



hergm: Hierarchical Exponential-Family Random Graph Models

Michael Schweinberger
Rice University

Pamela Luna
Rice University

Abstract

We describe the R package **hergm** that implements hierarchical exponential-family random graph models with local dependence. Hierarchical exponential-family random graph models with local dependence tend to be superior to conventional exponential-family random graph models with global dependence in terms of goodness-of-fit. The advantage of hierarchical exponential-family random graph models is rooted in the local dependence induced by them. We discuss the notion of local dependence and the construction of models with local dependence along with model estimation, goodness-of-fit, and simulation. Simulation results and three applications are presented.

Keywords: social networks, random graphs, Markov random graph models, exponential-family random graph models, stochastic block models, model-based clustering.

1. Introduction

Data that can be represented by random graphs arise in areas as diverse as the life sciences (e.g., protein interaction networks), the health sciences (e.g., contact networks facilitating the transmission of diseases), the social sciences (e.g., insurgent and terrorist networks), computer science (e.g., social networks and the World Wide Web), and engineering (e.g., power and transportation networks); see [Wasserman and Faust \(1994\)](#) and [Kolaczyk \(2009\)](#).

The R ([R Core Team 2017](#)) package **hergm** ([Schweinberger, Handcock, and Luna 2018](#)) implements a wide range of random graph models, including stochastic block models ([Nowicki and Snijders 2001](#)), exponential-family random graph models (ERGMs; [Frank and Strauss 1986](#); [Wasserman and Pattison 1996](#); [Snijders, Pattison, Robins, and Handcock 2006](#); [Hunter and Handcock 2006](#)), and hierarchical exponential-family random graph models (HERGMs; [Schweinberger and Handcock 2015](#)), with an emphasis on HERGMs with local dependence. HERGMs with local dependence are motivated by the observation that networks are local

in nature and – while edges in networks depend on other edges – most edges depend on a small number of other edges. The R package **hergm** is the first R package that implements HERGMs with local dependence. The package is available from the Comprehensive R Archive Network (CRAN) at <https://CRAN.R-project.org/package=hergm>.

We show in Section 2 how the R package **hergm** can be installed and loaded in R. We discuss the notion of local dependence and demonstrate how models with local dependence can be constructed in Section 3. Model estimation, goodness-of-fit, and simulation are described in Sections 4, 5, and 6, respectively. We present simulation results in Section 7 and three applications in Section 8.

Comparison with related R packages. The models implemented in R package **hergm** are related to, but distinct from, the models implemented in R packages **ergm** (Hunter, Handcock, Butts, Goodreau, and Morris 2008b), **Bergm** (Caimo and Friel 2014), **gergm** (Denny, Wilson, Cranmer, Desmarais, and Bhamidi 2017), and **tergm** (Krivitsky and Handcock 2016). While the R package **ergm** and all other mentioned R packages focus on ERGMs and generalizations of ERGMs, the R package **hergm** focuses on HERGMs with local dependence. HERGMs are ERGMs with additional structure, called neighborhood structure, which is exploited to construct models with local dependence. The (unobserved) neighborhood structure and local dependence give rise to unique computational and statistical challenges, which are addressed by R package **hergm**.

2. Installing R package **hergm**

The R package **hergm** (Schweinberger *et al.* 2018) is distributed under the GPL-3 license. It depends on the R packages **ergm** (Hunter *et al.* 2008b), **latentnet** (Krivitsky and Handcock 2008), **mcgibbsit** (Warnes and Burrows 2013), **network** (Butts 2008a), and **sna** (Butts 2008b). The R packages **ergm**, **latentnet**, **network**, and **sna** are described in the special issue of the *Journal of Statistical Software* devoted to the R suite of packages **statnet** (Handcock, Hunter, Butts, Goodreau, and Morris 2008, Volume 24, Issue 5).

The R package **hergm** and its dependencies are available at <https://CRAN.R-project.org/> and can be installed as follows:

```
R> install.packages("hergm")
```

The R package **hergm** can be loaded in R as follows:

```
R> library("hergm")
```

The results reported here are based on **hergm** version 3.2-1, **ergm** version 3.8.0, **latentnet** version 2.8.0, **mcgibbsit** version 1.1.0, **network** version 1.13.0, **parallel** version 3.3.2, and **sna** version 2.4.

3. Model specification

The R package **hergm** implements a wide range of models characterized by local dependence. We discuss the notion of local dependence in Section 3.1 and the construction of models with

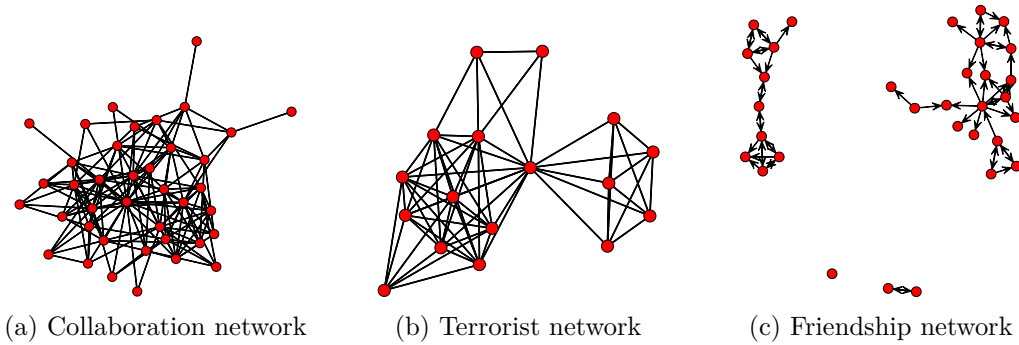


Figure 1: Three examples of networks described in Section 8: (a) collaboration network, (b) terrorist network, and (c) friendship network.

local dependence in Section 3.2. Throughout, we consider data that can be represented by random graphs with a set of nodes $\mathcal{A} = \{1, \dots, n\}$ and a set of edges $\mathcal{E} \subseteq \mathcal{A} \times \mathcal{A}$ representing, for example, friendships between members of a social network. The edges $X_{i,j} \in \{0, 1\}$ between nodes i and j are considered to be random variables and may be undirected – that is, $X_{i,j} = X_{j,i}$ with probability 1 – or directed. Self-edges are excluded, that is, $X_{i,i} = 0$ with probability 1. Three examples of networks are shown in Figure 1. The three networks are included in R package **hergm** as data sets and can be loaded in R and plotted as follows:

```
R> set.seed(0)
R> data("kapferer", package = "hergm")
R> gplot(kapferer, gmode = "graph", mode = "kamadakawai",
+       displaylabels = FALSE)
R> data("bali", package = "hergm")
R> gplot(bali, gmode = "graph", mode = "kamadakawai", displaylabels = FALSE)
R> data("bunt", package = "hergm")
R> gplot(bunt, mode = "kamadakawai", displaylabels = FALSE)
```

It is worth noting that network plots generated by the function `gplot` of R package **sna** (Butts 2008b) are not reproducible unless the seed of the pseudo-random number generator is specified, because the initial positions of nodes are chosen at random (unless users specify the initial positions of nodes). The three networks are described in more detail in Section 8.

3.1. Local dependence

A well-known problem of conventional ERGMs is that some of the most interesting ERGMs induce global dependence. The best-known example are the Markov random graph models of Frank and Strauss (1986). A Markov random graph model assumes that, if two edge variables $X_{i,j}$ and $X_{k,l}$ do not share nodes, then $X_{i,j}$ and $X_{k,l}$ are independent conditional on all other edge variables:

$$X_{i,j} \perp\!\!\!\perp X_{k,l} \mid \text{others} \quad \text{for all } \{i,j\} \cap \{k,l\} = \{\}. \quad (1)$$

Frank and Strauss (1986) showed that the resulting distribution of a random graph $\mathbf{X} = (X_{i,j})$ can be written in exponential-family form as follows:

$$P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x}) = \exp(\langle \boldsymbol{\theta}, s(\mathbf{x}) \rangle - \psi(\boldsymbol{\theta})), \quad \mathbf{x} \in \mathbb{X}, \quad (2)$$

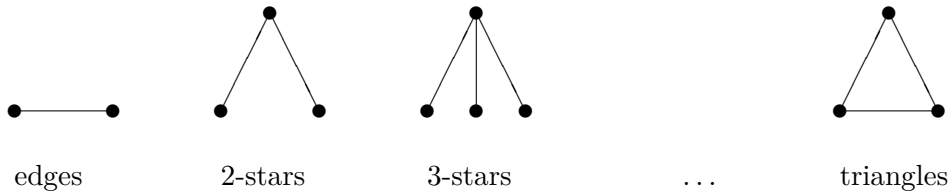


Figure 2: Sufficient statistics of Markov random graph models of undirected networks (Frank and Strauss 1986): Counts of the number of edges, k -stars ($k = 2, \dots, n - 1$), and triangles.

where \mathbb{X} denotes the set of possible graphs, $\langle \boldsymbol{\theta}, s(\mathbf{x}) \rangle = \sum_{i=1}^d \theta_i s_i(\mathbf{x})$ denotes the inner product of a vector of natural parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ and a vector of sufficient statistics $s : \mathbb{X} \mapsto \mathbb{R}^d$, and $\psi : \mathbb{R}^d \mapsto \mathbb{R}$ ensures that $P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x})$ sums to 1. The sufficient statistics of Markov random graph models of undirected networks are shown in Figure 2. The conditional independence assumption (1) allows each edge variable $X_{i,j}$ to depend on up to $2(n-2)$ other edge variables. If Markov random graph models were applied to large networks, such as the friendship network consisting of all users of Facebook, then each possible friendship would depend on billions of other possible friendships. Such models with global dependence are known to be near-degenerate in the sense that such models concentrate most probability mass on graphs that have either almost no edges or almost all possible edges. Two empirical examples are presented in Sections 8.2 and 8.3. Theoretical results can be found in Handcock (2003), Schweinberger (2011), and Chatterjee and Diaconis (2013). Models with global dependence are not useful, because many real-world networks do not resemble graphs that have almost no edges or almost all possible edges. Models with local dependence would be more appealing than models with global dependence, but it is not evident how to construct them: in contrast to spatial and temporal data that come with a natural neighborhood structure in the form of space or time, network data do not come with a natural neighborhood structure, and therefore it is not straightforward to construct models with local dependence.

A first step toward the construction of models with local dependence was taken by Schweinberger and Handcock (2015). The underlying idea can be described as follows:

Science suggests that dependence is local, that is, that every edge variable depends on a small number of other edge variables. Local dependence has both probabilistic advantages (e.g., weak dependence) and statistical advantages (e.g., consistency of estimators). If it is unknown which edge variable depends on which other edge variables, then the neighborhoods of edge variables should be inferred.

A simple form of local dependence was introduced by Schweinberger and Handcock (2015), which can be defined as follows.

Definition (Local and global dependence). *The dependence induced by a random graph model is called local if there exists a partition of the set of nodes \mathcal{A} into $K \geq 2$ non-empty subsets of nodes $\mathcal{A}_1, \dots, \mathcal{A}_K$, called neighborhoods, such that the dependence induced by the random graph model is restricted to within-neighborhood subgraphs, that is, the probability mass function of*

a random graph \mathbf{X} factorizes as follows:

$$\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x}) = \prod_{k=1}^K \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X}_{\mathcal{A}_k, \mathcal{A}_k} = \mathbf{x}_{\mathcal{A}_k, \mathcal{A}_k}) \prod_{l=k+1}^K \prod_{i \in \mathcal{A}_k < j \in \mathcal{A}_l} \mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = x_{i,j}, X_{j,i} = x_{j,i}), \quad (3)$$

where $\mathbf{X}_{\mathcal{A}_k, \mathcal{A}_k}$ denotes the within-neighborhood subgraph induced by neighborhood \mathcal{A}_k , that is, the subset of edge variables corresponding to nodes in neighborhood \mathcal{A}_k . Otherwise the dependence induced by the random graph model is called global.

ERGMs with local dependence are more appealing than ERGMs with global dependence on scientific, probabilistic, and statistical grounds:

1. *Scientific appeal:* It is well-known that networks are local in nature (e.g., [Pattison and Robins 2002](#)) and hence there are good reasons to believe that dependence in networks is local, and more so in large networks.
2. *Probabilistic appeal:* Models with local dependence induce weak dependence as long as all neighborhoods are small and models with weak dependence tend to be non-degenerate. These informal statements can be backed up by formal results: [Schweinberger and Stewart \(2017\)](#) proved that a wide range of ERGMs with local dependence are non-degenerate as long as the sizes of neighborhoods are either bounded above or grow slowly with the number of neighborhoods. In contrast, many ERGMs with global dependence are known to be near-degenerate (e.g., [Handcock 2003](#); [Schweinberger 2011](#); [Chatterjee and Diaconis 2013](#)). While some ERGMs with global dependence, such as ERGMs with geometrically weighted model terms ([Snijders et al. 2006](#); [Hunter and Handcock 2006](#)), have turned out to be useful in applications, there are no mathematical results that show whether, and under which conditions, such ERGMs are non-degenerate as the size of the network increases.
3. *Statistical appeal:* It is well-known that statistical inference for many ERGMs with global dependence is problematic: The results of [Shalizi and Rinaldo \(2013\)](#) suggest that maximum likelihood estimators of many ERGMs with global dependence may not be consistent estimators. In contrast, statistical inference for ERGMs with local dependence is promising: [Schweinberger and Stewart \(2017\)](#) showed that maximum likelihood estimators of ERGMs with local dependence are consistent estimators. These consistency results are the first consistency results that cover a large class of ERGMs with dependence and suggest that ERGMs with local dependence constitute a promising class of ERGMs.

The idea underlying HERGMs is that, if the neighborhood structure is unobserved, then it is natural to estimate it. We describe the specification of HERGMs with local dependence in Sections 3.2–3.6 and conclude with a short comparison of ERGMs and HERGMs in Section 3.7.

3.2. Constructing models with local dependence

A simple approach to constructing models with local dependence was introduced by [Schweinberger and Handcock \(2015\)](#).

Let $\mathbf{Z} = (Z_1, \dots, Z_n)$ be the neighborhood memberships of nodes $1, \dots, n$, where $Z_i = k$ indicates that node i is a member of neighborhood \mathcal{A}_k , $k = 1, \dots, K$. In practice, the neighborhood memberships of nodes may or may not be observed, as discussed in Section 3.5. A model with local dependence assumes that \mathbf{X} and \mathbf{Z} are governed by distributions of the form

$$\begin{aligned} \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x}, \mathbf{Z} = \mathbf{z}) &= \mathbb{P}(\mathbf{Z} = \mathbf{z}) \prod_{k=1}^K \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X}_{\mathcal{A}_k, \mathcal{A}_k} = \mathbf{x}_{\mathcal{A}_k, \mathcal{A}_k} \mid \mathbf{Z} = \mathbf{z}) \\ &\quad \times \prod_{l=k+1}^K \prod_{i \in \mathcal{A}_k < j \in \mathcal{A}_l} \mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = x_{i,j}, X_{j,i} = x_{j,i} \mid \mathbf{Z} = \mathbf{z}). \end{aligned}$$

In other words, conditional on $\mathbf{Z} = \mathbf{z}$, the model induces local dependence as long as there are $K \geq 2$ neighborhoods.

To parameterize models with local dependence, it is convenient to assume that the conditional distributions of within- and between-neighborhood subgraphs given $\mathbf{Z} = \mathbf{z}$ are members of exponential families of distributions (Barndorff-Nielsen 1978; Brown 1986). Then the conditional distribution of \mathbf{X} given $\mathbf{Z} = \mathbf{z}$ can be written in exponential-family form as follows:

$$\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) = \exp(\langle \boldsymbol{\eta}(\boldsymbol{\theta}, \mathbf{z}), s(\mathbf{x}, \mathbf{z}) \rangle - \psi(\boldsymbol{\theta}, \mathbf{z})), \quad (4)$$

where the natural parameter vector $\boldsymbol{\eta}(\boldsymbol{\theta}, \mathbf{z})$ is a linear function of the within- and between-neighborhood natural parameter vectors and the sufficient statistics vector $s(\mathbf{x}, \mathbf{z})$ is a linear function of the within- and between-neighborhood sufficient statistics vectors.

The R package **hergm** admits the specification of a wide range of models with local dependence. Models can be constructed by combining model terms of the R packages **ergm** and **hergm**. The help function of R package **hergm** lists all possible model terms: see `?ergm.terms` and `?hergm.terms`. The **ergm** terms include covariate terms (e.g., Morris, Handcock, and Hunter 2008), which can be used in **hergm**. The **hergm** terms include model terms inducing local dependence. Examples of **hergm** terms are shown in Tables 1 and 2. To construct models, one combines model terms. We give examples of models of undirected and directed networks in Sections 3.3 and 3.4, respectively.

3.3. Examples of models: Undirected networks

We give examples of how to construct models of undirected networks. Throughout, we assume that `network` represents an undirected network.

Stochastic block models (Nowicki and Snijders 2001). Stochastic block models assume that edge variables are independent conditional on the neighborhood memberships of nodes:

$$X_{i,j} \mid Z_i = z_i, Z_j = z_j \stackrel{\text{ind}}{\sim} \text{Bernoulli}(p_{z_i, z_j}), \quad (5)$$

implying

$$\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \propto \exp\left(\sum_{i < j} \eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j}\right), \quad (6)$$

where the edge parameter $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by

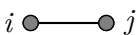
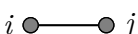
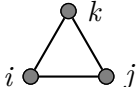
hergm	Subgraph	hergm term	Description
<code>edges_i</code>		$(\theta_{z_i} + \theta_{z_j}) x_{i,j}$	Local edge term (additive)
<code>edges_ij</code>		$\theta_{z_i, z_j} x_{i,j}$	Local edge term (non-additive)
<code>triangle_ijk</code>		$\theta_{z_i, z_j, z_k} x_{i,j} x_{j,k} x_{k,i}$	Local triangle term

Table 1: Examples of **hergm** terms inducing local dependence: Undirected networks.

- $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{z_i} + \theta_{z_j}$, which can be specified by
`R> ergm(network ~ edges_i)`
- $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{z_i, z_j}$, which can be specified by
`R> hergm(network ~ edges_ij)`

It is worth noting that the parameters θ_{z_i} and θ_{z_j} of model term `edges_i` and the parameters θ_{z_i, z_j} of model term `edges_ij` parameterize the log odds of the conditional probability of an edge between nodes i and j given $Z_i = z_i$ and $Z_j = z_j$, that is,

$$\log \frac{\mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = 1 \mid Z_i = z_i, Z_j = z_j)}{\mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = 0 \mid Z_i = z_i, Z_j = z_j)} = \begin{cases} \theta_{z_i} + \theta_{z_j} & (\text{edges_i}) \\ \theta_{z_i, z_j} & (\text{edges_ij}). \end{cases} \quad (7)$$

The first parameterization (`edges_i`) is more restrictive than the second parameterization (`edges_ij`), because it assumes that the log odds $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z})$ is additive in the propensities θ_{z_i} and θ_{z_j} of nodes i and j to form edges. The propensities θ_{z_i} and θ_{z_j} of nodes i and j to form edges depend on i and j through i 's and j 's neighborhood memberships z_i and z_j , respectively. The second parameterization does not assume that $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z})$ is additive in the propensities of nodes i and j to form edges and is therefore more general than the first parameterization. The parameters θ_k of model term `edges_i` and the parameters $\theta_{k,l}$ of model term `edges_ij` are parameters that are estimated based on the observed network.

HERGMs with local dependence (Schweinberger and Handcock 2015). While stochastic block models can be used to cluster nodes, such models do not capture transitivity and other dependencies of interest. Transitivity refers to the tendency of triples of nodes i, j, k to be transitive in the sense that, when i and j are connected by an edge and j and k are connected by an edge, then i and k tend to be connected by an edge as well. A simple term that captures transitivity is based on the number of triangles in the network. A global triangle term counts the total number of triangles in the network and induces global dependence, whereas a local triangle term counts the total number of triangles within neighborhoods and induces local dependence. ERGMs with global triangle terms are known to be near-degenerate (e.g., Handcock 2003; Schweinberger 2011; Chatterjee and Diaconis 2013). An alternative to ERGMs with global triangle terms are HERGMs with local triangle terms, which can be specified as follows:

$$\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \propto \exp \left(\sum_{i < j}^n \eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} + \sum_{i < j < k}^n \eta_{2,i,j,k}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} x_{j,k} x_{i,k} \right), \quad (8)$$

where the edge parameter $\eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by $\theta_{1,z_i} + \theta_{1,z_j}$ or θ_{1,z_i,z_j} and the triangle parameter $\eta_{2,i,j,k}(\boldsymbol{\theta}, \mathbf{z})$ is given by θ_{2,z_i,z_j,z_k} if the three nodes i, j, k are members of the same neighborhood and is 0 otherwise. It is worth noting that stochastic block models are special cases of HERGMs with local dependence where $\eta_{2,i,j,k}(\boldsymbol{\theta}) = 0$ for all i, j, k . In R package **hergm**, the model can be specified by

```
R> hergm(network ~ edges_ij + triangle_ijk)
```

where `edges_ij` can be replaced by `edges_i`.

The parameters of HERGMs can be interpreted along the same lines as the parameters of ERGMs (see, e.g., Krivitsky 2012, Section 5.1), that is, by inspecting the log odds of the full conditional probability of an edge between nodes i and j given all other edge variables $\mathbf{X}_{-i,j} = \mathbf{x}_{-i,j}$ and $\mathbf{Z} = \mathbf{z}$. As an example, consider HERGMs with `edges_ij` and `triangle_ijk` terms, in which case the full conditional probability of an edge between nodes i and j given $\mathbf{X}_{-i,j} = \mathbf{x}_{-i,j}$ and $\mathbf{Z} = \mathbf{z}$ can be written as

$$\log \frac{\mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = 1 \mid \mathbf{X}_{-i,j} = \mathbf{x}_{-i,j}, \mathbf{Z} = \mathbf{z})}{\mathbb{P}_{\boldsymbol{\theta}}(X_{i,j} = 0 \mid \mathbf{X}_{-i,j} = \mathbf{x}_{-i,j}, \mathbf{Z} = \mathbf{z})} = \begin{cases} \theta_{1,z_i,z_j} + \sum_{k \in \mathcal{A}_{z_i}, k \neq i,j} \theta_{2,z_i,z_j,z_k} x_{i,k} x_{j,k} & \text{if } z_i = z_j \text{ (within neighborhoods)} \\ \theta_{1,z_i,z_j} & \text{if } z_i \neq z_j \text{ (between neighborhoods)}. \end{cases}$$

In other words, the parameters $\theta_{1,k,k}$ and $\theta_{2,k}$ affect the log odds of the full conditional probabilities of edges within neighborhood \mathcal{A}_k while the parameters $\theta_{1,k,l}$ affect the log odds of the full conditional probabilities of edges between neighborhoods \mathcal{A}_k and \mathcal{A}_l ($k \neq l$).

3.4. Examples of models: Directed networks

We give examples of how to construct models of directed networks. Throughout, we assume that `network` represents a directed network.

Directed version of stochastic block models (Nowicki and Snijders 2001). If

$$X_{i,j} \mid Z_i = z_i, Z_j = z_j \stackrel{\text{ind}}{\sim} \text{Bernoulli}(p_{z_i,z_j}), \quad (9)$$

then

$$\mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \propto \exp \left(\sum_{i < j} \eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} \right), \quad (10)$$

where the edge parameter $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by

- $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{z_i}$, which can be specified by

```
R> hergm(network ~ arcs_i)
```

- $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{z_j}$, which can be specified by

```
R> hergm(network ~ arcs_j)
```

- $\eta_{i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{z_i,z_j}$, which can be specified by

```
R> hergm(network ~ edges_ij)
```

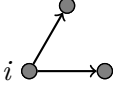
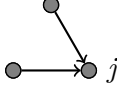
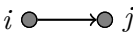
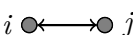

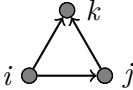
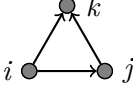
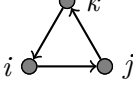
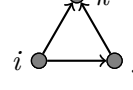
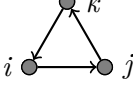

hergm	Subgraph	hergm term	Description
arcs_i		$\theta_{z_i} x_{i,j}$	Local outdegree term (sender effect)
arcs_j		$\theta_{z_j} x_{i,j}$	Local indegree term (receiver effect)
edges_ij		$\theta_{z_i, z_j} x_{i,j}$	Local edge term (non-additive)
mutual_i		$(\theta_{z_i} + \theta_{z_j}) x_{i,j} x_{j,i}$	Local mutual edge term (additive)
mutual_ij		$\theta_{z_i, z_j} x_{i,j} x_{j,i}$	Local mutual edge term (non-additive)
transitiveties_ijk		$\theta_{z_i, z_j} x_{i,j} \max_{k \in \mathcal{A}_{z_i}, k \neq i, j} x_{i,k} x_{j,k}$	Local transitive edge term
ttriple_ijk		$\theta_{z_i, z_j, z_k} x_{i,j} x_{j,k} x_{i,k}$	Local transitive triple term
ctriple_ijk		$\theta_{z_i, z_j, z_k} x_{i,j} x_{j,k} x_{k,i}$	Local cyclic triple term
triangle_ijk	 	$\theta_{z_i, z_j, z_k} x_{i,j} x_{j,k} x_{i,k} + \theta_{z_i, z_j, z_k} x_{i,j} x_{j,k} x_{k,i}$	Local transitive and cyclic triple term

Table 2: Examples of **hergm** terms inducing local dependence: Directed networks.

Directed version of stochastic block models with reciprocity (Vu, Hunter, and Schweinberger 2013). In many applications of directed networks, directed edges tend to be reciprocated in the sense that, if there is a directed edge from node i to node j , then there tends to be a directed edge from node j to node i as well. To capture reciprocity, the stochastic block model can be extended as follows:

$$P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \propto \exp \left(\sum_{i,j} \eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} + \sum_{i < j} \eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} x_{j,i} \right), \quad (11)$$

where the edge parameter $\eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by θ_{1,z_i} or θ_{1,z_j} or θ_{1,z_i, z_j} and the reciprocity parameter $\eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by

- $\eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{2,z_i} + \theta_{2,z_j}$, which can be specified by


```
R> hergm(network ~ edges_ij + mutual_i)
```
- $\eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z}) = \theta_{2,z_i, z_j}$, which can be specified by

```
R> hergm(network ~ edges_ij + mutual_ij)
```

where `edges_ij` can be replaced by `arcs_i` or `arcs_j`.

HERGMs with local dependence (Schweinberger and Handcock 2015). To capture both reciprocity and transitivity, the following model can be used:

$$P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}) \propto \exp \left(\sum_{i,j}^n \eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} + \sum_{i < j}^n \eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} x_{j,i} + \sum_{i,j,k}^n \eta_{3,i,j,k}(\boldsymbol{\theta}, \mathbf{z}) x_{i,j} x_{j,k} x_{i,k} \right),$$

where the edge parameter $\eta_{1,i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by θ_{1,z_i} or θ_{1,z_j} or θ_{1,z_i,z_j} , the reciprocity parameter $\eta_{2,i,j}(\boldsymbol{\theta}, \mathbf{z})$ is given by $\theta_{2,z_i} + \theta_{2,z_j}$ or θ_{2,z_i,z_j} , and the transitivity parameter $\eta_{3,i,j,k}(\boldsymbol{\theta}, \mathbf{z})$ is given by θ_{3,z_i,z_j,z_k} if the three nodes i, j, k are members of the same neighborhood and is 0 otherwise. In R package **hergm**, the model can be specified by

```
R> hergm(network ~ edges_ij + mutual_ij + ttriple_ijk)
```

where `edges_ij` can be replaced by `arcs_i` or `arcs_j` and `mutual_ij` can be replaced by `mutual_i`.

Other model terms. We focus here on transitivity as the main example of network dependence, because it is one of the most interesting dependencies and one of the most challenging dependencies owing to the fact that it induces model degeneracy (e.g., Handcock 2003; Schweinberger 2011; Chatterjee and Diaconis 2013). In other words, we focus on transitivity because we want to demonstrate that HERGMs can alleviate the problem of model degeneracy in one of the most challenging cases, which suggests that HERGMs should be able to alleviate model degeneracy in all other cases as well. However, there are many other model terms that can induce local dependence and could be implemented in R package **hergm**, including local versions of geometrically weighted model terms (Snijders *et al.* 2006; Hunter and Handcock 2006).

3.5. Neighborhood memberships: Observed or unobserved

In practice, the neighborhood memberships of nodes may or may not be observed. The R package **hergm** can handle both observed and unobserved neighborhood memberships.

If all neighborhood memberships are observed, at least two approaches are possible. The first approach regards the observed neighborhood structure, denoted by \mathbf{z}_{obs} , as fixed and bases inference with respect to parameter vector $\boldsymbol{\theta}$ on the likelihood function

$$L(\boldsymbol{\theta}) = P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}_{\text{obs}}). \quad (12)$$

The second approach considers the observed neighborhood structure \mathbf{z}_{obs} as the outcome of a random variable \mathbf{Z} with a distribution $P_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z})$ parameterized by a parameter vector $\boldsymbol{\pi}$ and bases inference with respect to parameter vector $\boldsymbol{\theta}$ on the likelihood function

$$L(\boldsymbol{\theta}, \boldsymbol{\pi}) = P_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}_{\text{obs}}) P_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z}_{\text{obs}}). \quad (13)$$

If $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ are variation-independent in the sense that the parameter space $\Omega_{\boldsymbol{\theta},\boldsymbol{\pi}}$ is a product space of the form $\Omega_{\boldsymbol{\theta},\boldsymbol{\pi}} = \Omega_{\boldsymbol{\theta}} \times \Omega_{\boldsymbol{\pi}}$, where $\Omega_{\boldsymbol{\theta}}$ is the parameter space of $\boldsymbol{\theta}$ and $\Omega_{\boldsymbol{\pi}}$ is the parameter space of $\boldsymbol{\pi}$, then the likelihood function factorizes:

$$L(\boldsymbol{\theta}, \boldsymbol{\pi}) = \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}_{\text{obs}}) \mathbb{P}_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z}_{\text{obs}}) = L(\boldsymbol{\theta}) L(\boldsymbol{\pi}), \quad (14)$$

where $L(\boldsymbol{\theta}) = \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}_{\text{obs}})$ and $L(\boldsymbol{\pi}) = \mathbb{P}_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z}_{\text{obs}})$. Therefore, if $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ are variation-independent, statistical inference with respect to $\boldsymbol{\theta}$ can be based on $L(\boldsymbol{\theta})$. Thus, the two approaches are equivalent as long as $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ are variation-independent. In general, the random-neighborhood approach is preferable to the fixed-neighborhood approach when it is believed that the neighborhoods are generated by a stochastic process of interest.

In R package **hergm**, the random-neighborhood approach is implemented. The random-neighborhood approach assumes that the neighborhood memberships are governed by distributions of the form

$$\mathbb{P}_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z}) = \prod_{i=1}^n \mathbb{P}_{\boldsymbol{\pi}}(Z_i = z_i) = \prod_{i=1}^n \pi_{z_i}, \quad (15)$$

where $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K)$ and π_k is the prior probability of being a member of neighborhood $k = 1, \dots, K$.

3.6. Priors

The R package **hergm** estimates models with local dependence by following a Bayesian approach. A Bayesian approach is based on posteriors, which requires the specification of priors. There are two classes of priors implemented in R package **hergm**, parametric and nonparametric priors.

If the number of neighborhoods K is known, parametric priors are a natural choice. A convenient parametric prior is given by

$$\begin{aligned} (\pi_1, \dots, \pi_K) &\sim \text{Dirichlet}(\alpha, \dots, \alpha), \quad \alpha > 0 \\ \boldsymbol{\theta}_{W,k} \mid \boldsymbol{\mu}_W, \boldsymbol{\Sigma}_W &\stackrel{\text{iid}}{\sim} \text{MVN}(\boldsymbol{\mu}_W, \boldsymbol{\Sigma}_W), \quad k = 1, \dots, K \\ \boldsymbol{\theta}_{B,k,l} \mid \boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B &\stackrel{\text{iid}}{\sim} \text{MVN}(\boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B), \quad k < l = 1, \dots, K, \end{aligned} \quad (16)$$

where $\boldsymbol{\theta}_{W,k}$ and $\boldsymbol{\theta}_{B,k,l}$ denote the vectors of within- and between-neighborhood parameters, $\boldsymbol{\mu}_W$ and $\boldsymbol{\mu}_B$ are mean vectors, and $\boldsymbol{\Sigma}_W$ and $\boldsymbol{\Sigma}_B$ are diagonal variance-covariance matrices. To reduce the number of parameters, it is convenient to assume that the between-neighborhood parameter vectors $\boldsymbol{\theta}_{B,k,l}$ are constant across pairs of neighborhoods, that is, $\boldsymbol{\theta}_{B,k,l} \equiv \boldsymbol{\theta}_B$, $k < l = 1, \dots, K$. In R package **hergm**, parametric priors can be specified by specifying the option `parametric` of function `hergm`:

```
R> hergm(formula, parametric = TRUE, max_number = 5)
```

where `formula` is a formula of the form `network ~ model` and `max_number` is the number of neighborhoods K . Examples of formulae are given in Sections 3.3 and 3.4. The default of option `parametric` is `FALSE` while the default of option `max_number` is the number of nodes n .

If the number of neighborhoods K is unknown, it is convenient to use nonparametric priors, motivated in part by the desire to sidestep the issue of selecting K and in part by the desire to encourage a small number of (non-empty) neighborhoods and parameters while allowing for a large number of (non-empty) neighborhoods and parameters. A nonparametric prior can be constructed as follows. Let π_k , $k = 1, 2, \dots$ be the prior probabilities of belonging to neighborhoods $k = 1, 2, \dots$, respectively. The probabilities π_k , $k = 1, 2, \dots$ are obtained by first generating draws from a $\text{Beta}(1, \alpha)$ distribution,

$$V_k \mid \alpha \stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha), \quad k = 1, 2, \dots, \quad \alpha > 0, \quad (17)$$

and then generating π_k , $k = 1, 2, \dots$ by stick-breaking. Informally speaking, stick-breaking starts with a stick of length 1, breaks off a piece of length V_1 from the stick of length 1 and sets $\pi_1 = V_1$, then breaks off a piece of length $V_2(1 - V_1)$ from the remaining stick of length $1 - V_1$ and sets $\pi_2 = V_2(1 - V_1)$, and so forth. Thus, the probabilities π_k , $k = 1, 2, \dots$ are given by

$$\begin{aligned} \pi_1 &= V_1 \\ \pi_k &= V_k \prod_{j=1}^{k-1} (1 - V_j), \quad k = 2, 3, \dots \end{aligned} \quad (18)$$

The stick-breaking construction of the probabilities π_k , $k = 1, 2, \dots$ implies that some probabilities π_k are large but most of them are small, so that the nonparametric prior encourages a small number of (non-empty) neighborhoods and parameters while allowing for a large number of (non-empty) neighborhoods and parameters. More details on nonparametric priors can be found in the classic paper by [Ferguson \(1973\)](#) and modern reviews by [Ramamoorthi and Srikanth \(2007\)](#) and [Teh \(2010\)](#). The parameters of neighborhoods are assumed to have multivariate Gaussian priors:

$$\begin{aligned} \boldsymbol{\theta}_{W,k} \mid \boldsymbol{\mu}_W, \boldsymbol{\Sigma}_W &\stackrel{\text{iid}}{\sim} \text{MVN}(\boldsymbol{\mu}_W, \boldsymbol{\Sigma}_W), \quad k = 1, 2, \dots \\ \boldsymbol{\theta}_B \mid \boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B &\sim \text{MVN}(\boldsymbol{\mu}_B, \boldsymbol{\Sigma}_B), \end{aligned} \quad (19)$$

where we assume that the between-neighborhood parameter vectors are constant across pairs of neighborhoods in order to reduce the number of parameters. In R package **hergm**, nonparametric priors can be specified by specifying the option `parametric` of function `hergm` as follows:

```
R> hergm(formula, parametric = FALSE)
```

We note that the default of option `parametric` is `FALSE`, so that the option `parametric` can be left unspecified for nonparametric priors. The so-called concentration parameter α , the mean vectors $\boldsymbol{\mu}_B$ and $\boldsymbol{\mu}_W$, and the inverse variance-covariance matrices $\boldsymbol{\Sigma}_B^{-1}$ and $\boldsymbol{\Sigma}_W^{-1}$ are hyper-parameters that need to be specified. In practice, it may not be obvious how the hyper-parameters should be chosen. In such situations, it is natural to specify hyper-priors for the hyper-parameters that cover a wide range of possible values of the hyper-parameters rather than specifying the hyper-parameters themselves. By default, the R package **hergm** specifies hyper-priors for the hyper-parameters and prints them to the R console, so that users do not need to specify hyper-parameters.

3.7. Comparing ERGMs and HERGMs

In practice, a natural question to ask is whether one should use ERGMs or HERGMs. We discuss some advantages of using HERGMs rather than ERGMs. It is worth noting that there is one disadvantage of using HERGMs rather than ERGMs: when the neighborhood memberships of most or all nodes are unobserved, then the Bayesian approach to HERGMs tends to be slower than the Bayesian approach to ERGMs (Caimo and Friel 2011) implemented in R package **Bergm** (Caimo and Friel 2014) and much slower than the maximum likelihood approach to ERGMs (Hunter and Handcock 2006) implemented in R package **ergm** (Hunter *et al.* 2008b). A more detailed discussion of the computing time of HERGMs and approaches to estimating HERGMs from large networks can be found in Section 9.

Local dependence: Scientific, probabilistic, and statistical advantages. There are important conceptual and theoretical reasons that suggest to use HERGMs with local dependence rather than ERGMs with global dependence. We refer readers to Section 3.1, where we cite scientific reasons (e.g., Pattison and Robins 2002), probabilistic advantages (e.g., weak dependence), and statistical advantages (e.g., consistency of estimators).

ERGMs are special cases of HERGMs and HERGMs tend to be superior to ERGMs in terms of goodness-of-fit. Any ERGM – with any set of model terms – can be viewed as a special case of an HERGM: For example, an ERGM with `edges` and `triangle` terms can be viewed as a special case of an HERGM with `edges_ij` and `triangle_ijk` terms which assumes that all nodes belong to the same neighborhood with probability 1. Given any data set and any set of model terms, an HERGM can be expected to fit the data at least as well as the ERGM that is nested in the HERGM, because the HERGM includes all ERGM distributions – that is, all conditional exponential-family distributions of the form $P_{\theta}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z})$ under which all nodes belong to the same neighborhood – and all more-than-one-neighborhood HERGM distributions – that is, all conditional exponential-family distributions of the form $P_{\theta}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z})$ under which the nodes belong to more than one neighborhood. Under some of the more-than-one-neighborhood HERGM distributions, the observed data may have a higher probability than under the one-neighborhood ERGM distributions. In fact, it is not only possible, but probable that some of the more-than-one-neighborhood HERGM distributions assign a higher probability to the observed data than the one-neighborhood ERGM distributions, because networks are local in nature (e.g., Pattison and Robins 2002) and HERGMs with local dependence respect the local nature of networks. Therefore, one could argue that HERGMs are preferable to ERGMs unless the assumption that all nodes belong to the same neighborhood is satisfied. The assumption that all nodes belong to the same neighborhood may well be satisfied in small networks (e.g., when a terrorist network consists of a single terrorist cell), but it may not be satisfied in large networks (e.g., when a terrorist network consists of more than one terrorist cell). The applications in Sections 8 demonstrate that “large” needs not be very “large” at all: HERGMs can outperform ERGMs in terms of goodness-of-fit when the number of nodes is as small as 17, as the terrorist network in Section 8.2 demonstrates.

In practice, an important concern is that – while HERGMs tend to be superior to ERGMs in terms of goodness-of-fit – HERGMs are more complex than ERGMs and might overfit in the sense that HERGMs place nodes with high posterior probability in more than one neighbor-

hood when data are generated by an ERGM – that is, by an HERGM with one neighborhood. We note that HERGMs with nonparametric priors as described in Section 3.6 discourage large numbers of neighborhoods and parameters and thus penalize model complexity, but HERGMs can nonetheless overfit. To shed light on the issues of under- and overfitting, we conduct two small simulation studies in Section 7 by using the generic simulation and estimation methods described in Sections 4–6. The results of the two simulation studies suggest that HERGMs can be used instead of ERGMs without being too concerned with under- and overfitting.

HERGMs admit more and simpler model terms than ERGMs. An additional advantage is that HERGMs admit more and simpler model terms than ERGMs: e.g., while ERGMs with (global) triangle terms induce strong dependence and tend to lead to model degeneracy, HERGMs with (local) triangle terms induce weak dependence as long as all neighborhoods are small and hence tend to be non-degenerate and useful in practice when ERGMs with (global) triangle terms are near-degenerate and useless (see, e.g., Sections 8.2 and 8.3). Therefore, HERGMs offer researchers the choice of more model terms than ERGMs and the choice of model terms that are simpler than the geometrically weighted model terms which are widely used in the ERGM literature (e.g., Lusher, Koskinen, and Robins 2013) but which are hard to interpret, as discussed in the seminal article of Snijders *et al.* (2006, p. 149).

HERGMs can detect interesting structure. Last, but not least, HERGMs can detect interesting structure in networks. An example is given by the terrorist network in Section 8.2, where HERGMs discover three groups of terrorists corresponding to the three safehouses at which the terrorists hid. A second example is given by the friendship network in Section 8.3, where HERGMs reveal the transitive structure underlying the friendship network.

4. Model estimation

The models with local dependence implemented in R package **hergm** are estimated by Bayesian methods, that is, inference concerning parameters and unobserved neighborhood memberships is based on the posterior. The posterior takes the form

$$p(\alpha, \boldsymbol{\mu}_B, \boldsymbol{\mu}_W, \boldsymbol{\Sigma}_B^{-1}, \boldsymbol{\Sigma}_W^{-1}, \boldsymbol{\pi}, \boldsymbol{\theta}_B, \boldsymbol{\theta}_W, \mathbf{z} \mid \mathbf{x}) \propto p(\alpha, \boldsymbol{\mu}_B, \boldsymbol{\mu}_W, \boldsymbol{\Sigma}_B^{-1}, \boldsymbol{\Sigma}_W^{-1}, \boldsymbol{\pi}, \boldsymbol{\theta}_B, \boldsymbol{\theta}_W) \times \mathbb{P}_{\boldsymbol{\pi}}(\mathbf{Z} = \mathbf{z}) \mathbb{P}_{\boldsymbol{\theta}}(\mathbf{X} = \mathbf{x} \mid \mathbf{Z} = \mathbf{z}), \quad (20)$$

where $p(\alpha, \boldsymbol{\mu}_B, \boldsymbol{\mu}_W, \boldsymbol{\Sigma}_B^{-1}, \boldsymbol{\Sigma}_W^{-1}, \boldsymbol{\pi}, \boldsymbol{\theta}_B, \boldsymbol{\theta}_W)$ is a prior as discussed in Section 3.6. The posterior can be approximated by sampling from the posterior using auxiliary-variable Markov chain Monte Carlo (MCMC) algorithms. A detailed description of auxiliary-variable MCMC algorithms would be too space-consuming and can be found in Schweinberger and Handcock (2015). We focus here on how the auxiliary-variable MCMC algorithm implemented in R package **hergm** can be used in practice. We first discuss how samples from the posterior can be generated by using the function **hergm** (Section 4.1) and how the convergence to the posterior can be assessed by using the **mcmc.diagnostics** method (Section 4.2). We then turn to the methods **summary**, **print**, and **plot**, which summarize samples from the posterior (Section 4.3). Last, but not least, we discuss two advanced topics, the label-switching problem (Section 4.4) and parallel computing (Section 4.5). We note that generic goodness-of-fit and simulation methods are discussed in Sections 5 and 6, respectively.

4.1. Sampling from the posterior

A sample of parameters and unobserved neighborhood memberships from the posterior can be generated by the function `hergm`. The function `hergm` can be used as follows:

```
R> object <- hergm(formula, sample_size = 1e+5)
```

where `formula` is a formula of the form `network ~ model` and `sample_size` is the size of the sample from the posterior. Examples of formulae are given in Sections 3.3 and 3.4. The default of option `sample_size` is 100,000. The function `hergm` returns an object of class ‘`hergm`’. All other user-level functions of R package `hergm` accept objects of class ‘`hergm`’ as arguments. The components of an object of class ‘`hergm`’ are described in the help function of function `hergm`: see `?hergm`. By default, the function `hergm` checks whether the auxiliary-variable MCMC algorithm implemented in function `hergm` has converged to the posterior. If there are signs of non-convergence, `hergm` warns users of possible non-convergence. We discuss convergence checks in Section 4.2. A summary of an object of class ‘`hergm`’ is provided by the methods `summary`, `print`, and `plot`. We discuss these methods in Section 4.3. Before doing so, we mention additional options that are useful for dealing with known and unknown neighborhood memberships.

Sampling from the posterior: Known neighborhood memberships

If some or all neighborhood memberships are known, one should take advantage of them. The most important example of networks with known neighborhood memberships are multilevel networks (Lazega and Snijders 2016). A classic example of a multilevel network is a network consisting of friendships among students within and between schools, where the schools are regarded as neighborhoods and the neighborhood memberships are known as long as it is known which student attended which school. The function `hergm` allows users to use known neighborhood memberships as follows:

```
R> object <- hergm(formula, indicator = indicator)
```

where `indicator` is a vector of length n and the elements of `indicator` are either integers between 1 and n (known neighborhood memberships) or NAs (unknown neighborhood memberships). The default of option `indicator` is `NULL`, that is, all neighborhood memberships are assumed to be unknown. Two examples are shown in Section 8.2, where the following two neighborhood membership vectors are specified:

```
R> indicator <- c(rep.int(1, 9), rep.int(2, 5), rep.int(1, 3))
```

and

```
R> indicator <- c(rep.int(NA, 9), rep.int(2, 5), rep.int(NA, 3))
```

The first example specifies all neighborhood memberships, whereas the second example specifies some but not all neighborhood memberships. The function `hergm` respects known neighborhood memberships and keeps them fixed while inferring the unknown neighborhood memberships.

Sampling from the posterior: Unknown neighborhood memberships

If some or all neighborhood memberships are unknown, the unknown neighborhood memberships are inferred along with the parameters of the model. To do so, one can use either parametric or nonparametric priors as discussed in Section 3.6.

If the number of neighborhoods K is known, parametric priors can be used by specifying the option `parametric` of function `hergm` as follows:

```
R> object <- hergm(formula, parametric = TRUE, max_number = 5)
```

where `max_number` is the number of neighborhoods K . The default of option `parametric` is `FALSE` while the default of option `max_number` is the number of nodes n .

If the number of neighborhoods K is unknown, it is convenient to use nonparametric priors as discussed in Section 3.6. If nonparametric priors are used, it has computational advantages to truncate them. We follow an approach advocated by [Ishwaran and James \(2001\)](#) and adapted by [Schweinberger and Handcock \(2015\)](#) and truncate nonparametric priors by specifying an upper bound K_{\max} on the number of neighborhoods K . In R package `hergm`, one can truncate nonparametric priors by specifying the option `max_number` of function `hergm`:

```
R> object <- hergm(formula, parametric = FALSE, max_number = 5)
```

If `parametric = FALSE`, the option `max_number` is an upper bound K_{\max} on the number of neighborhoods K . The default of option `max_number` is the number of nodes n , but smaller upper bounds result in reductions in computing time and are therefore preferable.

4.2. Convergence to the posterior

The convergence of the auxiliary-variable Markov chain Monte Carlo algorithm to the posterior can be assessed by using the method `mcmc.diagnostics`. The function `hergm` calls `mcmc.diagnostics` by default and hence users do not need to call `mcmc.diagnostics` unless users change the object of class ‘`hergm`’ returned by function `hergm` (e.g., by reducing the sample from the posterior). The generic method `mcmc.diagnostics` accepts an object of class ‘`hergm`’ as argument and can be used as follows:

```
R> mcmc.diagnostics(object)
```

The function returns the convergence checks of [Raftery and Lewis \(1996\)](#) implemented in function `mcgibbsit` of R package `mcgibbsit` along with trace plots of the parameters. The help function of R package `hergm` contains details on the convergence diagnostics returned by function `mcgibbsit`: see `?mcgibbsit`. If there are signs of non-convergence, the function `hergm` warns users of possible non-convergence. In addition, the function `hergm` stores the convergence checks of [Raftery and Lewis \(1996\)](#) in the component `mcmc.diagnostics` of the object of class ‘`hergm`’ returned by function `hergm`. These convergence checks can be retrieved from the object of class ‘`hergm`’ as follows:

```
R> object$mcmc.diagnostics
```

A detailed example can be found in Section 8.1, in particular see Figure 4 in Section 8.1.

4.3. Summary of the posterior

A sample of parameters and unknown neighborhood memberships from the posterior can be summarized by using the `summary` method. The `summary` method accepts an object of class ‘`hergm`’ as argument and can be used as follows:

```
R> summary(object)
```

To provide a summary of a sample from the posterior, the method `summary` relies on the two methods `print` and `plot`. The `print` method prints a summary of a sample of parameters from the posterior whereas the `plot` method plots a summary of a sample of unknown neighborhood memberships from the posterior. We describe them in turn.

Summary of the posterior: Parameters

The `print` method prints a summary of a sample of parameters from the posterior and can be used as follows:

```
R> object
```

The summary consists of the 2.5%, 50%, and 97.5% posterior quantiles of the parameters. The posterior medians (50% posterior quantiles) can be used as estimates of the parameters, while the 95% posterior credibility intervals (based on the 2.5% and 97.5% posterior quantiles) can be used to assess the posterior uncertainty about the parameters. An example can be found in Section 8.1.

Summary of the posterior: Unknown neighborhood memberships

The `plot` method plots a summary of a sample of unknown neighborhood memberships from the posterior and can be used as follows:

```
R> plot(object)
```

The plot represents the posterior membership probabilities of nodes by pie charts centered at the positions of the nodes. Examples are given by Figures 5, 8, and 11 in Sections 8.1, 8.2, and 8.3, respectively.

4.4. Advanced topics: Label-switching problem

The Bayesian auxiliary-variable MCMC algorithm implemented in function `hergm` gives rise to the so-called label-switching problem, which is well-known in the Bayesian literature on finite mixture models and related models (see, e.g., Stephens 2000). The label-switching problem is rooted in the invariance of the likelihood function to permutations of the labels of neighborhoods, that is, distinct labelings of the neighborhoods can give rise to the same value of the likelihood function. The posterior, which is proportional to the likelihood times the prior, is therefore permutation-invariant whenever the prior is permutation-invariant. The parametric priors implemented in R package `hergm` are permutation-invariant and hence give rise to permutation-invariant posteriors, whereas the nonparametric priors are not permutation-invariant and hence do not give rise to permutation-invariant posteriors. However, in many

applications the likelihood appears to dominate the prior and the posterior appears to be almost permutation-invariant even when nonparametric priors are used. As a result, when Bayesian MCMC algorithms are used to sample from the posterior, the labeling of the neighborhoods can switch from iteration to iteration, both under parametric and nonparametric priors. To summarize a MCMC sample of neighborhood-dependent parameters and unknown neighborhood memberships from the posterior, it is therefore advisable to undo the label-switching in the MCMC sample.

An attractive approach to undo the label-switching in the sample is by following the Bayesian decision-theoretic approach of [Stephens \(2000\)](#), which involves choosing a loss function and minimizing the posterior expected loss. Two loss functions are implemented in function `hergm` and can be chosen by using the option `relabel` of `hergm`: the loss function of [Schweinberger and Handcock \(2015\)](#) (`relabel = 1`) and the loss function of [Peng and Carvalho \(2016\)](#) (`relabel = 2`). The first loss function seems to be superior in terms of the reported posterior neighborhood probabilities, but is more expensive in terms of computing time. A rule of thumb is to use the first loss function when `max_number < 10` and to use the second loss function otherwise.

The label-switching in the sample can be undone by calling function `hergm` with either `relabel = 1` or `relabel = 2`. An example is given by

```
R> object <- hergm(formula, max_number = 5, relabel = 1)
```

The option `relabel = 1` is the default as long as `max_number < 10`, otherwise `relabel = 2` is the default. We note that the relabeling algorithm converges to a local minimum of the posterior expected loss, therefore it is advisable to run the relabeling algorithm multiple times using starting values chosen at random. To do so, one can take advantage of the option `number_runs` of function `hergm`:

```
R> object <- hergm(formula, max_number = 5, relabel = 1, number_runs = 3)
```

The option `number_runs = 3` ensures that the relabeling algorithm is run three times with starting values chosen at random. The default of option `number_runs` is 1. Examples are given in Sections 7, 8.1, and 8.2.

An alternative approach, which sidesteps the label-switching problem rather than solving it, is to focus on the posterior of functions of neighborhood memberships Z_1, \dots, Z_n that are invariant to the labeling of neighborhoods, as suggested by [Nowicki and Snijders \(2001\)](#). It is worth noting that focusing on functions of Z_1, \dots, Z_n that are invariant to the labeling of neighborhoods is restrictive, because not all functions of Z_1, \dots, Z_n are invariant to the labeling of neighborhoods and because the approach fails to undo the label-switching in the sample of neighborhood-dependent parameters, therefore summaries of the sample of neighborhood-dependent parameters may be problematic. However, the alternative approach is simple and its computing time is quadratic in the number of nodes n . If the number of neighborhoods K is large (e.g., $K = n$), then the computing time of the alternative approach is lower than the computing time of the relabeling algorithm `relabel = 1`, whose computing time is factorial in $K = n$ ([Schweinberger and Handcock 2015](#)). A simple example of a function of Z_1, \dots, Z_n that is invariant to the labeling of neighborhoods is given by the

indicator function

$$\mathbb{1}_{Z_i=Z_j} = \begin{cases} 1 & \text{if } Z_i = Z_j \\ 0 & \text{otherwise,} \end{cases} \quad (21)$$

that is, $\mathbb{1}_{Z_i=Z_j}$ is an indicator of whether nodes i and j are members of the same neighborhood; note that $\mathbb{1}_{Z_i=Z_j}$ does not depend on the labeling of the neighborhoods. The posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$ can be estimated by the corresponding sample proportions. A simple approach to visualizing the estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$ would be to construct a weighted graph, where the weights of the edges are the estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$. However, the resulting graphs would be dense, that is, the graphs would have large numbers of edges, and it is hard to discover interesting structure in dense graphs. An alternative that tends to produce sparse graphs is based on thresholding the estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$ as follows. Given estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$, one chooses a threshold close to 1 and constructs a graph by putting an edge between nodes i and j if and only if the estimate of the posterior probability of $\mathbb{1}_{Z_i=Z_j}$ exceeds the threshold. If the threshold is close to 1, the resulting graph tends to be sparse and shows which pairs of nodes are in the same neighborhood with high probability. In practice, multiple thresholds can be used and the resulting graphs can be compared. Such graphs can be produced by using the option `relabel = 3` of function `hergm`:

```
R> object <- hergm(formula, max_number = 5, relabel = 3)
```

and by using the `plot` method with the option `threshold`:

```
R> plot(object, threshold = c(.7, .8, .9))
```

The default of option `threshold` is `c(.7, .8, .9)`. The `plot` method with option `threshold = c(.7, .8, .9)` generates three graphs corresponding to the three thresholds .7, .8, and .9. It is possible to specify as many thresholds as desired, but each of the thresholds must be a real number between 0 and 1. An example is given by Figure 11 in Section 8.3.

4.5. Advanced topics: Parallel computing

If a multi-core computer or computing cluster is available, one can take advantage of parallel computing to reduce computing time. To do so, the option `parallel` of function `hergm` can be used. An example is given by

```
R> object <- hergm(formula, max_number = 5, parallel = 10)
```

The option `parallel` is an integer indicating the number of cores of the multi-core computer or computing cluster on which `hergm` is run. The default of option `parallel` is 1, which implies that `hergm` is run on one core. If the user specifies `parallel = 10`, the function `hergm` conducts 10 runs in parallel and combines the results. It is well-known that it is best to conduct one long MCMC run rather than multiple short MCMC runs (e.g., Geyer 1992), but parallel computing can sometimes enable the statistical analysis of large networks that would be infeasible without parallel computing.

5. Model goodness-of-fit

Hunter, Goodreau, and Handcock (2008a) argued that assessing the goodness-of-fit of models is important, because many models with global dependence place most probability mass on graphs that have either almost no edges or almost all possible edges and the goodness-of-fit of such models may be unacceptable. In the Bayesian framework, it is natural to assess the goodness-of-fit of models by generating posterior predictions of networks (Schweinberger and Handcock 2015).

Posterior predictions of networks can be generated by using the `gof` method as follows:

```
R> gof(object)
```

The `gof` method accepts an object of class ‘`hergm`’ as argument and generates posterior predictions of networks by using the `simulate` method described in Section 6. The posterior predictions of networks are compared to the observed network in terms of degrees, geodesic distances, and other interesting functions of networks. The results are presented in the form of plots that resemble the goodness-of-fit plots suggested by Hunter *et al.* (2008a) and implemented in R package `ergm` (Hunter *et al.* 2008b). Examples are given by Figures 6 and 7 in Section 8.2 and Figures 9 and 10 in Section 8.3.

6. Simulation of networks

The simulation of networks from specified models can be useful with a view to (a) conducting simulation studies and (b) assessing the goodness-of-fit of a model estimated by function `hergm`. We focus on (a) and note that (b) is discussed in Section 5.

A network can be simulated by using the `simulate` method as follows:

```
R> simulate(object)
```

where `object` is either an object of class ‘`hergm`’ or a formula of the form `network ~ model`. Examples of formulae are given in Sections 3.3 and 3.4. If the argument `object` of function `simulate` is an object of class ‘`hergm`’, the function simulates networks from the model stored in `object`, otherwise it simulates networks from the model specified by the formula of the form `network ~ model`. The `simulate` method returns the simulated networks in the form of edge lists, that is, for each simulated network, it returns a list of the pairs of nodes with an edge in the simulated network. An example is given in Section 7.

7. Simulation studies: Under- and overfitting

We argued in Section 3.7 that ERGMs are special cases of HERGMs with one neighborhood and that HERGMs tend to be superior to ERGMs in terms of goodness-of-fit. However, while HERGMs might have an advantage over ERGMs in terms of goodness-of-fit, HERGMs are more complex than ERGMs and might overfit in the sense that HERGMs place nodes with high posterior probability in more than one neighborhood when data are generated by an ERGM – that is, by an HERGM with one neighborhood. A related question is whether HERGMs underfit, that is, whether HERGMs place nodes with high posterior probability in

one neighborhood when data are generated by HERGMs with two or more neighborhoods. To shed light on the issue of under- and overfitting, we conduct two small simulation studies. Both simulation studies generate $N = 200$ directed networks with $n = 20$ nodes by using MCMC methods (e.g., [Hunter and Handcock 2006](#)). MCMC methods generate networks by starting with an initial network and constructing a Markov chain to sample from a specified model. We generate the initial network by assuming that edges are independent and by generating edges with probability .1 as follows:

```
R> set.seed(0)
R> d <- matrix(rbinom(n = 400, size = 1, prob = .1), 20, 20)
R> d <- as.network(d, directed = TRUE, loops = FALSE)
R> N <- 200
```

Both simulation studies use HERGMs with `edges_ij` and `transitivities_ijk` terms. We note that `transitivities_ijk` terms are special cases of geometrically weighted edgewise shared partner terms with decay parameter 0 (see, e.g., [Hunter 2007](#)).

The first simulation study generates $N = 200$ networks with $n = 20$ nodes by using an HERGM with one neighborhood and `edges_ij` and `transitivities_ijk` terms, which is equivalent to an ERGM with `edges` and `transitivities` terms. We do so as follows:

```
R> indicator <- rep.int(1, 20)
R> eta <- c(-1, -2, 1, 0)
R> edgelist <- simulate.hergm(d ~ edges_ij + transitivities_ijk,
+   max_number = 1, indicator = indicator, eta = eta, sample_size = N)
```

The parameter vector `eta <- c(-1, -2, 1, 0)` implies that the within- and between-neighborhood edge parameters are given by -1 and -2 , respectively, while the within- and between-neighborhood transitive edge parameters are given by 1 and 0 , respectively. We choose the between-neighborhood edge parameter to be lower than the within-neighborhood edge parameter, because between-neighborhood subgraphs tend to be more sparse than within-neighborhood subgraphs. In addition, we assume that the between-neighborhood transitive edge parameter is 0 , because otherwise the model would induce global dependence rather than local dependence. We note that the between-neighborhood parameters cannot affect the simulation results when all nodes belong to the same neighborhood, but the function `hergm` expects that both within- and between-neighborhood parameters are specified when the option `eta` of function `hergm` is used.

The second simulation study generates $N = 200$ networks with $n = 20$ nodes by using an HERGM with two neighborhoods consisting of 10 nodes each and `edges_ij` and `transitivities_ijk` terms as follows:

```
R> indicator <- c(rep.int(1, 10), rep.int(2, 10))
R> eta <- c(-1, -1, -2, 1, 1, 0)
R> edgelist <- simulate.hergm(d ~ edges_ij + transitivities_ijk,
+   max_number = 2, indicator = indicator, eta = eta, sample_size = N)
```

The parameter vector `eta <- c(-1, -1, -2, 1, 1, 0)` implies that the within-neighborhood edge parameters of neighborhoods 1 and 2 are given by -1 and the between-neighborhood

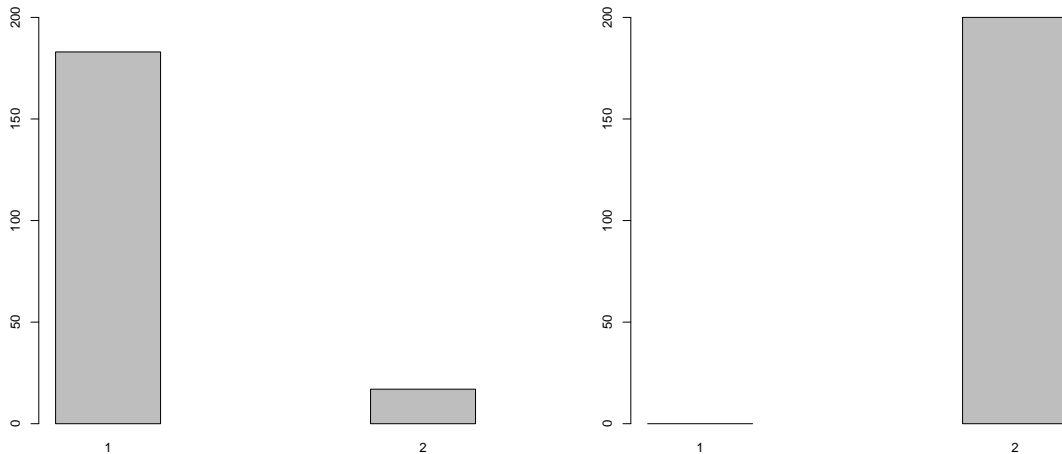


Figure 3: Simulation results: Number of neighborhoods recovered by HERGMs when the true number of neighborhoods is one (left) versus two (right). The two plots suggest that HERGMs can be used instead of ERGMs without being too concerned with under- and overfitting.

edge parameter is given by -2 , while the within-neighborhood transitive edge parameters of neighborhoods 1 and 2 are given by 1 and the between-neighborhood transitive edge parameter is given by 0.

For each simulated network, we determine the number of neighborhoods by first solving the label-switching problem discussed in Section 4.4 and estimating the posterior membership probabilities of nodes and then assigning each node to its maximum-posterior-probability neighborhood and counting the number of neighborhoods. To do so, we use the following R function:

```
R> estimate <- function(i, edgelists) {
+   d <- edgelists$edgelist[[i]]
+   d <- as.network(d, directed = TRUE, matrix.type = "edgelist")
+   object <- hergm(d ~ edges_ij + transitiveties_ijk, max_number = 2,
+     number_runs = 3, verbose = -1)
+   indicator <- vector(length = d$gal$n)
+   for (i in 1:d$gal$n) indicator[i] <- which.max(object$p_i_k[i, ])
+   number <- length(unique(indicator))
+   return(number)
+ }
```

The R function `estimate` extracts the i th simulated network from the list of edge lists (`edgelists$edgelist[[i]]`) and estimates an HERGM with at most two neighborhoods (`max_number = 2`) and `edges_ij` and `transitiveties_ijk` terms, using the default sample size (`sample_size = 1e+5`) and running the default relabeling algorithm (`relabel = 1`) three times with starting values chosen at random (`number_runs = 3`). The option `verbose = -1` ensures that the function `hergm` is silent, that is, `hergm` prints nothing to the R console. To reduce computing time, we run the estimation procedure on a computing cluster with 72 cores by using the function `mclapply` of R package **parallel**:

```
R> library("parallel")
```

```
R> RNGkind("L'Ecuyer-CMRG")
R> number <- mclapply(1:N, estimate, edgelist, mc.cores = 72)
R> number <- as.numeric(unlist(number))
R> barplot(table(factor(number, levels = 1:2)), ylim = c(0, N), space = 2)
```

We note that the function `mclapply` of R package **parallel** does not work on Windows-based operating systems as explained in the help function of `mclapply`: see `?mclapply`. To run the R script above on machines with Windows-based operating systems, one needs to replace `mc.cores = 72` by `mc.cores = 1`.

The plots generated by the R script above are shown in Figure 3. The plots show that when the true number of neighborhoods is one, HERGMs recover the true number of neighborhoods in 92% of the cases, and when the true number of neighborhoods is two, HERGMs recover the true number of neighborhoods in 100% of the cases. Therefore, HERGMs can be used instead of ERGMs without being too concerned with under- and overfitting.

8. Applications

We present three applications to demonstrate the main function `hergm` and the methods `mcmc.diagnostics`, `print`, `plot`, `summary`, and `gof` of R package **hergm** and note that other applications of R package **hergm** can be found in Schweinberger, Petrescu-Prahova, and Vu (2014), Schweinberger and Handcock (2015), and Smith, Calder, and Browning (2016).

8.1. Collaboration network

We start with some of the simplest HERGMs, stochastic block models (Nowicki and Snijders 2001). While stochastic block models are not the primary focus of R package **hergm**, we use them to demonstrate the methods `mcmc.diagnostics`, `print`, `plot`, and `summary`. We exploit the classic data set collected by Kapferer (1972) and used by Nowicki and Snijders (2001). The data correspond to collaborations among 39 workers in a tailor shop in Africa: an undirected edge between workers i and j indicates that i and j collaborated. A plot of the collaboration network is shown in Figure 1(a).

To cluster workers based on the propensity to collaborate with others, we use stochastic block models with `edges_i` terms. Here, we use parameter priors as described in Section 3.6 by using the options `parametric = TRUE` and `max_number = 2` of function `hergm`, because we are interested in partitioning the set of workers into two subsets according to the propensity to collaborate with others. The model with `edges_i` term is estimated as follows:

```
R> data("kapferer", package = "hergm")
R> set.seed(0)
R> object <- hergm(kapferer ~ edges_i, parametric = TRUE, max_number = 2,
+   sample_size = 1e+4)
```

Calling the function

```
R> mcmc.diagnostics(object)
```

results in the following warning:

Warning message:

```
In mcmc.diagnostics.hergm(object) :
```

```
  There are signs of non-convergence: to view details, enter
    'print(object$mcmc.diagnostics)'
  where object is the object returned by function hergm().
```

In other words, there are signs that the auxiliary-variable MCMC algorithm implemented in function `hergm` has either not converged to the posterior or has converged to the posterior but is exploring the posterior very slowly and therefore the sample size might have to be increased. To gain more insight into the convergence issues, we use the `print` method as suggested by function `hergm`:

```
R> object$mcmc.diagnostics
```

The `print` method reveals that the primary issue is the parameter vector $\theta = (\theta_1, \theta_2)$, where θ_1 and θ_2 are the propensities of workers in neighborhoods 1 and 2 to collaborate. The `print` method reports, among other things, the following convergence diagnostics concerning θ_1 and θ_2 :

```
Total Length      = 8000

[1,] Burn-in  Estimation Total Lower bound  Auto-Corr.  Between-Chain
[2,] (M)      (N)          (M+N) (Nmin)      factor (I)  Corr. factor (R)
[3,]
[4,] 36       6459         6495  600         10.8       NA
[5,] 40       6942         6982  600         11.6       NA
[6,] -----
[7,] 40       6942         6982
```

We note that the function `hergm` reduces the sample of size 10,000 – which we requested by using the option `sample_size = 1e+4` of function `hergm` – by discarding the first 2,000 sample points as burn-in – which is the default of the option `posterior.burnin` of function `hergm` – and returns a sample size of 8,000. The first line of the output above confirms that the sample size is 8,000 (see `Total Length = 8000`). The following lines show convergence diagnostics based on the function `mcgibbsit` of R package `mcgibbsit` (Warnes and Burrows 2013). The summary of the convergence diagnostics on line [7,] suggests that the burn-in of size 2,000 may have to be increased by 40 (see column `Burn-in (M)`), but the sample of size 8,000 is larger than the required sample size of 6,942 (see column `Estimation (N)`). However, while the sample size might be large enough, the so-called autocorrelation factor (see column `Auto-Corr. factor (I)`) is more than 10 for both θ_1 (line [4,]) and θ_2 (line [5,]), which triggered the warning; note that the autocorrelation factor is an estimate of the increase of the sample size due to the dependence of the MCMC sample and that, for instance, the autocorrelation factor corresponding to θ_1 is given by $N / N_{\min} = 6,459 / 600 = 10.8$ (line [4,]). It is therefore prudent to increase the sample size. We increase the sample size 10-fold:

```
R> set.seed(0)
```

```
R> object <- hergm(kapferer ~ edges_i, parametric = TRUE, max_number = 2,
+   sample_size = 1e+5, posterior.thinning = 8)
```

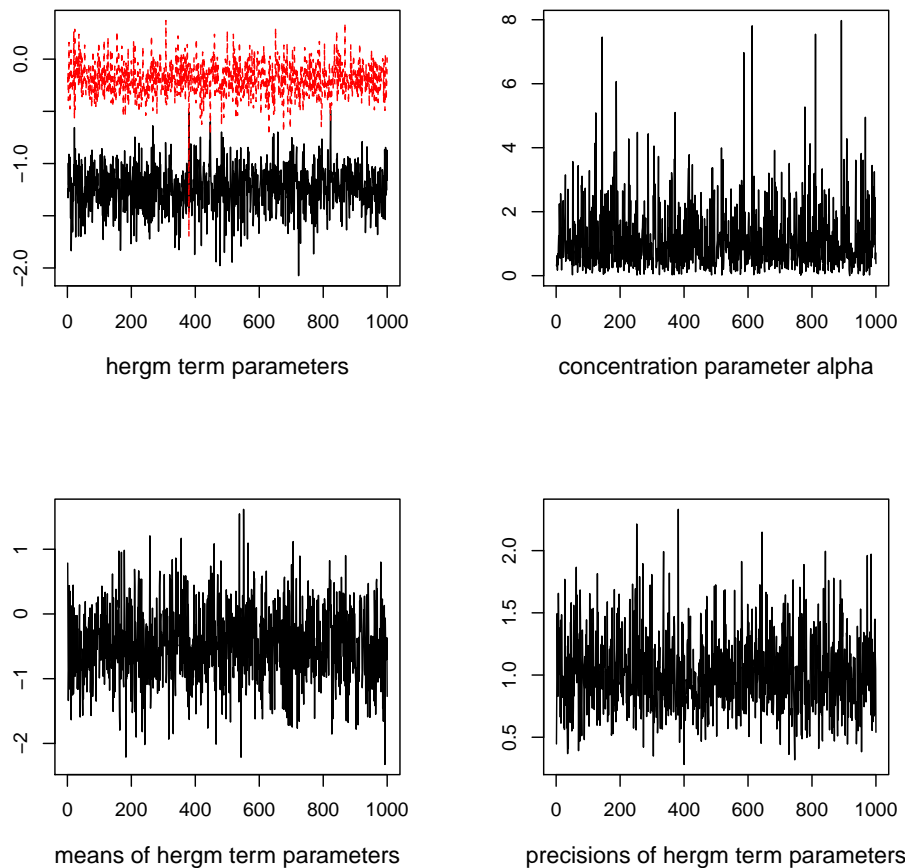



Figure 4: Collaboration network: Trace plots of the sample of parameter vector $\theta = (\theta_1, \theta_2)$ (hergm term parameters), concentration parameter α (concentration parameter alpha), within-neighborhood mean vector μ_W (means of hergm terms parameters), and within-neighborhood inverse variance-covariance matrix Σ_W^{-1} (precisions of hergm terms parameters); note that the between-neighborhood mean vector μ_B and inverse variance-covariance matrix Σ_B^{-1} are not needed in models with term `edges_i`, because the between-neighborhood parameters are functions of the parameters θ_1 and θ_2 .

We note that, to reduce the amount of memory consumed by the function `hergm`, the function `hergm` reduces the sample of size 100,000 (`sample_size = 1e+5`) to a sample of size 10,000 by returning every 10th sample point, then discards the first 2,000 sample points as burn-in (`posterior.burnin = 2000` by default) and returns every 8th sample point of the remaining 8,000 sample points (`posterior.thinning = 8`), giving a sample of size 1,000. We use the option `posterior.thinning = 8`, because it facilitates the plotting of results by reducing the number of sample points to be plotted.

The function `hergm` reports:

```
Convergence check using R function mcgibbsit()...OK
```

The convergence diagnostics can nevertheless be inspected using:

```
R> object$mcmc.diagnostics
```

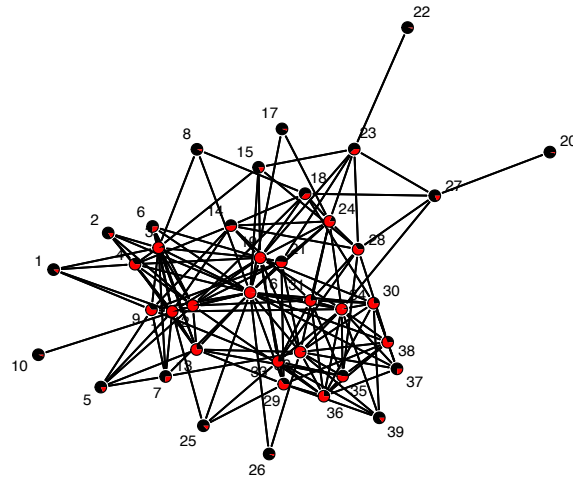


Figure 5: Collaboration network: Observed network with posterior membership probabilities of workers represented by pie charts centered at the positions of workers.

Total Length = 1000

[1,]	Burn-in	Estimation	Total	Lower bound	Auto-Corr. factor	Between-Chain Corr. factor
[2,]	(M)	(N)	(M+N)	(Nmin)	(I)	(R)
[3,]						
[4,]	3	673	676	600	1.13	NA
[5,]	2	620	622	600	1.04	NA
[6,]	-----	-----	-----	-----	-----	-----
[7,]	3	673	676			

Trace plots of the sample of parameters are shown in Figure 4 and confirm that there are no signs of non-convergence.

To summarize the sample of parameters from the posterior, we use the `print` method:

`R> object`

```
=====
Summary of model fit
=====
```

Formula: `kapferer ~ edges_i`

Size of MCMC sample from posterior: 1000

Posterior quantiles	2.5%	50%	97.5%
Concentration parameter alpha:	0.067	0.872	3.625
Mean of parameters of <code>hergm</code> term 1:	-1.621	-0.447	0.652
Precision of parameters of <code>hergm</code> term 1:	0.498	0.976	1.706

hergm term 1: parameter of block 1:	-1.694	-1.229	-0.823
hergm term 1: parameter of block 2:	-0.496	-0.192	0.148

The first three lines of the table correspond to the hyper-parameters α , $\boldsymbol{\mu}_W$, and $\boldsymbol{\Sigma}_W^{-1}$, which are of secondary interest; note that the between-neighborhood mean vector $\boldsymbol{\mu}_B$ and inverse variance-covariance matrix $\boldsymbol{\Sigma}_B^{-1}$ are not needed in models with `edges_i` term, because the between-neighborhood parameters are functions of the parameters θ_1 and θ_2 . The last two lines of the table correspond to the parameters θ_1 and θ_2 , which are of primary interest. To interpret the results concerning θ_1 and θ_2 , note that – under stochastic block models with `edges_i` term – the log odds of the conditional probability of an edge between nodes i and j given $Z_i = z_i$ and $Z_j = z_j$ is given by $\theta_{z_i} + \theta_{z_j}$ (see Section 3.3). The summary of the sample of parameters θ_1 and θ_2 from the posterior consists of the 2.5%, 50%, and 97.5% posterior quantiles of θ_1 and θ_2 . The posterior medians (50% posterior quantiles) -1.229 and -0.192 can be used as estimates of θ_1 and θ_2 , respectively. The posterior uncertainty about θ_1 and θ_2 can be assessed by inspecting the 95% posterior credibility intervals $(-1.694, -0.823)$ and $(-0.496, 0.148)$ of θ_1 and θ_2 , respectively, which are based on the 2.5% and 97.5% posterior quantiles of the two parameters. The 95% posterior credibility intervals of θ_1 and θ_2 do not overlap, suggesting that there are indeed two distinct groups of workers: one group of workers, corresponding to neighborhood 2, seems to be more inclined to collaborate than the other group of workers, corresponding to neighborhood 1.

To inspect these two groups of workers, we use the `plot` method:

```
R> plot(object)
```

The `plot` method generates the plot shown in Figure 5. The plot represents the posterior membership probabilities of workers by pie charts centered at the positions of workers. It is evident that the second group of workers (neighborhood 2, colored red) is at the center of the network whereas the other group of workers (neighborhood 1, colored black) is on the periphery. It is worth noting that there is uncertainty about the neighborhood memberships of some workers who are located neither in the center nor on the periphery of the network.

Last, but not least, we note that the `summary` method could have been used in place of the methods `print` and `plot`:

```
R> summary(object)
```

The `summary` method would have generated both summaries of the sample of parameters and neighborhood memberships.

8.2. Terrorist network

Koschade (2006) collected data on contacts between the 17 terrorists who conducted the Bali, Indonesia bombing in 2002. The terrorist network is shown in Figure 1(b). An undirected edge between terrorists i and j indicates a contact between i and j . The terrorist network consisted of two cells, the main group (terrorists 1–9 and 15–17) and the support group (terrorists 10–14). We take advantage of the known cell structure of the terrorist network to demonstrate how known neighborhood memberships can be used in R package `hergm`. Additional

M1: ERGM with global dependence	(max_number = 1)	265.3
M2: ERGM with known main and support group	(max_number = 2)	60.7
M3: HERGM with known support group	(max_number = 2)	54.6
M4: HERGM without known groups	(max_number = 5)	21.6

Table 3: Terrorist network: Comparison of models M1, M2, M3, and M4. The statistic reported here is the root mean-squared deviation of the posterior predicted number of triangles from the observed number of triangles. The lower the value of the statistic is, the better is the goodness-of-fit of the model. HERGMs with enough neighborhoods (M4) are best in terms of goodness-of-fit, outperforming ERGMs with global dependence (M1) and ERGMs with local dependence (M2) as well as HERGMs with too few neighborhoods (M3).

results, including comparisons with stochastic block models, can be found in [Schweinberger and Handcock \(2015\)](#).

To demonstrate how known neighborhood memberships can be used in R package **hergm**, we consider models with `edges_ij` and `triangle_ijk` terms that make various assumptions about the neighborhood memberships of terrorists. Model M1 assumes that all terrorists are members of the same neighborhood, which is equivalent to an ERGM with global dependence. Model M2 assumes that the terrorist network consists of two neighborhoods, corresponding to the main group and the support group of the terrorist network, which is equivalent to an ERGM with local dependence. Model M3 is an HERGM that assumes that the support group is known but the neighborhood memberships of all other terrorists are unknown. Model M4 is an HERGM that assumes that the neighborhood memberships of all terrorists are unknown.

We estimate models M1, M2, M3, and M4 as follows:

```
R> data("bali", package = "hergm")
R> set.seed(0)
R> indicator <- rep.int(1, 17)
R> object.m1 <- hergm(bali ~ edges_ij + triangle_ijk, max_number = 1,
+   indicator = indicator, sample_size = 2e+5)
R> set.seed(0)
R> indicator <- c(rep.int(1, 9), rep.int(2, 5), rep.int(1, 3))
R> object.m2 <- hergm(bali ~ edges_ij + triangle_ijk, max_number = 2,
+   indicator = indicator, sample_size = 2e+5)
R> set.seed(0)
R> indicator <- c(rep.int(NA, 9), rep.int(2, 5), rep.int(NA, 3))
R> object.m3 <- hergm(bali ~ edges_ij + triangle_ijk, max_number = 2,
+   indicator = indicator, sample_size = 2e+5)
R> set.seed(0)
R> object.m4 <- hergm(bali ~ edges_ij + triangle_ijk, max_number = 5,
+   sample_size = 2e+5, number_runs = 3)
```

Model M4 with at most `max_number = 5` neighborhoods is equivalent to the model used by [Schweinberger and Handcock \(2015\)](#).

We compare these models in terms of goodness-of-fit. The goodness-of-fit of the models can be assessed by using `gof`:

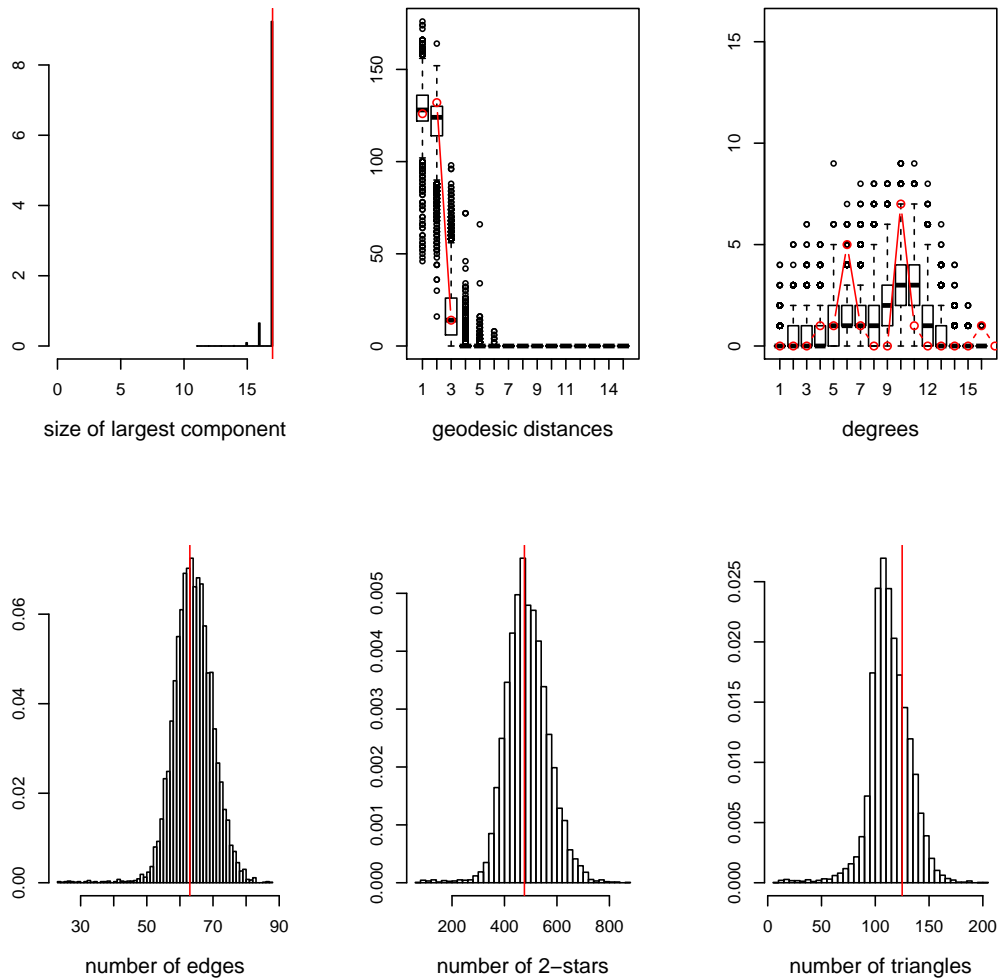


Figure 6: Terrorist network: Goodness-of-fit of model M4 with local dependence; observed values of statistics are indicated by the color red. The distributions are unimodal and centered at the observed values of the statistics.

```
R> g1 <- gof(object.m1)
R> g2 <- gof(object.m2)
R> g3 <- gof(object.m3)
R> g4 <- gof(object.m4)
```

For each of the models, we compute the root mean-squared deviation of the posterior predicted number of triangles from the observed number of triangles as a measure of goodness-of-fit:

```
R> triangles <- summary(bali ~ triangle)
R> d1 <- sqrt(sum((g1$triangles - triangles)^2) / length(g1$triangles))
R> d2 <- sqrt(sum((g2$triangles - triangles)^2) / length(g2$triangles))
R> d3 <- sqrt(sum((g3$triangles - triangles)^2) / length(g3$triangles))
R> d4 <- sqrt(sum((g4$triangles - triangles)^2) / length(g4$triangles))
```

The results are shown in Table 3. The table demonstrates that models M2, M3, and M4 with local dependence are superior to model M1 with global dependence and that HERGMs

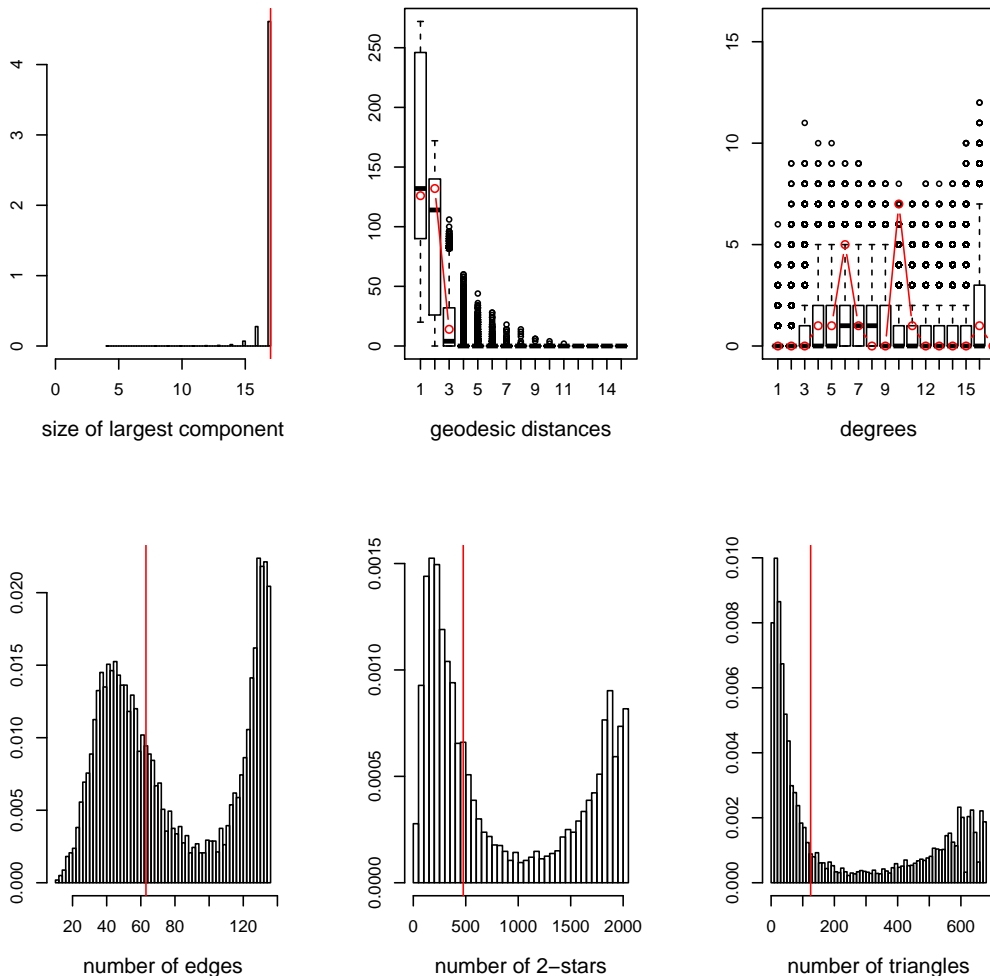


Figure 7: Terrorist network: Goodness-of-fit of model M1 with global dependence; observed values of statistics are indicated by the color red. The distributions are multimodal and place much probability mass on networks that are extreme in terms of the statistics.

with local dependence (M3, M4) outperform both ERGMs with global dependence (M1) and ERGMs with local dependence (M2) in terms of goodness-of-fit. Among all models, model M4 is best in terms of goodness-of-fit: model M4 reduces the root mean-squared deviation of the posterior predicted number of triangles from the observed number of triangles by more than 60% compared with models M2 and M3, underscoring the advantage that HERGMs with local dependence (M4) hold over ERGMs with local dependence (M2) and HERGMs with local dependence but too few neighborhoods (M3).

To compare the goodness-of-fit of the model M4 with local dependence and the model M1 with global dependence in more detail, we use `gof`:

```
R> gof(object.m4)
R> gof(object.m1)
```

The `gof` method produces the goodness-of-fit plots shown in Figures 6 and 7. The plots assess the goodness-of-fit of the models in terms of the size of the largest component (i.e.,

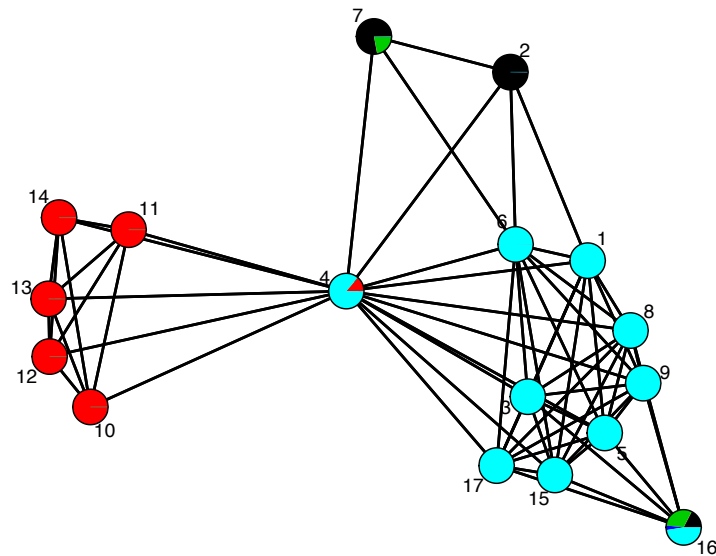


Figure 8: Terrorist network: Observed network with posterior membership probabilities of terrorists represented by pie charts centered at the positions of terrorists.

the size of the largest subset of nodes such that all pairs of nodes are connected by finite paths); the geodesic distances of pairs of nodes (i.e., the lengths of the shortest paths between pairs of nodes); the degrees of nodes; the number of edges; the number of 2-stars; and the number of triangles; note that these statistics are described in more detail by Wasserman and Faust (1994). Comparing Figures 6 and 7 reveals that the model with local dependence is superior to the model with global dependence in terms of goodness-of-fit. In particular, under the model with local dependence most of the probability mass is concentrated around the observed values of the statistics, whereas under the model with global dependence the distributions are multimodal, placing much probability mass on networks that are extreme in terms of the statistics.

To summarize the sample of neighborhood memberships from the posterior, we use `plot`:

```
R> plot(object.m4)
```

The `plot` method produces Figure 8. The figure is consistent with ground truth: the terrorists hid in three safehouses and the three dominant neighborhoods (represented by colors) correspond to the three safehouses (Koschade 2006; Schweinberger and Handcock 2015).

8.3. Friendship network

Van de Bunt (1999) and Van de Bunt, Van Duijn, and Snijders (1999) collected data on friendships between 32 freshmen at a European university at 7 time points. We use the last time point. A directed edge from student i to student j indicates that i considers j to be a “friend” or “best friend”. A plot of the friendship network is shown in Figure 1(c). We use the friendship network to demonstrate the option `relabel = 3` of function `hergm`, which is described in Section 4.4.

To capture both reciprocity and transitivity, we consider a model with `edges_ij`, `mutual_ij`, and `ttriple_ijk` terms. We estimate the model as follows:

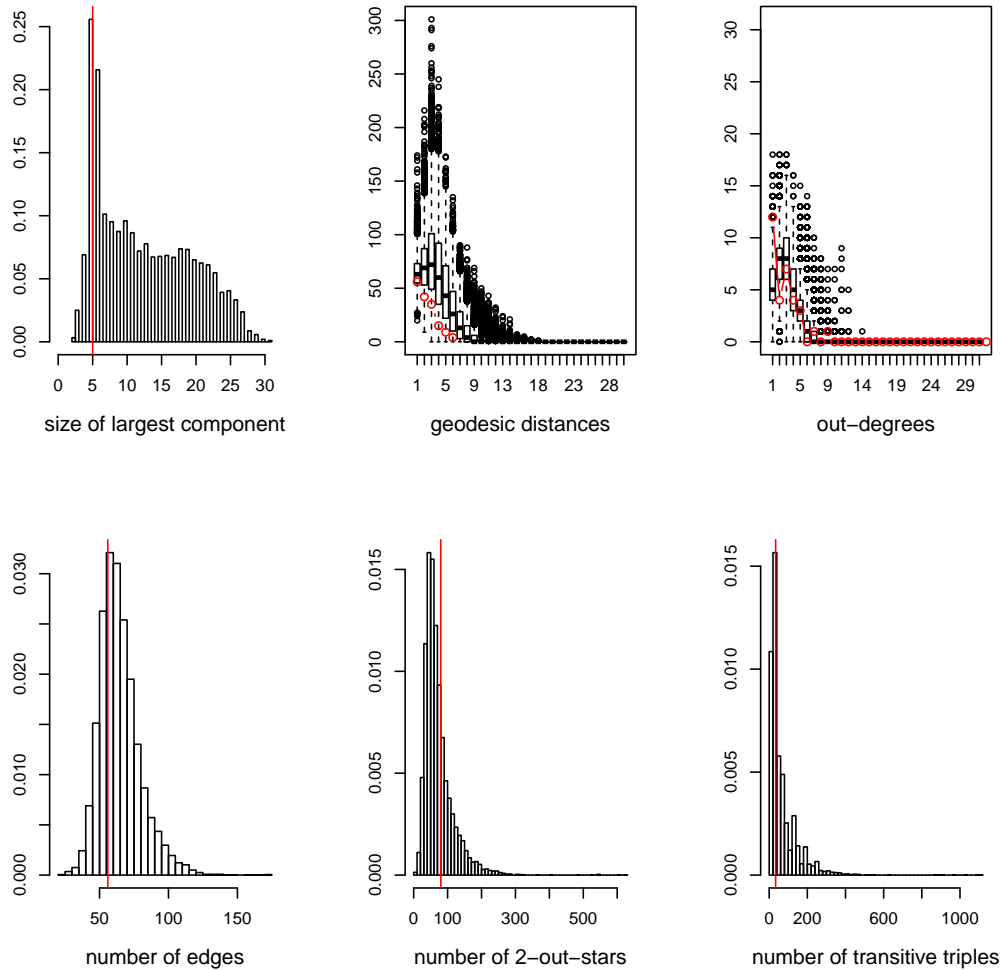


Figure 9: Friendship network: Goodness-of-fit of model with local dependence; observed values of statistics are indicated by the color red. The distributions are unimodal and centered at the observed values of the statistics.

```
R> data("bunt", package = "hergm")
R> set.seed(0)
R> object.local <- hergm(bunt ~ edges_ij + mutual_ij + ttriple_ijk,
+   max_number = 32, relabel = 3, sample_size = 2e+5)
```

We note that `max_number = 32` is the default of option `max_number` and implies that the number of neighborhoods K may be as large as the number of nodes $n = 32$. It can be used when the user is either unable or unwilling to choose an upper bound on the number of neighborhoods K that is smaller than the maximum number of neighborhoods, which is the number of nodes $n = 32$.

We first compare the model with local dependence to the corresponding model with global dependence. To do so, we estimate the model with global dependence as follows:

```
R> set.seed(0)
R> indicator <- rep.int(1, 32)
```

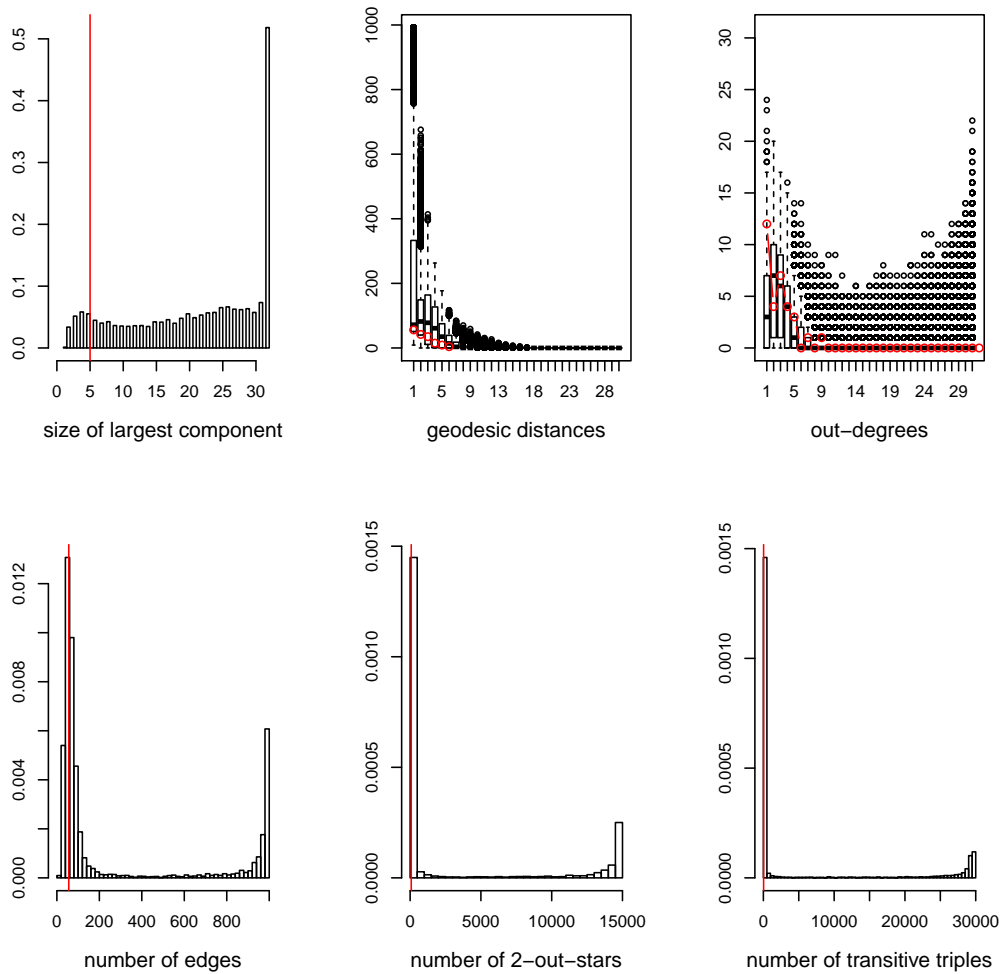



Figure 10: Friendship network: Goodness-of-fit of model with global dependence; observed values of statistics are indicated by the color red. The distributions are multimodal and place much probability mass on networks that are extreme in terms of the statistics.

```
R> object.global <- hergm(bunt ~ edges_ij + mutual_ij + ttriple_ijk,
+   max_number = 1, indicator = indicator, sample_size = 2e+5)
```

We then use the `gof` method to compare the two models in terms of goodness-of-fit:

```
R> gof(object.local)
R> gof(object.global)
```

The resulting goodness-of-fit plots are shown in Figures 9 and 10; note that R package **hergm** compares the goodness-of-fit of models in terms of statistics that resemble the statistics used in Figures 6 and 7 but that are tailored to directed networks rather than undirected networks (see, e.g., Wasserman and Faust 1994). It is evident that the model with local dependence outperforms the model with global dependence in terms of goodness-of-fit: note the multimodal nature of the posterior predictive distributions induced by global dependence and the fact that those distributions place much probability mass on networks that are extreme in terms of the statistics.

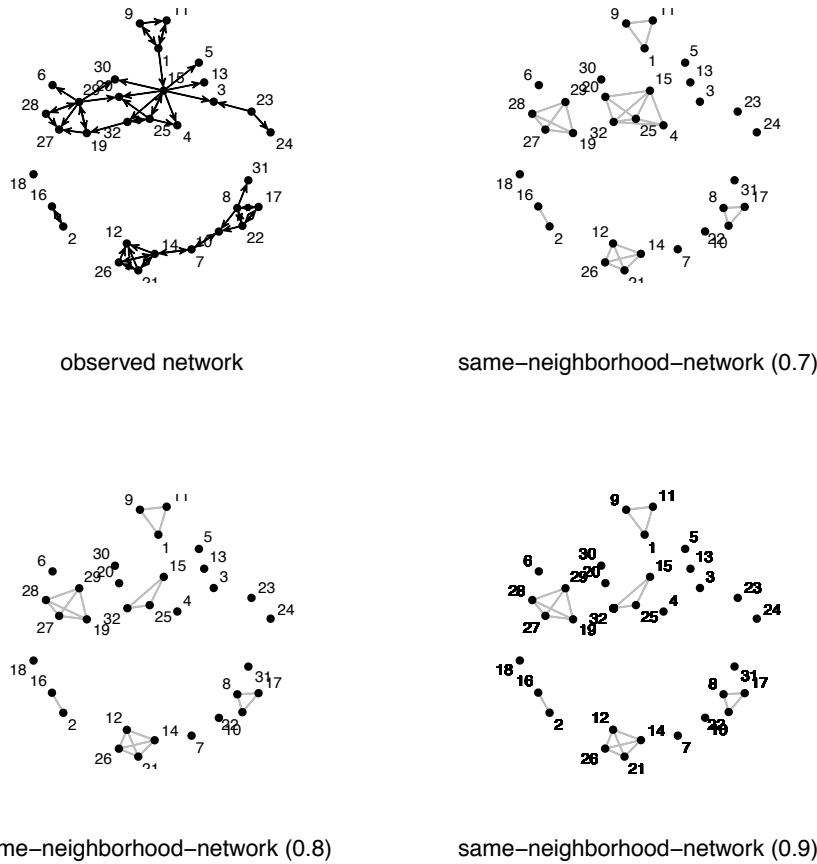


Figure 11: Friendship network: observed network and same-neighborhood graphs obtained by thresholding estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$ with thresholds .7, .8, and .9. All plots use the same coordinates.

To summarize the sample of neighborhood memberships from the posterior, we use `plot`:

```
R> plot(object.local)
```

We note that we have used the option `relabel = 3` of function `hergm`, because `max_number = 32` is large and hence the default relabeling algorithm `relabel = 1` is too time-consuming to be useful (see Section 4.4). The option `relabel = 3` plots estimates of the posterior probabilities of $\mathbb{1}_{Z_i=Z_j}$ by constructing a graph where students i and j are connected by an edge if and only if the estimate of the posterior probability of $\mathbb{1}_{Z_i=Z_j}$ exceeds a threshold selected by the user. We use the default thresholds .7, .8, and .9, which produce three graphs. The three graphs are displayed in Figure 11 along with the observed network. The figure reveals the transitive structure underlying the friendship network and suggests that there is a number of small and close-knit groups of friends who are in the same neighborhood with high posterior probability.

9. Discussion

The R package `hergm` implements a wide range of HERGMs with local dependence, which

are preferable to ERGMs with global dependence on scientific, probabilistic, and statistical grounds and which tend to outperform ERGMs with global dependence in terms of goodness-of-fit.

The models implemented in R package **hergm** can be extended in various ways: for example, local versions of all model terms of R package **ergm** terms could be added to R package **hergm**, including local versions of all geometrically weighted model terms (Snijders *et al.* 2006; Hunter and Handcock 2006; Hunter 2007). In addition, while we focused on model terms for binary random graphs, model terms for non-binary random graphs can be added by building on the work of Krivitsky (2012) and Desmarais and Cranmer (2012).

Last, but not least, it is worth noting that HERGMs can be estimated from networks with up to 100–200 nodes when edge variables are dependent conditional on neighborhood memberships and up to 1,000 nodes when edge variables are independent conditional on neighborhood memberships. An open problem is how HERGMs can be estimated from large networks with thousands or millions of nodes, which is where HERGMs are expected to have the greatest advantage in terms of goodness-of-fit, as discussed in Sections 3.1 and 3.7. We note that the computing time of HERGMs depends on the number of neighborhoods and the size of the largest neighborhood when the neighborhoods are known (which is the case in, e.g., multilevel networks; see Lazega and Snijders 2016). If the neighborhood memberships of some or all nodes are unknown, then the computing time depends in addition on the number of nodes with unknown neighborhood memberships, because the unknown neighborhood memberships of nodes need to be inferred. In practice, it is natural to estimate parameters and unknown neighborhood memberships by cycling through updates of parameters given neighborhood memberships and updates of neighborhood memberships given parameters. Given neighborhood memberships, parameters can be updated along the lines of ERGMs (e.g., Hunter and Handcock 2006; Caimo and Friel 2011), with the important advantage that the loglikelihood function decomposes into neighborhood-dependent loglikelihood functions that can be computed by parallel computing on computing clusters. Therefore, updates of parameters given neighborhood memberships are much less time-consuming for HERGMs with local dependence than for ERGMs with global dependence provided that all neighborhoods are small. The bottleneck of HERGMs in terms of computing time are updates of neighborhood memberships given parameters, which are challenging because updates of neighborhood memberships may depend on intractable neighborhood-dependent loglikelihood functions and the update of the neighborhood membership of one node depends on the neighborhood memberships of other nodes. However, not all is lost: there are promising directions for methods that can estimate HERGMs from large networks with thousands or tens of thousands of nodes. The basic idea is that HERGMs contain stochastic block models as special cases. Stochastic block models admit the estimation of neighborhood memberships from large networks with hundreds of thousands of nodes (e.g., Rohe, Chatterjee, and Yu 2011; Vu *et al.* 2013). Therefore, one can update neighborhood memberships given parameters by adapting methods from the stochastic block model literature. We intend to make such methods available in R package **hergm** in the future.

Acknowledgments

The first author acknowledges support from the National Science Foundation (NSF award DMS-1513644).

References

- Barndorff-Nielsen OE (1978). *Information and Exponential Families in Statistical Theory*. John Wiley & Sons, New York. doi:10.1002/9781118857281.
- Brown L (1986). *Fundamentals of Statistical Exponential Families: with Applications in Statistical Decision Theory*. Institute of Mathematical Statistics, Hayworth.
- Butts CT (2008a). “**network**: A Package for Managing Relational Data in R.” *Journal of Statistical Software*, **24**(2), 1–36. doi:10.18637/jss.v024.i02.
- Butts CT (2008b). “Social Network Analysis with **sna**.” *Journal of Statistical Software*, **24**(6), 1–51. doi:10.18637/jss.v024.i06.
- Caimo A, Friel N (2011). “Bayesian Inference for Exponential Random Graph Models.” *Social Networks*, **33**(1), 41–55. doi:10.1016/j.socnet.2010.09.004.
- Caimo A, Friel N (2014). “**Bergm**: Bayesian Exponential Random Graphs in R.” *Journal of Statistical Software*, **61**(2), 1–25. doi:10.18637/jss.v061.i02.
- Chatterjee S, Diaconis P (2013). “Estimating and Understanding Exponential Random Graph Models.” *The Annals of Statistics*, **41**(5), 2428–2461. doi:10.1214/13-aos1155.
- Denny MJ, Wilson JD, Cranmer S, Desmarais BA, Bhamidi S (2017). **GERGM**: *Estimation and Fit Diagnostics for Generalized Exponential Random Graph Models*. R package version 0.11.2, URL <https://CRAN.R-project.org/package=GERGM>.
- Desmarais SJ, Cranmer BA (2012). “Statistical Inference for Valued-Edge Networks: The Generalized Exponential Random Graph Model.” *PLoS ONE*, **7**(1), 1–12. doi:10.1371/journal.pone.0030136.
- Ferguson T (1973). “A Bayesian Analysis of Some Nonparametric Problems.” *The Annals of Statistics*, **1**(2), 209–230. doi:10.1214/aos/1176342360.
- Frank O, Strauss D (1986). “Markov Graphs.” *Journal of the American Statistical Association*, **81**(395), 832–842. doi:10.2307/2289017.
- Geyer CJ (1992). “Practical Markov Chain Monte Carlo.” *Statistical Science*, **7**(4), 473–483. doi:10.1214/ss/1177011137.
- Handcock MS (2003). “Statistical Models for Social Networks: Inference and Degeneracy.” In R Breiger, K Carley, P Pattison (eds.), *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, pp. 1–12. National Academies Press, Washington, D.C.
- Handcock MS, Hunter DR, Butts CT, Goodreau SM, Morris M (2008). “**statnet**: Software Tools for the Representation, Visualization, Analysis and Simulation of Network Data.” *Journal of Statistical Software*, **24**(1), 1–11. doi:10.18637/jss.v024.i01.
- Hunter DR (2007). “Curved Exponential Family Models for Social Networks.” *Social Networks*, **29**(2), 216–230. doi:10.1016/j.socnet.2006.08.005.

- Hunter DR, Goodreau SM, Handcock MS (2008a). “Goodness of Fit of Social Network Models.” *Journal of the American Statistical Association*, **103**(481), 248–258. doi:10.1198/016214507000000446.
- Hunter DR, Handcock MS (2006). “Inference in Curved Exponential Family Models for Networks.” *Journal of Computational and Graphical Statistics*, **15**(3), 565–583. doi:10.1198/106186006x133069.
- Hunter DR, Handcock MS, Butts CT, Goodreau SM, Morris M (2008b). “**ergm**: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks.” *Journal of Statistical Software*, **24**(3), 1–29. doi:10.18637/jss.v024.i03.
- Ishwaran H, James LF (2001). “Gibbs Sampling Methods for Stick-Breaking Priors.” *Journal of the American Statistical Association*, **96**(453), 161–173. doi:10.1198/016214501750332758.
- Kapferer B (1972). *Strategy and Transaction in an African Factory*. Manchester University Press, Manchester.
- Kolaczyk ED (2009). *Statistical Analysis of Network Data: Methods and Models*. Springer-Verlag, New York.
- Koschade S (2006). “A Social Network Analysis of Jemaah Islamiyah: The Applications to Counter-Terrorism and Intelligence.” *Studies in Conflict and Terrorism*, **29**(6), 559–575. doi:10.1080/10576100600798418.
- Krivitsky PN (2012). “Exponential-Family Models for Valued Networks.” *Electronic Journal of Statistics*, **6**, 1100–1128. doi:10.1214/12-ejs696.
- Krivitsky PN, Handcock MS (2008). “Fitting Position Latent Cluster Models for Social Networks with **latentnet**.” *Journal of Statistical Software*, **24**(5), 1–23. doi:10.18637/jss.v024.i05.
- Krivitsky PN, Handcock MS (2016). **tergm**: *Fit, Simulate and Diagnose Models for Network Evolution Based on Exponential-Family Random Graph Models*. The Statnet Project. R package version 3.4.0, URL <https://CRAN.R-project.org/package=tergm>.
- Lazega E, Snijders TAB (eds.) (2016). *Multilevel Network Analysis for the Social Sciences*. Springer-Verlag, Switzerland. doi:10.1007/978-3-319-24520-1.
- Lusher D, Koskinen J, Robins G (2013). *Exponential Random Graph Models for Social Networks*. Cambridge University Press, Cambridge, UK.
- Morris M, Handcock MS, Hunter DR (2008). “Specification of Exponential-Family Random Graph Models: Terms and Computational Aspects.” *Journal of Statistical Software*, **24**(4), 1–24. doi:10.18637/jss.v024.i04.
- Nowicki K, Snijders TAB (2001). “Estimation and Prediction for Stochastic Blockstructures.” *Journal of the American Statistical Association*, **96**(455), 1077–1087. doi:10.1198/016214501753208735.

- Pattison P, Robins G (2002). “Neighborhood-Based Models for Social Networks.” In RM Stolzenberg (ed.), *Sociological Methodology*, volume 32, pp. 301–337. Blackwell Publishing, Boston.
- Peng L, Carvalho L (2016). “Bayesian Degree-Corrected Blockmodels for Community Detection.” *Electronic Journal of Statistics*, **10**(2), 2746–2779. doi:10.1214/16-ejs1163.
- Raftery AE, Lewis SM (1996). “Implementing MCMC.” In WR Gilks, S Richardson, DJ Spiegelhalter (eds.), *Markov Chain Monte Carlo in Practice*, chapter 7, pp. 115–130. Chapman and Hall, London.
- Ramamoorthi RV, Srikanth KR (2007). “Dirichlet Processes.” In *Encyclopedia of Statistical Sciences*. John Wiley & Sons, New York.
- R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.
- Rohe K, Chatterjee S, Yu B (2011). “Spectral Clustering and the High-Dimensional Stochastic Block Model.” *The Annals of Statistics*, **39**(4), 1878–1915. doi:10.1214/11-aos887.
- Schweinberger M (2011). “Instability, Sensitivity, and Degeneracy of Discrete Exponential Families.” *Journal of the American Statistical Association*, **106**(496), 1361–1370. doi:10.1198/jasa.2011.tm10747.
- Schweinberger M, Handcock MS (2015). “Local Dependence in Random Graph Models: Characterization, Properties and Statistical Inference.” *Journal of the Royal Statistical Society B*, **77**(3), 647–676. doi:10.1111/rssb.12081.
- Schweinberger M, Handcock MS, Luna P (2018). *R Package hergm. Hierarchical Exponential-Family Random Graph Models*. R package version 3.2-1, URL <https://CRAN.R-project.org/package=hergm>.
- Schweinberger M, Petrescu-Prahova M, Vu DQ (2014). “Disaster Response on September 11, 2001 through the Lens of Statistical Network Analysis.” *Social Networks*, **37**, 42–55. doi:10.1016/j.socnet.2013.12.001.
- Schweinberger M, Stewart J (2017). “Finite-Graph Concentration and Consistency Results for Random Graphs with Complex Topological Structures.” arXiv:1702.01812 [math.ST], URL <https://arxiv.org/abs/1702.01812>.
- Shalizi CR, Rinaldo A (2013). “Consistency under Sampling of Exponential Random Graph Models.” *The Annals of Statistics*, **41**(2), 508–535. doi:10.1214/12-aos1044.
- Smith A, Calder CA, Browning CR (2016). “Empirical Reference Distributions for Networks of Different Size.” *Social Networks*, **47**, 24–37. doi:10.1016/j.socnet.2016.03.004.
- Snijders TAB, Pattison PE, Robins GL, Handcock MS (2006). “New Specifications for Exponential Random Graph Models.” *Sociological Methodology*, **36**(1), 99–153. doi:10.1111/j.1467-9531.2006.00176.x.
- Stephens M (2000). “Dealing with Label-Switching in Mixture Models.” *Journal of the Royal Statistical Society B*, **62**(4), 795–809. doi:10.1111/1467-9868.00265.

- Teh YW (2010). “Dirichlet Processes.” In C Sammut, GI Webb (eds.), *Encyclopedia of Machine Learning*. Springer-Verlag.
- Van de Bunt GG (1999). *Friends by Choice. An Actor-Oriented Statistical Network Model for Friendship Networks through Time*. Thesis Publishers, Amsterdam.
- Van de Bunt GG, Van Duijn MAJ, Snijders TAB (1999). “Friendship Networks through Time: An Actor-Oriented Statistical Network Model.” *Computational and Mathematical Organization Theory*, **5**(2), 167–192. doi:10.1023/a:1009683123448.
- Vu DQ, Hunter DR, Schweinberger M (2013). “Model-Based Clustering of Large Networks.” *The Annals of Applied Statistics*, **7**(2), 1010–1039. doi:10.1214/12-aos617.
- Warnes GR, Burrows R (2013). *mcgibbsit: Warnes and Raftery’s MCGibbsit MCMC Diagnostic*. R package version 1.1.0, URL <https://CRAN.R-project.org/package=mcgibbsit>.
- Wasserman S, Faust K (1994). *Social Network Analysis: Methods and Applications*. Cambridge University Press, Cambridge.
- Wasserman S, Pattison P (1996). “Logit Models and Logistic Regression for Social Networks: I. An Introduction to Markov Graphs and p^* .” *Psychometrika*, **61**(3), 401–425. doi:10.1007/bf02294547.

Affiliation:

Michael Schweinberger, Pamela Luna
Department of Statistics
Rice University
6100 Main St - MS-138
Houston, TX 77005, Unites States of America
E-mail: michael.schweinberger@rice.edu