ESSAYS ON BIOINFORMATICS AND SOCIAL NETWORK ANALYSIS: STATISTICAL AND COMPUTATIONAL METHODS FOR COMPLEX SYSTEMS

by

George G. Vega Yon

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(BIOSTATISTICS)

August 2020

Dedication

To my parents, Hilda and George, my children, Mateo and Tomas, and my beloved wife, Valentina.

Acknowledgments

There is no way in which I could ever name all of those who I believe helped me get here. Throughout my life, I have been lucky to have enjoyed the guidance of many inspiring mentors. In high school, Mr. Plácido Castro, who was my math teacher, was one of the first ones who inspired me and encouraged me to follow the academic path. Ever since I met them during my undergrad years, professors Eduardo Fajnzylber and Jorge Fábrega have given me great advice. And I am especially grateful to Jorge, who introduced me to the world of social networks analysis. While many people helped me during my Ph.D. application process, I am certain that I would not be here if it weren't because of Prof. Mike Alvarez. who saw something in me that others didn't, so I'm in debt to him. I was lucky enough to cross paths with Prof. Tom Valente. Working with Tom has made a huge impact both at a personal and professional level. Continuing Jorge's work, he furthered my incursion into the world of social networks. While they didn't officially mentor me, professors Kim Siegmund and Jim Gauderman also provided valuable guidance and support during the whole process. I am also very grateful to my doctoral committee, professors Paul Marjoram, Kayla de la Haye, Paul Thomas, Duncan Thomas, and Emilio Ferrara. Having come from a very different discipline, my committee's patience and selfless dedication have been key for the completion of my doctoral work. Moreover, I am especially thankful to Paul, who has always gone the extra mile with his guidance and support both on a personal and academic level.

While having outstanding mentors has been fundamental to my career, the love, and support that my family and friends have given me have also been invaluable. Our friends, Carlos, Ruby, Camila, Cristian, Alejandro, Maricruz, Sylvia, Esteban, Charlie, and Angela, all of whom my wife

and I think of like our family. My siblings Paola and Bastian, my parents, Hilda and George, and my mother and father-in-law Cecilia and Felipe, while far away, have always been there for us. My children, Mateo and Tomas, who fill my life with love and joy, and my wife, Valentina, who's strength, patience, and selfless love makes a world of difference in my life.

Contents

List of Tables

List of Figures

Abstract

Although many may disagree or we may be taught the contrary, life is inherently non-independent. While in many situations it is *safe* and *convenient* to assume that our unit of analysis–whether it is genes or people–can be thought of as "independent draws from a population", in many cases such an assumption cannot be made. This dissertation is about this issue–how can we deal with inherently complex and interconnected data–and furthermore, using modern computational tools, how to take advantage of this feature to obtain a better understanding of our world. In this document, I present two problems that, while very different, both exist in the realm of complex interconnected data: phylogenetics and social networks.

Understanding the individual role that genes play in life is a key issue in biomedical-sciences. While information regarding gene function is continuously growing, the number of genes with unknown biological purpose is far greater. Because of this, scientists have dedicated much of their time to building and designing tools that automatically infer gene function. In an effort to contribute to this task, I present a further attempt to do such. While very simple, our model of gene-function evolution has some key features that have the potential to make a significant impact in the field: (a) compared to other methods, ours is highly-scalable, which means that it is possible to simultaneously analyze hundreds of so-called gene-families, comprising of thousands of genes, (b) it supports our biological intuition, in the sense that our model's data-driven results coherently agree with existing beliefs regarding how gene-functions evolved, (c) the prediction accuracy of our model is comparable to other more complex alternatives, and (d) perhaps most importantly, our model can be used to both support new annotations and to suggest areas in

which existing annotations show inconsistencies that may indicate errors or controversies in the literature.

In the second part of my thesis, I discuss a novel extension of an extensively studied family of statistical models, Exponential-Family Random Graph Models (ERGMs). My extension leverages "modern" computational power to revamp an under-studied yet very important problem in social network analysis, that of small networks. Statistical models for social networks have enabled researchers to study complex social phenomena that give rise to observed patterns of relationships among social actors and to gain a rich understanding of the interdependent nature of social ties and actors. Much of this research has focused on social networks within medium to large social groups. To date, these advances in statistical models for social networks, and in particular of ERGMS, have rarely been applied to the study of small networks, despite small network data in teams, families, and personal networks being common in many fields. We revisit the estimation of ERGMs for small networks and propose using exhaustive enumeration. As part of this work, I developed "ergmito", an R package that implements the estimation of pooled ERGMs for small networks using Maximum Likelihood Estimation (MLE). Based on the results of an extensive simulation study, I conclude that there are several benefits of direct MLE estimation compared to approximate methods and that this creates opportunities for valuable methodological innovations that can be applied to modeling social networks with ERGMs.

Chapter 1

Introduction

> "To return to the difficulty which has been stated with respect both to definitions and to numbers, what is the cause of their unity? In the case of all things which have several parts and in which the totality is not, as it were, a mere heap, but the whole is something beside the parts [...]"
>
> – Aristotle, Metaphysics, Part VI

In this document, I present a body of work that I have developed aimed at solving two problems that, while they may live in different scientific fields, are very related to each other as (1) both deal with network-based data structures, and (2) because of that, both lay in the realms of non-independent data.

In the first project, developed in chapter 2, I present a novel gene-functional evolution model that is aimed at augmenting genetic annotations using phylogenetics. Phylogenetic trees (representing the evolutionary relationships between genetic sequences [79]), hold an important amount of information that it is believed can lead to a great advances when trying to understand functional-evolution from the genetic point of view [113]. Intuitively, the state of un-annotated genes can be inferred, or at least approximated, looking at their *evolutionary siblings*, genes with which they form a phylogenetic family. As a gene evolves, biology tells us that the moments during which new species emerge asre likely to be very much related to genetic mutation events,

meaning the moments that genes diverge from their *parents* by losing or gaining one or multiple functions.

This evolutionary model, which is elaborated on at length in chapter 2, is a very powerful tool that can be easily scaled to incorporate multiple functions and families in a way that existing similar models cannot, but it does not completely deal with some problems associated with the sparseness of available data. Therefore, chapter 4 details two possible extensions that may be pursued to help deal with this problem.

While a major focus of the science of network/social networks has recently been on applications to large systems (*Big Data*), understanding the local dynamics that bind individuals together remains a very important theme. In chapter 5, I present an extension to a well-studied family of statistical models, Exponential-Family Random Graph Models (ERGMs), in which I apply them to a particular class of data, small graphs. In general, the intractability of the probability function of ERGMs has encouraged the entire field to focus on approximate methods for parameter estimation [1, 45, 58, 74, 103, 120]. Here, I revisit this parameter estimation method looking at a case in which exact computation of the ERGM probability function is feasible, i.e., the case of the small networks.

The ability to use the exact probability function opens the door to a large set of statistical methods and tools that, as of today, cannot be applied when only approximate solutions are available. After presenting some results on the statistical properties of Maximum Likelihood Estimators (MLE) for small ERGMs, chapter 6 discusses further research opportunities.

Finally, given that statistical computing is the backbone of a significant part of my research, I dedicate the last chapter to describing some of the software tools that I have developed throughout the years while working on these problems. When it comes to methods development, I believe that it is the responsibility of scientists to provide the community with the appropriate tools needed to implement the proposed methods. This lays the groundwork for the advancement of the field.

Chapter 2

On the Automatic Prediction of Gene Functions

This chapter is the result of a recently submitted paper to the journal *PLOS Computational Biology* in which I have the following co-authors: Duncan D. Thomas, John Morrison, Huaiyu Mi, Paul D. Thomas, and Paul Marjoram. The latest version can be found as a pre-print on BioRxiv https://doi.org/10.1101/2020.05.14.095687.

2.1 Introduction

The overwhelming majority of sequences in public databases remain experimentally uncharacterized, a trend that is increasing rapidly with the ease of modern sequencing technologies. To give a rough idea of the disparity between characterized and uncharacterized sequences, there are $\sim$ 15 million protein sequences in the UniProt database that are candidates for annotation, while, only 81,000 (0.3%) have been annotated with a Gene Ontology (GO) term based on experimental evidence. It is therefore a high priority to develop powerful and reliable computational methods for inferring protein function. Many methods have been developed, and a growing number of these have been assessed in the two Critical Assessment of Function Prediction (CAFA) experiments held to date [65, 91].

In previous work, we developed a semi-automated method for inferring gene function based on creating an explicit model of function evolution through a gene tree [41]. This approach adopts the "phylogenetic" formulation of function prediction first proposed by Eisen [31], and the use of GO terms to describe function as implemented in the SIFTER software (Statistical Inference of Function Through Evolutionary Relationships) developed by Engelhardt et al. [33]. To date, our semi-automated method has been applied to over 5000 distinct gene families, resulting in millions of annotations for protein coding genes from 142 different fully sequenced genomes. However, this approach requires manual review of GO annotations, and manual construction of distinct models of gene function evolution for each of the 5000 families. Even using extensive curation and complex software, called PAINT, the semi-automated inference process has taken multiple person-years. Further, the semi-automated process cannot keep up with the revisions that are constantly necessary due to continued growth in experimentally supported GO annotations.

Here, we describe an attempt to develop a fully automated, probabilistic model of function evolution, that leverages the manually constructed evolutionary models from PAINT, for both training and assessing our new method. Related work has previously been undertaken in this area, and a probabilistic framework for function prediction has been implemented in SIFTER. However, this framework assumes a model of function evolution that limits its applicability in function prediction. First, it was developed specifically only to treat molecular function, and not cellular component and biological process terms that could, in principle, be predicted in a phylogenetic framework. Cellular component and biological process GO terms are the most commonly used in most applications of GO [112], so these are of particular practical importance. And second, while it has proven to provide good predictions (about 73% accuracy using the area under the curve statistic), it cannot be scaled and the model itself provides no theoretical insights whatsoever. In practice, perhaps the most serious problem for any inference method is the sparseness of experimental annotations compared to the size of the tree. As a result, standard parameter estimation techniques for the SIFTER framework are impossible, and consequently for SIFTER2.0, the transition matrix parameters are fixed at somewhat arbitrary values [32].

In order to overcome these problems, we propose a much simpler evolutionary model, in which (like the semi-automated PAINT approach) each function is treated as an independent character that can take the value 1 (present) or 0 (absent) at any given node in the phylogenetic tree. Using information about experimental annotations available in the GO database, and phylogenetic trees from the PANTHER project [78], we show that we can build a parsimonious and highly scalable model of functional evolution that provides both intuitive insights on the evolution of functions and highly accurate predictions.

The content of the paper is as follows: section 2.2 presents mathematical notation and formally introduces the model, including parameter estimation and calculation of posterior probabilities. Section 2.3 presents a large-scale application in which we take a sample of annotations from the GO database along with their corresponding phylogenies, fit our model, and analyze the results. Finally, section 2.4 discusses method limitations and future research directions for this project.

2.2 Methods

In general terms, we propose a probabilistic model that reflects the process by which gene functions are propagated through an evolutionary tree. The fundamental idea is that for any given node in the tree, we can write down the probability of observing a function to be present for the gene as a function of model parameters and the functional state of its parent node, essentially modeling probability of gaining and losing function.

The baseline model has 5 parameters: the probability that the most recent common ancestor of the entire tree, i.e. the root node, had the function of interest, the probability of functional gain (the offspring node gains the function when the parent node did not have it), the probability of functional loss (the offspring node loses the function when the parent node had it), and two additional parameters capturing the probability that the gene was incorrectly labeled, i.e. the probability of classifying an absent function as present and vice-versa. We also consider two simple extensions: specifying the functional gain and loss by type of evolutionary event (speciation

or duplication), and pooling information across trees.

As explained later, in this version of our model, if there are multiple gene functions of interest, we analyze one function at a time (i.e., we treat those functions as independent). But later we also show results for a joint analysis of multiple functions, assuming they share parameter values. We then discuss further extensions to our model in the Discussion section.

We assume that our starting data consists of a set of gene annotations for a given tree. We further assume that those annotations occur only at the leaves of the tree (as is typical) and that those leaves were annotated via manual curation derived from experimental data (and are therefore subject to the misclassification probabilities outlined above). Our goal is then to predict the functional state of un-annotated leaves (and, conceivably, to assess the likely quality of the existing annotations). Our perspective is Bayesian. Therefore, we proceed by estimating the posterior distributions of the parameters of the evolutionary model and then, conditional on those estimated distributions, computing the posterior probability of function for each node on the tree.

We now give full details, beginning with some notation.

2.2.1 Definitions - Annotated Phylogenetic Tree

A phylogenetic tree $\Lambda \equiv (\mathcal{N}, \mathcal{E})$ is a tuple of nodes $\mathcal{N}$, and edges $\mathcal{E} \equiv \{(n, m) \in \mathcal{N} \times \mathcal{N} : n \mapsto m\}$ defined by the binary operator $\mapsto$ *parent of*. We define $\mathbf{O}(n) \equiv \{m \in \mathcal{N} : (n, m) \in \mathcal{E}, n \in \mathcal{N}\}$ as the set of offspring of node n, and $\mathbf{p}(m) \equiv \{n \in \mathcal{N} : (n, m) \in \mathcal{E}, m \in \mathcal{N}\}$ as the parent node of node m. Given a tree Λ, the set of leaf nodes is defined as $L(\Lambda) \equiv \{m \in N : \mathbf{O}(m) = \{\emptyset\}\}$.

Let $\mathbf{X} \equiv \{x_n\}_{n \in \mathcal{N}}$ be a vector of annotations in which the element x_n denotes the state of the function at node n, taking the value 1 if such function is present, and 0 otherwise. We define an Annotated Phylogenetic Tree as the tuple $D \equiv (\Lambda, \mathbf{X})$.

Our goal is to infer the true state of $\mathbf{X}$, while only observing an imperfect approximation of it, $\mathbf{Z} = \{z_l\}_{l \in \mathcal{N}}$, derived from experimentally supported GO annotations [112]. Typically only a

Symbol	Description
$\Lambda \equiv (\mathcal{N}, \mathcal{E})$	Phylogenetic Tree.
$\mathbf{p}(n)$	Parent of node n.
$\mathbf{O}(n)$	Offspring of node n.
$\mathbf{X} \equiv \{x_n\}_{n \in \mathcal{N}}$	True annotations.
$\mathbf{Z} \equiv \{z_n\}_{n \in \mathcal{N}}$	Experimental annotations.
$D \equiv (\Lambda, \mathbf{X})$	Annotated Phylogenetic Tree.
$\tilde{D} \equiv (\Lambda, \mathbf{Z})$	Experimentally Annotated Phylogenetic Tree.
$\tilde{D}_n$	Induced Experimentally Annotated Subtree of node n.
$\tilde{D}_n^c$	Complement of $\tilde{D}_n$.

Table 2.1: Mathematical Notation

small subset of the leaf nodes will have been annotated. We refer to the tuple $\tilde{D} \equiv (\Lambda, \mathbf{Z})$ as an Experimentally Annotated Phylogenetic Tree. Finally, let $\tilde{D}_n \subset \tilde{D}$ be the induced experimentally annotated subtree that includes all information – i.e., tree structure and observed annotations – regarding the descendants of node n (including node n itself). This object constitutes a key element of the likelihood calculation. Table 2.1 summarizes the mathematical notation.

2.2.2 Likelihood of an Annotated Phylogenetic Tree

Baseline model

Our evolutionary model for gene function is characterized by the phylogeny and five model parameters: the probability that the root node has the function, π, the probability of an offspring node either gaining or losing a function, μ_{01} and μ_{10}, and the probability that either an absent function is misclassified as present or that a present function is misclassified as absent, ψ_{01} and ψ_{10} respectively, in the input data.

To simplify notation, we will write $\psi = (\psi_{01}, \psi_{10})$ and $\mu = (\mu_{01}, \mu_{10})$ when referring to those pairs. Table 2.2 summarizes the model parameters.

Following [37], we use a pruning algorithm to compute the likelihood of an annotated phylogenetic tree. In doing so, we visit each node in the tree following a post-order traversal search, i.e., we start at the leaf nodes and follow their ancestors up the tree until we reach the root node.

Parameter	Description
π	Probability that the root node has the function
μ_{01}	Probability of gaining a function
μ_{10}	Probability of losing a function
ψ_{01}	Probability of experimental mislabeling of a 0
ψ_{10}	Probability of experimental mislabeling of a 1

Table 2.2: Model parameters

Given the true state of leaf node l, the mislabeling probabilities are defined as follows:

$$\mathbb{P}\left(z_l = 1 \mid x_l = 0\right) = \psi_{01}, \quad \mathbb{P}\left(z_l = 0 \mid x_l = 1\right) = \psi_{10}$$

We can now calculate the probability of leaf l having state z_l, given that its true state is x_l as:

$$\mathbb{P}\left(z_l \mid x_l, \psi\right) = \begin{cases} \psi_{01} z_l + \left(1 - \psi_{01}\right)\left(1 - z_l\right), & \text{if } x_l = 0 \\ \psi_{10}\left(1 - z_l\right) + \left(1 - \psi_{10}\right) z_l, & \text{if } x_l = 1 \end{cases} \tag{2.1}$$

Similarly, the functional gain and functional loss probabilities are defined as follows:

$$\mathbb{P}\left(x_m = 1 \mid x_{\mathbf{p}(m)} = 0\right) = \mu_{01}, \quad \mathbb{P}\left(x_m = 0 \mid x_{\mathbf{p}(m)} = 1\right) = \mu_{10}$$

Note that in this version of our model we assume that these probabilities are independent of the time that has passed along the branch connecting the two nodes. We return to this point in the Discussion. Now, for any internal node $n \in \mathcal{N}$, we can calculate the probability of it having state x_n, given the state of its parent $\mathbf{p}\left(n\right)$ and the vector of parameters μ, as:

$$\mathbb{P}\left(x_m \mid x_{\mathbf{p}(m)}, \mu\right) = \begin{cases} \mu_{01} x_m + \left(1 - \mu_{01}\right)\left(1 - x_m\right) & \text{if } x_{\mathbf{p}(m)} = 0 \\ \mu_{10}\left(1 - x_m\right) + \left(1 - \mu_{10}\right) x_m & \text{if } x_{\mathbf{p}(m)} = 1 \end{cases} \tag{2.2}$$

Together with (2.1), and following [37], this allows us to calculate the probability of the interior node n having state x_n conditional on $\tilde{D}_n$, its induced subtree, as the product of the conditional probabilities of the induced subtrees of its offspring:

$$\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n, \psi, \mu\right) = \prod_{m \in \mathbf{O}(n)} \mathbb{P}\left(\tilde{D}_m \;\middle|\; x_n\right), \tag{2.3}$$

where

$$\mathbb{P}\left(\tilde{D}_m \;\middle|\; x_n\right) = \begin{cases} \sum_{x_m \in \{0,1\}} \mathbb{P}\left(\tilde{D}_m \;\middle|\; x_m, \psi, \mu\right) \mathbb{P}\left(x_m \mid x_n, \mu\right) & \text{if } m \text{ is interior,} \\ \sum_{x_m \in \{0,1\}} \mathbb{P}\left(x_m \mid z_m, \psi\right) \mathbb{P}\left(x_m \mid x_n, \mu\right) & \text{if } m \text{ is leaf.} \end{cases}$$

This is a recursive function that depends upon knowing the offspring state probabilities for internal nodes, which, since we are using a pruning algorithm, will already have been calculated as part of the process. Finally, the probability of the experimentally annotated phylogenetic tree can be computed using the root node conditional state probabilities:

$$\mathbb{P}\left(\tilde{D} \;\middle|\; \psi, \mu, \pi\right) = \sum_{x_0 \in \{0,1\}} \mathbb{P}\left(x_0 \mid \pi\right) \mathbb{P}\left(\tilde{D}_0 \;\middle|\; x_0, \psi, \mu\right) \tag{2.4}$$

where $\mathbb{P}\left(x_0 \mid \pi\right) = \pi x_0 + (1 - \pi)(1 - x_0)$.

In the next section we introduce additional refinements that allow us take into account prior biological knowledge that might constrain the parameter space and alleviate the typical sparseness of the curated data – we return to these issues in the Discussion.

Pooled data models

As mentioned earlier, the sparsity of experimental annotations typically observed in this context makes inference challenging. However, in order to improve inference we might attempt to "borrow strength" across a set of functions, by combining them in a single analysis. As a "proof of principle" of this idea, we will also show results in which we assume that sets of annotated phylogenetic trees share population parameters. This way, we can estimate a joint model by pooling those annotated phylogenetic trees, providing stronger inference about the evolutionary parameters and therefore, we hope, more accurate inference of gene annotations. We note that we have strived to make sure our software implementation extremely computationally efficient, allowing us to

estimate a joint model with hundreds of annotated trees in reasonable time on a single machine.

Our results show that using a pooled-data model for parameter estimation greatly increases the model's predictive power. In the Discussion we will outline possible future, less simplified, approaches to this problem.

Type of evolutionary event

The non-leaf nodes in the phylogenetic trees that we are considering here come in two types, reflecting duplication and speciation events. It has been widely-observed that change of gene function most commonly occurs after duplication event (broadly speaking, the extra copy of a gene is free to change function because the original copy is still present.) Speciation events, on the other hand, are mostly driven by external causes that do not necessarily relate to functional changes in one specific gene, meaning that, while we may observe some differences between siblings, their function generally remains the same.

Our model reflects this biological insight by allowing the functional gain and loss probabilities to differ by type of event. In particular, instead of having a single parameter pair (μ_{01}, μ_{10}), we now have two pairs of functional gain and loss parameters, (μ_{01d}, μ_{10d}) and (μ_{01s}, μ_{10s}), the gain and loss probabilities for duplication and speciation events, respectively.

2.2.3 Estimation and Prediction

We adopt a Bayesian framework, introducing priors for the model parameters. In particular, as described in section 2.3, we use Beta priors for all model parameters. Estimation is then performed using Markov chain Monte Carlo [MCMC] using an Adaptive Metropolis transition kernel [48] with reflecting boundaries at $[0, 1]$ (see, for example, [125]) as implemented in the *fmcmc* R package [115]. We do this using the R package *aphylo* which we have developed to implement this model (`https://github.com/USCbiostats/aphylo`). The R package also allows estimating model parameters using Maximum Likelihood Estimation [MLE] and Maximum A Posteriori estimates [MAP].

Regarding model predictions, once we fit the model parameters we use the calculated probabilities during the computation of the likelihood function (pruning process) and feed them into the posterior probability prediction algorithm, which is exhaustively described in subsection 2.5.1 (see also [62]). It is important to notice that, in general, predictions are made using a leave-one-out approach, meaning that to compute the probability that a given gene has a given function, we remove the annotation for that gene from the data before calculating the overall likelihood of the tree. Otherwise we would be including that gene's own observed annotation when predicting itself. The latter point is relevant for evaluating prediction accuracy. Prediction of unannotated genes, on the other hand, is performed using all the available information.

In general, whenever using MCMC to estimate the model parameters, we used 4 parallel chains and verified convergence using the Gelman-Rubin [10] statistic. In all cases, we sampled every tenth iteration after applying a burn-in of half of the sample. As a rule of thumb, the processes were considered to have reached a stationary state once the potential scale reduction factor statistic was below 1.1. When we report point estimates for parameters we use the mean value across sampled iterations for all chains after burn-in.

To measure prediction quality, we use the Mean Absolute Error [MAE], which we calculate as follows:

$$\text{MAE} = |\mathbf{Z}|^{-1} \sum_{n} |z_n - \hat{x}_n| \tag{2.5}$$

Where $|Z|$ is the number of observed annotations, z_n is the observed annotation (0/1), and $\hat{x}_n$ is the predicted probability that the n-th observation has the function of interest. Since z_n is binary, perfect predictions result in a score of 0, completely wrong predictions result in a score of 1, and a random coin toss results in a score of 0.5.

While other statistics, such as accuracy and Area Under the Receiver Operating Characteristic Curve [AUCR] (see [36] for a general overview), are most commonly used for binary prediction models, we prefer to use MAE since it is: (a) robust to unbalanced data, (b) probabilistic and thus does not depend on choice of threshold for prediction, and (c) robust to probabilistic noise,

see [20, 38] for a more general discussion.[1]

We now turn our attention to an application using experimental annotations from the Gene Ontology Project.

2.3 Results

To evaluate performance of our model, we used data from the Gene Ontology project. In particular, in order to assess the potential utility of combining information across similar genes, we used our model in two different ways: a pooled-data model, in which we treated all trees as if they had the same evolutionary parameter values, and a one-tree-at-a-time model, in which we estimated parameters, and then imputed functional status, for each tree separately.

To reduce sparsity of annotations somewhat, from the entire set of available annotated phylogenies in PANTHER [78] (about 15,000), we selected only those in which there were at least two annotations of each type (i.e., at least two genes annotated as possessing a given function, and two as lacking the function). This yielded a total of 141 experimentally annotated trees. A key feature of the resulting dataset is that most leaves still have no annotation, as experimental evidence is still very sparse. Furthermore, in order to test the robustness of our results, we compared 4 analyses: MCMC with a uniform prior for parameter values, MCMC with a Beta prior (for both of which we report the mean,) Maximum Likelihood Estimation [MLE], and Maximum A Posteriori estimates [MAP]. The shape parameters used for MCMC and MAP with Beta priors, which are detailed in Table 2.3, were chosen to reflect the biological intuition that change of state probabilities should be higher at duplication events.

Finally, accuracy was measured using leave-one-out cross-validated Mean Absolute Error [MAE].

[1] During the development of this paper, we designed an alternative accuracy measurement that we ultimately excluded from the final version of this manuscript. Appendix A shows a detailed description of such measurement that is also based on MAE but has an interesting statistical interpretation that can be used for statistical evaluation of prediction accuracy.

Parameter	α	θ	Mean $\alpha/(\alpha+\theta)$
ψ_{01}, ψ_{10}	2	9	0.18
μ_{01d}, μ_{10d}	9	2	0.81
μ_{01s}, μ_{10s}	2	9	0.18
π	2	9	0.18

Table 2.3: Parameters for the Beta priors used in MCMC and MAP estimation. Shape parameters (α, θ) were set with the prior that functional gain and loss, (μ_{01d}, μ_{10d}), are more likely at duplication events.

2.3.1 Pooled-data Model

First we show results for the pooled-data model, in which we combine the analysis of multiple trees, assuming parameter values do not vary between trees. The resulting parameter estimates are presented in Table 2.4, and are seen to reflect biological intuition: we see high probabilities of change of function at duplication events, which occurs regardless of whether we use an informative prior or not. While both probabilities are high, gain of function is roughly twice as likely as loss of function at such nodes. We note that we estimate relatively high values of ψ_{01} and low values for ψ_{10}. We believe this is largely due to the sparsity of 0, i.e. "*absent*", experimental annotations, which means that high values for ψ_{10} are implausible and implies that any mislabelings must perforce be that of assigning function falsely. Thus we think the relatively high estimates of ψ_{01} are an artefact that will likely disappear in the presence of less sparse annotation data.

We note that parameter estimates are consistent across estimation methods. Here we used both non-informative (uniform) and relatively informative priors. We see that choice of prior had no significant effect on the final set of estimates, meaning that these are driven by data and not by the specified priors. The only exception is for the root node, π, which is hard to estimate since the root node exists far back in evolutionary time [40]. We also conducted a simulation study across a broader range of parameter values, using simulated datasets, in which we saw similar behavior. These results are shown in subsection 2.5.2. However, we note that when analyzing one tree at a time, this robustness is lost [results not shown] because the sparsity of data typical in any single given tree does not permit strong inference regarding evolutionary parameters. Therefore,

the prior becomes more influential.

	(1)	(2)	(3)	(4)
Mislabeling				
ψ_{01}	0.23	0.28	0.21	0.27
ψ_{10}	0.01	0.01	0.00	0.01
Duplication events				
μ_{d01}	0.97	0.96	1.00	0.97
μ_{d10}	0.52	0.62	0.51	0.61
Speciation events				
μ_{s01}	0.05	0.06	0.05	0.05
μ_{s10}	0.01	0.02	0.01	0.02
Root node				
π	0.79	0.46	0.83	0.48
Method	MCMC	MCMC	MLE	MAP
Prior	Uniform	Beta	-	Beta
AUC	0.76	0.75	0.77	0.75
MAE	0.35	0.37	0.34	0.36

Table 2.4: Parameter estimates for model with experimentally annotated trees. Overall, the parameter estimates show a high level of correlation across methods. Regardless of the method, with the exception of the root node probabilities, most parameter estimates remain the same. As expected, mutation rates at duplication nodes, indexed with a d, are higher than those observed in speciation nodes, indexed with an s.

We note that each instance of the estimation process (reaching stationary distribution in MCMC or finding the maxima in MLE and MAP) took between 1 to 3 minutes using a regular laptop computer. To put this number into context, we use the time calculator provided by SIFTER [32, 33], which is available at http://sifter.berkeley.edu/complexity/, as a benchmark. Since, as of today, SIFTER is designed to be used with one tree at a time, we compared our run time with the estimated run time for SIFTER for a tree with 590 leaves (the average size in our data). In such case, they estimate their algorithm would take about 1 minute to complete, with an upper bound of $\sim$9 minutes, which translates into about 141 minutes to run on our entire dataset (upper bound of $\sim$21 hours). However, in SIFTER's defense, we note that their model allows for a greater level of generality than does our current implementation of our own method. Specifically, they allow for correlated changes across multiple gene functions. In the simplest

case, using "truncation level" 1 as defined in their paper, the estimated run time for an analysis comparable in size with the one conducted here, 141 functions and 83,000 leaves, is 6 hours. However, for truncation level 2, the estimated run time is 2.2 years. In the Discussion section we outline how we propose to extend our method in future to allow for for more sophisticated joint analyses across function than the one we present here.

2.3.2 One-tree-at-a-time

The joint analysis of trees we describe in the previous section makes the enormous assumption that parameter values do not vary across trees. This assumption is made for pragmatic reasons: while it is not likely to be correct (and more general approaches will be described in the Discussion), it is a simple analysis to implement and it allows us to quickly assess a baseline for the improvement in overall prediction accuracy that might result from performing inference jointly across trees. So, in this section we report results of a one-at-a-time analysis of gene function and assess whether the pooled model outperforms a one-at-a-time approach with regards to accuracy, even using our extremely restrictive assumptions.

For each phylogeny, we fitted two different models, one with an uninformative uniform prior and another with the same beta priors used for the pooled-data models, this is, beta priors with expected values close to zero or one, depending on the parameter. Again, as before, model predictions were undertaken in a leave-one-out approach and corresponding MAEs were calculated for each set of predictions. Table 2.5 shows the differences in Mean Absolute Error [MAE] between the various approaches.

From Table 2.5 we can see three main points: First, in the case of the one-at-a-time predictions, informative priors have a significantly positive impact on accuracy. Compared to the predictions using beta priors, predictions based on the MAE estimates with uniform priors have a significantly higher MAE with a 95% confidence interval [c.i.] ranging [0.06, 0.09]. Second, predictions based on the pooled-data estimates outperform predictions based on one-at-a-time parameter estimates, regardless of the prior used. And third, in the case of predictions based on

	Pooled-data	One-at-a-time	
	Beta prior	Unif. prior	Beta Prior
Pooled-data			
Unif. prior	[-0.02,-0.01]	[-0.14,-0.10]	[-0.06,-0.03]
Beta prior	-	[-0.12,-0.09]	[-0.04,-0.01]
One-at-a-time			
Unif. prior	-	-	[0.06, 0.09]

Table 2.5: Differences in Mean Absolute Error [MAE]. Each cell shows the 95% confidence interval for the difference in MAE resulting from two methods (row method minus column method). Cells are color coded blue when the method on that row has a significantly smaller MAE than the method on that column; Conversely, cells are colored red when the method in that column outperforms the method in that row. Overall, predictions calculated using the parameter estimates from *pooled-data* predictions outperform *one-at-a-time*.

pooled-data estimates, model predictions based on the parameter estimates using a uniform prior have significantly smaller MAE than predictions based on parameter estimates using a beta prior with a 95% c.i. equal to [-0.02, -0.01].

Given the sparsity of experimental annotation on most trees, a natural question to ask is: How important is it to have good quality parameter estimates in order to achieve good-quality predictions of gene function? In the next section we perform a sensitivity analysis to address this question.

2.3.3 Sensitivity Analysis

A fundamental question to ask is how much the accuracy of the posterior probabilities for gene function depends on the inferred values of the underlying evolutionary parameters used to make the predictions. Figure 2.1 shows how the distribution of the Mean Absolute Error [MAE] changes as a function of using parameters different from those obtained in the fitting process. In particular, each block of boxplots shows a *ceteris paribus* type of analysis, this is, MAE as a function of changing one parameter at a time while fixing the others.

In the case of the mislabeling probabilities (first row), it is not until they reach values close to one that the prediction accuracy significantly deteriorates. While higher values of ψ_{10} have a consistent negative impact on accuracy, the same is not evident for ψ_{01}. The latter could be a

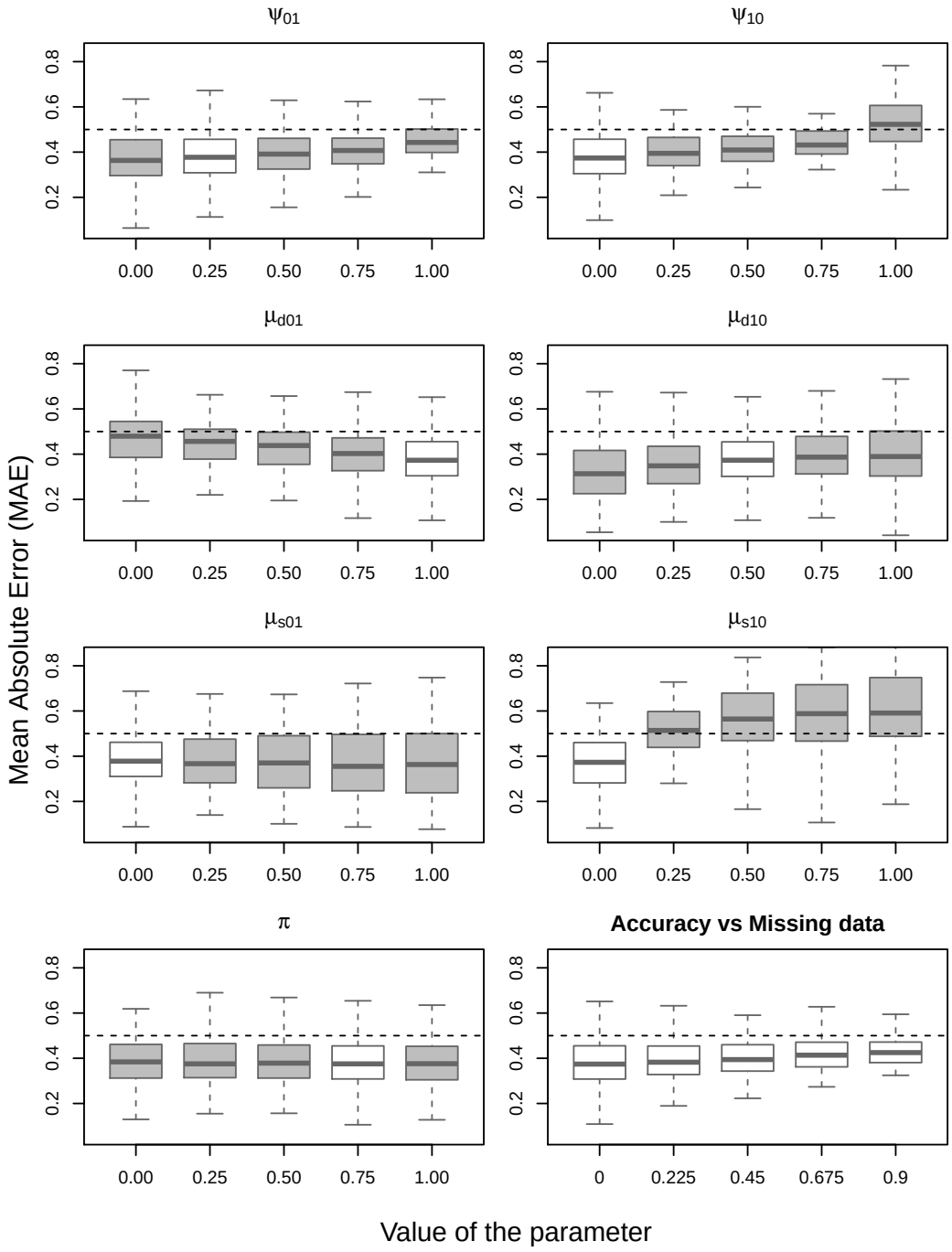


Figure 2.1: Sensitivity analysis. Boxplots of MAEs as a function of single parameter updates. The first seven plots show how the MAEs change as a function of fixing the given parameter value raging from 0 to 1. In each of these plots, the white box indicates the parameter value used to generate the data (i.e., the "correct" parameter value). The last boxplot shows the distribution of the MAEs as a function of the amount of available annotation data, that is, how accuracy changes as the degree of missing annotations increases.

consequence of the low prevalence of 0 ("no function") annotations in the data, as illustrated in Figure 2.2.

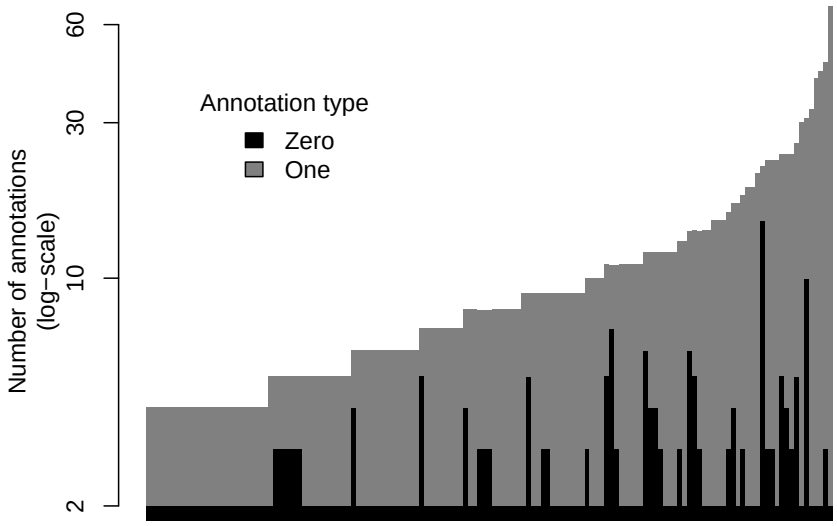


Figure 2.2: Number of annotated leaves for all 141 trees by type of annotation. Most of the annotations available in GO are positive assertions of function, that is, 1s. Furthermore, as observed in the figure, all of the trees used in this study have two "Not" (i.e., 0) annotations, which is a direct consequence of the selection criteria used, as we explicitly set a minimum of two annotations of each type per tree+function.

Functional gain and loss probabilities (second and third rows) have a larger effect on MAEs when misspecified, especially at speciation events, (μ_{s01}, μ_{s10}). This is largely because speciation events are much more common than duplication events across trees. Figure 2.3 shows a detailed picture of the number of internal nodes by type of event. Of the seven parameters in the model, the root node probability π has a negligible impact on MAE.

Finally, we evaluated the impact of removing annotations on accuracy. While a significant drop may not be obvious from the last box in Figure 2.1, a paired t-test found a significant drop in the accuracy levels.

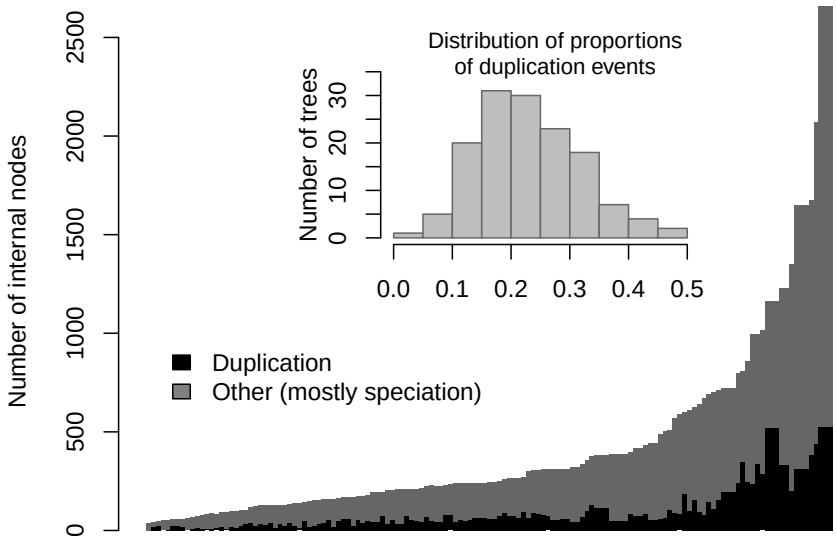


Figure 2.3: Number of internal nodes by type of event. Each bar represents one of the 141 trees used in the paper by type of event (mostly speciation). The embedded histogram shows the distribution of the prevalance of duplication events per tree. The minority of the events, about 20%, are duplications.

2.3.4 Featured Examples

An important determinant of the quality of the predictions is how the annotations are co-located. For the model to be able to accurately predict a 1, it is necessary either that it follows immediately after a duplication event in a neighborhood of zeros (so that function can gained with reasonable probability), or that a group of relatives within a clade have the function (i.e., its siblings have the function). Figures 2.4 and 2.5 show examples of high and low accuracy predictions respectively, illustrating how co-location impacts accuracy.

In the previous analyses, we reported predictions based on point estimates. It is more informative to leverage the fact that we have posterior distributions for parameter values. Hence, in this section, in which we are looking in detail at specific trees, we exploit the full posterior distribution, and generate credible intervals by repeatedly sampling parameter values from their posterior distribution and then imputing function using each set of sampled parameter values. As before, we used a leave-one-out approach to make predictions, meaning that to predict any given leaf, we removed any annotation for that leaf and recalculated the probabilities described

in subsection 2.5.1.

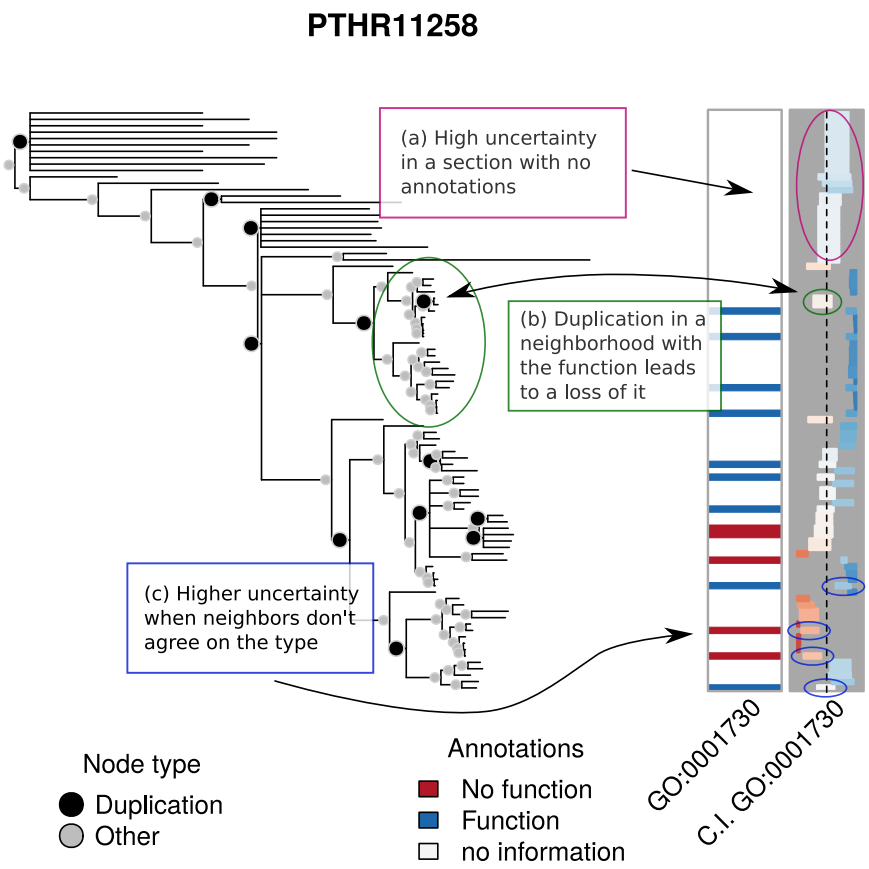


Figure 2.4: *High accuracy predictions* The first set of annotations, first column, shows the experimental annotations of the term GO:0001730 for PTHR11258. The second column shows the 95% C.I. of the predicted annotations. The column ranges from 0 (left end) to 1 (right-end). Bars closer to the left are colored red to indicate that lack of function is suggested, while bars closer to the right are colored blue to indicate function is suggested. Depth of color corresponds to strength of inference. The AUC for this analysis is 0.91 and the Mean Absolute Error is 0.34.

In the high accuracy example, Figure 2.4, which predicts the go term GO:0001730 on PANTHER family PTHR11258, we highlight three features of the figure. First, in box (a), the confidence intervals for the posterior probabilities in sections of the tree that have no annotations tend to be wider, which is a consequence of the degree to which posterior probabilities depend upon having nearby annotated nodes. Second, duplication events serve as turning points for function, as they can either trigger a functional loss, as highlighted in box (b), or a functional

gain. The available data within the clade sets the overall trend for whether a function is present or not. The clade in box (b) is particularly interesting since there is only one duplication event and all of the annotated leaves have the function, which triggers a potential loss for the descendants of the duplication event. Finally, in regions in which labels are heterogeneous, it is harder to make a decisive prediction, as should be the case. This is why, in the leave-one-out predictions, the highlighted predictions in box (c) show wider confidence intervals and posterior probabilities closer to 0.5 than to the true state.

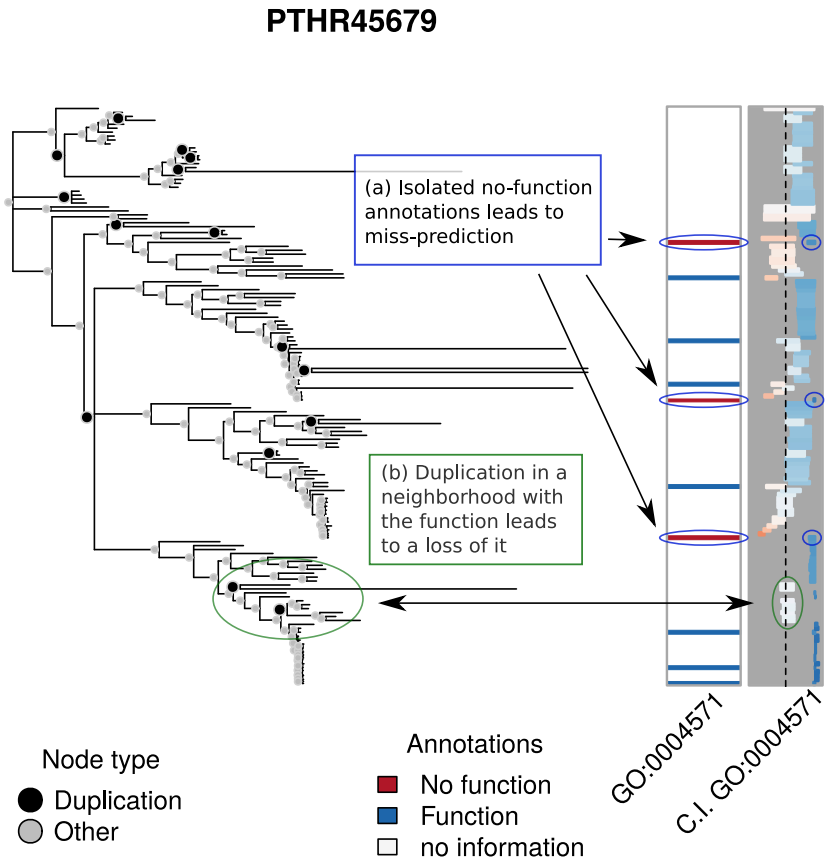


Figure 2.5: *Low accuracy predictions*. The first set of annotations, first column, shows the experimental annotations of the term GO:0004571 for PTHR45679. The second column shows the 95% C.I. of the predicted annotations using leave-one-out. Bars closer to the left are colored red to indicate that lack of function is suggested, while bars closer to the right are colored blue to indicate function is suggested. Depth of color corresponds to strength of inference. The AUC for this analysis is 0.33 and the Mean Absolute Error is 0.52.

Figure 2.5 illustrates one way in which low accuracy can result. In particular, as highlighted by box (a), while the number of annotations is relatively large, (more than 10 over the entire tree, see Figure 2.2), the 'absent' (zero) annotations are distributed sparsely across the tree, having no immediate neighbor with an annotation of the same type. In fact, in every case, their closest annotated neighbors have the function. As a result, all zero-type annotations are wrongly labelled, with the model assigning a high probability of having the function. Similar to what is seen in the high accuracy case, duplication events within a clade with genes of mostly one type have a large impact on the posterior probabilities, as indicated by box (b).

In an effort to understand why the predictions were so poor in this case, we reviewed the family in detail. Because all GO annotations include references to the scientific publications upon which they are based, we were able to review the literature for genes in this family, the ER degradation enhancing mannosidase (EDEM) family. It turns out that the inconsistency of the annotations in the tree reflect a bona fide scientific controversy [85], with some publications reporting evidence that these proteins function as enzymes that cleave mannose linkages in glycoproteins, while others report that they bind but do not cleave these linkages. This explains the divergent experimental annotations among members of this family, with some genes being annotated as having mannosidase activity (e.g. MNL1 in budding yeast), others as lacking this activity (e.g. mnl1 in fission yeast), and some with conflicting annotations (e.g. human EDEM1). The inability of our model to correctly predict the experimental annotations actually has a practical utility in this case, by identifying an area of inconsistency in the GO knowledgebase.

2.3.5 Discoveries

We now explore to what extent predictions from our model can be used to suggest function. Using the estimates from the pooled-data model with uniform priors, we calculated posterior probabilities for all the genes involved in the 141 GO trees+functions analyzed. Moreover, we calculated 95% credible intervals for each prediction using the posterior distribution of the parameters and from there obtained a set of potential new annotations.

While all posterior distributions are predictions, in order to focus on leaves for which state was predicted with a greater degree of certainty, we now consider the subset of predicted annotations whose 95% credible interval was either entirely below 0.1 or entirely above above 0.9, i.e., low or high chance of having the function respectively. We then compared the list of annotations to those available from the QuickGO API, regardless of the evidence code, which resulted in a list of 295 novel predictions.

Ten of the predictions were for genes in the mouse, a well-studied organism [56]. We searched the literature for evidence of the predicted functions, and uncovered evidence for six of the ten predictions: estradiol dehydrogenase activity for Hsd17b1 [49], involvement in response to iron ion for Slc11a2 [101], involvement of Bok in promoting apoptosis [16], DNA binding activity of Nfib [57], extracellular location for Lipg [5], the adenylate kinase activity of Ak2 [67]. The full list, including the programs used to generate it, is available in this website: https://github.com/USCbiostats/aphylo-simulations.

2.4 Discussion

In this paper, we presented a model for the evolution of gene function that allows rapid inference of that function, along with the associated evolutionary parameters. Such a scheme allows for the hope of *automated* updating of gene function annotations as more experimental information is gathered. We note that our approach results in probabilistic inference of function, thereby capturing uncertainty in a concise and natural way. In simulation studies, using phylogenies from PANTHER, we demonstrate that our approach performs well. Furthermore, our computational implementation of this approach allows for rapid inference across thousands of trees in a short time period.

We believe that basing inference upon an evolutionary model allows us to capture biological properties of gene evolution, and that doing so results in more accurate inference. However, being based on a model also means being "wrong" [9]. We now discuss the ways in which we are

wrong, and how we believe that despite this "wrongness" our approach is still useful.

Branch length. We have treated the probability of a change of function from node-to-node as unrelated to the length of the branch that connects those nodes. This choice follows the prevailing model of gene functional change occurring relatively rapidly after gene duplication, rather than gradually over time. Extension to models in which this is not true is conceptually straightforward, involving a move from discrete to continuous underlying Markov Chains. Parameters μ_{01} and μ_{10} are then treated as rates rather than probabilities: $\mathbb{P}\left(x_m = 1 \mid x_{\mathbf{p}(m)} = 0, \tau\right)$, $\mathbb{P}\left(x_m = 0 \mid x_{\mathbf{p}(m)} = 1, \tau\right)$, where τ is the length of the branch connecting that pair of nodes.

Multiple gene functions or families. When analyzing multiple gene functions, we have either treated them as if they all had the same parameters values, or were all completely independent. These cases represent two ends of a spectrum, and neither end is likely to be the truth. Even absent any specific knowledge regarding how genes might be related to each other, the sparseness of experimental annotation, and the low frequency of *"no function"* annotations, makes it desirable to take advantage of multiple annotations across functions in order to obtain better parameter inference (see Table 2.6). If there are p gene functions in a particular gene family of interest, a fully saturated model would require $7 \times 2p$ parameters (4 sets of mutation rates, 2 sets of misclassification probabilities, and 1 set of root node probabilities). This, clearly, rapidly becomes very challenging, even for "not very large at all" p. Thus, more sophisticated approaches will be needed.

For now, we believe that the pooled analysis here demonstrates that there is likely to be some utility in developing approaches that can effectively impute annotations for multiple genes or gene functions jointly, particularly when, as is generally the case at present, annotation data is sparse. In future work we intend to explore hierarchical approaches to this problem. For example, it seems reasonable to assume that even if different gene families have different parameter values, the basic evolutionary biology would be similar across all gene families and this could be reflected by modeling the various parameters as random effects in a hierarchical framework. Thus, one would fit the entire ensemble of genes, or gene functions, simultaneously, estimating parameters

for their overall means and variances across families. Such an approach would also allow for formal testing for parameter value differences between different families. While such an approach will be challenging to implement, we intend to pursue it in future work.

We also intend to explore approaches in which we use a simple loglinear model to capture the key features. For example, in such an approach we might include p parameters for the marginal probabilities and $\binom{p}{2}$ parameters for pairwise associations for the mutation rates and baseline probabilities, for example, while treating the misclassification probabilities as independent. There are many possible extensions of this loglinear framework. For example, it seems likely at after a duplication event, as a function is lost in one branch a new function will be gained in that branch while the other branch remains intact. Such possibilities can be readily incorporated within this framework by modeling gain/loss probabilities jointly.

N Ann. per Tree	N	Cum. count	Cum. prop.
1	1173	1173	0.11
2	753	1926	0.17
3	675	2601	0.23
4	573	3174	0.29
5	567	3741	0.34
6	485	4226	0.38
7	504	4730	0.43
8	387	5117	0.46
9	323	5440	0.49
10 or more	5666	11106	1.00

Table 2.6: Distribution of trees per number of functions (annotations) in PANTHERDB. At least 50% of the trees used in this paper have 10 or more annotations per tree.

Parameters suggest biological interpretations. Our model, while simple, has several advantages and appears to perform well overall. We are able to determine parameters not only for one family at a time, but shared parameters over the set of all 141 protein families that have both positive and negative examples of a given function. The parameters have straightforward interpretations. In agreement with the prevailing model of the importance of gene duplication, the probability of function change (either gain or loss) derived from our model is much larger following gene duplication than following speciation. The high probability of an arbitrary function being present

at the root of the tree (π) is consistent with the observation that functions are often highly conserved across long evolutionary time spans [66]. Our model also contains some features that may offer additional biological insights. Our sensitivity analysis shows that a small probability of functional loss following speciation is particularly important to prediction accuracy. In other words, functions, once gained, strongly tend to be inherited in the absence of gene duplication. This includes not only molecular functions as generally accepted, but cellular components (the places where proteins are active), and biological processes (larger processes that proteins contribute to) as well.

Utility of predictions. The predictions from our method may have utility, both in guiding experimental work, or in highlighting areas of conflicting scientific results. High probability predictions are likely to be correct: for the ten such predictions we made for mouse genes, we found experimental evidence for six of them (which were not yet in the GO knowledgebase), and for the remaining four, we found no evidence of either presence or absence of that function. In our leave-one-out predictions of experimentally characterized functions of the EDEM family, we found cases where our predictions were particularly poor, but upon close examination turned out to be indicative of actual conflicts in experimental results from different studies. Deeper analyses of discrepant predictions could be helpful in identifying similar cases.

Improving the input data using taxon constraints. Another type information that can be leveraged is taxon constraints. Taxon constraints – which we define as a set of biological assumptions that restrict the set of possible values for given nodes on the tree (either by assuming gene function will be present or absent there) – can be used to inform our analysis. Within the context of our model, it is simple to specifying that a clade can or cannot have a particular function, (essentially treating it as if it were fully annotated,without error). Thus, inclusion of such constraints would reduce the uncertainty in the model by effectively decreasing the overall depth of the unannotated parts of the tree (distance from the most recent common ancestor to the tip). More importantly, it would also act to increase the number of available annotations, most likely adding those of type *absent*, which as we show in section 2.3, are the scarcest ones.

Use in epidemiologic analyses. We emphasize that the goal of this method is not simply to assign presence or absence of various gene functions to presently unannotated human genes, but to estimate the probabilities π_{gp} that each gene g has each function p. In analyzing epidemiologic studies of gene-disease associations, we anticipate using this annotation information as "prior covariates" in a hierarchical model, in which the first level would be a conventional multiple logistic regression for the log relative risk coefficients θ_g for each gene g and the second level would be a regression of these θ_g on the vector $\pi_g = (\pi_{pg})$ of estimated function probabilities. This second level model could take various forms, depending upon whether one thought the functions were informative about both the magnitude and sign of the relative risks or only their magnitudes. In former case, one might adopt a linear regression of the form $\theta_g \sim N(\alpha_0 + \pi_g'\alpha, \sigma^2)$. In the latter case, one might instead model their dispersion as $\theta_g \sim N(0, \lambda_g)$ or $\theta_g \sim$ Laplace(λ_g) where $\lambda_g = \exp\left\{\alpha_0 + \pi_g'\alpha\right\}$, corresponding to an individualized form of ridge or Lasso penalized regression respectively.

In summary, we have presented a parsimonious evolutionary model of gene function. While we intend to further develop this model to reflect additional biological features, we note that in its current form it has the following key features: (a) It is computationally scalable, making it trivial to jointly analyze hundreds of annotated trees in finite time. (b) It yields data-driven results that are aligned with our biological intuition, in particular, supporting the idea that functional changes are most likely to occur during gene-duplication events. (c) Notwithstanding its simplicity, it provides probabilistic predictions with an accuracy level comparable to that of other, more complex, phylo-based models. (d) Perhaps most importantly, it can be used to both support new annotations and to suggest areas in which existing annotations show inconsistencies that may indicate errors or controversies in those experimental annotations.

2.5 Material and Methods

2.5.1 Prediction of Annotations

Let $\tilde{D}_n^c$ denote an annotated tree with all tree structure and annotations below node n removed, the complement of the induced sub-tree of n, $\tilde{D}_n$. Ultimately we are interested on the conditional probability of the nth node having the function of interest given $\tilde{D}$, the observed tree and annotations. Let $s \in \{0, 1\}$, then we need to compute:

$$\begin{aligned}
\mathbb{P}\left(x_n = s \,\middle|\, \tilde{D}\right) &= \frac{\mathbb{P}\left(x_n = s, \tilde{D}\right)}{\mathbb{P}\left(\tilde{D}\right)} \\
&= \frac{\mathbb{P}\left(x_n = s, \tilde{D}\right)}{\mathbb{P}\left(\tilde{D} \,\middle|\, x_n = 1\right)\mathbb{P}\left(x_n = 1\right) + \mathbb{P}\left(\tilde{D} \,\middle|\, x_n = 0\right)\mathbb{P}\left(x_n = 0\right)} \\
&= \frac{\mathbb{P}\left(\tilde{D}, x_n = s\right)}{\mathbb{P}\left(\tilde{D}, x_n = 1\right) + \mathbb{P}\left(\tilde{D}, x_n = 0\right)}
\end{aligned} \tag{2.6}$$

Using conditional independence (which follows from the Markovian property), the joint probability of $\left(\tilde{D}, x_n\right)$ can be decomposed into two pieces, the "pruning probability", which has already been calculated using the peeling algorithm described in section 2.2.2, and the joint probability of $\left(\tilde{D}_n^c, x_n\right)$:

$$\begin{aligned}
\mathbb{P}\left(\tilde{D}, x_n = s\right) &= \mathbb{P}\left(\tilde{D} \,\middle|\, x_n = s\right)\mathbb{P}\left(x_n = s\right) \\
&\qquad \text{(by conditional independence)} \\
&= \mathbb{P}\left(\tilde{D}_n \,\middle|\, x_n = s\right)\mathbb{P}\left(\tilde{D}_n^c \,\middle|\, x_n = s\right)\mathbb{P}\left(x_n = s\right) \\
&= \mathbb{P}\left(\tilde{D}_n \,\middle|\, x_n = s\right)\mathbb{P}\left(\tilde{D}_n^c, x_n = s\right).
\end{aligned} \tag{2.7}$$

Using the law of total probability, the second term of (2.7) can be expressed in terms of n's

parent state, $x_{\mathbf{p}(n)}$, as:

$$\mathbb{P}\left(\tilde{D}_n^c, x_n = s\right) = \mathbb{P}\left(\tilde{D}_n^c, x_n = s \;\middle|\; x_{\mathbf{p}(n)} = 1\right)\mathbb{P}\left(x_{\mathbf{p}(n)} = 1\right) + \mathbb{P}\left(\tilde{D}_n^c, x_n = s \;\middle|\; x_{\mathbf{p}(n)} = 0\right)\mathbb{P}\left(x_{\mathbf{p}(n)} = 0\right) \tag{2.8}$$

Again, given the state of the parent node, $x_{\mathbf{p}(n)}$, x_n and $\tilde{D}_n^c$ are conditionally independent, and with $s' \in \{0, 1\}$, we have

$$\begin{aligned}\mathbb{P}\left(\tilde{D}_n^c, x_n = s \;\middle|\; x_{\mathbf{p}(n)} = s'\right) &= \mathbb{P}\left(\tilde{D}_n^c \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_n = s \;\middle|\; x_{\mathbf{p}(n)} = s'\right) \\ &= \frac{\mathbb{P}\left(\tilde{D}_n^c, x_{\mathbf{p}(n)} = s'\right)}{\mathbb{P}\left(x_{\mathbf{p}(n)} = s'\right)}\mathbb{P}\left(x_n = s \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\end{aligned}$$

A couple of observations from the previous equation. First, while $\tilde{D}_n^c$ includes node $\mathbf{p}(n)$, it does not include information about its state, $x_{\mathbf{p}(n)}$, since only leaf annotations are observed. Second, the equation now includes $\mathbb{P}\left(x_n \;\middle|\; x_{\mathbf{p}(n)}\right)$, this is, the model's transition probabilities, (μ_{01}, μ_{10}). With the above equation we can write (2.8) as:

$$\begin{aligned}\mathbb{P}\left(\tilde{D}_n^c, x_n = s\right) = \\ &\mathbb{P}\left(\tilde{D}_n^c, x_{\mathbf{p}(n)} = 1\right)\mathbb{P}\left(x_n = s \;\middle|\; x_{\mathbf{p}(n)} = 1\right) + \\ &\mathbb{P}\left(\tilde{D}_n^c, x_{\mathbf{p}(n)} = 0\right)\mathbb{P}\left(x_n = s \;\middle|\; x_{\mathbf{p}(n)} = 0\right) \quad (2.8')\end{aligned}$$

In (2.8'), the only missing piece is $\mathbb{P}\left(\tilde{D}_n^c, x_{\mathbf{p}(n)}\right)$, which can be expressed as

$$\mathbb{P}\left(\tilde{D}_n^c \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_{\mathbf{p}(n)} = s'\right)$$

Furthermore, another application of the Markovian property allows us to decompose $\mathbb{P}\left(\tilde{D} \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$

as a product of conditional probabilities. Thus, $\mathbb{P}\left(\tilde{D} \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$ can be expressed as

$$= \mathbb{P}\left(\tilde{D}_{\mathbf{p}(n)} \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(\tilde{D}^c_{\mathbf{p}(n)} \;\middle|\; x_{\mathbf{p}(n)} = s'\right).$$

(which, by definition, is)

$$= \left\{\prod_{o\in\mathbf{O}(\mathbf{p}(n))} \mathbb{P}\left(\tilde{D}_o \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\right\}\mathbb{P}\left(\tilde{D}^c_{\mathbf{p}(n)} \;\middle|\; x_{\mathbf{p}(n)} = s'\right).$$

(and, taking the pruning probability of node n out of the product operator, gives)

$$= \left\{\prod_{o\in\mathbf{O}(\mathbf{p}(n))\setminus\{n\}} \mathbb{P}\left(\tilde{D}_o \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\right\}\mathbb{P}\left(\tilde{D}^c_{\mathbf{p}(n)} \;\middle|\; x_{\mathbf{p}(n)} = s'\right) \times \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$$
$$= \mathbb{P}\left(\tilde{D}^c_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$$

Where the last equality holds by definition of $\tilde{D}^c_n$. In essence, this shows that we can decompose the conditional probability $\tilde{D}|x_{\mathbf{p}(n)} = s'$ by splitting the tree at either n or $\mathbf{p}\,(n)$. This allows us to calculate $\mathbb{P}\left(\tilde{D}^c_n, x_{\mathbf{p}(n)}\right)$:

$$\mathbb{P}\left(\tilde{D}^c_n, x_{\mathbf{p}(n)} = s'\right) = \mathbb{P}\left(\tilde{D}^c_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_{\mathbf{p}(n)} = s'\right)$$

(multiplying and dividing by $\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$ we get),

$$= \frac{\mathbb{P}\left(\tilde{D} \;\middle|\; x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_{\mathbf{p}(n)} = s'\right)}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)}$$

(Which, using the previous set of equations, is)

$$= \frac{\mathbb{P}\left(\tilde{D}, x_{\mathbf{p}(n)} = s'\right)}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right)} \tag{2.9}$$

The denominator of (2.9) can then be rewritten as follows:

$$\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_{\mathbf{p}(n)} = s'\right) = \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1, x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_n = 1 \;\middle|\; x_{\mathbf{p}(n)} = s'\right) +$$
$$\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0, x_{\mathbf{p}(n)} = s'\right)\mathbb{P}\left(x_n = 0 \;\middle|\; x_{\mathbf{p}(n)} = s'\right)$$

$$\text{(given thatwe know that is } x_n,\ x_{\mathbf{p}(n)} \text{ is no longer informative for } \tilde{D}_n)$$
$$= \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)\mathbb{P}\left(x_n = 1 \mid x_{\mathbf{p}(n)} = s'\right) +$$
$$\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0\right)\mathbb{P}\left(x_n = 0 \mid x_{\mathbf{p}(n)} = s'\right).$$

Now, we can substitute this into the denominator of (2.9) and write:

$$\mathbb{P}\left(\tilde{D}_n^c, x_{\mathbf{p}(n)} = s'\right) = \frac{\mathbb{P}\left(\tilde{D}, x_{\mathbf{p}(n)} = s'\right)}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)\mathbb{P}\left(x_n = 1 \mid x_{\mathbf{p}(n)} = s'\right) + \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0\right)\mathbb{P}\left(x_n = 0 \mid x_{\mathbf{p}(n)} = s'\right)}. \quad (2.9')$$

This way, the probability of observing $\left(\tilde{D}_n^c, x_n = s\right)$, (2.8'), equals:

$$\mathbb{P}\left(\tilde{D}_n^c, x_n = s\right) = \frac{\mathbb{P}\left(\tilde{D}, x_{\mathbf{p}(n)} = 1\right)\mathbb{P}\left(x_n = s \mid x_{\mathbf{p}(n)} = 1\right)}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)(1-\mu_{10}) + \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0\right)\mu_{10}} + \frac{\mathbb{P}\left(\tilde{D}, x_{\mathbf{p}(n)} = 0\right)\mathbb{P}\left(x_n = s \mid x_{\mathbf{p}(n)} = 0\right)}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)\mu_{01} + \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0\right)(1-\mu_{01})} \quad (2.10)$$

Together with the pruning probabilities calculated during the model fitting process, this equation can be computed using a recursive algorithm, in particular the pre-order traversal [84], in which we iterate through the nodes from root to leaves.

When node n is the root node, the posterior probability can be calculated in a straightforward way:

$$\mathbb{P}\left(x_n = 1 \;\middle|\; \tilde{D}\right) = \frac{\mathbb{P}\left(x_n = 1, \tilde{D}\right)}{\mathbb{P}\left(x_n = 1, \tilde{D}\right) + \mathbb{P}\left(x_n = 0, \tilde{D}\right)}$$

$$= \frac{\mathbb{P}\left(\tilde{D} \;\middle|\; x_n = 1\right)\mathbb{P}\left(x_n = 1\right)}{\mathbb{P}\left(\tilde{D} \;\middle|\; x_n = 1\right)\mathbb{P}\left(x_n = 1\right) + \mathbb{P}\left(\tilde{D} \;\middle|\; x_n = 0\right)\mathbb{P}\left(x_n = 0\right)}$$
$$= \frac{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)\pi}{\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 1\right)\pi + \mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n = 0\right)(1-\pi),}$$

since the terms $\mathbb{P}\left(\tilde{D}_n \;\middle|\; x_n\right)$ have already been calculated as part of the pruning algorithm.

2.5.2 Monte Carlo Study

We assess model performance by quantifying the quality of our predictions under several scenarios using annotations constructed by simulating the evolutionary process on *real* trees obtained from PANTHER. In particular, we simulate the following scenarios:

1. **Fully annotated**: For each tree in PANTHER we simulated the evolution of gene function, and the annotation of that function at the tree tips, using the model described in this paper. For each tree we drew a different set of model parameters from the following Beta distribution:

Parameter	α	θ	Mean $\alpha/(\alpha+\theta)$
μ_{01d}	38	2	0.95
μ_{10d}	10	10	0.50
μ_{01s}, μ_{10s}	2	38	0.05
π	2	38	0.05

   We then used the simulated annotations to estimate the model parameters and gene function states. For this case we exclude mislabeling, i.e. all leaves are correctly annotated.

2. **Partially annotated**: Here, we took the set of simulations produced in scenario 1 above, but now estimated the model using a partially annotated tree. Specifically, we randomly dropped a proportion of leaf annotations $m \sim$ Uniform$(0.1, 0.9)$. Once again, we assumed no mislabeling. (So, $\psi_{01} = 0, \psi_{10} = 0$).

3. **Partially annotated with mislabeling** Finally, we take the data from scenario 2 but allow for the possibility of mislabeling in the annotations. Specifically, for each tree we draw values for ψ_{01} and ψ_{10} from a Beta$(2, 38)$ distribution.

In order to assess the effect of the prior distribution, in each scenario we performed estimation twice using two different priors: a well-specified prior, i.e., the one used during the data-generating-process, and a biased prior in which the α shape was twice of that of the data-generating-process.

Accuracy

Figure 2.6 shows the distribution of AUCs and Mean Absolute Errors [MAEs] for the third scenario by prior used and proportion of missing labels.

Overall, the predictions are good with a relatively low MAE/high AUC. Furthermore, as seen in Figure 2.6, both AUC and MAE worsen off as the data becomes more sparse (fewer annotated leaves).

Bias

We now consider bias. Figure 2.7 shows the distribution of the empirical bias, defined as the population parameter minus the parameter estimate, for the first scenario (fully annotated tree). Since the tree is fully annotated and there is no mislabeling, the plot only shows the parameters for functional gain, loss and the root node probability. Of the three parameters, π is the one which the model has the hardest time to recover, what's more, it generally seems to be over-estimated. On the other hand, μ_{01}, μ_{10} estimates do significantly better than those for π, and moreover, in sufficiently large trees the model with the correct prior is able to recover the true parameter value.

Figure 2.8 shows the empirical distribution of the parameter estimates in the third simulation scheme: a partially annotated tree with mislabelling. As the proportion of missing annotations increases, the model tends to, as expected, do worse.

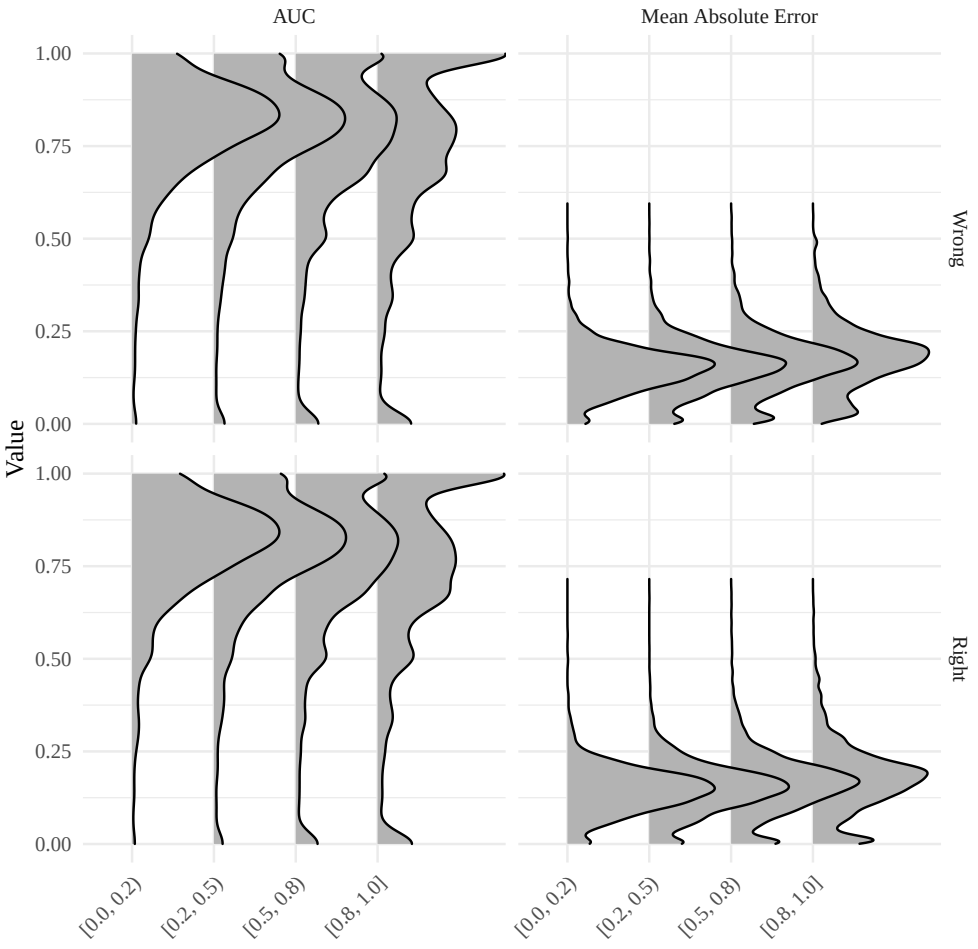


Figure 2.6: Distribution of AUCs and MAEs for the scenario with partially annotated trees and mislabeling. The x-axis shows the proportion of missing annotations, while the y-axis shows the score (AUC or MAE).

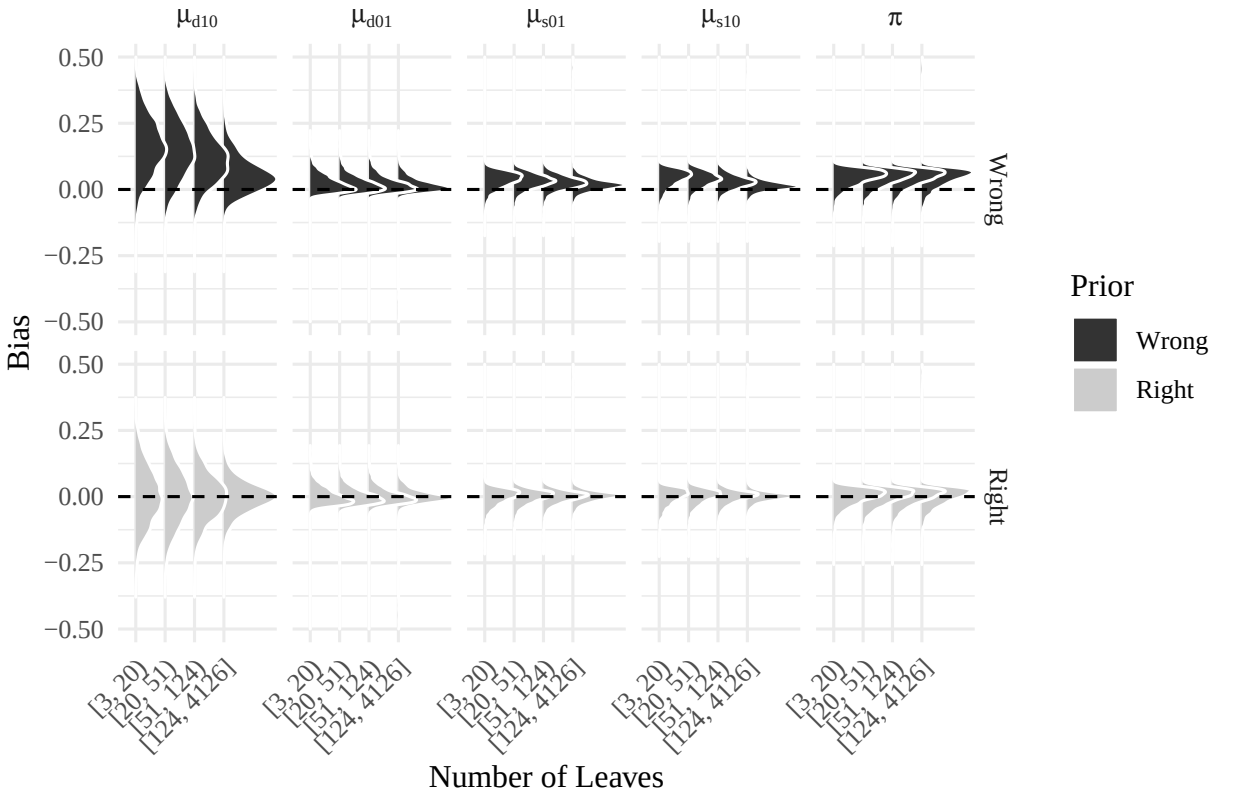


Figure 2.7: Empirical bias distribution for the Fully annotated scenario by type of prior, parameter, and number of leaves.

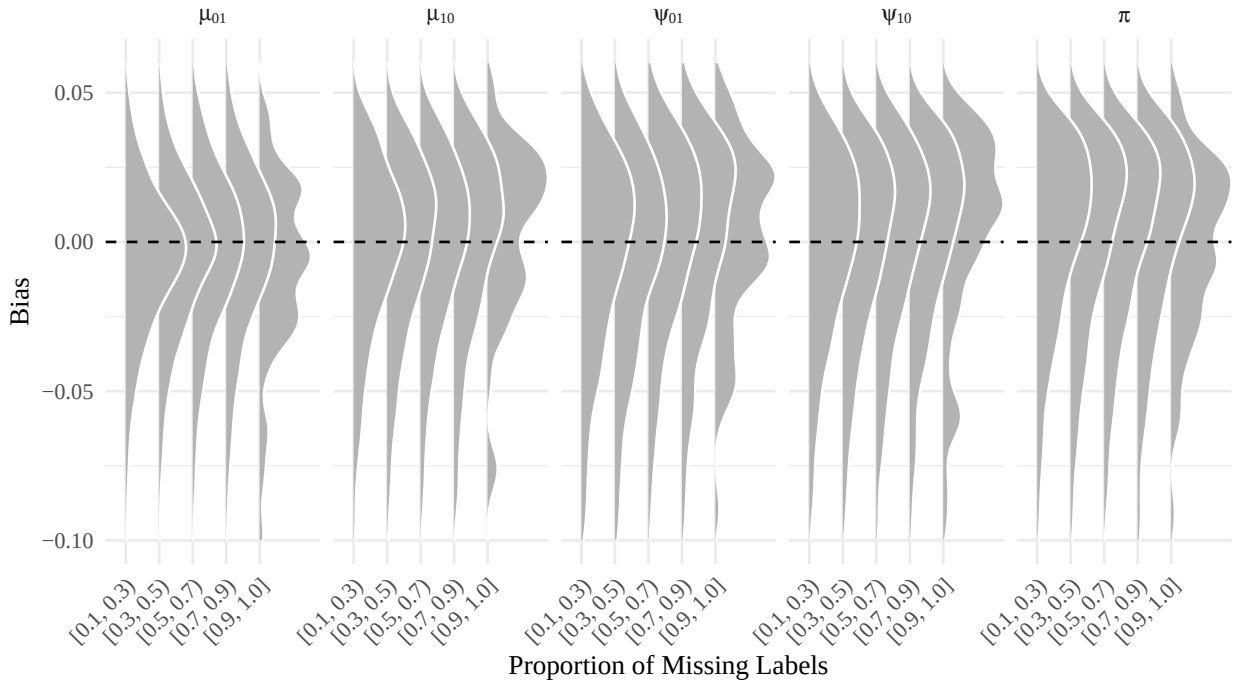


Figure 2.8: Empirical bias distribution for the partially annotated scenario by parameter and proportion of missing labels.

2.5.3 Limiting Probabilities

In the case that the model includes a single set of gain and loss probabilities, (i.e. no difference according to type of node), in the limit as the number of branches between the root and the node goes to infinity, the probability that a given node has a function can be calculated as

$$\begin{aligned}
\mathbb{P}(z_n = 1) &= (1 - \mu_{10})\,\mathbb{P}(z_{n-1} = 1) + \mu_{01}\mathbb{P}(z_{n-1} = 0), \\
&\quad \text{since in the limit } \mathbb{P}(z_n = 1) = \mathbb{P}(z_{n+k} = 1), \quad \forall k \\
&= (1 - \mu_{10})\,\mathbb{P}(z_n = 1) + \mu_{01}\mathbb{P}(z_n = 0), \\
&\quad \text{finally} \\
\mathbb{P}(z_n = 1) &= \frac{\mu_{01}}{\mu_{01} + \mu_{10}} \qquad (2.11)
\end{aligned}$$

However, when the gain and loss probabilities differ by type of node, the unconditional probability of observing having function is computed as follows:

$$\mathbb{P}(z_n = 1) = \mathbb{P}(z_n = 1 \mid C_n = D)\mathbb{P}(C_n = D) + \mathbb{P}(z_n = 1 \mid C_n = \neg D)\mathbb{P}(C_n = \neg D) \quad (2.12)$$

Where $C_n \in \{D, \neg D\}$ denotes the type of event (duplication or not duplication, respectively). We need to calculate $\mathbb{P}(z_n = 1 \mid C_n = D)$ and $\mathbb{P}(z_n = 1 \mid C_n = \neg D)$. WLOG let's start by the first:

$$\begin{aligned}\mathbb{P}(z_n = 1 \mid C_n = D) =& \mathbb{P}(z_n = 1 \mid C_n = D, z_{n-1} = 1)\mathbb{P}(z_{n-1} = 1 \mid C_n = D) + \\ & \mathbb{P}(z_n = 1 \mid C_n = D, z_{n-1} = 0)\mathbb{P}(z_{n-1} = 0 \mid C_n = D) \\ =& \left(1 - {\mu_{10}}^D\right)\mathbb{P}(z_{n-1} = 1 \mid C_n = D) + {\mu_{01}}^D\mathbb{P}(z_{n-1} = 0 \mid C_n = D)\end{aligned}$$

Now, following the same argument made in (2.11),

$$\mu^D = \frac{{\mu_{01}}^D}{{\mu_{01}}^D + {\mu_{10}}^D}$$

Where $\mu^D = \mathbb{P}(z_n \mid D_n, D_{n-1})$. Likewise, $\mu^{\neg D} = \frac{{\mu_{01}}^{\neg D}}{{\mu_{01}}^{\neg D} + {\mu_{10}}^{\neg D}}$. Observe that the parameter is only indexed by the class of the n-th leaf as the class of its parent is not relevant for this calculation.

$$\mathbb{P}(z_n = 1) = \mu^D\mathbb{P}(C_n = D) + \mu^{\neg D}\mathbb{P}(C_n = \neg D) \quad (2.12')$$

Where the probability of a node having a given class can be approximated by the observed proportion of that class in the given tree.

Acknowledgements

Supported by National Cancer Institute Grant #1P01CA196596. Computation for the work described in this paper was supported by the University of Southern California's Center for High-Performance Computing (https://hpcc.usc.edu).

Chapter 3

Discrete Exponential Family Models

3.1 Introduction

Discrete Exponential Family Models (DEFMs) [4] have a long tradition in various disciplines. First studied as Ising models, which were originally developed to explain ferromagnetism [63, if you understand German], or, as we will see in later chapters, used for understanding the local-phenomena that govern macro structures in social networks with Exponential Family Random Graph Models (ERGMs) [39, 55, 93, 104, 121, and others], DEFMs have thus drawn attention, and hence matured, from various fields.

In general, one of the most recognizable features of this family of statistical models is the fact that the normalizing constant of the likelihood function is generically intractable, meaning that exact calculation using these models is elusive for all but the few cases in which the support can be fully enumerated. Because of this, most research has focused on ways to estimate the likelihood without having to deal with the normalizing constant, and thus, most current approaches used to fit these models rely on simulation-based methods. Yet, there is still much we can do with the cases that most of the literature has considered irrelevant - those in which approximation is not necessary.

In this chapter, we will focus on DEFMs for binary arrays that have a support that is sufficiently

small so that the likelihood function is tractable. I will show some of the computational aspects of estimation when it is feasible in this context, and further discuss some statistical properties of this class that can sometimes be useful. Ultimately, what I will present here serves as an preamble to the following chapters in which I will feature different application of DEFMs.

Taking advantage of the fact that ERGMs are perhaps the most relatable framework in which to understand these discrete binary families–notably because sufficient statistics in ERGMs describe familiar structures like dyads, triangles, and stars–the chapter will be developed using graphs as the main reference point. Overall, the only difference with respect to other types of binary arrays is the fact that graphs here are represented as *square* binary arrays.

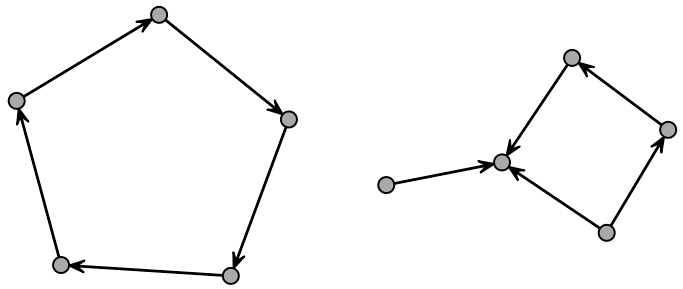
3.2 Fundamentals

In the case of Exponential-Family Random Graph Models, the unit of analysis corresponds to what could either be a directed or un-directed graph. Furthermore, it is commonly assumed that the analyzed network has no self ties (meaning that no vertex is directly connected to itself) and has been drawn from a population with a known and finite number of vertices, denoted by the set $\mathcal{G}$. Under the ERGM specification, the probability of observing a graph $\mathbf{g} \in \mathcal{G}$ is described as follows:

$$\mathbb{P}\left(\mathbf{G} = \mathbf{g} \mid X = x\right) = \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}, x\right)\right\}}{\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}', x\right)\right\}}, \quad \forall \mathbf{g} \in \mathcal{G}$$

where X is an array of vertex attributes (e.g., covariates), θ is a column-vector of length k (model parameters), and $s\left(\cdot\right)$ is a function that returns a column-vector of sufficient statistics, also of length k, which may or may not depend on X. Expanding the expression $\theta^{\mathbf{t}} s\left(\mathbf{g}, x\right)$ we see that this is ultimately a linear combination of the form $\theta_1 s\left(\mathbf{g}, x\right)_1 + \cdots + \theta_k s\left(\mathbf{g}, x\right)_k$. In simple terms, rather than predicting the graph as a whole, i.e. tie-by-tie, ERGMs capture the key features of the graph using the observed vector of sufficient statistics $s\left(\mathbf{g}, X\right)$, meaning that under this model two topologically different graphs can be equally likely if they have the same set of observed sufficient statistics. An example of this is illustrated in Figure 3.1, where both networks, while

they may seem very different, have the same probability of being observed under the Bernoulli Random Graph model, which is equivalent to an ERGM in which $s(\cdot) = \#$ Edges.



(a) A ring (b) A kite

Figure 3.1: Although these two networks may look very different, *a ring* and *a kite*, under the Bernoulli, also known as Erdős-Rényi, random graph model, both are equally likely, as the only sufficient statistic for this model is the number of observed ties, which is five in both cases.

While this equation can be found in practically every paper discussing ERGMs, outside the discussion of the Log-likelihood function, to the best of my knowledge neither the Gradient nor the Hessian functions of ERGMs have been the focus of any research papers. The reason for this could be that most modern estimation methods rely on approximating the ratio of the normalizing constants for two different sets of parameter values, in particular, that for the ERGM with the true population parameter, θ, over that for the ERGM with an arbitrary but known parameter θ_0. To be explicit, the ratio of these two normalizing constants is:

$$\begin{aligned}
\frac{\sum_{\mathbf{g}\in\mathcal{G}} \exp\left\{\theta^{\mathbf{t}} s(\mathbf{g}, X)\right\}}{\sum_{\mathbf{g}\in\mathcal{G}} \exp\left\{{\theta_0}^{\mathbf{t}} s(\mathbf{g}, X)\right\}} &= \sum_{\mathbf{g}\in\mathcal{G}} \left(\frac{1}{\sum_{\mathbf{g}\in\mathcal{G}} \exp\left\{{\theta_0}^{\mathbf{t}} s(\mathbf{g}, X)\right\}} \times \exp\left\{\theta^{\mathbf{t}} s(\mathbf{g}, X)\right\} \right) \\
&= \sum_{\mathbf{g}\in\mathcal{G}} \left(\frac{\exp\left\{{\theta_0}^{\mathbf{t}} s(\mathbf{g}, X)\right\}}{\sum_{\mathbf{g}\in\mathcal{G}} \exp\left\{{\theta_0}^{\mathbf{t}} s(\mathbf{g}, X)\right\}} \times \exp\left\{(\theta - \theta_0)^{\mathbf{t}} s(\mathbf{g}, X)\right\} \right) \\
&= \sum_{\mathbf{g}\in\mathcal{G}} \left(\mathbb{P}(\mathbf{G} = \mathbf{g} \mid \theta_0, X) \times \exp\left\{(\theta - \theta_0)^{\mathbf{t}} s(\mathbf{g}, X)\right\} \right) \\
&= \mathbb{E}\left\{ \exp\left\{(\theta - \theta_0)^{\mathbf{t}} s(\mathbf{g}, X)\right\} \,\middle|\, \theta_0, X \right\}
\end{aligned}$$

where the last equality follows from the definition of expected value [45]. Since θ_0 is known, this quantity can be approximated using the law of large numbers. In particular, this can be used to maximize the ratio of log-likelihoods. The objective function of this maximization problem can then be approximated by simulating m networks from the distribution with parameter θ_0:

$$l(\theta) - l(\theta_0) \approx (\theta - \theta_0)^{\mathbf{t}} s(\mathbf{g}_{obs}, X) - \log\left\{\frac{1}{m}\sum_{i=1}^{m} \exp\left\{(\theta - \theta_0)^{\mathbf{t}}\right\} s(\mathbf{g}_i, X)\right\}.$$

For more details see [61].

As mentioned in the introduction to this chapter, when dealing with small networks, we do not need to approximate these quantities, as exhaustive enumeration allows us to compute them exactly. In the rest of this section, and for the sake of completeness, I will declare the exact form that both the Gradient and Hessian functions of ERGMs have.

Before we proceed, from this point and until the end of this section, we will be looking at the centered version of the sufficient statistics, in particular, all equations in this section will be derived from the following an **equivalent** re-statement of the likelihood function:

$$\mathbb{P}(\mathbf{G} = \mathbf{g} \mid \theta, X = x) = \frac{1}{\sum_{\mathbf{g}' \in \mathcal{G}} \exp\left\{\theta^{\mathbf{t}} \Delta s(\mathbf{g}', x)\right\}}, \quad \forall \mathbf{g} \in \mathcal{G}$$

where $\Delta s(\mathbf{g}') = s(\mathbf{g}', x) - s(\mathbf{g}, x)$. Which, from the computational point of view, is easier to manipulate.

Log-likelihood the ERGM log-likelihood is given by the following equation:

$$l(\theta) = -\log\left\{\sum_{\mathbf{g}' \in \mathbf{G}} \exp\left\{\theta^{\mathbf{t}} \Delta s(\mathbf{g}')\right\}\right\}$$

In general, since from the perspective of $s(\mathbf{g}, X)$ sets of topologically different graphs can be thought as equivalent, we can reduce the computational complexity of this calculations by looking at the isomorphic sufficient statistics, this is, group up elements based on the set of

unique vectors of sufficient statistics:

$$-\log\left\{\sum_{\mathbf{s}\in\mathcal{S}}\mathbf{w_s}\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}\right\}$$

where $\mathcal{S}$ is the support of the sufficient statistics under $\mathcal{G}$, $\mathbf{s}\in\mathcal{S}$ is a realized set of values, and $\mathbf{w_s}\equiv\left|\left\{\mathbf{g}\in\mathcal{G}:\Delta s\left(\mathbf{g}\right)=\mathbf{s}\right\}\right|$ is the number of networks in $\mathcal{G}$ for which the centered sufficient statistics equal $\mathbf{s}$. Furthermore, we can write this in matrix form:

$$-\log\left\{\mathbf{W}\times\exp\left\{\mathbf{S}\times\theta\right\}\right\}$$

where $\mathbf{W}\equiv\left\{\mathbf{w_s}\right\}_{\mathbf{s}\in\mathbf{S}}$ is a row vector of, length w and $\mathbf{S}$ is a matrix of size $w\times k$.

Gradient The partial derivative of the log-likelihood with respect to the j-th parameter is:

$$\frac{\delta}{\delta\theta_j}l\left(\theta\right)=-\frac{\sum_{\mathbf{s}\in\mathcal{S}}\mathbf{w_s}\mathbf{s}_j\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}}{\sum_{\mathbf{s}\in\mathcal{S}}\mathbf{w_s}\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}},\quad\forall j.$$

We can also write this using matrix notation as follows:

$$\nabla l\left(\theta\right)=-\mathbf{S^t}\times\left[\mathbf{W}\circ\exp\left\{\mathbf{S}\times\theta\right\}\right]/\lambda(\theta),$$

where $\circ$ is the element-wise product and $\lambda(\theta)=\mathbf{W}\times\exp\left\{\mathbf{S}\times\theta\right\}$.

Hessian In the case of the hessian, the $(j,l)th$ element of $\frac{\delta^2}{\delta\theta_k\delta\theta_u}l\left(theta\right)$ can be computed as:

$$\frac{-\left(\sum_{\mathbf{s}'\in\mathcal{S}}\mathbf{s}'_j\mathbf{s}'_l\mathbf{w_s}\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}\right)}{\lambda(\theta)}+\frac{\left(\sum_{\mathbf{s}'\in\mathcal{S}}\mathbf{s}'_j\mathbf{w_s}\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}\right)\left(\sum_{\mathbf{s}'\in\mathcal{S}}\mathbf{s}'_l\mathbf{w_s}\exp\left\{\theta^{\mathbf{t}}\mathbf{s}\right\}\right)}{\lambda(\theta)^2}$$

where $\mathbf{s}_j$ as the j-th element of the vector $\mathbf{s}$. Once again, we can simplify this using matrix notation:

$$\frac{-\mathbf{W}\times\left[\mathbf{S}_j\circ\mathbf{S}_l\circ\exp\left\{\mathbf{S}\times\theta\right\}\right]}{\lambda(\theta)}+\frac{\left(\mathbf{W}\times\left[\mathbf{S}_j\circ\exp\left\{\mathbf{S}\times\theta\right\}\right]\right)\left(\mathbf{W}\times\left[\mathbf{S}_l\circ\exp\left\{\mathbf{S}\times\theta\right\}\right]\right)}{\lambda(\theta)^2}$$

where $\mathbf{S}_j$ is the j-th column of the matrix $\mathbf{S}$.

3.3 Asymptotic Properties of Ill-defined Models

In the case that the MLE does not exist, i.e. $\theta_k \to \pm\infty$, which occurs when the observed sufficient statistics are not in the interior of the support - for example, for a fully connected network - the limit of the log-likelihood, the gradient, and hessian are finite. This is relevant as special care needs to be taken when dealing with these cases.

While, in general, models in which the MLEs diverge may seem uninteresting, the fact that ERGMs describe a discrete rather than continuous random variable makes this type of event than for other families of distributions. Furthermore, in the case of methods such as bootstrapping, forward/backward selection, or other classes of algorithms that involve some level of automation during the model fitting process, it is important to know how to correctly deal with the non-existence of MLEs. Fortunately, as we will show, in the limit, the log-likelihood and its derivatives are all well defined.

Given that the k-th sufficient statistic lies on the boundary, and thus the MLE of that parameter equals $\pm\infty$, there are two possible cases:

1. **The observed value of the k-th sufficient statistic is at the *lower* bound** $\mathbf{s}_k = \min_{\mathbf{s}' \in \mathcal{S}} \mathbf{s}'_k$ This happens, for example, when trying to fit a model using triangles as a sufficient statistic on a graph that features no triangles. In this case, since the sufficient statistics are centered around the observed values, we have $\mathbf{s}'_k \geq 0$, the theoretical MLE for θ_k goes to $-\infty$, and thus, in the limit, we have

$$\lim_{\theta_k \to -\infty} \exp\left\{\theta^{\mathbf{t}} \mathbf{s}'\right\} = \begin{cases} \exp\left\{\sum_{j \neq k} \mathbf{s}'_j \theta_j\right\} & \text{if } \mathbf{s}'_k = 0 \\ 0 & \text{if } \mathbf{s}'_k > 0 \end{cases}$$

2. **The observed value of the k-th sufficient statistic is at the *upper* bound** $\mathbf{s}_k = \max_{\mathbf{s}' \in \mathcal{S}} \mathbf{s}'_k$ The most common case here is when the sufficient statistic is saturated, for

example, in a fully connected graph, where the MLE goes to infinity, $\theta_k \to +\infty$. Similar to before, since again the sufficient statistics are centered around the observed values we have $\mathbf{s}'_k \leq 0$, in the limit the previous expression is well defined:

$$\lim_{\theta_k \to +\infty} \exp\left\{\theta^{\mathbf{t}} \mathbf{s}'\right\} = \begin{cases} \exp\left\{\sum_{j \neq k} \mathbf{s}'_j \theta_j\right\} & \text{if } \mathbf{s}'_k = 0 \\ 0 & \text{if } \mathbf{s}'_k < 0 \end{cases}$$

The two previous statements can be interpreted as follows: only graphs whose k-th sufficient statistic matches that of the observed data influence the likelihood and therefore the fitting of the model. Therefore, a key to compute the limiting values of the Log-likelihood, Gradient, and Hessian will be partitioning the summations over $\mathbf{s}' \in \mathcal{S}$ into two different sets as follows:

$$\mathcal{S}_0 \equiv \{\mathbf{s} \in \mathcal{S} : \mathbf{s}_u = 0, \forall u \in U\} \tag{3.1}$$

$$\mathcal{S}_1 \equiv \mathcal{S} \setminus \mathcal{S}_0 \tag{3.2}$$

where U is the set of sufficient statistics that are on the boundary on the observed graph, and thus have an MLE that diverges to $\pm\infty$, e.g. $U = \{\text{triangle term}\}$ in a model with no observed triangles. In this partition $\mathcal{S}_0$ contains all the realizations of sufficient statistics for which the statistics in U match the corresponding observed sufficient statistics. With this definition in hand, we will now show the asymptotic behavior of the functions in question as the parameter diverges to infinity.

Log-likelihood Using the above partition of $\mathcal{S}$, the log-likelihood can be written as follows

$$-\log\left\{\sum_{\mathbf{s}' \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}'} \exp\left\{\sum_{j \notin U} \mathbf{s}'_j \theta_j\right\} + \sum_{\mathbf{s}' \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}'} \exp\left\{\theta^{\mathbf{t}} \Delta s\left(\mathbf{g}'\right)\right\}.\right\}$$

Then, without loss of generality, as $\theta_u \to -\infty$:

$$
\begin{aligned}
&\lim_{\theta_u \to -\infty} -\log\left\{\sum_{\mathbf{s}' \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}'} \exp\left\{\sum_{j \notin U} \mathbf{s}'_j \theta_j\right\} + \sum_{\mathbf{s}' \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}'} \exp\left\{\theta^{\mathbf{t}} \Delta s\left(\mathbf{g}'\right)\right\}\right\} \\
&= -\log\left\{\lim_{\theta_u \to -\infty} \sum_{\mathbf{s}' \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}'} \exp\left\{\sum_{j \notin U} \mathbf{s}'_j \theta_j\right\} + \sum_{\mathbf{s}' \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}'} \exp\left\{\theta^{\mathbf{t}} \Delta s\left(\mathbf{g}'\right)\right\}\right\} \\
&= -\log\left\{\sum_{\mathbf{s}' \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}'} \exp\left\{\sum_{j \notin U} \mathbf{s}'_j \theta_j\right\}\right\}
\end{aligned}
$$

where the second equality follows from the fact that the logarithm function is continuous over the domain $(0, 1)$.

Gradient The gradient for θ_j is wrriten as

$$
-\frac{\sum_{\mathbf{s} \in \mathcal{S}} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}}{\sum_{\mathbf{s} \in \mathcal{S}} \mathbf{w}_{\mathbf{s}} \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}} = -\frac{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\} + \sum_{\mathbf{s} \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}}{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\} + \sum_{\mathbf{s} \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}} \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}}.
$$

Without loss of generality, if $\mathbf{s}_u = \min_{\mathbf{s}' \in \mathcal{S}} \mathbf{s}'_u$, the limit of the above expression as $\theta_u \to -\infty$ is evaluated as follows:

$$
\lim_{\theta_u \to -\infty} - \frac{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\} + \sum_{\mathbf{s} \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}}{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\} + \sum_{\mathbf{s} \in \mathcal{S}_1} \mathbf{w}_{\mathbf{s}} \exp\left\{\theta^{\mathbf{t}} \mathbf{s}\right\}} \tag{3.3}
$$

$$
= -\frac{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \mathbf{s}_j \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\}}{\sum_{\mathbf{s} \in \mathcal{S}_0} \mathbf{w}_{\mathbf{s}} \exp\left\{\sum_{j \notin U} \theta_j \mathbf{s}_j\right\}}. \tag{3.4}
$$

If $j = u$, the above expression reduces to 0 as $\mathbf{s}'_u = 0 \; \forall \; \mathbf{s}' \in \mathcal{S}_0$, otherwise the number is well defined.

Hessian Just like the other cases, we can rewrite the hessian partitioning the function into summations over $\mathcal{S}_0$ and $\mathcal{S}_1$. for brevity, we do not rewrite the whole expression, but without loss of

generality, the limiting Hessian, and in particular, its (j,l)-th component, as $\theta_u \to -\infty$ equals:

$$\frac{-\left(\sum_{\mathbf{s}'\in\mathcal{S}_0}\mathbf{s}'_j\mathbf{s}'_l\mathbf{w}_{\mathbf{s}}\exp\left\{\sum_{j\notin U}\theta_j\mathbf{s}_j\right\}\right)}{\sum_{\mathbf{s}\in\mathcal{S}_0}\mathbf{w}_{\mathbf{s}}\exp\left\{\sum_{j\notin U}\theta_j\mathbf{s}_j\right\}}+ \frac{\left(\sum_{\mathbf{s}'\in\mathcal{S}_0}\mathbf{s}'_j\mathbf{w}_{\mathbf{s}}\exp\left\{\sum_{j\notin U}\theta_j\mathbf{s}_j\right\}\right)\left(\sum_{\mathbf{s}'\in\mathcal{S}_0}\mathbf{s}'_l\mathbf{w}_{\mathbf{s}}\exp\left\{\sum_{j\notin U}\theta_j\mathbf{s}_j\right\}\right)}{\left(\sum_{\mathbf{s}\in\mathcal{S}_0}\mathbf{w}_{\mathbf{s}}\exp\left\{\sum_{j\notin U}\theta_j\mathbf{s}_j\right\}\right)^2}.$$

In the case that either j or l is equal to u, the hessian equals 0. For values other than u, the hessian is non-zero. This last result is useful in that we can then apply the Moore-Penrose (see [27, 89]) generalized inverse and thus use a pseudo-covariance matrix, which is a common approach taken by practitioners in a variety of fields [46, 117]. Furthermore, the limiting expressions of the log-likelihood, gradient, and hessian have less terms to be consider, which can drastically reduce the computational complexity of these functions. Appendix B provides some examples illustrating how to calculate these values using the *ergmito* R package [114].

For further details about the existence of MLEs, model degeneracy and related issues pertaining discrete exponential family models, see e.g., [34, 64, 92, 97].

3.4 The Conditional Distribution

Conditional graph distributions can be considered to have a rather long history in complex systems analysis (e.g., CUG tests in the *social networks* literature [2, 35] and rewiring algorithms in the *network science* literature [80, 81]). Across various disciplines, the usual strategy is to analyze, or try to characterize differences across types of networks, by means of permutation tests. In the context of networks, the principle consists of asserting whether an observed network, or set of networks, differs from a null network distribution generated as a function of *rewiring* the observed network while fixing one of its properties: for example, the overall degree distribution, the degree sequence, the number of clusters, etc. ERGMs can be viewed as a generalized version of all these tests [12], and from this perspective conditional ERGM distributions may prove very useful.

Formally, the marginal distribution of the sufficient statistic s_k, conditional on sufficient statistic s_l, can be calculated as follows:

$$\begin{aligned}
\mathbb{P}\left(s\left(\mathbf{G}\right)_k = s_k \mid s\left(\mathbf{G}\right)_l = s_l, \theta\right) =& \frac{\mathbb{P}\left(s\left(\mathbf{G}\right) = s\left(\mathbf{g}\right) \mid \theta\right)}{\mathbb{P}\left(s\left(\mathbf{G}\right)_l = s_l \mid \theta\right)} \\
& \text{(since both have the same normalizing constant)} \\
=& \frac{\exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}\right)\right\}}{\sum_{\mathbf{g}': s(\mathbf{g}')_l = s(\mathbf{g})_l} \exp\left\{\theta^{\mathbf{t}} s\left(\mathbf{g}'\right)\right\}} \\
& \text{(furthermore, this is invariant to } \theta_l\text{)} \\
=& \frac{\exp\left\{\theta_{-l}^{\mathbf{t}} s\left(\mathbf{g}\right)_{-l}\right\} \exp\left\{s_l \theta_l\right\}}{\sum_{\mathbf{g}': s(\mathbf{g}')_l = s(\mathbf{g})_l} \exp\left\{\theta_{-l}^{\mathbf{t}} s\left(\mathbf{g}'\right)_{-l}\right\} \exp\left\{s_l \theta_l\right\}} \\
=& \frac{\exp\left\{\theta_{-l}^{\mathbf{t}} s\left(\mathbf{g}\right)_{-l}\right\}}{\sum_{\mathbf{g}': s(\mathbf{g}')_l = s(\mathbf{g})_l} \exp\left\{\theta_{-l}^{\mathbf{t}} s\left(\mathbf{g}'\right)_{-l}\right\}}.
\end{aligned} \tag{3.5}$$

In other words, once we condition on the l-th sufficient statistic, the marginal distribution of s_k becomes invariant to the value of θ_l since all the information is already contained in s_l. A very interesting property that is akin to the "memory-less" property of the Exponential distribution. Chapter 6 uses this in the context of goodness-of-fit assessment.

Chapter 4

Next Steps for Phylogenetic Models

4.1 Hierarchical Bayesian Framework

One natural extension of the phylogenetic model presented in chapter 2 is to use a hierarchical-Bayes approach in which, instead of assuming that all functions share the same set of population parameters, we can build a model in which each tree has different rates of functional gain and loss drawn from a common distribution. this enables us to "borrow strength" across trees when inferring parameter values. Formally, in a general hierarchical-Bayes framework we have the following structure:

$$\tilde{D}_t \sim f(\theta_t), \quad \theta_t \sim \text{Beta}\left(\alpha_\theta, \theta_\theta\right), \quad (\alpha_\theta, \theta_\theta) \sim h(\omega) \tag{4.1}$$

where $\tilde{D}_t$ is the t-th observed annotated phylogenetic tree, θ_t is the set of model parameters described in chapter 2, i.e., mis-classification, gain, loss, etc., associated with that tree, $(\alpha_\theta, \theta_\theta)$ are the hyper parameters for θ, and $h(\omega)$ is a hyper-prior for the hyper-parameters. This is similar to what is viewed in the literature as a random-effects model. Here, while we are relaxing the assumption that all trees/functions have exactly the same evolutionary rates, we are still assuming that those rates come from the same distribution, which at some level could still be too stringent

(if there were in fact two populations, say).

In our case, if we wanted to estimate the model described in (4.1) using the 141 experimentally annotated trees showcased in chapter 2, excluding hyper-priors, we would need to fit a model with $7 \times 141 + 7 \times 2 = 1,001$ parameters, a daunting task regardless of the maximization approach taken. Because the estimation of models of the type of (4.1) is in general very difficult, we can follow a simplified approach using *Empirical Bayes* [30]. The Empirical Bayes approach provides a very good alternative to the full search by instead fixing the population-level parameters at values that are broadly consistent with the currently observed data.

In simple terms, Empirical Bayes works as follows:

1. We start by obtaining a raw average of the population-level parameters (α_θ and θ_θ in (4.1)), call these $(\hat{\alpha}_\theta, \hat{\theta}_\theta)$,

2. We then fix $(\alpha_\theta, \theta_\theta)$ to $(\hat{\alpha}_\theta, \hat{\theta}_\theta)$, and estimate the model described in (4.1). While the number of parameters to estimate does not dramatically decrease, the fact that we are fixing the hyper-parameters to these data-driven estimates turns out to lead to an important improvement in the stability of the log-likelihood function.

The only problem with this "standard" Empirical Bayes procedure is that we may have no direct way of obtaining raw estimates of $(\alpha_\theta, \theta_\theta)$. To overcome this, we can rely on the pooled-data estimates to obtain an approximation to the hyper-parameters. In particular, we can proceed as follows:

1. Fit the pooled data model with a uniform (frequentist) prior as described in chapter 2 using MCMC. This will result in a posterior distribution for the evolutionary model parameters.

2. Using either the Method of Moments [MoM] or Maximum Likelihood Estimation [MLE], fit a model to obtain $(\hat{\alpha}_\theta, \hat{\theta}_\theta)$. This procedure would yield a total of seven pairs of parameters estimates, in particular, one alpha and theta pair for each of the seven parameters of our gene-function evolutionary model, $(\psi_{01}, \psi_{10}, \mu_{01d}, \mu_{10d}, \mu_{01s}, \mu_{10s}, \pi)$.

3. Finally, with the new set of $(\hat{\alpha}_\theta, \hat{\theta}_\theta)$, fit the joint hierarchical model using MCMC.

Both strategies, either using a fully-Bayesian approach to estimate the hyper-parameters, or using Empirical Bayes to fix the hyper-parameters to data-informed values, are now fully-implemented in my `aphylo` R package.

At this point in time, I have tested this model using both methods, full hierarchical Bayesian and Empirical Bayes, with limited success. The most successful analysis so far came from fitting a model using the Empirical Bayes approach on a subset of the 141 trees using only annotations tagged as *Molecular Function*. This included a total of 74 functions/trees. While the entire procedure did not fully converge under the criterion described in chapter 2 (running four chains and having the Gelman-Rubin statistic below the 1.1 threshold [42]), ultimately, the predictions generated by this model were similar to those generated by the pooled-data model including the 141 trees. We note that convergence of individual-level parameters in such contexts is often challenge, so frequently the focus is on population-level inference, or, in a context such as ours, final imputation accuracy. As an alternative to this fully integrated model, in the next section I describe a simpler modeling strategy that leverages the fact that annotations can be grouped in three natural major *classes*.

4.2 Pooling by Class of Annotation

While a Hierarchical Bayesian approach might be viewed as the correct "full glory" statistical treatment of the problem, as noted in the previous section such models often behave poorly, and their computational complexity can often grow very quickly, making the estimation process a daunting task, and in some cases, infeasible. So here we provide a simpler, more pragmatic alternative, that still seems to perform well.

In our context the gene annotations can be classified in three major classes. So, instead of modeling each gene-function as having its own set of evolutionary rates, all drawn from a common population distribution, a more pragmatic approach is to assume that functions within the same

class of annotations share a common set of model parameters. In the Gene Ontology project [GO project] [3, 22], gene functions are classified as being related to one of three different functions:

(a) Molecular function: genes whose products relate to molecular activities.

(b) Biological process: genes whose products contribute to pathways and larger processes.

(c) Cellular component: gene whose products are active in specific cellular components.

Given this information, before fitting the full Hierarchical Bayes model using the three classes, I first approached the data by estimating a single model per type of annotation. From the original set of 141 experimentally annotated trees that were used in chapter 2, I fitted three different models including 74, 45, and 22 trees with annotations of type molecular function, biological process, and cellular component, respectively. So now we assume common parameter values across all trees within each of the three classes. Table 4.1 shows the resulting parameter estimates fitted using the same Markov Chain Monte Carlo method, and using a uniform prior, as described in the aforementioned chapter.

There are four major points to highlight from these new sets of results. First, the parameter estimates are significantly different from those obtained with the pooled-data approach. While the *molecular function* estimates are very close to those obtained with the former model, the parameter estimates for *biological process* and *cellular component* are very different. The model suggests that gene functions associated to biological processes have a significantly lower rate of mutation, changing from the 97% chance obtained in the pooled-data model, to just a 10% probability of gaining a function at a duplication event for this model. Similarly, estimated loss of function rate at duplication nodes changes from 51% in the pooled-data model to just 3% in the model restricted just to genes involved in biological processes. Second, prediction accuracy significantly improved in the case of *biological process*. Compared to a Mean Absolute Error (MAE) of 0.34 in the pooled-data model, MAE under the new strategy decreased drastically to 0.26; the same is not true for the case of *molecular function*, which did not observe any meaningful changes, and *cellular component*, which, on the contrary, resulted on significantly

	Pooled	Type of Annotation		
		Molecular Function	Biological Process	Cellular Component
Mislabeling				
ψ_{01}	0.23	0.18	0.09	0.66
ψ_{10}	0.01	0.01	0.01	0.33
Duplication Events				
μ_{d01}	0.97	0.97	0.10	0.55
μ_{d10}	0.52	0.51	0.03	0.56
Speciation Events				
μ_{s01}	0.05	0.05	0.05	0.37
μ_{s10}	0.01	0.01	0.02	0.37
Root node				
π	0.79	0.71	0.88	0.52
Trees	141	74	45	22
Accuracy under the by-aspect model				
AUC	-	0.77	0.83	0.53
MAE	-	0.34	0.26	0.50
Accuracy under the pooled-data model				
AUC	-	0.77	0.75	0.75
MAE	-	0.35	0.34	0.37

Table 4.1: MCMC estimates for experimentally annotated trees. The first column shows the estimates under the pooled-data model in 2.4, while the following three columns report the estimates obtained when fitting the model using a pooled-data approach, but doing so by type of annotation. Readers should be aware that the estimation process of the fourth column, *cellular component*, did not fully converge, likely due to sparsity of annotations within that category.

lower prediction accuracy. We note that the number of genes related to *cellular component* is small, and the amount of annotation on those phylogenies is low, perhaps explaining the loss of accuracy here. Third, as further evidence of this, parameter estimates for the *cellular component* model are not informative whatsoever. The posterior sample never reached a stable point, which is why the posterior averages, which are the values reported in the table, are all relatively close to 0.5; this is the same behavior observed in estimation processes involving a single tree with no informative prior. It is possible that estimates will improve when a better fit is obtained. Finally, because the parameter estimates of *molecular function* are so close to those obtained in the pooled-data model, it is reasonable to believe that trees with molecular functions were driving the pooled-data estimation process.

Regarding the latter point, we can perform a sensitivity analysis by using cross-validation and measuring how much influence each tree, or group of trees, has on the overall estimation process. While the `aphylo` package has some of the required infrastructure needed for such analyses, more work needs to be done. Lastly, and perhaps of great importance, even though prediction accuracy improved for genes related to *biological process*, the corresponding parameter estimates are not in line with the common understanding that the evolutionary rates of *biological processes* should be higher compared to those of *molecular functions*, (our model suggests the opposite). We will explore this issue in the near future.

In conclusion, these results suggest that it may not be reasonable to simply group all genes together when performing a joint analysis of the data. dividing them into natural classes, as we did here, is one reasonable response to this, given the difficulties of fitting the fully hierarchical model. However, in the following section, I present a promising alternative to this model that builds on the ideas presented in chapter 3.

4.3 Transition Probabilities as a Function of Sufficient Statistics

An important problem that we have met in our analyses so far is the sparseness of annotations in the experimental data in general, but particularly in those of the type *absent* (no function). This lack of *NOTs* in the data was what motivated us to propose a pooled-data approach, so that we could augment the available data with the hope of improving the accuracy of our parameter estimates. Yet, as illustrated in the previous section, even if the Hierarchical Bayes approach proves successful, we are still assuming that both functions and siblings evolve independently, which is an unrealistic assumption.

We now build a generalized version of the model in which both function and sibling inter-dependence are incorporated. Using the ideas developed in chapter 3 for discrete exponential distributions, instead of analyzing the functional gains and losses of siblings as what, in principle, were *independent and identically distributed* Bernoulli variables, parameterized by μ_{01}/μ_{10}, we can instead model the transition probabilities using a discrete exponential distribution:

$$\mathbb{P}\left(\mathbf{X}=\mathbf{x} \mid x_n\right)=\frac{\exp\left\{\mu^{\mathbf{t}} s(\mathbf{x}, x_n)\right\}}{\sum_{\mathbf{x}'} \exp\left\{\mu^{\mathbf{t}} s(\mathbf{x}', x_n)\right\}} \tag{4.2}$$

where $\mathbf{x} \equiv \{x_{n1}, x_{n2}, \dots\}$ is an array of size P (functions) $\times$ $|\mathbf{O}(n)|$ (offspring) representing the state of node n's offspring, x_n is a binary vector representing the state of node n, μ is a column vector of parameters, and $s(\cdot)$ is a column vector of sufficient statistics which may include terms such as: the total number of functional gains, the number of co-gains or co-loses, etc. The point here is that by choosing appropriate summary statistics, and associated coefficients, we can enforce appropriate degrees of biological reality upon the allowed transitions in the offspring nodes for the set of genes under consideration. We give an example of this below.

While this calculation is indeed highly complex in the sense that we need to go through all possible states of $\mathbf{x}$, as discussed in chapter 3, the isomorphism of the sufficient statistics imply that we don't need to fully enumerate $\mathbf{x}$ but rather only the support of the sufficient statistics,

which depending on the specified model could be significantly smaller, and therefore faster.

Compared to the more canonical approach to incorporate interdependencies between siblings and functions in which usually the entire joint Markov transition matrix is estimated, requiring us to fit a model with hundreds (if not tens of thousands) of parameters, this new approach has the potential of incorporating arbitrarily complex features of the Markov transition process while at the same time keeping the number of free parameters low, making it a computationally efficient alternative. Table 4.2 shows four concrete examples of what kinds of statistics could be included in the model.

		Transitions to	
		Case 1	Case 2
Parent	A $\begin{bmatrix} 0 \end{bmatrix}$ B $\begin{bmatrix} 1 \end{bmatrix}$ C $\begin{bmatrix} 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$
Sufficient statistics			
# functional gains		1	1
Only one offspring changes (yes/no)		1	0
# Changes (gain+loss)		2	3
Subfunctionalizations (yes/no)		0	1

Table 4.2: Examples of sufficient statistics for phylogentic modelling. The parent node has three functions, A, B, and C, and two offspring. The table shows what would be the counts for four different sufficient statistics that could be used in this context (functional gains, having a single offspring changing, number of total changes, and subfunctionalization, which consists of an offspring specializing by inheriting just a subset of its parent's functions), and in particular, what would be under two different cases.

In the next subsection I explain how this new strategy still fits within the framework Felsenstein's pruning algorithm [37], which is a key component for ensuring tractability of the overall model fitting process.

4.3.1 Independent Functions, Dependent Siblings

We illustrate in the case of a single gene function. In such a case, the main modification of the model would be on the pruning probabilities relating to the offspring set of each node. From the

original definition of the algorithm, we have:

$$\mathbb{P}\left(\tilde{D}_n \,\middle|\, x_n, \psi, \mu\right) = \prod_{m \in \mathbf{O}(n)} \mathbb{P}\left(\tilde{D}_m \,\middle|\, x_n\right), \tag{2.3}$$

In principle, the conditional probability of the induced-subtree can be inferred using the law of total probability by conditioning on the possible states of the offspring, $\mathbf{x} \equiv \{x_m\}_{m \in \mathbf{O}(n)} \in \{0,1\}^{|\mathbf{O}(n)|}$. As we are now assuming that these are jointly distributed, we write the following:

$$\mathbb{P}\left(\tilde{D}_n \,\middle|\, x_n, \psi, \mu\right) = \sum_{\mathbf{x}} \mathbb{P}\left(\tilde{D}_n \,\middle|\, \mathbf{x}, x_n\right) \mathbb{P}\left(\mathbf{x} \mid x_n\right)$$

(Since we are given the state of the offspring, we can rewrite this as the joint probability of each offspring-induced subtree, to get)

$$= \sum_{\mathbf{x}} \mathbb{P}\left(\tilde{D}_{m1}, \tilde{D}_{m2}, \ldots \,\middle|\, x_{m1}, x_{m2}, \ldots, x_n\right) \mathbb{P}\left(\mathbf{x} \mid x_n\right)$$

(Because of the Markov assumption, these probabilities are conditionally independent, and so we get)

$$= \sum_{\mathbf{x}} \mathbb{P}\left(\mathbf{x} \mid x_n\right) \prod_{m \in \mathbf{O}(n)} \mathbb{P}\left(\tilde{D}_m \,\middle|\, x_m, x_n\right)$$

$$= \sum_{\mathbf{x}} \mathbb{P}\left(\mathbf{x} \mid x_n\right) \prod_{m \in \mathbf{O}(n)} \mathbb{P}\left(\tilde{D}_m \,\middle|\, x_m\right) \tag{4.3}$$

The last line follows from another application of the Markov assumption, as information that x_n provides is irrelevant once we know x_m. This way, we have yet another recursive definition of the pruning probabilities.

4.3.2 Dependent Functions, Dependent Siblings

when we consider more than one function, the main difference with (4.3) is that x_n is now multivariate, so we have:

$$\mathbb{P}\left(\tilde{D}_n \,\middle|\, \mathbf{x}_n, \psi, \mu\right) = \sum_{\mathbf{x}} \mathbb{P}\left(\mathbf{x} \mid \mathbf{x}_n\right) \prod_{m \in \mathbf{O}(n)} \mathbb{P}\left(\tilde{D}_m \,\middle|\, \mathbf{x}_m\right) \tag{4.4}$$

derived in a similar way to that shown above.

4.3.3 Sufficient Statistics for the Phylogenetic Model

To be able to compute (4.4), we first need to know the normalizing constant of the transition probability, and hence, fully enumerate all possible states of $\mathbf{x}$ and calculate its corresponding sufficient statistics. While this only needs to be done once, it is desirable to reduce the computational complexity as much as possible since the algorithm will require us to do such enumeration for all nodes and for all possible states of the parent nodes.

Again, borrowing ideas from the ERGM literature, we can drastically reduce the computational complexity for calculating the support by using what is known as *change statistics*. In simple terms, since the process of exhaustive enumeration requires adding/removing one entry of the array/graph at a time, we can take advantage of this and, instead of fully counting all the functional gains, functional loses, functional co-gains (trying to capture correlation between pairs of functions), etc., we can keep track of a vector of *running* counts and look only at how the new deleted/added entry increments/decrements these statistics. Let's take a look at a couple of examples.

Functional gains The simplest term one could think of is counting the number of functional gains. In this case, the overall statistic is computed as follows

$$s\left(\mathbf{x}, x_n\right)_{\text{gains } k} = x_{nk} \sum_{o \in \mathbf{O}(n)} x_{mk}$$

where x_{nk} equals to one if the n-th node has function k, and zero otherwise. Notice that the whole statistic will be relevant if and only if the state of the parent node is 0. Now, let $\mathbf{x}^{+}_{hk}$ denote the matrix of offspring states in which the only change from $\mathbf{x}$ is that offspring h gained function k, then the change statistic equals:

$$\delta(h,k)_{\text{gains } k} = s\left(\mathbf{x}^{+}_{hk}, x_n\right)_{\text{gains } k} - s\left(\mathbf{x}, x_n\right)_{\text{gains } k} = \begin{cases} 1, & \text{if } x_{nk} = 0 \\ 0, & \text{otherwise} \end{cases}$$

thus, in essence, this statistic always increments in one unit if the parent node did not had such function.

Subfunctionalization One theoretical aspect that the simpler version of our model cannot capture is what is known as subfunctionalization. This happens when the offspring become specialized and each performs a subset of the functions inherited from its parent. A first approach to this could be looking at pairwise subfunctionalizations, this is, instead of analyzing all possible *types* of subfunctionalization, i.e. involving two, three, four functions, etc., we could start by looking at pairs of functions. For example, the sufficient statistic for counting how many subfunctionalization events we observe for functions (k, j) can be computed as follows:

$$s\left(\mathbf{x}, x_n\right)_{\text{subfun } kj} = \sum_{u<h:\{u,h\}\in\mathbf{O}(n)} \{x_{uj}(1-x_{uk})(1-x_{hj})x_{hk} + (1-x_{uj})x_{uk}x_{hj}(1-x_{hk})\}$$

where, without loss of generality, we assume that $(x_{nk}, x_{nj}) = (1, 1)$. This statistic, while straightforward, can become computationally cumbersome since we need to evaluate whether we observe the phenomena on all pairs of offspring. On the other hand, using sufficient statistics, without loss of generality, if the offspring h gains function k, then the change statistic becomes:

$$\delta(h,k)_{\text{subfun } kj} = \begin{cases} \sum_{u\in\mathbf{O}(n)\setminus h}(1-x_{uj})x_{uk}, & \text{if } x_{hj} = 0 \\ -\sum_{u\in\mathbf{O}(n)\setminus h} x_{uj}(1-x_{uk}), & \text{otherwise} \end{cases}$$

which involves less calculation.

Neofunctionalization Another key statistic, neofunctionalization happens when one copy retains the ancestral function, leaving the other copy free to evolve a new function. The neo-

functionalized copy can be modeled as losing one or more ancestral functions, and gaining one or more new functions. Like in the case of subfunctionalization, we could first approach neofunctionalization by looking at pairs of functions. Without loss of generality, assuming that $(x_{nk}, x_{nj}) = (1, 0)$, the statistic can be computed as follows:

$$s\left(\mathbf{x}, x_n\right)_{\text{neofun } kj} = \sum_{u<h:\{u,h\}\in\mathbf{O}(n)} \{x_{uj}(1-x_{uk})(1-x_{hj})x_{hk} + (1-x_{uj})x_{uk}x_{hj}(1-x_{hk})\}$$

Finally, the corresponding change statistic can be calculated as:

$$\delta(h,k)_{\text{neofun } kj} = \begin{cases} \sum_{u\in\mathbf{O}(n)\setminus h}(1-x_{uj})x_{uk}, & \text{if } x_{hj}=0 \\ -\sum_{u\in\mathbf{O}(n)\setminus h} x_{uj}(1-x_{uk}), & \text{otherwise} \end{cases}$$

which differs from the subfunctionalization equations only on the state of the parent node–the subfunctionalization statistic requires the parent to have both functions, while the newfunctionalization statistic requires the parent to have only one of the two functions.

This approach, using change statistics to calculate the support of the sufficient statistics, can be further optimized by leveraging the fact that support sets will be essentially shared across nodes in most models, in particular, we can save more computational time by keeping a hash table, i.e. a lookup storage system, which we can use to keep track of shared support sets. For example, if in our model: (a) Data included 500 internal nodes, (b) all internal nodes had the same number of offspring, e.g. two, and (c) we were not considering branch lengths; instead of doing full enumerations 500 **times**, it would suffice to computing it **only once**, as all 500 events' would share the same support space. This and other related ideas are currently being implemented on the *barray* C++ template library described in section 7.6.

Chapter 5

Exponential Random Graph Models for Small Networks

This chapter, currently in the second round of revisions in the journal *Social Networks*, is the result of collaboration with Andrew Slaughter and Kayla de la Haye. The latest version of the manuscript can be found as a pre-print on arXiv https://arxiv.org/abs/1904.10406.

5.1 Introduction

Statistical models for social networks have enabled researchers to study complex social phenomena that give rise to observed patterns of relationships among social actors, and to gain a rich understanding of the *interdependent* nature of social ties and social actors [74, 105]. For example, this research has provided new insights into the role that the attributes of social actors (e.g., their characteristics, beliefs, and decisions), and endogenous structural processes (e.g., social balance, and relationship reciprocity) play in shaping social networks across different populations and social settings, and how these social networks, in turn, influence individuals and groups.

Much of this research has focused on social networks within medium to large social groups: networks ranging from dozens or hundreds of members (e.g., classrooms and organizations)

to millions (e.g., online social networks). However, modern advances in statistical models for social networks have rarely been applied to the study of small networks, despite small network data from teams, families, and personal (ego-centric) networks being common in many fields that study social phenomena [8, 17, 23, 54]. The study of small networks often uses descriptive statistics that summarize basic structural features of the network; for example, the density, degree distribution, or triad count. However, researchers in these fields are often interested in testing hypotheses about *why* localized social structures, such as reciprocity, balance, and homophily, emerge in these small groups. A key limitation to such work has been the availability of statistical models for networks that can flexibly test and control for the kind of dependencies inherent to network data. In this paper, we propose an approach for applying one of the most widely used statistical models for social networks–exponential random graph models, or ERGMs–to small graphs, to enable new research on "little networks".

5.2 Exponential-Family Random Graph Models

Exponential-family random graph models (ERGMs) are one of the most popular tools used by social scientists to understand social networks and test hypotheses about these networks [39, 55, 93, 104, 121, and others]. In this family of models, an observed graph y , comprised of a set of nodes (vertices) and ties (edges), is characterized by a set of sufficient statistics defined on the graph, $s(y)$, and parameters ω. In a model that also includes node characteristics X, this leads to the following equation:

$$\mathbb{P}(Y = y \mid \omega, X) = \frac{\exp\{\omega^{\mathbf{t}} s(y, X)\}}{\kappa(\omega, X)}, \quad \forall y \in \mathcal{Y} \tag{5.1}$$

Where $\kappa(\omega, X) = \sum_{y \in \mathcal{Y}} \exp\{\theta^{\mathbf{t}} s(y, X)\}$ is the normalizing constant, and $\mathcal{Y}$ is the support of the model that is usually assumed to include all graphs of the same type (e.g., directed or undirected) and size, that do not include self-ties. In the directed graph case, the size of $\mathcal{Y}$ equals $2^{n(n-1)}$ possible graphs. This makes the exact calculation of $\kappa(\omega, X)$, and therefore of

(5.1), computationally expensive. A sophisticated array of parameters can be specified for ERGMs that reflect social and structural process of interest to social scientists, such as social closure, connectivity, and other affiliation preferences. Figure 5.1 shows some examples of the structures (statistics) that can be estimated with ERGMs.

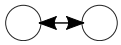
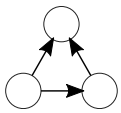
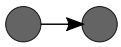
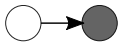
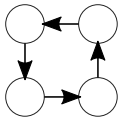
Representation	Description
	Mutual Ties (Reciprocity) $\sum_{i \neq j} y_{ij} y_{ji}$
	Transitive Triad (Balance) $\sum_{i \neq j \neq k} y_{ij} y_{jk} y_{ik}$
	Homophily $\sum_{i \neq j} y_{ij} \mathbf{1}(x_i = x_j)$
	Attribute-receiver effect $\sum_{i \neq j} y_{ij} x_j$
	Four Cycle $\sum_{i \neq j \neq k \neq l} y_{ij} y_{jk} y_{kl} y_{li}$

Figure 5.1: Besides of the common edge count statistic (number of ties in a graph), ERGMs allow measuring other more complex structures that can be captured as sufficient statistics.

While other methods for studying small graphs exist, e.g. non-parametric tests like the Conditionally Uniform Graph tests (CUG tests in the *social networks* literature [2, 35] and rewiring algorithms in the *network science* literature [80, 81]), all in all, ERGMs have more flexibility because they can be used to test complex hypotheses in a multivariate framework. As noted in [12], most of these non-parametric methods can be written in the form of (5.1), which means that ERGMs can be viewed as a generalized version of many of these tests.

Although "small networks" is a topic mentioned several times in the literature on social network models [39, 105, 121], interest in larger social networks has dominated the field.[1] Thus, ERGM methods have been developed to accommodate larger networks (although it is only very recent developments that have begun to scale well to "very large" networks of several thousand nodes or more [108]). One example of this is the calculation of the likelihood function: rather than

[1] This is perhaps because, as put by [105], small networks are considered to be "uninteresting special cases"

being calculated using exhaustive enumeration (which we will refer to as "exact likelihood"), the most popular software packages used for estimating these models apply simulation-based estimation methods. As a consequence, current methods used to estimate ERGMs for medium to large networks do not translate well to small network data (i.e., 6 or fewer nodes in a directed network), and applications of these statistical network models to small networks are rare.

One major technical and theoretical issue in ERGM estimation generally, which is exacerbated with small networks, is the problem of *non-existence of Maximum Likelihood Estimation (MLE)*. Non-existence of MLEs (or the convex-hull problem) occurs when the observed graph's statistics lie in a region on or near the boundary of the support [4], and can be stressed when estimation depends on Monte Carlo Integration [50]. Small networks, which are more likely to be nearly empty or nearly full, have a smaller region of support, and are more likely to be on or near the boundary of that support. For example, if we are trying to estimate an ERGM in a network with only three nodes, in the scenario where the graph is directed and does not allow for self-ties, the chances of obtaining a graph with either one or zero ties (i.e., empty or almost completely empty), or a graph with five or six ties (i.e., fully or almost fully connected) is about 20% using a uniform sampler.[2]

Because researchers studying small networks often have observed *samples* of small networks (e.g., multiple team, family, or personal/egocentric networks), a common work-around to the issue of non-existence of MLE is to combine the independent small networks into a single larger block-diagonal graph. Estimation then proceeds by assuming that ties between blocks are impossible (i.e., treated as structural zeros in estimation). The major problems with this approach are that it can be complicated to fit, and difficult to extend. As an example of the former, the same set of constraints (the structural zeros) that allow for the model to be fit can also make the estimation procedure more difficult, and increase the possibility of sampling problems during MCMC estimation. However, a more important challenge with the block-diagonal approach are difficulties with extension. A basic "complete pooling" model, which assumes a common data

[2]For more on the discussion on existence, degeneracy, and instability see [64, 92, 97].

generating process across all networks, is straightforward to define. However, relaxing that assumption to allow for variability across graphs (i.e., unpooled or partially-pooled models) can be problematic; it would typically require the creation of block-wise node membership attributes, and complex interaction terms involving subgraph statistics and node membership variables. Moreover, extending this framework to not only allow for between-group variability, but to explicitly *predict* it (for example, as a function of additional group-level variables), is not straightforward with this complete-pooling approach.

To overcome the challenges described above for fitting ERGMs to small networks, we leverage the fact that in the case of small networks, the full likelihood function *is* tractable. This allows the direct estimation of model parameters without using Markov Chain Monte Carlo (MCMC) or other approximate methods, avoiding some of the convergence issues associated with the convex-hull problem [50]. It also makes it much easier to combine ERGMs with other statistical techniques, opening the door for many possibilities of richer methods to model and understand small-group network structure and dynamics. In this paper, we describe how modern computational power allows for the complete specification of the likelihood for small graphs, and how this specification allows us to use the standard tools of MLE, instead of approximate methods. We present examples using these techniques; provide some initial results on empirical bias, type I error rates, and power based on a simulation study; illustrate the flexibility of this method with an empirical application; and discuss future extensions these techniques make feasible.

5.3 ERGM*itos*: ERGMs for Small Networks

With modern computers, calculating the exact likelihood function of an ERGM for a small network becomes computationally feasible. This has an important implication: the process for estimating the parameters of an ERGM for small networks can be done directly. Many innovative techniques have been developed to handle models with intractable normalizing constants (e.g., Markov Chain Monte Carlo [MCMC] based estimation methods, Bayesian techniques such as the exchange

sampler, etc.), and often these techniques work quite well. Of course, no techniques are without tradeoffs; MCMC-based estimation can be sensitive to starting values, and the quality of standard errors can depend on the availability of analytic gradients [88]. Bayesian techniques like the exchange sampler [82] may be comparatively slow, which may be an issue when many networks are to be analyzed.

Moreover, simulation-based methods may have particular susceptibilities to the convex-hull problem. As stated by [50, p. 7], "[i]f the model used to simulate the graphs is not close enough to produce realizations that cover the observed values of the statistics, the MC-MLE will not exist even in cases where the MLE does." For example, many common network models, such as triangle-based models, can lead to bimodal distributions of graph statistics that simulation-based methods have difficulty with; even when the MLE falls between the modes [60]. Therefore, even though the non-existence issue is not completely avoided, a method based on exact (non-simulation) inference may not only provide a better solution (in general) by avoiding the additional uncertainty induced by simulations and approximations, but it may also help to mitigate the problem in cases where the MLE exists.

Of course, the statistical analysis of a single small network could be uninformative due to the small numbers of dyads, and a high restriction in the variability of possible subgraph statistics. Fortunately, research on small networks typically involves collecting data from *samples* of small groups (vs. the more typical 'case studies' of single larger networks), which allows for the development of models to analyze structural variation both within and across small networks. If we assume that the sample of networks comes from a population of networks (groups) that are governed by the same data generating process, we end up with the following likelihood, defining a completely-pooled model:

$$\mathbb{P}\left(Y_1 = y_1, \ldots, Y_P = y_P \mid \omega, X_1, \ldots, X_p\right) = \prod_{p=1}^{P} \frac{\exp\left\{\omega^{\mathbf{t}} s\left(y_p, X_p\right)\right\}}{\kappa_p\left(\omega, X_p\right)} \tag{5.2}$$

Where P denotes the number of networks used in the model, and $\kappa_p\left(\omega, X_p\right)$ is explicitly calculated, unlike existing approaches to ERGM estimation. We call this framework, which is a

revisited version of ERGM in the case of small networks, ERGM*ito*. In general, this extension can be feasibly applied to small graphs containing at most 6 nodes if directed, or 8 if undirected.

Not to be confused with *pooled estimators* – i.e. aggregating various parameter estimates from independent model fits– pooled-data models have several benefits, including the ability to consider small networks that otherwise would be excluded from an analysis; e.g., because they are fully connected or empty graphs. Moreover, as we will emphasize later in section 5.5, as long as at least one network in the sample has values on the boundary for each type of sufficient statistic, the MLEs will generally exist [see 50].

One issue that may be of concern is the feasibility of the underlying assumptions when estimating pooled-data models with networks of different sizes. Because parameter estimates often encode network size, one may argue that pooling networks of different sizes into a single model may not be appropriate. However, there are several ways to control for size-induced heterogeneity; for example, including fixed or random effects at the graph level to account for size, or using approaches such as those described in [13, 68, 69]. In the cases presented in this paper, we focus on samples of networks that are of similar sizes (networks of size 4 and 5); thus, these issues are unlikely to be of great concern within a small range of values, although we demonstrate how they can be accounted for in our applied example (section 5.6).

In the following sections we illustrate and investigate the properties of estimating ERGMs for small networks using this approach. All simulations and model fitting were conducted using the R package *ergmito*, which has been developed to implement the methods described in this paper.

5.4 Illustration With Simulated Data: fivenets

5.4.1 Data-generating-process and Model Fitting

Starting with a s simple example, we now look at a simulated data set that was created using the data-generating-process of ERGM*itos*. This particular dataset, which we call "fivenets", is

included in the in the R package *ergmito*[3]. The data set contains five small graphs with nodal attributes (we use gender in the following example), with the networks generated using the following specification:

$$\mathbb{P}\left(Y = y \mid X, \omega\right) = \frac{\exp\left\{\omega_{edges}\left(\sum_{i,j} y_{ij}\right) + \omega_{same}\left(\sum_{i,j} y_{ij}\mathbf{1}\left(X_i = X_j\right)\right)\right\}}{\kappa\left(\omega, X\right)}$$

where $\omega_{edges} = -2.0$ and $\omega_{same} = 2.0$. Using this equation we draw five networks of size four. The process of "homophily" is represented by a parameter that is defined as the number of ties in which ego and alter have the same gender, ω_{same}. Before drawing the networks we randomly generated the node attribute (gender) to each vertex as a Bernoulli with parameter 0.5. Figure 5.2 shows the generated networks, including their nodal attributes.

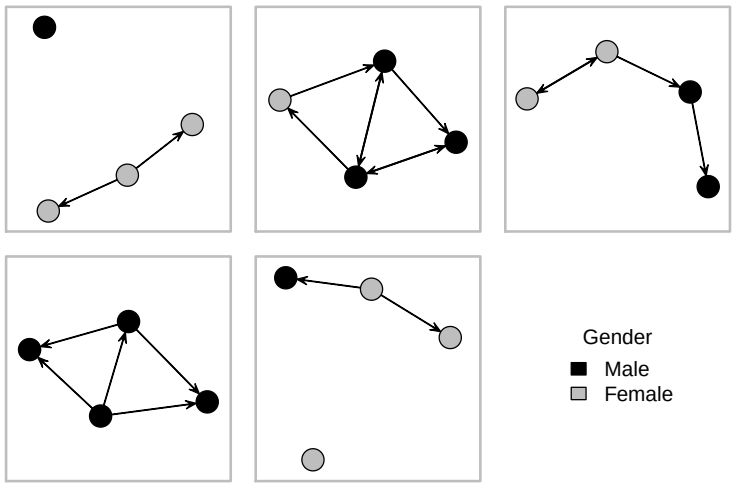


Figure 5.2: Fivenets data set. These graphs were randomly drawn from an ERGM distribution with two parameters: number of edges and gender homophily, with parameters equal to -2.0 and 2.0 respectively.

Using the *ergmito* R package, we fit three different models to the data: (1) a Bernoulli graph, which is a model that only includes the "edges" parameter, (2) a model with "gender homophily" as its only parameter, and finally (3) a model including both "edges" and "gender homophily", which is the correct specification of the model. Some details regarding the computational aspects

[3]The R package is available to be downloaded at https://github.com/muriteams/ergmito and https://cran.r-project.org/package=ergmito.

of the model fitting process are provided in C.1.

In general, while practitioners are accustomed to dealing with a single set of observed sufficient statistics, sometimes called "target" statistics, pooled models instead feature an array of such statistics. Table 5.1 displays the counts used in this model, from the Fivenets data.

Net id	edgecount	count of gender homophilic ties
1	2	2
2	7	5
3	4	3
4	5	5
5	2	1

Table 5.1: Observed sufficient for the *fivenets* dataset. In the case of pooled-data models, there is no one set of observed (target) sufficient statistics, but an array of such statistics. This table shows the *edgecount* and the *count of gender homophilic ties* in the *fivenets* dataset.

Table 5.2 shows the estimation results of the three different specifications of the model and, as expected, model (3) has the best overall fit to the data. Furthermore, since all three models were fitted using MLE, we can compare the edgecount and homophily models with the full model using Likelihood Ratio tests [127].

	Homopholy	Edgecount	Full model
Edgecount		-0.69^{*}	-1.70^{**}
		(0.27)	(0.54)
Homophily (on Gender)	-0.12		1.59^{*}
	(0.34)		(0.64)
LR-test statistic (χ^2)	7.04^{**}	13.72^{***}	
AIC	85.06	78.38	73.34
BIC	87.15	80.48	77.53
Log Likelihood	-41.53	-38.19	-34.67
Num. networks	5	5	5

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 5.2: Fitted ERGMitos using the fivenets dataset. Looking at AICs and LR-test statistics, the full model (last column of the table) is the one with the best fit to the observed data. Moreover, the 95% level CI of each covers the true parameters: $\hat{\theta}_{edges} \in [-2.77, -0.64]$; $\hat{\theta}_{Homophily} \in [0.33, 2.85]$.

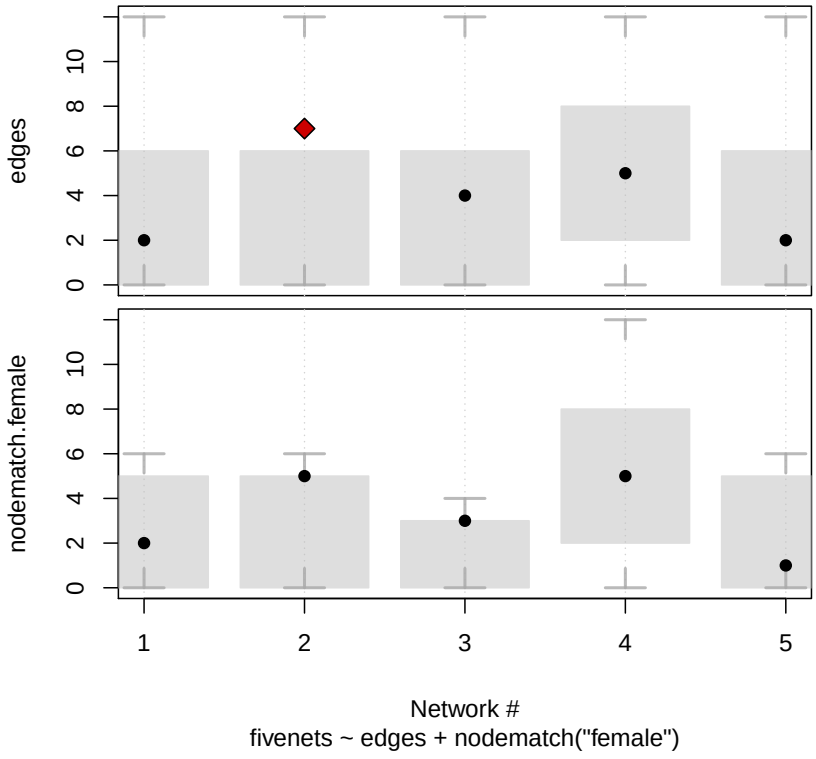
It is important to note that the *ergm* package can also be used to calculate exact likelihoods, and that this feature has been available for a long time. Some of the additional features and extensions provided in the *ergmito* package, which are illustrated in subsequent sections of the paper, are: a simple way of estimating pooled-data models, simulating small networks using exact likelihoods, evaluating goodness-of-fit at the graph level for pooled-data models, and including arbitrary effects like interaction effects and transformation of the canonical ERGM terms. The *goodness of fit* of this model is evaluated in the following section.

5.4.2 Goodness-of-fit in ERGM*itos*

Researchers that apply ERGMs should be familiar with the graphical goodness-of-fit (GOF) diagnostics that are used to assess how well the estimated model can reproduce graphs that are similar to the observed graph on a range of local and global graph statistics [58]. In the case of ERGM*itos* applied to small networks, local graph statistics will be more relevant than global statistics to assess GOF. For example, the graph geodesic distribution (i.e., the distribution of shortest-path lengths) is often used to assess GOF for larger networks, but this is clearly less relevant in the case of small networks (like in our case, containing at most 6 nodes if directed, or 8 if undirected) because the shortest-path length between any two nodes typically lies between one and three steps. Therefore, we focus the GOF analysis on the parameters fit in the model as the minimum set of local graph statistics, as shown in Figure 5.3; and depending on the model complexity a more comprehensive set of local statistics may be needed. An important difference in our approach compared to traditional GOF assessments for ERGMs is that we are able to enumerate the full support of the model, and so instead of showing a boxplot we present a 90% exact confidence interval per-statistic per network, comparing the fitted model's distribution with the observed parameters.

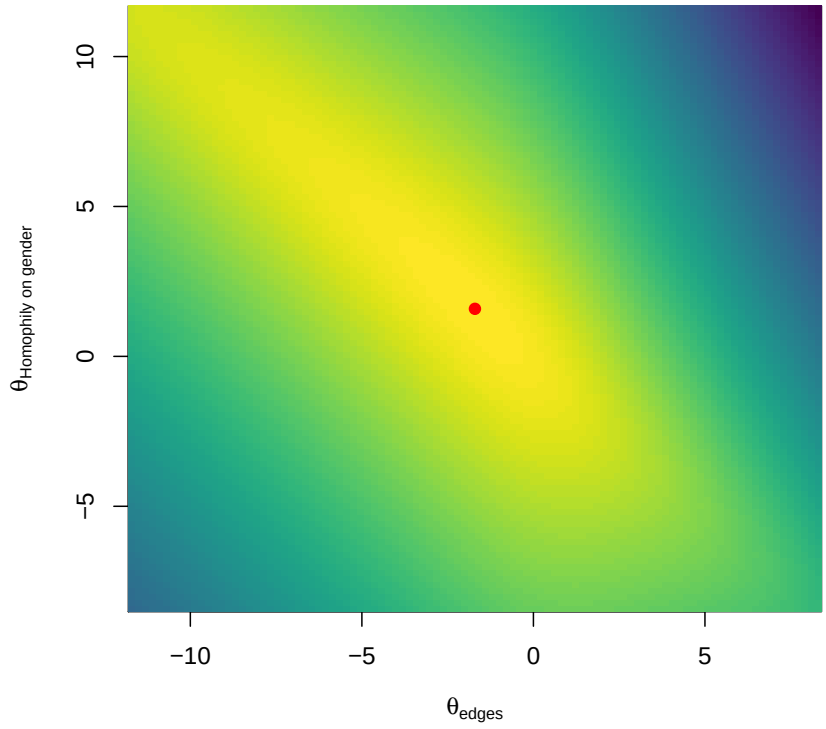
An important advantage of the ERGM*itos* over "regular" ERGMs is that we can observe the surface of the log-likelihood over different combinations of parameters in a rather straightforward way. This, together with the GOF analysis should be a routine step done after every ERGM*ito*

Figure 5.3: Goodness-of-fit in ERGM*itos*. This illustrates how the observed sufficient statistics of each one of the 5 networks (x-axis) locate in the overall estimated distribution based on the fitted ERGM*ito*. The gray lines in each box show the minimum and maximum value that the sufficient statistics can take in each one of the 5 networks, whereas the dotted lines provide a 90% confidence interval. The dots are the observed statistics in each network.



fit. Figure 5.4 shows the surface of the log-likelihood function around the solution parameters to the maximization problem.

Figure 5.4: Surface of the log-likelihood function of the pooled ERGM*ito* model. Lighter colors represent higher values while darker ones represent lower values. The red dot corresponds to the location of the MLE estimate of the model.



The ability to calculate the surface of the exact likelihood function provides additional tools for assessing the quality of the estimated set of parameters. One good use of this diagnostic is to evaluate the roughness of the log-likelihood function, which in principle should give us an idea of the likelihood of the maximization process failing to reach a global maxima, or estimates being close to problematic (e.g., generating empty or fully connected graphs) areas of the parameter space.

5.5 Simulation Study

We conducted two sets of simulations where we compare the performance of the Maximum Likelihood Estimator [MLE] with that of the Monte Carlo MLE [MC-MLE] and Robbins-Monro Stochastic Approximation [RM] in terms of bias, power, type I error rates, and overall computation time. In the first set of simulations, we analyze empirical bias, empirical power, and overall computation time of each estimator in a scenario where the ERGM is defined by *edgecounts* and *transitive triads*. For the second set of simulations, we look at empirical type I error rates when ERGMs are mis-specified by including a *transitive triad* term in the context of a data-generating-process that only includes an *edgecount* statistic.

The code used to reproduce this entire section can be found at https://github.com/muriteams/ergmito-simulations.

5.5.1 Empirical Bias and Power

Using the ERGM*ito* R package, we generated 20,000 samples (datasets), with each sample consisting of several small networks defined by the parameters $edges$ (edgecount) and $ttriads$ (number of transitive triads). Each sample was generated using different combinations of parameters. While all come from an ERGM model defined by edgecounts and number of transitive triads, for every sample we specified: (1) population parameters for the ERGM, (2) the size of the sample (i.e., the number of networks in the sample), and (3) the composition of the sample in terms of the combination of networks of size four and five. A detailed description of each one of these three components used to draw the samples follows:

1. **Population parameters**: First we drew two numbers from a piece-wise Uniform distribution with values in $[-2, -.1] \cup [.1, 2]$, $(\omega_{edges}, \omega_{ttriads})$, which corresponded to the parameters associated to the statistics **edgecount** and **number of transitive triads**. This specifies the ERGM from which we will draw the networks from. This is akin to the approach taken by [98] , although we took a more conservative approach than their ranges

of (-5,0) and (0, 5) for the parameters "edges" and "triangles" in order to increase the number of *irrelevant* draws (i.e., samples composed mostly of either empty of fully connected graphs, or networks with no transitive triads).

2. **Number of networks per sample** Then, we specified the number of networks to generate from the models defined in the previous step, using one of the following sample sizes $\{5, 10, 30, 50, 100, 150, 200, 300\}$. The 20,000 simulations were equally split across the various sample sizes (i.e., the simulation study was based on 2,500 samples comprised of 5 networks; 2,500 samples comprised of 10 networks, etc.)

3. **Number of nodes per network** Finally, the composition of each sample, in terms of the number of nodes that each network has, was uniformly-random selected from the pairs $\{N, 0\}, \{N-1, 1\}, \ldots, \{1, N-1\}, \{0, N\}$, where the first number of each pair is the number of networks of size 4, and the second is the number of networks of size 5 in the sample. As an example, if the sample size selected in the previous step was 30, then the possible pairs to select from would be $\{30, 0\}, \{29, 1\}, \ldots, \{1, 29\}, \{0, 30\}$, so that samples in which all networks were of size 4 (meaning we draw the pair $\{30, 0\}$) or size 5 (again, selecting the pair $\{0, 30\}$) were equally likely.

For each one of the 20,000 simulated datasets, we then estimated the model using MLE, as implemented in the *ergmito* R package, and MC-MLE and RM, as implemented in statnet's *ergm* R package [51, 61]. In the case of the latter two, the pooled estimation was done by fitting what is known in the literature as a block-diagonal model in which (a) networks are stacked together in a single adjacency matrix, and (b) the sampling space for the MCMC process is constrained to sample from graphs where ties are only possible within blocks. In the case of the MCMC estimator, we set the control parameters *interval* and *samplesize* to 2,048, with a burn-in of 2,048 x 16 = 32,768; all double the of the current default values specified in the *ergm* package, so that we could increase the precision of our estimates. And in cases where the algorithm failed to return any estimates, we increased the control parameters *interval* and *samplesize* to 10,000.

Analysis preface

After simulating the data and estimating the models, we found that there were several cases in which the programs implementing the three algorithms did not converge, and either returned estimates with a warning to the user, or failed without returning a meaningful message to the user. First, the MLE implementation in *ergmito* had zero failures, meaning that, even if the optimization failed to converge, the program provided the user with a meaningful report in all cases. Second, while the MC-MLE implementation of the *ergm* package did fail without returning *any* form of results in some cases (97 of the 20,000), in each of these instances the program provided the user with a meaningful report of what caused the error. Third, in the case of the Robbins-Monro algorithm [RM], as implemented in the *ergm* package, we observed a high error rate: in about 25% of the samples, the *ergm* function failed during the estimation process, and returned an uninformative error message to the user ("*NA/NaN/Inf in foreign function call (arg 13)*"). This error rate should be interpreted with some context; the implementation of the RM algorithm has received less attention, and thus less optimization, that the MC-MLE method. While the *PNet* [119] software provides a more mature implementation of the RM algorithm, we chose to use *statnet*'s implementation as it was better suited for the implementation of our simulation study. Table 5.3 shows the number of errors as a function of sample size (number of networks) for each estimation method.

Nearly all of the errors (cases in which the software failed and returned with an error) observed in RM, all but three occur on realizations of the data-generating-process that yielded uninteresting cases, where either of the observed sufficient statistics was on the boundary of their support, e.g. fully connected graphs or graphs with no triads.

With respect to those cases in which the algorithm failed to converge (which includes both software errors and the program reporting lack of convergence), Figure 5.5 shows the distribution of the sufficient statistic split based on whether the algorithm converged or failed to do so. As shown in the figure, when the algorithms did not converge it was typically due to sufficient statistics falling on the boundary of its support (convex-hull problem). This was especially true for

	# of errors		
Sample size	MLE	MC-MLE	RM
5	0	44	1,274
10	0	21	1,058
30	0	10	760
50	0	3	668
100	0	6	583
150	0	3	507
200	0	4	508
300	0	6	460
Total	0	97	5,818

Table 5.3: Number of times the program failed to fit a model and returned with an error. This shows the overall error rate over the full set of 20,000 simulated samples. All but 3 errors of the RM implementation happened on cases where the sufficient statistics were on the boundary.

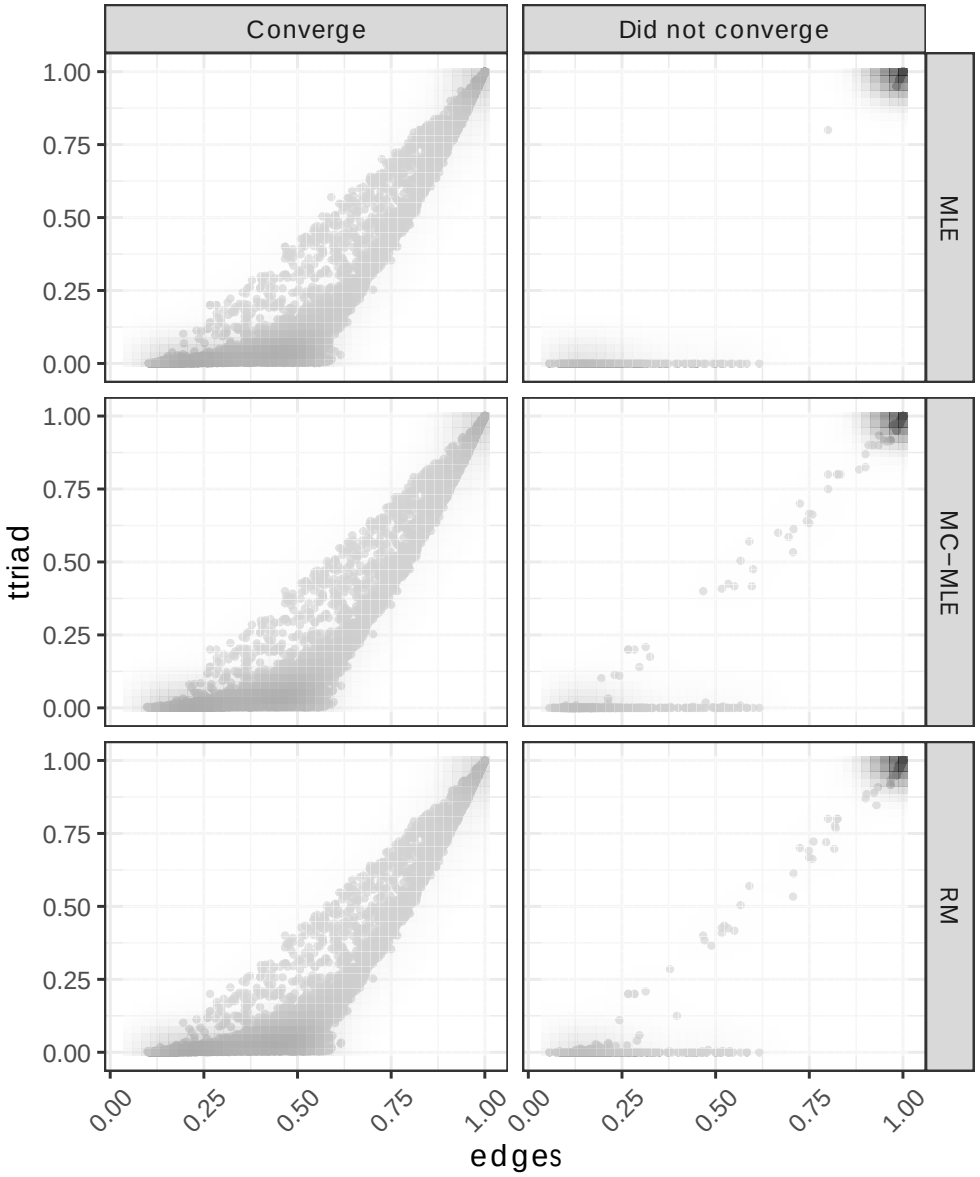
the case of the MLE implementation of *ergmito*, as all but one of the convergence failures were on the boundary. While MLEs can be obtained in some of those cases (see appendix C.2 and [50]), in general, estimating such models has no practical utility. We therefore focused our analysis on samples of networks for which the aforementioned model is appropriate: all subsequent analyses include only those data sets where the observed sufficient statistics, *edgecounts* and *number of transitive triads*, were not on the boundary of its support for *at least* one network in the sample. In other words, we *included* the sample if it: (a) had at least one graph that was not fully connected, and (b) had at least one transitive triad in at least one network. Of the 20,000 simulated data sets, 14,185 met the criteria.

Overall, practitioners should bear in mind that the cause of errors that arise during the estimation process can be based on the method or software, and when this is captured by the program it can be informative to both users and developers.

Empirical Bias and Power

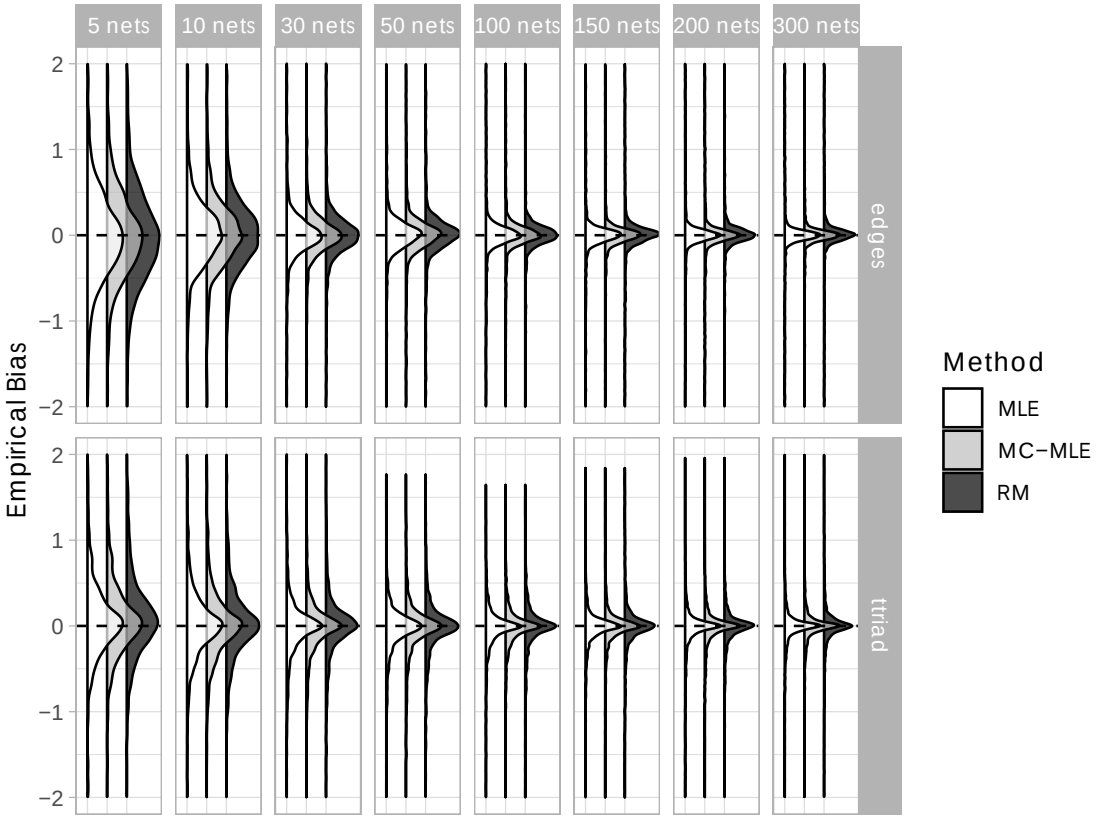
As shown in Figure 5.6, all three estimation methods behaved similarly in terms of empirical bias in the models studied here. As the size of the sample of networks in the dataset increased

Figure 5.5: Distribution of the average sufficient statistics per sample. Since samples can contain networks of sizes four and five, we have re-scaled the sufficient statistics counts by each network size's corresponding maximum value so these range from zero to one. Most of the cases in which methods failed to converge happened in scenarios where either all the graphs in the sample were fully connected or there was no transitive triad; exactly the cases that we excluded for the reminder of the analysis.



(i.e., when there were more networks within the sample), the empirical bias of all three, MLE, MC-MLE and RM, decreased, as expected.

Figure 5.6: Empirical distribution of the bias per model parameter, for MC-MLE and MLE estimation methods. In general we see that the parameter estimates' bias is centered around zero and both MC-MLE (ERGM) and MLE (ERGMito) have about the same bias in our simulation study.



Looking closer at the biases, we noticed that, while all methods show some kind of bias, MLE has (on average) the smallest. As showed in Table 5.4, at the 95% confidence level, all three methods tend to overestimate the *edges* parameter. On the other hand, with the exception of the RM method, both MLE and MC-MLE tend to underestimate the *transitive triads* parameter; yet, the RM method has the widest confidence interval for that parameter.

Empirical power levels, calculated as the proportion of times that the method reported a significant effect at the 5% level in the same direction as the data-generating-process parameter, is depicted in Figure 5.7. For each method, a single bar in the figure shows the empirical power

	MLE	MC-MLE	RM
edges	[0.27, 0.36]	[1.23, 1.65]	[0.55, 1.54]
ttriads	[-0.05, -0.03]	[-0.22, -0.16]	[-0.15, 0.48]

Table 5.4: Empirical bias. Each cell shows the 95% confidence interval of each methods' empirical bias.

level for the corresponding combination of sample size (x-axis), parameter (columns), and effect size (rows). There are three main findings to highlight: first, as expected, power increases as both sample size and effect size increase; second, both MLE and MC-MLE behave very similarly with no statistically significant differences across sample and effect size; and third, compared to MLE, RM had a statistically significant smaller power level at various sample and effect sizes combinations, with the largest differences observed on transitive triads Although there may be some inherent properties of each method that may benefit MLEs, this again may be due to less emphasis on the implementation of RM in the *ergm* package. Finally, as an anecdotal observation, it is interesting to see that, in the case of effect sizes of magnitude [0.5, 1.0), the discovery rate for the *ttriads* parameter reaches nearly 0.75 for sample sizes between 30 to 50 networks, which is a rather common sample size in the study of small networks such as teams, families, and sometimes ego-networks.

Figure 5.8 shows the effect of the composition of the sample in each dataset, in terms of the proportion of networks of size five (vs. size four), and the number of networks per dataset. In this case, we observe no meaningful patterns that would indicate the dataset composition is related to power.

One remarkable difference between the three estimation methods featured by the simulations is the overall computing time needed to fit the models. While the computation of exact likelihoods and gradients is still very computationally intensive, the total time needed to obtain MLEs is still significantly less than what is needed to by the other two methods. As shown in Figure 5.9, MLE can be orders of magnitude faster than MC-MLE and RM. Therefore, while all three estimators show very similar properties in terms of power and bias, practitioners will benefit by using MLE

Figure 5.7: Empirical power by dataset size and effect size (the later considering only magnitude), for ERGM and ERGM*ito* estimation methods. Power increases for both MC-MLE (ERGM) and MLE (ERGM*ito*) with increases in the size of the dataset and effect size. There are indistinguishable differences in power between the two estimation methods.

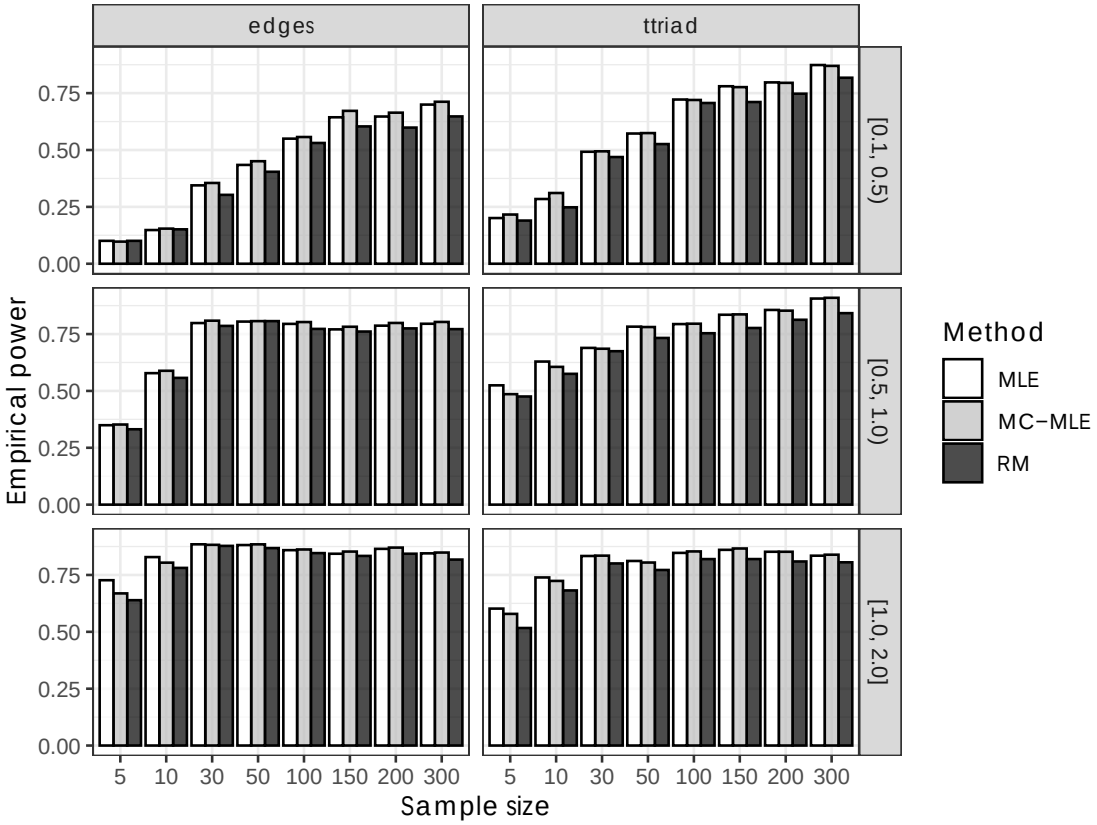
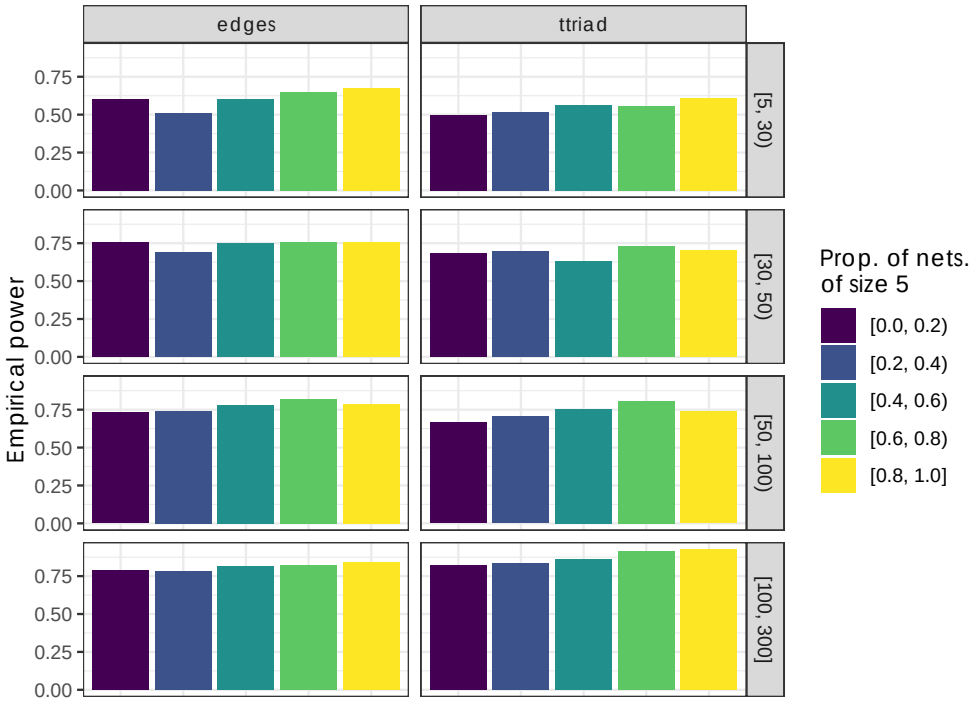


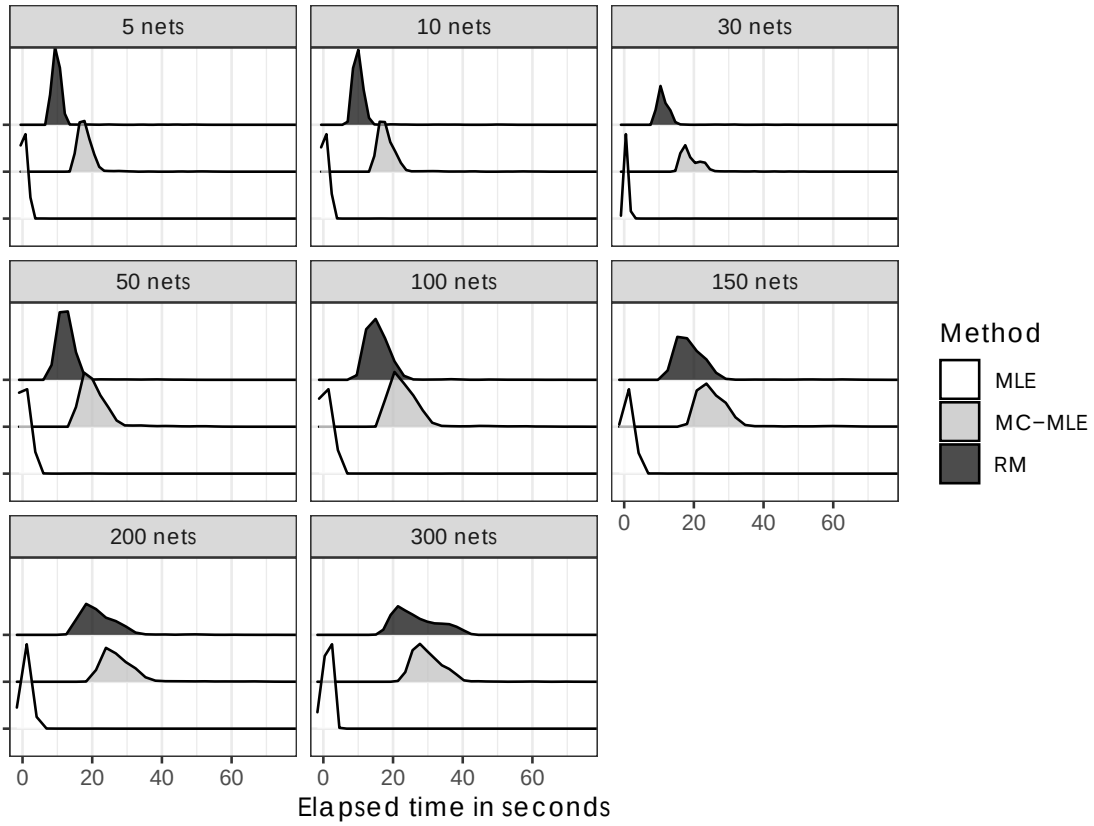
Figure 5.8: Empirical power by proportion of networks of size five in the sample (color coded) and sample size (rows).



when modeling small networks because it may substantially reduce computation time.

Nevertheless, while MLE is generally faster than the other two methods, there are some scenarios in which the speed gains may not be as dramatic as those shown here. The biggest computational bottleneck that the MLE estimation faces is the calculation of the full support of the sufficient statistics. In the case of structure-only statistics, *ergmito*, and actually *ergm*, computes the full distribution very quickly, but, as the model starts to become more complex, such calculation becomes more and more expensive. Yet, once the full enumeration of the support of sufficient statistics is done, finding MLEs becomes *trivial*, making the implementation of other statistical tools such as bootstrap or forward/backward model selection feasible to implement. Bootstrapping of ERGMs is illustrated in section 5.6.

Figure 5.9: Distribution of elapsed time (in seconds) for the estimation process for MC-MLE (ERGM) versus MLE (using ERGM*ito*). Overall, the MLE implementation is orders of magnitude faster compared to the time required by the MC-MLE implementation to do the parameter estimation.



5.5.2 Type I Error Rates

Using the same procedure described in subsection 5.5.1, we simulated 35,000 datasets comprised of Bernoulli networks (i.e., an ERGM model only defined by the *edgecounts* sufficient statistic). In this case, we drew different sets of sample sizes: for each of $\{5, 10, 15, 20, 30, 50, 100\}$ we generated 5,000 datasets using the Bernoulli model with *edgecount* parameter uniformly distributed in the range $[-2, -.1] \cup [.1, 2]$. We then estimated the models using MLE, MC-MLE, and RM and calculated the type I error rates using a misspecified model; that is, fitting ERGMs that included a *transitive triads* count statistic. As with the previous simulations, we only analyze datasets that either had at least one not fully connected graph and had at least one transitivite triad in at least one network. Fortunately, as Table 5.5 shows, most of the cases did.

Table 5.5 shows the type I error rates per sample size for each of the three methods. In general, MLE report lower error rates compared to MC-MLE and RM, when models were fit to datasets with sample sizes of 20 or fewer networks, the MLE had a better performance than MC-MLE as it reported smaller type I error rates that were much closer to the nominal 5% level. Datasets with 30 or more networks had no significantly different type I error rates between the two methods. Compared to RM, the simulation study shows MLE has a better performance when estimating pooled-data models with 10 or less networks. No significant difference is observed when dealing with samples of 15 or more networks.

Sample size	N. Sims.	P(Type I error)			χ^2 (vs MLE)	
		MLE	MC-MLE	RM	MC-MLE	RM
5	4,325	0.066	0.086	0.086	11.36 ***	11.36 ***
10	4,677	0.063	0.078	0.073	8.44 ***	3.73 *
15	4,818	0.060	0.072	0.063	5.50 **	0.41
20	4,889	0.054	0.065	0.061	5.30 **	2.05
30	4,946	0.053	0.059	0.055	1.60	0.07
50	4,987	0.053	0.055	0.047	0.16	1.67
100	4,999	0.054	0.054	0.050	0.00	0.81

Table 5.5: Empirical Type I error rates. The χ^2 statistic is from a 2-sample test for equality of proportions, and the significance levels are given by *** $p < 0.01$, ** $p < 0.05$, and * $p < 0.10$.

5.6 Extended Application: The Role of Gender-homophily on the Formation of Small Teams

In this final section, we apply the ERGM*itos* framework to a set of observed social networks in an experimental setting. The data was generated as part of a study that examined the emergence of social networks in small teams.

The analytic sample consists of 31 small mixed-gender teams that include either four (17 teams) or five members (14 teams). Individuals recruited for the study were University students, participating for research credit or compensation, who were assigned to the teams with two conditioning factors: (1) they did not know the other teammates, and (2) there must be at least one team member who identified as male, and one who identified as female. On average, 55% of each team's members were female, with no statistically significant difference between the teams (test of equal proportions) nor within the teams when compared to a null of 0.5 (exact binomial test). Each team met face-to-face in a laboratory setting to complete about one hour of group tasks. Immediately after the completion of the group tasks, the team networks were measured using name generators administered in an online survey (that was completed in the lab). *Advice seeking* was one relationship measured, via the question "*Who did you go to for advice, information, or help to complete the group tasks?*", and participants could select as many or as few teammates as they liked. These data were used to generate directed graphs that represent the advice-seeking network in each team, where $y_{ij} = 1$ if i identified j as someone they sought advice from.

One research question of interest in the field of team science is what is the role of gender and gender-based homophily (i.e., the preference for individuals to form social ties with teammates who match them on gender) in the formation of team networks. Using the *ergmito* R package to model the team advice networks and test hypotheses about gender and network dynamics, we illustrate how exact calculation of ERGM likelihoods can be leverage to go beyond traditional ERGM analysis. Overall, the analysis consists of two parts: (1) building a baseline model that only

includes structural features of the graph, and (2) using that model to test if gender-homophily is a prevalent feature of the data, while also controlling for other gender-based terms in a multivariate fashion.

In the structural-terms-only model, we fitted five different models based on the following terms:

- **Edge count** (edges): This accounts for the overall density of the graph and is usually compared to that of a constant term in regression analyses. This is calculated as $\sum_{ij} y_{ij}$.
- **Number of transitive triads** (ttriads): This statistic, also known as balanced triangles or transitive triples, captures the phenomenon of social clustering and balance; where "*the friend of my friend is my friend*". In this context it indicates that "*the advisor of my advisor is my advisor*". This term is calculated as follows: $\sum_i \sum_{j<k} y_{ij} y_{jk} y_{ik}$.

To illustrate the flexibility of estimating ERGMs with the *ergmito* R package, we generated three additional terms to be included in the models using the *edges* and *ttriads* terms. First, we included two interaction effects, one per term, with an indicator variable which equals to one if the corresponding network was of size five, and zero if it was size four. We also added an offset term as that proposed by [68] which has the nice property of being size-invariant; i.e., it preserves the mean degree as the network size increases. All of these additional terms allowed us to control for differences as a function of the network size. A valuable benefit of these additional statistics is that users can add interaction effects or variable transformations to the models; a feature that, currently, is not easily achieved in other available frameworks (see for example [52, 59]). Just like we showed earlier in Table 5.1, Table 5.6 shows an example of the target statistics used in the models for 6 of the 31 networks (i.e., the array of observed sufficient statistics). With these five statistics we estimated five different models, including a bootstrapped version of the one with the best overall fit. Table 5.7 shows the results.

The results, Table 5.7, indicate that transitive triads (*ttriads*) were more prevalent than expected by chance; which is common in positive affiliation and collaboration networks. Parameter

(1) Size (n)	(2) edges	(3) ttriads	(4) edges $\times$ $\mathbf{1}(n=5)$	(5) ttriads $\times$ $\mathbf{1}(n=5)$	(6) edges $\times$ $\log\{1/n\}$
4	10	14	0	0	-13.86
4	6	2	0	0	-8.32
4	4	0	0	0	-5.55
5	6	1	6	1	-9.66
5	8	8	8	8	-12.88
5	6	2	6	2	-9.66
			. . . 25 more rows. . .		

Table 5.6: Example of observed sufficient statistics for the team advice networks. Pooled-data ERGMs have multiple observed sufficient statistics (also known as target statistics). Furthermore, as shown here, we can manipulate common statistics as *edges* (2) and *ttriads* (3) to include, e.g. interaction effects (4) and (5), or more complex transformations, e.g. (6).

	(1)	(2)	(3)	(4)	(5)	(3b)
edges	-0.72^{***}	0.73^{***}	-0.53^{***}	-0.85^{***}	-0.56^{*}	-0.53^{***}
	(0.13)	(0.13)	(0.15)	(0.14)	(0.23)	(0.12)
ttriad	0.29^{***}	0.33^{***}	0.36^{***}	0.50^{***}	0.38^{***}	0.36^{***}
	(0.05)	(0.05)	(0.06)	(0.07)	(0.11)	(0.05)
edges $\times$ $\mathbf{1}(n=5)$			-0.53^{***}		-0.49	-0.53^{***}
			(0.12)		(0.28)	(0.12)
ttriad $\times$ $\mathbf{1}(n=5)$				-0.22^{***}	-0.02	
				(0.05)	(0.12)	
offset						
edges $\times$ log $\{(\}$ $1/n)$		Yes				
AIC	651.38	641.02	637.28	640.40	639.26	637.28
BIC	659.74	649.39	649.83	652.95	655.99	649.83
Log Likelihood	-323.69	-318.51	-315.64	-317.20	-315.63	-315.64
Num. networks	31	31	31	31	31	31
Time (seconds)	0.55	0.99	0.74	0.76	0.74	10.12
N replicates						1000
N Used replicates						1000

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 5.7: Structural models. Model (2) includes Krivitsky et al (2011) offset term. Besides of the common GOF statistics, the table includes the number of networks used, elapsed time to fit the model, and, in the case of Model (3b) which is a bootstrapped version of model (3), number of replicates fitted and included in the bootstrap variance estimate.

estimates for the *ttriads* term were also robust with significant and positive effects across the different model specifications. Second, we found that controlling for size of the network mattered. The results of models (3) and (4) show that allowing networks of size 5 to have different parameters associated with number of edges or transitive triads (with networks of size 4 as a reference), significantly improved model fit relative to model (1). Yet, as shown in model (5), these interaction effects were not jointly significant. Regarding model (2), which includes the offset edges $\times \log\{1/n\}$, we see that the *edges* parameter flips from negative 0.72, to positive 0.73, which should be interpreted in the context of this offset change. For example, in the case of the Bernoulli model, the probability of an individual tie for a network of size 4 would be $\text{logit}^{-1}(-\log\{4\} + 0.73) \approx \text{logit}^{-1}(-0.66) \approx 0.34$, i.e. less than 0.5 which is the expected value under the null.

Of the five models, model (3) had the best overall fit, the lowest AIC and BIC, and so it was retained as the structural baseline model for the subsequent analyses. To finalize this first stage of analysis, we calculated the standard errors of model (3) using bootstrap [see 124]; with the results reported in column (3b). This final model had no meaningful changes in standard errors compared to (3); although they were slightly smaller compared to MLEs in (3). Additionally, the elapsed time for this bootstrapping process was negligible: remarkably, we fit 1,000 ERGMs in about 10 seconds, which further highlights how speed and model specification-flexibility are key features of fitting ERGMs using Maximum Likelihood.

The second phase of model specification, which uses model (3) as baseline, focused on evaluating the role of gender and gender-homophily in the advice networks, using the following terms:

- **Gender homophily**: This term equals to the number of ties in which ego and alter are matched on gender. This was calculated as: $\sum_{ij} y_{ij} \mathbf{1}(X_i = X_j)$, where X_i is one if i is a female, and zero otherwise.

- **Female-sender effect**: This term, also known as attribute-activity effect, captures the propensity of females to send ties. Is is calculated as: $\sum_{ij} y_{ij} X_i$.

- **Female-receiver effect**: This term captures the propensity of females to receive ties. It is calculated as: $\sum_{ij} y_{ij} X_j$.

Ttaking advantage of the flexibility that the *ergmito* package, and ultimately, using exact likelihoods provides, we also explored modifying the model by means of transformations and offset terms. First, with the purpose of improving the predictive capability of our model, we included the square root of the count of gender-homophilic ties. Other transformations such as interactions with other terms, or centering around a given constant (for example, some population average) could also be implemented. Second, while not the case in our data, we illustrate a hypothetical scenario where the teams had to have at least 5 ties, and we constrained the support of the sufficient statistics to only include networks with five or more ties. We did this by using an offset parameter that equaled $-\infty$ if the network had four or less ties, and zero otherwise. Figure 5.10 illustrates the differences between the Cumulative Distribution Function (CDF) associated with *edges* statistic (the probability of observing up to given number of ties, x-axis) calculated from a model with (red line) and without (blue line) the constrained space.

Using offset terms to constraint the support of the model is not a new thing. The *ergm* package features this capability as well, in addition to specialized algorithms to constrain samplings space. Users can also set offsets to $-\infty$ to forbid some configurations, yet, in the case of *ergmito* combining offset terms with the capability of mixing-transforming variables in the model provides the user with greater flexibility. As we did before, an example of six of the 31 networks is shown in Table 5.8.

Like in the first round of ERGMs, the standard errors of the final *best model* were re-calculated using bootstrap. Table 5.9 shows the results.

As illustrated in Table 5.9, in our first three specifications we found no evidence that gender-homophily was a prevalent feature of the advice networks, as the baseline (1), its constrained version (2), and the baseline including a transformed version of gender-homophily (3) failed to reject the null $\theta_{\text{Homophily}} = 0$. Of the other gender-based effects, only the female-sender effect, model (4), was significant. With a coefficient equal to 0.46, the model indicates that,

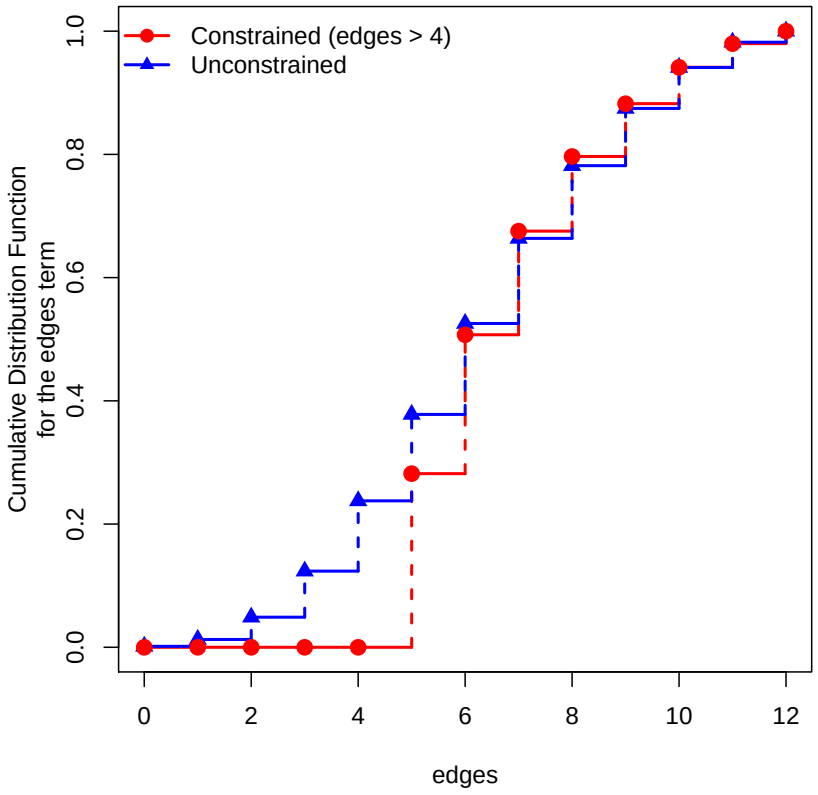


Figure 5.10: Marginalized Cumulative Distribution Function (CDF) for the *edges* sufficient statistic for a network of size 4 (up to 12 ties). The blue line shows the CDF for the edges term according to model (1) in Table 5.9, which was fitted without constraining the support of the sufficient statistics, while the red line shows the CDF of model (2), also in Table 5.9, which constrained the support to networks with at least 5 ties. Once again, since we can fully enumerate the support, both CDFs are exact, and thus, not simulated.

(1) n	(2) Homophily (gender)	(3) Receiver (female)	(4) Sender (female)	(5) Homophily$^{1/2}$
4	3	5	6	1.73
4	1	4	3	1.00
4	3	4	3	1.73
5	2	2	4	1.41
5	4	7	5	2.00
5	3	4	3	1.73
		. . . 25 more rows. . .		

Table 5.8: Example of observed sufficient statistics for the team advice networks (bis). For the second set of ERGMs, we included gender-based effects: homophily (2), receiver (3), and sender (4). Variable (5) is the square root of variable (2).

	(1)	(2)	(3)	(4)	(5)	(4b)
edges	-0.52^{**}	-0.91^{***}	-0.54^{**}	-0.72^{***}	-0.48^{*}	-0.72^{***}
	(0.17)	(0.23)	(0.18)	(0.19)	(0.19)	(0.17)
ttriads	0.36^{***}	0.46^{***}	0.37^{***}	0.36^{***}	0.36^{***}	0.36^{***}
	(0.06)	(0.06)	(0.06)	(0.06)	(0.06)	(0.05)
Homophily (gender)	-0.03	-0.01	-0.20	-0.12	-0.01	-0.12
	(0.20)	(0.21)	(0.46)	(0.20)	(0.20)	(0.20)
edges $\times$ $\mathbf{1}(n = 5)$	-0.53^{***}	-0.47^{**}	-0.52^{***}	-0.53^{***}	-0.53^{***}	-0.53^{***}
	(0.12)	(0.16)	(0.13)	(0.13)	(0.12)	(0.13)
(Homophily)$^{1/2}$			0.54			
			(1.32)			
Sender (female)				0.46^{*}		0.46^{*}
				(0.18)		(0.18)
Receiver (female)					-0.08	
					(0.18)	
Constraint (offset)						
edge > 4		Yes				
AIC	639.26	569.93	641.08	634.68	641.07	634.68
BIC	655.99	586.66	661.99	655.59	661.98	655.59
Log Likelihood	-315.63	-280.96	-315.54	-312.34	-315.53	-312.34
Num. networks	31	28	31	31	31	31
Time (seconds)	2.26	2.32	2.28	5.10	5.19	83.97
N replicates						1000
N Used replicates						1000

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 5.9: Testing for gender homopholy. Models (1) through (3) include either an interaction term or a transformation of the term *Homphily (gender)*. Models (4) and (5) include female sender and receiver effects, while model (4b) is a bootstrapped version of model (4). Model (2) constraints the sample space by setting an offset restricting the support to networks with at least 5 edges. Furthermore, since three of the 31 teams had less than five ties, these were excluded from the analysis, hence (2) includes 28 of the 31 available networks.

compared to males, females tended to nominate more of their team members as people they sought advice from. Furthermore, we found that the term *Sender (female)* was a confounder of gender-homophily, with the latter changing from -0.03 in model (1), to -0.12 when the female-sender effect is included. Overall, these final models indicate that the team networks are best explained by preferences for balanced advice-seeking triads, and a tendency for females to seek advice from more of their teammates, compared to males.

With Model (4), Table 5.9, having the best fit overall (smallest AIC and BIC), we re-calculated its standard errors using bootstrapp, model (4b) with the elapsed time, again, remarkably short ($\sim$84 seconds to fit a thousand models). While model (4) took roughly five seconds to be fitted, most of the computation time lies on calculating the support of the space of sufficient statistics. Once the the support of the sufficient statistics has been calculated, the optimization takes only a fraction of the time, which is why the bootstrap version of model (4) took about 0.09 seconds per repetition, and not 5.27 as the user may have expected. Details on the computational resources used for this section and the simulation studies are shown in C.3.

As part of the *ad hoc* diagnostics, Figure 5.11 shows the distribution of the sufficient statistic under the fitted model, 95% exact confidence intervals (CI), versus the observed set of sufficient statistics. With the exception of two networks–one that is a full graph and another that only has one tie–the CIs generated by model (4) are able to cover all other network and term combinations.

We further discuss our results in the following section.

5.7 Discussion

In this paper we revisit and extend Exponential Family Random Graph Models (ERGM) for the case of small networks. Given the interest in testing hypotheses about small networks in the literature, but limited application of statistical models to small network data [39, 55, 93, 104, 121, and others], we shed new light to ERGMs for small networks, which we call ERGM*itos*. An appealing feature of ERGM*itos* is that it allows direct use of the the full likelihood, with all

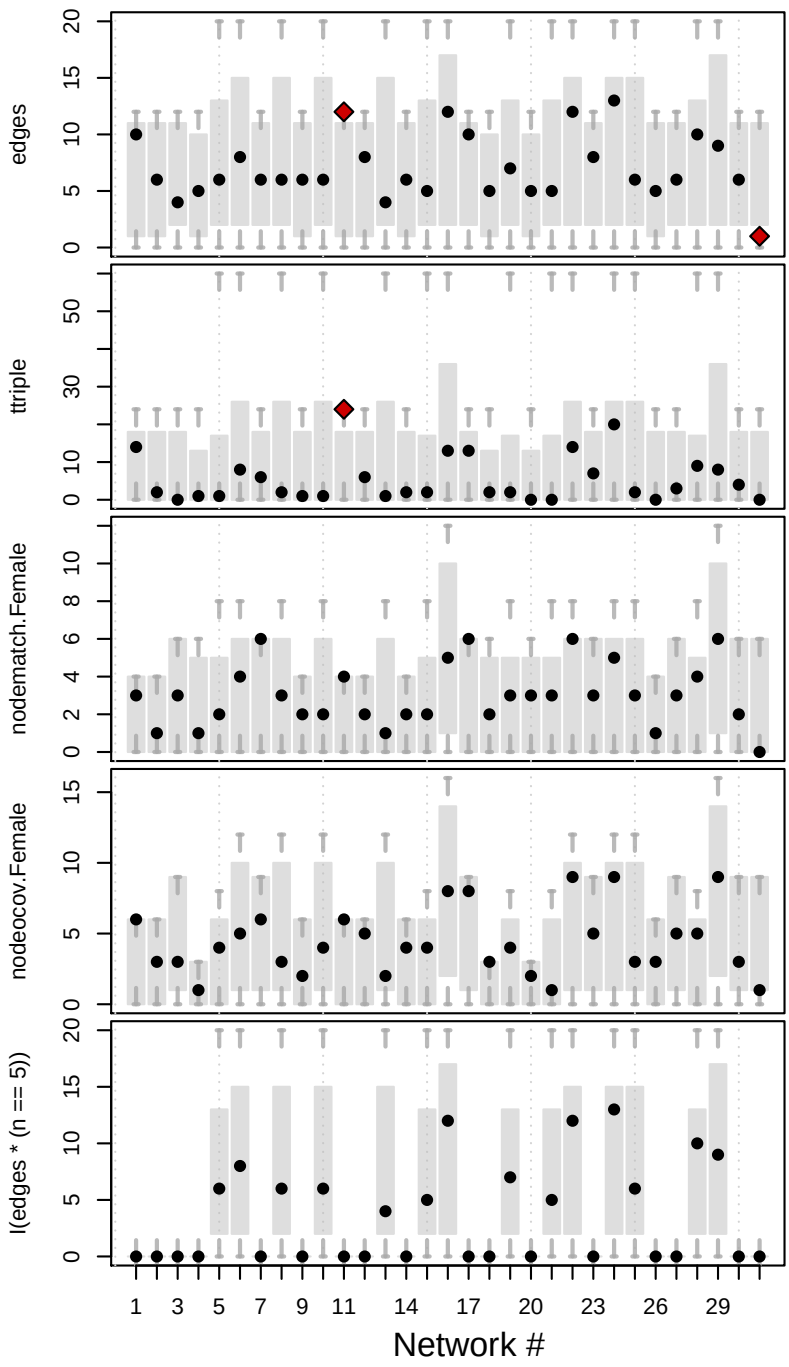


Figure 5.11: Distribution of the sufficient statistics under the ERGM specified by the parameters from model (4) in Table 5.9. Each bar represents the exact 95% confidence interval for the corresponding network+term combination, while the black dots show the location of the corresponding observed statistic. Red diamonds mark the observed statistics that fall out of the 95% confidence interval. Of the 31 networks in the sample, it is only in two networks that the CIs don't cover the observed statistic, one that is fully connected and another that only has one tie.

that that entails, rather than costly approximations via MCMC methods. Overall, this approach provides a couple of important benefits for small network data: (1) it increases the chances of obtaining estimates in the case that the observed sufficient statistics are near the boundary of its support (the convex-hull problem); as suggested by our simulation studies, (2) it has the potential to improve power and reduce type I error rates, compared to MC-MLE and the Robbins-Monro stochastic approximation; and (3) ERGM*itos* can be significantly faster to estimate (orders of magnitude faster), which in turn makes methods like bootstrap or other computationally intensive algorithms immediately available to be used with ERGMs, which to date has been unthinkable.

Another major benefit of using the full likelihood directly, when feasible, is that it gives researchers tremendous flexibility in terms of constructing and estimating new models. In terms of estimation, for example, the ability to easily calculate the likelihood allows researchers to make use of standard tools and techniques for ML estimation and MCMC estimation. This is important, because current techniques for intractable models (e.g., auxiliary variable MCMC, MC-MLE, and noise-contrastive estimation), while effective, are not necessarily straightforward to implement for non-experts. This places a high barrier to entry for researchers who would like to develop statistical models that go beyond the standard packages. As a simple example, take recent work on multilevel network models [102]. In that work, constructing a multilevel Bayesian model of a sample of networks, while conceptually straightforward, required the development of custom code and algorithms to implement. By contrast, having the full likelihood available in *R* means that the same models studied in that paper (assuming small-N networks) can be constructed and estimated as easily as any other non-intractable model for which we can calculate the likelihood, using the full range of traditional tools and algorithms for ML and Bayesian estimation. This frees researchers from focusing only on models that are implementable in packages like *statnet*, and allows greater freedom to think about ways that models for graphs can be modified and incorporated into other statistical models. In addition, being able to estimate gradients opens the possibility of estimating models using modern Bayesian algorithms like Hamiltonian Monte Carlo (HMC) and stochastic gradient langevin dynamics (SGLD), which may offer advantages in

terms of speed or scalability, respectively.

The development and evaluation of ERGM*itos* in this simulation study also brings up topics for future work. One is the evaluation of model goodness-of-fit, and identifying statistics that are most important to evaluate with small networks, and that are reasonable to expect in a model that would suggest a "good fit". Because ERGM*itos* enable a rather simple way of conducting simulation studies (relative to traditional ERGMs), this will facilitate this work in future. Another topic to explore in future work is the value of ERGM*itos* for estimating ERGMs for very large networks, by drawing *samples* of local network structures from a large graph. There is ongoing work extending ERGMs to very large networks [108, 109], and ERGM*itos* could be a valuable approach for fitting these pooled models to a large sample (e.g., in the order of the thousands) of small local network structures drawn from a large network. Although this is an exciting extension to explore, this must proceed cautiously; while some matters as the "projectivity problem" [100] are solvable [68, 69, 99], *size-invariant* ERGMs (models in which the parameter estimates are *scalable* across graph sizes), is an area of research very much under development.

In sum, ERGM*itos* provide a promising extension to the ERGM framework for the analysis of small social networks. In addition to all the theoretical benefits that using exact likelihoods carry with it [50], features related to computational efficiency and flexibility open the door to new *and* old statistical tools that have been unreachable in the ERGM framework. Ultimately, as a fundamental building block of larger social systems, a richer understanding of the local social processes that give rise to the formation of networks in small social groups is key for our understanding of larger social structures that these constitute.

5.8 Acknowledgements

This material is based upon work support by, or in part by, the U.S. Army Research Laboratory and the U.S. Army Research Office under grant number W911NF-15-1-0577. The views, opinions, and/or findings contained in this paper are those of the authors and shall not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documents

Computation for the work described in this paper was supported by the University of Southern California's Center for High-Performance Computing (hpcc.usc.edu).

Chapter 6

Next steps for the ERGMitos

6.1 Exponential Random Graph for Large Networks

The lack of ERGMs applied to large networks is not coincidental. As explained earlier, since the normalizing constant grows at an exponential rate $O(2^{n(n-1)})$, the number of terms goes to infinity very quickly. Yet, there have been some efforts to solve this problems, some of them approximations, as with the MC-MLE approach, and others simply using pseudo-likelihood approaches.

Statistical Exponential Random Graph Models (SERGMs) This framework, first proposed by [18], relies on the fact that for most models exhaustive enumeration is not needed. In principle, we can reduce the number of terms in the normalizing constant by aggregating them according to whether or not sets of graphs share the same sufficient statistic values. For example, if our model only includes *edgecounts* and we are looking at graphs of size 5, the number of unique configurations, which approximates a million, can be reduced to simply $5 \times (5-1) = 20$.

Although encouraging, one of the main assumptions required for SERGMs to work is that we are actually able to sample directly from the space of sufficient statistics. While this may be possible in some cases, like simple counts of edges, isolates, or triangles, more complicated structures, such as attribute in-degree, may not be straightforward. Furthermore, hypotheses

in which the sufficient statistics are a function of nodal attributes makes the *trick* of reducing computation to the iso-statistics useless.

Parametric bootstrap Schmidt et al., [95] proposes a novel method that uses MPLE with bootstrapping for fitting ERGMs for large networks. The method largely relies on the assumption that MPLE provides unbiased estimates of the model parameters, an idea obtained from [26]. While Schmidt et al., shows that for some cases the power of a parametric boostrap version of MPLE is comparable to fitting ERGMs using MC-MLE approaches, the authors do not analyze the type I error rates of their proposed method, nor do they provide a sufficiently extensive simulation study looking at structures other than edgecounts, homophily on a binary nodal attribute, and geometrically weighted edge-wise shared partners (GWESP) [104].

Equilibrium Expectation Algorithm An alternative method for estimating ERGMs for large networks using the *Equilibrium Expectation algorithm* (EE) was presented in [14, 107]. The method has been viewed as a good alternative when estimating networks as big as 1 million nodes. One important issue is that the parameter estimation process is still rather slow. For example, the EE estimator took roughly 7 minutes to estimate a model with $\sim$5,000 nodes, whereas a single ergmito run on a block-diagonal network with 80,000 vertices took 4 seconds (see table S2 from [14])[1].

Snowball sampling Finally, another interesting attempt to estimate ERGMs for large networks can be found in [110]. They propose using a *divide and conquer* strategy. In particular, the authors use what is called snowball sampling [21, 47] to sample from the network in question and do a joint estimation of the subgraphs conditioning on the overlapping ties. The method itself also takes into account the scaling problems present in ERGMs that were mentioned earlier in chapter 5. Like the EE algorithm, their implementation is also undertaken using parallel computing with MPI (and OpenMP if using a single machine). One important issue that the authors pointed out is the fact that this new approach still has some issues when it comes to estimating parameters

[1] To be honest, I am not sure if the timing they show includes the computation of the variance-covariance matrix. In our case, 4 seconds includes it. The current implementation of EE algorithm does work with MPI, meaning that they are parallelizing some of the estimation process.

such as alternating k-stars. Furthermore, in the paper it is also mentioned that more analysis using terms other than homophily and closure—which they use for their simulation studies—is needed. Their approach still relies on MCMC for the full estimation process.

Given this context, one possible next step for the ERGMito framework is to explore the possibility of estimating large networks using an approach similar to that of the parametric boostrap and snowball sampling methods. As ERGMitos provides estimates with lower type I error rates and a significantly faster estimation process compared to MCMC-based approaches, we can imagine a hybrid method combining the precision and speed of ERGMito with some divide-and-conquer strategy.

6.2 Goodness-of-fit

Another important path that we wish to follow is to build a framework for conducting goodness-of-fit (GOF) assessment for ERGMito estimates. Currently, while there is a body of literature that points to best practices and suggests ways in which ERGM model fitting can be assessed [19, 61, 73, 86, 118], as noted in chapter 5, there is no clear view of what statistics should be considered for GOF of little ERGMs. We do understand (at least intuitively) that statistics like *shortest-path-length* may not make much sense in small networks (since there is not much variability), but ultimately we have no quantitative/formal guideline for this.

Following the ideas presented in chapter 3, and leveraging the fact that full enumeration allows us to directly compute ERGM probabilities, I started exploring the ways in which the different statistics used in ERGMs are interrelated. In particular, we can use the fact that the marginal distribution of sufficient statistic A conditional on sufficient statistic B is invariant to the parameter value of B (a sort of *memory-less* property of discrete exponential distributions).

6.2.1 Conditional Distribution: A First Step

Practitioners usually assess goodness-of-fit by evaluating whether the chosen ERGM specification can predict other properties that were not directly included in the model. For example, one usually tries to look at average path lengths or distribution of path lengths and test if the current model was able to capture these higher order properties of the graph that were not included in the model fitting process. Because ERGMitos deal with such a small networks, thinking about these macro-level structures may not be appropriate. Yet, we can take a step back and, instead of directly looking at the GOF conundrum, start by looking at how these sufficient statistics relate to each other by means of conditional distributions.

First introduced in chapter 3, marginal conditional distributions in the context of discrete exponential family models like ERGMs can prove very useful. Formally, reproducing (3.5) from chapter 3, the marginal distribution of sufficient statistic $s(\mathbf{g})_k$ conditional on the sufficient statistic $s(\mathbf{g})_l$ can be calculated as:

$$\mathbb{P}\left(s(\mathbf{G})_k = s_k \mid s(\mathbf{G})_l = s_l, \theta\right) = \frac{\exp\left\{\theta^{\mathbf{t}}_{-l} s(\mathbf{g})_{-l}\right\}}{\sum_{\mathbf{g}':s(\mathbf{g}')_l = s(\mathbf{g})_l} \exp\left\{\theta^{\mathbf{t}}_{-l} s(\mathbf{g}')_{-l}\right\}} \tag{3.5}$$

which has the nice property of being invariant (orthogonal) to θ_l. We can use this probability function to measure how much information statistic l carries about k. With that, and as a proof of concept, I have selected four statistics that are commonly used in the ERGM literature, including the *Edge count* term, to analyze how these are interrelated by means of (3.5); these statistics are: *Number of Mutual Ties*, *Number of Transitive Triads*, *Number of Homophilic Ties*, and *Attribute-Receiver effect*, with the latter two using the binary attribute *Gender*, (which equals to one if the corresponding node is female and zero otherwise). The statistics are shown in Figure 6.1.

Overall, we expect to see that, compared to the *Gender-Homophily* and *Gender-Receiver effect* statistics, *Transitive Triads* and *Mutual Ties* should have a higher interdependence between each other as these statistics can be classified as what is known as Markov-graph structures [39]; which

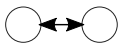
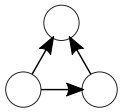
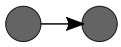
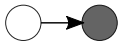
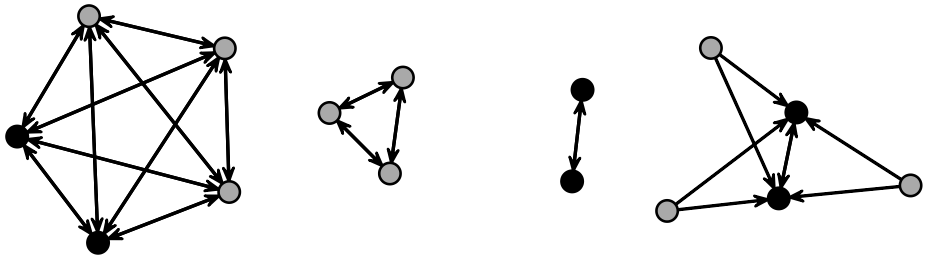
Representation	Description
	Mutual Ties (Reciprocity) $\sum_{i \neq j} y_{ij} y_{ji}$
	Transitive Triad (Balance) $\sum_{i \neq j \neq k} y_{ij} y_{jk} y_{ik}$
	Gender-Homophily $\sum_{i \neq j} y_{ij} \mathbf{1}(x_i = x_j)$
	Gender-receiver effect $\sum_{i \neq j} y_{ij} x_j$

Figure 6.1: A similar version of Figure 5.1. In this case, $x_i = 1$ if the i-th node is female and zero otherwise.

in simple terms can be described as sufficient statistics that count structures involving two or more ties, e.g. mutual ties, triangles, stars, etc. Both *Gender-Homophily* and *Gender-Receiver effect* are dyadic independent terms, as both are defined as simple conditional counts of ties, Figure 6.1 illustrates this.

My proposal is to look at 95% confidence intervals of each statistic, calling this statistic the *focal statistic*, while conditioning on a second statistic, calling this *conditioning statistic*, to measure what we define as predictive power, i.e., the ability to accurately predict the observed focal statistic given the conditioning statistic. For example, we could ask how many mutual ties (the focal statistic) are we expected to see given that we observe 10 transitive triads (conditioning statistic). In the extreme case in which both statistics are essentially co-linear, we should see a one-to-one correspondence, with the upper and lower bounds of the confidence interval overlapping each other. On the contrary, if the two statistics are not associated whatsoever, we should see a wide confidence interval covering the entire support of the focal statistic. Nevertheless, since most structures used as sufficient statistics in ERGMs are ultimately a function of the number of edges, we should still see a mild association in most cases, but again, mostly driven by the fact that counts inherently change with the density of the graph. Figures 6.3 through 6.6 depict the 95% Confidence Interval [CI] (shaded area), and fiftieth-percentile (red dashed-lines), of these statistics when conditioning on each other (rows), while at the same time, either

assuming that the focal statistic has no say in the data-generating-process [DGP], (so fixing $\theta_{\text{focal statistic}} = 0$), column (a), or on the contrary, assuming that the term is part of the DGP, (fixing $\theta_{\text{focal statistic}} = 1$), column (b). The baseline graph used to calculate these statistics was a network of size 5 with the gender attribute equal to $(0, 0, 0, 1, 1)$, i.e., three males and two females. Figure Figure 6.2 illustrates three possible configurations of this and, in particular, what would be the required arrangement (number and assignment of ties) needed to saturate the analyzed sufficient statistics.



(a) Max number for Mutuals/Triads

(b) Max number for Homophily

(c) Max number for Receiver Effect

Figure 6.2: Networks with saturated statistics. All three networks have five vertices composed of two females (black vertices) and three males (gray vertices). Network (a) shows the required set of ties to saturate the *Mutuals* and *Transitive Triads* sufficient statistics, which actually coincides with a fully connected graph, containing $(5 \times 4)/2 = 10$ mutual ties and $5 \times 4 \times 3 = 60$ transitive triads. Graph (b) shows the required configuration to saturate the *Gender-Homophily* sufficient statistic, containing $3 \times 2 + 2 \times 1 = 8$ homophilic ties. Graph (c) shows the minimum arrangement to saturate the *Gender-Receiver Effect* statistic, also having $4 + 4 = 8$ ties (so each female receives 4 ties).

In the case in which the focal statistic is the *Number of Mutual Ties*, see Figure 6.3, we see the following: The number of *transitive triads* (second row) has similar predictive power to the *number of edges* (first row), yet, because each transitive triad adds three new ties (an odd number), the conditional expected number of mutual ties, which consists of an even number of ties, and in particular the correpsonding 95% CI, is not as smooth as that observed when conditioning on the value of the *edge count* statistic. On the other hand, neither *Gender-Homophily* nor *Gender-*

Receiver Effect have a significant effect on the number of mutual ties, other that what we may expect because of the indirect dependency through the overall number of ties in the graph. In general, changing θ_{Mutual} from zero to one, i.e., column (a) vs column (b), seems not to make a significant difference on the analysis.

When the focal statistic is the *Number of Gender-Homophilic Ties*, Figure 6.4 shows the following: First, no conditioning statistic has high predictive power for homophily. At best, when using transitive triads as the conditioning statistic (third row) the CI of the expected number of Gender-Homophilic ties seems slightly narrower compared to the other three. Conversely, the *Gender-Receiver Effect* statistic (last row) seems not to have any meaningful association with the homophily term, and what's more has a very flat slope compared to the other three conditioning statistics.

As seen in Figure 6.5, the CI for the expected number of transitive triads is in general very narrow. Since this statistic reflects relatively high order structures (because more ties are involved), its range of possible values given the number of edge counts or mutual ties is comparatively smaller than what we observed in the case of mutual ties, Figure 6.3. Again, neither *Gender-Homophily* nor *Gender-Receiver Effect* seem to be very related to the transitivity, which appears to be a common theme across these results. Yet, as $\theta_{transitivity} : 0 \rightarrow 1$, column (b), all four conditioning statistics become very good at predicting the number of transitive triads in the graph.

Finally, the CI for the *Gender-Receiver Effect* statistic, Figure 6.6, is relatively large in all cases, with no evident great association to any of the other statistics; which is similar to what was observed in the case of *Gender-Homophily*. Changing $\theta_{\text{Receiver Effect}}$ from zero to one does not result in any dramatic change to the CIs, other than a tendency to increase its value slightly.

In general there were few strong associations between the analyzed sufficient statistics. Nonetheless, when it comes to predicting the number of transitive triads, all of the analyzed statistics seem to have a some ability to accurately predict it, specially when $\theta_{\text{Transitive Triads}} \neq 0$; a rather paradoxical result. With respect to the relatively poor predictive power of the *Gender-Homophily* and *Gender-Receiver Effect* sufficient statistics, besides the fact that these are non-

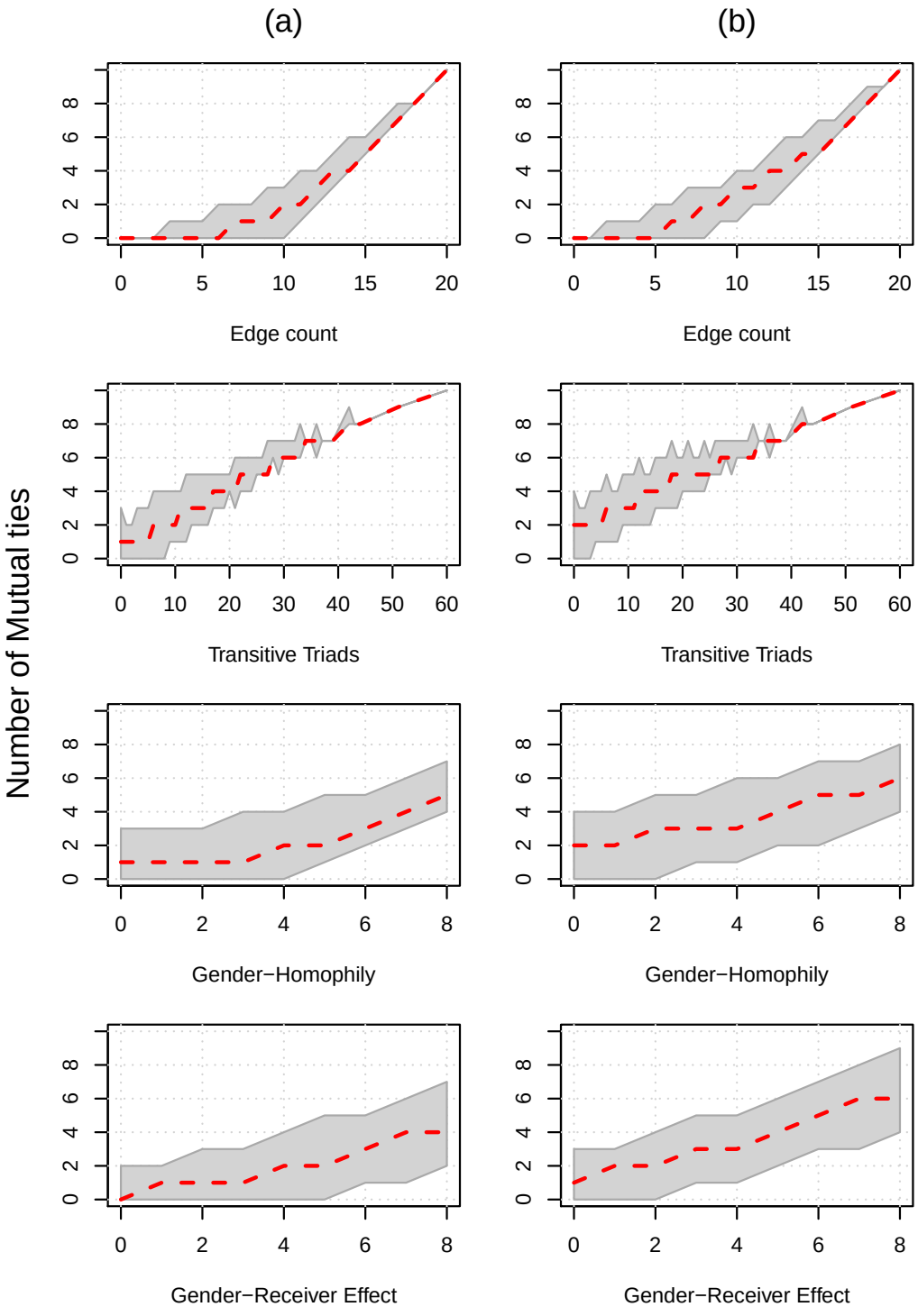


Figure 6.3: Conditional distribution of the number of mutual ties sufficient statistic. Column **(a)** shows the distribution under the Bernoulli model, this is, the parameter of mutual ties is set to zero, whereas column **(b)** shows the distribution of the statistic assuming its parameter equals one.

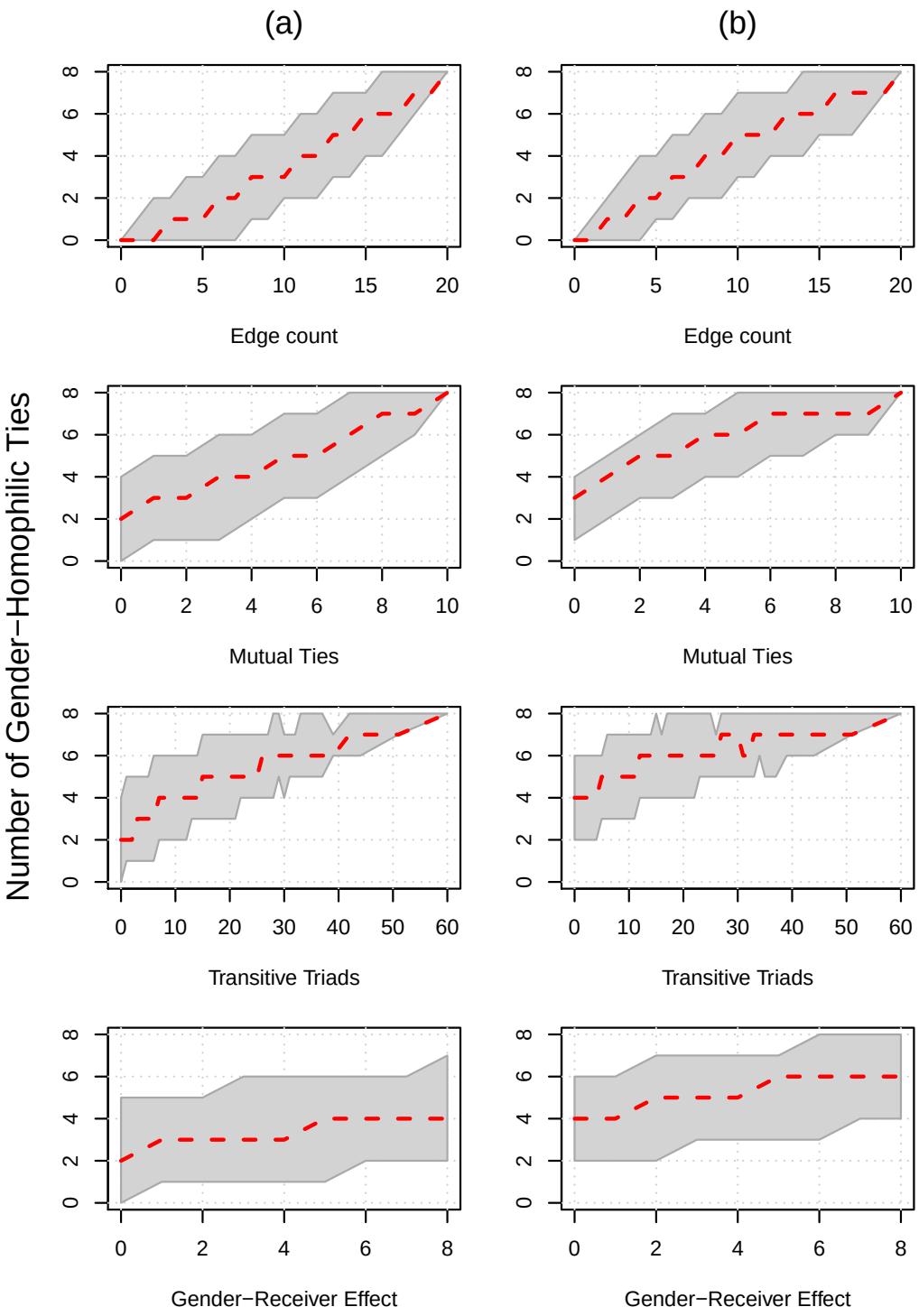


Figure 6.4: Conditional distribution of the number of gender homophilic ties sufficient statistic. Column **(a)** shows the distribution under the Bernoulli model, this is, the parameter of homophily is set to zero, whereas column **(b)** shows the distribution of the statistic assuming its parameter equals one.

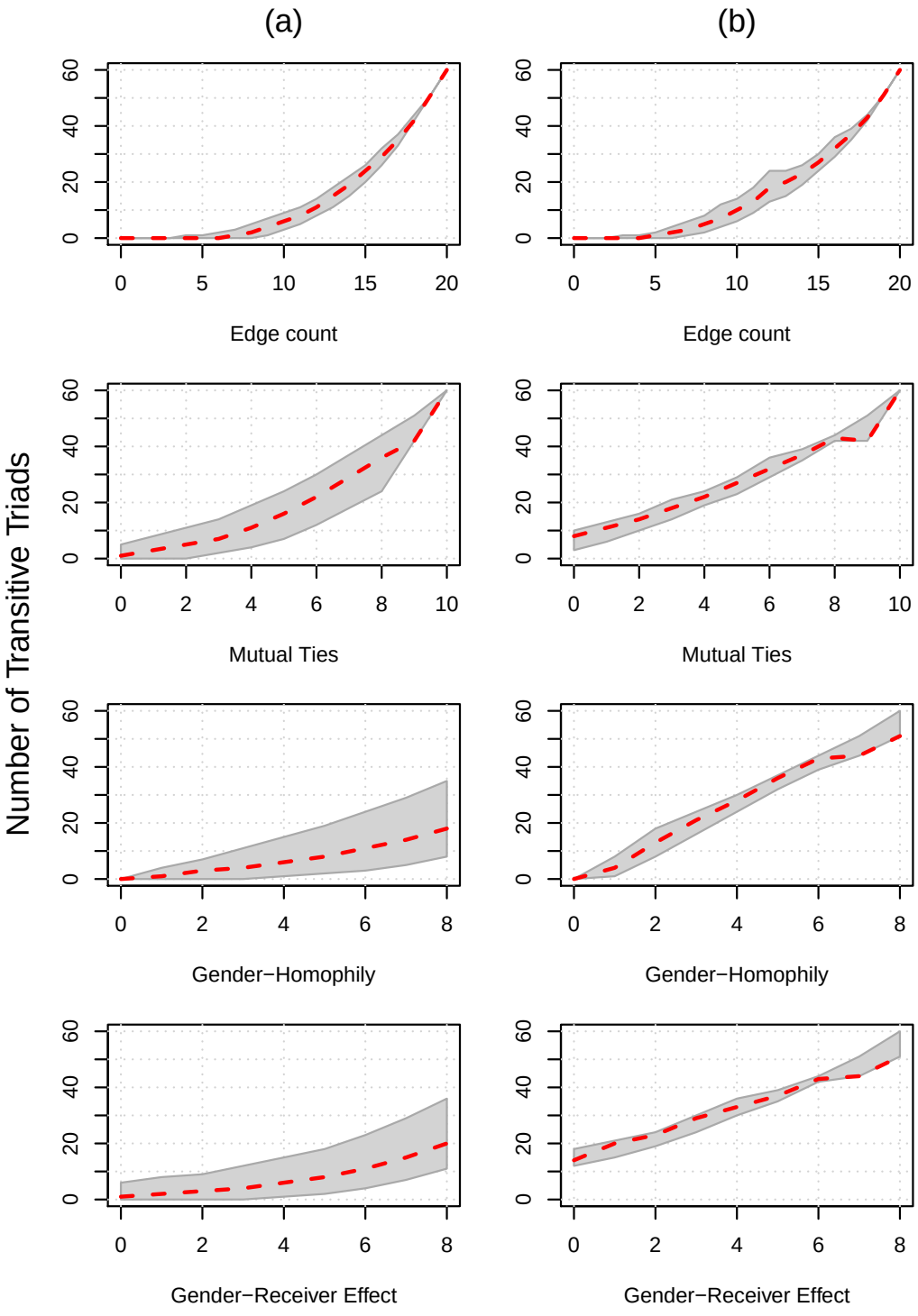


Figure 6.5: Conditional distribution of the number of transitive triads sufficient statistic. Column **(a)** shows the distribution under the Bernoulli model, this is, the parameter of transitivity is set to zero, whereas column **(b)** shows the distribution of the statistic assuming its parameter equals one.

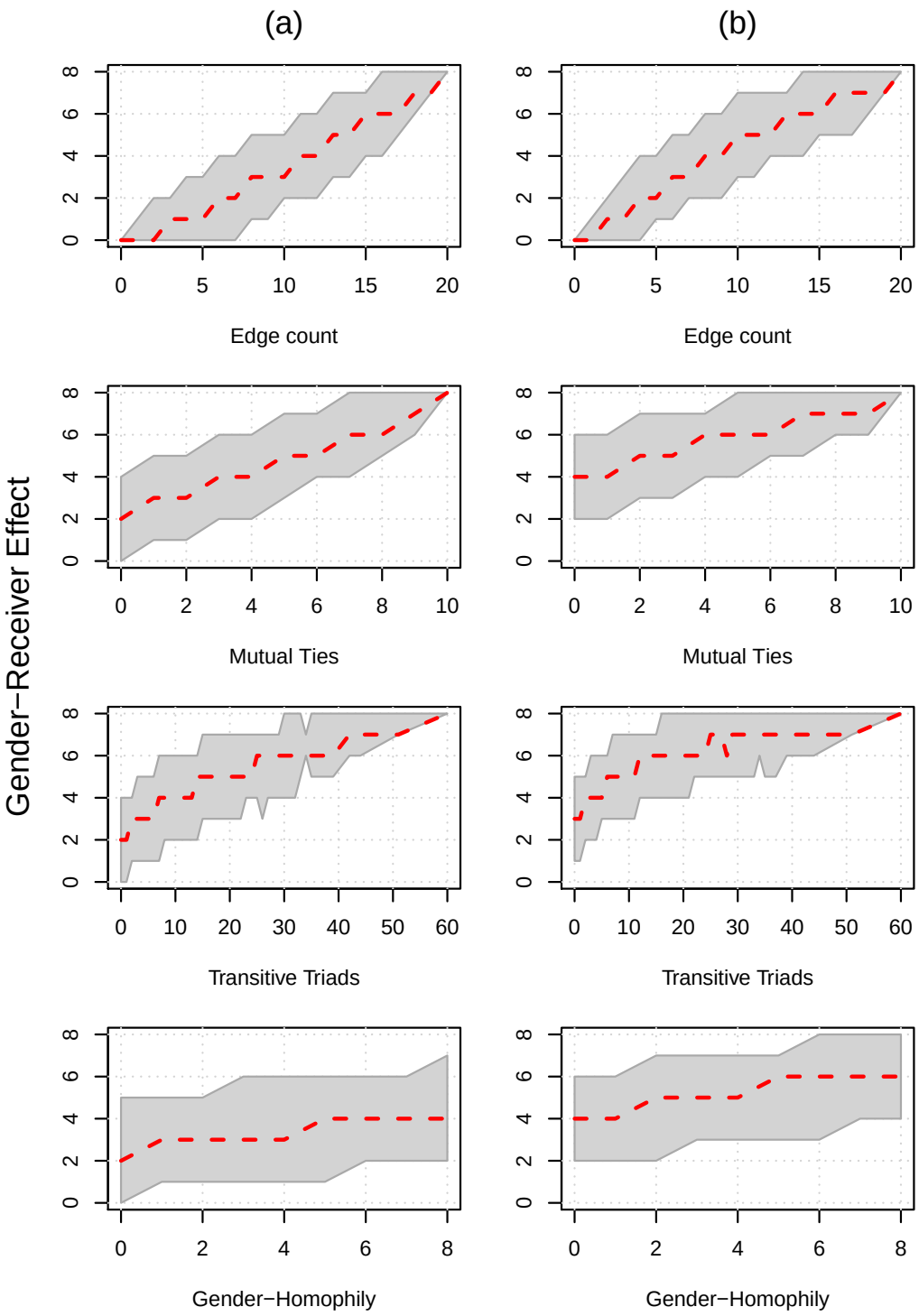


Figure 6.6: Conditional distribution of the value of the gender-receiver effect sufficient statistic. Column **(a)** shows the distribution under the Bernoulli model, this is, the parameter of receiver-effect is set to zero, whereas column **(b)** shows the distribution of the statistic assuming its parameter equals one.

Markovian, part of this could be due to the fact that they reach saturation level very quickly (see Figure 6.2), as in our case they only needed eight out of the 20 possible ties to find a configuration that had the maximum number of either of the statistics. This sort of *censored data* implies that if we observe a homophily/attribute-receiver count equal to 6, 7, or 8, there could be any number of ties between 6 to 18, 7 to 19, or 8 to 20 respectively, i.e. a *information gap* of 12 ties.

There are still many things that we could do to understand, and furthermore, try to identify a proper way of assessing goodness-of-fit for little ERGMs. Furthermore, while what motivated this exercise was reflecting on goodness-of-fit in small networks, looking at the conditional distributions has the potential to serve other purposes. For example: (a) looking for sets of statistics that are independent to each other, i.e. reduce co-linearity, (b) assessing how correlated are sets of statistics and using this information to reduce model complexity, i.e., building a parsimonious model, or (c) to evaluate how stringent would be a permutation test in which we generate a null distribution by conditioning on a given statistic. Ultimately, more simulation studies and mathematical analyses need to be carried out to shed lights on the right strategies for assessing goodness-of-fit in the context of small network modelling.

Chapter 7

Contributed Software

During the development of this work, I have dedicated a significant amount of time to implementing the methods discussed here and building scientific software that allows myself and other researchers to make use of these methods. This chapter lists some of what I consider are the most important pieces of software produced as a result of the methods described throughout this document.

7.1 ergmito: Estimation of Little ERGMs Using Exact Likelihood

> Part of the following section has been extracted directly from the package's website https://github.com/muriteams/ergmito

This R package, which has been developed on top of the amazing work that the Statnet team [51] has done, implements estimation and simulation methods for Exponential Random Graph Models of small networks, in particular, up to 5 nodes in directed networks and 7 in un-directed graphs. In the case of small networks, the calculation of the likelihood of ERGMs becomes computationally feasible, which allows us to avoid approximations and do exact calculation, ultimately obtaining MLEs directly.

- Simulation of streams of small networks using exact statistics.
- Vectorized algorithms for calculating sufficient statistics on vectors of networks.
- Estimation of ergmito models using MLE.
- Performing Goodness-of-fit post estimation analyses.

Currently the ergmito package can be downloaded directly from the project's website or from the Comprehensive R Archive Network (CRAN), https://cran.r-project.org/package=ergmito.

7.2 pruner: Implementing the Felsenstein's Tree Pruning algorithm

The following section has been extracted directly from the package's website https://github.com/USCbiostats/pruner

This C++ template library provides classes to implement Felsenstein tree-pruning algorithm for efficiently computing likelihood functions on phylogenies.

The library reads in a tree object in the form of std::vector< unsigned int > specifying the source and target of each dyad (edge) and allows the user to store arbitrary arguments using memory pointers, and std::function to be called with those arbitrary arguments and tree structure data.

Trees are stored as two lists: each nodes' offspring, and each nodes' parents, which can be accessed at any point using TreeIterator class that implements tree traversals (pre and post order for pruning).

7.3 aphylo: Statistical Inference of Annotated Phylogenetic Trees

The following section has been extracted directly from the package's website https://github.com/USCbiostats/aphylo

The aphylo R package implements estimation and data imputation methods for Functional Annotations in Phylogenetic Trees. The core function consists of the computation of the log-likelihood of observing a given phylogenetic tree with functional annotation on its leaves, and probabilities associated to gain and loss of functionalities, including probabilities of experimental misclassification. Furthermore, the log-likelihood is computed using peeling algorithms, which required developing and implementing efficient algorithms for re-coding and preparing phylogenetic tree data so that can be used with the package. Finally, aphylo works smoothly with popular tools for analysis of phylogenetic data such as ape R package, "Analyses of Phylogenetics and Evolution".

The package is under MIT License, and is been developed by the Computing and Software Cores of the Biostatistics Division's NIH Project Grant (P01) at the Department of Preventive Medicine at the University of Southern California.

7.4 fmcmc: A Friendly MCMC Framework

The following section has been extracted directly from the paper titled *fmcmc: A friendly MCMC framework*, which was published with Paul Marjoram in the Journal of Open Source Software.

Markov Chain Monte Carlo (MCMC) is used in a variety of statistical and computational venues such as: statistical inference, Markov quadrature (also known as Monte Carlo integration), stochastic optimization, among others. The **fmcmc** R [90] package provides a flexible framework for implementing MCMC methods that use the Metropolis-Hastings algorithm [53, 77]. **fmcmc** provides the following out-of-the-box features that can be valuable for both practitioners of MCMC and educators:

- Seamless efficient multiple-chain sampling using parallel computing,
- User-defined transition kernels, and
- Automatic stop using convergence monitoring.

In the case of transition kernels, users can use any one of the transition kernels shipped with the package (e.g. the Gaussian kernel and its bounded version, the Gaussian kernel with reflective boundaries), this allows a degree of flexibility that is not possible with existing MCMC packages.

The main function automatically checks convergence during execution, and stop the MCMC run once convergence has been reached, rather than having to pre-determine a fixed number of iterations. Users can either use one of the convergence monitoring checking functions that are part of the package, for example: The Gelman and Rubin's [42], Geweke's [43], etc., or build their own to be used within the framework.

While there are several other R packages that either implement MCMC algorithms or provide wrappers for implementations in other languages (see for example: [15, 44, 76, 83, 94, 106, 111]), to our knowledge, the 'fmcmc' package is the first one to provide a framework as flexible as the one described here and to be implemented fully within R.

7.5 slurmR: A Lightweight Wrapper for HPC With Slurm

> The following section has been extracted directly from the paper titled *slurmR: A lightweight wrapper for HPC with Slurm*, which was published with Paul Marjoram in the Journal of Open Source Software.

Nowadays, high-performance-computing (HPC) clusters are commonly available tools for either *in* or *out* of cloud settings. Slurm Workload Manager [126, see] is a program written in C that is used to efficiently manage resources in HPC clusters.

While the R programming language [90] has not been developed for HPC settings, there are currently several ways in which R can be enhanced by means of HPC. The 'slurmR' R package is one of those ways.

The 'slurmR' R package provides tools for using R in HPC settings that work with Slurm. It provides wrappers and auxiliary functions that allow the user to seamlessly integrate their analysis pipeline with HPC, putting emphasis on providing the user with a family of functions similar to those that the 'parallel' R package [90] provides.

While there are other tools for integrating R in a HPC envirnment that works with Slurm–see for example `rslurm` [75], `batchtools` [71], `drake` [70], `future.batchtools` [6], `clustermq` [96]–`slurmR` has some advantages regarding syntax, number of dependencies, and flexibility (in terms of the integration with Slurm itself). In particular, you may want to use `slurmR` if you:

1. Need a dependency-free tool. Besides Slurm itself[1], this R package only depends on other R packages that are part of base R.,

2. Need an R package that is fully integrated with Slurm, e.g., submitting jobs with an arbitrary set of Slurm parameters without the need of using templates, call Slurm commands from within R like `sacct`, `scancel`, `squeue`, `sbatch`, etc. with their corresponding flags, and

[1] In fact, users can install this R package regardless of whether or not they have Slurm on their systems. The debug mode of this software allows users to setup jobs (including R scripts and batch files) without having to submit them to a Slurm job-scheduler.

3. Want to use an R package that is ready-to-go. Once loaded, users can submit jobs by just specifying how many cores, for example, they need.

   Other features that are included with this R package, and that are available in some others, are:

4. It uses a syntax similar to the apply family of functions in the parallel R package, including `Slurm_lapply`, `Slurm_sapply`, `Slurm_EvalQ`, and `Slurm_Map`,

5. It can resubmit failed jobs: A very common issue with heterogenous computing clusters is the fact that some jobs succeed while others fail. Partial-job-resubmission is out-of-the-box as users can specify which jobs (as in Job Arrays) should be re-run.

Both of the latter two features also available in **batchtools**. A comparison table of R packages that work with Slurm is available at https://github.com/USCbiostats/slurmR.

In summary, 'slurmR' provides a dependency-free and purpose-built alternative for R users working in a HPC environment with Slurm.

7.6 barray: Tools for Discrete Exponential-Family Models

This is another C++ header-template library that provides data structures and efficient algorithms for enumerating the sample space of arbitrary binary arrays and, taking advantage of the concept of change statistics, fully compute the support of discrete exponential family models.

`barray` only depends on the C++11 standard and, since is a header-only, can be included in C++ programs and compiled on the fly. The current version includes the following features:

- A template class to represent arbitrary arrays as sparse matrices using two adjacency lists which provides fast access to row and column iterators. Furthermore, each sub-list is stored as a hash-map which has linear average complexity on the search, addition, and deletion operations.
- It includes a template class for computing a graph powerset.
- It has a small collection of change statistics for social networks and phylogenetic models.
- Users can compute observed statistics and generate the full support of sufficient statistics to build discrete exponential-family distribution models.

Although it was developed with the main goal of providing a computational efficient way of building discrete exponential models for small arrays, it has shown, at least anecdotally, to have potential for larger data objects as in some cases it can be orders of magnitude faster that other software tools. The latest version of the package can be downloaded from https://github.com/USCbiostats/binaryarrays.

Bibliography

[1] Ryan Admiraal and Mark S Handcock. "Sequential Importance Sampling for Bipartite Graphs With Applications to Likelihood-Based". In: *Statistics* (2006).

[2] Brigham S Anderson, Carter Butts, and Kathleen Carley. "The interaction of size and density with graph-level indices". In: *Social Networks* 21.3 (July 1999), pp. 239–267. DOI: 10.1016/S0378-8733(99)00011-8. URL: http://www.sciencedirect.com/science/article/pii/S0378873399000118%20https://linkinghub.elsevier.com/retrieve/pii/S0378873399000118.

[3] Michael Ashburner et al. "Gene Ontology: tool for the unification of biology". In: *Nature Genetics* 25.1 (May 2000), pp. 25–29. ISSN: 1061-4036. DOI: 10.1038/75556. URL: http://www.nature.com/articles/ng0500%7B%5C_%7D25.

[4] O. Barndorff-Nielsen. *Information and Exponential Families in Statistical Theory*. 2014. ISBN: 9781118857281. DOI: 10.1002/9781118857281.

[5] Debapriya Basu et al. "Measurement of the phospholipase activity of endothelial lipase in mouse plasma". In: *Journal of Lipid Research* 54.1 (Jan. 2013), pp. 282–289. ISSN: 0022-2275. DOI: 10.1194/jlr.D031112. URL: http://www.jlr.org/lookup/doi/10.1194/jlr.D031112.

[6] Henrik Bengtsson. *future.batchtools: A Future API for Parallel and Distributed Processing using 'batchtools'*. R package version 0.7.2. 2019. URL: https://CRAN.R-project.org/package=future.batchtools.

[7] D. Binns et al. "QuickGO: a web-based tool for Gene Ontology searching". In: *Bioinformatics* 25.22 (Nov. 2009), pp. 3045–3046. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btp536. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btp536.

[8] E. Bott and E.B. Spillius. *Family and Social Network*. Families & Marriage. Routledge, 2001. ISBN: 9780415264174. URL: https://books.google.com/books?id=XVwMqHb56TsC.

[9] George E. P. Box. "Science and Statistics". In: *Journal of the American Statistical Association* 71.356 (Dec. 1976), pp. 791–799. ISSN: 0162-1459. DOI: 10.1080/01621459.1976.10480949. URL: http://www.tandfonline.com/doi/abs/10.1080/01621459.1976.10480949.

[10] S Brooks et al. *Handbook of Markov Chain Monte Carlo*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press, 2011. ISBN: 9781420079425. URL: https://books.google.com/books?id=qfRsAIKZ4rIC.

[11] Carter T. Butts. *sna: Tools for Social Network Analysis*. R package version 2.5. 2019. URL: https://CRAN.R-project.org/package=sna.

[12] Carter T. Butts. "Social network analysis: A methodological introduction". In: *Asian Journal Of Social Psychology* 11.1 (Mar. 2008), pp. 13–41. ISSN: 1367-2223. DOI: 10.1111/j.1467-839X.2007.00241.x. URL: http://doi.wiley.com/10.1111/j.1467-839X.2007.00241.x.

[13] Carter T. Butts and Zack W. Almquist. "A Flexible Parameterization for Baseline Mean Degree in Multiple-Network ERGMs". In: *Journal of Mathematical Sociology* 39.3 (2015), pp. 163–167. DOI: 10.1080/0022250X.2014.967851.

[14] Maksym Byshkin et al. "Fast Maximum Likelihood Estimation via Equilibrium Expectation for Large Network Data". In: *Scientific Reports* 8.1 (Dec. 2018), p. 11509. ISSN: 2045-2322. DOI: 10.1038/s41598-018-29725-8. URL: http://www.nature.com/articles/s41598-018-29725-8.

[15] Bob Carpenter et al. "Stan : A Probabilistic Programming Language". In: *Journal of Statistical Software* 76.1 (Jan. 2017). ISSN: 1548-7660. DOI: 10.18637/jss.v076.i01.

[16] Marcos A. Carpio et al. "BCL-2 family member BOK promotes apoptosis in response to endoplasmic reticulum stress". In: *Proceedings of the National Academy of Sciences* 112.23 (June 2015), pp. 7201–7206. ISSN: 0027-8424. DOI: 10.1073/pnas.1421063112. URL: http://www.pnas.org/lookup/doi/10.1073/pnas.1421063112.

[17] Dorothy R. Carter et al. "Social network approaches to leadership: An integrative conceptual review." In: *Journal of Applied Psychology* 100.3 (May 2015), pp. 597–622. ISSN: 1939-1854. DOI: 10.1037/a0038922. URL: http://doi.apa.org/getdoi.cfm?doi=10.1037/a0038922.

[18] Arun G. Chandrasekhar and Matthew O. Jackson. "Tractable and Consistent Random Graph Models". In: (Oct. 2012). arXiv: 1210.7375. URL: http://arxiv.org/abs/1210.7375.

[19] Sixing Chen and Jukka-Pekka Onnela. "A Bootstrap Method for Goodness of Fit and Model Selection with a Single Observed Network". In: *Scientific Reports* 9.1 (Dec. 2019), p. 16674. ISSN: 2045-2322. DOI: 10.1038/s41598-019-53166-6. URL: http://www.nature.com/articles/s41598-019-53166-6.

[20] Davide Chicco. "Ten quick tips for machine learning in computational biology". In: *BioData Mining* 10.1 (Dec. 2017), p. 35. ISSN: 1756-0381. DOI: 10.1186/s13040-017-0155-3. URL: https://biodatamining.biomedcentral.com/articles/10.1186/s13040-017-0155-3.

[21] James Coleman. "Relational Analysis: The Study of Social Organizations with Survey Methods". In: *Human Organization* 17.4 (Dec. 1958), pp. 28–36. ISSN: 0018-7259. DOI: 10.17730/humo.17.4.q5604m676260q8n7. URL: http://sfaajournals.net/doi/10.17730/humo.17.4.q5604m676260q8n7.

[22] The Gene Ontology Consortium. "The Gene Ontology Resource: 20 years and still GOing strong". In: *Nucleic Acids Research* 47.D1 (Nov. 2018), pp. D330–D338. ISSN: 0305-1048. DOI: 10.1093/nar/gky1055. eprint: https://academic.oup.com/nar/article-pdf/47/D1/D330/27437640/gky1055.pdf. URL: https://doi.org/10.1093/nar/gky1055.

[23] N. Crossley et al. *Social Network Analysis for Ego-Nets: Social Network Analysis for Actor-Centred Networks*. SAGE Publications, 2015. ISBN: 9781473927339. URL: https://books.google.com/books?id=v0IdCAAAQBAJ.

[24] Leonardo Dagum and Ramesh Menon. "OpenMP: an industry standard API for shared-memory programming". In: *IEEE computational science and engineering* 5.1 (1998), pp. 46–55.

[25] James A Davis and Samuel Leinhardt. "The structure of positive interpersonal relations in small groups." In: (1967).

[26] B.A. Desmarais and S.J. Cranmer. "Statistical mechanics of networks: Estimation and uncertainty". In: *Physica A: Statistical Mechanics and its Applications* 391.4 (Feb. 2012), pp. 1865–1876. ISSN: 03784371. DOI: 10.1016/j.physa.2011.10.018. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378437111008168.

[27] Arnold Dresden. "The fourteenth western meeting of the American Mathematical Society". In: *Bulletin of the American Mathematical Society* 26.9 (June 1920), pp. 385–397. ISSN: 0002-9904. DOI: 10.1090/S0002-9904-1920-03322-7. URL: http://www.ams.org/journal-getitem?pii=S0002-9904-1920-03322-7.

[28] Dirk Eddelbuettel and Romain François. "Rcpp: Seamless R and C++ Integration". In: *Journal of Statistical Software* 40.8 (2011), pp. 1–18. DOI: 10.18637/jss.v040.i08. URL: http://www.jstatsoft.org/v40/i08/.

[29] Dirk Eddelbuettel and Conrad Sanderson. "RcppArmadillo: Accelerating R with high-performance C++ linear algebra". In: *Computational Statistics and Data Analysis* 71 (Mar. 2014), pp. 1054–1063. URL: http://dx.doi.org/10.1016/j.csda.2013.02.005.

[30] B. Efron and T. Hastie. *Computer Age Statistical Inference*. Institute of Mathematical Statistics Monographs. Cambridge University Press, 2016. ISBN: 9781107149892. URL: https://books.google.com/books?id=Sj1yDAAAQBAJ.

[31] Jonathan A. Eisen. "Phylogenomics: Improving Functional Predictions for Uncharacterized Genes by Evolutionary Analysis". In: *Genome Research* 8.3 (1998), pp. 163–167. DOI: 10.1101/gr.8.3.163. eprint: http://genome.cshlp.org/content/8/3/163.full.pdf+html. URL: http://genome.cshlp.org/content/8/3/163.short.

[32] Barbara E Engelhardt et al. "Genome-scale phylogenetic function annotation of large and diverse protein families". In: *Genome research* 21.11 (2011), pp. 1969–1980. DOI: 10.1101/gr.104687.109.

[33] Barbara E Engelhardt et al. "Protein Molecular Function Prediction by Bayesian Phylogenomics". In: *PLOS Computational Biology* 1.5 (2005). DOI: 10.1371/journal.pcbi.0010045. URL: https://doi.org/10.1371/journal.pcbi.0010045.

[34] Nicholas Eriksson et al. "Polyhedral conditions for the nonexistence of the MLE for hierarchical log-linear models". In: *Journal of Symbolic Computation* 41.2 (Feb. 2006), pp. 222–233. ISSN: 07477171. DOI: 10.1016/j.jsc.2005.04.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0747717105001112.

[35] Katherine Faust. "A puzzle concerning triads in social networks: Graph constraints and the triad census". In: *Social Networks* 32.3 (2010), pp. 221–233. DOI: 10.1016/j.socnet.2010.03.004. URL: http://dx.doi.org/10.1016/j.socnet.2010.03.004.

[36] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern Recognition Letters* 27.8 (June 2006), pp. 861–874. ISSN: 01678655. DOI: 10.1016/j.patrec.2005.10.010. URL: https://linkinghub.elsevier.com/retrieve/pii/S016786550500303X.

[37] Joseph Felsenstein. "Evolutionary trees from DNA sequences: A maximum likelihood approach". In: *Journal of Molecular Evolution* 17.6 (Nov. 1981), pp. 368–376. ISSN: 1432-1432. DOI: 10.1007/BF01734359. URL: https://doi.org/10.1007/BF01734359.

[38] C. Ferri, J. Hernández-Orallo, and R. Modroiu. "An experimental comparison of performance measures for classification". In: *Pattern Recognition Letters* 30.1 (Jan. 2009), pp. 27–38. ISSN: 01678655. DOI: 10.1016/j.patrec.2008.08.010. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167865508002687.

[39] O Frank and David Strauss. "Markov graphs". In: *Journal of the American Statistical Association* 81.395 (1986), pp. 832–842. DOI: 10.2307/2289017. URL: http://amstat.tandfonline.com/doi/abs/10.1080/01621459.1986.10478342.

[40] Olivier Gascuel and Mike Steel. "Predicting the Ancestral Character Changes in a Tree is Typically Easier than Predicting the Root State". In: *Systematic Biology* 63.3 (May 2014), pp. 421–435. ISSN: 1076-836X. DOI: 10.1093/sysbio/syu010. URL: https://academic.oup.com/sysbio/article-lookup/doi/10.1093/sysbio/syu010.

[41] Pascale Gaudet et al. "Phylogenetic-based propagation of functional annotations within the Gene Ontology consortium". In: *Briefings in Bioinformatics* 12.5 (2011), pp. 449–462. DOI: 10.1093/bib/bbr042. eprint: /oup/backfile/content_public/journal/bib/12/5/10.1093/bib/bbr042/2/bbr042.pdf. URL: +%20http://dx.doi.org/10.1093/bib/bbr042.

[42] Andrew Gelman, Donald B Rubin, et al. "Inference from iterative simulation using multiple sequences". In: *Statistical science* 7.4 (1992), pp. 457–472. DOI: 10.1214/ss/1177011136.

[43] John Geweke et al. *Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments*. Vol. 196. Federal Reserve Bank of Minneapolis, Research Department Minneapolis, MN, 1991.

[44] Charles J. Geyer and Leif T. Johnson. *mcmc: Markov Chain Monte Carlo*. R package version 0.9-6. 2019. URL: https://CRAN.R-project.org/package=mcmc.

[45] Charles J. Geyer and Elizabeth A. Thompson. "Constrained Monte Carlo Maximum Likelihood for Dependent Data". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 54.3 (1992), pp. 657–699. ISSN: 00359246. URL: http://www.jstor.org/stable/2345852.

[46] Jeff Gill and Gary King. "What to Do When Your Hessian is Not Invertible: Alternatives to Model Respecification in Nonlinear Estimation". In: *Sociological Methods & Research* 33.1 (2004), pp. 54–87. DOI: 10.1177/0049124103262681. URL: https://doi.org/10.1177/0049124103262681.

[47] Leo A. Goodman. "Snowball Sampling". In: *The Annals of Mathematical Statistics* 32.1 (Mar. 1961), pp. 148–170. ISSN: 0003-4851. DOI: 10.1214/aoms/1177705148. URL: http://projecteuclid.org/euclid.aoms/1177705148.

[48] Heikki Haario, Eero Saksman, and Johanna Tamminen. "An adaptive Metropolis algorithm". In: *Bernoulli* 7.2 (2001), pp. 223–242. URL: https://projecteuclid.org/euclid.bj/1080222083.

[49] Janne Hakkarainen et al. "Hydroxysteroid (17θ) dehydrogenase 1 expressed by Sertoli cells contributes to steroid synthesis and is required for male fertility". In: *The FASEB Journal* 32.6 (June 2018), pp. 3229–3241. ISSN: 0892-6638. DOI: 10.1096/fj.201700921R. URL: https://onlinelibrary.wiley.com/doi/abs/10.1096/fj.201700921R.

[50] Mark S. Handcock. "Assessing Degeneracy in Statistical Models of Social Networks". In: *Working Paper No. 39* 76.39 (2003), pp. 33–50. ISSN: 1936900X. DOI: 10.1.1.81.5086. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.81.5086.

[51] Mark S. Handcock et al. *ergm: Fit, Simulate and Diagnose Exponential-Family Models for Networks*. R package version 3.9.4. The Statnet Project (http://www.statnet.org). 2018. URL: https://CRAN.R-project.org/package=ergm.

[52] Mark S. Handcock et al. *ergm.userterms: User-specified Terms for the statnet Suite of Packages*. R package version 3.10.0. The Statnet Project (https://statnet.org). 2019. URL: https://CRAN.R-project.org/package=ergm.userterms.

[53] W Keith Hastings. "Monte Carlo sampling methods using Markov chains and their applications". In: (1970). DOI: 10.2307/2334940.

[54] Kaisa Henttonen. "Exploring social networks on the team level-A review of the empirical literature". In: *Journal of Engineering and Technology Management* 27.1 (2010), pp. 74–109. ISSN: 0923-4748. DOI: https://doi.org/10.1016/j.jengtecman.2010.03.005.

[55] Paul W. Holland and Samuel Leinhardt. "An exponential family of probability distributions for directed graphs". In: *Journal of the American Statistical Association* 76.373 (1981), pp. 33–50. DOI: 10.2307/2287037.

[56] Douglas G. Howe et al. "Model organism data evolving in support of translational medicine". In: *Lab Animal* 47.10 (Oct. 2018), pp. 277–289. ISSN: 0093-7355. DOI: 10.1038/s41684-018-0150-4. URL: http://www.nature.com/articles/s41684-018-0150-4.

[57] Yu-Chih Hsu et al. "Mesenchymal Nuclear factor I B regulates cell proliferation and epithelial differentiation during lung maturation". In: *Developmental Biology* 354.2 (June 2011), pp. 242–252. ISSN: 00121606. DOI: 10.1016/j.ydbio.2011.04.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0012160611002120.

[58] David R Hunter, Steven M Goodreau, and Mark S Handcock. "Goodness of Fit of Social Network Models". In: *Journal of the American Statistical Association* 103.481 (2008), pp. 248–258. DOI: 10.1198/016214507000000446. eprint: https://doi.org/10.1198/016214507000000446. URL: https://doi.org/10.1198/016214507000000446.

[59] David R. Hunter, Steven M. Goodreau, and Mark S. Handcock. "ergm.userterms: A Template Package for Extending statnet". In: *Journal of Statistical Software* 52.2 (2013), pp. 1–25.

[60] David R. Hunter, Pavel N. Krivitsky, and Michael Schweinberger. "Computational Statistical Methods for Social Network Models". In: *Journal of Computational and Graphical Statistics* 21.4 (2012). PMID: 23828720, pp. 856–882. DOI: 10.1080/10618600.2012.732921. eprint: https://doi.org/10.1080/10618600.2012.732921. URL: https://doi.org/10.1080/10618600.2012.732921.

[61] David R. Hunter et al. "ergm: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks". In: *Journal of Statistical Software* 24.3 (2008), pp. 1–29.

[62] Sohta A Ishikawa et al. "A Fast Likelihood Method to Reconstruct and Visualize Ancestral Scenarios". In: *Molecular Biology and Evolution* 36.9 (Sept. 2019). Ed. by Tal Pupko, pp. 2069–2085. ISSN: 0737-4038. DOI: 10.1093/molbev/msz131. URL: https://academic.oup.com/mbe/article/36/9/2069/5498561.

[63] Ernst Ising. "Beitrag zur Theorie des Ferromagnetismus". In: *Zeitschrift für Physik* 31.1 (Feb. 1925), pp. 253–258. ISSN: 0044-3328. DOI: 10.1007/BF02980577. URL: http://link.springer.com/10.1007/BF02980577.

[64] Martin Jacobsen. "Existence and Unicity of MLEs in Discrete Exponential Family Distributions". In: *Scandinavian Journal of Statistics* 16.4 (1989), pp. 335–349. ISSN: 03036898, 14679469. URL: http://www.jstor.org/stable/4616145.

[65] Yuxiang Jiang et al. "An expanded evaluation of protein function prediction methods shows an improvement in accuracy". In: *Genome Biology* 17.1 (Sept. 2016), p. 184. ISSN: 1474-760X. DOI: 10.1186/s13059-016-1037-6. URL: https://doi.org/10.1186/s13059-016-1037-6.

[66] A. H. Kachroo et al. "Systematic humanization of yeast genes reveals conserved functions and genetic modularity". In: *Science* 348.6237 (May 2015), pp. 921–925. ISSN: 0036-8075. DOI: 10.1126/science.aaa0769. URL: https://www.sciencemag.org/lookup/doi/10.1126/science.aaa0769.

[67] Aleksandr Klepinin et al. "Simple oxygraphic analysis for the presence of adenylate kinase 1 and 2 in normal and tumor cells". In: *Journal of Bioenergetics and Biomembranes* 48.5 (Oct. 2016), pp. 531–548. ISSN: 0145-479X. DOI: 10.1007/s10863-016-9687-3. URL: http://link.springer.com/10.1007/s10863-016-9687-3.

[68] Pavel N. Krivitsky, Mark S. Handcock, and Martina Morris. "Adjusting for network size and composition effects in exponential-family random graph models". In: *Statistical Methodology* 8.4 (July 2011), pp. 319–339. ISSN: 15723127. DOI: 10.1016/j.stamet.2011.01.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1572312711000086.

[69] Pavel N. Krivitsky and Eric D. Kolaczyk. "On the Question of Effective Sample Size in Network Modeling: An Asymptotic Inquiry". In: *Statistical Science* 30.2 (May 2015), pp. 184–198. ISSN: 0883-4237. DOI: 10.1214/14-STS502. URL: http://projecteuclid.org/euclid.ss/1433341477.

[70] William Michael Landau. "The drake R package: a pipeline toolkit for reproducibility and high-performance computing". In: *The Journal of Open Source Software* 3.21 (Jan. 2018), p. 550. DOI: 10.21105/joss.00550.

[71] Michel Lang, Bernd Bischl, and Dirk Surmann. "batchtools: Tools for R to work on batch systems". In: *The Journal of Open Source Software* 2.10 (Feb. 2017). DOI: 10.21105/joss.00135. URL: https://doi.org/10.21105/joss.00135.

[72] Philip Leifeld. "texreg: Conversion of Statistical Model Output in R to LaTeX and HTML Tables". In: *Journal of Statistical Software* 55.8 (2013), p. 24. URL: http://www.jstatsoft.org/v55/i08/.

[73] Yin Li and Keumhee Chough Carriere. "Assessing Goodness of Fit of Exponential Random Graph Models". In: *International Journal of Statistics and Probability* 2.4 (Oct. 2013). ISSN: 1927-7040. DOI: 10.5539/ijsp.v2n4p64. URL: http://www.ccsenet.org/journal/index.php/ijsp/article/view/27900.

[74] D. Lusher, J. Koskinen, and G. Robins. *Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 2012. ISBN: 9781139851039.

[75] Philippe Marchand, Mike Smorul, and Ian Carrol. *rslurm: Submit R Calculations to a Slurm Cluster*. R package version 0.4.0.9000-1. 2017. URL: https://github.com/SESYNC-ci/rslurm.

[76] Andrew Martin, Kevin Quinn, and Jong Hee Park. "MCMCpack: Markov Chain Monte Carlo in R". In: *Journal of Statistical Software, Articles* 42.9 (2011), pp. 1–21. ISSN: 1548-7660. DOI: 10.18637/jss.v042.i09. URL: https://www.jstatsoft.org/v042/i09.

[77] Nicholas Metropolis et al. "Equation of state calculations by fast computing machines". In: *The journal of chemical physics* 21.6 (1953), pp. 1087–1092. DOI: 10.1063/1.1699114.

[78] Huaiyu Mi et al. "PANTHER version 11: expanded annotation data from Gene Ontology and Reactome pathways, and data analysis tool enhancements". In: *Nucleic Acids Research* 45.D1 (2017), pp. D183–D189. DOI: 10.1093/nar/gkw1138. eprint: /oup/backfile/content_public/journal/nar/45/d1/10.1093_nar_gkw1138/3/gkw1138.pdf. URL: +%20http://dx.doi.org/10.1093/nar/gkw1138.

[79] Huaiyu Mi et al. "PANTHER version 14: more genomes, a new PANTHER GO-slim and improvements in enrichment analysis tools". In: *Nucleic Acids Research* 47.D1 (Jan. 2019), pp. D419–D426. ISSN: 0305-1048. DOI: 10.1093/nar/gky1038. URL: https://academic.oup.com/nar/article/47/D1/D419/5165346.

[80] R Milo et al. "On the uniform generation of random graphs with prescribed degree sequences". In: *Arxiv preprint condmat0312028* cond-mat/0 (2004), pp. 1–4. arXiv: 0312028 [cond-mat]. URL: http://arxiv.org/abs/cond-mat/0312028.

[81] Ron Milo et al. "Superfamilies of Evolved and Designed Networks". In: *Science* 303.5663 (Mar. 2004), pp. 1538–1542. DOI: 10.1126/science.1089167. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.1089167.

[82] J. Møller et al. "An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants". In: *Biometrika* 93.2 (June 2006), pp. 451–458. ISSN: 0006-3444. DOI: 10.1093/biomet/93.2.451. eprint: http://oup.prod.sis.lan/biomet/article-pdf/93/2/451/590012/932451.pdf. URL: https://doi.org/10.1093/biomet/93.2.451.

[83] Richard D. Morey. *HybridMC: Implementation of the Hybrid Monte Carlo and Multipoint Hybrid Monte Carlo sampling techniques*. R package version 0.2. 2009. URL: https://CRAN.R-project.org/package=HybridMC.

[84] Joseph M. Morris. "Traversing binary trees simply and cheaply". In: *Information Processing Letters* 9.5 (Dec. 1979), pp. 197–200. ISSN: 00200190. DOI: 10.1016/0020-0190(79)90068-1. URL: http://linkinghub.elsevier.com/retrieve/pii/0020019079900681.

[85] Satoshi Ninagawa et al. "EDEM2 initiates mammalian glycoprotein ERAD by catalyzing the first mannose trimming step". In: *Journal of Cell Biology* 206.3 (Aug. 2014), pp. 347–356. ISSN: 1540-8140. DOI: 10.1083/jcb.201404075. URL: https://rupress.org/jcb/article/206/3/347/37772/EDEM2-initiates-mammalian-glycoprotein-ERAD-by.

[86] Sarah Ouadah, Stéphane Robin, and Pierre Latouche. "Degree-based goodness-of-fit tests for heterogeneous random graph models : independent and exchangeable cases". In: (July 2015). arXiv: 1507.08140. URL: http://arxiv.org/abs/1507.08140.

[87] E. Paradis and K. Schliep. "ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R". In: *Bioinformatics* 35 (2018), pp. 526–528.

[88] Jaewoo Park and Murali Haran. "Bayesian Inference in the Presence of Intractable Normalizing Functions". In: *Journal of the American Statistical Association* 113.523 (2018), pp. 1372–1390. DOI: 10.1080/01621459.2018.1448824. eprint: https://doi.org/

10.1080/01621459.2018.1448824. URL: https://doi.org/10.1080/01621459.2018.1448824.

[89] R. Penrose. "A generalized inverse for matrices". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 51.3 (July 1955), pp. 406–413. ISSN: 0305-0041. DOI: 10.1017/S0305004100030401. URL: https://www.cambridge.org/core/product/identifier/S0305004100030401/type/journal%7B%5C_%7Darticle.

[90] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2018. URL: https://www.R-project.org/.

[91] Predrag Radivojac et al. "A large-scale evaluation of computational protein function prediction". In: *Nature Methods* 10 (Jan. 2013), p. 221. URL: http://dx.doi.org/10.1038/nmeth.2340%20http://10.0.4.14/nmeth.2340%20https://www.nature.com/articles/nmeth.2340%7B%5C#%7Dsupplementary-information.

[92] Alessandro Rinaldo, Stephen E. Fienberg, and Yi Zhou. "On the geometry of discrete exponential families with application to exponential random graph models". In: *Electronic Journal of Statistics* 3 (2009), pp. 446–484. ISSN: 1935-7524. DOI: 10.1214/08-EJS350. URL: https://projecteuclid.org/euclid.ejs/1243343761.

[93] Garry Robins et al. "An introduction to exponential random graph (p*) models for social networks". In: *Social Networks* 29.2 (2007), pp. 173–191. DOI: 10.1016/j.socnet.2006.08.002.

[94] Andreas Scheidegger. *adaptMCMC: Implementation of a Generic Adaptive Monte Carlo Markov Chain Sampler*. R package version 1.3. 2018. URL: https://CRAN.R-project.org/package=adaptMCMC.

[95] Christian S. Schmid and Bruce A. Desmarais. "Exponential random graph models with big networks: Maximum pseudolikelihood estimation and the parametric bootstrap". In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2017, pp. 116–121. ISBN: 978-1-5386-2715-0. DOI: 10.1109/BigData.2017.8257919. URL: http:

//arxiv.org/abs/1708.02598%20http://ieeexplore.ieee.org/document/8257919/.

[96] Michael Schubert. "clustermq enables efficient parallelisation of genomic analyses". en. In: *Bioinformatics* (May 2019). DOI: 10.1093/bioinformatics/btz284. URL: https://github.com/mschubert/clustermq.

[97] Michael Schweinberger. "Instability, Sensitivity, and Degeneracy of Discrete Exponential Families". In: *Journal of the American Statistical Association* 106.496 (Dec. 2011), pp. 1361–1370. ISSN: 0162-1459. DOI: 10.1198/jasa.2011.tm10747. URL: http://www.tandfonline.com/doi/abs/10.1198/jasa.2011.tm10747.

[98] Michael Schweinberger and Mark S. Handcock. "Local dependence in random graph models: characterization, properties and statistical inference". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 77.3 (June 2015), pp. 647–676. ISSN: 13697412. DOI: 10.1111/rssb.12081. URL: http://doi.wiley.com/10.1111/rssb.12081.

[99] Michael Schweinberger, Pavel N. Krivitsky, and Carter T. Butts. *A note on the role of projectivity in likelihood-based inference for random graph models*. 2017. arXiv: 1707.00211 [math.ST].

[100] Cosma Rohilla Shalizi and Alessandro Rinaldo. "Consistency under sampling of exponential random graph models". In: *Ann. Statist.* 41.2 (Apr. 2013), pp. 508–535. DOI: 10.1214/12-AOS1044. URL: https://doi.org/10.1214/12-AOS1044.

[101] Ali Shawki et al. "Intestinal brush-border Na + /H + exchanger-3 drives H + -coupled iron absorption in the mouse". In: *American Journal of Physiology-Gastrointestinal and Liver Physiology* 311.3 (Sept. 2016), G423–G430. ISSN: 0193-1857. DOI: 10.1152/ajpgi.00167.2016. URL: https://www.physiology.org/doi/10.1152/ajpgi.00167.2016.

[102] Andrew J Slaughter and Laura M Koehly. "Multilevel models for social networks: hierarchical Bayesian approaches to exponential random graph modeling". In: *Social Networks* 44 (2016), pp. 334–345. DOI: 10.1016/j.socnet.2015.11.002.

[103] Tom A B Snijders. "Markov Chain Monte Carlo Estimation of Exponential Random Graph Models". In: (2002).

[104] Tom A B Snijders et al. "New specifications for exponential random graph models". In: *Sociological Methodology* 36.1 (Dec. 2006), pp. 99–153. ISSN: 0081-1750. DOI: 10.1111/j.1467-9531.2006.00176.x. URL: http://www.jstor.org/stable/25046693%20http://smx.sagepub.com/lookup/doi/10.1111/j.1467-9531.2006.00176.x.

[105] Tom A. B. Snijders. "Statistical Models for Social Networks". In: *Annual Review of Sociology* 37.1 (2011), pp. 131–153. DOI: 10.1146/annurev.soc.012809.102709.

[106] Stan Development Team. *RStan: the R interface to Stan*. R package version 2.18.2. 2018. URL: http://mc-stan.org/.

[107] Alex Stivala, Garry Robins, and Alessandro Lomi. *Exponential random graph model parameter estimation for very large directed networks*. 2019. arXiv: 1904.08063 [stat.ME].

[108] Alex Stivala, Garry Robins, and Alessandro Lomi. "Exponential random graph model parameter estimation for very large directed networks". In: *PLOS ONE* 15.1 (Jan. 2020). Ed. by Inés P. Mariño, e0227804. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0227804. URL: https://dx.plos.org/10.1371/journal.pone.0227804.

[109] Alex D. Stivala et al. "Snowball sampling for estimating exponential random graph models for large networks". In: *Social Networks* 47 (2016), pp. 167–188. ISSN: 0378-8733. DOI: https://doi.org/10.1016/j.socnet.2015.11.003. URL: http://www.sciencedirect.com/science/article/pii/S0378873315000878.

[110] Alex D. Stivala et al. "Snowball sampling for estimating exponential random graph models for large networks". In: *Social Networks* 47 (Oct. 2016), pp. 167–188. ISSN: 03788733.

DOI: 10.1016/j.socnet.2015.11.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378873315000878.

[111] Sibylle Sturtz, Uwe Ligges, and Andrew Gelman. "R2WinBUGS: A Package for Running WinBUGS from R". In: *Journal of Statistical Software* 12.3 (2005), pp. 1–16. DOI: 10.18637/jss.v012.i03. URL: http://www.jstatsoft.org.

[112] The Gene Ontology Consortium. "Expansion of the Gene Ontology knowledgebase and resources". In: *Nucleic Acids Research* 45.D1 (2017), pp. D331–D338. DOI: 10.1093/nar/gkw1108. eprint: /oup/backfile/content_public/journal/nar/45/d1/10.1093_nar_gkw1108/3/gkw1108.pdf. URL: +%20http://dx.doi.org/10.1093/nar/gkw1108.

[113] P. D. Thomas. "PANTHER: A Library of Protein Families and Subfamilies Indexed by Function". In: *Genome Research* 13.9 (Sept. 2003), pp. 2129–2141. ISSN: 1088-9051. DOI: 10.1101/gr.772403. URL: http://www.genome.org/cgi/doi/10.1101/gr.772403.

[114] George Vega Yon. *ergmito: Exponential Random Graph Models for Small Networks*. R package version 0.2-1-9999. 2020. URL: https://cran.r-project.org/package=ergmito.

[115] George Vega Yon and Paul Marjoram. "fmcmc: A friendly MCMC framework". In: *Journal of Open Source Software* 4.39 (July 2019), p. 1427. ISSN: 2475-9066. DOI: 10.21105/joss.01427. URL: http://joss.theoj.org/papers/10.21105/joss.01427.

[116] George Vega Yon and Paul Marjoram. "slurmR: A lightweight wrapper for HPC with Slurm". In: *The Journal of Open Source Software* 4.39 (July 2019). DOI: 10.21105/joss.01493. URL: https://doi.org/10.21105/joss.01493.

[117] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Fourth. ISBN 0-387-95457-0. New York: Springer, 2002. URL: http://www.stats.ox.ac.uk/pub/MASS4.

[118] Cheng Wang et al. "Model Adequacy Checking/Goodness-of-fit Testing for Behavior in Joint Dynamic Network/Behavior Models, with an Extension to Two-mode Networks". In: *Sociological Methods & Research* (Apr. 2020), p. 004912412091493. ISSN: 0049-1241. DOI: 10.1177/0049124120914933. URL: http://journals.sagepub.com/doi/10.1177/0049124120914933.

[119] Peng Wang, Garry Robins, and Philippa Pattison. "PNet: A program for the simulation and estimation of exponential random graph models". In: *University of Melbourne* (2006).

[120] Peng Wang et al. "Exponential random graph (p*) models for affiliation networks". In: 31 (2009), pp. 12–25. DOI: 10.1016/j.socnet.2008.08.002.

[121] Stanley Wasserman and Philippa Pattison. "Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p*". In: *Psychometrika* 61.3 (1996), pp. 401–425. DOI: 10.1007/BF02294547.

[122] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN: 978-3-319-24277-4. URL: https://ggplot2.tidyverse.org.

[123] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN: 978-3-319-24277-4. URL: http://ggplot2.org.

[124] Jeffrey M Wooldridge. *Econometric Analysis of Cross Section and Panel Data*. 2nd. Cambridge: MIT Press, 2010, p. 1064. ISBN: 978-0-262-23258-6.

[125] Ziheng Yang and Carlos E. Rodríguez. "Searching for efficient Markov chain Monte Carlo proposal kernels". In: *Proceedings of the National Academy of Sciences* 110.48 (2013), pp. 19307–19312. DOI: 10.1073/pnas.1311790110. eprint: http://www.pnas.org/content/110/48/19307.full.pdf. URL: http://www.pnas.org/content/110/48/19307.abstract.

[126] Andy B Yoo, Morris A Jette, and Mark Grondona. "SLURM: Simple Linux Utility for Resource Management". In: *Job Scheduling Strategies for Parallel Processing*. Ed. by Dror Feitelson, Larry Rudolph, and Uwe Schwiegelshohn. Berlin, Heidelberg: Springer Berlin

Heidelberg, 2003, pp. 44–60. ISBN: 978-3-540-39727-4. DOI: 10.1007/10968987_3. URL: http://link.springer.com/10.1007/10968987%7B%5C_%7D3.

[127] Achim Zeileis and Torsten Hothorn. "Diagnostic Checking in Regression Relationships". In: *R News* 2.3 (2002), pp. 7–10. URL: https://CRAN.R-project.org/doc/Rnews/.

Appendix A

Statistically Measuring Prediction Accuracy

When it comes to prediction accuracy, there are several statistics that are widely-used. One of the most popular ones is that calculated from the Receiver Operating Characteristic [ROC] curve, the Area Under the Curve [AUC] (see [36] for a general overview). Yet, while widely popular, using AUCs is not usually recommended for imbalanced samples [20], a very common feature in biomedical sciences data, and what's more, has been shown to be very sensitive to probabilistic noise and the number of elements per class [38]. Among other alternatives that have a better behavior in those two areas is the Mean Square Error [MSE] [38], which we will leverage here.

Instead of looking at the raw prediction, this is, the predicted probabilities of having the function in question, we build a statistic that uses each predicted value $\hat{x}_n$ as the parameter of a Bernoulli distribution, thereby, rather than comparing z_n with $\hat{x}_n$, we compare it with the random variable $b_n \sim \text{Bernoulli}(\hat{x}_n)$. In particular, we use a MSE-based statistic that measures accuracy in a probabilistic way. The proposed statistic is defined as follows:

$$\delta = 1 - |\mathbf{Z}|^{-1} \sum_{n:z_n \in \{0,1\}} (z_n - b_n)^2$$

Since $\{z_n, b_n\} \in \{0, 1\}$, we can re-write the expression in the following way:

$$\delta = 1 - |\mathbf{Z}|^{-1} \left[\sum_{n:z_n=0} b_n + \sum_{n:z_n=1} (1 - b_n) \right]$$

Which is straight forward to compute, have we observed b_n. Instead, we can look at its expected value:

$$\mathbb{E}\left\{\delta \middle| \mathbf{Z}\right\} = 1 - |\mathbf{Z}|^{-1} \left[\sum_{n:z_n=0} \hat{x}_n + \sum_{n:z_n=1} (1 - \hat{x}_n) \right] \tag{A.1}$$

Following an explosion of creativity, we call it *prediction score*. Like most, this statistic is designed such that perfect predictions result in a score of 1, completely wrong predictions result in a score of 0, and a random coin toss results in a score of 0.5.

As δ is a random variable, we can build a relevant null distribution that can be used as a reference for evaluating the quality of the model predictions, in other words, we are able to compute whether the observed value of $\mathbb{E}\left\{\delta \middle| \mathbf{Z}\right\}$ is significantly different from a relevant null. Besides of the obvious case in which we set $b_n \sim \text{Bernoulli}(1/2)$, the fair-coin-toss, one could use a more stringent benchmark using different parameters for predicting zeros and ones. For the more general case, let $\mathbb{P}\left(b_n = 1 \mid z_n = 0\right) = \alpha_0$ and $\mathbb{P}\left(b_n = 1 \mid z_n = 1\right) = \alpha_1$, then $\sum_{n:z_n=0} b_n + \sum_{n:z_n=1}(1 - b_n)$ is a convolution of two binomials with:

$$\begin{aligned}
\hat{X}_0 &= \sum_{n:z_n=0} b_n \sim \text{Binomial}(|n : z_n = 0|, \alpha_0), \\
\hat{X}_1 &= \sum_{n:z_n=1} (1 - b_n) \sim \text{Binomial}(|n : z_n = 1|, 1 - \alpha_1) \quad \text{and} \\
\mathbb{P}\left(\hat{X}_0 + \hat{X}_1 = k\right) &= \sum_{l=0}^{k} \mathbb{P}\left(\hat{X}_1 = l\right) \mathbb{P}\left(\hat{X}_0 = k - l\right)
\end{aligned}$$

From our point of view, the best benchmark is setting the reference probabilities as a function of the observed prevalence of ones in the data. Furthermore, just like we do when calculating the posterior probabilities in subsection 2.2.3, we set α_0 and α_1 to be the prevalence of ones as if we were calculating those in a leave-one-out scheme, this is:

$$\alpha_0 = \frac{|n : z_n = 1|}{|\mathbf{Z}| - 1}, \quad \alpha_1 = \frac{|n : z_n = 1| - 1}{|\mathbf{Z}| - 1}$$

Appendix B

Limiting Behavior of Little ERGMs using the ergmito R package

Suppose that we have five networks of size 4 as the one included in the ergmito package [114], `fivenets`, and we wanted to fit the following model `fivenets` $\sim$ `edges + triadcensus(15)`, with `triadcensus(15)` equal to a fully connected triad, also known as triad class 300 using Davis and Leinhardt triadic classification system [25]. Since the `fivenets` dataset has no fully connected triad, the MLE of that term diverges:

```
library(ergmito)
data(fivenets)
(ans <- ergmito(fivenets ~ edges + triadcensus(15)))
```

```
Warning: The observed statistics (target.statistics) are near or at
the boundary of its support, i.e. the Maximum Likelihood Estimates
may not exist or be hard to be estimated. In particular, the
statistic(s) "edges", "triadcensus.300".

ERGMito estimates
note: A subset of the parameters estimates was replaced with +/-Inf,
and the Hessian matrix is not p.s.d.

```

```
Coefficients:
           edges  triadcensus.300
         -0.6851             -Inf
```

The log-likelihood of this model can be computed directly with ergmito using a large negative instead of `-Inf`, or by using the equation that shows the limiting value of the log-likelihood:

```
# Approach using the limiting value
l <- with(ans$formulae, {
  sapply(1:nnets(ans), function(i) {
    # Preparing suff stats
    S <- t(t(stats_statmat[[i]]) - target_stats[i, ])
    W <- stats_weights[[i]]
    theta <- coef(ans)["edges"]

    # Index of members of S0
    s0idx <- which(S[,2] == 0)

    - log(sum(W[s0idx] * exp(S[s0idx,1] * theta)))
  })
})
sum(l)
```

```
[1] -38.16338
```

```
# Which should be equivalent to the approach with large negative number
ans$formulae$loglik(c(coef(ans)["edges"], -1e100))
```

```
[1] -38.16338
```

We can calculate the gradient of the triad 300 model alternatively using the above expression:

```
g <- with(ans$formulae, {
  sapply(1:nnets(ans), function(i) {
    # Preparing suff stats
    S <- t(t(stats_statmat[[i]]) - target_stats[i, ])
```

```
    W <- stats_weights[[i]]
    theta <- coef(ans)["edges"]

    # Index of members of S0
    s0idx <- which(S[,2] == 0)

    - sum(W[s0idx] * S[s0idx,1] * exp(S[s0idx,1] * theta))/
    sum(W[s0idx] * exp(S[s0idx,1] * theta))
  })
})
sum(g)
```

```
[1] 0.002926372
```

```
# Which is equivalent to
ans$formulae$grad(c(coef(ans)["edges"], -1e100))
```

```
             [,1]
[1,] 0.002926372
[2,] 0.000000000
```

HESSIAN

again, using the example with the triad 300 term we can compute the hessian as follows:

```
h <- with(ans$formulae, {
  sapply(1:nnets(ans), function(i) {
    # Preparing suff stats
    S <- t(t(stats_statmat[[i]]) - target_stats[i, ])
    W <- stats_weights[[i]]
    theta <- coef(ans)["edges"]

    # Index of members of S0
    s0idx <- which(S[,2] == 0)

    # First part
```

```
    - sum(W[s0idx] * S[s0idx,1]^2 * exp(S[s0idx,1] * theta))/
    sum(W[s0idx] * exp(S[s0idx,1] * theta)) +
    # Second bit
    sum(W[s0idx] * S[s0idx,1] * exp(S[s0idx,1] * theta)) ^ 2/
    sum(W[s0idx] * exp(S[s0idx,1] * theta))^2
  })
})
sum(h)
```

```
[1] -12.97199
```

which should be equal to using the full hessian function but with a very large negative velue for the parameter associated with the statistic triad 300:

```
ans$formulae$hess(c(coef(ans)["edges"], -1e100))
```

```
          [,1] [,2]
[1,] -12.97199    0
[2,]   0.00000    0
```

Appendix C

Implementation Details of the ergmito R package

C.1 MLE

The estimation process of ERGM*itos* (as a pooled-data of small networks) is done entirely on R using the *ergmito* R package. While a significant amount of the implementation of the methods described here was done using `Rcpp` [28], a core component of the package is based on statnet's *ergm* R package, and in particular, in the function `ergm.allstats` which does exhaustive enumeration of statistics in a compact way. In general, the estimation process for any list of networks is as follows:

1. Analyze the model to be estimated: Extract the networks from the left-hand-side as specified in the ergm package, and calculate the exact statistics using the ergm.allstats function.

2. With the full enumeration of statistics, build the joint likelihood function of the model in a compact form (i.e., using the weights instead of the full enumeration of the support of the model). This improves speed when it comes to evaluating the log-likelihood function.

3. Because we are dealing with exact statistics, it is also possible to calculate the exact gradient

function. We compute the gradient as follows:

$$\sum_{p} \nabla l_p(\theta) = s\left(y_p, X_p\right)^{\mathbf{t}} - \frac{{Q_p}^{\mathbf{t}} \left({W_p}^{\mathbf{t}} \circ \exp\left\{Q_p\theta\right\}\right)}{\kappa_p} \tag{C.1}$$

Where $s\left(y, X\right)$ is a vector of observed sufficient statistics (usually called target statistics), Q is a matrix of sufficient statistics, in particular, the isomorphic sufficient statistics associated with the model, and W is a vector of frequency weights.

These first three steps carry the most part of the computing time.

4. Finally, the joint log-likelihood is maximized using the BFGS algorith implemented in the the optim function in the stats package.

The final set of estimates is analyzed separately by another program included in the package. The next section describes the evaluation steps followed by ERGM*ito*.

C.2 Evaluation of Estimates

After the optimization procedure finalizes, the *ergmito* package performs a series of tests checking the quality of the estimates. In particular, we conduct the following evaluations after every call to the main optimization function:

1. In the case that the observed sufficient statistics lied on the boundary of its support, the parameter estimate is set to be equal to the corresponding $\pm\infty$.

2. If all parameters turn out to be $\pm\infty$ after this check, the function will send a warning message to the user and the function returns without computing the variance-covariance matrix. In general, the entries of the Hessian that involve a parameter estimate that diverge will be set to zero, which in turns results in zero variance-covariance for those entries when computing the Moore-Penrose generalized inverse.

3. If, on the other hand, a fraction of the parameters were switched to $\pm\infty$, the function recalculates the Hessian and the log-likelihood using the value $\text{sign}(\hat{\theta}_i) \times 10^5$. This is done instead of using ∞, because in most cases using infinite will result in the function being undefined. Again, the function will warn users about this issue. Nevertheless, this is mostly an implementation issue that we are already working on since the limiting values for the log-likelihood, gradient, and hessian are well defined in these cases (see [50]).

4. In the case that the Hessian matrix is non-invertible (not positive-semi-definite [p.s.d.]), we use the Moore-Penrose generalized inverse algorithm as implemented in the R package MASS [117]. For more on the interpretation of variance-covariance matrices when the Hessian is not p.s.d., see [46].

The possible return codes are:

00 optim converged, no issues reported.

01 optim converged, but the Hessian is not p.s.d.

10 optim did not converged, but the estimates look OK.

11 optim did not converged, and the Hessian is not p.s.d.

20 A subset of the parameters estimates was replaced with +/-Inf.

21 A subset of the parameters estimates was replaced with +/-Inf, and the Hessian matrix is not p.s.d.

30 All parameters went to +/-Inf suggesting that the MLE may not exists.

C.3 Computation Details for the ERGMitos Simulation

All the simulations presented in this paper were executed in a large High-Performance-Computing cluster. In general, we use *Slurm* [126] over job arrays with 400 processors. Running the 20,000 simulations took about two hours on the cluster.

In all other cases, i.e. not needing a large computing cluster, model fitting was done using a laptop computer with Ubuntu 18.04 LTS, 8GB of RAM, a quad-core processor Intel® Core i5-7200U CPU @ 2.50GHz, and using R version 3.6.3. The number of cores is relevant as the current implementation of the *ergmito* R package uses `RcppArmadillo` R package [29] which can be compiled using OpenMP [24], meaning that matrix algebra is multi-threaded.