seed(1)
> N=10
> Net = gen.Network(method = "cohort", p = 10,
+                           Nsub = N, sparsity = .2,
+                           REsize = 10, REprob = .75,
+                           REnoise = 1, Nobs = 75)
> # run cross-validation
```

```
> CV = cv.MNS(dat = Net$Data,
+             l1range = seq(.05, .075, length.out = 5),
+             alpharange = seq(.25, .75, length.out = 3),
+             K=5, parallel=TRUE)
> # fit MNS model:
> mns = MNS(dat = Net$Data,
+           lambda_pop = CV$l1 * (1-CV$alpha),
+           lambda_random = CV$l1 * (CV$alpha))
```

3.2 Visualization

The `MNS` package also contains the `plot.MNS` function. This is an `S3` function for objects of class MNS. It can be used both to visualize simulated networks, as shown in Section 2.4, or to plot the output of the MNS algorithm. The `view` argument determines which results are plotted. The `"pop"`, `"var"` and `"sub"` options plot the population, variable and subject-specific networks respectively.

Following from the example in Section 3.1 the population network, shown in Figure 4, can be plotted as follows:

```
> plot(mns, view="pop")
```

As before, edge color is indicative of whether the partial correlation between each of the two nodes is positive or negative while the edge thickness indicates the magnitude of the partial correlation.

Moreover, since an edge-by-edge estimate of variability is provided for networks it is possible to identify specific functional relations which are irregular across the cohort. The network of variable edges is plotted by changing the `view` argument:

```
> plot(mns, view="var")
```

The result is shown in Figure 5. Since variability is reported on an edge-by-edge basis the results are easily interpretable. As before, the edge thickness is proportional to the magnitude of the variance.

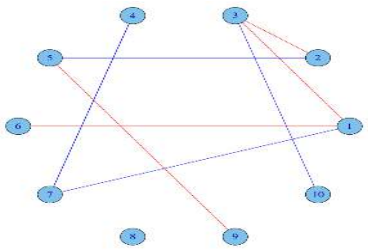

Figure 4: Estimated population network. This is the set of edges which are reproducible across the entire cohort.

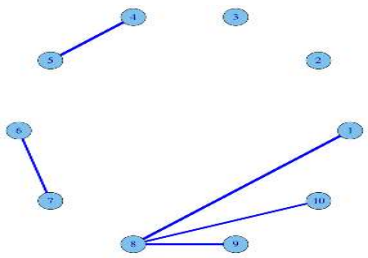

Figure 5: Estimate variable edges. Plotted edges are highly variable across the entire cohort of subjects.

Finally, it is also possible to view the estimated networks on a subject-specific basis. As before, this is achieved by changing the `view` argument. Further, the `subID` argument selects which subjects to plot. In the code below we plot the networks for the 2nd, 4th and 6th subjects.

```
> plot(mns, view="sub", subID=c(2,4,6))
```

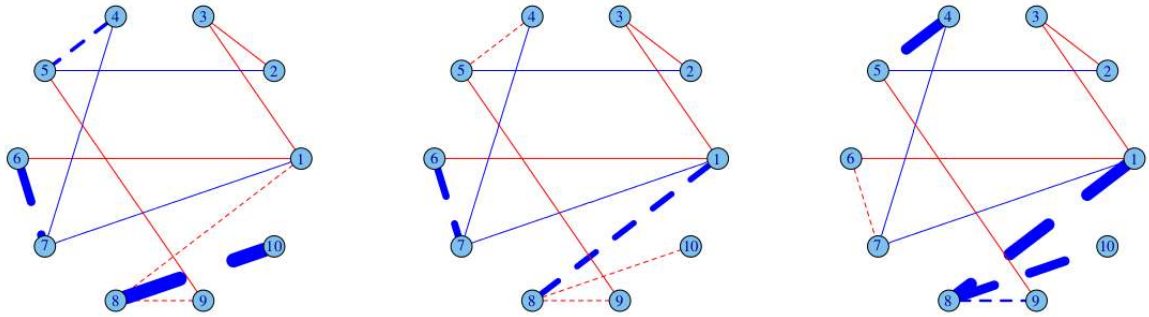

Figure 6: Estimated subject-specific networks for subjects 2, 4 and 6. Population edges are plotted as solid lines while variable edges are plotted as dashed lines.

The results, shown in Figure 6, show the estimated networks for three of the ten subjects. As before, the color and thickness of each edge indicates the nature and magnitude of the partial correlation. Population edges are plotted as solid lines whiles variable edges are plotted as dashed lines.

4 Conclusion

In this vignette we have introduced the `MNS` package and highlighted the functionality and use of its functions. The `MNS` package has been written in order to estimate multiple related Gaussian graphical models. While the motivation for this work has been estimating resting-state functional connectivity networks from fMRI data the methods described in this vignette (and in Monti et al. [2015a]) can be applied in any scenario where the objective is to estimate multiple graphical models. Another possible application of the `MNS` algorithm would be to study variability over time within a single subject. For example, it has been suggested that functional connectivity is non-stationary [Monti et al., 2014, 2015b]. Thus by dividing resting-state data into blocks it would be possible to study variability within a scan for a single subject.

In order to empirically validate the `MNS` algorithm, we have presented a novel algorithm through which to simulate realistic networks. The proposed method can simulate networks with varying levels of inter-subject variability. Moreover, the resulting networks demonstrate several well-documents properties observed in functional connectivity networks; these include a power-law distribution for the node degree as well as significant inter-subject variability.

References

A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439): 509–512, 1999.

A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani. The architecture of complex weighted networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(11):3747–3752, 2004.

E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009.

P. Danaher, P. Wang, and D. Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.

A. Di Martino, C. Yan, Q. Li, E. Denio, F. Castellanos, K. Alaerts, J. Anderson, M. Assaf, S. Bookheimer, and M. Dapretto. The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism. *Molecular psychiatry*, 19(6):659–667, 2014.

P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

E. Finn, X. Shen, D. Scheinost, M. Rosenberg, J. Huang, M. Chun, X. Papademetris, and T. Constable. Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity. *Nature neuroscience*, 2015.

K. Friston. Functional and effective connectivity: a review. *Brain connectivity*, 1(1):13–36, 2011.

C. Kelly, B. Biswal, C. Craddock, X. Castellanos, and M. Milham. Characterizing variation in the functional connectome: promise and pitfalls. *Trends in cognitive sciences*, 16(3): 181–188, 2012.

N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, pages 1436–1462, 2006.

R. P. Monti, P. Hellyer, D. Sharp, R. Leech, C. Anagnostopoulos, and G. Montana. Estimating time-varying brain connectivity networks from functional MRI time series. *Neuroimage*, 103:427–443, 2014.

R. P. Monti, C. Anagnostopoulos, and G. Montana. Inferring brain connectivity networks from functional MRI data via mixed neighbourhood selection. *Preprint*, 2015a.

R. P. Monti, R. Lorenz, P. Hellyer, R. Leech, C. Anagnostopoulos, and G. Montana. Graph embeddings of dynamic functional connectivity reveal discriminative patterns of task engagement in HCP data. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 1–4. IEEE, 2015b.

S. Mueller, D. Wang, M. Fox, T. Yeo, J. Sepulcre, M. Sabuncu, R. Shafee, J. Lu, and H. Liu. Individual variability in functional connectivity architecture of the human brain. *Neuron*, 77(3):586–595, 2013.

M. Rubinov and O. Sporns. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, 52(3):1059–1069, 2010.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

K. Van Dijk, M. Sabuncu, and R. Buckner. The influence of head motion on intrinsic functional connectivity MRI. *Neuroimage*, 59(1):431–438, 2012.

G. Varoquaux and C. Craddock. Learning and comparing functional connectomes across subjects. *NeuroImage*, 80:405–415, 2013.

G. Varoquaux, A. Gramfort, J. Poline, and G. Thirion. Brain covariance selection: better individual functional connectivity models using population prior. In *Advances in Neural Information Processing Systems*, pages 2334–2342, 2010.