Parameter Estimation in Social Network Models

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Saisuke Okabayashi

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Charles J. Geyer, Adviser

May 2011

Acknowledgements

To my parents, Sahoko and Michio,
and my brothers, Kensuke and Yusuke

Abstract

A social network is an example of a phenomenon with complex stochastic dependence that is commonly modeled with a class of exponential families called exponential random graph models (ERGM). Maximum likelihood estimators (MLE) for such exponential families can be difficult to estimate when the likelihood is difficult to compute. Most methodologies rely on iterated estimates and are sensitive to the starting value, failing to converge if started too far from the solution. Even more problematic is that the MLE may not exist, a situation that occurs with positive probability for ERGMs. In such a case, the MLE is actually "at infinity" in some direction of the parameter space.

Here we present a simple line search algorithm to find the MLE of a regular exponential family when the MLE exists and is unique. The algorithm can be started from any initial value and avoids trial-and-error experimentation. When the MLE does not exist, our algorithm adapts Geyer's (2009a) approach to detect non-existent MLEs and construct one-sided confidence intervals for how close the parameter is to infinity.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation: social network models

Is it possible to build a statistical model that captures the behavioral tendencies of individuals in how they form relationships?

This is the question that led us to study parameter estimation methods for exponential families, with a particular interest in models used to describe social network data. Formally, a *social network* is the collection of *actors* and the *relations*, or *ties*, between each pair of actors. Social scientists have studied social networks as a discipline since the 1930s when Jacob L. Moreno introduced the sociogram, a diagram that corresponds to the mathematical graph with individuals in a group represented by nodes and the presence of a relation between pairs of individuals by an edge (Wasserman and Faust, 1994, Chapter 3). A sociogram depicting the marriage network data among sixteen important families in Renaissance Florence (Padgett, 1994) is depicted in Figure 1.1 and another depicting the affinity, or "liking" relation, among 18 monks in a monastery in New England in the late 1960s (Sampson, 1968) is depicted in Figure 1.2. In such settings, a social scientist is often interested in understanding whether relations arise out of friendliness or a strategy for alliance building, that is,

driven by actor-specific attributes or by the structure of the relations themselves.

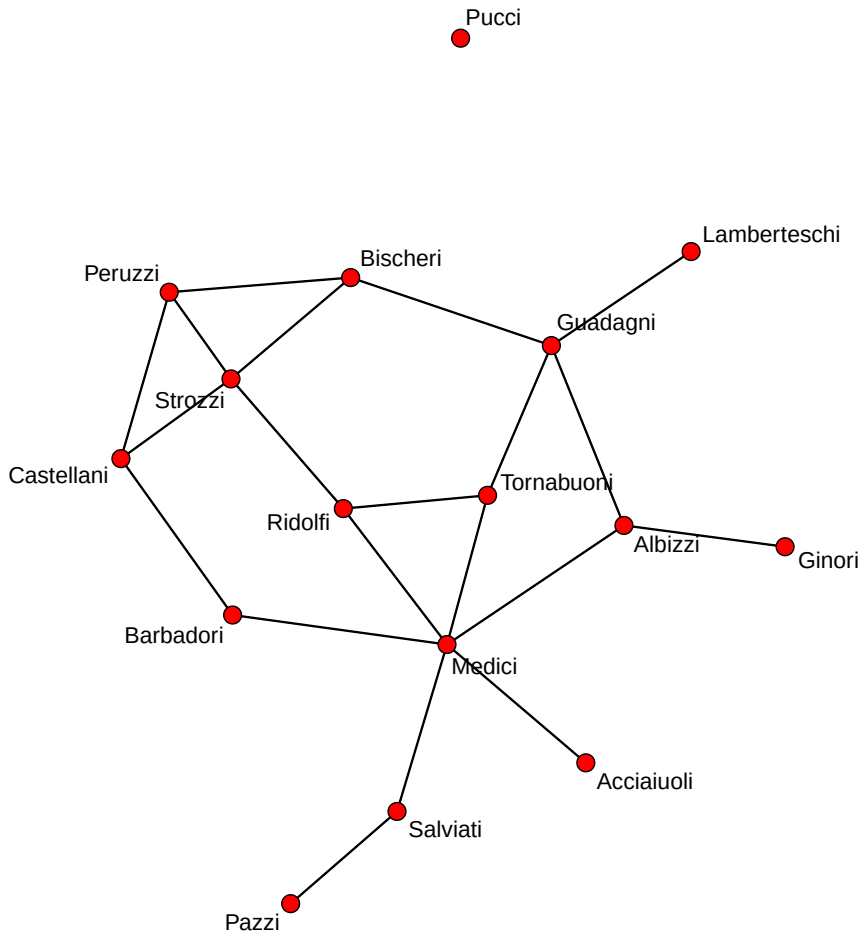

Figure 1.1: Padgett's (1994) Florentine marriage network. Padgett recorded the marriage network among 16 Florentine families around 1430. At the time, two factions, one revolving around the Medicis and the other around the Strozzis, vied for political control of the city. Data is available through and plotted using the `ergm` package (Handcock, Hunter, Butts, Goodreau, and Morris, 2010) in R (R Development Core Team, 2010).

Stochastic network models were developed as early as 1959 in the seminal works of Gilbert (1959) and Erdös and Rényi (1959), resulting in a simple probabilistic model that is now referred to as the Bernoulli model or the Erdös-Rényi-Gilbert

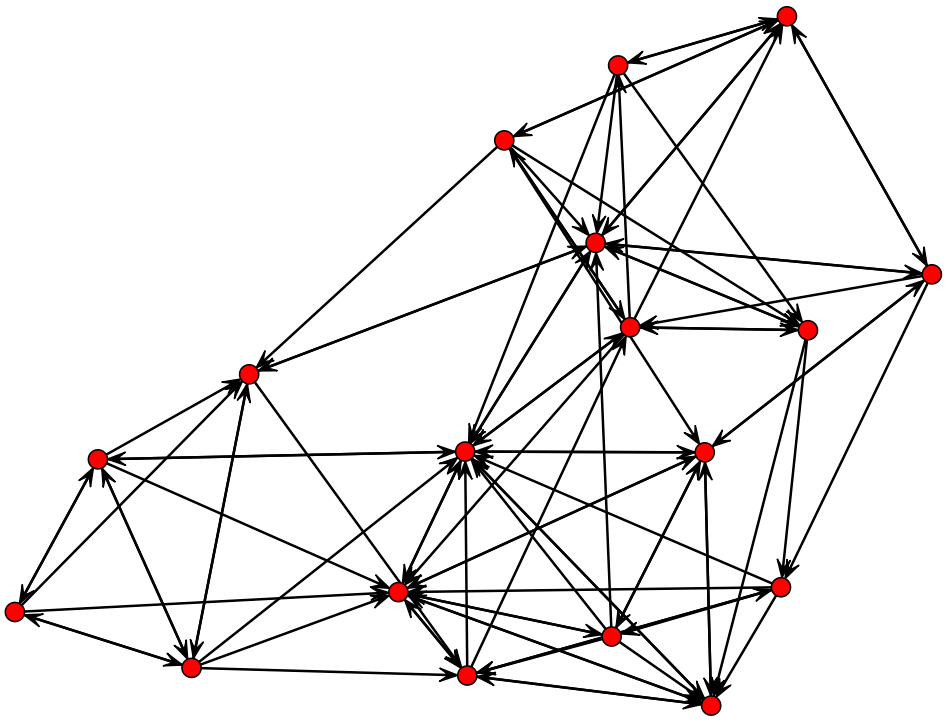

Figure 1.2: Sampson's (1968) monastery affinity network. Sampson collected data to define a "liking" network among 18 monks in a New England monastery in the late 1960s. Since "liking" is a directional relation, the presence of a relation is depicted by an arrow rather than a line. Data is available through and plotted using the `ergm` package (Handcock et al., 2010) in R.

model. Over the last forty years, more sophisticated stochastic network models have flourished; notable landmarks include the p_1 model of Holland and Leinhardt (1981) which captures the reciprocal tendency of relationships in directed networks, and the p^* model of Frank and Strauss (1986) which describes the transitive tendency of relationships in undirected networks. These models laid the foundation for the more general *exponential random graph models (ERGM)*, a class of exponential family models that are now routinely used to model network data and are presented in detail

in Section 1.3.[1]

At their core, stochastic network models attempt to describe whether or not a tie forms between each pair of actors in a group. It has long been observed that relations, especially those involving humans, do not form in isolation; rather, they form in an interdependent manner. Whether or not a relation forms between individuals A and B may very well depend on whether or not relations form between B and C and C and A. This has profound implications for the social scientist, necessitating a new "network" perspective that recognizes the relational structures, such as a triangle of relations between actors A, B, and C, as factors in the analysis (Wasserman and Faust, 1994, Chapter 1). This perspective has garnered increasing attention across different social and behavioral science disciplines over the last forty years, as evidenced by the rapid growth in publication of social network related papers (Knoke and Yang, 2008, Chapter 1).

For the statistician, the network perspective means that the model should not break the network down into independent components, each containing two actors. Instead, a good model will consider important relational structures in the observed network and clarify their contribution in shaping the global outcome. Accompanying the model must also be a computational algorithm to calibrate the model parameters to the network data. In fact, writing down the expression for an uncalibrated ERGM turns out to be quite straightforward; it is fitting the model to data that is an open research problem and the focus of this dissertation.

Typically, parameter estimation for statistical models is done through the method of *maximum likelihood* in which values for the parameters called *maximum likelihood estimators (MLE)* that index the model to make the observed data most likely are calculated. Many methodologies have been developed over the years to address the dif-

[1] To be precise, they should be called "exponential *family* random graph models", but apparently this is too cumbersome.

ficulties in this process for exponential families including Besag's *pseudolikelihood* approach (Besag, 1974; Strauss and Ikeda, 1990), Geyer and Thompson's (1992) *Markov chain Monte Carlo Maximum Likelihood* (MCMC-ML) approach, and *stochastic approximation*.[2] In fact, it was the absence of usable methodologies in parameter estimation that restricted earlier models like p_1 and p^* to their simpler modeling capabilities.

Once model parameters are properly estimated, the fitted model can be used to simulate new random networks whose distribution retains essential characteristics of the observed network. Researchers can then use this distribution to test hypotheses about the process of relationship formation. In this dissertation, we consider the ability to do statistical inference as the end goal of parameter estimation. Although we focus on an algorithm to find MLEs, our end goal is the maximum likelihood *distribution*, which can then be used to construct confidence intervals and evaluate hypotheses.

Finally, it should be noted that although problems from the social sciences provide much of the motivation for social network analysis and our research here, network models can in fact be applied to problems in a broad range of disciplines including political science, biology, epidemiology, and computer science and may be of interest to business practitioners in the utility or transportation sectors. Actors often represent individuals but they may also represent entities like nation-states, protein molecules, airports, computers, or corporations. The relation that connects actors is often friendship or actor A "liking" actor B, as in Sampson's monk data depicted in Figure 1.2, but the relation can be any kind, such as a business transaction or the Internet connectedness between computers. Figure 1.3 depicts the *Escherichia coli* gene transcription network among 423 mRNA molecules (Shen-Orr, Milo, Mangan and Alon, 2002; Salgado, Santos-Zavaleta, Gama-Castro, Millán-Zárate, Díaz-Peredo, Sánchez-

[2]Stochastic approximation has more general applications in root finding, but is frequently used for parameter estimation.

Solano, Pérez-Rueda, Bonavides-Martínez and Collado-Vides, 2001). This data was modeled with an ERGM by Saul and Filkov (2007) and again by Hummel, Hunter and Handcock (2010).

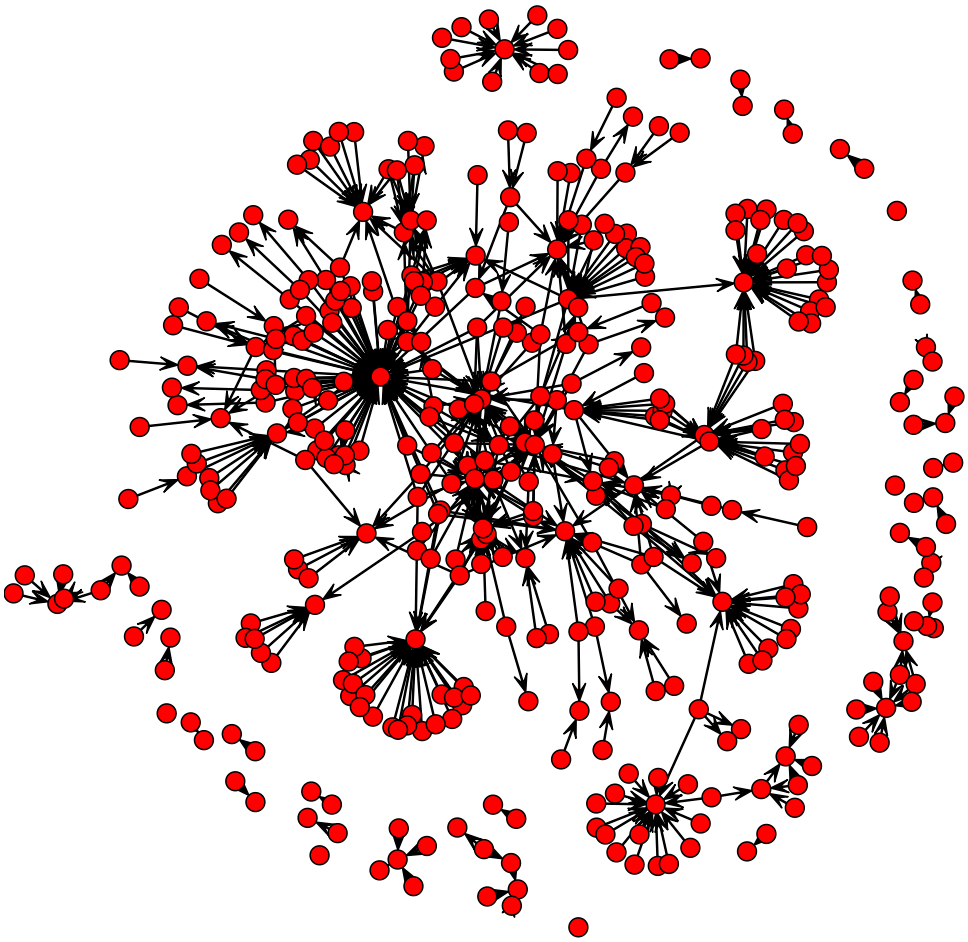

Figure 1.3: *E. coli* transcriptional regulation network of Shen-Orr et al. (2002). Each node is an operon (a group of genes transcribed into a single mRNA molecule) and a directed edge from one node to another indicates that the first encodes the transcription factor that regulates the second. Data is available through and plotted using the `ergm` package (Handcock et al., 2010) in R.

1.1.1 Why model networks?

Before we step into a lengthy of discussion of how we model network data, it is worth reflecting upon the question of why network models are desirable or interesting. We have already given one answer to this question in the context of the social scientist trying to differentiate between competing underlying forces that can shape the global structure of relationships: a network model will allow the researcher to identify if one or more actor specific variables and network structures contribute to the characteristic of the overall network. That is, a good statistical model can see past noise in the data to provide clarity of the underlying forces that shape the structure of an observed network (Goodreau, Kitts and Morris, 2009).

In the context of epidemiology, a model for a *contact network*, which is a network of the potential disease-causing ties between individuals, may be useful in understanding the general mechanism by which an epidemic can spread. This may in turn help formulate an intervention strategy (Welch, Bansal and Hunter, to appear). In computational biology, network models may be used to better understand the processes by which proteins interact (Goldenberg, Zheng, Fienberg and Airoldi, 2009). Some other applications of networks are the links between sites on the Internet or the international relations between countries. At its essence, a network is a conduit for flow, whether it be the flow of diseases, commodities, data, capital, or ideas, and network models can help us understand the mechanism of this flow (Kolaczyk, 2010).

1.2 Exponential random graph models

A network can be modeled as a random $n \times n$ matrix Y, where n is the number of actors. Each entry Y_{ij} in the random matrix Y is itself a random variable representing

a relation from actor i to actor j, such that:

$$Y_{ij} = \begin{cases} 1 & \text{if a relation exists } \textit{from} \text{ actor } i \textit{ to} \text{ actor } j \text{ (notation: } i \to j) \\ 0 & \text{otherwise} \end{cases}$$

where i and j take values in $1, \ldots, n$, $i \neq j$. Note that Y_{ij} take only values of 0 or 1, reflecting our restriction on networks to those with *dichotomous* relations, that is, the relation between a pair of actors is either present or absent. In addition, we do not allow $i \to i$ and always have $Y_{ii} = 0$. In the special case that $Y_{ij} = Y_{ji}$ and thus the matrix Y is symmetric, the network is referred to as an *undirected* network or graph, such as in the case of the Florentine marriage network depicted in Figure 1.1. A network is *directed* if it is not undirected, as in the case of monastery affinity network depicted in Figure 1.2.

The exponential family random graph model (ERGM) commonly used in the network literature for Y has probability mass function of the following form:

$$f_\eta(y) = P_\eta(Y = y) = \frac{1}{\kappa(\eta)} e^{\langle \eta, g(y) \rangle} \qquad y \in \mathcal{Y}, \tag{1.1}$$

where $g(y)$ is a d-vector of *natural statistics*, η is a d-vector of *natural parameters*, $\langle \cdot, \cdot \rangle$ denotes the bilinear form

$$\langle \eta, g \rangle = \sum_{i=1}^{d} g_i \eta_i,$$

and $\mathcal{Y}$ is the sample space of all possible networks with n actors. So that (1.1) integrates to 1, $\kappa(\eta)$ is a normalizing constant such that

$$\kappa(\eta) = \int e^{\langle \eta, g(x) \rangle} \, d\mu(x) \tag{1.2}$$

where μ is a measure on $\mathcal{Y}$. In fact, (1.1) is exactly the form of an exponential family distribution in *canonical form* (Lehmann, 1983, Chapter 1.4). The only distinction that makes this an ERGM is that $\mathcal{Y}$ is specified to be the discrete state space of all possible network configurations.

We rely on many properties of exponential families. Define the *natural parameter space* Ξ as the set of points $\eta = (\eta_1, \ldots, \eta_d)$ that are parameter values indexing distributions in the model. An exponential family is *full* if the natural parameter space is

$$\Xi = \{\eta \in \mathbb{R}^d : \kappa(\eta) < \infty\}, \tag{1.3}$$

and *regular* if, in addition, Ξ is an open set. We say an exponential family is *minimal* if neither the natural parameter nor the natural statistic is concentrated on a hyperplane. Minimality guarantees that if an MLE, denoted $\hat{\eta}_{\text{MLE}}$, exists, it is unique (Geyer, 2009a).

Define the *cumulant* function as $c(\eta) = \log \kappa(\eta)$ and denote the observed data as y_{obs}. We can then express the log likelihood of (1.1) as

$$\ell(\eta) = \langle \eta, g(y_{\text{obs}}) \rangle - c(\eta). \tag{1.4}$$

The appeal of exponential families in the setting of complex dependence phenomena such as networks stems from their simplicity and maximum entropy property (Jaynes, 1978; Geyer and Thompson, 1992). By choosing statistics of interest on the data, one fully specifies a model that, from a maximum entropy perspective, gives the most reasonable inference possible derived solely from those statistics. Furthermore, exponential families have been well-studied (Barndorff-Nielsen, 1978; Brown, 1986) and utilized over the decades and have desirable properties such as the MLE uniqueness noted above.

Ideally then, a network researcher need only specify relational structures of interest to define an ERGM. Wasserman and Pattison (1996); Pattison and Wasserman (1999); Anderson, Wasserman and Crouch (1999); Robins, Pattison, Kalish and Lusher (2007a) describe many of the classical network statistics that one might include in the natural statistic vector $g(y)$. In these works, the researchers' primary consideration in defining a network statistic is a relational structure's scientific interpretability. For example, in a directed affinity network, a sociologist may be interested in the propensity for individuals to form *reciprocal* relations, where ties exist $i \to j$ and $j \to i$, or *transitive* relations, where ties exist $i \to j$, $j \to k$, $i \to k$. The statistics vector $g(y)$ that captures these can be defined by

$$g(y) = \left(\sum_{i<j} y_{ij} y_{ji}, \sum_{i \neq j \neq k} y_{ij} y_{jk} y_{ik} \right)$$

where its components count the number of reciprocal and transitive relational structures in the network y. Such a model, with parameters appropriately calibrated to the affinity network, should then generate networks that exhibit a frequency of reciprocal and transitive relations similar to that of the observed data. We will not discuss the merits of all the different network statistics here; in fact, there is essentially an unlimited number of potential network statistics. What is important is that these statistics can be transparently calculated for a given network and the inclusion of them in $g(y)$, paired with their parameter components in η, allows the model (1.1) to calculate probabilities of different global network outcomes y. Hunter, Goodreau and Handcock (2011) even provide an interface for one to customize and create her own network statistic of interest and model it in the `ergm` package in R.

More recently, Snijders, Pattison, Robins and Handcock (2006); Hunter and Handcock (2006); Robins, Snijders, Wang, Handcock and Pattison (2007b) developed new network statistics with particular emphasis on the properties of the distributions as

opposed to the scientific interpretability of the statistics themselves. It had repeatedly been discovered that for some fitted models, the random networks generated from them did not at all resemble the observed network. These new statistics are intended to reduce the occurrence of such cases. A fairly complete description of these more recent network statistics is in Morris, Handcock and Hunter (2008). We make use of one of these network statistics in the example in Section 5.4.

1.2.1 Intractable normalizing constant

We now return to the issue of why parameter estimation for ERGMs and similar exponential family models on discrete state spaces can be challenging. Despite the tremendous appeal of the exponential family framework, one immediate problem is that the summation in (1.2) for the normalizing constant $\kappa(\eta)$ is over all possible networks in the sample space $\mathcal{Y}$ and can be prohibitively expensive to evaluate for networks of even moderate size. For an undirected network with n actors, there are $2^{\binom{n}{2}}$ different possible networks in $\mathcal{Y}$. Table 1.1 shows how rapidly this number grows; unless dealing with networks of 9 actors or less, the likelihood function should not be directly evaluated. See Appendix A for our approach for counting simple network statistics in the 9-node network relying on numerous tricks in data representation and computation.

1.2.2 Covariate data

Information about a particular actor, say gender, can be incorporated as covariate data into the model with little difficulty. Often, a social scientist looks to include such information because she is interested in whether or not there is an *assortative mixing* effect, that is, whether individuals of the same "type" tend to make more relations with others of that type (Goodreau et al., 2009). Suppose we wish to

Table 1.1: Sample space size for undirected networks with different numbers of actors.

Nodes	Possible Edges	Total Graphs
5	$\binom{5}{2} = 10$	$2^{10} = 1024$
6	$\binom{6}{2} = 15$	$2^{15} = 32,768$
7	$\binom{7}{2} = 21$	$2^{21} = 2,097,152$
8	$\binom{8}{2} = 28$	$2^{28} = 268,435,456$
9	$\binom{9}{2} = 36$	$2^{36} = 68,719,476,736$
10	$\binom{10}{2} = 45$	$2^{45} = 3.518437 \times 10^{13}$

incorporate p such exogenous attributes for our n-actor network. This information can be represented by an $n \times n \times p$ matrix X, whose ijkth element is the value of the kth attribute in the potential relation from actor i to j (Fienberg and Wasserman, 1981; Hunter, Handcock, Butts, Goodreau and Morris, 2008b). We include such factors in our model of an adolescent friendship data set in Section 5.4.

Like the other network statistics discussed, the statistics that comprise X can be transparently calculated from data and hence included in the canonical statistics vector as $g(y, X)$, with the canonical parameter vector η lengthened accordingly. The parameter estimation methodology is unchanged from before (so long as p is not excessively large), and while this greatly expands the usefulness of ERGMs to the researcher, there are no new issues for us to consider from a statistical modeling perspective. Thus we will continue to use $g(y)$ instead of $g(y, X)$ for simplicity.

1.3 Examples of ERGMs

We now review the classical Erdős-Rényi-Gilbert, p_1, and p^* models to give both a sense of commonly used network structures as well as parameter estimation methodologies.

1.3.1 Erdős-Rényi-Gilbert model

The simplest example of an exponential random graph is the Erdős-Rényi-Gilbert model (Erdös and Rényi, 1959; Gilbert, 1959), also referred to as a Bernoulli network model, which assumes that each actor forms a relation to every other actor independently with the same probability p (Hunter et al., 2008b). The ERGM can be expressed as

$$P_\eta(Y = y) = \frac{1}{\kappa(\eta)} e^{\eta g(y)} \qquad y \in \mathcal{Y},$$

where the only network statistic is a count of the number of edges for the directed network

$$g(y) = \sum_{i \neq j} y_{ij}$$

and the probability of a tie formation between any pair of actors is the same,

$$p = P_\eta(Y_{ij} = 1) = \frac{e^\eta}{1 + e^\eta}.$$

Then the log likelihood can be expressed as

$$\ell(\eta) = \eta \sum_{i \neq j} y_{ij} - n(n-1) \log(1 + e^\eta).$$

The MLE of η, $\hat{\eta}_{\text{MLE}}$, can be found analytically to be the logit of the fraction of ties that are present in the observed data set,

$$\hat{\eta}_{\text{MLE}} = \text{logit} \left(\frac{\sum_{i \neq j} y_{\text{obs},ij}}{n(n-1)} \right)$$

where $n(n-1)$ is the number of possible ties in a directed network with n actors.

The MLE for η is thus easily calculated from the observed data. The independence assumption, however, is too unrealistic for all but the simplest of cases; usually, a researcher is interested in modeling different probabilities of tie formations between actors. Thus the Erdős-Rényi-Gilbert model may be most useful as a "null" model, though it is arguably too simple even for this.

1.3.2 The p_1 model

Holland and Leinhardt (1981) made advances in relaxing this independence assumption with their p_1 model. They focused on two empirical observations from sociometric studies:

- Reciprocation: there tend to be a "surplus" of mutual relationships in network data sets compared to a uniform distribution of directed relationships.
- Stars: some individuals attract a surplus of choices compared to a uniform distribution of directed relationships.

Holland and Leinhardt then constructed a family of distributions with parameters to control the probability of observing different numbers of mutual relationships and stars. Focusing on the *dyad*, the set of a pair of actors and the possible relations between them, as the basic building block, they proposed the following model:

$$P(Y = y) = \frac{1}{K(\rho, \theta, \{\alpha_i\}, \{\beta_j\})} \exp\left\{\rho m(y) + \theta y_{++} + \sum_i \alpha_i y_{i+} + \sum_j \beta_j y_{+j}\right\}$$

subject to $\sum_i \alpha_i = \sum_j \beta_j = 0$, where

$$\begin{aligned}
\rho &= \text{“force of reciprocation” or mutuality parameter} \\
m(y) &= \sum_{i \neq j} y_{ij} y_{ji}, \quad \text{total number of mutual relationships in } y \\
\theta &= \text{“density” or overall choice effect parameter} \\
y_{++} &= \sum_{i \neq j} y_{ij}, \quad \text{total number of relations in } y \\
\alpha_i &= \text{“productivity” or “expansiveness” effect parameter for node } i \\
y_{i+} &= \sum_{j} y_{ij}, \quad \text{“out-degree” for node } i \text{ in } y \\
\beta_j &= \text{“attractiveness” or “popularity” effect parameter for node } j \\
y_{+j} &= \sum_{i} y_{ij}, \quad \text{“in-degree” for node } j \text{ in } y \\
K &= \text{normalizing constant}
\end{aligned}$$

By defining new dyad random variables, $D_{ij} = (Y_{ij}, Y_{ji})$, Holland and Leinhardt showed that with some algebraic manipulation, the form of the model above can be viewed as a log-linear model with independent dyad random variables D_{ij}. This makes it possible to use a standard logistic regression to calculate MLEs of the parameters.

The statistical independence at the dyad level, however, means that this model will not capture triangular tie configurations (or anything more complicated) in which dyads are dependent. Also, to reduce the number of parameters, the model assumes $\rho_{ij} = \rho$ and $\theta_{ij} = \theta$, meaning that the tendency towards reciprocity and forming relations is assumed to be the same across all actors. This example illustrates how the parameter estimation methodology limits the scope of the model and what types of behavior it can capture.

1.3.3 Markov graph model

Frank and Strauss (1986) relaxed the independence assumption further with the implementation of *Markov dependence* in which two dyads are independent, conditional on the rest of the graph, when they do not share a node. The model uses only three configurations in an undirected network, expressed as:

$$P(Y = y) = \frac{1}{K(\theta, \sigma, \tau)} \exp\{\theta L + \sigma S + \tau T\}$$

where

$\theta =$ edge parameter

$L = \sum_{i<j} y_{ij}$, total number of edges

$\sigma =$ 2-star parameter, propensity for individuals to have connections with two actors

$S = \sum_{i<j<k} y_{ij} y_{ik}$, total number of 2-stars ($i \leftrightarrow j$, $i \leftrightarrow k$)

$\tau =$ Triangle parameter, represents clustering

$T = \sum_{i<j<k} y_{ij} y_{jk} y_{ik}$, total number of triangles ($i \leftrightarrow j$, $j \leftrightarrow k$, $i \leftrightarrow k$)

None of the above parameters have subscript indices, reflecting the simplification from a *homogeneity* assumption where parameters are equated if the network structures are the same ignoring the labels on the nodes (also called *isomorphic* configurations. In fact, this is the same simplification Holland and Leinhardt employ for the ρ and θ parameters in their p_1 model).

The model is the first to break dyad independence, made possible by Frank and Strauss' methods of parameter estimation. In particular, Frank and Strauss ran Markov chain Monte Carlo simulations of the model at multiple values for a pa-

rameter to determine which fit the data best. The authors also obtained *maximum pseudolikelihood estimators (MPLE)* from a standard logistic regression (we discuss the pseudolikelihood approach in Section 1.4.1) and observed that the MPLEs are close to those they arrived at from their rigorous simulations. Frank and Strauss seem to conclude that MPLEs are generally quite acceptable.

1.4 ERGM parameter estimation

Exponential random graph models have a deceptively simple form (1.1) that belies the challenges involved in model parameter estimation. As discussed in Section 1.2.1, the difficulties arise from a normalizing constant (1.2) which may involve a summation over an astronomical number of terms. In this section, we discuss commonly used parameter estimation methods used for exponential families with complex dependence like ERGMs. All approaches avoid evaluating the likelihood function directly but have properties that we find undesirable. The methods are aided by a strictly concave log likelihood function (see Section 2.1) and thus there is no worry about local maxima. There can be at most one local maximum, which (if it exists) is the global maximum.

1.4.1 Maximum pseudolikelihood method

As mentioned in Section 1.3.2, Frank and Strauss (1986) successfully applied the maximum pseudolikelihood method to social network models, a method first used in lattice systems in plant biology (Besag, 1974, 1975). Strauss and Ikeda (1990) further justified the use of maximum pseudolikelihood estimators (MPLE) as reasonable approximations for MLEs in social network models. Wasserman and Pattison (1996); Pattison and Wasserman (1999); Anderson, Wasserman and Crouch (1999) leaned on this result to broaden the scope of network models, allowing for any combination of network structures to be considered. The result is (1.1), the more general form of the

ERGM that is currently used.

The method of maximum pseudolikelihood finds the values for the parameters that maximize the *pseudolikelihood function* for the observed data set, which can be constructed from the densities of Y_{ij} conditional on the rest of network, denoted by $f_{Y_{ij}}(y_{ij} \mid \text{rest})$. The pseudolikelihood function $PL(\eta)$ is defined to be the product of these densities,

$$PL(\eta) = \prod_{i \neq j} f_{Y_{ij}}(y_{ij} \mid \text{rest}), \tag{1.5}$$

and in general is different than the likelihood function.

Define the vector of *change statistics* $\delta_g(y)_{ij}$ to be

$$\delta_g(y)_{ij} = g(y_{ij}^+) - g(y_{ij}^-)$$

where y_{ij}^+ and y_{ij}^- represent networks with $y_{ij} = 1$ and $y_{ij} = 0$, respectively, while leaving the rest of the network as y. Thus $\delta_g(y)_{ij}$ is the change in $g(y)$ when y_{ij} changes from 0 to 1. The conditional distribution of $Y_{ij} \mid$ rest for an ERGM is then a Bernoulli distribution with log odds

$$\log\left(\frac{P(Y_{ij} = 1 \mid \text{rest})}{P(Y_{ij} = 0 \mid \text{rest})}\right) = \eta^T \delta_g(y)_{ij}. \tag{1.6}$$

The form of (1.6) in the context of (1.5) naturally lends itself to the use of logistic regression to find values of η that maximize $PL(\eta)$. These values of η are called the maximum pseudolikelihood estimators.

In the case where the Y_{ij} are in fact mutually independent, the MPLEs will equal the MLEs. Strauss and Ikeda (1990) show that in many cases with dyad dependence, MPLE still yield reasonable approximations of the true MLEs.

However, Geyer and Thompson (1992); Snijders (2002); Robins, Pattison, Kalish

and Lusher (2007a); van Duijn, Gile and Handcock (2009) demonstrated that the pseudolikelihood approach can produce very misleading results when dependence is strong. Okabayashi, Johnson and Geyer (2011) showed that the pseudolikelihood approach can be improved upon in a generalization called the *composite likelihood* approach; this method is identical to pseudolikelihood except that the conditional distributions used to build the *composite likelihood function*—analogous to the pseudolikelihood function (1.5)—are for multiple ties rather than a single tie. Hummel et al. (2010) raised two additional issues with the pseudolikelihood approach, which also apply to the composite likelihood approach. The first is that this approach requires an actual network y_{obs} from which to derive the MPLE, as opposed to a network statistics $g(y_{\text{obs}})$ which may be some vector of theoretical interest to which one would like to fit an ERGM. Thus given a $g(y_{\text{obs}})$, one would then need to take the extra step of finding a y_{obs} that matches this vector of interest. The second point is that there are several networks that yield the same $g(y_{\text{obs}})$. Because ERGMs depend only on $g(y_{\text{obs}})$ and not y_{obs}, these will all yield the same MLE. However, there is no guarantee that they will yield the same MPLE. Network software packages such as `statnet` (Handcock, Hunter, Butts, Goodreau and Morris, 2003) in the R platform now overwhelmingly use MLE methods, using MPLEs only as starting points as in the example in Section 5.4.

1.4.2 Newton-Raphson

Newton-Raphson is one of the most commonly used root-finding algorithms in optimization, attractive for its speed of convergence. It relies on iterated updates of a root function, which in our setting is the gradient of the log likelihood, $\nabla\ell(\eta)$. The algorithm also requires the Hessian matrix $\nabla^2\ell(\eta)$ and has updates of the following

form:

$$\eta_{k+1} = \eta_k - \left[\nabla^2 \ell(\eta_k)\right]^{-1} \nabla \ell(\eta_k). \tag{1.7}$$

The algorithm may fail to converge when the initial η_k is far from the solution. However, when Newton-Raphson does converge, it converges extremely fast, where the number of accurate digits roughly doubles at each step.

We use a stochastic version of Newton-Raphson in Section 5.2 where we wish to calculate MLEs for comparison purposes and are able to start the algorithm from the known true parameter value. Both $\nabla \ell(\eta)$ and $\nabla^2 \ell(\eta)$ can be approximated using MCMC (Penttinen, 1984).

1.4.3 Markov chain Monte Carlo maximum likelihood

Geyer and Thompson (1992); Corander et al. (1998); Snijders (2002) developed Markov chain Monte Carlo (MCMC) methods to approximate the MLE of an exponential family. Of these, Geyer and Thompson's Markov chain Monte Carlo-maximum likelihood estimator (MCMC-MLE) method appears to have become the standard approach in the network literature (Hunter and Handcock, 2006; Snijders et al., 2006; Hunter et al., 2008a) and is the default algorithm in the `statnet` suite (Handcock et al., 2003) in the R platform for network models. An additional attraction of MCMC-MLE is that it provides a way to give accurate error estimates (Geyer, 1994; Hunter and Handcock, 2006).

The MCMC-MLE approach is theoretically guaranteed to converge to the MLE if it exists. Rather than maximizing the log likelihood (1.4) with respect to η, Geyer and Thompson consider the log of the likelihood ratio $r(\eta, \eta^0)$, where η^0 is fixed at a

known value,

$$\begin{aligned} r(\eta, \eta^0) &= \ell(\eta) - \ell(\eta^0) \\ &= \langle \eta - \eta^0, g(y_{\text{obs}}) \rangle - \log \left[\exp\big(c(\eta) - c(\eta^0)\big) \right]. \end{aligned} \tag{1.8}$$

The value of η that maximizes an approximation of $r(\eta, \eta^0)$ is then a good estimate of the MLE, assuming it exists. In order to avoid evaluating the problematic normalizing constant, this approach equates the ratio of normalizing constants $\exp\left(c(\eta) - c(\eta^0)\right)$ to an expectation using (1.2) as follows:

$$\begin{aligned} \exp\big(c(\eta) - c(\eta^0)\big) &= \frac{\int \exp\big(\langle \eta, g(x) \rangle\big)\, d\mu(x)}{\kappa(\eta^0)} \\ &= \frac{\int \exp\left(\langle \eta - \eta^0, g(x) \rangle + \langle \eta^0, g(x) \rangle\right)\, d\mu(x)}{\kappa(\eta^0)} \\ &= \int \exp\big(\langle \eta - \eta^0, g(x \rangle\big)\, \frac{e^{\langle \eta^0, g(x) \rangle}}{\kappa(\eta^0)}\, d\mu(x) \\ &= \mathrm{E}_{\eta^0} \exp\big(\langle \eta - \eta^0, g(Y) \rangle\big). \end{aligned} \tag{1.9}$$

While we choose η_0 to index a distribution in the same family as the one indexed by η in (1.9), this can be generalized (Geyer, 1996). We keep this restriction here because the MCMC samplers in the `statnet` package only simulate from distributions in the model and so can only use this formula.

By the Markov chain strong law of large numbers (Meyn and Tweedie, 2009, Theorem 17.0.1), this expectation can be approximated by the sample mean for large Monte Carlo sample size,

$$\frac{1}{m} \sum_{i=1}^{m} \exp\big(\langle \eta - \eta^0, g(Y_i) \rangle\big)$$

where $Y_1, \ldots, Y_m$ are draws from the exponential family distribution with parameter

η^0. This sample can be generated using MCMC methods such as the Metropolis-Hastings algorithm (Geyer, forthcoming), which is typically used for ERGMs.

Thus (1.8) can be approximated by

$$\hat{r}_m(\eta, \eta^0) = \langle \eta - \eta^0, g(y_{\text{obs}}) \rangle - \log \left[\frac{1}{m} \sum_{i=1}^{m} \exp\left(\langle \eta - \eta^0, g(Y_i) \rangle \right) \right] \tag{1.10}$$

and for any fixed η

$$\hat{r}_m(\eta, \eta^0) \to r(\eta, \eta^0) \text{ a.s. as } m \to \infty.$$

This holds for all η in any countable set and thus for some dense set. If we call $\hat{\eta}_m$ the maximizer of (1.10) and assume that the MLE $\hat{\eta}_{\text{MLE}}$ exists, Geyer and Thompson show that

$$\hat{\eta}_m \to \hat{\eta}_{\text{MLE}}, \quad \text{a.s.}$$

whenever the Markov chain is ergodic.

This approach has been shown in practice to be sensitive to initial parameter values when used without the trust region methodology recommended in Geyer and Thompson (1992), and the algorithm may require enormous—sometimes infeasibly large—Monte Carlo sample sizes when the starting value η^0 is far from the MLE (Hunter et al., 2008b). In addition, Geyer and Thompson recommend iterating the algorithm several times, where each successive maximizing value will be closer to $\hat{\eta}_{\text{MLE}}$ than the previous if sample sizes are sufficiently large. At the time of this writing, the MCMC-MLE routine in `statnet` uses by default 10,000 Monte Carlo samples spaced 100 samples apart and a maximum of three iterations, using the MPLE as the initial value for η^0 (Handcock et al., 2003). Improvement of the MCMC-MLE approach is an active area of research (Bartz, Liu and Blitzstein, 2009). An extension of this

method by Hummel et al. (2010) is discussed in the next section.

Hunter et al. (2008b) illustrate the practical difficulty associated with a poor initial value in the MCMC-MLE algorithm with Sampson's monastery data set depicted in Figure 1.2. The observed network y_{obs} is directed with 18 actors and 88 ties present out of $18 \cdot 17 = 306$ possible ties. To illustrate the issue, Hunter et al. use the Erdős-Rényi-Gilbert model described in Section 1.3.1 with network statistic $g(y)$ equal to the total number of edges present. As noted earlier, the true MLE is equal to

$$\hat{\eta}_{\text{MLE}} = \text{logit}\left(\frac{g(y_{\text{obs}})}{n(n-1)}\right) = \text{logit}\left(\frac{88}{306}\right) = -0.9072.$$

When η^0 is chosen to be 1, however, Hunter et al. show that it is extremely difficult for the algorithm to attain the MLE in a single iteration of the MCMC-MLE algorithm. For this value of η, the model dictates that each of the 306 possible edges occur independently with probability $p = \frac{1}{1+e^{-\eta}} = \frac{1}{1+e^{-1}} = 0.731$. This is a very high probability relative to the sparsity of relations in the observed data set, which suggest a much smaller probability of tie formation of $88/306 = 0.288$. In fact, the probability of obtaining fewer than 88 ties for the $\eta = 1$ model is nearly zero at 2.3×10^{-59}, calculated using a binomial distribution with $n = 306$, $p = 0.731$.

The MCMC-MLE algorithm maximizes the approximated log likelihood ratio (1.10). However, the first derivative of $\hat{r}(\eta, \eta^0)$ with respect to η shows that if the MCMC sampler is unable to generate any $g(Y) < g(y_{\text{obs}})$, the derivative of (1.10) will be strictly negative, resulting in $\hat{r}(\eta, \eta^0)$ with no maximum. This is depicted by the dotted-line in Figure 1.4 (top). The problem is not present when the initial value is close to the true MLE, such as when $\eta^0 = -1$; the corresponding $r(\eta, \eta^0)$ and $\hat{r}(\eta, \eta^0)$ functions are depicted in Figure 1.4 (bottom). With the default three iterations of MCMC-MLE in the `statnet` package, the algorithm starting with $\eta^0 = 1$ is only able to obtain an estimate for η as small as $\eta = -0.364$. With 10 iterations, it does in fact

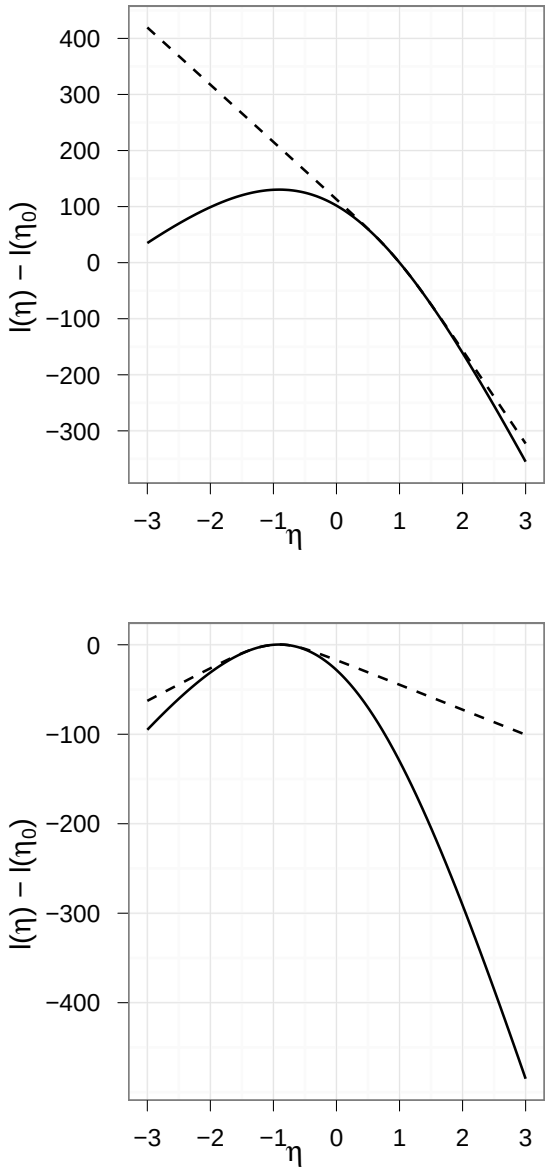

Figure 1.4: Log likelihood ratio approximations in Sampson's monastery example. Top: $\eta^0 = 1$, Bottom: $\eta^0 = -1$. Solid lines are exact log likelihood ratios $\ell(\eta) - \ell(\eta^0)$, dotted lines are the approximation by (1.10) using 100,000 MCMC samples. MLE is at -0.9072. Data for plots are produced using code accompanying Hummel et al. (2010).

arrive at the MLE of -0.907.

1.4.4 Steplength MCMC-MLE

Hummel et al. (2010) expanded the range for which iterated MCMC-MLE will converge by making incremental updates in η towards $\hat{\eta}_{\text{MLE}}$ possible at each iteration. As noted in the previous example with Sampson's monastery data, $\hat{r}(\eta, \eta^0)$ will have no maximum if $g(y_{\text{obs}})$ is outside the range of MCMC samples. Hummel et al.'s approach involves replacing $g(y_{\text{obs}})$ with a value $\hat{\xi}_1$ close to $g(y_{\text{obs}})$ but within the range of the current model's MCMC samples. This makes it possible to maximize $\hat{r}(\eta, \eta^0)$ and update η^0, with the resulting η necessarily closer to $\hat{\eta}_{\text{MLE}}$.

The approach is effective for Sampson's monastery example discussed, and was also applied to the *E. coli* biological network data of Shen-Orr et al. (2002) when starting from the MPLE. Care is needed in finding a $\hat{\xi}_1$ that is inside the range of the current model. In addition, if η_0 is very far from $\hat{\eta}_{\text{MLE}}$, it is not clear that the steps taken by this approach will be large enough to traverse the distance between η^0 and $\hat{\eta}_{\text{MLE}}$ in any reasonable amount of time.

1.4.5 Stochastic approximation

Variations on the Robbins-Monro *stochastic approximation* (SA) algorithm (Robbins and Monro, 1951) have been applied to find the MLE in similar contexts: Younes (1988, 1989); Moyeed and Baddeley (1991); Gu and Zhu (2001) applied MCMC stochastic approximation to spatial models and Snijders (2002) to ERGMs. SA procedures for finding the MLE of a parameter η generate iterated estimates η_k to find the root of a gradient function $h(\eta)$:

$$\eta_{k+1} = \eta_k + \alpha_k U_k, \tag{1.11}$$

where α_k is a step size and is typically a member of a decreasing sequence of positive numbers, and U_k is a random variable from the distribution specified by η_k that noisily estimates the gradient function $h(\eta_k)$.

Restrictive conditions are required of α_k and U_k to establish convergence of the sequence η_k. In Robbins-Monro SA (Robbins and Monro, 1951), the step size α_k must be a sequence of positive constants that satisfies

$$\sum \alpha_k^2 < \infty$$

for which the choice of

$$\alpha_k = \frac{A}{B+k} \tag{1.12}$$

is commonly used, where A and B are constants that must be specified by the user. This specification requires experimentation and care: there can be significant variation in performance depending on choice of these constants. Some recent research show that sequences that go to 0 slower than $1/k$ can result in an improved rate of convergence, where rate of convergence is measured by the asymptotic covariance of the normalized estimates about their limit point (Kushner and Yin, 1997, Chapter 11).

The conditions on U_k are more restrictive. Popular approaches include constraining the sequence of estimators η_k to a compact set specified *a priori*, or assuming that the noise component of U_k be a martingale difference sequence. As commonly observed (Chen, 2002; Andrieu, Moulines and Priouret, 2005; Liang, 2010) these may be difficult to satisfy in practice. See Andrieu et al. (2005); Liang (2010) for recent developments that impose less restrictive conditions using truncated updates.

An issue for any recursive search algorithm is the choice of starting point. It is often the case that algorithms are good at finding the MLE when the starting point

is close to it, but of course the location of the MLE is unknown. For any exponential family with bounded support, Fisher information becomes singular as the canonical parameter η goes to ∞ (Rinaldo, Fienberg and Zhou, 2009). Hence methods which rely on the Fisher information matrix may fail when the starting point for η is far from the MLE (Younes, 1989; Gu and Zhu, 2001). Of course, one may try different starting points until a "good" one is found, but this comes with no theoretical guarantees and can be cumbersome in practice.

1.4.6 Summary of parameter estimation methods

Tables 1.2 and 1.3 summarize the advantages and disadvantages of the exact and MCMC parameter estimations discussed. All methods assume that the MLE exists before the search algorithm is applied and that the log likelihood is strictly concave.

Table 1.2: Comparison of exact parameter estimation methods. MPLE=Maximum pseudolikelihood estimator, Newton=Newton-Raphson.

Method	Appeal	Drawbacks
MPLE	• Simplicity • Speed, via logistic regression	• Maximizes pseudolikelihood, not likelihood
Newton	• Converges rapidly when it converges	• Highly sensitive to starting point • Requires second derivative matrix and its inverse

Table 1.3: Comparison of MCMC parameter estimation methods. MCMC Newton=MCMC Newton-Raphson, MCMC-MLE=Markov chain Monte Carlo-maximum likelihood estimator, step MCMC-MLE=steplength MCMC-MLE, MCMC SA=stochastic approximation.

Method	Appeal	Drawbacks
MCMC Newton	• Converges rapidly when it converges	• Requires calculation of second derivative matrix and its inverse, which typically requires staring point very close to solution
MCMC-MLE	• Theoretically guaranteed to converge to MLE	• Highly sensitive to starting point • May require enormous MC sample sizes • May require several iterations
step MCMC-MLE	• Theoretically guaranteed to converge to MLE • Increased range	• Sensitive to starting point • Requires setup expertise • May require several iterations
MCMC SA	• Simple updates • Theoretically guaranteed to converge	• Requires trial-and-error calibration • May converge too slowly to be practical • Convergence conditions not easily implemented in practice

1.5 Non-existent MLEs in exponential families

Parameter estimation via maximum likelihood for exponential family models with complex dependence—already a challenging problem because the likelihood function may be computationally infeasible—is further obfuscated by the possibility that the MLE may not exist. In such a case, the strictly concave likelihood function is increasing in some direction of the parameter space, called a *direction of recession*, so that the MLE is actually off "at infinity". The theoretical background for this situation has been understood for many years (Barndorff-Nielsen, 1978; Brown, 1986): MLE existence is a geometric problem relating to properties of the *convex support* of the model, which is the smallest closed convex set that the contains the natural statistic (Geyer, 2009a). The MLE does not exist in the conventional sense if the observed

data is in the relative boundary of this convex support (see Section 2.4). When this occurs, it may exist in the *Barndorff-Nielsen completion* of the family. Before this dissertation, there have been no computationally convenient approaches to handling this issue for social network models and other discrete exponential family models with complex dependence.

Geyer and Thompson (1992) separated the search for the MLE of an exponential family model with complex dependence into a two-phase algorithm, where phase I determines whether or not an MLE exists in the conventional sense via linear programming, and phase II finds the MLE when it exists. Geyer and Thompson's MCMC-MLE method discussed in Section 1.4.3 corresponds to phase II of this approach. In the special case of generalized linear models including logistic regression and contingency tables, Geyer (2009a) outlines a methodology to detect non-existent MLEs and construct one-sided confidence intervals for the natural parameters. The approach applies linear programming to the polyhedral convex support of the model to determine the *face* of the convex support in which the observed data lies. This face in turn defines the convex support for a new exponential family for which the MLE does exist and can be found using conventional methods (see Section 2.4.3). The linear programming functionality has been implemented in the R package `rcdd` (Geyer and Meeden, 2009).

Handcock (2003); Rinaldo, Fienberg and Zhou (2009) focus on problematic instances of parameter estimation for ERGMs, which are loosely referred to as *degeneracy*. Strictly speaking, a distribution in an exponential family is degenerate when it concentrates the distribution of the natural statistic on a hyperplane. If the family is minimal, this occurs if and only if the MLE does not exist in the conventional sense. This in turn means that the observed data lies on the relative boundary of the convex support. *Near* degenerate means that the observed data lies nearly on this relative boundary. However, the terms degeneracy and near degeneracy are now invoked if

there are undesirable behaviors of the fitted model or in the model fitting process itself (Handcock, 2003; Goodreau, 2007; Robins et al., 2007b; Goodreau et al., 2008). We will only use the terms for their original meanings here.

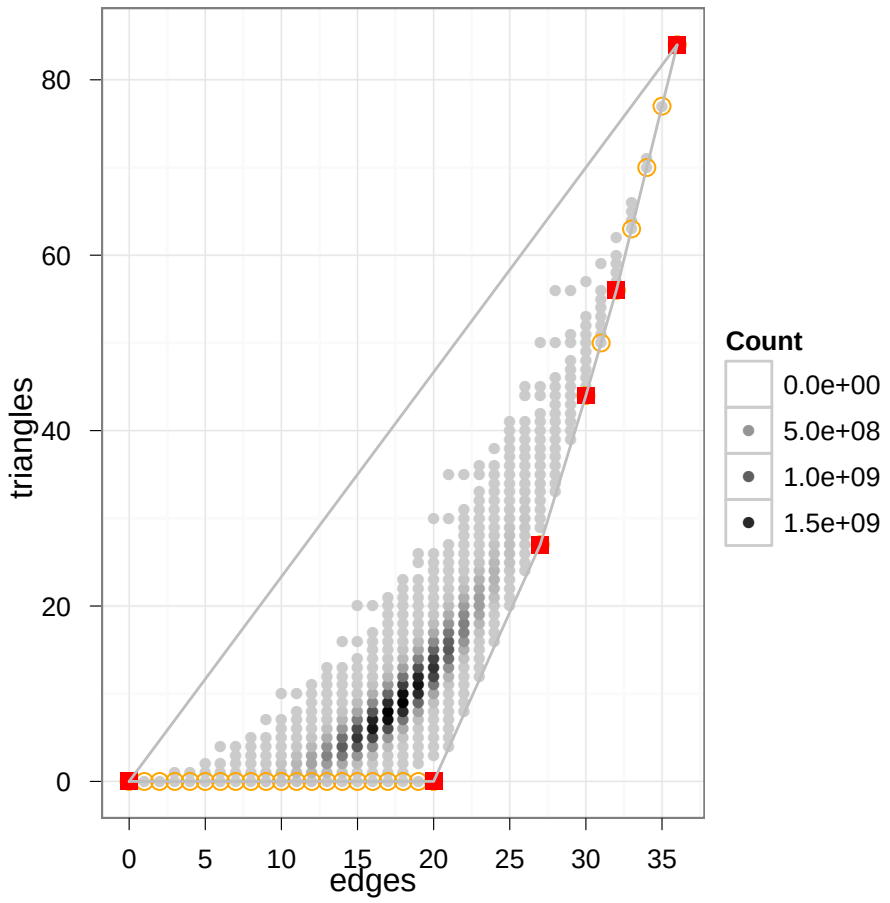

Figure 1.5: Sample space of two-dimensional statistic for 9-node network model. Rinaldo et al. (2009) focused on edge and triangle statistics. The shading of a point corresponds to the frequency of graphs with that network statistic. The convex support is the gray, sail-shaped polytope. Points on the boundary have an outlined circle and extreme points are square.

To better understand ERGM pathologies, Handcock; Rinaldo et al. study small networks with 7 and 9 nodes. At these sizes, it is still possible to explicitly evaluate the

likelihood function (1.1) and thus find MLEs using standard numerical optimization routines. As in (Geyer, 2009a), the emphasis is on the geometric properties of the polyhedral convex support of the model; the convex support for the 9-node ERGM with a two-dimensional network statistic of edges and triangles studied by Rinaldo et al. is depicted in Figure 1.5. Rinaldo et al. showed that the normal cones (see Section 2.2) of this support explain which portions of the natural parameter space are more likely to correspond to degenerate and nearly degenerate models. Some of the perceived pathologies then, are fully explained by exponential family theory and are not pathologies at all.

However, it is possible for a well-fit model to poorly describe or resemble the observed data (Handcock, 2003; Hunter et al., 2008b). This may include cases of multimodality of the distribution of the natural statistic and extreme sensitivity to parameter values. In the multimodality case, the observed data is the average of the distribution, but there are multiple modes, none which resemble the observed data. If the distribution is in addition extremely sensitive to parameter values, it is possible for a very small change in the parameter value to lead to a very different distribution, perhaps going from multimodal to unimodal. When these behaviors are too problematic, the model itself should be changed. ERGMs give tremendous flexibility in model specification but with no guarantees that a model will behave exactly as desired. Hunter, Goodreau and Handcock (2008a) devise graphical tests for evaluating the goodness of fit of ERGMs. Their tools, available in the `statnet` package, compare characteristics of the distribution of networks simulated from the fitted model to the observed data, which can then be utilized to help choose between network statistics to include in a model. The issues described here are not with the parameter estimation method itself and thus are not the focus of this dissertation.

1.6 Research overview

In this dissertation, we propose a simple and practical line search algorithm which accomplishes the following objectives:

1. The algorithm converges to the MLE of any regular exponential family when the MLE exists. It does so while
   (a) avoiding the hassle of trial-and-error in algorithm calibration
   (b) being insensitive to starting point and can thus be started from "long range".
2. When the MLE does not exist, the algorithm finds the MLE in the Barndorff-Nielsen completion and the accompanying direction of recession necessary to construct one-sided confidence intervals in the manner of Geyer (2009a).

In short, our proposed algorithm, outlined in the next section, addresses many of the drawbacks of the methods described in Section 1.3 as well as providing a general methodology for dealing with non-existent MLEs. We are not aware of any other general approaches for handling non-existent MLEs for ERGMs.

We prove convergence for our algorithm in the case where the MLE exists and the first derivative of the log likelihood can be calculated exactly. When the first derivative cannot be calculated exactly, it has a particularly convenient form that is easily estimable with MCMC, making the algorithm still useful in application.

The appeal of this algorithm is its usability: no trial and error is needed. Experimentation with multiple starting points or tuning parameters is not necessary and no unrealistic *a priori* information about the problem need be specified. It is currently used in the `aster2` contributed R package (Geyer, 2010) as the safeguard for steepest ascent and Newton-Raphson iterations in finding the MLE for aster models.

Our algorithm utilizes first derivative information only, evaluating neither the likelihood function itself nor derivatives of higher order than first. The desire to avoid calculation of higher order derivatives in our algorithm is motivated not just by computational considerations, but also by how much useful information can be extracted from them. If the current value for η is far from the MLE, the Fisher information matrix may be near-singular and algorithms like (unsafeguarded) Newton-Raphson algorithm may fail. For this reason, the best use of our algorithm may be from "long range," filling a gap in the MLE estimation toolbox. It may be expedient to switch to another algorithm like Newton-Raphson or MCMC-MLE after significant progress is made and the Fisher information matrix becomes useful.

In the case where the MLE does not exist in the conventional sense, it is actually "at infinity" in some direction of the parameter space. Our algorithm returns the information necessary to construct one-sided confidence intervals for how close η is to infinity. Previous work in this area with ERGMs utilized full knowledge of the convex support (Handcock, 2003; Rinaldo et al., 2009), something not generally available for ERGMs, and provided no such confidence intervals.

1.6.1 Long range search algorithm overview

Our algorithm generates iterated estimates η_k of the MLE with the update

$$\eta_{k+1} = \eta_k + \alpha_k p_k \tag{1.13}$$

where α_k is a *step size* and p_k is a *search direction* and is restricted to be an ascent direction of the log likelihood. Despite the visual similarity between (1.11) and (1.13), the line search approach treats the search direction p_k in (1.13) as constant whereas in SA the corresponding U_k in (1.11) is random. Furthermore, line search algorithms have more restrictions on the step size α_k. The step size conditions in the classical

gradient ascent algorithm, which is the basis for our algorithm, force a sufficiently large increase in the objective function at every step, guaranteeing convergence to the global maximum, if it exists.

Theorem 3.2 in Nocedal and Wright (1999) implies the global convergence of the steepest ascent algorithm for a continuously differentiable function, $\ell(\eta)$. It requires the step length α_k to satisfy the Wolfe conditions for *sufficient increase* and *curvature*:

$$\begin{aligned} \ell(\eta_k + \alpha_k \eta_k) &\geq \ell(\eta_k) + c_1 \alpha_k \nabla \ell(\eta_k)^T p_k \\ \nabla \ell(\eta_k + \alpha_k p_k)^T p_k &\leq c_2 \nabla \ell(\eta_k)^T p_k \end{aligned} \tag{1.14}$$

where ∇ is the gradient operator and $0 < c_1 < c_2 < 1$. Variations of these conditions exist in the numerical optimization literature (Nocedal and Wright (1999, Chapter 3), Sun and Yuan (2006, Chapter 2)), but all require evaluating the objective function.

Exponential families we consider are an unusual case in optimization in that the objective function is harder to compute than its derivatives and hence not previously considered by optimization theorists. In our algorithm, we replace (1.14) with a single modified curvature condition:

$$0 \leq \nabla \ell(\eta_k + \alpha_k p_k)^T p_k \leq c \nabla \ell(\eta_k)^T p_k \tag{1.15}$$

for some $0 < c < 1$. This replacement is possible while still guaranteeing sufficient increase and convergence to the MLE (if it exists) because we have the additional property that the exponential family log likelihood function we consider is strictly concave. The restrictions on the step size α_k along a particular direction p_k are depicted in Figure 1.6.

Our algorithm can be outlined as follows. Let $\|\cdot\|$ denote the Euclidean norm function, and ϵ a small value greater than 0.

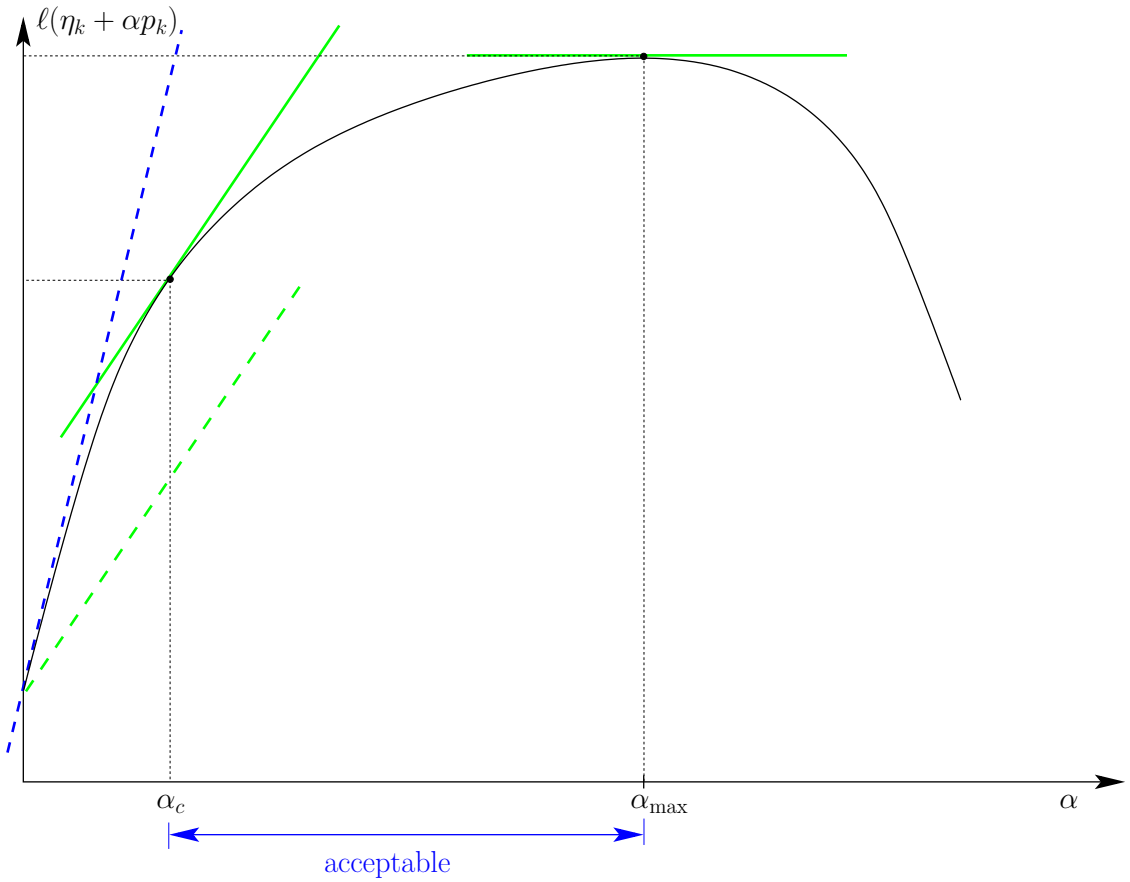

Figure 1.6: Acceptable region for step size along a search direction according to modified curvature condition (1.15). The step sizes α_c and $\alpha_{\max}$ correspond to values of $\nabla\ell(\eta_k + \alpha p_k)^T p_k$ equaling $c\nabla\ell(\eta_k)^T p_k$ and 0, respectively. The condition ensures sufficient increase in the log likelihood along the search direction p_k.

Get an initial value, η_0.
Set $k = 0$.
Calculate $\nabla\ell(\eta_k)$, the direction of steepest ascent.
Set $p_k = \nabla\ell(\eta_k)$.
while $\|\nabla\ell(\eta_k)\| > \epsilon$
Find a step size α_k that satisfies the modified curvature condition

$$0 \leq \nabla\ell(\eta_k + \alpha_k p_k)^T p_k \leq c\nabla\ell(\eta_k)^T p_k$$

for some $0 < c < 1$.
$\eta_{k+1} = \eta_k + \alpha_k p_k$.

Calculate $\nabla \ell(\eta_{k+1})$.
Find the new search direction p_{k+1}, which must be an ascent direction.
$k = k + 1$.
end while

The exit condition from this algorithm is that $\|\nabla \ell(\eta_k)\|$ is close to zero. As noted earlier, the log likelihood $\ell(\eta)$ is a strictly concave function; when the global maximum $\hat{\eta}_{\text{MLE}}$ exists, $\|\nabla \ell(\hat{\eta}_{\text{MLE}})\| = 0$. When the MLE does not exist in the conventional sense, $\ell(\eta)$ does not bend back down as depicted in Figure 1.6, but instead continues to increase along some direction. Hence the maximizer can be thought of as "at infinity". The log likelihood function is, however, still bounded by the log likelihood of the *limiting conditional model*, which is another exponential family defined on a support for which the MLE is guaranteed to exist, as discussed in Section 2.4.3. As part of the process to construct one-sided confidence intervals for how close η is to infinity, our algorithm looks to find the maximizer for this new model. Some modification of this algorithm is necessary, which we discuss in Chapter 4. In particular, the objective function $\ell(\eta)$ that is being maximized must switch to that of the new model. However, the overall structure of the algorithm is the same.

Chapter 2

Background Theory

All of the parameter estimation issues discussed thus far for ERGMs are relevant to the larger class of discrete state space exponential family models. The latter are commonly used to model phenomena with dependent structure, where the outcomes of the response variable of interest are in fact dependent on one another. For example, the Ising model (Ising, 1925; Potts, 1952) is an exponential family model that has been used to model ferromagnetism, where neighboring pixels (representing atoms in a crystal lattice) are more likely to have the same spin orientation. Other examples of phenomena with dependent structure modeled with exponential families include plant ecology (Besag, 1974, 1975), DNA fingerprint data (Geyer and Thompson, 1992) and the lifetime fitness of plants (Shaw, Geyer, Wagenius, Hangelbroek and Etterson, 2008).

Although motivated by ERGMs, the methods we propose are rooted in fundamental exponential family theory and applicable to any exponential family model. Thus the theory presented in this section is from the perspective of this more general setting; in fact, only Theorem 2.8 and Corollary 2.5 require the additional assumption of a finite state space.

2.1 Exponential family theory

Much of the fundamental background for exponential families has already been covered in Section 1.2: our interest is in the properties of regular exponential families on a finite sample space $\mathcal{Y}$ with log likelihood (1.4). As noted earlier, when the sample space $\mathcal{Y}$ is even moderately large, the cumulant function $c(\eta)$ involves a summation that may be prohibitively expensive to evaluate.

A useful property of all exponential families (Lehmann and Casella, 1998, p. 27) when η is in the interior of Ξ is that

$$\begin{aligned} \mathrm{E}_\eta(g(Y)) &= \nabla c(\eta) \\ \mathrm{Var}_\eta(g(Y)) &= \nabla^2 c(\eta). \end{aligned}$$

Thus we can express first and second derivatives of the log likelihood (1.4) and Fisher information $I(\eta)$ as

$$\nabla \ell(\eta) = g(y) - \mathrm{E}_\eta\, g(Y) \tag{2.1}$$

$$\nabla^2 \ell(\eta) = -\mathrm{Var}_\eta\, g(Y) \tag{2.2}$$

$$\mathrm{I}(\eta) = -\mathrm{E}_\eta\, \nabla^2 \ell(\eta) = \mathrm{Var}_\eta\, g(Y) \tag{2.3}$$

and thereby avoid evaluation of the problematic cumulant function c.

Strict concavity of the log likelihood function is assured by (2.2) being strictly positive definite, which in turn is assured by the natural statistic not being concentrated on a hyperplane. The global maximum, if it is exists, is attained when η is such that $\nabla \ell(\eta) = 0$, or, using (2.1), by setting "expected equal to observed"

$$\mathrm{E}_\eta\, g(Y) = g(y_{\mathrm{obs}}). \tag{2.4}$$

2.2 Convex analysis

The issue of MLE existence in the conventional sense in an exponential family is closely tied to the geometric properties of the convex support of the model (Geyer, 2009a; Rinaldo et al., 2009; Barndorff-Nielsen, 1978, Chapter 9). We describe the relevant theory from convex analysis as it pertains to the case of discrete state space exponential families.

Define the *convex hull* of a set of points V, denoted $\operatorname{con} V$, to be

$$\left\{ \sum_i \alpha_i v_i : \alpha_i \in \mathbb{R}, \alpha_i \geq 0, \sum_i \alpha_i = 1 \right\},$$

and the *positive hull* of a set of points V, denoted $\operatorname{pos} V$, to be

$$\left\{ \sum_i \alpha_i v_i : \alpha_i \in \mathbb{R}, \alpha_i \geq 0 \right\}.$$

A *convex polytope* C is the convex hull of a finite set of points V. By the Minkowski-Weyl theorem (Rockafellar, 1970, Theorem 19.1), this convex set can be represented equivalently as the intersection of a finite collection of closed half-spaces. These two representations of a convex polytope are referred to as the *V-representation* and *H-representation*, respectively. The H-representation can be expressed as the solution set of a finite set of linear equations and inequalities,

$$C = \{x : Ax \leq b\},$$

where A is a matrix and b a vector (each linear equation expressed as a pair of inequalities).

The *relative interior* of a convex set C, denoted $\operatorname{rint} C$, is the interior relative to its affine hull. The concept is motivated by the idea that a triangle in $\mathbb{R}^3$ has empty

interior since no three-dimensional ball lies in the triangle. The *closure* of a set C, denoted $\operatorname{cl} C$, is equal to

$$\operatorname{cl} C = \bigcap \{C + \epsilon B \mid \epsilon > 0\}$$

where B is a Euclidean ball (Rockafellar, 1970, Section 6). The *relative boundary* of a set C, denoted $\operatorname{rbd} C$, is the set difference $\operatorname{cl} C \setminus \operatorname{rint} C$.

A nonempty *face* F of a convex polytope C is a convex subset of C such that every line segment in C with a relative interior point in F has both end points in F (Rockafellar, 1970, Section 18). It is itself a convex polytope. A *proper* face is a face that is not the empty set or C, and *facets* are proper faces of the highest dimension.

For a polyhedral convex C, every face F is *exposed* (Rockafellar, 1970, Section 18), meaning there exists a vector δ such that

$$F = \{x \in C : \langle x, \delta \rangle = \sup_{y \in C} \langle y, \delta \rangle\}.$$

Smooth convex sets like the convex hull of a torus, do not have every face exposed; here we deal only with convex polyhedral sets, so all the faces we encounter are exposed.

The *tangent cone* of a polyhedral convex set C at a point $x \in C$ is

$$T_C(x) = \{s(w - x) : w \in C \text{ and } s \geq 0\}.$$

The *normal cone* of a convex set C in $\mathbb{R}^d$ at a point $x \in C$ is

$$N_C(x) = \{\delta \in \mathbb{R}^d : \langle w - x, \delta \rangle \leq 0 \text{ for all } w \in C\}.$$

Tangent and normal cones are *polars* of each other, that is, each determines the

other. The normal cone at x can be defined in terms of the tangent cone at x by

$$N_C(x) = \{w \in \mathbb{R}^d : \langle w, v \rangle \leq 0 \text{ for all } v \in T_C(x)\}$$

and the tangent cone at x in terms of the normal cone at x by

$$T_C(x) = \{v \in \mathbb{R}^d : \langle w, v \rangle \leq 0 \text{ for all } w \in N_C(x)\}.$$

2.3 Mean value parameterization

An alternative parameterization of an exponential family is the mean value parameterization. For a natural parameter η, we can define the mean parameter μ such that

$$\mu = \mathrm{E}_\eta \, g(Y).$$

This maps any $\eta \in \Xi$ for which the expectation exists (and it does exist for all $\eta \in \operatorname{rint} \Xi$) to a point in the relative interior of the convex support. Thus mean value parameters are more easily interpretable quantities, since they reside in the same space as the natural statistics (Handcock, 2003; Rinaldo et al., 2009). In particular, $\hat{\mu}_{\text{MLE}} = g(y_{\text{obs}})$ is the MLE in the mean value parameterization by (2.4). A point on the relative boundary of the convex support cannot be the mean of any distribution in the family. The models with $g(y_{\text{obs}})$ on this relative boundary are the degenerate models mentioned in Section 1.5; thus even if the MLE does not exist in the natural parameterization it does still exist in this mean value parameterization.

2.4 MLE existence in exponential families

We now present two equivalent approaches to determining the existence of an MLE in the conventional sense.

2.4.1 Approach of Barndorff-Nielsen (1978)

The well-known condition for the existence of the MLE relates the relative location of the observed statistic to the boundaries of the convex support and is formally stated as follows:

Theorem 2.1 (Corollary 9.6 in Barndorff-Nielsen (1978))**.** *For a regular exponential family with convex support C and observed statistic $g(y_{obs})$, the MLE exists in the conventional sense if and only if $g(y_{obs}) \in \operatorname{rint}(C)$ and is unique if and only if $\operatorname{int}(C)$ is nonempty, which happens if and only if $\operatorname{rint}(C) = \operatorname{int}(C)$.*

While the above theorem is concise, we note that in order to apply this result, the geometry of C must be known.

2.4.2 Approach of Geyer (2009a)

Geyer (1990, 2009a) related MLE existence to not only the relative boundary of the convex support, but also to behavior of the log likelihood function along certain directions.

We retrace the steps leading to this result, and present simpler proofs where possible:

Theorem 2.2 (Theorem 2.2 in Geyer (1990))**.** *For a full exponential family with*

- *density function $f_\eta(y)$ as in* (1.1)*,*
- *natural parameter space Ξ as in* (1.3)*,*

- *natural statistic* $g(Y)$,
- *convex support* C,
- *a non-zero direction* δ,
- $\sigma_C(\delta) = \sup_{g(y)\in C}\langle g(y), \delta\rangle$,
- $H = \{\, w : \langle w, \delta\rangle = \sigma_C(\delta)\,\}$,
- $P_\eta(g(Y) \in H) > 0$ *for some distribution in the family.*

Then

$$e^{c(\eta+s\delta)-bs} \to \begin{cases} 0 & b > \sigma_c(\delta) \\ e^{c(\eta)} P_\eta(g(Y)\in H) & b = \sigma_c(\delta) \\ +\infty & b < \sigma_c(\delta) \end{cases} \qquad \text{as } s \to +\infty.$$

Here, H is the supporting hyperplane to the set C with normal vector δ.

Proof. **Case:** $b = \sigma_C(\delta)$**.**

Starting with (1.2),

$$e^{c(\eta)} = \kappa(\eta) = \int e^{\langle \eta, g(y)\rangle}\, d\mu(y),$$

so that

$$\begin{aligned} e^{c(\eta+s\delta)-bs} &= \int e^{\langle \eta+s\delta, g(y)\rangle - bs}\, d\mu(y). \\ &= \int e^{\langle \eta, g(y)\rangle + s[\langle g(y),\delta\rangle - b]}\, d\mu(y). \end{aligned}$$

Multiplying by $\frac{f_\eta(y)}{f_\eta(y)}$,

$$\begin{aligned} e^{c(\eta+s\delta)-bs} &= \int e^{\langle \eta, g(y)\rangle + s[\langle g(y),\delta\rangle - b]} \frac{f_\eta(y)}{e^{\langle \eta, g(y)\rangle - c(\eta)}}\, d\mu(y) \\ &= \int e^{s[\langle g(y),\delta\rangle - b] + c(\eta)} f_\eta(y)\, d\mu(y) \\ &= \mathrm{E}_\eta\, e^{s[\langle g(Y),\delta\rangle - b] + c(\eta)}. \end{aligned}$$

The monotone convergence theorem can be applied to reverse the order of the limit and expectation for monotone sequences of random variables. For $\langle g(Y), \delta\rangle \leq b$, we have a monotonically decreasing sequence of random variables and for $\langle g(Y), \delta\rangle > b$, the sequence is increasing. Thus,

$$\lim_{s\to\infty} \mathrm{E}_\eta\, e^{s[\langle g(Y),\delta\rangle - b] + c(\eta)} = E_\eta \lim_{s\to\infty} e^{s[\langle g(Y),\delta\rangle - b] + c(\eta)}.$$

Ignoring the expectation and examining just the limit component of the above,

$$\lim_{s\to\infty} e^{s[\langle g(Y),\delta\rangle - b] + c(\eta)} = \begin{cases} 0 & \langle g(Y), \delta\rangle < b \\ e^{c(\eta)} & \langle g(Y), \delta\rangle = b \\ +\infty & \langle g(Y), \delta\rangle > b. \end{cases}$$

In this first case we consider, $b = \sigma_C(\delta) = \sup_{g(y)\in C}\langle g(y), \delta\rangle$, so $\langle g(Y), \delta\rangle$ can never be greater than b and thus the $+\infty$ outcome above is not possible. We can rewrite the above result succinctly as

$$\lim_{s\to\infty} e^{s[\langle g(Y),\delta\rangle - b] + c(\eta)} = I(\langle g(Y), \delta\rangle = b) e^{c(\eta)} = I(g(Y) \in H) e^{c(\eta)}.$$

Then returning to the original expression of interest,

$$\lim_{s\to\infty} e^{c(\eta+s\delta)-bs} = \lim_{s\to\infty} \mathrm{E}_\eta\, e^{s[\langle g(Y),\delta\rangle - b]+c(\eta)} = e^{c(\eta)} P(g(Y)\in H).$$

Case: $b > \sigma_C(\delta)$.

$$\begin{aligned}
\lim_{s\to\infty} e^{c(\eta+s\delta)-bs} &= \lim_{s\to\infty} e^{c(\eta+s\delta)-\sigma_C(\delta)s+\sigma_C(\delta)s-bs} \\
&= \left(\lim_{s\to\infty} e^{c(\eta+s\delta)-\sigma_C(\delta)s}\right)\left(\lim_{s\to\infty} e^{s[\sigma_C(\delta)-b]}\right) \\
&= \left(e^{c(\eta)} P(g(Y)\in H)\right)\cdot 0 = 0.
\end{aligned}$$

Case: $b < \sigma_C(\delta)$.

$$\begin{aligned}
\lim_{s\to\infty} e^{c(\eta+s\delta)-bs} &= \lim_{s\to\infty} e^{c(\eta+s\delta)-\sigma_C(\delta)s+\sigma_C(\delta)s-bs} \\
&= \left(\lim_{s\to\infty} e^{c(\eta+s\delta)-\sigma_C(\delta)s}\right)\left(\lim_{s\to\infty} e^{s[\sigma_C(\delta)-b]}\right) \\
&= \left(e^{c(\eta)} P(g(Y)\in H)\right)\cdot(+\infty) = +\infty,
\end{aligned}$$

since $P_\eta(g(Y)\in H) > 0$ by assumption. □

We now define directions of constancy and recession in terms of Theorems 1 and 2 of Geyer (2009a) which we state without complete proofs.

Theorem 2.3 (Direction of Constancy: Theorem 1 in Geyer (2009a))**.** *For a full exponential family with*

- *log likelihood function* $\ell(\eta)$ *as in* (1.4)*,*
- *natural parameter space* Ξ *as in* (1.3)*,*
- *natural statistic* $g(Y)$*,*
- *observed data* y_{obs}

the following are equivalent:

1. *For all* $\eta \in \Xi$*, the function* $s \mapsto \ell(\eta + s\delta)$ *is constant on* $\mathbb{R}$ *(Geyer, 2009a, Theorem 1(b)).*

2. *The parameter values* η *and* $\eta + s\delta$ *correspond to the same probability distribution for all* $\eta \in \Xi$ *and all real* s *(Geyer, 2009a, Theorem 1(d)).*

3. $\langle g(Y) - g(y_{obs}), \delta \rangle = 0$ *almost surely for all distributions in the family (Geyer, 2009a, Theorem 1(f)).*

4. $\delta \in N_C(g(y_{obs}))$ *and* $-\delta \in N_C(g(y_{obs}))$ *(Geyer, 2009a, Theorem 1(g)).*

5. $\langle w, \delta \rangle = 0$ *for all* $w \in T_C(g(y_{obs}))$ *(Geyer, 2009a, Theorem 1(h)).*

Any vector δ that satisfies any of the conditions above is called a *direction of constancy* of the log likelihood. The set of all directions of constancy is called the *constancy space* of the log likelihood, which is a vector subspace.

Theorem 2.4 (Direction of Recession: Theorem 3 in Geyer (2009a))**.** *For a full exponential family with the same setting as Theorem 2.3, the following are equivalent:*

1. *For all* $\eta \in \Xi$*, the function* $s \mapsto \ell(\eta + s\delta)$ *is nondecreasing on* $\mathbb{R}$ *(Geyer, 2009a, Theorem 3(b)).*

2. $\langle g(Y) - g(y_{obs}), \delta \rangle \leq 0$ *almost surely for all distributions in the family. (Geyer, 2009a, Theorem 3(d)).*

3. $\delta \in N_C(g(y_{obs}))$ *(Geyer, 2009a, Theorem 3(e)).*

4. $\langle w, \delta \rangle \leq 0$ *for all* $w \in T_C(g(y_{obs}))$ *(Geyer, 2009a, Theorem 3(f)).*

Any vector δ that satisfies any of the conditions above is called a *direction of recession* of the log likelihood. Every direction of constancy is a direction of recession.

Geyer uses Corollary 2.4.1 in Geyer (1990) to prove the equivalence of conditions 1 and 2 of Theorem 2.4. Here we present a more accessible proof without the use of this corollary. It is the equivalence of these two conditions that relates the behavior of the log likelihood function to the convex support of the model.

Proof. **From** $2 \to 1$: Assume $\langle g(Y) - g(y_{\text{obs}}), \delta \rangle \leq 0$ almost surely. Take expectations of both sides with respect to the distribution indexed by the parameter value $\eta + s\delta$:

$$\langle \mathrm{E}_{\eta+s\delta}\, g(Y) - g(y_{\text{obs}}), \delta \rangle \leq 0. \tag{2.5}$$

Now, taking the derivative of $\ell(\eta + s\delta)$ with respect to s,

$$\begin{aligned} \frac{d\ell(\eta + s\delta)}{ds} &= \frac{d}{ds} \left(\langle g(y_{\text{obs}}), \eta + s\delta \rangle - c(\eta + s\delta) \right) \\ &= \langle g(y_{\text{obs}}), \delta \rangle - \langle \mathrm{E}_{\eta+s\delta}\, g(Y), \delta \rangle \\ &= -\langle \mathrm{E}_{\eta+s\delta}\, g(Y) - g(y_{\text{obs}}), \delta \rangle, \end{aligned}$$

which is greater than or equal 0 by (2.5). Thus $\ell(\eta + s\delta)$ is a non-decreasing function of s.

From $1 \to 2$:

$$\begin{aligned} \ell(\eta + s\delta) &= \langle g(y_{\text{obs}}), \eta + s\delta \rangle - c(\eta + s\delta) \\ &= \langle g(y_{\text{obs}}), \eta \rangle + s\langle g(y_{\text{obs}}), \delta \rangle - bs + bs - c(\eta + s\delta) \\ &= \langle g(y_{\text{obs}}), \eta \rangle + s[\langle g(y_{\text{obs}}), \delta \rangle - b] - \log e^{c(\eta+s\delta)-bs}. \end{aligned} \tag{2.6}$$

By assumption, $\ell(\eta + s\delta)$ is a non-decreasing function of s. Then for $\ell(\eta) > -\infty$, $\ell(\eta + s\delta) \geq \ell(\eta) > -\infty$ for any $s > 0$ and thus $\lim_{s\to\infty} \ell(\eta + s\delta) > -\infty$. By

Theorem 2.2, the limit of (2.6) is greater than $-\infty$ only if

$$\langle g(y_{\text{obs}}), \delta \rangle - b \geq 0,$$

and

$$b \geq \sigma_C(\delta).$$

Then

$$\sigma_C(\delta) - \langle g(y_{\text{obs}}), \delta \rangle \leq b - \langle g(y_{\text{obs}}), \delta \rangle \leq 0,$$

and recalling that $\sigma_C(\delta) = \sup_{g(y) \in C} \langle g(y), \delta \rangle$, we conclude that

$$\langle g(Y) - g(y_{\text{obs}}), \delta \rangle \leq 0, \quad \text{almost surely.}$$

□

Theorems 2.3 and 2.4 induce the following criteria about the existence of the MLE in the conventional sense:

Theorem 2.5 (Extension of Theorem 4 in Geyer (2009a))**.** *For a full exponential family with the setting of Theorem 2.3, the following are equivalent*

1. *the MLE exist.*
2. *Every direction of recession is a direction of constancy.*
3. *$N_C(g(y_{obs}))$ is a vector subspace.*
4. *$T_C(g(y_{obs}))$ is a vector subspace.*
5. *$g(y_{obs}) \in \operatorname{rint} C$.*

Proof. The equivalence of 1 – 4 are shown in Geyer (2009a). We have added condition 5 above and show the equivalency of this condition to condition 4. By Theorem 6.4 in Rockafellar (1970), the point $g(y_{\text{obs}}) \in C$ if and only if $v \in T_C(g(y_{\text{obs}}))$ implies $-v \in T_C(g(y_{\text{obs}}))$. But by Proposition 3.8 in Rockafellar and Wets (2004), the latter condition on $T_C(g(y_{\text{obs}}))$ is necessary and sufficient for $T_C(g(y_{\text{obs}}))$ to be a vector subspace. □

Corollary 2.1 (Corollary 5 in Geyer (2009a))**.** *For a full exponential family with the setting as Theorem 2.3, if δ is a direction of recession that is not a direction of constancy, then for all $\eta \in \Xi$, the function $s \mapsto \ell(\eta + s\delta)$ is strictly increasing on the interval where it is finite.*

With the addition of Condition 5 in Theorem 2.5, we have now tied this approach to the one previously described of Barndorff-Nielsen in Theorem 2.1. The MLE does not exist in the conventional sense if there exists a vector δ that is a direction of recession but not a direction of constancy. According to Corollary 2.1, this is very intuitive: it means the log likelihood is always increasing in that direction. Theorem 2.5 relates this precisely to the setting where $g(y_{\text{obs}})$ is on the relative boundary of the convex support.

2.4.3 Limiting conditional model

When the MLE does not exist for an exponential family in the conventional sense, there exists a direction of recession that is not a direction of constancy along which the log likelihood is strictly increasing. The behavior of the density function of the distribution along such a direction is described in the following theorem:

Theorem 2.6 (Theorem 6 in Geyer (2009a))**.** *For a full exponential family with the setting of Theorem 2.3, and additionally,*

1. *density function* $f_\eta(y)$ *defined by* (1.1),

2. *direction of recession* δ,

3. $H = \{w \in \mathbb{R}^d : \langle w - g(y_{obs}), \delta\rangle = 0\}$,

4. $P(g(Y) \in H) > 0$ *for some distribution in the family,*

5. $\sigma_C(\delta) = \sup_{g(y)\in C}\langle g(y), \delta\rangle$,

then for all $\eta \in \Xi$

$$\lim_{s\to\infty} f_{\eta+s\delta}(y) = \begin{cases} 0 & \langle g(y), \delta\rangle < \sigma_C(\delta) \\ \frac{f_\eta(y)}{P_\eta(g(Y)\in H)} & \langle g(y), \delta\rangle = \sigma_C(\delta) \\ +\infty & \langle g(y), \delta\rangle > \sigma_C(\delta). \end{cases} \tag{2.7}$$

If δ *is not a direction of constancy, then* $s \mapsto P_{\eta+s\delta}(g(Y) \in H)$ *is continuous and strictly increasing, and* $P_{\eta+s\delta}(g(Y) \in H) \to 1$ *as* $s \to \infty$.

Proof. By Theorem 2.4, if δ is a direction of recession, then $\langle g(Y) - g(y_{\text{obs}}), \delta\rangle \leq 0$ almost surely, which implies that $\langle g(Y), \delta\rangle \leq \langle g(y_{\text{obs}}), \delta\rangle$. So, the largest value that $\langle g(Y), \delta\rangle$ can take is $\langle g(y_{\text{obs}}), \delta\rangle$. Then

$$\sigma_C(\delta) = \sup_{g(y)\in C} \langle g(y), \delta\rangle = \langle g(y_{\text{obs}}), \delta\rangle.$$

From (1.1), we can express $f_{\eta+s\delta}(y)$ as

$$\begin{aligned} f_{\eta+s\delta}(y) &= e^{\langle \eta+s\delta, g(y)\rangle - c(\eta+s\delta)} \\ &= e^{\langle \eta, g(y)\rangle - c(\eta) + c(\eta) + s\langle \delta, g(y)\rangle - c(\eta+s\delta)} \\ &= f_\eta(y)\frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-\langle g(y),\delta\rangle s}}. \end{aligned}$$

The denominator in the fraction above has the form of the starting expression in Theorem 2.2 with $\langle g(y), \delta \rangle$ in place of b. Thus we can take the limit of $f_{\eta+s\delta}(y)$ as $s \to +\infty$ by applying Theorem 2.2,

$$f_{\eta+s\delta}(y) = f_\eta(y) \frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-\langle g(y),\delta\rangle s}} \to \begin{cases} 0 & \langle g(y), \delta \rangle < \sigma_C(\delta) \\ \frac{f_\eta(y)}{P_\eta(g(Y)\in H)} & \langle g(y), \delta \rangle = \sigma_C(\delta) \\ +\infty & \langle g(y), \delta \rangle > \sigma_C(\delta). \end{cases}$$

This gets us (2.7). We next show the behavior of $P_{\eta+s\delta}(g(Y) \in H)$ as $s \to +\infty$ when δ is a direction of recession that is not a direction of constancy.

$$\begin{aligned} P_{\eta+s\delta}(g(Y) \in H) &= \int_H e^{\langle g(y),\eta+s\delta\rangle - c(\eta+s\delta)} \, d\mu(y) \\ &= \int_H \frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-\langle g(y),\delta\rangle s}} f_\eta(\eta) \, d\mu(y) \\ &= \mathrm{E}_\eta \left(I_H \frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-\langle g(y),\delta\rangle s}} \right) \end{aligned}$$

Since the indicator function I_H evaluates to zero unless $\langle g(y), \delta \rangle = \langle g(y_{\text{obs}}), \delta \rangle$, this expression can pulled out of the expectation along with other constants, so that

$$\begin{aligned} P_{\eta+s\delta}(g(Y) \in H) &= \frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-s\langle g(y_{\text{obs}}),\delta\rangle}} \, \mathrm{E}_\eta \, I_H \\ &= \frac{e^{c(\eta)}}{e^{c(\eta+s\delta)-s\langle g(y_{\text{obs}}),\delta\rangle}} P_\eta(g(Y) \in H). \end{aligned}$$

Then taking the limit of $s \to +\infty$ via Theorem 2.2 gives

$$P_{\eta+s\delta}(g(Y) \in H) \to \frac{e^{c(\eta)}}{e^{c(\eta)} P_\eta(g(Y) \in H)} P_\eta(g(Y) \in H) = 1$$

as desired. □

There are many implications of Theorem 2.6 described in Geyer (2009a) which we repeat here and go into more detail for further clarity.

The event $\langle g(Y), \delta \rangle > \langle g(y_{\text{obs}}), \delta \rangle$ has zero probability by Theorem 2.4(2) and thus the $+\infty$ case need not be considered. The right-hand side of (2.7) can be viewed as a conditional density of a distribution with parameter η given $g(Y) \in H$, and expressed as $f_\eta(\cdot \mid g(y) \in H)$. This is because the numerator is in fact a joint density, which can be reasoned as follows:

Let $W = I(g(Y) \in H)$, which has density

$$f_\eta(w) = \begin{cases} P_\eta(g(Y) \in H) & \text{for } w = 1 \\ P_\eta(g(Y) \notin H) & \text{for } w = 0. \end{cases}$$

The conditional density of $W \mid Y$ is

$$f_\eta(w \mid y) = \begin{cases} 1, & \text{for } w = 0 \text{ and } g(y) \notin H \\ 0, & \text{for } w = 1 \text{ and } g(y) \notin H \\ 0, & \text{for } w = 0 \text{ and } g(y) \in H \\ 1, & \text{for } w = 1 \text{ and } g(y) \in H. \end{cases}$$

Then the conditional density for $Y \mid g(Y) \in H$ can be expressed

$$\begin{aligned} f_\eta(y \mid g(y) \in H) &= f_\eta(y \mid w = 1) \\ &= \frac{f_\eta(w = 1, y)}{f_\eta(w = 1)} = \frac{f_\eta(w = 1 \mid y) f_\eta(y)}{P_\eta(g(Y) \in H)} \\ &= \begin{cases} \frac{0 \cdot f_\eta(y)}{P_\eta(g(Y) \in H)} & \text{for } g(y) \notin H \\ \frac{1 \cdot f_\eta(y)}{P_\eta(g(Y) \in H)} & \text{for } g(y) \in H \end{cases} \end{aligned}$$

which is exactly the right-hand side of (2.7).

The log likelihood for $f_\eta(\,\cdot \mid g(y) \in H)$ for $\eta \in \Xi$ can be expressed as

$$\langle \eta, g(y_{\text{obs}}) \rangle - c(\eta) - \log P_\eta(g(Y) \in H)$$

which clearly has exponential family form with the same natural parameter η and statistic $g(y_{\text{obs}})$ as the original exponential family, and is called the *limiting conditional model* (LCM) (Geyer, 2009a). The cumulant function is different but is always finite since it only involves a summation over a finite number of terms. The parameter space for $f_\eta(\,\cdot \mid g(y) \in H)$ is still the whole Euclidean space, as in the original family.

When δ is a direction of recession that is not a direction of constancy, we can succinctly summarize the result of Theorem 2.6 as

$$\lim_{s \to \infty} f_{\eta + s\delta}(y) = f_\eta(y \mid g(y) \in H).$$

The last part of Theorem 2.6 says that for the sequence of distributions with parameter value $\eta + s\delta$, the probability of $g(Y)$ occurring on the hyperplane H is strictly increasing in s, going to 1. We may think of this as "probability accumulating on the boundary." This intuition is the basis for our extended algorithm in Section 4.1.

We can express the log likelihood of the LCM as a function of the log likelihood of the original family,

$$\ell_{LCM}(\eta) = \ell(\eta) - \log P_\eta(g(Y) \in H) \tag{2.8}$$

implying

$$\ell(\eta) < \ell_{LCM}(\eta).$$

The inequality above is strict: by Corollary 2.1, the log likelihood is strictly increasing on the entire interval on which it is finite and hence $P_\eta(g(Y) \in H)$ cannot equal 1. Thus even though the MLE does not exist for the original family, its log likelihood $\ell(\,\cdot\,)$ is bounded by the LCM log likelihood and increases strictly towards $\ell_{LCM}(\eta)$ for a given η as s increases. This is illustrated in Figure 2.1 in the case of a two-dimensional parameter space.

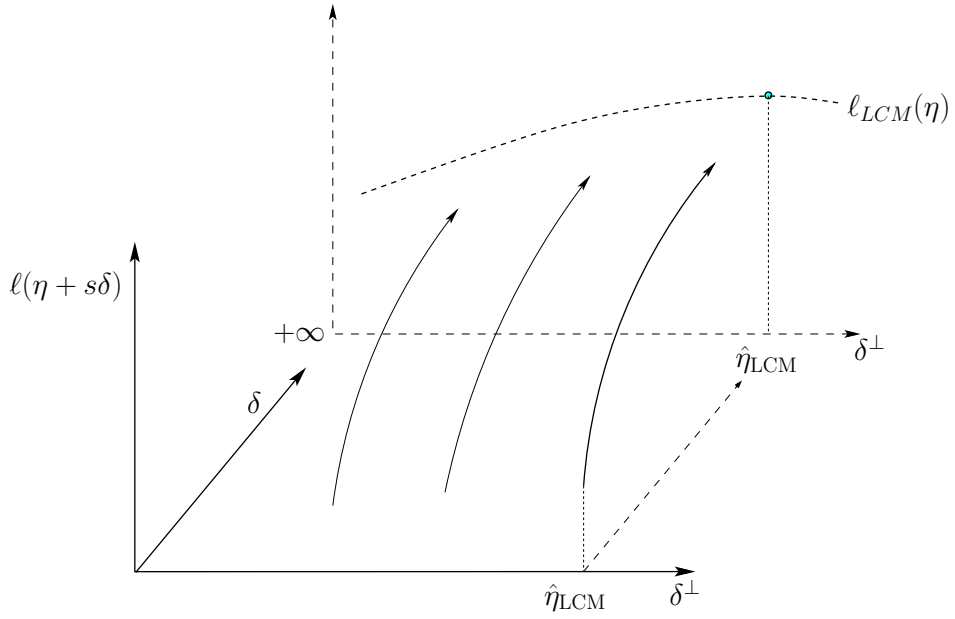

Figure 2.1: The log likelihood in the the direction δ, converging to the LCM log likelihood for a two-dimensional parameter space. The log likelihood $\ell(\hat{\eta}_{\text{LCM}} + s\delta)$ converges to the maximum of $\ell_{LCM}(\eta)$ as $s \to +\infty$.

Another consequence of Theorem 2.6 is in the mean value parameter space:

Corollary 2.2. *Under the setting of Theorem 2.6, let δ be a direction of recession that is not a direction of constancy. Then*

$$\mathrm{E}_{\eta+s\delta}\, g(Y) \to \mathrm{E}_\eta\left(g(Y) \mid g(Y) \in H\right) \tag{2.9}$$

as $s \to +\infty$.

Proof. By definition,

$$\mathrm{E}_{\eta+s\delta}\, g(Y) = \int g(y) f_{\eta+s\delta}(y)\, d\mu(y).$$

There exist a constant $M > g(y)$ for all $y \in \mathcal{Y}$ for the family to be full. Then by the dominated convergence theorem and the application of Theorem 2.6, the limit of the above expectation gets us (2.9). □

2.4.4 Generic direction of recession

We now define a special kind of direction of recession called a *generic direction of recession* (GDOR) that facilitates finding the LCM for which the MLE exists. A vector δ is a GDOR if in addition to being a direction of recession, $\delta \in \operatorname{rint} N_C(g(y_{\text{obs}}))$ and $N_C(g(y_{\text{obs}}))$ is not a vector subspace. Since normal cones are convex sets and relative interiors of nonempty convex sets are nonempty, by Theorem 2.5, a GDOR exists if and only if the MLE does not exist.

The following theorems and corollaries provide us with the foundation for finding and using GDORs. In essence, GDORs give us a convenient way to find an LCM for which the MLE exists. This first theorem provides the framework through which we can find a GDOR.

Theorem 2.7 (Theorem 7 from Geyer (2009a))**.** *For a full exponential family having polyhedral convex support C and observed value of the natural statistic $g(y_{obs})$ such that $g(y_{obs}) \in C$, let $T_C(g(y_{obs})) = \operatorname{con}(V)$, and define*

$$L = \{v \in V : -v \in T_C(g(y_{obs}))\}.$$

Then a GDOR exists if and only if $L \neq V$, in which case a vector δ is a GDOR if

and only if

$$\begin{aligned} \langle w, \delta \rangle &= 0, \quad w \in L \\ \langle w, \delta \rangle &< 0, \quad w \in V \setminus L \end{aligned}$$

It is not immediately obvious why defining L in such a manner is helpful; in fact, the motivation is rooted in linear programming. We demonstrate the application of this theorem in Chapter 4.

Corollary 2.3 (Corollary 8 from Geyer (2009a))**.** *Under the assumptions of Theorem 2.7, a GDOR is not a direction of constancy.*

Corollary 2.4 (Corollary 9 from Geyer (2009a))**.** *Under the assumptions of Theorem 2.7, suppose δ is a GDOR. Then*

$$\begin{aligned} T_{C \cap H}(g(y_{obs})) &= \operatorname{span} L \\ C \cap H &= C \cap (g(y_{obs}) + \operatorname{span} L). \end{aligned}$$

The following theorem shows that $C \cap H$ defines the support of the LCM in the *finite* state space setting that is of interest to us.

Theorem 2.8. *For a full exponential family having finite polyhedral convex support C and observed value of the natural statistic $g(y_{obs})$ such that $g(y_{obs}) \in C$, suppose δ is a GDOR. Then $C \cap H$ defines the convex support of the LCM.*

Proof. Theorem 2.6 says that if δ is a GDOR, which is a direction of recession that

is not a direction of constancy, then for the original model,

$$P_{\eta+s\delta}(g(Y) \in H) \to 1.$$

Since the LCM is the limiting distribution of the original model as $s \to +\infty$, it follows that $P^{LCM}(g(Y) \in H) = 1$. This puts 0 probability on points outside of H.

The original model puts 0 probability on any $x \notin C$, including $H \setminus C$. Because of the pointwise convergence of the original density to the LCM, since the original density puts zero probability on any $x \in H \setminus C$, the limiting model must as well. Then

$$\begin{aligned} P^{LCM}(g(Y) \in H) &= P^{LCM}\left(g(Y) \in ((H \setminus C) \cup (H \cap C))\right) \\ &= P^{LCM}\left(g(Y) \in H \setminus C\right) + P^{LCM}\left(g(Y) \in H \cap C\right) \\ &= 0 + P^{LCM}\left(g(Y) \in H \cap C\right) = 1. \end{aligned}$$

It is left to show that there is no smaller set than $C \cap H$ that can be the convex support of the LCM. Let $S = g(\mathcal{Y})$. The convex support of the LCM by construction is $\text{con}(S \cap H)$, which is closed since S is finite. Is $(C \cap H) = \text{con}(S \cap H)$? If a point $x \in C$, it is a convex combination of the points in S,

$$x = \alpha_1 w_1 + \alpha_2 w_2 + \cdots + \alpha_n w_n$$

where $\alpha_i \geq 0$, $\sum_i \alpha_i = 1$, and $w_i \in S$.

We claim for any $x \in C \cap H$, the coefficients of the w_i not in $S \cap H$ must all be zero. Suppose to get a contradiction that for an $x \in S \cap H$, there exist $\alpha_j > 0$ for

$w_j \notin S \cap H$ for $j \in J$. Then

$$\begin{aligned}\langle x - g(y_{\text{obs}}), \delta \rangle &= \left\langle \sum_i \alpha_i w_i - g(y_{\text{obs}}), \delta \right\rangle \\ &= \left\langle \sum_i \alpha_i \left(w_i - g(y_{\text{obs}}) \right), \delta \right\rangle \\ &= \left\langle \sum_{i \notin J} \alpha_i \left(w_i - g(y_{\text{obs}}) \right), \delta \right\rangle + \left\langle \sum_{j \in J} \alpha_j \left(w_j - g(y_{\text{obs}}) \right), \delta \right\rangle\end{aligned}$$

where the first term is less than or equal to zero by the definition of a direction of recession, and the second term is strictly less than zero since $w_j \notin H$. Thus $\langle x - g(y_{\text{obs}}), \delta \rangle < 0$, which means $x \notin H$, leading to a contradiction.

Thus x will equal a convex combination of just the points in $S \cap H$. Since this is true if and only if $x \in C \cap H$, we have that $C \cap H = \text{con}(S \cap H)$. □

Corollary 2.5. *For a full exponential family having finite polyhedral convex support C and observed value of the natural statistic $g(y_{obs})$ such that $g(y_{obs}) \in C$, suppose δ is a GDOR. Then the LCM found by using δ has convex support $C \cap H$ for which $g(y_{obs})$ lies in its relative interior, and the MLE exists.*

Proof. By Theorem 2.8, the LCM found by maximizing along δ has convex support $C \cap H$. By Corollary 2.4, the tangent cone $T_{C \cap H}(g(y_{\text{obs}}))$ is in fact a vector subspace and thus $g(y_{\text{obs}}) \in \text{rint}(C \cap H)$. Finally, by Theorem 2.5, the MLE for this model must then exist. □

When we find the MLE in a LCM, $\hat{\eta}_{\text{LCM}}$, we say we have found an MLE in the Barndorff-Nielsen completion of the original family. This LCM MLE maximizes the likelihood in the family that is the union of the LCM family and the original family.

Note that the LCM is not identifiable, since its support is necessarily lower dimensional than that of the original model. That is, there must exist a constancy space,

$\Gamma_{\lim}$, such that

$$\ell(\eta+\gamma)^{LCM} = \ell(\eta)^{LCM} \tag{2.10}$$

for any $\gamma \in \Gamma_{\lim}$. By Theorem 2.3, a GDOR δ is one such γ.

2.5 One-sided confidence interval

Theorem 2.6 and Corollary 2.5 tells us that when $\ell(\,\cdot\,)$ is the log likelihood for the original model, s is a scalar, and δ a GDOR, then

$$\lim_{s\to+\infty} \ell(\hat{\eta}_{\text{LCM}} + s\delta) = \ell(\hat{\eta}_{\text{LCM}})^{LCM} = \sup_{\mathbb{R}^d} \ell(\eta).$$

The MLE for the original model, which we may think of as "at infinity", has more structure in this context—it is the value $\hat{\eta}_{\text{LCM}}$ sent to infinity in direction δ.

We can then characterize the distributions in the original model that are in the neighborhood of the LCM MLE distribution by means of a confidence interval. As s goes from $-\infty$ to $+\infty$, the probability of observing $g(Y) \in H$ strictly increases to 1. To construct a one-sided $1-\alpha$ confidence interval for how "close" distributions in the original model indexed by $\hat{\eta}_{\text{LCM}} + s\delta$ are to the LCM MLE distribution, we find the unique s, call it $\hat{s}$, such that

$$P_{\hat{\eta}_{\text{LCM}}+s\delta}\left(g(Y) \in H\right) = \alpha.$$

Then $[\hat{s}, +\infty)$ is a $1-\alpha$ confidence interval for the scalar parameter s, and, in turn,

$$\{\hat{\eta}_{\text{LCM}} + t\delta : t \geq \hat{s}\}$$

gives a $1-\alpha$ confidence interval for the parameter $\hat{\eta}_{\text{LCM}} + s\delta$. These are a range of indices to distributions in the neighborhood of the LCM MLE distribution.

Chapter 3

Algorithm

3.1 Long range search algorithm

We now present our line search algorithm, which will converge to the MLE for any regular exponential family if the MLE exists. When the MLE does not exist, the log likelihood is strictly increasing but bounded above by the log likelihood of the LCM, which itself must have a maximum. In such a case, this algorithm will climb the log likelihood until the gradient is close to zero and $\mathrm{E}_{\eta_k} g(Y)$ nears $g(y_{\text{obs}})$. Chapter 4 details how to adapt the algorithm to this setting so that the LCM support is identified and the MLE for the LCM subsequently found. In this chapter, we focus on the basic algorithm for the setting where the MLE exists.

The algorithm and requirements are presented in Theorem 3.1 and 3.2. The theory is divided into two parts: the first guarantees that the log likelihood gradient converges to zero, the second shows that when the MLE exists, this is equivalent to finding the MLE.

Theorem 3.1 (Exponential family zero gradient attainment)**.** *Consider any line search of the form*

$$\eta_{k+1} = \eta_k + \alpha_k p_k \tag{3.1}$$

used to maximize the log likelihood function $\ell(\cdot)$ of a regular exponential family on a finite sample space, where the search direction p_k *is a non-zero* ascent direction *such that the angle θ_k between the search direction p_k and steepest ascent direction $\nabla\ell(\eta_k)$ is restricted to be less than 90 degrees by*

$$\cos\theta_k \geq \delta > 0 \tag{3.2}$$

for some fixed $\delta > 0$.

Then, unless $\nabla\ell(\eta_k) = 0$, in which case η_k is already the solution and the search is complete, it is possible to find a step length α_k that satisfies the curvature condition

$$0 \leq \nabla\ell(\eta_k + \alpha_k p_k)^T p_k \leq c\nabla\ell(\eta_k)^T p_k \tag{3.3}$$

for some fixed $0 < c < 1$.

Furthermore, repeated iterations of (3.1) *satisfying* (3.2) *and* (3.3) *will produce a sequence, $\eta_1, \eta_2, \ldots$ such that*

$$\lim_{k\to\infty} \|\nabla\ell(\eta_k)\| = 0.$$

Proof of Theorem 3.1. Let $f(\cdot)$ represent the negative log likelihood $-\ell(\cdot)$, the objective function to be minimized. We proceed from the perspective of a minimization of a function $f(\cdot)$ since this is the convention in the optimization literature (Nocedal and Wright, 1999; Rockafellar and Wets, 2004).

The negative log likelihood function $-\ell(\cdot)$ is strictly convex by (2.2), and continuous since it is infinitely differentiable by Theorem 5.8 in Lehmann and Casella (1998). It is bounded below by the negative LCM log likelihood as described by (2.8), which is guaranteed to have a global minimum.

Then, unless $\nabla f(x_k) = 0$ in which case x_k is already the solution, for each k, we

can uniquely define α_{c_k} as follows:

$$\nabla f(x_k + \alpha_{c_k} p_k)^T p_k = c \nabla f(x_k)^T p_k \tag{3.4}$$

The point α_{c_k} is uniquely defined because it is the minimizer of $\alpha \mapsto f(x_k + \alpha p_k) - \alpha c \nabla f(x_k)^T p_k$. We may also define $\alpha_{\min_k}$ as follows:

$$\alpha_{\min_k} = \begin{cases} \alpha \text{ s.t. } \nabla f(x_k + \alpha p_k)^T p_k = 0 & \text{if such an } \alpha \text{ exists} \\ +\infty & \text{otherwise.} \end{cases} \tag{3.5}$$

These values appear on the α-axis in Figure 3.1 for the case where a minimizer exists for $\alpha \mapsto f(x_k + \alpha p_k)$.

By the strict convexity of f and Theorem 2.14(b) in Rockafellar and Wets (2004),

$$f(x_k + \alpha_{c_k} p_k) < f(x_k) + \left[\nabla f(x_k + \alpha_{c_k} p_k)\right]^T \alpha_{c_k} p_k.$$

Applying (3.4) to the right hand side of the above gives

$$f(x_k + \alpha_{c_k} p_k) < f(x_k) + \alpha_{c_k} c \nabla f(x_k)^T p_k. \tag{3.6}$$

(See points a and b in Figure 3.1).

The subproblem $\alpha \mapsto f(x_k + \alpha p_k)$ is strictly convex and hence monotonically decreasing at α_k such that $\alpha_{c_k} \leq \alpha_k < \alpha_{\min_k}$ (in Figure 3.1, see points b and c). That is,

$$f(x_k + \alpha_{min} p_k) \leq f(x_k + \alpha_k p_k) \leq f(x_k + \alpha_{c_k} p_k). \tag{3.7}$$

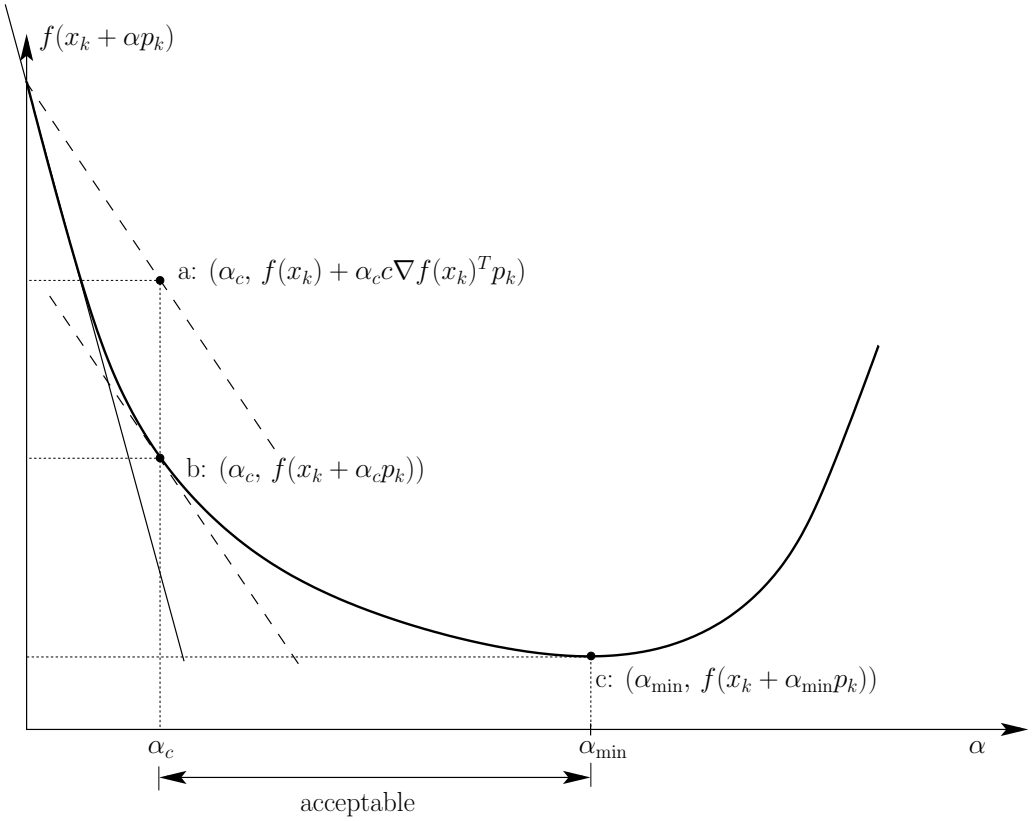

Figure 3.1: Acceptable region for α according to curvature condition (3.3) when restricting to direction p_k.

Combining the second inequality of (3.7) with (3.6), we have

$$f(x_k + \alpha_k p_k) < f(x_k) + \alpha_{c_k} c \nabla f(x_k)^T p_k, \tag{3.8}$$

which can be rearranged as

$$f(x_k) - f(x_k + \alpha_k p_k) > -\alpha_{c_k} c \nabla f(x_k)^T p_k. \tag{3.9}$$

This last inequality (3.9) expresses a lower bound for the amount of decrease in our objective function at each step (the right-hand side is positive since $\nabla f(x_k)^T p_k < 0$

by assumption that p_k is a descent direction). It is this lower bound that we will use to cover the distance to the minimum of the objective function.

We now turn our attention to (3.4). Define $x_{c_k} = x_k + \alpha_{c_k} p_k$. Then

$$\nabla f(x_{c_k})^T p_k = c \nabla f(x_k)^T p_k.$$

Subtracting $\nabla f(x_k)^T p_k$ from both sides gives

$$\left(\nabla f(x_{c_k}) - \nabla f(x_k)\right)^T p_k = (c-1) \nabla f(x_k)^T p_k. \tag{3.10}$$

By (2.2), $\nabla^2 \ell(\eta)$ is bounded for finite state space $g(\mathcal{Y})$, which is true by assumption. Thus $|\nabla^2 f(x)| \leq K$ for some constant K for all x. Then by Theorems 9.2 and 9.7 in Rockafellar and Wets (2004), $\nabla f(x)$ is Lipschitz continuous relative to the convex set $\mathbb{R}^d$.

Thus there exists a constant $L < \infty$ such that

$$\|\nabla f(x) - \nabla f(\tilde{x})\| \leq L \|x - \tilde{x}\| \quad \text{for all } x, \tilde{x} \in \mathbb{R}^d. \tag{3.11}$$

Applying (3.11) to x_{c_k} and x_k, we have

$$\|\nabla f(x_{c_k}) - \nabla f(x_k)\| \leq L \|x_{c_k} - x_k\|$$

or

$$\|\nabla f(x_{c_k}) - \nabla f(x_k)\| \leq L \|\alpha_{c_k} p_k\|.$$

Multiplying both sides by $\|p_k\|$ gives

$$\|\nabla f(x_{c_k}) - \nabla f(x_k)\| \cdot \|p_k\| \leq \alpha_{c_k} L \|p_k\|^2$$

and by Cauchy-Schwarz this implies

$$\begin{aligned}(\nabla f(x_{c_k}) - \nabla f(x_k))^T p_k &\leq \|\nabla f(x_{c_k}) - \nabla f(x_k)\| \cdot \|p_k\| \\ &\leq \alpha_{c_k} L \|p_k\|^2.\end{aligned} \tag{3.12}$$

Substituting (3.10) into the left-hand side of this last inequality (3.12) gives

$$(c-1)\nabla f(x_k)^T p_k \leq \alpha_{c_k} L \|p_k\|^2$$

or

$$-\alpha_{c_k} \leq \frac{(1-c)}{L} \frac{\nabla f(x_k)^T p_k}{\|p_k\|^2}. \tag{3.13}$$

Write out the first $k+1$ inequalities of (3.9):

$$\begin{aligned} f(x_1) &< f(x_0) + \alpha_{c_0} c \nabla f(x_0)^T p_0 \\ f(x_2) &< f(x_1) + \alpha_{c_1} c \nabla f(x_1)^T p_1 \\ &\dots \\ f(x_k) &< f(x_{k-1}) + \alpha_{c_{k-1}} c \nabla f(x_{k-1})^T p_{k-1} \\ f(x_{k+1}) &< f(x_k) + \alpha_{c_k} c \nabla f(x_k)^T p_k \end{aligned} \tag{3.14}$$

Telescoping the right-hand side of (3.14),

$$f(x_{k+1}) < f(x_0) + c\sum_{j=0}^{k} \alpha_{c_j} \nabla f(x_j)^T p_j.$$

Noting that $\nabla f(x_j)^T p_j < 0$, we can substitute our upper bound (3.13) for $-\alpha_{c_j}$ in the right-hand side above,

$$f(x_{k+1}) < f(x_0) - c\sum_{j=0}^{k} \frac{(1-c)}{L} \frac{\nabla f(x_j)^T p_j}{\|p_j\|^2} \nabla f(x_j)^T p_j$$

which simplifies to

$$f(x_{k+1}) < f(x_0) - c\sum_{j=0}^{k} \frac{(1-c)}{L} \frac{(\nabla f(x_j)^T p_j)^2}{\|p_j\|^2}.$$

Because $f(x)$ is bounded below by assumption, there exists some $M < \infty$ such that $f(x_0) - f(x_{k+1}) < M$ for all k. Then rearranging the above yields,

$$\frac{c(1-c)}{L} \sum_{j=0}^{k} \frac{(\nabla f(x_j)^T p_j)^2}{\|p_j\|^2} < M < \infty.$$

The angle θ_j between the search direction p_k and steepest descent direction $-\nabla f(x_k)$ can be expressed by $\cos\theta_j = \frac{-\nabla f(x_j)^T p_j}{\|\nabla f(x_j)\| \cdot \|p_j\|}$. Substituting this into the equation above and taking $k \to \infty$,

$$\frac{c(1-c)}{L} \sum_{j=0}^{\infty} \|\nabla f(x_j)\|^2 \cos^2\theta_j < \infty.$$

Since $0 < c < 1$,

$$\sum_{j=0}^{\infty} \|\nabla f(x_j)\|^2 \cos^2 \theta_j < \infty. \tag{3.15}$$

The convergent series in (3.15) implies that

$$\|\nabla f(x_k)\|^2 \cos^2 \theta_k \to 0 \text{ as } k \to \infty.$$

With the additional restriction on the search direction p_k such that $\cos \theta_k \geq \delta > 0$ for some choice of δ, for all choices of k, we get the desired convergence result of

$$\lim_{k \to \infty} \|\nabla f(x_k)\| = 0.$$

□

In fact, Theorem 3.1 and its proof can be adapted to the more general setting of optimizing any proper, upper semi-continuous function with an added assumption of bounded level sets of the function, as done in Okabayashi and Geyer (2011). Because we wish to apply this to the specific setting of exponential families where the MLE might not exist, however, we cannot assume bounded level sets. Instead we rely on other properties of the exponential family log likelihood that guarantee us the necessary Lipschitz continuous gradient.

We now apply Theorem 3.1 to find the MLE when it is known to exist:

Theorem 3.2 (MLE convergence)**.** *For a regular exponential family with minimal representation where the MLE exists, the line search described in Theorem 3.1 can be applied to the negative log likelihood function $-\ell(\eta)$ so that a search starting at any $\eta_0 \in \Xi$ will converge to the MLE of η.*

Theorem 3.1 shows that the gradient of the objective function converges to 0. The

proof for Theorem 3.2 is concerned with the conditions for mapping this convergence to the convergence of the iterated parameter estimates η_k to the unique MLE. In particular, the mapping from η_k to the gradient must be globally invertible.

Proof of Theorem 3.2. The Fisher information for a regular exponential family is non-singular by (2.3) and thus invertible. If we consider the map defined by

$$h(\eta) = \nabla c(\eta)$$

where $c(\eta) = \log \kappa(\eta)$ is the cumulant function introduced in Section 1.2, its first derivative matrix is

$$\nabla h(\eta) = \nabla^2 c(\eta) = I(\eta) \tag{3.16}$$

which is again non-singular. Since this is true for any η, by the inverse function theorem, h is everywhere locally invertible.

In fact, h is globally invertible. For any μ in the range of h, consider the function

$$q(\eta) = \mu^T \eta - c(\eta).$$

Since $\nabla^2 q(\eta) = -I(\eta)$ by (3.16), q is strictly concave. Therefore, a maximizer for q, call it $\hat{\eta}$, is unique if it exists and satisfies the first-order condition

$$\nabla q(\hat{\eta}) = 0.$$

This in turn implies that

$$\mu - h(\hat{\eta}) = 0$$

or

$$\mu = h(\hat{\eta}).$$

Because of the assumption that μ is in the range of h, this means that $\hat{\eta}$ in fact exists, and by the strict concavity of q, is unique. This implies that h must be one-to-one and hence globally invertible.

Since c is infinitely differentiable by Theorem 2.7.1 in Lehmann and Romano (2005), so is h, and by the inverse function theorem, so is h^{-1} (even if we do not know the form of h^{-1}). The first derivative of h^{-1} can be expressed as

$$\nabla h^{-1}(\mu) = [\nabla h(\eta)]^{-1} = [I(\eta)]^{-1}$$

when $\mu = h(\eta)$ and is thus non-singular everywhere, including at the MLE of η, $\hat{\eta}_{\text{MLE}}$.

Thus our algorithm, which concludes that $\|\nabla \ell(\eta_k)\| = \|g(y) - h(\eta_k)\| \to 0$, implies that

$$\mu_k = h(\eta_k) \to g(y),$$

or

$$h^{-1}(\mu_k) \to h^{-1}\left(g(y)\right),$$

or

$$\eta_k \to \hat{\eta}_{\text{MLE}}.$$

□

3.2 Refinements of the algorithm

In Theorem 3.1, we restricted our search direction p_k to be an ascent direction, so that $\nabla\ell(\eta_k)^T p_k > 0$ or, alternatively, the angle θ_k between the search direction p_k and steepest ascent direction $\nabla\ell(\eta_k)$ is less than 90 degrees. However, this still leaves many possibilities for the choice of p_k other than steepest ascent. In addition, we have specified restrictions on the step size α_k in the curvature condition (3.3) with $0 < c < 1$, but it would be useful to know if certain values of c are better than others.

3.3 Search directions

In our examples in Chapter 5 for maximizing the log likelihood function $\ell(\eta)$, we default to steepest ascent directions in our implementation for transparency. Although often effective in early steps, steepest ascent directions can result in an inefficient zigzagging trajectory of the sequence η_k (Sun and Yuan, 2006, Section 3.1) as illustrated in Figure 3.2 (top). This is especially problematic when the MLE does not exist in the conventional sense—the MLE is actually off at infinity and a zig-zagging route may make it especially difficult to realize this.

Conjugate gradient methods may partially address this phenomena and cover the sample space more efficiently (Nocedal and Wright, 1999, Chapter 5). It is easy to implement a variant of the Polak-Ribière method (Nocedal and Wright, 1999, pp. 120–122) here, requiring little more in terms of calculation or storage. The search direction p_k would update with an extra intermediate step as follows:

$$\gamma_{k+1}^{PR} = \max\left(0, \frac{[\nabla\ell(\eta_{k+1})]^T(\nabla\ell(\eta_{k+1}) - \nabla\ell(\eta_k))}{\|\nabla\ell(\eta_k)\|^2}\right)$$
$$p_{k+1} = \nabla\ell(\eta_{k+1}) + \gamma_{k+1}^{PR}\, p_k.$$

Note that when $\gamma_{k+1}^{PR} = 0$, p_{k+1} will just be $\nabla\ell(\eta_{k+1})$, the direction of steepest ascent, and so this serves as a "reset". The resulting search direction p_{k+1} is always an ascent direction since

$$\begin{aligned}[\nabla\ell(\eta_{k+1})]^T p_{k+1} &= [\nabla\ell(\eta_{k+1})]^T \left[\nabla\ell(\eta_{k+1}) + \gamma_{k+1}^{PR}\, p_k\right] \\ &= \|\nabla\ell(\eta_{k+1})\|^2 + \gamma_{k+1}^{PR}[\nabla\ell(\eta_{k+1})]^T p_k > 0\end{aligned}$$

by (3.3).

Another pragmatic approach may be to use a search direction resulting from a regression through the previous few values of η_k. In the setting where MCMC samples are used to approximate the gradient, an empirically determined generic direction of recession, which we discuss how to find in Section 4.6, may also be useful (see Figure 3.2 (bottom)).

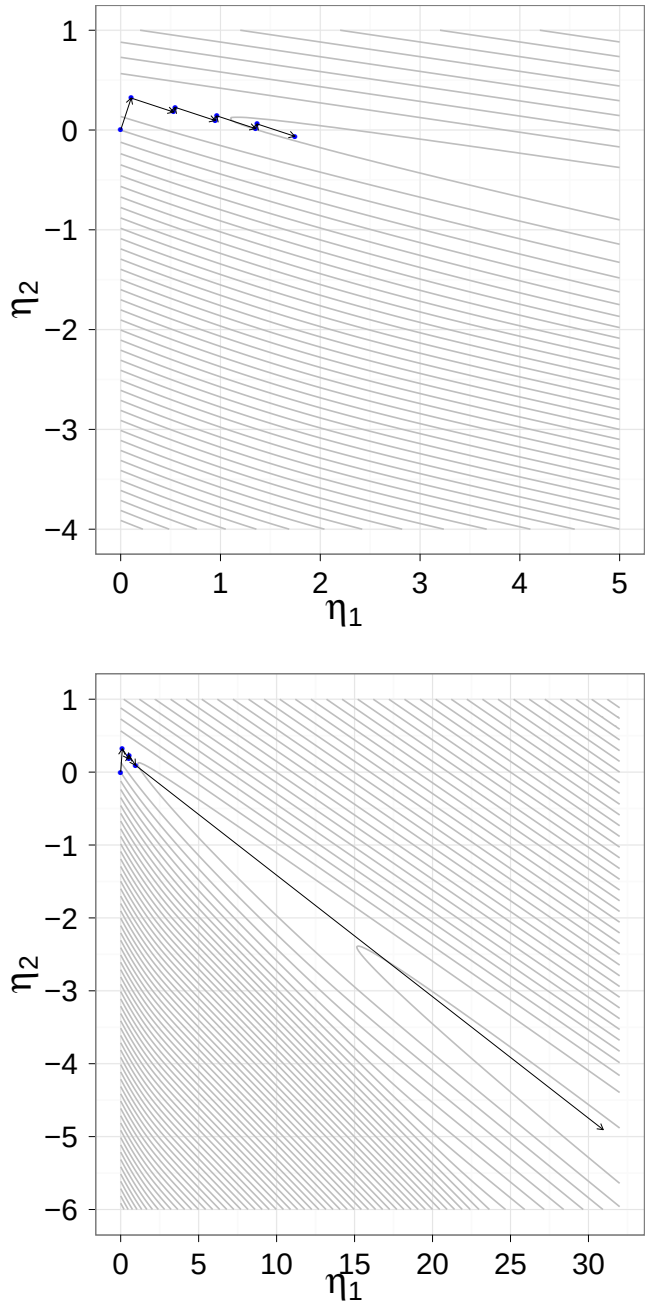

Figure 3.2: Contour plots of an ERGM log likelihood when the MLE does not exist. The surface tends to flatten, though technically it is still concave. This can cause the steepest descent algorithms to zigzag, which is inefficient (top). However, by periodically using search directions determined by a regression through previous points, or in this case an empirical generic direction of recession, the algorithm can make much larger steps (bottom).

3.4 Step size

We now turn our attention to the optimal step size α_k when our objective function is the log likelihood of an exponential family. Taking the derivative of $\ell(\eta_k + \alpha_k p_k)$ with respect to α_k shows that the log likelihood is maximized as a function of α_k along the direction p_k when

$$\nabla \ell(\eta_{k+1})^T p_k = 0.$$

By choosing c to be small, say 0.2, we ensure that the step taken is close to maximizing the log likelihood along the search direction. This is also apparent in Figure 1.6.

Making c too small, however, may make it difficult to find an α_k that meets the curvature condition (3.3) since this search must be done numerically. In fact, as the line search nears the MLE and $\nabla \ell(\eta_k)$ gets smaller, the rightmost term in (3.3) gets smaller in magnitude (it equals $c\|\nabla \ell(\eta_k)\|^2$ if using steepest ascent directions), making a numerical search for α_k more challenging. This issue is exacerbated when the quantities are noisily approximated via MCMC, as discussed in the next section. We currently construct a spline to approximate $\nabla \ell(\eta_k + \alpha_k p_k)^T p_k$ as a function of α_k based on previous guesses using the `gam` function in the `mgcv` package (Wood, 2011) in R. However, this approach does not use the fact that we have a strictly concave function of α_k and thus a more effective approximation should be possible.

3.5 MCMC approximations

Our algorithm requires us to be able to calculate $\nabla\ell(\eta)$ using (2.1). For many applications, we will need to approximate $\mathrm{E}_\eta\, g(Y)$ using MCMC. That is,

$$\nabla\ell(\eta) = g(y) - \mathrm{E}_\eta\, g(Y) \approx g(y) - \frac{1}{m}\sum_{i=1}^{m} g(Y_i), \tag{3.17}$$

where $Y_1, \ldots, Y_m$ are MCMC draws from the distribution with parameter η. There are many MCMC algorithms such as Metropolis-Hastings (Geyer, forthcoming), which is typically used for ERGMs, or Swendsen-Wang (Swendsen and Wang, 1987), which we use for the Ising model example in Section 5.2. We show examples in the next section where $\nabla\ell(\eta)$ can be calculated exactly and where it must be approximated.

The accuracy of the approximation in (3.17) increases with Monte Carlo sample size m. When the current estimate is far away from the MLE, we can use smaller m to save time and work with a fairly noisy approximation of the gradient. However, when the current estimate approaches the MLE, larger m are necessary.

Our algorithm relies on the computed values of $\nabla\ell(\eta)$ in the curvature condition (3.3), as well as the stop condition for the algorithm, $\|\nabla\ell(\eta_k)\| < \epsilon$. Given that we may only have approximations of $\nabla\ell(\eta)$, we cannot know for certain if either of these conditions is truly met. We can ameliorate this by constructing confidence intervals for each of the inequalities.

For the inequalities in (3.3), we can estimate asymptotic standard errors of the quantities $\nabla\ell(\eta_k + \alpha_k p_k)^T p_k$ and $c\nabla\ell(\eta_k)^T p_k - \nabla\ell(\eta_k + \alpha_k p_k)^T p_k$ by appealing to the Markov chain Central limit theorem (Chan and Geyer, 1994; Jones, 2004; Roberts and Rosenthal, 1997, 2004). The `initseq` function from the R package `mcmc` (Geyer, 2009c) can be used to estimate asymptotic standard errors for univariate functionals of reversible Markov chains: given an MCMC sample for a univariate quantity,

`initseq` returns a value (divided by sample size) that is an estimate of the asymptotic variance in the Markov chain central limit theorem. Both of the quantities in (3.3) are univariate. In the second expression, $c\nabla\ell(\eta_k)^T p_k - \nabla\ell(\eta_k + \alpha_k p_k)^T p_k$, the MCMC sample generated for $\nabla\ell(\eta_k + \alpha_k p_k)^T p_k$ is independent of the sample generated for $c\nabla\ell(\eta_k)^T p_k$. Thus `initseq` can be applied to each sample separately and the results summed for an estimated variance. We can then be approximately 95% confident (non-simultaneously) that α_k satisfies (1.15) if

$$\begin{aligned} \nabla\ell(\eta_k + \alpha_k p_k)^T p_k - 1.645 \cdot \mathrm{se}_1 > 0 \\ c\nabla\ell(\eta_k)^T p_k - \nabla\ell(\eta_k + \alpha_k p_k)^T p_k - 1.645 \cdot \mathrm{se}_2 > 0 \end{aligned}$$

where se_1 and se_2 are the asymptotic standard errors for $\nabla\ell(\eta_k + \alpha_k p_k)^T p_k$ and $c\nabla\ell(\eta_k)^T p_k - \nabla\ell(\eta_k + \alpha_k p_k)^T p_k$, respectively, calculated as described.

The delta method can be applied to estimate a standard error for $\|\nabla\ell(\eta_k)\|$. The multivariate version of the delta method states that for a sequence of r.v. B_n such that

$$\sqrt{n}(B_n - \beta) \xrightarrow{\mathcal{D}} N(0, \Sigma)$$

and a function $h(\,\cdot\,)$ where $\nabla h(\beta)$ is defined and non-zero,

$$\sqrt{n}\,(h(B_n) - h(\beta)) \xrightarrow{\mathcal{D}} N\left(0, \nabla h(\beta)^T \Sigma \nabla h(\beta)\right).$$

We set $B_n = \nabla\ell_n(\eta)$, the sequence of MCMC approximations of the gradient, and $\beta = \nabla\ell(\eta)$, where we know that $B_n \xrightarrow{a.s.} B$ by SLLN. We do not know Σ, the variance of $\nabla\ell(\eta)$, but we will approximate this with $\hat{\Sigma}$, the scaled sample variance-covariance matrix of our MCMC batches of the canonical statistic (the `initseq` function requires

a univariate vector and so cannot be used here). That is,

$$\hat{\Sigma} = \frac{1}{\text{nbatch}} \frac{1}{m-1} \sum_{i=1}^{m} (g(Y_i) - \overline{g(Y)})(g(Y_i) - \overline{g(Y)})^T.$$

Thus the asymptotic variance is calculated by

$$V\left(\|\nabla \ell(\eta_k)\|\right) = \frac{1}{\|\nabla \ell(\eta_k)\|^2} \nabla \ell(\eta_k)^T \, \hat{\Sigma} \, \nabla \ell(\eta_k)$$

and we can be approximately 95% confident that $\|\nabla \ell(\eta_k)\| > \epsilon$ if

$$\|\nabla \ell(\eta_k)\| - 1.645 \sqrt{V\left(\|\nabla \ell(\eta_k)\|\right)} > \epsilon.$$

In practice, however, use of confidence intervals does not appear necessary with Monte Carlo sample sizes that are set large enough so that these standard errors are initially small relative to the point estimates. The ratio of point estimate to standard error of course decreases as the algorithm progresses and the estimate of the parameter nears the MLE, reflected in $\nabla \ell(\eta_k)$ nearing 0. Thus these confidence intervals are most useful as a guide for when to increase the MCMC sample size, or when to switch methods, or when to terminate the algorithm.

3.6 Combining with other algorithms

We believe the best use of this algorithm is in combination with other faster methods like MCMC-MLE (Geyer and Thompson, 1992) or Newton-Raphson safeguarded by our line search algorithm. Our algorithm with steepest ascent or conjugate gradient search direction should be used initially from "long range", when one has no good intuition for an initial value.

It is well known that when the objective function is quadratic, the conjugate

gradient method with exact arithmetic converges to the solution in at most d steps, where d is the dimension of the problem (Nocedal and Wright, 1999, Chapter 5). As a rule of thumb then, we think using our algorithm for $2d$ steps before switching seems reasonable when using conjugate gradient directions. Determining when we are inside the "radius of convergence" for algorithms like Newton-Raphson or MCMC-MLE is an area for further research.

In the case of MCMC-MLE, it was shown in Section 1.4.3 that getting a distribution close enough to $g(y_{\text{obs}})$ in mean value parameterization so that it is within the convex hull of the MCMC samples is a necessary condition for this algorithm to converge (in fact, this appears to be the impetus for the steplength MCMC-MLE approach of Hummel et al. (2010)). However, this is not sufficient. An effective approach may be to examine the importance sampling weights used in (1.10) in the previous iteration, and look for them to stabilize. This "rearview" approach should inform us if the previous value for η_k was close enough to apply MCMC-MLE; applying MCMC-MLE to the current distribution should then converge.

Chapter 4

Computational Geometry

The pseudocode presented in Section 1.6.1 provides the basic framework for the application of our algorithm. For cases where the MLE is known to exist in the conventional sense and the gradient $\nabla\ell(\eta)$ cannot be calculated exactly, an appropriate MCMC sampler can be used to approximate this quantity as described in Section 3.5. An efficient root finding algorithm to find a step size α_k that satisfies the curvature condition (3.3) is also highly desirable (and is in fact an area for further research). However, the overall algorithm is the same.

Some modification is required for the setting where the MLE may not exist. We need to apply the theory developed in Section 2.4.3, which relies heavily on linear programming methodologies. For example, even when coordinates of $g(y_{\text{obs}})$ and the vertices that define a convex polytope C are known, it is not a trivial matter to know if this point lies in the exterior, interior, or exactly on a boundary of C. Further complicating this process is that the representation of numbers in computers is not exact, e.g., `0.3 / 3 == 0.1` evaluates to `FALSE`. Fortunately the rational arithmetic-based linear programming tools we need have already been made available to us through the `rcdd` package (Geyer and Meeden, 2009) in R.

In this chapter, we first present an extended pseudocode of our algorithm to handle the possible non-existence of MLEs: when the algorithm determines that the

MLE does not exist in the original model, it proceeds to look for the MLE in the LCM. We then point out the specific functionality that is necessary, and describe the commands by which each of these operations can be performed in R. We do not provide theoretical derivations for the linear programming operations here and refer the interested reader to Fukuda (2004, 2008).

4.1 Algorithm pseudocode extension

Get an initial value, $\eta_1 = (0, \dots 0)$.
Sample $g(Y_1), \dots, g(Y_m)$ from distribution with parameter η_1.
Set $gy.hull = \text{con}(g(Y_1), \dots, g(Y_m))$.
Approximate

$$\nabla \ell(\eta_1) \approx g(y_{\text{obs}}) - \frac{1}{m} \sum_{i=1}^{m} g(Y_i).$$

Set $p_1 = \nabla \ell(\eta_1)$, $k = 1$, $LCM.flag =$ FALSE, $c = 0.2$.

while $\|\nabla \ell(\eta_k)\| > \epsilon$ **do**
Find a step size α_k that satisfies the curvature condition (3.3)

$$0 \leq \nabla \ell(\eta_k + \alpha_k p_k)^T p_k \leq c \nabla \ell(\eta_k)^T p_k.$$

$\eta_{k+1} = \eta_k + \alpha_k p_k$.
Sample $g(Y_1), \dots, g(Y_m)$ from distribution with parameter η_{k+1}.
if $LCM.flag =$TRUE **then**
if Any new points refute claim that $\tilde{F}$ is a face **then**
Restore $gy.hull$ to its previous state.
Set $LCM.flag = FALSE$.
else
Restrict sample to those on empirical face $\tilde{F}$.
end if
end if
Call the resulting sample points $g(Y_{(1)}), \dots, g(Y_{(p)})$.
Update $gy.hull$ to include $g(Y_{(1)}), \dots, g(Y_{(p)})$.

Question: $g(y_{\mathbf{obs}}) \notin gy.hull$**?**
if Yes, $g(y_{\text{obs}})$ is in exterior of $gy.hull$ **then**
Continue.
else
No, $g(y_{\text{obs}}) \in \text{rint}(gy.hull)$ or $g(y_{\text{obs}}) \in \text{rbd}(gy.hull)$
Find empirical face $\tilde{F}$ of $gy.hull$ in which $g(y_{\text{obs}})$ lies in relative interior.
if $\tilde{F} == gy.hull$ **then**
$g(y_{\text{obs}}) \in \text{rint}(gy.hull)$ ▷ MLE exists
else
$g(y_{\text{obs}}) \in \text{rbd}(gy.hull)$ and $g(y_{\text{obs}}) \in \text{rint}\,\tilde{F}$
Find an empirical GDOR, $\tilde{\delta}$.
Either:
(1) MLE exists; $gy.hull$ just happen to touch $g(y_{\text{obs}})$,
(2) MLE does *not* exist; both $gy.hull$ and true C touch $g(y_{\text{obs}})$.
if $> 60\%$ of the sample points are in $\tilde{F}$ (via $\tilde{\delta}$) **then**
Conclude that we are in case (2); case (1) is very unlikely.
Set $LCM.flag =$ TRUE.
Set $gy.hull = \tilde{F}$.
end if
end if
end if

Approximate

$$\nabla \ell(\eta_{k+1}) \approx g(y_{\text{obs}}) - \frac{1}{p} \sum_{i=1}^{p} g(Y_{(i)}). \tag{4.1}$$

Find the new search direction p_{k+1}, which must be an ascent direction.
This may be a regression direction, an empirical GDOR $\tilde{\delta}$, or $\nabla \ell(\eta_{k+1})$.
$k = k + 1$.
end while

The driving question in determining MLE existence in the conventional sense is whether or not the observed statistic lies in the relative boundary of the convex support. The above pseudocode goes about this in a seemingly roundabout way—instead of checking if $g(y_{\text{obs}}) \notin gy.hull$, why not directly check if $g(y_{\text{obs}}) \in \text{rbd}(gy.hull)$? The answer to this question lies in the computational efficiency of certain operations. As we discuss in the rest of this chapter, calculating the H-representation of convex hulls

can be extremely expensive and should thus be avoided. Fortunately, there are alternatives, but these require a less direct route. The above pseudocode avoids calculating H-representations of hulls entirely.

Note that if it is determined that the MLE does not exist in the conventional sense, the algorithm smoothly transitions to looking for the MLE in the LCM, using the empirical face $\tilde{F}$ as the support for this new model. The approximation for $\nabla\ell(\eta)$ also switches from one for the original model to one for the LCM based on the restricted sample.

The operations in the above pseudocode that involve computational geometry are the following:

- Find the convex hull of a set of points.
- Update a convex hull to incorporate new points.
- Determine if a point is in the exterior of a convex hull.
- Determine the empirical face of a convex hull in which a points lies in the relative interior.
- Find an empirical GDOR.
- From a sample, find those that lie in a specific face.

The rest of this chapter is devoted to describing how to perform the above operations. We show in detail how to perform these in the R platform (version 2.12.1) using the `rcdd` package (version 1.1-3). While software and syntax may change, we think it is still useful to provide actual R output that illustrate the inputs and results, and hopefully illuminate the underlying issues. Most of the `rcdd` commands have been adapted from Geyer (2009b) and the vignette accompanying the package.

4.2 Convex polytope representation

As discussed in Section 2.2, a convex polyhedron C, which we may think of as the convex hull of a finite set of points V, can equivalently be expressed as the intersection of a finite collection of closed half-spaces. These representations are referred to as the V-representation and the H-representation, respectively.

4.2.1 V-representation

The `rcdd` package requires the V-representation for a finite set of points to be inputed in the following matrix form:

$$\begin{bmatrix} l & b & V \end{bmatrix} \tag{4.2}$$

where l and b are column vectors and V is a matrix such that the polyhedron is the set of points y of the form

$$y = \lambda^T V$$

where

$$\lambda_j \geq 0, \quad \text{where } l_j = 0$$

and

$$\sum_j b_j \lambda_j = 1, \quad \text{unless all } b_j = 0.$$

The entries for l and b in each row define how the linear combination of points in the same row in V should be taken:

$l = 0$ or 1. A 0 indicates nonnegative coefficients only,

$b = 0$ or 1. A 1 indicates this row should be included in the sum to one.

For bounded polytopes of a finite set of points, we take $l = 0$ and $b = 1$. We may also be interested in the V-representation for cones, in which case we set $b = 0$. This representation can also handle more abstract generators that are not of interest to us here.

4.2.2 H-representation

The `rcdd` package requires the H-representation for a finite collection of closed half-spaces be inputed in the following matrix form:

$$\begin{bmatrix} 0 & b_1 & -A_1 \\ 1 & b_2 & -A_2 \end{bmatrix} \tag{4.3}$$

where b_1 and b_2 are column vectors and A_1 and A_2 are matrices of coefficients characterizing the finite set of linear equalities and inequalities

$$\begin{aligned} A_1 x &\leq b_1 \\ A_2 x &= b_2. \end{aligned} \tag{4.4}$$

Thus a zero in the first column corresponds to an equality constraint, a one to an inequality constraint. The set of points x that satisfy the above constraints is equivalent to the convex polyhedron.

The H-representation provides a direct way to assess whether some new point q

is in or out of the convex hull simply by plugging it in for x in (4.4) and seeing if the equalities and inequalities are satisfied. We illustrate this in Section 4.3.1.

4.2.3 V-rep to H-rep and back

The `rcdd` package provides us with a simple function `scdd` to go back and forth between the H-representation and V-representation, taking as input the matrix form described by either (4.2) or (4.3) accompanied by its representation type ("H" or "V"), and returning its alternate representation, e.g.,

```
Hrep <- scdd( Vrep, representation = "V" )
```

However, the conversion of V-representation $\rightarrow$ H-representation can be a very computationally intensive operation, especially as the dimension of the polygon increases; Fukuda (2008) notes that the number of inequalities often increases exponentially in the number of dimensions. Thus despite the immediate appeal of the H-representation, we look to find more computationally efficient alternatives. In fact, the algorithm we have outlined in the pseudocode in Section 4.1 avoids any calls to `scdd`.

4.2.4 Finding the convex hull

Finding the convex hull of a set of points is a well-studied problem. It is of interest to us here because we want to "discover" the convex support C of our exponential family by repeatedly taking MCMC samples in different areas close to $g(y_{\text{obs}})$. The process by which these samples approach $g(y_{\text{obs}})$ is discussed in more detail in Section 5.5.

We may think of the convex hull in two ways: the extreme points of C, that is, a minimal V-representation, or the set of inequalities and equalities of the H-representation. Determining the extreme points of a convex hull is a much simpler problem which Fukuda (2004) refers to as "redundancy removal" as opposed to the

latter, which is the usual "convex hull problem". Although we look to utilize the V-representation only, we describe how to do each.

Extreme points of a convex hull

Given a set of non-unique points V, we can find the extreme points in two steps: first, find the unique points in this sample, and then eliminate redundant generators. The first operation can be done using the `unique` function, which is in the `base` package of R, the second requires defining a V-representation and then applying the `redundant` function in the `rcdd` package which eliminates the redundant generators. We demonstrate this using 10 samples of a two-dimensional vector:

```
> sample.gy <- t(matrix( c(
+ 19, 11,
+ 21, 14,
+ 19, 11,
+ 21, 13,
+ 21, 13,
+ 23, 21,
+ 14, 3,
+ 21, 16,
+ 15, 4,
+ 17, 8), ncol = 10) )
> sample.gy.unique <- unique( sample.gy )
> ( sample.gy.Vrep   <- cbind( 0, 1, sample.gy.unique ) )
     [,1] [,2] [,3] [,4]
[1,]    0    1   19   11
[2,]    0    1   21   14
[3,]    0    1   21   13
[4,]    0    1   23   21
[5,]    0    1   14    3
[6,]    0    1   21   16
[7,]    0    1   15    4
[8,]    0    1   17    8
> ( sample.gy.reduced <- redundant( d2q(sample.gy.Vrep), rep = "V" ) )
```

```
$output
     [,1] [,2] [,3] [,4]
[1,] "0"  "1"  "21" "13"
[2,] "0"  "1"  "23" "21"
[3,] "0"  "1"  "14" "3"
[4,] "0"  "1"  "15" "4"
attr(,"representation")
[1] "V"

$redundant
[1] 1 2 6 8

$new.position
[1] 0 0 1 2 3 0 4 0
```

In the first step, $(19, 11)$ and $(21, 13)$ each appear twice and are eliminated using `unique`. Then, a V-rep of these points is constructed. Before passing to `redundant`, we first convert it to rational number format with the function `d2q`. The `redundant` function, which can be applied to both rational and decimal format, then eliminates all but four of these points. These can be converted back to decimal format using `q2d`. We do any comparisons with rational numbers for the reasons discussed earlier.

It would have been possible to go directly from the original sample to the reduced representation using `redundant`, but using `unique` first reduces the calculations performed by `redundant`.

H-representation of a convex hull

We can convert this to an H-representation of the convex hull using the `scdd` function described previously:

```
> ( sample.gy.Hrep <- scdd( sample.gy.reduced$output, rep = "V" ) )
$output
     [,1] [,2]   [,3]   [,4]
```

```
[1,] "0"  "11"   "-1"   "1"
[2,] "0"  "37/2" "-3/2" "1"
[3,] "0"  "71"   "-4"   "1"
[4,] "0"  "-25"  "2"    "-1"
attr(,"representation")
[1] "H"
```

This route of `unique` → `redundant` → `scdd` is our preferred approach to obtain the H-representation of the convex hull from a set of points since `scdd` is the most computationally demanding operation.

Adding points to convex hull

Suppose we generate additional points and want to combine these with the previously calculated convex hull. We obtain the non-redundant points as before:

```
> sample.gy2 <- t(matrix( c(
+ 20, 14,
+ 11, 1,
+ 19, 10,
+ 12, 3,
+ 15, 5,
+ 18, 7,
+ 15, 6,
+ 20, 13,
+ 17, 9,
+ 18, 5), ncol=10) )
> sample.gy2.unique  <- unique( sample.gy2 )
> ( sample.gy2.Vrep   <- cbind( 0, 1, sample.gy2.unique ) )
      [,1] [,2] [,3] [,4]
 [1,]    0    1   20   14
 [2,]    0    1   11    1
 [3,]    0    1   19   10
 [4,]    0    1   12    3
 [5,]    0    1   15    5
 [6,]    0    1   18    7
```

```
 [7,]    0    1   15    6
 [8,]    0    1   20   13
 [9,]    0    1   17    9
[10,]    0    1   18    5
> ( sample.gy2.reduced <- redundant( d2q(sample.gy2.Vrep), rep = "V" ) )
$output
     [,1] [,2] [,3] [,4]
[1,] "0"  "1"  "20" "14"
[2,] "0"  "1"  "11" "1"
[3,] "0"  "1"  "12" "3"
[4,] "0"  "1"  "20" "13"
[5,] "0"  "1"  "18" "5"
attr(,"representation")
[1] "V"

$redundant
[1] 3 5 6 7 9

$new.position
 [1] 1 2 0 3 0 0 0 4 0 5
```

We then combine these new reduced points with the extreme points of the previously obtained convex hull. On this combined set, we apply `unique`, `redundant` as before, and, if we desire the H-representation, `scdd`.

```
> sample.gy.combined <- unique( rbind( sample.gy.reduced$output,
sample.gy2.reduced$output ) )
> ( sample.gy.combined <- redundant( sample.gy.combined, rep = "V" ) )
$output
     [,1] [,2] [,3] [,4]
[1,] "0"  "1"  "21" "13"
[2,] "0"  "1"  "23" "21"
[3,] "0"  "1"  "11" "1"
[4,] "0"  "1"  "12" "3"
[5,] "0"  "1"  "18" "5"
attr(,"representation")
[1] "V"
```

```
$redundant
[1] 3 4 5 8

$new.position
[1] 1 2 0 0 0 3 4 0 5

> ( sample.gy.Hrep <- scdd( sample.gy.combined$output, rep = "V" ) )
$output
     [,1] [,2]      [,3]    [,4]
[1,] "0"  "43"      "-8/3"  "1"
[2,] "0"  "37/4"    "-1"    "7/4"
[3,] "0"  "-21"     "2"     "-1"
[4,] "0"  "-183/11" "18/11" "-1"
[5,] "0"  "71"      "-4"    "1"
attr(,"representation")
[1] "H"
```

The separate and combined convex hulls are displayed in Figure 4.1. There are five inequalities here, one for each side of the hull.

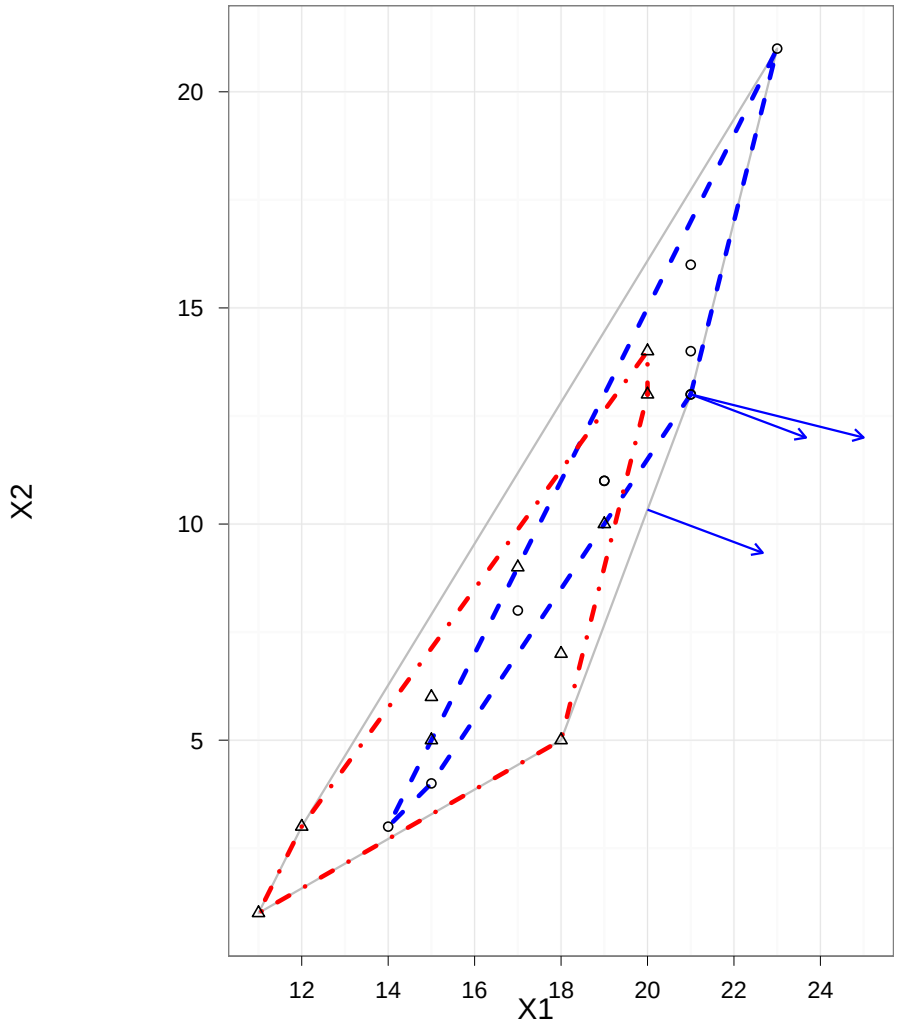

Figure 4.1: Convex hulls for two separate ten-point samples (dotted lines). The convex hull for the combined is the solid line. Boundaries of the normal cones at two points, (20,31/3), and (21,13), shifted to start at those two points, are depicted as arrows.

4.3 Point in exterior, interior, and boundary of a convex hull

Determining whether an inquiry point—typically $g(y_{\text{obs}})$ in our application—lies in the exterior, interior, or boundary of a convex hull is of central importance to our

method. We present two different approaches here.

4.3.1 Using H-representation of convex hull

We first describe the more intuitive approach, which is also less preferable because it requires calculating the H-representation. We illustrate it using the combined convex hull found in the previous section. We pick three points—(20,10), (20,12), (21,13)—exterior, interior, and boundary points, according to Figure 4.1. To the vector comprising these points, we multiply by the matrix A from the H-representation of the hull and subtract b. The sign of the resulting quantities tells us if a particular inequality has been satisfied: positive indicates that it has not, negative indicates that it is strictly satisfied, and a zero indicates that it is met with an equality. Rational arithmetic in such comparisons is essential.

```
> x <- rbind( c(20,10), c(20,12), c(21,13) )
> l <- sample.gy.Hrep$output[ ,1]
> b <- sample.gy.Hrep$output[ ,2]
> a <- sample.gy.Hrep$output[ ,-c(1,2)]
> a <- qneg( a )
> ( axb <- qmatmult( a, t(x) ) )
     [,1]     [,2]     [,3]
[1,] "1/3"    "-5/3"   "0"
[2,] "-27/4"  "-41/4"  "-11"
[3,] "-9"     "-7"     "-8"
[4,] "-67/11" "-45/11" "-52/11"
[5,] "-1"     "-3"     "0"
> axb <- sweep( axb, 1, b, FUN=qmq ) # subtract b
> apply( axb, 2, function(pair) max(qsign(pair)) )
[1]  1 -1  0
```

For each inquiry point, we need only consider the largest sign of the five quantities to classify the point as exterior, interior, or boundary. The first point (20,10) which we observed to be an exterior point satisfies four of the five inequalities, but violates

the first with a positive result of $\frac{1}{3}$. Thus we get a positive sign (represented by a 1), indicating an exterior point. The second point (20,12) which we observed to be an interior point is strictly negative on all five inequalities, indicating that they have all been strictly satisfied, resulting in a negative sign (represented by a -1). Finally, the point (21,13) which we observed to be on the boundary satisfies three inequalities strictly, and meets two with equality, returning a 0 for the largest sign. This agrees with our observation that (21,13) is an extreme point of the hull and thus on the boundary.

4.3.2 Strongly separating hyperplane

Fukuda (2008) describes a more efficient way to determine whether a single inquiry point q is in or out of the convex hull of many other points p_i. This approach looks for a strongly separating hyperplane H between a point q and the convex hull of a set of points $\{p_i : i \in I\}$ where

$$H = \{x : z^T x = z_0\}$$

where z is a vector and z_0 a scalar satisfying

$$\begin{aligned} z^T p_i &< z_0 \quad i \in I \\ z^T q &> z_0. \end{aligned}$$

This can be accomplished by solving the linear programming problem

$$\text{maximize } f(z_0, z) = z^T q - z_0 \tag{4.5}$$

$$\begin{aligned} \text{subject to } z^T p_i - z_0 &\leq 0, \quad i \in I \\ z^T q - z_0 &\leq 1, \end{aligned} \tag{4.6}$$

where the last inequality is artificially added so that the linear program has a bounded solution (Fukuda, 2008; Geyer and Meeden, 2009). If the maximized value of f is strictly positive, then a strongly separating hyperplane exists and q is in the exterior of the convex hull. Otherwise, q is on the boundary or the interior. Note that it does not distinguish between these latter two cases.

A general linear programming routine `lpcdd` which uses rational arithmetic is provided in the `rcdd` package and can be applied to this problem. The function maximizes the linear function $x \mapsto a^T x$ subject to the constraint that x lie in the polyhedral convex set having the H-representation specified. We demonstrate its use on an exterior, boundary, and interior point, using the convex hull we constructed in the previous section. Note that `lpcdd` can be applied to the V-representation of points, eliminating the need for a call to `scdd`; the H-representation we refer to here is the one we construct directly to represent the constraints (4.6) rather than a conversion of a convex hull V-representation. Some care needs to be taken in constructing this input, as illustrated in the following examples.

Exterior point

Let $q = (20, 10)$, on the exterior.

```
> q <- c(20,10)
> ( hull.pts <- sample.gy.combined$output[ , -c(1:2) ] ) # get points in hull
     [,1] [,2]
[1,] "21" "13"
[2,] "23" "21"
[3,] "11" "1"
[4,] "12" "3"
[5,] "18" "5"
> hrep <- cbind( 0, 0, 1, qneg(hull.pts) ) # s.t. 1st ineq
> ( hrep <- rbind( hrep, c(0, 1, 1, -q) ) ) # 2nd (artificial) ineq
     [,1] [,2] [,3] [,4]  [,5]
[1,] "0"  "0"  "1"  "-21" "-13"
```

```
[2,] "0"  "0"  "1"  "-23" "-21"
[3,] "0"  "0"  "1"  "-11" "-1"
[4,] "0"  "0"  "1"  "-12" "-3"
[5,] "0"  "0"  "1"  "-18" "-5"
[6,] "0"  "1"  "1"  "-20" "-10"
> lpcdd( hrep, d2q( c(-1, q) ), minimize = FALSE )$optimal.value
[1] "1"
```

The `hrep` variable above holds the coefficients for z_0 (scalar) and z (vector), in that order, as described by (4.6), where the last row is the artificially added second constraint. Clearly only the V-representation of the hull is used. The objective function to be maximized is similarly passed the coefficients for (z_0, z), as described by (4.5). The maximum value of 1 is strictly positive so $(20, 10)$ is on the exterior, as expected.

Interior and boundary point

We applied this approach to $(20, 12)$, an interior point, and again to $(21, 13)$, a boundary point. In both cases, the maximum value of f is 0, indicating that these points are not in the exterior. However, this approach does not help us distinguish between interior and boundary points.

4.4 Linearity

So far we have advocated using V-representations of convex hulls and applying the `lpcdd` function to determine whether or not an inquiry point is in the exterior of the hull. To subsequently determine if a non-exterior point is on the boundary or the interior, one could apply the H-representation approach described in Section 4.3.1, but as noted, this is computationally inefficient. Here we describe another approach that is not only more computationally efficient but also provides us with additional useful information in the context of our MLE search objective.

Let V be the set of extreme points of a polyhedral convex hull and let q be a point of inquiry.

Define

$$W = \{v - q : v \in V\},$$

which generates $T_{\text{con}(V)}(q)$. The *linearity* of a set of points W is

$$L = \{w \in W : -w \in T_{\text{con}(V)}(q)\}.$$

If $\text{con}(V) = C$, the polyhedral convex support of an exponential family, and $q = g(y_{\text{obs}})$, then the set L here is the same L as in Theorem 2.7. By Theorem 2.8 and Corollary 2.4, if a GDOR exists, then L determines the convex support of the LCM, $C \cap H$, by

$$C \cap H = C \cap (g(y_{\text{obs}}) + \text{span}\, L).$$

4.4.1 The `linearity` function

The `linearity` function in the `rcdd` takes as input rays in the form of W above and identities from these the generators that characterize a vector subspace. It can take as input the V- or H-representation and solves the following linear programming problem for each $w \in W$:

$$\begin{aligned} &\text{maximize } \langle w, \delta \rangle \\ &\text{subject to } \langle v, \delta \rangle \geq 0, \quad v \in W \setminus \{w\}, \end{aligned}$$

where δ is the state vector of the linear programming problem. Geyer (2009a) shows that $w \in L$ if and only if the optimal value of the linear programming problem is

nonpositive. The `linearity` function returns the row index for every $w \in W$ that returns a nonpositive value. We demonstrate its functionality on an interior point and two points on the boundary, using the same convex hull as before.

Interior point

Let $q = (20, 12)$, a point on the interior.

```
> q <- c(20,12)
> ( W <- sweep( hull.pts, 2, q, FUN = qmq ) )
     [,1] [,2]
[1,] "1"  "1"
[2,] "3"  "9"
[3,] "-9" "-11"
[4,] "-8" "-9"
[5,] "-2" "-7"
> W <- cbind( 0, 0, W )
> ( lin.active <- linearity( W, rep = "V" ) )
[1] 1 2 3 4 5
> ( L <- W[ lin.active, , drop=FALSE] )
     [,1] [,2] [,3] [,4]
[1,] "0"  "0"  "1"  "1"
[2,] "0"  "0"  "3"  "9"
[3,] "0"  "0"  "-9" "-11"
[4,] "0"  "0"  "-8" "-9"
[5,] "0"  "0"  "-2" "-7"
```

Here, `linearity` returned the indices, assigned to `lin.active`, for all rays inputted. Adding back q to the rays will return the original extreme points of the convex hull. Thus the resulting face here is the original hull.

Boundary point, one-dimensional face

Let $q = (20, 31/3)$, which we found by plugging 20 into one of the H-representation inequalities of the hull. This point falls on the line $y = \frac{8}{3}x - 43$ between the extreme

points $(18, 5)$ and $(21, 13)$. Of course in the ERGM setting we consider, it is impossible to generate a non-integer count of network structures, but we ignore this issue here.

```
> q <- c("20","31/3")
> ( W <- sweep( hull.pts, 2, q, FUN = qmq ) )
     [,1] [,2]
[1,] "1"  "8/3"
[2,] "3"  "32/3"
[3,] "-9" "-28/3"
[4,] "-8" "-22/3"
[5,] "-2" "-16/3"
> W <- cbind( 0, 0, W )
> ( lin.active <- linearity( W, rep = "V" ) )
[1] 1 5
> ( L <- W[ lin.active, , drop=FALSE] )
     [,1] [,2] [,3] [,4]
[1,] "0"  "0"  "1"  "8/3"
[2,] "0"  "0"  "-2" "-16/3"
> hull.pts[ lin.active, , drop=FALSE ]
     [,1] [,2]
[1,] "21" "13"
[2,] "18" "5"
```

The `linearity` function identifies the rays in opposite directions running along $y = \frac{8}{3}x + 43$, starting at $(20, 31/3)$. From this, we identify a one-dimensional face, the line segment between $(18, 5)$ and $(21, 13)$ in which $(20, 31/3)$ lies in the relative interior.

Boundary point, zero-dimensional face

Let $q = (21, 13)$, an extreme point of the face.

```
> q <- c(21,13)
> ( W <- sweep( hull.pts, 2, q, FUN = qmq ) )
     [,1]  [,2]
```

```
[1,] "0"   "0"
[2,] "2"   "8"
[3,] "-10" "-12"
[4,] "-9"  "-10"
[5,] "-3"  "-8"
> W <- cbind( 0, 0, W )
> ( lin.active <- linearity( W, rep = "V" ) )
[1] 1
> ( L <- W[ lin.active, , drop=FALSE] )
     [,1] [,2] [,3] [,4]
[1,] "0"  "0"  "0"  "0"
```

In this example, `linearity` identifies only the zero vector, which trivially satisfies the definition of a subspace. Thus the point $(21, 13)$ is a zero-dimensional face, lying in the relative interior of itself.

4.4.2 Finding an empirical face

We began this chapter under the more realistic setting that we do not know C and are generating MCMC samples in order to uncover its geometry. The convex hull we have to work with then is that of our accumulated samples. After ascertaining that $g(y_{\text{obs}})$ is either in the relative interior or boundary of this hull, we apply `linearity` to the extreme points of it to identify the empirical face $\tilde{F}$ in which $g(y_{\text{obs}})$ lies in the relative interior. If `linearity` returns all indices, then $g(y_{\text{obs}})$ is a relative interior point, not just of the convex hull of our samples, but of C. Otherwise, `linearity` returns the indices that characterize the face $\tilde{F}$ in the boundary of our sample hull; more work is needed to determine if $\tilde{F}$ is in the relative boundary of C.

4.5 Normal and tangent cones

We may be interested in the normal cone and tangent cones after it has been determined that q is on the boundary of a hull. The approaches described in this section

rely on an H-representation of the convex hull and so we avoid using these in our algorithm. Nevertheless, these methods are useful to have for comparison purposes.

The H-representation of a polyhedral convex set C can be presented as

$$C = \{x : \langle a_i, x\rangle \leq b_i, i \in I \text{ and } \langle a_i, x\rangle = b_i, i \in E\}.$$

The tangent cone at a particular point $x \in C$ can then simply be gleaned from this H-representation by

$$T_C(x) = \{y : \langle a_i, y\rangle \leq 0, i \in A \text{ and } \langle a_i, y\rangle = 0, i \in E\}$$

where

$$A = \{i \in I : \langle a_i, x\rangle = b_i\}$$

is the active set of inequality constraints (the inactive constraints have been dropped, and now x is the new origin). We may then use the polarity of the tangent and normal cones to read the V-representation of the normal cone off the H-representation of the tangent cone:

$$N_C(x) = \text{pos}(\{a_i : i \in A \cup E\} \cup \{-a_i : i \in E\}).$$

We apply this approach to the same boundary points and convex hull.

4.5.1 Boundary point, one-dimensional face

Let $q = (20, 31/3)$, a point on the line segment of the boundary between $(18, 5)$ and $(21, 13)$.

```
> q <- as.matrix( c("20","31/3") )
> sample.gy.Hrep
$output
     [,1] [,2]      [,3]    [,4]
[1,] "0"  "43"      "-8/3"  "1"
[2,] "0"  "37/4"    "-1"    "7/4"
[3,] "0"  "-21"     "2"     "-1"
[4,] "0"  "-183/11" "18/11" "-1"
[5,] "0"  "71"      "-4"    "1"
attr(,"representation")
[1] "H"

> l <- sample.gy.Hrep$output[ ,1]
> b <- sample.gy.Hrep$output[ ,2]
> a <- sample.gy.Hrep$output[ ,-c(1,2)]
> ( active <- qpq( qmatmult( a, q ), b ) == "0" )
      [,1]
[1,]  TRUE
[2,] FALSE
[3,] FALSE
[4,] FALSE
[5,] FALSE
> tcone.Hrep <- sample.gy.Hrep$output[ active, , drop=FALSE]
> tcone.Hrep[ ,2 ] <- "0"
> tcone.Hrep
     [,1] [,2] [,3]   [,4]
[1,] "0"  "0"  "-8/3" "1"
> # normal cone
> ( ncone.Vrep <- qneg( tcone.Hrep ) )
     [,1] [,2] [,3]  [,4]
[1,] "0"  "0"  "8/3" "-1"
```

Since this point lies in the relative interior of the line segment, the normal cone consists of a single direction, $(8/3, -1)$. This direction (shifted to originate at $(20, 31/3)$) is depicted as an arrow in Figure 4.1.

4.5.2 Boundary point, zero-dimensional face

We apply the same methodology to the point $(21, 13)$, an extreme point of the hull. In this case, the method returns two directions, $(8/3, -1)$ and $(4, -1)$, which are the boundaries of the normal cone at this point. They are depicted as arrows in Figure 4.1 (after shifting to originate at $(21, 13)$).

4.6 Calculating a GDOR

A GDOR δ is a vector in the relative interior of $N_C(g(y_{\text{obs}}))$ when $N_C(g(y_{\text{obs}}))$ is not a vector subspace. By Theorem 2.7, δ is a GDOR if and only if

$$\begin{aligned} \langle w, \delta \rangle &= 0 \quad w \in L \\ \langle w, \delta \rangle &< 0 \quad w \in W \setminus L \end{aligned}$$

where L is the linearity set and W is the set of generators of the tangent cone $T_C(g(y_{\text{obs}}))$. Geyer (2009a) describes a linear programming problem based on these criteria to find one such GDOR:

$$\text{maximize } \epsilon \tag{4.7}$$

$$\begin{aligned} \text{subject to } \epsilon &\leq 1 \\ \langle v, \delta \rangle &= 0, \quad v \in L \\ \langle v, \delta \rangle &\leq -\epsilon, \quad v \in W \setminus L, \end{aligned} \tag{4.8}$$

where δ is a vector, ϵ a scalar, and (δ, ϵ) the state vector of the linear program. We can again use `lpcdd` to solve this problem, where constructing the H-representation to represent (4.8) requires some care. We demonstrate this using the same convex hull and inquiry points as before, treating the convex hull as the convex support C

and the points q as $g(y_{\text{obs}})$.

4.6.1 A GDOR when $g(y_{\text{obs}})$ on one-dimensional face

Let $q = (20, 31/3)$, a point on the line segment of the boundary between $(18, 5)$ and $(21, 13)$.

```
> q <-  c("20","31/3")
> d <- length( q )
> ( hull.pts <- sample.gy.combined$output[ , -c(1:2) ] ) # get points in hull
     [,1] [,2]
[1,] "21" "13"
[2,] "23" "21"
[3,] "11" "1"
[4,] "12" "3"
[5,] "18" "5"
> ( W <- sweep( hull.pts, 2, q, FUN = qmq ) )
     [,1] [,2]
[1,] "1"  "8/3"
[2,] "3"  "32/3"
[3,] "-9" "-28/3"
[4,] "-8" "-22/3"
[5,] "-2" "-16/3"
> W <- cbind( 0, 0, W )
> ( lin.active <- linearity( W, rep = "V" ) )
[1] 1 5
> ( objv <- d2q( c( rep(0,d), 1) ) )
[1] "0" "0" "1"
> hrep <- cbind( qneg(W), 0 )
> hrep[ lin.active, 1 ] <- 1
> hrep[ -lin.active, ncol(hrep) ] <- "-1"
> hrep <- rbind( hrep, c( 0, 1, rep(0,d), -1 ) )
> hrep
     [,1] [,2] [,3] [,4]    [,5]
[1,] "1"  "0"  "-1" "-8/3"  "0"
[2,] "0"  "0"  "-3" "-32/3" "-1"
[3,] "0"  "0"  "9"  "28/3"  "-1"
```

```
[4,] "0"  "0"  "8"  "22/3"  "-1"
[5,] "1"  "0"  "2"  "16/3"  "0"
[6,] "0"  "1"  "0"  "0"     "-1"
> lpout <- lpcdd( hrep, objv, minimize = FALSE )
> ( gdor  <- lpout$primal.solution[ -(d+1) ] )
[1] "1"    "-3/8"
```

We need to first apply `linearity` to separate the set W into points in the linearity set L, to which equality constraints are applied, and $W \setminus L$, to which inequality constraints are applied. The `hrep` variable then holds the coefficients for δ (vector) and ϵ (scalar), as described by (4.8). Note that the first column for the linearity set needs to be set to 1 to denote an equality constraint. The GDOR of $(1, -\frac{3}{8})$ agrees with the one found earlier in Section 4.5.1.

4.6.2 A GDOR when $g(y_{\mathbf{obs}})$ on zero-dimensional face

Applying this same approach to $(21, 13)$, an extreme point of the hull, we find that $(2, -5/8)$ is a GDOR. In Section 4.5.2, we identified the directions $(8/3, -1)$ and $(4, -1)$ as the boundaries of the normal cone at this same point. Our GDOR can be rescaled to $(16/5, -1)$ and hence falls nicely in the relative interior of this normal cone. Note that our approach gives us only one of the possible GDORs and no information about the boundaries of the relevant normal cone.

4.6.3 Empirical GDORs

The examples in this section assumed the convex hull we had previously defined was the convex support C of an exponential family model and the various inquiry points were values for $g(y_{\text{obs}})$. If the convex support C was not our convex hull, the determined directions may still be useful. We refer to these as empirical GDORs and denote them $\tilde{\delta}$.

4.7 Subsampling

Given an empirical face $\tilde{F}$ characterized by an empirical GDOR $\tilde{\delta}$, we may need to find points in a new sample $g(Y_1), \ldots, g(Y_m)$ that fall on this face. This is straightforward to do since $\tilde{\delta}$ is orthogonal to any point in that face: we calculate the dot product of all the new sample points (minus the observed value) with $\tilde{\delta}$ and take the points for which this evaluate to zero. That is, take $g(Y_i)$ for which

$$\langle g(Y_i) - g(y_{\text{obs}}), \tilde{\delta} \rangle = 0.$$

We demonstrate this at the point $(20, 31/3)$, using $\tilde{\delta} = (1, -3/8)$ that we obtained in Section 4.6.1.

```
> gdor <- as.matrix(c( "1", "-3/8" ) )
> q <- c( "20", "31/3" )
> sample.gy.3 <- t(matrix( c(
+ "17", "7/3",
+ "18", "5",
+ "19", "23/3",
+ "20", "31/3",
+ "21", "13",
+ "22", "47/3",
+ "20", "10",
+ "20", "32/3",
+ "19", "8",
+ "19", "25/3"
+ ), ncol = 10) )
> W <- sweep( sample.gy.3, 2, q, FUN = qmq )
> qmatmult( W, gdor )
      [,1]
 [1,] "0"
 [2,] "0"
 [3,] "0"
 [4,] "0"
 [5,] "0"
```

```
 [6,] "0"
 [7,] "1/8"
 [8,] "-1/8"
 [9,] "-1/8"
[10,] "-1/4"
```

This method identifies the first six points as being orthogonal to $\tilde{\delta}$. Recalling that our face in this example is the line segment between $(18, 5)$ and $(21, 13)$, this is correct for the second through fifth points, but the first and sixth points, $(17, 7/3)$ and $(22, 47/3)$, are *not* in this face since they fall outside the line segment.

This is not cause for concern in the context of our algorithm since it is not possible to generate any MCMC sample points outside of C. In fact, any new points we find in a sample that meet this criteria are points that should have been included in the empirical face $\tilde{F}$ but were missed; the definition of $\tilde{F}$ should then be updated to include them or it may result in the wrong convex support for the LCM.

If in our algorithm we find a new sample point w for which $\langle w, \tilde{\delta} \rangle > 0$, then it means that the empirical $\tilde{\delta}$ we calculated is not a true GDOR of the model, and in turn, that the empirical face $\tilde{F}$ we had found is not in the relative boundary of the convex support C. The new point w is "closer" to the boundary than at least some of the extreme points characterizing our empirical face $\tilde{F}$. This may be useful in catching the problem described in Section 5.5.5.

When it is determined that an empirical face $\tilde{F}$ is in fact the support of the new LCM, then the subsample of the draws from the original model with parameter value η restricted to the support is a sample from the LCM with parameter value η. This is a direct consequence of Theorem 2.6, which tells us that

$$P_{LCM,\eta}(\,\cdot\,) = P_{\eta}(\,\cdot \mid g(Y) \in H).$$

Chapter 5

Examples

We apply our algorithm to several different settings:

- Logistic regression where the MLE exists. The gradient $\nabla \ell(\beta)$ can be calculated exactly in this problem and thus our algorithm is guaranteed to converge to the MLE by Theorem 3.2. We compare parameter estimates to those attained by the `glm` function in R and compare performance to stochastic approximation. We choose a starting value for which Newton-Raphson fails.

- Ising model where the MLE exists. The gradient $\nabla \ell(\eta)$ must be approximated in this setting via MCMC. Although our algorithm is not guaranteed to converge in theory, it is still effective in practice. We compare the performance to MCMC stochastic approximation.

- Sampson's monastery data. We revisit the example from Section 1.4.3 which showed how MCMC-MLE struggles to converge when the starting value is far from the true MLE. The model used here is the simple Erdős-Rényi-Gilbert model described in Section 1.3.1 with network statistic $g(y)$ equal to the total number of edges present.

- An adolescent friendship network data set of 1,461 students in grade 7–12 derived from the National Longitudinal Study of Adolescent Health (Resnick,

Bearman, Blum, Bauman, Harris, Jones, Tabor, Beunning, Sieving, Shew, Ireland, Bearinger and Udry, 1997). Goodreau et al. (2008) fit a more realistic model to this large network data set, involving covariate factors as well as network structures. We compare our results to those attained by Goodreau et al. who use MCMC-MLE starting from a very good MPLE estimate.

- ERGM with two-dimensional sufficient statistics on a 9-node undirected network. Here we explore cases where the MLE may not exist in the conventional sense. Our algorithm finds the MLE if it exists, and finds the LCM MLE when it does not. We then construct one-sided confidence intervals for the natural parameters.

5.1 Example: logistic regression

We illustrate the application of our algorithm in the case of a logistic regression with a starting point far from the solution. In such a case, the Hessian matrix is often near-singular and algorithms such as Newton-Raphson which rely on it will fail. For classical SA with step size $1/k$, the magnitudes of the updates diminishes too quickly for the parameter estimates to approach the MLE in a reasonable amount of time.

The response vector Y has components that are Bernoulli trials with mean vector p. The natural parameter is $\theta_i = \log\left(\frac{p_i}{1-p_i}\right)$, which is modeled component-wise as a linear function of the predictors $1, x_1, \ldots, x_q$, so that

$$\theta_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_q x_{q\,i} = \beta^T x_i \qquad i = 1, \ldots, n$$

where $\beta = (\beta_0, \ldots, \beta_q)$ and $x_i = (1, x_{1i}, \ldots, x_{qi})$.

Defining the model matrix M to be the $n \times (q+1)$ matrix with the x_i as rows, we can express $\theta = M\beta$. This in turn allows us to reparameterize the exponential family

as one with β as the natural parameter vector and $M^T y$ the vector of statistics with log likelihood

$$\ell(\beta) = \beta^T (M^T y) - c(\beta),$$

where y is the vector of observed Bernoulli responses. By (2.1), the gradient is

$$\nabla \ell(\beta) = M^T y - \mathrm{E}_\beta(M^T Y) = M^T (y - \mathrm{E}_\beta(Y)),$$

where $\mathrm{E}_\beta(Y) = \frac{1}{1+\exp(-M\beta)}$ can be calculated exactly. This allows us to directly apply Theorem 3.2.

Suppose we specify our true parameter value to be $\beta = (0, 2, 2, 1, 1, 0, 0, 0)$ and use 100 independent draws from a correlated multivariate normal distribution centered at 0 as our predictors to generate 100 independent Bernoulli trials. Fitting these data using the R function `glm`, we find the MLE of β to be

$$\hat{\beta}_{\mathrm{MLE}} = (0.635, 5.949, 1.273, 0.180, 1.006, 1.536, -2.252, -0.472),$$

where the disparity to the true value of β results from a relatively small sample size of $n = 100$. We use $\beta_0 = (5, 4, 3, 2, 1, 0, -1, -2)$ for the starting point for our line search algorithm, a point for which Newton-Raphson fails due to a nearly singular Hessian matrix.

We measure the performance of our algorithm in terms of the total number of iterations used, where each iteration requires evaluation of the gradient, $\nabla \ell(\beta_k + \alpha_k p_k)$. Typically, several iterations take place in an inner loop to find a step size α_k that meets the curvature condition (3.3), a process that grows increasingly difficult as the estimates near the MLE since the rightmost term in (3.3) gets smaller in magnitude. Once an acceptable step size is found, the parameter estimate β_k is updated and a

new search direction is determined, requiring another evaluation of the gradient.

Our algorithm took 54 iterations over 20 different search directions to obtain $\|\nabla \ell(\beta_k)\| < 0.01$ and arrive at an estimate for the MLE that differs from the `glm` result by 0.0117 in Euclidean distance (See Table 5.1). Using the Polak-Ribière conjugate gradient method described in the previous section resulted in comparably sharp MLE estimates (see Table 5.1) in fewer iterations—28 over 11 search directions—a noticeable improvement.

We also applied SA with step size $1/k$ (setting $A = 1$, $B = 0$ in (1.12)) from the same starting point β_0. The choice of constants A and B in the step size is of course not likely to be optimal; however, we want to apply SA without trial and error experimentation. After 10,000 iterations, the parameter estimates look nothing at all like the MLE (See Table 5.1). The starting point β_0 is so far from the MLE and the step sizes so small that the algorithm does not converge in a reasonable amount of time. Table 5.2 shows the first 20 step sizes used by SA and our line search. Our line search continues to use step sizes of relatively large magnitude even well into the process. It should be noted that these 20 step sizes correspond to the first 20 iterations of SA but all 54 iterations of our line search algorithm since it spends several iterations finding an acceptable step size for each update.

Table 5.1: Comparison of MLEs for β in Example 1: MLE = `glm`, Steep = line search using steepest ascent, CG = line search using conjugate gradient, and SA = SA with step size = $1/k$, terminated at 10,000 iterations, n = number of iterations. Our proposed algorithm arrives at nearly identical MLE estimates to `glm`.

	n	$\beta[1]$	$\beta[2]$	$\beta[3]$	$\beta[4]$	$\beta[5]$	$\beta[6]$	$\beta[7]$	$\beta[8]$
True β		0.000	2.000	2.000	1.000	1.000	0.000	0.000	0.000
$\hat{\beta}_{\text{MLE}}$		0.635	5.949	1.273	0.180	1.006	1.536	−2.252	−0.472
$\hat{\beta}_{\text{Steep}}$	54	0.633	5.938	1.272	0.181	1.005	1.535	−2.249	−0.470
$\hat{\beta}_{\text{CG}}$	28	0.631	5.936	1.272	0.181	1.003	1.532	−2.244	−0.470
$\hat{\beta}_{\text{SA}}$	10^4	1.280	10.619	5.588	4.005	2.478	−7.153	1.255	0.264

Table 5.2: The first 20 step sizes used by SA (with step size $1/k$) and our algorithm for Example 1. The step sizes used by our algorithm do not diminish like $1/k$.

k	$\alpha_{\text{SA}} = 1/k$	α_{Steep}	α_{CG}
1	1.000	0.192	0.192
2	0.500	0.319	0.319
3	0.333	0.403	0.416
4	0.250	0.353	0.561
5	0.200	0.380	0.491
6	0.167	0.333	1.092
7	0.143	0.420	0.359
8	0.125	0.307	0.314
9	0.111	0.442	0.275
10	0.100	0.283	0.318
11	0.091	0.483	0.278
12	0.083	0.241	-
13	0.077	0.745	-
14	0.071	0.203	-
15	0.067	1.224	-
16	0.063	0.173	-
17	0.059	2.510	-
18	0.056	0.195	-
19	0.053	0.944	-
20	0.050	0.173	-

5.2 Example: Ising model

In this example, we apply our gradient-based line search algorithm to an Ising model (Ising, 1925) on a toroidal square lattice. Ising models are exponential families where each entry in the square lattice takes the value of either zero or one. A realized sample is shown in Figure 5.1. The sufficient statistic vector is two-dimensional, comprising the number of entries with value one and the number of "neighbor" entries with the same value. Entries are considered "neighbors" if they are adjacent to one another

horizontally or vertically (but not diagonally).

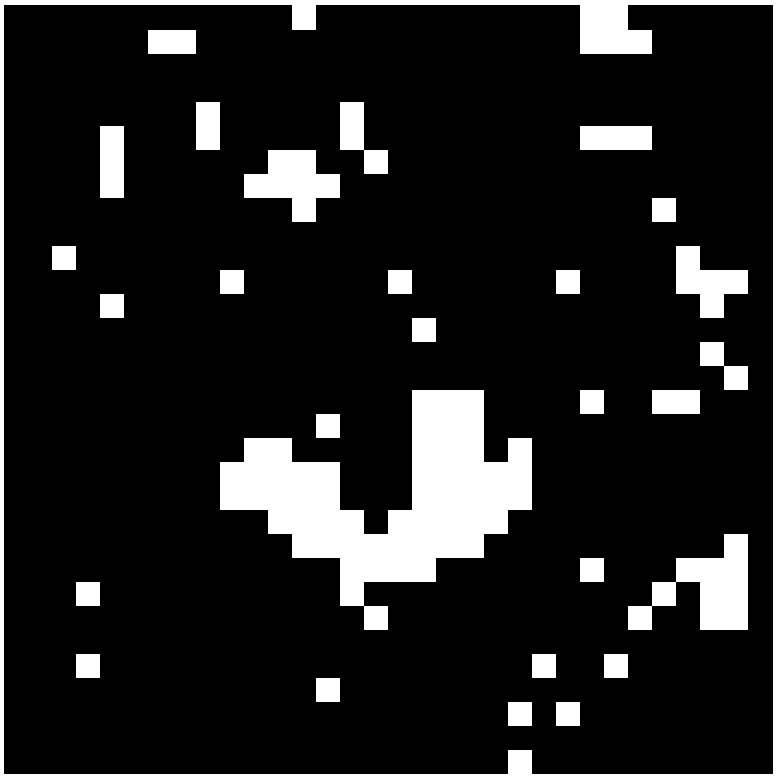

Figure 5.1: A realized sample from an Ising model on a 32×32 lattice with $\eta = \big(0, \log(1+\sqrt{2})\big)$. This value of η corresponds to the phase transition point, where the sample images are mostly one color with small but significant portions of the other color. There is no preference for the dominant color to be white or black.

Here we describe the toroidal square lattice as an $n \times n$ matrix Y and each entry as Y_{ij}, where i and j take values in $1, \ldots, n$ considered as a cyclical set (addition is

done modulo n). The sufficient statistic, $g(y)$, has components:

$$\begin{aligned} g_1(y) &= \sum_{i=1}^{n} \sum_{j=1}^{n} I(Y_{ij} = 1), \\ g_2(y) &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \big[I(Y_{ij} = Y_{i-1,j}) + I(Y_{ij} = Y_{i,j-1}) \\ &\qquad + I(Y_{ij} = Y_{i+1,j}) + I(Y_{ij} = Y_{i,j+1}) \big], \end{aligned}$$

where $I(\cdot)$ denotes the indicator function taking logical expressions to the numbers zero and one, false expressions to zero and true expressions to one.

Because of the interdependence of neighboring entries in the lattice, there is no closed form expressing $\nabla \ell(\eta)$ as in the logistic example. Instead, we need to approximate $\nabla \ell(\eta)$ using MCMC as described by (3.17). As discussed in Section 3.5, Theorem 3.2 cannot be applied directly, but as we demonstrate here, satisfactory estimates are still attained. The MCMC draws are performed here using the Swendsen-Wang algorithm (Swendsen and Wang, 1987; Wang and Swendsen, 1990), available in the contributed R package `potts` (Geyer and Johnson, 2010).

We choose $\eta = \big(0, \log(1 + \sqrt{2})\big)$ to generate a 32×32 lattice, which we will use as our observed data (Figure 5.1). This value for η is of particular interest because it corresponds to the phase transition point (Potts, 1952) and has been shown to be difficult to estimate (Geyer, 1990). In order to get a good estimate of the MLE to which we can compare our algorithm's results, we use 10 iterations of MCMC Newton-Raphson (Penttinen, 1984) starting from the true value of η so that we can be confident it will converge.

We apply our line search algorithm to this data using a far off initial value of $\eta^{(0)} = (2, 0.001)$ and a fixed MCMC sample size of 10,000. Our algorithm used 62 iterations (gradient evaluations) over 17 search directions to get $\|\nabla \ell(\eta_k)\| < 0.005$ and arrive at an estimate of the MLE that differs from Newton-Raphson by 0.0037 (see Table 5.3).

Using the Polak-Ribière conjugate gradient method resulted in comparably sharp MLE estimates using 45 iterations over 7 search directions. The total MCMC sample sizes used were $62 \times 10,0000 = 620,000$ and $45 \times 10,0000 = 450,000$, respectively.

Table 5.3: Comparison of MLEs for η in Example 2: MLE = Newton-Raphson starting from the true η, Steep = line search using steepest ascent, CG = line search using conjugate gradient, and SA = SA with step size = $1/k$. All algorithms converged.

	MC Samples (thousands)	$\eta[1]$	$\eta[2]$
True η		0.000	0.881
$\hat{\eta}_{\text{MLE}}$		−0.007	0.896
$\hat{\eta}_{\text{Steep}}$	620	−0.011	0.895
$\hat{\eta}_{\text{CG}}$	450	−0.008	0.895
$\hat{\eta}_{\text{SA}}$	1368	−0.010	0.895

We also applied MCMC SA, again with step size $1/k$ from the same starting point $\eta^{(0)}$, and used a MCMC sample size of 1,000 for gradient calculation. Here SA converged in 1368 iterations or 1,368,000 MC samples, comparable to our algorithm (see Table 5.3). Table 5.4 shows the first 17 step sizes used by SA and our line search. The step sizes used by our line search are initially very small compared to $1/k$, but stay in a range of about 1/300 to 1/3000. So, the $1/k$ step size used by SA in fact occasionally satisfies our curvature condition when k is large.

An interesting side note about the MLE distribution for Ising models and their generalization to more than two colors called Potts models is that for these moderate size lattices (including 32×32 or even 100×100), the distribution appears to be skewed and biased (Okabayashi et al., 2011). In the sample run above, the MLE happens to be very close to the true parameter value. However, by calculating MLEs for 1000 different 32×32 observed lattices, we find the empirical average for the $\eta[2]$ component to be 0.856 with a standard error of 0.036, compared to the true value of 0.881. The histogram for the $\eta[2]$ component of the MLE is shown in Figure 5.2.

Table 5.4: The first 17 step sizes used by SA (with step size $1/k$) and our algorithm for Example 2. The step sizes used by our algorithm are initially much smaller than $1/k$.

k	$\alpha_{\text{SA}} = 1/k$	α_{Steep}	α_{CG}
1	1.0000	0.0029	0.0029
2	0.5000	0.0005	0.0005
3	0.3333	0.0017	0.0017
4	0.2500	0.0013	0.0045
5	0.2000	0.0017	0.0007
6	0.1667	0.0011	0.0002
7	0.1429	0.0021	0.0015
8	0.1250	0.0009	
9	0.1111	0.0020	
10	0.1000	0.0007	
11	0.0909	0.0018	
12	0.0833	0.0006	
13	0.0769	0.0013	
14	0.0714	0.0006	
15	0.0667	0.0007	
16	0.0625	0.0003	
17	0.0588	0.0013	

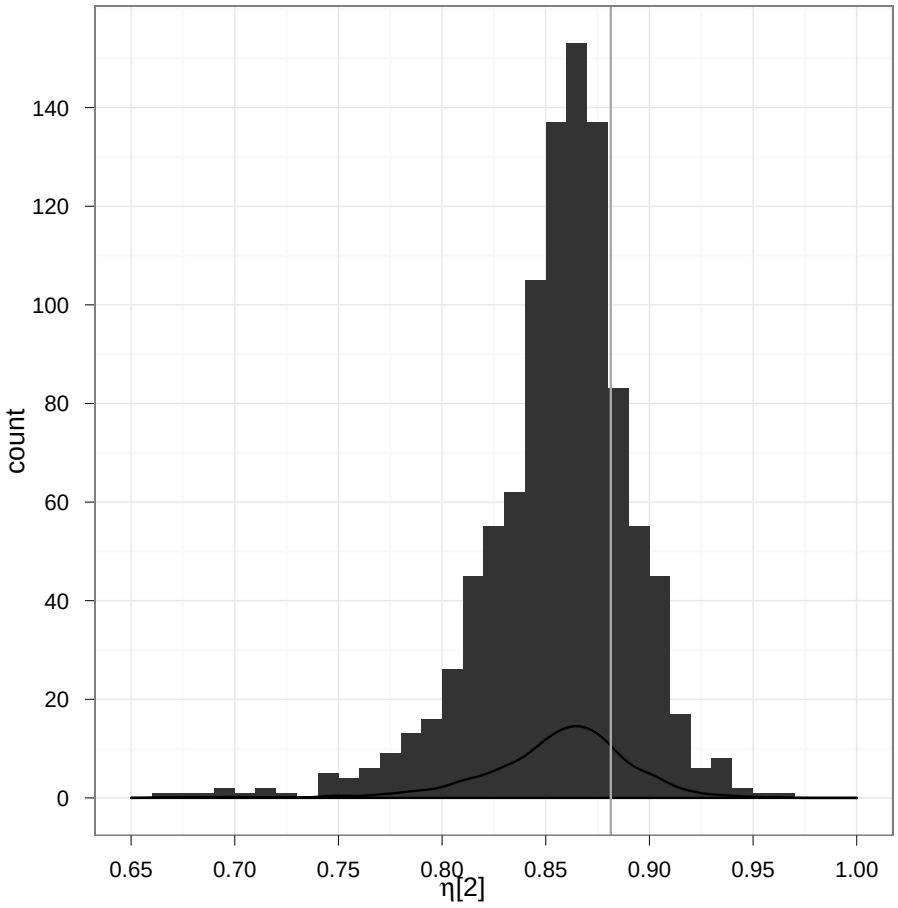

Figure 5.2: Histogram of the $\eta[2]$ component for 1000 MLEs for an Ising model on a 32×32 lattice with $\eta = \big(0, \log(1+\sqrt{2})\big)$. The vertical line is at $\log(1+\sqrt{2}) = 0.881$.

5.3 Example: Sampson's monastery data

As discussed in Section 1.4.3, MCMC-MLE may struggle to converge when the starting value is far from the true MLE. This was illustrated on Sampson's monastery data set with the Erdős-Rényi-Gilbert model with network statistic $g(y)$ equal to the total number of edges present. The observed network, depicted in Figure 1.2, is directed with 18 actors and 88 ties present out of $18 \cdot 17 = 306$ possible ties. The true MLE is equal to

$$\hat{\eta}_{\text{MLE}} = \text{logit}\left(\frac{g(y_{\text{obs}})}{n(n-1)}\right) = \text{logit}\left(\frac{88}{306}\right) = -0.9072.$$

With $\eta^0 = 1$, the MCMC-MLE approximated log likelihood ratio (using 100,000 samples) cannot be maximized, as shown in Figure 1.4. We start from this same initial value, using the `simulate.formula` function from the `ergm` package in R to generate 10,000 MCMC samples, $g(Y_1), \ldots, g(Y_{10000})$, with each sample spaced 1000 apart from one another.

It took our algorithm 21 gradient evaluations over 5 different search directions to get $\|\nabla \ell(\eta_k)\| < 0.1$ and arrive at an estimate for the MLE of -0.9069. The intermediate steps are presented in Table 5.5. Similar results were attained using much smaller MCMC sample sizes (as small as 200).

Table 5.5: Long range algorithm applied to Sampson's monastery data. The starting point of $\eta^0 = 1$ can be thought of as "far" from the true MLE of -0.9072. The inner loop count is the number of iterations spent searching for an α_k that meets the curvature condition.

k	η_k	$\|\nabla\ell(\eta_k)\|$	α_k	inner loop count
0	1.0000	135.64	0.012	8
1	−0.6649	15.99	0.014	2
2	−0.8817	1.57	0.014	3
3	−0.9043	0.22	0.012	8
4	−0.9069	0.03		

5.4 Example: Faux Magnolia High School

Goodreau (2007); Goodreau et al. (2008) analyzed an adolescent friendship network data set of 1,461 students in grade 7–12 derived from the National Longitudinal Study of Adolescent Health (Resnick et al., 1997). The data set, which Goodreau et al. refer to as Faux Magnolia High School, is a model-based simulation from the original data set, where the simulation is necessitated to maintain confidentiality.

Goodreau et al. (2008) run analyses and settle on a model with the following terms: Grade, Race, Sex, edges, and geometrically weighted edgewise shared partners (GWESP). The model selection issue is one we do not address here—see Hunter et al. (2008a) for a discussion on methods goodness-of-fit. Grade, Race, and Sex are covariate factors specific to the actors and in this model reflect whether or not two actors in a dyad are of the same type. The issue of incorporating these factors into an ERGM was discussed in Section 1.2.2. The GWESP network statistic is not one we have discussed previously but has been alluded to in Section 1.2 as one of the recently developed statistics intended to reduce problematic models (the loose usage of "degeneracy") (Snijders et al., 2006; Hunter and Handcock, 2006; Robins et al.,

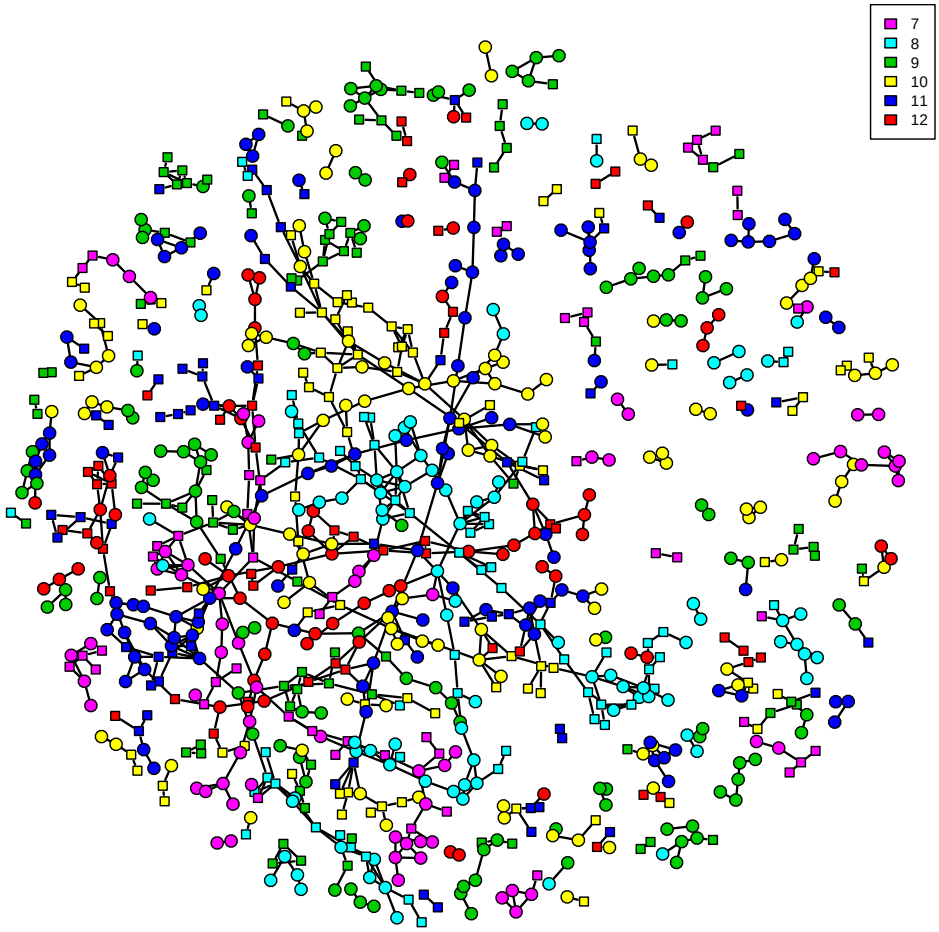

Figure 5.3: Faux Magnolia High friendship network without isolates, colored by grade. Boys are depicted by square nodes, girls by circles. The data is made available in `ergm` (Handcock et al., 2010).

2007b). This particular statistic can be thought of as a generalization of triangles.

We ran our algorithm through 22 search directions, starting from an initial value for η of $(0, 0, 0, 0, 0)$. As in the example with Sampson's monastery data in Section 5.3, we rely on the `simulate.formula` command provided in the `ergm` package to generate MCMC samples from the distribution indexed by η. The results of our estimates are reported in Table 5.6.

We compare our estimates to those obtained by running the default MCMC-MLE

Table 5.6: Comparison of MLEs for η for MCMC-MLE and our algorithm in Faux Magnolia High example. MCMC-MLE is the default algorithm in `ergm`, and is run here through 5 iterations starting at the MPLE ($\hat{\eta}_{\text{MLE}}$ below). Our algorithm is run for 22 steps ($\hat{\eta}_{\text{Steep}}$ below), starting from $(0, 0, 0, 0, 0)$.

	edges	Grade	Race	Sex	GWESP
$\hat{\eta}_{\text{MLE}}$	-9.790	2.755	0.906	0.780	1.813
$\hat{\eta}_{\text{Steep}}$	-9.893	2.81	0.943	0.794	1.814

algorithm in `ergm`, which starts from an MPLE of

$$\hat{\eta}_{MPLE} = (-9.8665, 2.8626, 0.9944, 0.8261, 1.6950).$$

It is noteworthy that MCMC-MLE has exactly the difficulties described in Section 1.4.3 when started at $(0, 0, 0, 0, 0)$ as well: after 15 iterations, all the coefficient estimates would still effectively be zeros (< 0.05 in absolute value). Thus its successful convergence here is due entirely to its use of a remarkably good MPLE starting point, which cannot always be relied upon.

Our algorithm is time consuming. Whereas running MCMC-MLE through 15 iterations with MCMC sample sizes of 100,000 at intervals of 100 steps as done by Goodreau et al. (2008) takes a few minutes, we used nearly 150 iterations over 22 search directions, each iteration using MCMC sample sizes of 10,000 spaced 5,000 iterations apart, taking nearly two hours. We actually stopped the algorithm at 22 search directions; we did not attain our desired cutoff of $\|\nabla\ell(\eta)\| < 0.1$ and were only able to attain values of $\|\nabla\ell(\eta)\|$ as small as 10.9.

A few remarks:

- The goal here was to show that our algorithm can find the MLE for a more realistic model and a large data set, starting from any initial value. Although it is not computationally expedient, we obtain a reasonable estimate after 22

updates of η_k.

- Our recommended use of this algorithm is in combination with other algorithms, as discussed in Section 3.6. Thus, although it is satisfying to see a reasonable estimate of the MLE, what is more useful is the value of η_k at say $2d$ iterations, where d is the dimension of η. Here we found that at 10 iterations using steepest ascent directions,

$$\eta_{10} = (-9.992, 2.915, 0.982, 0.79, 1.73)$$

and when using conjugate gradient directions,

$$\eta_{10} = (-9.963, 2.965, 0.927, 0.742, 1.784).$$

From these values, MCMC-MLE is able to attain the MLE in one update. In fact, our algorithm itself will not improve much from this point without increasing sample sizes. The difference between η_{10} and $\hat{\eta}_{\text{MLE}}$ is already quite small, 0.277 in Euclidean distance.

Although safeguarding other algorithms is an option, a better condition for switching is nonetheless desirable. An idea based on importance sampling weights was mentioned in Section 3.6 and warrants further research.

- MCMC sampler. We have developed our algorithm assuming that it would have access to an effective MCMC sampler. In this example, some tuning of the MCMC sampler is necessary. This is unsurprising given the very large number of points in the sample space and the challenge of generating less dependent discrete samples. Hunter et al. (2008b); Morris et al. (2008) discuss the ERGM MCMC sampler issues further.

- Variable MCMC sample sizes. A more pragmatic way to implement our algorithm would have been to increase the MCMC sample sizes as the algorithm progressed. That is, when we are very far away, crude estimates of the gradient and step size should be sufficient to get η "closer" and satisfy the curvature condition (3.3). As noted in Section 3.5, when $\|\nabla\ell(\eta)\|$ gets smaller, the curvature condition is more difficult to satisfy and sharper estimates—requiring larger MCMC sizes—are necessary.

5.5 Example: 9-node network

We now consider the cases for ERGMs where the MLE may not exist. Like Handcock (2003); Rinaldo et al. (2009), we focus on a small 9-node networks with only two network statistics. Because we wish to compare our algorithm to the truth—whether or not the MLE exists, and what its value is if it does—we work with networks for which it is feasible to evaluate the likelihood function. For an undirected 9-node network, this requires a summation over $2^{\binom{9}{2}}$, or about 69 billion different graphs. Determining the existence of the MLE requires knowledge of the convex support of the sufficient statistics. See Appendix A for our approach for enumerating all possible graphs in a 9-node network with counts of three network statistics: edges, two-stars, and triangles.

For comparison purposes with Rinaldo et al. (2009), we use the same model with the number of edges and triangles as the natural statistics. A two-dimensional statistic also naturally lends itself to more easily interpretable figures. The relevant convex support for this model is depicted in Figure 1.5. Although there are nearly 69 billion points in the sample space $\mathcal{Y}$, there are only 444 possible combinations of edges and triangles. Let the space of these sufficient statistics be denoted $\mathcal{T} = g(\mathcal{Y})$. The summation in the normalizing constant (1.2) over all graphs in $\mathcal{Y}$ can then be

simplified to a weighted sum over the 444 points in $\mathcal{T}$ as follows:

$$\kappa(\eta) = \sum_{y \in \mathcal{Y}} e^{\langle g(y), \eta \rangle} = \sum_{t \in \mathcal{T}} e^{\langle t, \eta \rangle} \nu(t)$$

where $\nu(t)$ is the frequency of the network statistic t across all $y \in \mathcal{Y}$. This does require knowing $\nu(t)$ for all $t \in \mathcal{T}$, which we calculate once and save to file in our enumeration of all graphs.

In fact, we will find it easier to work directly with the distribution of the sufficient statistic $T = g(Y)$:

$$\begin{aligned} P_\eta(T = t) &= \sum_{y \in \mathcal{Y}: g(y) = t} P_\eta(Y = y) \\ &= \sum_{y \in \mathcal{Y}: g(y) = t} \frac{1}{\kappa(\eta)} e^{\langle g(y), \eta \rangle} \\ &= \sum_{y \in \mathcal{Y}: g(y) = t} \frac{1}{\kappa(\eta)} e^{\langle t, \eta \rangle} \\ &= \frac{1}{\kappa(\eta)} e^{\langle t, \eta \rangle} \nu(t). \end{aligned}$$

We can then calculate exact values for the gradient of the log likelihood $\nabla \ell(\eta)$ by (2.1),

$$\begin{aligned} \nabla \ell(\eta) &= g(y_{\text{obs}}) - \mathrm{E}_\eta \, g(Y) \\ &= g(y_{\text{obs}}) - \mathrm{E}_\eta \, T \\ &= g(y_{\text{obs}}) - \sum_{t \in \mathcal{T}} t P_\eta(T = t) \\ &= g(y_{\text{obs}}) - \frac{1}{\kappa(\eta)} \sum_{t \in \mathcal{T}} t e^{\langle t, \eta \rangle} \nu(t). \end{aligned}$$

To address concerns for overflow where e^x can become too large, we can subtract

$a = \max_t \langle t, \eta \rangle$ in the exponent, so that

$$\nabla \ell(\eta) = g(y_{\text{obs}}) - \frac{1}{\sum_{t \in \mathcal{T}} e^{\langle t, \eta \rangle - a} \nu(t)} \sum_{t \in \mathcal{T}} t e^{\langle t, \eta \rangle - a} \nu(t).$$

The second derivative of the log likelihood $\nabla^2 \ell(\eta)$ can also be expressed in term of the sufficient statistic. By (2.2),

$$\begin{aligned}
\nabla^2 \ell(\eta) &= -I(\eta) = -\operatorname{Var}_\eta g(Y) \\
&= -\operatorname{Var}_\eta T \\
&= -\operatorname{E}(T - \operatorname{E}_\eta T)(T - \operatorname{E}_\eta T)^T \\
&= -\sum_{t \in \mathcal{T}} (t - \operatorname{E}_\eta T)(t - \operatorname{E}_\eta T)^T P(T = t) \\
&= -\frac{1}{\kappa(\eta)} \sum_{t \in \mathcal{T}} (t - \operatorname{E}_\eta T)(t - \operatorname{E}_\eta T)^T e^{\langle t, \eta \rangle} \nu(t).
\end{aligned}$$

The same adjustment for overflow used for $\nabla \ell(\eta)$ can be applied here.

Being able to evaluate $\nabla \ell(\eta)$ and $\nabla^2 \ell(\eta)$ exactly is useful when searching directly for MLEs by likelihood evaluation; optimization routines such as `trust` (Geyer, 2009d) in R will use these as inputs.

We are now equipped with knowledge of the convex support and a log likelihood we can evaluate and numerically maximize. The linear programming functions in `rcdd` described in Chapter 4 can be used to determine whether an observed network falls on the boundary or interior of the convex support. These will allow us to establish the "truth" against which we can compare our algorithm.

The key difference between our work here and the approaches of (Handcock, 2003; Rinaldo et al., 2009) is that our algorithm does not assume knowledge of the convex support C and thus must determine "on the fly" whether or not the MLE exists. We do take liberty with two short cuts in the application of our algorithm to make our

analysis cleaner:

1. MC sampler. We use a perfect Monte Carlo sampler to generate our draws $g(Y_1), \ldots, g(Y_m)$ from the distribution with parameter η, rather than a MCMC sampler. Although we have replicated the results of the algorithm using the MCMC sampler provided by the `simulate.formula` function in the R package `ergm` (Handcock et al., 2010), we wish to identify here the issues that would still be present even with a perfect MC sampler. This can easily be built for this small network since the likelihood function can be evaluated exactly.

2. Step length search. The search for the step length α_k that satisfied the curvature condition (3.3) can be time consuming, as discussed in Section 3.5 and evident in our previous examples. Here we use our ability to evaluate the gradient function $\nabla\ell(\cdot)$ exactly; we solve for the value of α_k that sets $\nabla\ell(\eta_k + \alpha_k p_k)^T p_k = 0$ using a simple root-finding algorithm `uniroot`, with a slight adjustment to satisfy (3.3). So, for a given direction p_k, we find the step size that maximizes the log likelihood in that direction. We do not use this shortcut in any other portions of our method that require $\nabla\ell(\cdot)$, such as updating search directions p_k.

We believe that neither of these undermine the validity of our overall methodology, though they undoubtedly simplify and speed up its application.

5.5.1 Two-dimensional sufficient statistic

The MLE for an exponential family model will exist if and only if the observed statistic lies in the relative interior of the polyhedral convex support C, which is the sail-shaped polytope in Figure 1.5. Here, C has six sides (and six vertices).

5.5.2 Case: MLE exists

Suppose the observed network data has 29 edges and 47 triangles. Then $g(y_{\text{obs}}) = (29, 47)$ lies in the interior of the convex support as depicted in Figure 5.4 (top) and the MLE exists. Our algorithm, arbitrarily starting at the natural parameter value of $\eta = (0, 0)$, generates 10,000 Monte Carlo samples from the model with this initial parameter value. As the algorithm climbs the log likelihood surface, the cloud of sample points will move towards the observed statistic and eventually engulf it. When the value for η is equal to the MLE, the mean of the MC sample points will equal $(29, 47)$ (see Figure 5.4 (bottom)). For this problem, the MLE for η is found to be

$$\hat{\eta}_{\text{MLE}} = (-0.389, 0.418),$$

which matches the results we get from applying `trust` to numerically maximize the log likelihood function.

5.5.3 Case: MLE does not exist; observed statistic in one-dimensional face

Suppose the observed network has network has 31 edges and 50 triangles. Then $g(y_{\text{obs}}) = (31, 50)$ lies on the boundary of the convex support as depicted in Figure 5.5. To be precise, the observed network statistic $(31, 50)$ lies in the relative interior of a line segment on the boundary of the convex support with end points $(30, 44)$ and $(32, 56)$. By Theorem 2.5, the MLE does not exist in the conventional sense.

Our algorithm begins as before, generating MC sample point clouds from the distribution indexed by $\eta = (0, 0)$. Because the convex support is not assumed to be known, the algorithm cannot determine at the outset that the MLE does not

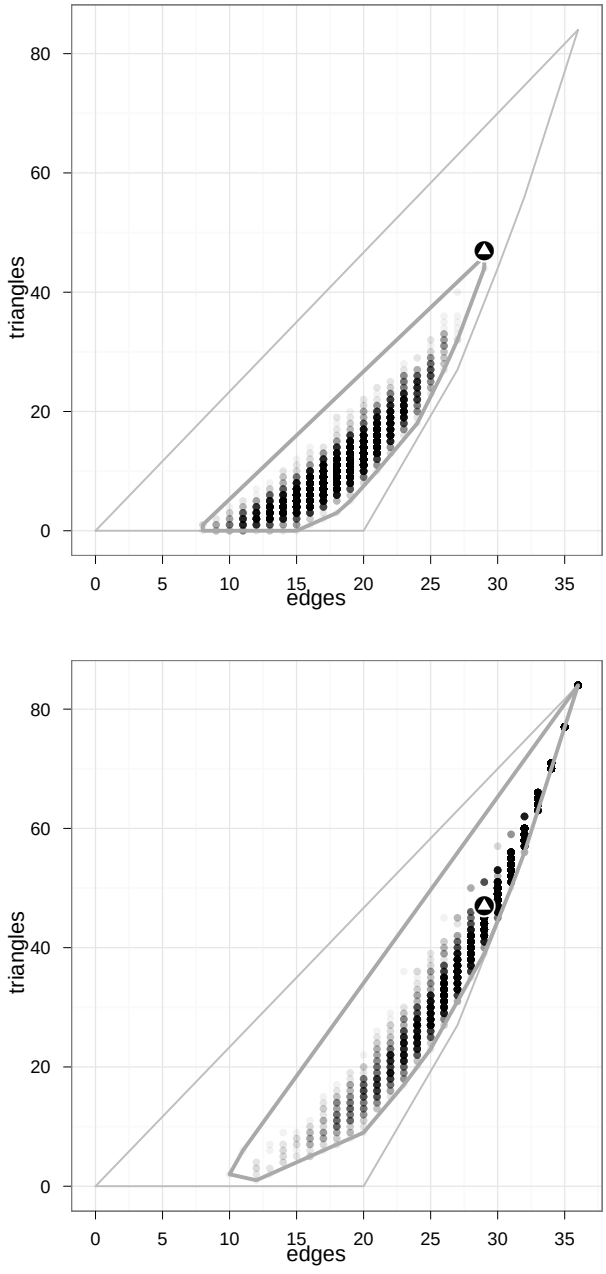

Figure 5.4: Network statistics for 10,000 Monte Carlo samples generated from models with $\eta = (0, 0)$, top, and $\eta = \hat{\eta}_{\text{MLE}} = (-0.389, 0.418)$, bottom. The observed network statistic is $(29, 47)$, depicted by the larger point with triangle. When η is the MLE, the observed statistic is the mean of the MC sample points generated.

exist; rather, it can only generate more sample point clouds as it climbs the log likelihood surface, with successive point clouds moving towards the observed statistic. Eventually, the observed statistic will lie exactly on the boundary of a point cloud. When this occurs, the algorithm must

1. determine empirically the face $\tilde{F}$ in which the observed statistic lies in the relative interior of,

2. decide if $\tilde{F}$ is in fact the boundary of the convex support C.

We use $\tilde{F}$ and $\tilde{\delta}$ to distinguish that these are empirical faces and GDORs which in fact may not be actual faces of C and GDORs.

The first task can be done using the linear programming methods described in Chapter 4. In particular, by use of the `linearity` function, we can identify the extreme points of the empirical face $\tilde{F}$.

The second task turns out to be more difficult. We first calculate an empirical GDOR by the methods in Section 4.6, which here give us

$$\tilde{\delta} = (6, -1).$$

If this $\tilde{\delta}$ is a true GDOR δ, $\tilde{F}$ is a true face F of C, and $g(y_{\text{obs}}) \in \operatorname{rbd} C$, then

$$\langle w - g(y_{\text{obs}}), \tilde{\delta} = 0 \rangle \tag{5.1}$$

for all $w \in \tilde{F}$. Thus (5.1) gives an easy way to check if any point is in the empirical face $\tilde{F}$. We then assume that if a substantial portion of the sample points generated, say more than 60%, land on this empirically determined face $\tilde{F}$, then it is in fact a boundary of the convex support C.

In this example, the algorithm determines empirically that three points from the MC sample—$(30, 44)$, $(31, 50)$, and $(32, 56)$—lie on a one-dimensional face $\tilde{F}$, as de-

picted in Figure 5.5. If less than 60% of the sample points land on this empirical face, the algorithm continues to sample, trying to get a sample point cloud even closer to the observed statistic. We choose such a high proportion for a cutoff to eliminate—or at least, greatly reduce—the possibility of misidentifying a boundary. (We deal later with a case where 100% of the MC sample points form an empirical face $\tilde{F}$ which is *not* on the boundary of C.) The empirical GDOR $\tilde{\delta}$ that we calculate can also be used as an effective search direction when it is not yet determined that $\tilde{F}$ is in the boundary of C. If $\tilde{F}$ is in the boundary of C, then by Theorem 2.6, this threshold will be met. A sample point cloud meeting this requirement is depicted in Figure 5.5.

The 60% cutoff number may seem arbitrary, and to a degree it is. A high cutoff is necessary to avoid misidentifying a boundary, but we also do not want this cutoff to be too high. A distribution that assigns very high probability to the points in $\tilde{F}$ will be difficult to sample from effectively with MCMC. The state proposals generated in a Metropolis algorithm will mostly be rejected since they do not land in $\tilde{F}$ and thus the chain will not mix well in practice. Although this is not an issue for our perfect MC sampler, we recognize that this will be a concern when using an actual MCMC sampler.

By identifying the empirical face $\tilde{F}$ in whose relative interior the observed statistic lies, the algorithm has not only concluded that the MLE does not exist in the conventional sense, it has also defined $\tilde{F}$ as the convex support for the new limiting conditional model for which the MLE must exist by Corollary 2.5. The algorithm then maximizes this new exponential family using the same iterative approach as before. The gradient of the LCM log likelihood is approximated using (4.1),

$$\nabla \ell(\eta)^{LCM} \approx g(y_{\text{obs}}) - \frac{1}{p} \sum_{i=1}^{p} g(Y_{(p)})$$

where $g(Y_{(1)}), \ldots, g(Y_{(p)})$ is the subsample of the MC sample points restricted to the

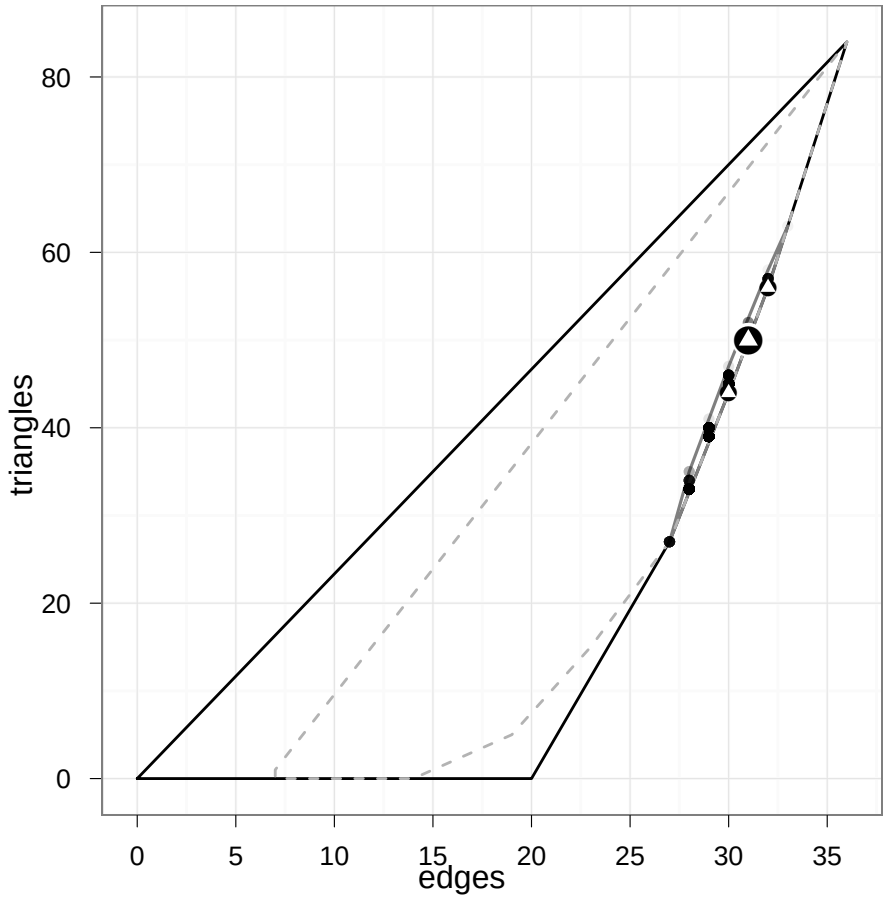

Figure 5.5: Network statistics for 10,000 Monte Carlo samples when the MLE does not exist. The observed statistic, $(31, 50)$, is the largest point on the boundary marked with a triangle. A face has been identified, defined by $(30, 44)$, $(31, 50)$, and $(32, 56)$ and marked by triangles. Here, 63% of the MC sample points fall on these three points. The lighter colored polytope is the convex hull of all previously sampled points.

empirical face, $(30, 44)$, $(31, 50)$, and $(32, 56)$ in this case. The maximizer $\hat{\eta}_{\text{LCM}}$ of this log likelihood is found to be

$$\hat{\eta}_{\text{LCM}} = (122.7, -20.4).$$

The LCM, however, is not identifiable, since the support is now only one-dimensional compared to two in the original model. That is, there must exist a constancy space of this new model, Γ_{lim}, such that

$$\ell(\eta+\gamma)^{LCM} = \ell(\eta)^{LCM} \tag{5.2}$$

for any $\gamma \in \Gamma_{\text{lim}}$. By Theorem 2.3, the GDOR $\tilde{\delta}$ that the algorithm found is one such γ.

To construct one-sided 95% confidence intervals for how close η is to infinity, we need to find the value of s for which the distribution indexed by $\hat{\eta}_{\text{LCM}}+\gamma+s\tilde{\delta}$ assigns a probability of 0.05 to the event that a sample falls on the face of interest. That is, find the unique $\hat{s}$ such that

$$P_{\hat{\eta}_{\text{LCM}}+\gamma+\hat{s}\tilde{\delta}}(g(Y) \in \tilde{F}) = 0.05.$$

We can find this by generating MC samples from the distribution with parameter $\hat{\eta}_{\text{LCM}}+\gamma+s\tilde{\delta}$. Some numerical root solving finds $\hat{s} = -18.9$. Thus the 95% CI for $\hat{\eta}_{\text{LCM}}+s\tilde{\delta}$ can be expressed as

$$\left\{ \begin{pmatrix} 122.7 \\ -20.4 \end{pmatrix} + s \begin{pmatrix} 6 \\ -1 \end{pmatrix} : \quad s \geq \hat{s} = -18.9 \right\}$$

or

$$\left\{ \begin{pmatrix} 9.15 \\ -1.50 \end{pmatrix} + s \begin{pmatrix} 6 \\ -1 \end{pmatrix} : \quad s \geq 0 \right\}.$$

It would be misleading to construct confidence regions for η here because the above CI only captures the uncertainty of the parameter in the direction of $\tilde{\delta}$; there

is still the usual uncertainty regarding the estimate of $\hat{\eta}_{\text{LCM}}$ in the LCM, that is, in an orthogonal direction to $\tilde{\delta}$, which we do not calculate here. This could be done by finding the inverse Fisher information matrix in the LCM to use as a standard error.

To summarize, our algorithm realized correctly that the MLE for this model does not exist in the conventional sense. The algorithm identified an empirical face $\tilde{F}$ and empirical GDOR $\tilde{\delta}$ by applying methods in computational geometry to the convex hull of all generated MCMC sample statistics. By getting a sufficient proportion of a new MCMC sample to land in the empirical face $\tilde{F}$, the algorithm concluded that $\tilde{F}$ is in fact in the relative boundary of the convex support C. At this point, the algorithm has decided that the MLE does not exist in the conventional sense. The algorithm then found $\hat{\eta}_{\text{LCM}}$, the maximizer of the exponential family which has $\tilde{F}$ as its convex support. Finally, the algorithm used the values of $\hat{\eta}_{\text{LCM}}$ and $\tilde{\delta}$ to find a distribution along $\tilde{\delta}$ that put 5% probability on $\tilde{F}$, marking the lower end point of a confidence interval for how close η is to infinity.

5.5.4 Case: MLE does not exist; observed statistic in zero-dimensional face

If the observed network has statistic $g(y_{\text{obs}}) = (27, 27)$, then $g(y_{\text{obs}})$ is an extreme point of the convex support C (it is one of the six defining vertices of C, depicted by a square point in Figure 1.5). In this case, the point is itself the face with zero-dimension and the MLE does not exist.

The algorithm proceeds as before, and concludes that the point $(27, 27)$ is the empirical face $\tilde{F}$ and lies in the boundary of C, using the same 60% criteria as before. In this case, the limiting conditioning model is completely unidentifiable, and thus

any value for η will be a maximizer. Our particular implementation found that

$$\hat{\eta}_{\text{LCM}} = (28.6, -6.9).$$

The normal cone to the observed statistic $N_C(g(y_{\text{obs}}))$ in this case is a two-dimensional cone (and not a subspace), bounded by two directions,

$$\{(3.857, -1), (5.667, -1)\}.$$

The linear programming routine in our algorithm found an empirical GDOR of $\tilde{\delta} = (4.4, -1)$, clearly in the relative interior of this normal cone, which is itself not calculated by our algorithm but done here for comparison purposes using the methods described in Section 4.5. Proceeding as before, the 95% CI for $\hat{\eta}_{\text{LCM}} + s\tilde{\delta}$ can be expressed as

$$\left\{ \begin{pmatrix} 28.6 \\ -6.9 \end{pmatrix} + s \begin{pmatrix} 4.4 \\ -1 \end{pmatrix} : \quad s \geq \hat{s} = -3.2 \right\}$$

or

$$\left\{ \begin{pmatrix} 14.5 \\ -3.7 \end{pmatrix} + s \begin{pmatrix} 4.4 \\ -1 \end{pmatrix} : \quad s \geq 0 \right\}.$$

5.5.5 Case: MLE exists but observed statistic is very near boundary

If the observed data has network statistic $(21, 4)$, it is in the relative interior of the convex support C (see Figure 5.6) and so the MLE exists. It lies very close to the boundary, however, and from a mean value parameter perspective, this observed statistic corresponds to a nearly degenerate distribution. The Shannon entropy func-

tion would show this point to have extremely small entropy, corresponding to a model that puts most of its probability on very few points (Rinaldo et al., 2009).

The log likelihood for this model is extremely flat, causing some disagreement among numerical optimization routines for MLE estimates, though the log likelihood values themselves are nearly identical. Using a trust optimization routine, we calculate that

$$\hat{\eta}_{\text{MLE}} = (28.86, -7.76).$$

The most problematic aspect of this model for us here, however, is that our algorithm may conclude—incorrectly—that the MLE does *not* exist, and proceeds to calculate the MLE of an empirical $\widetilde{\text{LCM}}$ as in the previous examples (we continue the use of the tilde '$\sim$' to denote empirically derived results). In these examples, we are using an MC sample size of 10,000. We are using a perfect Monte Carlo sampler from the exact distribution rather than an MCMC sampler, so this is not an MCMC issue.

How does this happen? Does this make our algorithm not useful in practice?

Our algorithm begins in the same manner as described previously. As the algorithm proceeds uphill on the log likelihood function, it may iterate η_k to a value (for example, $\eta_k = (35.34, -9.56)$) where all 10,000 MC sample points generated from the distribution with this parameter value fall on the six points on the line segment between $(20, 0)$ and $(25, 20)$, as depicted in Figure 5.6. The observed value $(21, 4)$ is one of these six points, and the algorithm concludes that it has found an empirical face $\tilde{F}$ in the boundary of C.

In order for the algorithm to have correctly concluded that $(21, 4)$ was in the relative interior of C, the extreme point $(27, 27)$ would need to have occurred in the MC sample (as it occasionally did, in which case the algorithm found the MLE as usual). However, though this point is not far from the end point $(25, 20)$ of our line

segment, the model indexed by the current parameter value assigns a probability of 0.000032 to this point. In fact, even the MLE model (indexed by $\hat{\eta}_{\text{MLE}}$ found above) only assigns a probability of 0.00045 to this point. So, the extreme point that is critical for correctly determining the convex support is assigned very low probability and may not appear in a MC sample of 10,000 points.

Thus in most of our trials, our algorithm treats the line segment as a boundary of the convex support in which the observed statistic lies in the relative interior, and hence the support of the empirical $\widetilde{\text{LCM}}$. It proceeds as in the first example, finding the MLE of this empirical $\widetilde{\text{LCM}}$,

$$\hat{\eta}_{\widetilde{\text{LCM}}} = (35.38, -9.39),$$

after which it then calculates one-sided confidence intervals for η.

On first glance this is very troubling: our algorithm arrives at the wrong conclusion about the existence of the MLE. However, we believe the algorithm is actually performing better than it might initially appear. A step back should be taken and the goal of the analysis revisited.

What is the purpose in finding the MLE? In particular, what does one do with these values once they are found?

If the desired analysis is to interpret these numbers by their sign and magnitude, then we indeed have a problem—our numbers are wrong. However, if the MLE is viewed as an index to a specific distribution that assigns the highest probability to the observed data, then we claim that the distribution we have found—the empirical $\widetilde{\text{LCM}}$ indexed by parameter value $\hat{\eta}_{\widetilde{\text{LCM}}}$—is in fact remarkably similar to the original model indexed by the true MLE. A reasonable metric for this comparison is a sum of

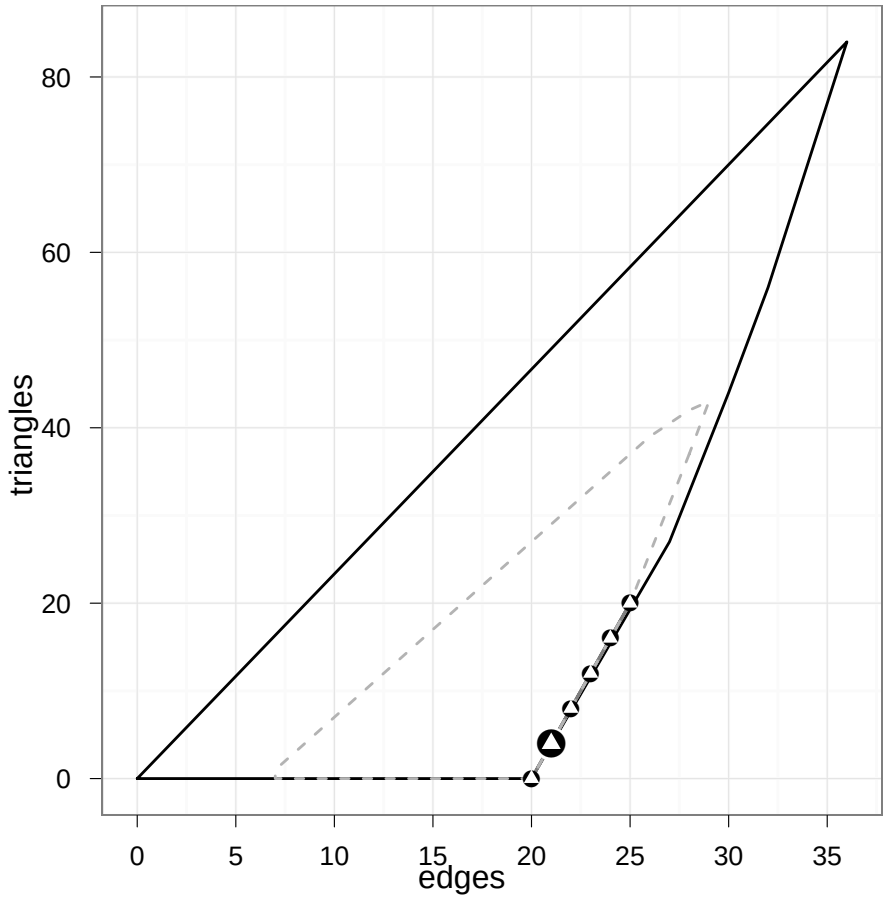

Figure 5.6: Network statistics for 10,000 Monte Carlo samples when the observed statistic is very close to a boundary. The observed statistic is $(21, 4)$, marked by the largest point. Here, all 10,000 samples fall on six points on a line segment that appears to be in the boundary of the convex support. However, all points except $(20, 0)$ are in the interior. The absence of the extreme point $(27, 27)$ in this sample—and all previous samples—leads to this misidentification. The light grey polytope is the convex hull of all previously sampled points.

the absolute difference in probabilities assigned to each point in the sample space,

$$\sum_{y \in \mathcal{Y}} |P_{\hat{\eta}_{\text{MLE}}}(g(Y) = g(y)) - P_{\widetilde{\text{LCM}}, \hat{\eta}_{\widetilde{\text{LCM}}}}(g(Y) = g(y))|.$$

The probabilities assigned by each of these distributions to the six points in the empirical face is summarized in Table 5.7.

Table 5.7: Probabilities assigned by the empirical $\widetilde{\text{LCM}}$ and original MLE model to six points in an empirical face of a 9-node network. The observed statistic is $(21, 4)$. For the MLE model, the probabilities do not sum to 1 since it assigns positive probability to points other than these six.

	Edges	Triangles	$\widetilde{\text{LCM}}$	MLE
1	20	0	0.3414	0.3425
2	22	8	0.2019	0.2009
3	21	4	0.3914	0.3911
4	23	12	0.0566	0.0561
5	24	16	0.0081	0.0080
6	25	20	0.0005	0.0005
			1.0000	0.9990

Here, the sum of the absolute value of differences on these empirical points is only 0.0031. Including the additional 0.001 of probability to points in $C \setminus \tilde{F}$ that are outside the empirical $\widetilde{\text{LCM}}$ support, this total still only comes to 0.0041, a difference that would seem insignificant for most applications.

Of course, there are practical issues: even if a researcher is interested in the MLE distribution, she may want the software to simply return MLE values to keep around for later analysis. Here, we are suggesting that the analysis return empirical $\widetilde{\text{LCM}}$ MLE values and an empirical $\widetilde{\text{LCM}}$ model (perhaps in the form of a GDOR or convex support). The researcher may understandably be confused, especially if she knew in advance that the MLE was guaranteed to exist. To make matters yet more confusing, it is possible that the algorithm *does* produce the extreme point $(27, 27)$ in a sample and finds the MLE in the original model. Thus by randomness the researcher might get either conclusion about MLE existence. However, we emphasize again that any probability calculations—and hence inference—would be nearly identical. This

example highlights how the use of an approximate method in Monte Carlo to produce exact results will occasionally yield wrong conclusions.

Finally, it may be of interest to note that $\hat{\eta}_{\widetilde{\text{LCM}}}$ and $\hat{\eta}_{\text{MLE}}$ index nearly identical distributions in the empirical $\widetilde{\text{LCM}}$, with the difference due almost entirely to the lack of identifiability of the empirical $\widetilde{\text{LCM}}$. A GDOR to the empirical face is $\tilde{\delta} = (4, -1)$, which is also a direction of constancy for the empirical $\widetilde{\text{LCM}}$. Then by (5.2),

$$\begin{aligned}\ell(\hat{\eta}_{\widetilde{\text{LCM}}})^{\widetilde{\text{LCM}}} &= \ell(\hat{\eta}_{\widetilde{\text{LCM}}} + \gamma)^{\widetilde{\text{LCM}}} \\ &= \ell(\hat{\eta}_{\widetilde{\text{LCM}}} + k\,(4, -1))^{\widetilde{\text{LCM}}}.\end{aligned}$$

If we had perfect knowledge and chose $k = -1.63$,

$$\hat{\eta}_{\widetilde{\text{LCM}}} - 1.63\,(4, -1) = (28.86, -7.76)$$

matching the MLE, $\hat{\eta}_{\text{MLE}} = (28.86, -7.76)$, to the significant figures considered.

Chapter 6

Discussion

The goal of this dissertation is to introduce a new practical approach for finding the MLE of a discrete exponential family when the MLE exists in the conventional sense, and calculating one-sided confidence intervals for the parameters when it does not.

The algorithm avoids the trial and error experimentation of tuning parameters and starting points commonly associated with optimization routines not invented by optimization specialists. Our algorithm is modeled after standard algorithms discussed in optimization textbooks (Fletcher, 1987; Nocedal and Wright, 1999; Sun and Yuan, 2006), all of which are safeguarded to ensure rapid automatic convergence.

In the setting where the MLE exists, convergence is guaranteed when the gradient can be calculated exactly. Even when the gradient cannot be calculated exactly and is only estimable via MCMC, the algorithm is still useful in practice, as demonstrated by the Ising model and Faux Magnolia High examples. We have also described a way to construct and use confidence intervals to make convergence highly probable.

For the setting where the MLE does not exist and the convex support is not known (as in ERGMs), our algorithm uses MCMC sampling to both direct the search towards the face in which the observed statistic lies while exploring the geometry of the convex support. We use computationally efficient methods that avoid calculating H-representations of convex hulls and only do comparisons using rational arithmetic.

We know of no other general approach for dealing with the setting where the convex support is unknown in advance.

Our algorithm can be computationally demanding. When the log likelihood gradient approaches zero, the curvature condition for step size becomes more difficult to satisfy and the method may require several iterations of MCMC sampling and perhaps an increase in MCMC sample size. Eventual increase in MCMC sample size is unavoidable, because the achievable accuracy is inversely proportional to the square root of the MCMC sample size, as in all Monte Carlo. Thus when the MLE exists, we believe the best use of this algorithm is in combination with other faster methods like MCMC-MLE (Geyer and Thompson, 1992) or Newton-Raphson safeguarded by our line search algorithm. Our algorithm should be used from "long range", when one has no good intuition for an initial value and is concerned about picking one that is far from the MLE. The switch to another algorithm such as MCMC-MLE need not require manual intervention, though the criteria for switching needs to be further studied. When used in combination in this manner, we do not think the confidence intervals are necessary as the curvature condition is quite easily satisfied when the current iteration is far from the MLE.

One way to improve performance is to use conjugate gradient search directions rather than steepest ascent. In our examples, this reduced the number of iterations by over 20%. However, in other problems we tried with different dimensionality, this performance varied significantly and it appears that no guarantee can be made about quantity of improvement in performance, though in all cases we examined, it never did worse. This is no surprise, because the necessity of "preconditioning" for good performance of the conjugate gradient algorithm is well known (but we know of no literature about "preconditioners" for maximum likelihood in exponential families).

6.1 Areas for further research

6.1.1 Convergence

There are several outstanding issues. We have not showed convergence of the algorithm when the gradient is approximated via MCMC. This is a more difficult theoretical problem and is the motivation for stochastic approximation research. Further work is necessary to determine if one can adapt our restrictive curvature condition (3.3) to the approach of Andrieu et al. (2005) or Liang (2010) in MCMC stochastic approximation.

6.1.2 Step size search

The process of finding a step size α to satisfy the curvature condition (3.3) can be improved. The spline approximation of $\nabla\ell(\eta + \alpha p)^T p$ as a function of α should ideally incorporate the strict concavity of the function. As noted above as well as in Section 3.5, when the gradient must be approximated via MCMC, (3.3) becomes increasingly difficult to satisfy as the gradient approaches zero. Our approach currently generates two independent MCMC samples, one for the distribution at η, the other at $\eta+\alpha p$. By using simulating tempering in the manner of Geyer and Thompson (1995), we might effectively introduce correlation between these samples, making (3.3) easier to satisfy.

6.1.3 Switch criteria

The switching criteria to another algorithm were alluded to above: while our Ising model example in Section 5.2 illustrated that it is in fact possible to get reasonable parameter estimates only using our algorithm, we believe it will be more computationally efficient to switch to another algorithm like Newton-Raphson or MCMC-MLE.

But when? The Faux Magnolia example in Section 5.4 showed that using our algorithm for $2d$ steepest ascent updates was enough, but we saw other trials where it was not. As noted in Section 3.6, MCMC-MLE converges if and only if the importance weights for the sample in (1.10) stabilize. Establishing criteria on these importance weights should make it possible to switch to MCMC-MLE only when it can attain the MLE in one update.

6.1.4 Monte Carlo standard errors

A final remaining issue is estimation of Monte Carlo error of the estimates. Here too we recommend switching to another algorithm at the end. The MCMC-MLE procedure gives accurate error estimates (Geyer, 1994; Hunter and Handcock, 2006). For very small steps these are essentially the same as the Monte Carlo error of a single unsafeguarded Newton-Raphson step, so the method in (Geyer, 1994) can be used for either.

References

Anderson, C. J., Wasserman, S. and Crouch, B. (1999). A p* primer: Logit models for social networks. *Social Networks*, **21** 36–66.

Andrieu, C., Moulines, E. and Priouret, P. (2005). Stability of stochastic approximation under verifable conditions. *SIAM Journal on Control and Optimization*, **44** 283–312.

Barndorff-Nielsen, O. (1978). *Information and Exponential Families in Statistical Theory*. John Wiley & Sons.

Bartz, K., Liu, J. and Blitzstein, J. (2009). Monte Carlo maximum likelihood for exponential random graph models: From snowballs to umbrella densities.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society,* Series B, **36** 192–236.

Besag, J. (1975). Statistical analysis of non-lattice data. *The Statistician*, **24** 179–195.

Brown, L. D. (1986). *Fundamentals of Statistical Exponential Families: with Applications in Statistical Decision Theory*. Institute of Mathematical Statistics, Hayward, CA.

Chan, K. S. and Geyer, C. J. (1994). Discussion of the paper by Tierny. *Annals of Statistics*, **22** 1747–1758.

Chen, H.-F. (2002). *Stochastic Approximation and Its Applications*. Kluwer Academic Publishers, Dordrecht.

Corander, J., Dahmström, K. and Dahmström, P. (1998). Maximum likelihood estimation for markov graphs. Tech. Rep. 8, Stockholm University, S-106 91 Stockholm, Sweden.

Erdös, P. and Rényi, A. (1959). On random graphs. I. *Publicationes Mathematicae*, **6** 290–297.

Fienberg, S. E. and Wasserman, S. S. (1981). Categorical data analysis of single sociometric relations. *Sociological Methodology*, **12** 156–192.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. John Wiley & Sons.

Frank, O. and Strauss, D. (1986). Markov graphs. *Journal of the American Statistical Association*, **81** 832–842.

Fukuda, K. (2004). Frequently asked questions in polyhedral computation. Tech. rep., Swiss Federal Institute of Technology. URL `http://www.ifor.math.ethz.ch/ fukuda/polyfaq/polyfaq.html`.

Fukuda, K. (2008). `cddlib` package, version 094f.

Geyer, C. J. (1990). *Likelihood and Exponential Families*. Ph.D. thesis, University of Washington. URL `http://purl.umn.edu/56330`.

Geyer, C. J. (1994). On the convergence of monte carlo maximum likelihood calculations. *Journal of the Royal Statistical Society,* Series B, **56** 261–274.

Geyer, C. J. (1996). Esitmation and optimization of functions. In *Markov Chain Monte Carlo in Practice* (W. R. Gilks, S. Richardson and D. J. Spiegelhalter, eds.). Chapman & Hall, London, UK, 241–258.

Geyer, C. J. (2009a). Likelihood inference in exponential families and directions of recession. *Electronic Journal of Statistics*, **3** 259–289.

Geyer, C. J. (2009b). More supporting theory and data analysis for "likelihood inference in exponential families and directions of recession". Tech. Rep. 673, University of Minnesota.

Geyer, C. J. (2009c). `mcmc`: Markov chain Monte Carlo. R pakage version 0.7-3., URL `http://CRAN.R-project.org/package=mcmc`.

Geyer, C. J. (2009d). `trust`: Trust Region Optimization. R package version 0.1-2., URL `http://www.stat.umn.edu/geyer/trust/`.

Geyer, C. J. (2010). `aster2`: Aster models. R pakage version 0.1, URL `http://CRAN.R-project.org/package=aster`.

Geyer, C. J. (forthcoming). Introduction to MCMC. In *Handbook of Markov Chain Monte Carlo* (S. P. Brooks, A. E. Gelman, G. L. Jones and X. L. Meng, eds.). Chapman & Hall/CRC, Boca Raton, FL.

Geyer, C. J. and Johnson, L. T. (2010). `potts`: Markov chain Monte Carlo for Potts models. R package version 0.4.

Geyer, C. J. and Meeden, G. D. (2009). `rcdd`: R package rcdd (C Double Description for R). R pakage version 1.1-3. Incorporates code from cddlib (ver 0.94f) written by Komei Fukuda, URL `http://cran.r-project.org/web/packages/rcdd`.

Geyer, C. J. and Thompson, E. A. (1992). Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society,* Series B, **54** 657–699.

Geyer, C. J. and Thompson, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association*, **90** 909–920.

Gilbert, E. N. (1959). Random graphs. *Annals of Mathematical Statistics*, **30** 1141–1144.

GOLDENBERG, A., ZHENG, A. X., FIENBERG, S. E. and AIROLDI, E. M. (2009). A survey of statistical network models. *Foundations and Trends in Machine Learning*, **2** 129–233.

GOODREAU, S. M. (2007). Advances in exponential random graph (p*) models applied to a large social network. *Social Networks*, **29** 231–248.

GOODREAU, S. M., HANDCOCK, M. S., HUNTER, D. R., BUTTS, C. T. and MORRIS, M. (2008). A statnet tutorial. *Journal of Statistical Software*, **24**. URL `http://www.jstatsoft.org/v24/i09`.

GOODREAU, S. M., KITTS, J. A. and MORRIS, M. (2009). Birds of a feather, or friend of a friend? Using exponential random graph models to investigate adolescent social networks. *Demography*, **46** 103—125.

GU, M. G. and ZHU, H.-T. (2001). Maximum likelihood estimation for spatial models by Markov chain Monte Carlo stochastic approximation. *Journal of the Royal Statistical Society,* Series B, **63** 339–355.

HANDCOCK, M. S. (2003). Assessing degeneracy in statistical models of social networks. Working Paper 39, University of Washington, Seattle.

HANDCOCK, M. S., HUNTER, D. R., BUTTS, C. T., GOODREAU, P. N., STEVEN M. KRIVITSKY, and MORRIS, M. (2010). `ergm`: A package to fit, simulate, and diagnose exponentail-family models for networks. Version 2.2-7. Project home page at `http://statnetproject.org`, URL `http://CRAN.R-project.org/package=ergm`.

HANDCOCK, M. S., HUNTER, D. R., BUTTS, C. T., GOODREAU, S. M. and MORRIS, M. (2003). `statnet`: Software tools for the statistical modeling of network data. Version 2.0. Project home page at `http://statnetproject.org`, URL `http://CRAN.R-project.org/package=statnet`.

HOLLAND, P. W. and LEINHARDT, S. (1981). An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association*, **76** 33–50.

Hummel, R., Hunter, D. R. and Handcock, M. S. (2010). A steplength algorithm for fitting ERGMs. Tech. Rep. 10-03, Pennsylvania State University.

Hunter, D. R., Goodreau, S. M. and Handcock, M. S. (2008a). Goodness of fit of social network models. *Journal of the American Statistical Association*, **103** 248–258.

Hunter, D. R., Goodreau, S. M. and Handcock, M. S. (2011). `ergm.userterms`: A template package for extendent `statnet`.

Hunter, D. R. and Handcock, M. S. (2006). Inference in curved exponential family models for networks. *Journal of Computational and Graphical Statistics*, **15** 565–583.

Hunter, D. R., Handcock, M. S., Butts, C. T., Goodreau, S. M. and Morris, M. (2008b). ergm: A package to fit, simulate and diagnose exponential-family models for networks. *Journal of Statistical Software*, **24**. URL `http://www.jstatsoft.org/v24/i03`.

Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik A Hadrons and Nuclei*, **31** 253–258.

Jaynes, E. (1978). Where do we stand on maximum entroy? In *The Maximum Entropy Formalism* (R. D. Levine and M. Tribus, eds.). Cambridge: Massassachusetts Institute of Technology Press.

Jones, G. L. (2004). On the Markov chain central limit theorem. *Probability Surveys*, **1** 299–320.

Knoke, D. and Yang, S. (2008). *Social Network Analysis*. 2nd ed. Sage Publications, Inc., Thousand Oaks, CA.

Kolaczyk, E. D. (2010). Tutorial: Statistical analysis of network data.

Kushner, H. J. and Yin, G. G. (1997). *Stochastic Approximation Algorithms and Applications*. Springer, New York.

Lehmann, E. L. (1983). *Theory of Point Estimation*. 1st ed. John Wiley & Sons.

Lehmann, E. L. and Casella, G. (1998). *Theory of Point Estimation*. 2nd ed. Springer.

Lehmann, E. L. and Romano, J. P. (2005). *Testing Statistical Hypotheses*. 3rd ed. Springer.

Liang, F. (2010). Trajectory averaging for stochastic approximation mcmc algorithms. *The Annals of Applied Statistics*, **38** 2823–2856.

Meyn, S. and Tweedie, R. (2009). *Markov Chains and Stochastic Stability*. 2nd ed. Cambridge University Press, Cambridge.

Morris, M., Handcock, M. S. and Hunter, D. R. (2008). Specification of exponential-family random graph models: Terms and computational aspects. *Journal of Statistical Software*, **24**.

Moyeed, R. A. and Baddeley, A. J. (1991). Stochastic approximation of the MLE for a spatial point pattern. *Scandinavian Journal of Statistics*, **18** 39–50.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. 1st ed. Springer.

Okabayashi, S. and Geyer, C. J. (2011). Long range search for maximum likelihood in exponential families. *Electronic Journal of Statistics*, **Revised and resubmitted.**

Okabayashi, S., Johnson, L. and Geyer, C. J. (2011). Extending pseudo-likelihood for Potts models. *Statistica Sinica*, **21** 331–347.

Padgett, J. F. (1994). *Marriage and Elite Structure in Renaissance Florence, 1282-1500*. Ph.D. thesis, Paper delivered to the Social Science History Association.

Pattison, P. and Wasserman, S. (1999). Logit models and logistic regression for social networks: Ii. multivariate relations. *British Journal of Mathematical and Statistical Psychology*, **52** 169–193.

Penttinen, A. (1984). Modelling interactions in spatial point patterns: parameter estimation by the maximum likelihood method. *Jyväskylä Studies in Computer Science, Economics and Statistics*, **7**.

Potts, R. B. (1952). Some generalized order-disorder transformations. *Proceedings of the Cambridge Philosphical Society*, **48** 106–109.

R Development Core Team (2010). R: A language and environment for statistical computing. ISBN 3-900051-07-0, URL `http://www.R-project.org/`.

Resnick, M. D., Bearman, P. S., Blum, R. W., Bauman, K. E., Harris, K. M., Jones, J., Tabor, J., Beunning, T., Sieving, R., Shew, M., Ireland, M., Bearinger, L. and Udry, J. R. (1997). Protecting adolescents from harm: Findings from the national longitudinal study on adolescent health. *Journal of the American Medical Association*, **278** 823—832.

Rinaldo, A., Fienberg, S. E. and Zhou, Y. (2009). On the geometry of discrete exponential families with application to exponential random graph models. *Electronic Journal of Statistics*, **3** 446–484.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, **22** 400–407.

Roberts, G. O. and Rosenthal, J. S. (1997). Geometric ergodicity and hybrid Markov chains. *Electronic Communications in Probability*, **2** 13–25.

Roberts, G. O. and Rosenthal, J. S. (2004). General state space Markov chains and MCMC algorithms. *Probability Surveys*, **1** 20–71.

Robins, G., Pattison, P., Kalish, Y. and Lusher, D. (2007a). An introduction to exponential random graph (p*) models for social networks. *Social Networks*, **29** 173–191.

Robins, G., Snijders, T., Wang, P., Handcock, M. S. and Pattison, P. (2007b). Recent developments in exponential random graph (p*) models fo social networks. *Social Networks*, **29** 192–215.

Rockafellar, R. T. (1970). *Convex Analysis*. Princeton University Press.

Rockafellar, R. T. and Wets, R. J.-B. (2004). *Variational Analysis*. corrected second printing, Springer-Verlag, Berlin.

Salgado, H., Santos-Zavaleta, A., Gama-Castro, S., Millán-Zárate, D., Díaz-Peredo, E., Sánchez-Solano, F., Pérez-Rueda, E., Bonavides-Martínez, C. and Collado-Vides, J. (2001). Regulondb (version 3.2): Transcriptional regulation and operon organization in *Escherichia coli* k-12. *Nucleic Acids Research*, **29** 72–74.

Sampson, S. F. (1968). *A novitiate in a period of change: An experimental and case study of relationships*. Ph.D. thesis, Cornell University.

Saul, Z. M. and Filkov, V. (2007). Exploring biological network structure using exponential random graph models. *Bioinformatics*, **23** 2604–02611.

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H. and Etterson, J. R. (2008). Unifying life-history analyses for inference of fitness and population growth. *The American Naturalist*, **172** E35–E47.

Shen-Orr, S. S., Milo, R., Mangan, S. and Alon, U. (2002). Network motifs in the transcriptional regulation network of *Escerichia coli*. *Nature Genetics*, **31** 64–68.

Snijders, T. A. B. (2002). Markov chain Monte Carlo estimation of exponential random graph models. *Journal of Social Structure*, **3**.

Snijders, T. A. B., Pattison, P. E., Robins, G. L. and Handcock, M. S. (2006). New specifications for exponential random graph models. *Sociological Methodology*, **36** 99–153.

Strauss, D. and Ikeda, M. (1990). Pseudolikelihood estimation for social networks. *Journal of the American Statistical Association*, **85** 204–212.

SUN, W. and YUAN, Y.-X. (2006). *Optimization Theory and Methods: Nonlinear Programming*. Springer.

SWENDSEN, R. H. and WANG, J.-S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Physics Review Letters*, **58** 86–88.

URBANEK, S. (2010). `multicore`: Parallel processing of R code on machines with multiple cores or CPUs. R pakage version 0.1-3, URL `http://www.rforge.net/multicore/`.

VAN DUIJN, M. A. J., GILE, K. J. and HANDCOCK, M. S. (2009). A framework for the comparison of maximum pseudo-likelihood and maximum likelihood estimation of exponential family random graph models. *Social Networks*, **31** 52–62.

WANG, J. S. and SWENDSEN, R. H. (1990). Cluster Monte Carlo algorithms. *Physics A*, **167** 565–579.

WASSERMAN, S. and FAUST, K. (1994). *Social Network Analysis: Methods and Applications*. Cambridge University Press.

WASSERMAN, S. and PATTISON, P. (1996). Logit models and logistic regression for social networks: I. an introduction to markov graphs and p*. *Psychometrika*, **61** 401–425.

WELCH, D., BANSAL, S. and HUNTER, D. R. (to appear). Statistical inference to advance network models in epidemiology. *Epidemics*.

WOOD, S. (2011). `mgcv`: GAMs with GCV/AIC/REML smoothness estimation and GAMMs by PQL. R pakage version 1.7-4., URL `http://cran.r-project.org/web/packages/mgcv`.

YOUNES, L. (1988). Estimation and annealing for Gibbsian fields. *Ann. Inst. Henri Poincare*, **24** 269–294.

YOUNES, L. (1989). Parametric inference for imperfectly observed gibbsian fields. *Probability Theory and Related Fields*, **82** 625–645.

Appendix A

Brute Force Network Statistic Counting

As noted in Section 1.2, even for an undirected 9-node network, there are $2^{\binom{9}{2}}$, or about 69 billion different possible graphs. Counting the network statistics of edges, two-stars, and triangles for this network is not a trivial calculation and can take an enormous amount of time if not coded efficiently (our first straightforward implementation entirely in R would have taken over a year). Thus every effort must be made to represent the data as efficiently as possible, implementing loops in C and avoiding calculations. We have adapted some of these ideas from code accompanying Rinaldo et al. (2009).

A few quantities are easily obtained: the maximum number of edge is $\binom{n}{2} = \binom{9}{2} = 36$. The maximum number of triangles is $\binom{n}{3} = \binom{9}{3} = 84$, since a triangle requires 3 of the 9 actors. For each triangle, there are 3 different two-stars, so the maximum number of two-stars is $3 \cdot 84 = 252$.

The approach we have implemented requires first creating a collection of graphs, one graph for each possible network structure of interest (triangle or two star). Each graph in this collection is empty except for one network structure. For example, one graph in this collection is the one representing a triangle between the 1st, 2nd, and 3rd nodes. In matrix form, this network has all zeros except for ones in entires of (1,2), (1,3), (2,3), (2,1), (3,1), (3,2). We then compare each of the 69 billion possible graphs to this collection. Every time a comparison yields a match, that particular configuration is present and the count for that statistic is incremented by one. The

reason this is efficient is because we can actually represent a graph as a single binary number and do bitwise comparisons for far greater speed.

We explain this in further detail with triangles. We know there are $\binom{9}{3} = 84$ possible triangles, and so we create 84 graphs, each with only one triangle present among 3 actors. So, the first graph in this collection is a graph with ties present between actors 1, 2, and 3, and no other ties present, the second is a graph with ties present between actors 1, 2, and 4. Because the network is undirected, the adjacency matrix is symmetric and can be fully described by just its flattened upper triangle, which in our convention of going down vertically and then across is $(1, 1, 1, 0, 0, 0, \ldots, 0)$. We can treat this vector as a 36-digit number, `111000000000000000000000000000000000`. If we think of this as a number in base 2, we can convert it to a base 10 number less than 2^{36}. For this first matrix, it is 60,129,542,144. Proceeding in this manner, we will have a collection of 84 numbers corresponding to all the possible triangles in the network.

We now turn our attention to iterating through the $2^{36} \approx 69$ billion possible graphs. Using the `long int` representation in C (64-bit), we can in fact loop from 0 to $2^{36} - 1$. We can do the reverse process of the above, converting the loop index to a 36-digit number that is the collapsed upper triangle of a specific graph. For example, the last number, $2^{36} - 1$ has binary form `111111111111111111111111111111111111` which is the complete graph.

For each graph index, we can loop through our set of 84 triangle numbers and perform bitwise logic operations (the `&` operator in C) to compare the binary form of the graph index to each of the triangle numbers. In binary form, if there are ones in all the digits that the triangle number has a one in, then the graph has this particular triangle present. For example, the operation $(2^{36} - 1)$ & $60,129,542,144$ would return `111000000000000000000000000000000000` in binary form, indicating that this particular triangle is present. This is confirmed by comparing the result of the binary operation back to the original triangle number, which would return TRUE. Proceeding through all the other triangle numbers, we get a count of how many triangles are present for this graph index.

A similar method is employed for counting two-stars, where we would similarly

first calculate the $3 \times 84 = 252$ two-star graph numbers before iterating through all 69 billion graphs.

To count edges, there are well-known tricks to count the number of ones in a binary number in C which cleverly use the '`>>`' shift operator. By taking the graph index `>> 1`, it moves all the digits to the right by dropping off the rightmost digit and adding a 0 to the farthest left digit. So,

```
111111111111111111111111111111111111 >> 1
```

returns

```
011111111111111111111111111111111111
```

where there are now 35 ones instead of 36. In this manner, the shifts can be continued and ones counted (done by another bit comparison to '01' using the '`&`' again) until there are no more ones. This approach avoids any arithmetic and is thus much faster. It should be noted that a computer stores the `long int` in binary form and so the 36-digit representation used in this section is entirely for our benefit—no actual conversions need be programmed by us.

Finally, we can further speed up the calculations by parallelizing this computation, in our case across 8 CPUs. We can count the number of edges, two-stars, and triangles in the first $2^{36}/8$ graphs at the same time that we count them in the last $2^{36}/8$ graphs. To do this, we simply use the `mclapply` function in the `multicore` (Urbanek, 2010) package in R. At the time of this writing, we performed this calculation on an 8-CPU 2.9GHz linux box which took 2 hours 15 minutes to complete.