Exponential random graphs

Alan Terry

Dissertation submitted for the MSc in Data Analysis,
Networks, and Nonlinear Dynamics,
Department of Mathematics, University of York, UK

Carried out at:
Complexity Research Group, BT, Martlesham
Supervisor: Keith Briggs

August 22, 2005

Contents

Abstract

The accurate modelling of real-world networks has significant implications for many areas of every day life. Our understanding of transport networks, the internet and world wide web, and the spread of diseases, could be dramatically enhanced by improved modelling techniques. Here we examine a very general concept, the exponential random graph (ERG). We derive the ERG model from maximum entropy principles and assess its appropriateness as a generic tool for network analysis and prediction. Both undirected and directed networks will be considered. It will be seen that in certain simple examples the analysis is elegant and the predictions unsophisticated. For less trivial cases we explore the theory and applicability of Markov Chain Monte Carlo simulations. We will find that simulations of up to 30 million steps may yield inconclusive results. Suggestions for further research will therefore be made.

Acknowledgements

Many thanks must go to my supervisor, Keith Briggs, whose help and guidance were invaluable throughout the project. I offer my gratitude moreover to BT, the company at which I was based for these four months. The people were friendly and the environment was stimulating.

Chapter 1

Exponential random graphs

"Home computers are now being called upon to perform many new functions, including the consumption of homework formerly eaten by the dog."

Random quote, Doug Larson

1.1 Modelling networks

If we ask a group of people, such as the students in a class, which of the other people in the group they consider to be friends, we may represent the information gleaned as a social network. The students are represented as nodes and there is a directed edge (i, j), from i to j, if person i considers person j to be a friend. An example of such a network is given in figure 1.1. Some basic graph definitions are given in table 1.1.

Networks, directed or undirected, can be used to represent a very wide variety of self-interactive systems. Besides social networks modelling friendship choices [66, 77], there are, for instance:

- Social networks representing
  - business relationships between companies [62, 63]
  - diseases spreading [43, 60]
- Computer and information networks like
  - the internet [32]
  - the world wide web [4, 12, 17, 49]

directed graph or network	A *directed graph* or *network* is an ordered pair of disjoint sets (V, E), where V is a non-empty set of *nodes* or *vertices*, and E is a set of ordered pairs of nodes. The elements of E are *edges*. An edge (i, j) is said to go *from* node i *to* node j. A graph with n nodes is said to be *on* n nodes.
undirected	An *undirected graph* or *network* is defined in the same way as a directed graph, except that E now contains *unordered* pairs of nodes. (Some authors consider a network to be both directed and weighted, for example [92]. We do not. And we do not consider weighted graphs here.)
adjacency matrix	The *adjacency matrix* a of a graph with n nodes is the $n \times n$ matrix where the value a_{ij} of the entry in row i and column j is the number of edges from i to j. (Note that an undirected graph therefore has a symmetric adjacency matrix.)
simple	An undirected graph is *simple* if there is no more than one edge connecting any pair of nodes. A directed graph is *simple* if no more than one edge connects any *ordered* pair of nodes. All graphs in this project are assumed to be simple unless otherwise stated.
isomorphic	Two graphs g_1 and g_2 are *isomorphic* if there is a permutation mapping the nodes of g_1 to g_2 that preserves adjacency. In the sets (or ensembles) of graphs considered in this project, isomorphisms will be allowed unless otherwise stated.
self-loop	A *self-loop* is an edge (i, j) such that node i = node j. All graphs in this project are assumed to have *no* self-loops.
degree	Undirected graph with no self-loops: The *degree* of a node i is the number of edges of the form (i, j) where j is any node. Directed, no self-loops: The *out-degree* of node i is the number of edges (i, j) where j is any node. The *in-degree* of node i is the number of edges (j, i) where j is any node.
null, complete	The *null* graph on n nodes has no edges and the *complete* graph on n nodes has every edge allowed [92]. (Some authors may define the null graph to have no nodes as well. We do not.)

Table 1.1: Some basic graph definitions

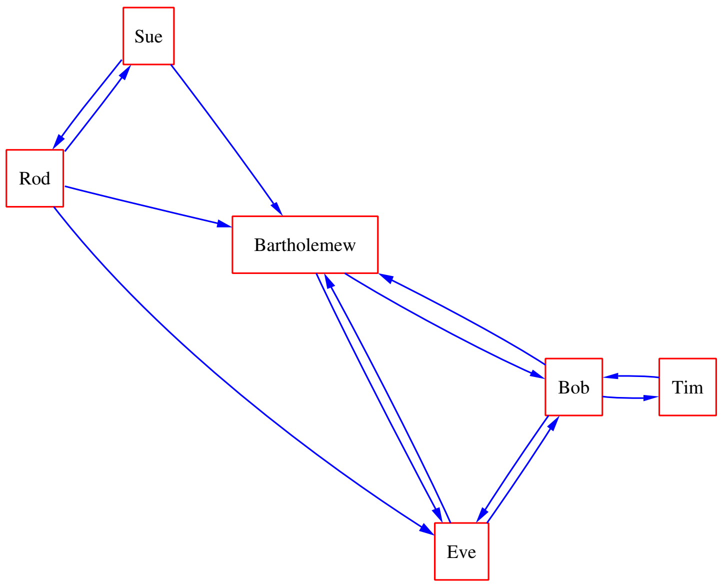

Figure 1.1: Example of a social network: friendship choices. (An arrow from a node i to a node j means i considers j to be a friend.)

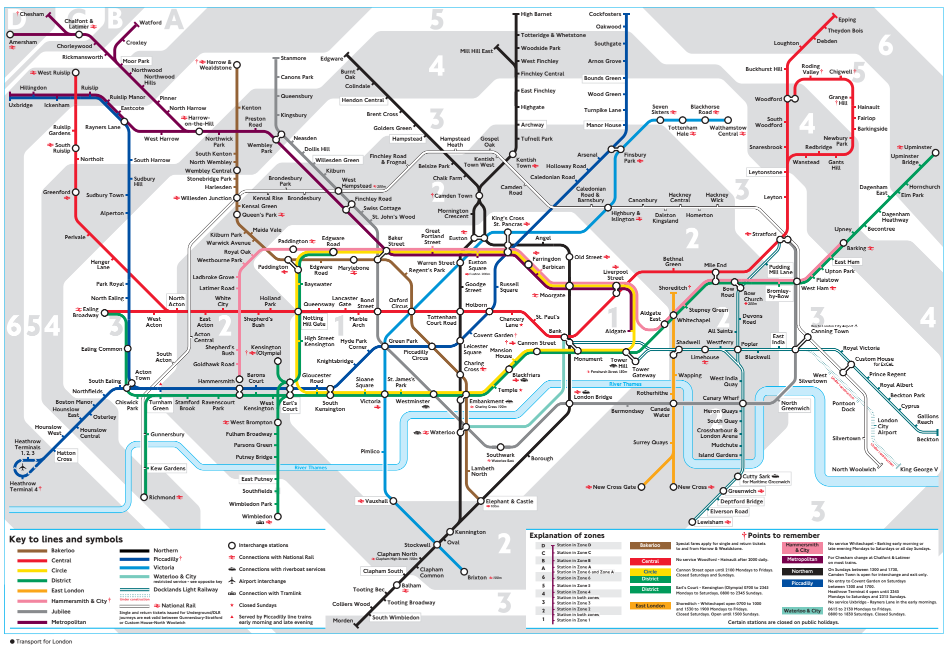

Figure 1.2: London underground map

- Traffic networks such as
  - roads [56]
  - railways [80]
  - metros [57] (figure 1.2)
  - aviation routes [5]
- And biological networks like
  - food webs [19, 76, 65, 55]
  - metabolic networks [52, 34, 85]

Our intention is to lay down a broad framework for the study of all networks.

1.1.1 Commonly observed features of real-world networks

Empirical observations of many kinds of network reveal numerous features that are intuitively obvious. Firstly, there is typically a high level of reciprocation [68, 54, 22] (defined in table 1.3), higher than chance would seem to allow [66]. In figure 1.1, for example, we see that if one student i claims that another student j is a friend, then it is rather likely that student j will claim that i is a friend. Intuitively we may say that a person is unlikely to be friends with someone else if they don't get friendship in return. As a second example, consider road traffic networks. If flow is allowed along a road between two locations, it is normally allowed in both directions. But notice that there are situations that preclude reciprocation, such as the social relationship of power [38, 39]. Essentially, if person i is more powerful than person j, then j is not more powerful than i!

Another widely observed feature is transitivity (table 1.3). In 1970 Davis found in an extensive empirical study of positive interpersonal affect [23] that transitivity is the crucial factor in differentiating observed data from a pattern of random edges. Transitivity is encapsulated by the old adage "A friend of a friend is a friend", and it is easy to see how it might arise - if a person i knows a person j, and person j knows person k, then i may well be expected to know k. On the other hand, the processes that lead to transitivity can also be somewhat involved [83].

A third feature commonly seen is differential attractiveness [48]. In the friendship pattern of figure 1.1, for instance, Bartholomew is significantly more popular, more attractive to the other students, than Tim. In the case of the world wide web, some websites are far more popular than others. In food webs, certain animals are much easier to prey on or catch. In aviation networks, certain cities make more attractive holiday destinations. Many studies have shown that the range of popularity of the nodes in various networks, which is given by the degree distribution (see table 1.3), is by no means inherently random. Often the degree distribution is seen to fit a power-law [3, 25, 64]. Examples of this include:

- the world wide web [4, 12, 17]
- the internet [32, 87]
- telephone call graphs [1]
- networks of human sexual contacts [60]

A network with a power-law degree distribution is sometimes also called *scale-free* [71]. The degree distribution may otherwise fit an exponential distribution [5, 80] and sequences of greater complexity may also arise [5, 26, 69].

Arguably more important than any of the above features, however, is the fact that many networks are not constant in time, that is, they are *dynamic* [10, 47, 89]. The world wide web is growing steadily [24], new roads are always being built [56], and loyalties or friendships shift [53]. It is clear, then, that any attempt to construct a general framework for modelling networks will require some powerful tools.

1.2 Modelling networks: a brief history

How have networks been modelled to date? Until the development and spread of computers, network modelling was somewhat restricted and the field has only sprung to life comparatively recently. Having said that, the first truly general model was introduced by Solomonoff and Rapoport in 1951 [84]. They considered the collection of all undirected graphs (without self-loops or multiple edges) on a fixed number of nodes n on which edges existed with a constant probability p that was independent of the nodes it connected. They thereby derived a manner of modelling graphs that were in an obvious sense inherently random and indeed their construction is known as the random graph model. It was studied fairly extensively by Erdős and Rényi in the late 1950s and early 1960s [29, 30, 31]. Also called the Bernoulli model, it was moreover the first example of an exponential random graph (ERG) model. We will meet it again in chapter 2. ERG models will be the focus of this project.

As we shall see, ERG models involve a discrete probability density function containing an exponential family. This probability function is defined over a set of graphs. Hence the terms *exponential* and *random graph* in the name of the model.

In 1981 Holland and Leinhardt classified the literature on social networks prevalent at the time into three types: tests of randomness, pattern detection, and measures of structure [46]. Table 1.2 gives examples of these types. Holland and Leinhardt in addition drew attention to the paucity of statistical tools available for social network analysis. They commented on the sparsity of papers that considered the fitting and estimating of parametric probability models for digraph (directed graph) data and, building on the work of Besag (1974) [15], constructed

Class of network	Example	Reference
Tests of randomness	The random distribution of zero blocks in an adjacency matrix	[91]
Pattern detection	Clique-finding algorithms	[74, 2]
	Block-modelling procedures	[16]
	Spatial representations of digraphs	[59]
Structural measures	Connectivity	[61, 13]
	Centrality	[67, 36, 37]

Table 1.2: Social network classification as at 1981 [46]

a fairly specific model of this kind to "begin to fill the gap", as they saw it. Their construction was the second example of an ERG model.

They called it the p_1 model. Sometimes ERG models are called p^* models as a generalisation of the p_1 model [88], although in fact p^* models are slightly more general than ERGs.

In the mid 1980s Frank and Strauss [35] developed ERG models substantially, and further developments were made by others throughout the 1990s [88, 6]. In the last five years or so an increasing number of physicists have conducted theoretical studies of specific cases [14, 75, 18]. ERG models are now in common use within the statistical and social network analysis communities as a practical aid and there are even standard computer packages available for simulating and manipulating them, such as ERGM, PREPSTAR, and SIENA [83, 82].

But other network models have also emerged since the 1980s, which, like ERG models, are not defined as a single network but as a probability distribution over many possible networks. Examples are the small-world model [90] and various different preferential attachment or scale-free models [11, 27], which respectively model transitivity and power-law degree distributions. These two models (but not necessarily ERG models) begin with the intention of modelling real-world networks with *particular* properties. Mechanisms that may be responsible for these properties are identified and incorporated into the model and the networks produced are typically rewarded for their similarity to the real-world network and used as a tool for further modelling.

As a *general* framework for network modelling, however, the only candidate to have so far emerged is the ERG model, a rigorous derivation of which will now be presented. How we actually use the model is deferred to section 1.4 and beyond.

An excellent discussion of network modelling is given in [71].

1.3 The ERG Model

Suppose we have a real-world network or graph (the terms will be used interchangeably). We define its *graph observables* (or simply observables) to be its measurable properties. (In the literature the observables are sometimes, slightly mistakenly, called the sufficient statistic [82, 50].) Examples of observables are the number of edges or the number of nodes of degree 2 and more are given in table 1.3. The network may not be constant in time, but we shall assume that it is changing slowly enough over a given period to have essentially constant *expected* values for its graph observables over that period. For convenience the number of nodes will be assumed to be fixed. There is no reason in principle why this has to be so.

Imagine that we have taken one or more *measurements* from the network for one or more of its graph observables. For each of these observables we find the average value of its measurements. We shall assume that these average values are the expected values for the real-world network for the observables in question, since we have nothing else to use as estimates for their expected values. We would like to say as much as possible about the network given this limited knowledge of it. Then we have a problem of inference from incomplete information.

Now there is a prominent 'Bayesian' school of thought, popularised above all by Edwin Jaynes [51], which considers such problems from a purely statistical point of view and which maintains that the most correct manner of solving them resides in maximising an entity called *Gibbs' entropy* or simply entropy. In statistical mechanics this approach is already known to give good results, and the extension to general network problems is in many ways quite natural [72]. Therefore we shall also adopt this approach. (Our derivation of the ERG model will in fact parallel the derivation of the Boltzmann distribution in statistical mechanics [42].)

We define the *ensemble* G for our real-world network to be the collection of possible configurations that it may reasonably be expected to attain. This obviously includes any observed configurations of the network! If we know that it has n nodes and there is no reason to think that n will change, or change significantly, and if we know that it is undirected, then a sensible choice for G might be all undirected graphs on n nodes.

Thus, we have one or more measurements of one or more graph observables, these have given us expectation values for the observables in question, and we suppose that we have chosen a sensible ensemble G. (Indeed a summary of the ingredients of the model are given in table 1.4.) We now use these expectations to derive a probability distribution P over the ensemble. Our intention is to choose P so that, for the observables that we have taken measurements of, expected values over the ensemble equal their measured estimates. To do this, we will employ the method of Lagrange multipliers to maximise the entropy, building in the conditions on the expectations and a normalising condition for P as constraints.

Observable	Definition
edges	The total number of *edges* of a graph
number of twostars	For an undirected graph: a *twostar* is a distinct pair of edges with a common node. For a directed graph: Denote by (i,j) an edge *from* i *to* j. Then an *in-twostar* is a distinct pair of edges (i,h) and (j,h), an *out-twostar* is a pair $(h,i),(h,j)$, and a *twopath* or *mixed-star* is a pair $(i,h),(h,j)$.
number of k-stars	An intuitive generalisation of twostars
degree sequence	Undirected graph: the *degree sequence* is the set of degrees of the nodes, sometimes written in non-decreasing order. Directed: Same idea, but here we can have *in-degree sequence* and *out-degree sequence*
degree distribution	Undirected graph: The *degree distribution* is the set of values $n_0, n_1, \ldots, n_{\max}$ where n_i is the number of nodes of degree i and max is the maximum possible degree. Directed: Same idea, but here we can have *in-degree distribution* and *out-degree distribution*
reciprocity	For a directed graph, an edge (i,j) is *reciprocated* if there is also an edge (j,i). The *reciprocity* is the number of node pairs with edges running between them in both directions.
transitivity (clustering coefficient)	Undirected graph: the *clustering coefficient* C is the average probability that two neighbours of a given node are also neighbours of one another, where a neighbour of node i is a node j such that (i,j) is an edge [90]. Directed graph: Similar but more possibilities
mean shortest path	Undirected graph g: A *path* from a node i to a node j is a sequence of nodes $i = n_1, n_2, \ldots, n_{t-1}, n_t = j$, such that for any two consecutive nodes n_l, n_{l+1}, there is an edge (n_l, n_{l+1}). A path $n_1, \ldots, n_t$ has length $t-1$. If there is a path between any two nodes in g, then g is *connected*. If g is connected, the *mean shortest path* is the mean value of the shortest paths connecting all the nodes. The directed graph case is analogous but more complicated.
diameter	For an undirected connected graph: The *diameter* is the maximum shortest path. Directed connected: Similar
girth	For an undirected graph g: A *cycle* is a path $n_1, n_2, \ldots, n_t$ such that $n_1 = n_t$ and otherwise $n_i \neq n_j$. The *girth* is the length of the shortest cycle, which is the length of the cycle with fewest nodes. If a graph has *no* cycles, we could set girth $= 0$. Directed case: Similar

Table 1.3: Examples of graph observables

Ingredient	Description/assumptions	Examples	Notation
real-world network	The network is not evolving quickly. Usually the number of nodes n is assumed fixed.	See section 1.1	No standard notation
ensemble	The set of all possible graphs that the real-world network may reasonably be expected to become	If the real-world network is undirected on n nodes, a possible ensemble is all undirected graphs on n nodes.	Ensemble = G; an element of G is g
graph observables	Measurable properties of a graph	See table 1.3	r observables may be denoted $x_1, x_2, \ldots, x_r$
measured estimates	Measured values of particular graph observables for a real-world network. Used to give estimated expected values for the observables.	A single measurement of the number of edges of a road network	Estimated expected values for the r observables $x_1, \ldots, x_r$ may be denoted $\langle x_1 \rangle, \ldots, \langle x_r \rangle$

Table 1.4: The ERG machinery: Ingredients of the model

To be more specific, let $\{x_i\}$, $i = 1, 2, \ldots, r$ be the observables that we have chosen to measure. Denote by $\langle x_i \rangle$ the measured estimate of the expectation of x_i. Also, let $x_i(g)$ be the value of an observable x_i for a graph g. Finally let E[] denote the *expectation operator*. Then for the expectation conditions or constraints we write

$$\mathrm{E}[x_i] = \sum_{g \in G} P(g) x_i(g) = \langle x_i \rangle \tag{1.1}$$

for $i = 1, 2, \ldots, r$. The normalising condition is simply

$$\sum_{g \in G} P(g) = 1, \tag{1.2}$$

and Gibbs' entropy is defined as

$$S = -\sum_{g \in G} P(g) \ln P(g). \tag{1.3}$$

We are now in a position to introduce Lagrange multipliers to maximise S subject to constraints (1.1) and (1.2). Before doing so, it is instructive to give an intuitive understanding of why this is worthwhile. In maximising the entropy the effect is to make as random as possible, given our limited knowledge of just a few expectation values, those observables that have no dependence on what we have measured. This is the crucial advantage that ERG models have over other models. ERG models do not unknowingly mess around with graph properties that we are not measuring or 'controlling'. Notice further that by maximising the entropy we are circumventing the issue of having a vastly underdetermined problem. In choosing P to satisfy (1.1) and (1.2) there are, in most cases, far more degrees of freedom than constraints.

Let the Lagrange multipliers be α, $\{\theta_i\}$ for $i = 1, 2, \ldots, r$. Then the maximum entropy is achieved for the distribution satisfying

$$\frac{\partial}{\partial P_\theta(g)} \left[S + \alpha \left(1 - \sum_{g \in G} P_\theta(g) \right) + \sum_{i=1}^{r} \theta_i \left(-\langle x_i \rangle + \sum_{g \in G} P_\theta(g) x_i(g) \right) \right] = 0 \tag{1.4}$$

for all graphs $g \in G$ where the subscript θ denotes parametric dependence on the θ_i. We have included the constraints in such a way as to avoid unattractive minus signs later. Equation (1.4) yields

$$\ln P_\theta(g) + 1 - \alpha - \sum_{i=1}^{r} \theta_i x_i(g) = 0 \tag{1.5}$$

or alternatively

$$P_\theta(g) = \frac{e^{H_\theta(g)}}{Z_\theta} \tag{1.6}$$

where $H_\theta(g)$ is called the *Hamiltonian* (a term borrowed from statistical mechanics) of g

$$H_\theta(g) = \sum_{i=1}^{r} \theta_i \, x_i(g) \tag{1.7}$$

and $Z_\theta = e^{1-\alpha}$ is a normalising constant called the *partition function*. Substituting (1.6) into (1.2) gives

$$Z_\theta = e^{1-\alpha} = \sum_{g \in G} e^{H_\theta(g)}. \tag{1.8}$$

We say that a parameter θ_i is *coupled* to a graph observable x_i if it is the coefficient of x_i in the Hamiltonian. Now we may rewrite the expectation constraints (1.1) as

$$\langle x_i \rangle = \frac{1}{Z_\theta} \sum_{g \in G} x_i(g) \exp\left(\sum_{i=1}^{r} \theta_i x_i(g) \right) \tag{1.9}$$

for $i = 1, 2, \ldots, r$. We may further re-express (1.9) as

$$\langle x_i \rangle = \frac{1}{Z_\theta} \frac{\partial}{\partial \theta_i} Z_\theta = \frac{\partial}{\partial \theta_i} (\ln Z_\theta) = \frac{\partial}{\partial \theta_i} F_\theta, \tag{1.10}$$

where $F_\theta = \ln Z_\theta$ is defined as the *free energy*.

Equations (1.6) to (1.9) define the ERG model. (The normalising condition is included in the model as equation (1.8).) To initiate the model, as it were, we first solve the expectation constraints (1.9) for the θ_i. This allows us to calculate the Hamiltonians in (1.7), which in turn allows the calculation of Z_θ (and α, though we never use α explicitly) in (1.8). And now we can find the probabilities in (1.6). Throughout the text we shall refer often to the idea of 'initialising' the model, by which we will mean solving both the expectation constraints *and* finding the partition function. We shall call a specific instance of the ERG model *an* ERG model. The equations of the ERG model are collected together for ease of reference in table 1.5.

The expectation constraints (1.9) consist of r equations in r unknowns. This sounds sensible enough but does not in itself guarantee the existence of a solution. It will be of practical significance for us to be aware of the kinds of solutions that are possible. We would like to know, for example, if it is possible in some cases for there to be no solutions? Can there be a unique solution? If so, when? If there is more than one solution, which should we choose? Can there be infinitely many? Some of these questions will be raised again in chapter 4.

1.4 Using the ERG model

We have derived the model to possess specific expectation values over the ensemble for those graph properties that we have measured. These expectation values have been set to equal our measured estimates. Then, one way to use the model

normalising constraint	$\sum_{g\in G} P_\theta(g) = 1$
expectation constraints	$\langle x_i\rangle = \frac{1}{Z_\theta}\sum_{g\in G} x_i(g)\exp\left[\sum_i \theta_i x_i(g)\right]$ for $i = 1, 2, \ldots, r$ when there are r observables
expectation constraints in terms of free energy	$\langle x_i\rangle = \frac{1}{Z_\theta}\frac{\partial}{\partial\theta_i}Z_\theta = \frac{\partial}{\partial\theta_i}F_\theta$ for $i = 1, 2, \ldots, r$ when there are r observables
free energy	$F_\theta = \ln Z_\theta$
probability formula	$P_\theta(g) = \frac{e^{H_\theta(g)}}{Z_\theta}$
Hamiltonian	$H_\theta(g) = \sum_{i=1}^r \theta_i\, x_i(g)$ when there are r observables
partition function	$Z_\theta = e^{1-\alpha} = \sum_{g\in G} e^{H_\theta(g)}$

Table 1.5: The ERG machinery: Equations of the model

is to calculate the expectation value $\mathrm{E}_\theta[x]$ over the ensemble for a graph property x that we have *not* measured. Our calculated value would be

$$\mathrm{E}_\theta[x] = \sum_{g\in G} P_\theta(g)x(g). \tag{1.11}$$

This would be the best prediction for $\mathrm{E}_\theta[x]$ given our limited knowledge of the network, owing to the derivation of the model from maximising entropy [72].

The probability attached to a graph in the ensemble may be thought of as the probability of our real-world network attaining that particular configuration at any given time. Typically we will have a large ensemble so these probabilities are likely to be very small. It may therefore be more meaningful to not consider individual probabilities but rather to consider their ratios. We may like to know, for instance, if graph g_1 is more likely than graph g_2. We would find out by computing

$$\frac{P_\theta(g_1)}{P_\theta(g_2)} = \frac{e^{H_\theta(g_1)}/Z_\theta}{e^{H_\theta(g_2)}/Z_\theta} = e^{H_\theta(g_1)-H_\theta(g_2)}. \tag{1.12}$$

Clearly, then, $P_\theta(g_1) > P_\theta(g_2)$ if $H_\theta(g_1) > H_\theta(g_2)$. Observe that the only real computation now required is to find for g_1 and g_2 the values of the graph properties that we have chosen to measure (assuming that θ has been found from solving equation (1.9)).

But with a very large ensemble it is still not terribly meaningful to consider specific graphs. If the ensemble G is all undirected graphs on n nodes, then the order of G is $2^{\binom{n}{2}}$, which, for large n, is simply colossal. The corresponding probabilities over the ensemble will then in all likelihood be absolutely tiny. It may therefore make sense to consider collections of graphs satisfying a particular condition. For example, we may wish to estimate the proportion T of the time that our real-world network has more than twice the expected number of edges. In fact, if $m(g)$ denotes the number of edges of a graph $g \in G$ and $\mathrm{E}_\theta[m]$ is the expected number of edges over the ensemble, we can write

$$T = \sum_{\substack{g \in G \\ m(g) > 2\mathrm{E}_\theta[m]}} P_\theta(g). \tag{1.13}$$

This is a sensible application, notwithstanding the computational difficulties that may be involved for a large ensemble. Computational difficulties will be discussed in chapter 5.

1.5 A look ahead

Some ERG models may be simplified sufficiently to be considered exactly soluble. We shall be more precise in chapter 2 when we discuss several examples of such models. In chapter 3 we will systematically construct two new soluble models in seeking to create a general method for the construction of soluble models. We shall return in chapter 4 to the issue of when solutions may exist to the expectation constraints (1.10) and how to find them, resorting in particular to a well-known function, the likelihood. The final chapter will focus on the extensive computation usually needed to solve ERG models and will end with a real-world application.

Chapter 2

Soluble models

"The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work."

John von Neumann

2.1 What should we measure?

Imagine we have a particular real-world network and that we want to model it. We have the ERG machinery, that is, equations (1.6) to (1.9), at our disposal and we wish to set it in motion. Which graph observables x_i should we measure? Ideally we should measure those observables that we are most interested in predicting or learning about. This is because, if we want to ask for the probability of a graph with specific values for specific observables, equation (1.6) only allows this if the observables are in the Hamiltonian, that is, if they are the observables for which we have taken actual measurements. (But notice that we can still find expected values over the ensemble for observables for which we don't have actual measurements.)

It should also be borne in mind that the model is made better by having more observables (because then we provide it with more information), so we should theoretically have as many as possible. Now a directed network on n nodes is completely defined by the number of edges between each ordered pair of nodes. There are $2\binom{n}{2}$ such pairs. The model would attain its greatest theoretical accuracy then, if we had a separate observable for each of the $2\binom{n}{2}$ ordered pairs of nodes. But, for example, the world wide web has *billions* of static nodes [41, 8], which would mean billions of billions of observables. We would therefore have to solve billions of billions of expectation constraint equations (equation (1.9)) to be able to initiate the model. And even if we did do this the model would essentially be useless because the calculation of $P_\theta(g)$ for any g would itself require billions of billions of calculations. This computational restriction would remain severe for any sizeable subgraph of the world wide web and large *un*directed networks will face similar restrictions.

Finally it may be difficult to take measurements of certain observables for a large network. The information may just not be available or may take so long to acquire as to be out of date before it can be used! (The idea of modelling networks with changing expected values for their observables was mentioned at the start of section 1.3.)

Thus, in practice, our choice of observables will be determined by what we want to predict or learn about, how good a model we want, how powerful our computers are, and what it is possible to measure. It should further be noted that a better model is likely to result from taking measurements of a real-world network at several different points in time, instead of just once, for then our estimated expectations for the observables will probably be more accurate.

2.2 Measuring the total number of edges

We have already mentioned the Bernoulli model (section 1.2) as the first ERG model to be created. To motivate a proper description of it, we suppose, as ever, that we have a real-world network and wish to model it. We suppose, further, that the network has n nodes and is undirected with no self-loops or multiple edges, and that we have chosen, for whatever reason, to consider only one graph observable, namely the total number of edges m. We denote by θ the parameter coupled to m in the Hamiltonian. A sensible choice for the ensemble G is all undirected graphs on n nodes, with no self-loops or multiple edges, and we make this choice. The ERG model subject to these assumptions is then the Bernoulli model and it has the attractive property of being exactly soluble.

Definition 2.1. *An ERG model is* exactly soluble *if both the partition function Z_θ (equation (1.8)) and the expectation constraints (equation (1.9)) may be simplified enough as to not involve a sum over the ensemble.*

We derive these simplified equations for the Bernoulli model now, using ideas in [72].

The Hamiltonian is just

$$H_\theta(g) = \theta m(g) \tag{2.1}$$

where $m(g)$ is the number of edges of a graph $g \in G$. Since the number of edges of an undirected graph g may be defined in terms of its adjacency matrix a as

$m(g) = \sum_{i<j} a_{ij}$, we may simplify the partition function as follows:

$$\begin{aligned} Z_\theta &= \sum_{g \in G} e^{H_\theta(g)} = \sum_{\{a_{ij}\} \in G} \exp(\theta \sum_{i<j} a_{ij}) = \prod_{i<j} \sum_{a_{ij}=0}^{1} e^{\theta a_{ij}} \\ &= \prod_{i<j} (1 + e^\theta) = (1 + e^\theta)^{\binom{n}{2}}. \end{aligned} \tag{2.2}$$

This is of course our expression for Z_θ. Next we simplify the expectation constraint (1.10):

$$\begin{aligned} \langle m \rangle &= \frac{\partial}{\partial \theta} (\ln Z_\theta) = \frac{\partial}{\partial \theta} \left[\binom{n}{2} \ln(1 + e^\theta) \right] \\ &= \binom{n}{2} \left(\frac{1}{e^{-\theta} + 1} \right), \end{aligned} \tag{2.3}$$

and this is our expression for $\langle m \rangle$. It may be solved uniquely for θ, thereby solving or initialising the entire model, since we can then immediately calculate Z_θ. We have:

$$\theta = -\ln \left[\frac{\binom{n}{2} - \langle m \rangle}{\langle m \rangle} \right]. \tag{2.4}$$

It is now easy to find the probability of a graph g in the ensemble. All we need to know is the number of edges $m(g)$. Observe that if $\theta < 0$ then the null graph has the biggest Hamiltonian and is therefore the most likely graph. Similarly if $\theta > 0$ the most likely graph is always the complete graph. If the estimated expectation $\langle m \rangle$ is half the maximum possible number of edges, that is, if $\langle m \rangle = \frac{1}{2}\binom{n}{2}$, then $\theta = 0$ and all graphs are equally likely. So the model does not give very sophisticated predictions. But this is to be expected - our assumptions were no more sophisticated.

Another point of interest concerns our observed graphs. If we have one observed graph g_{obs}, and if $m(g_{\text{obs}}) \neq \frac{1}{2}\binom{n}{2}$, g_{obs} cannot be the most likely graph unless it is either the null graph or the complete graph. An experienced statistician may not be surprised by such a fact but it struck me as sufficiently counter-intuitive to be worth drawing attention to. More generally, an ERG model based entirely on a number of observed configurations of a real-world network, and designed to mimic that network, does not necessarily make *any* of the observed configurations the most likely.

Conventionally the parameter θ is re-expressed in terms of $p = \frac{1}{e^{-\theta}+1}$, giving $\langle m \rangle = \binom{n}{2} p$ by (2.3). Moreover we may write the probability of a graph $g \in G$ as

$$P_\theta(g) = \frac{e^{\theta m(g)}}{Z_\theta} = \frac{e^{\theta m(g)}}{(1 + e^\theta)^{\binom{n}{2}}} = p^{m(g)} (1 - p)^{\binom{n}{2} - m(g)}. \tag{2.5}$$

Hence we may interpret $P(g)$ as the probability for a graph in which each of the $\binom{n}{2}$ possible edges appears with independent probability p.

The Bernoulli model has been widely studied since its introduction in 1951 [84] and has the advantage of being exactly soluble for many of its average properties [29, 30, 31]. More recent papers have focused on its inadequacies as a model of real-world networks, particularly in relation to clustering and its degree distribution [86, 3]. Real-world networks typically show strong clustering [90, 89] and power-law degree distributions [4, 32, 17], whereas the Bernoulli model shows weak clustering and a Poissonian degree distribution. Attempts have been made to adjust and extend the model to incorporate such real-world features [70, 71]. The focus of this project, the ERG model, is one such generalisation.

2.3 Some graphical examples

As a complement to the theory discussed so far, we present some illustrations in this section. Firstly, we consider a modified version of the Bernoulli model. If the graphs in the ensemble of the Bernoulli model are on n nodes, then there are $2^{\binom{n}{2}}$ of them, which is a prohibitively large number, even for comparatively small n. Now the Bernoulli model has only one graph observable, the total number of edges. So, instead of *all* undirected graphs on n nodes, a less redundant ensemble would be all *non-isomorphic* undirected graphs on n nodes. This ensemble is much smaller. (It has approximately $\frac{2^{\binom{n}{2}}}{n!}$ graphs for large n [81]). It is also an ensemble which I had ready access to, as output from a python program by my supervisor, Keith Briggs. The ensemble (on 4 nodes) is shown in figure 2.1. I wrote a python program (see appendix B) to use this ensemble (on 4 nodes) to calculate, directly from equations (1.6) to (1.8), the probabilities of each of the graphs when the observable was the number of edges. This enabled me to plot the whole distribution for a range of values of the parameter θ (see figure 2.2). I repeated this when the observable was the number of twostars, plotting the whole distribution for a range of values of its parameter, which I denote by α to avoid confusion with the θ of the previous sentence (see figure 2.3). I repeated this again for the model with *both* edges *and* twostars as observables (with parameters θ and α respectively). I set $\theta = 0.1$ and plotted the distribution for a range of values of α to produce figure 2.4.

Notice in figure 2.2 that for $\theta < 0$ a graph becomes more likely the fewer edges it has, just as we saw in the traditional Bernoulli model (section 2.2). The reason is the same as it was there - for $\theta < 0$ a graph with fewer edges will have a larger Hamiltonian, hence a larger probability. Similarly, for $\theta > 0$, we can understand why the graphs in figure 2.2 become more likely as the number of edges in the graph *increases*. Exactly the same kind of arguments allow us to understand the range of behaviour shown in figure 2.3, where the observable is the number of twostars.

Figure 2.4 is more interesting because it shows the interaction of these two observables. Nevertheless it is easy to make sense of the pictured distributions by similar reasoning. The value of θ is fixed at 0.1, which is positive, so that

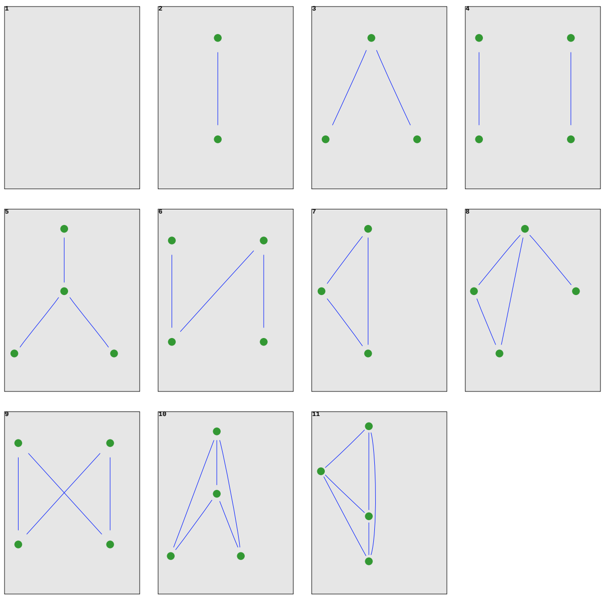

Figure 2.1: All non-isomorphic undirected graphs on 4 nodes (isolated nodes not shown, hence null graph produces an empty box). The number in the top left corner of each box is the graph number on the horizontal scales in the bar charts of figures 2.2 to 2.4.

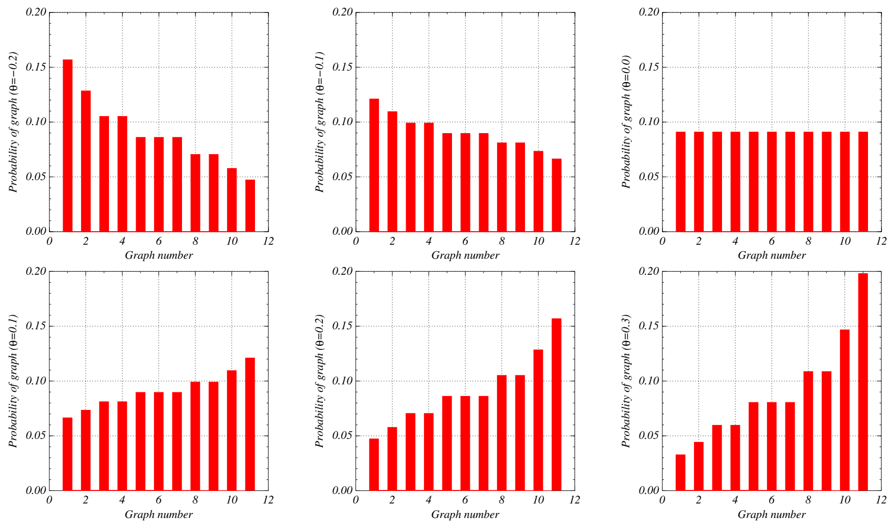

Figure 2.2: Observable: edges. Pictures are for a range of values of the parameter θ. The graph number is explained in figure 2.1.

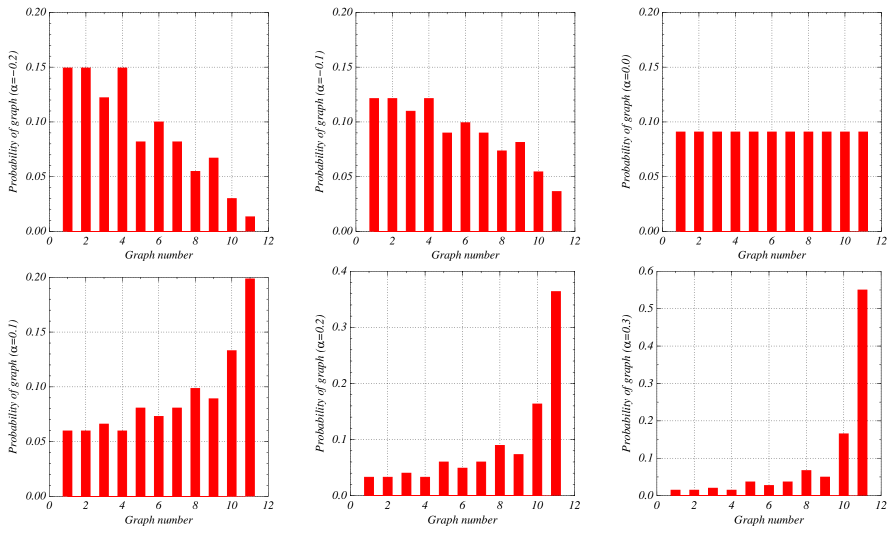

Figure 2.3: Observable: twostars. Pictures are for a range of values of the parameter α. The graph number is explained in figure 2.1.

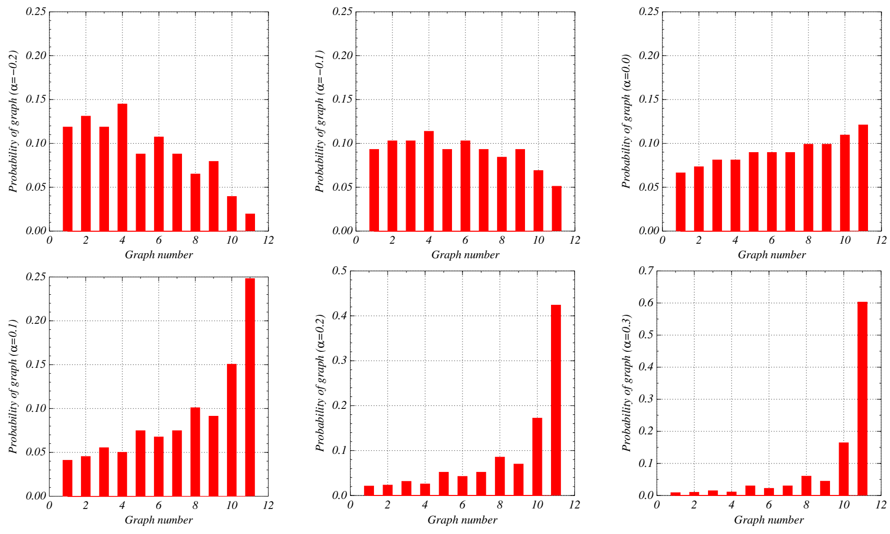

Figure 2.4: Observables: edges and twostars. Pictures are for a range of values of the two-star parameter α with the edge parameter fixed at $\theta = 0.1$. The graph number is explained in figure 2.1.

graphs with more edges will be, if you like, rewarded, that is, made more likely. But if $\alpha < 0$, a graph with *fewer* twostars will also be rewarded. So graphs with more edges but at the same time fewer twostars will be the most rewarded. By these ideas, and by referring to the key of graphs (figure 2.1) the reader should find figure 2.4 to be plausible.

Whether any of the models in this section are exactly soluble is unknown. I suspect that they are not, because the ensemble is not easy to manipulate algebraically. However, there do exist other ERG models that *are* soluble, and we turn our attention to these.

2.4 Further examples

Newman and Park [72] discuss other examples of exactly or partially soluble models. The ensemble of the Bernoulli model may be extended to allow multiple edge graphs, or it may be changed to directed graphs, and the solubility is not lost. To use these other ensembles we would, of course, have to be dealing with a real-world network for which the ensemble was appropriate as possible states or configurations of the network. They also consider modelling the degree sequence. To be specific they take as observables the degree of every node in the graph, and, as an ensemble, all undirected graphs on n nodes with no self loops or multiple edges. Then, if $k_i(g)$ is the degree of node i in a graph g, the degree sequence of g is the set with elements $k_i(g)$ for $i = 1$ to n, and the Hamiltonian may be written

$$H_\theta(g) = \sum_{i=1}^{n} \theta_i k_i(g). \tag{2.6}$$

Thus there is one parameter θ_i for every node i.

Now for an undirected graph with adjacency matrix a, $k_i(g) = \sum_j a_{ij}$, where the sum is over all nodes j. This allows the Hamiltonian to be re-expressed:

$$H_\theta(g) = \sum_{ij} \theta_i a_{ij} = \sum_{i<j} (\theta_i + \theta_j) a_{ij}. \tag{2.7}$$

Hence the partition function is

$$\begin{aligned} Z_\theta &= \sum_{ij} \exp\left(\sum_{i<j} (\theta_i + \theta_j) a_{ij} \right) \\ &= \prod_{i<j} \sum_{a_{ij}=0}^{1} e^{(\theta_i+\theta_j)a_{ij}} = \prod_{i<j} (1 + e^{(\theta_i+\theta_j)}) \end{aligned} \tag{2.8}$$

and the expectation constraints (1.10) become

$$\begin{aligned}\langle k_u\rangle &= \frac{\partial}{\partial\theta_u}(\ln Z_\theta) = \frac{\partial}{\partial\theta_u}\left(\sum_{i<j}(1+e^{(\theta_i+\theta_j)})\right)\\ &= \sum_{i\neq u} e^{(\theta_i+\theta_u)}\end{aligned} \tag{2.9}$$

for $u = 1, 2, \ldots, n$.

Newman and Park [72] make no comment on the solution of the constraint equations (2.9) but we observe here that they consist of n nonlinear equations and that it is unlikely in general that an analytic solution exists. The equations may be solved numerically (results in chapter 4 tell us there *is* a solution) but for large n even this may not be practical. It is a good idea, then, to cut down the number of observables and to make this number independent of n. We could, for example, take as observables the number of nodes of degree zero, one, two, and so on, up to nine, say. With these ten observables we would still gain insight into the probable degree distribution of the real-world network, whilst simultaneously reducing the amount of computation drastically if the network is large.

Perhaps half a dozen other ERG models have been solved in the past few years, either exactly or exactly in some limit on the size of the graphs. Not many of these have names yet. A sensible policy would be to name them, as concisely as possible, after their ensemble and observables. It would take too long to describe them all but a few details and some references can be found in table 2.1, where I have introduced my own provisional naming scheme.

Model	Ensemble	Observable(s)	Partition function	Reference
Bernoulli	Undirected graphs on n nodes	Number of edges	$(1+e^{\theta})^{\binom{n}{2}}$	[72]
Directed Bernoulli	Directed graphs on n nodes	Number of edges	$(1+e^{\theta})^{2\binom{n}{2}}$	[72]
Reciprocity	Directed graphs on n nodes	Number of edges, reciprocity	$\left[\frac{1+(e^{\alpha}-1)p^2}{(1-p)^2}\right]^{\binom{n}{2}}$, where $p=\frac{1}{e^{-\theta}+1}$, and α and θ are the reciprocity and edge parameters respectively	[72, 46]
Undirected twostar	Undirected graphs on n nodes	Number of edges, number of twostars	: $\left[\frac{1}{1-p}\right]^{\binom{n}{2}}\{1+\alpha n^3p^2+\mathrm{O}(\alpha^2)\}$, where this is approximate (and improves as $n\rightarrow\infty$), $p=\frac{1}{e^{-\theta}+1}$, and α and θ are the two:star and edge parameters respectively	[72, 75]
Terry 1	Non-isomorphic undirected graphs on n nodes, containing one non-intersecting walk	Length of walk	$\frac{1-e^{\theta n}}{1-e^{\theta}}$	This work, section 3.2.
Terry 2	Same	Absolute difference of length of walk and estimated expected value for this from a real-world network	$\frac{e^{\theta}-1-e^{\theta(n-m)}}{1-e^{\theta}}$, where m depends on the estimated expected value of the walk length	This work, section 3.3.

Table 2.1: Some exactly and approximately soluble models

Chapter 3

Two new models

"In science one tries to tell people, in such a way as to be understood by everyone, something that no one ever knew before. But in poetry it's the exact opposite."
Paul Dirac

3.1 Grappling

As I grappled with the sophisticated conceptual framework for ERG models, it occurred to me that any suitably simple pattern in the values of the observables might give rise to an exactly soluble model. I was led to consider a single graph observable x taking values in arithmetic progression over the ensemble G. To my excitement a soluble model ensued as well as ideas about how to interpret it and greater understanding of the power of the ERG machinery. Another model followed. I outline my results in this chapter.

3.2 The Terry 1 model

Suppose we have one observable x and an ensemble G of n graphs over which x takes values in arithmetic progression:

$$a, a+d, a+2d, \ldots, a+(n-1)d, \tag{3.1}$$

for constants a and d. Then the partition function (1.8) becomes

$$\begin{aligned} Z_\theta &= \sum_{g \in G} e^{\theta x(g)} = \sum_{i=0}^{n-1} e^{\theta(a+id)} \\ &= e^{\theta a} \sum_{i=0}^{n-1} (e^{\theta d})^i = e^{\theta a} \frac{1-e^{\theta n d}}{1-e^{\theta d}} \end{aligned} \tag{3.2}$$

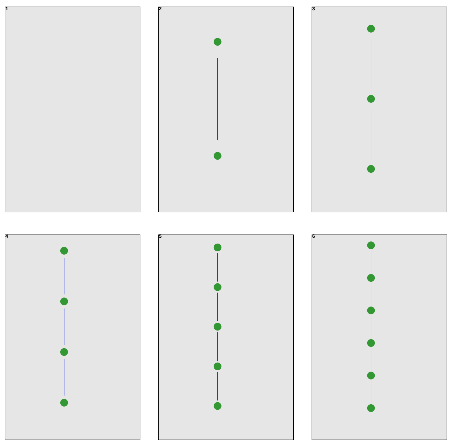

Figure 3.1: The ensemble of the Terry 1 model on 6 nodes (isolated nodes not shown, hence null graph produces an empty box)

where, on the last step, we have used the formula for the partial sum of a geometric progression. The expectation constraint (1.10) is

$$\begin{aligned}\langle x\rangle &= \frac{\partial}{\partial\theta}(\ln Z_\theta) = \frac{\partial}{\partial\theta}\left[\theta a + \ln(1-e^{\theta nd}) - \ln(1-e^{\theta d})\right] \\ &= a + \frac{de^{\theta d}}{1-e^{\theta d}} - \frac{nde^{\theta nd}}{1-e^{\theta nd}},\end{aligned} \tag{3.3}$$

which may be solved numerically for θ to initiate the model (results in chapter 4 ensure that a solution exists).

Definition 3.1. *The generalised Terry 1 model is given by equations (3.1) to (3.3).*

What kind of ensemble could give rise to an arithmetic progression in a graph observable, and what precisely would the observable measure? One possibility for the ensemble is the set of all non-isomorphic undirected graphs on n nodes possessing a single non-intersecting walk of any length and otherwise isolated nodes. The observable would then be the length of the walk and would take the values $0, 1, 2, \dots, (n-1)$. An example of this possible ensemble, for $n = 6$ nodes, is shown in figure 3.1.

Definition 3.2. *The Terry 1 model on n nodes is such that:*

- *The ensemble is all non-isomorphic undirected graphs on n nodes possessing a single non-intersecting walk of any length and otherwise isolated nodes*
- *The observable is the length of the walk and takes values $0, 1, 2, \dots, (n-1)$*
- *The defining equations are (3.1) to (3.3) where $a = 0$ and $d = 1$*

For arithmetic progressions taking values different to $0, 1, 2, \ldots, (n-1)$ there may be no simple ensemble-observable interpretation. This raises an important question. If we don't even know what the ensemble or observable are, how can we be sure that any exist that will give us the particular arithmetic progression we have in mind?

To answer this, we must think back to how we derived the ERG model in general (section 1.3). Suppose we were to change our assumption that the ensemble G was a set of graphs to a new assumption that G could be a set of *anything*. And suppose we define the set of observables to be those properties of the elements of G that we can measure. Then the derivation of the model remains valid, except that it is no longer necessarily a model of exponential random graphs but of just about anything, if I may be excused for sounding imprecise. ERG models are a special case of a much deeper strand of thought, but it wasn't until I used the ERG machinery to construct something that may not actually be an ERG model that I fully appreciated this.

As for real-world networks, or situations, that the Terry 1 model could represent, we could say this: for $a = 0$ (centimetres), $d = 1$ (centimetre), $a + (n-1)d = 500$ (centimetres), the length of the walk could give the length of a crawl by a certain snail on a particular morning, to the nearest centimetre, assuming any length of crawl from 0 to 500cm were possible. Unfortunately for $\theta < 0$, the most likely crawl predicted by the model will always be the shortest one, and for $\theta > 0$ it will be the longest one. (For $\theta = 0$ all crawls become equally likely.) Then we have the same problem as in the Bernoulli model - the extreme situations or graphs are always the most likely. It would be preferable if we could adjust the values of the observable to take into account the deviation of the possible situations from the expected situation $\langle x \rangle$. Such deliberations led me to construct a new model.

3.3 The Terry 2 model

Suppose we have a real-world snail, then, and suppose further that we have measured a few of its morning crawls and averaged them to find an estimate $\langle x \rangle$ for their expected value. We also know (somehow) that the possible crawl lengths are from a units to $a + (n-1)d$ units to the nearest multiple $d > 0$ of some unit measure of length. (So the possible crawl lengths are just the values in the sequence (3.1).) It must be that $a \leqslant \langle x \rangle \leqslant a + (n-1)d$ and for convenience we round $\langle x \rangle$ to the nearest crawl length in the set of possible crawl lengths just mentioned. So $\langle x \rangle$ is rounded to $a + md$, say, for some m satisfying $0 \leqslant m \leqslant n-1$. Now consider the new sequence or observable

$$|a - \langle x \rangle|, |a + d - \langle x \rangle|, \ldots |a + (n-1)d - \langle x \rangle|, \tag{3.4}$$

which may also be written

$$md, (m-1)d, (m-2)d, \ldots, d, 0, d, \ldots, (n-1-m)d. \tag{3.5}$$

The observable taking these values measures the distance of possible crawl lengths from the estimated average. The ensemble G will naturally be a set over which an observable may take these values, and one possible ensemble (where $a = 0$ and $d = 1$) is once again the ensemble of the Terry 1 model (definition 3.2). The partition function (1.8) is

$$\begin{aligned} Z_\theta &= \sum_{g\in G} e^{\theta x(g)} = \sum_{i=0}^{m} (e^{\theta d})^i + \sum_{i=1}^{n-1-m} (e^{\theta d})^i \\ &= \frac{e^{\theta d} + 1 - e^{\theta d(m+1)} - e^{\theta d(n-m)}}{1 - e^{\theta d}}, \end{aligned} \tag{3.6}$$

and the expectation constraint (1.10) is

$$\begin{aligned} \langle x \rangle &= a + md = \frac{\partial}{\partial\theta}(\ln Z_\theta) \\ &= \frac{de^{\theta d} - d(m+1)e^{\theta d(m+1)} - d(n-m)e^{\theta d(n-m)}}{e^{\theta d} + 1 - e^{\theta d(m+1)} - e^{\theta d(n-m)}} + \frac{e^{\theta d}}{1 - e^{\theta d}}. \end{aligned} \tag{3.7}$$

Definition 3.3. *The generalised Terry 2 model is given by equations (3.4) to (3.7).*

Definition 3.4. *The Terry 2 model on n nodes is such that:*

- *The ensemble is all non-isomorphic undirected graphs on n nodes possessing a single non-intersecting walk of any length and otherwise isolated nodes*
- *The length of a real-world 'walk' (where walks may be from length 0 to $n-1$) is estimated by averaging some observations. For convenience this average is rounded to the nearest unit, say m*
- *The observable on a graph g in the ensemble is the absolute difference of the length of the walk on g and the estimated average m; it takes values $m, m-1, \dots, 1, 0, 1, \dots, n-1-m$*
- *The defining equations are (3.4) to (3.7) where $a = 0$ and $d = 1$*

The distribution given by the generalised Terry 2 model will be 'nice' if equation (3.7) has a negative solution for θ (we discuss when this can exist in subsection 3.3.1). In other words, the probability that a crawl length differs from the estimated expected crawl $\langle x \rangle$ by a given amount, will grow smaller as this amount grows bigger. But we should be careful here. The estimated expected crawl length is unlikely to be a very accurate reflection of the *true* expected crawl length unless we average a high number of measured crawls. Only then can we have confidence in our model.

Suppose we have such confidence. Then a 'nice' distribution, combined with a knowledge of snail feeding habits, might be used by a biologist to gain some insight into more general behaviour, although the distribution is probably too simple for any such insight to be really significant.

Notice that whilst we derived the model by thinking about snails, it could be interpreted in other ways. For example, we might have an estimate for the expected height of a man in a given age range in a given country as well as estimates for the shortest and tallest heights in this group (and suppose we don't necessarily know that the heights are normally distributed). With this information, and a choice for d and n, we can look for a solution for $\theta < 0$ in equation (3.7) in order to initiate the model. Assuming the solution to (3.7) was negative, the model could be applied in some situations. If, for instance, the producer of a Snow White pantomime wanted an idea of how widely he would have to advertise to fill the seven dwarf parts, he could acquire this by recourse to the model. The model only gives probabilities of distances from the estimated expectation, so that in asking what proportion of people might be short enough for dwarfs, the producer would be asking for giants as well. So he could use the proportion he gets as an upper bound.

3.3.1 Negative solutions and 'nice' distributions

We now ask if a solution $\theta < 0$ may indeed exist for equation (3.7)? We shall restrict our attention to the Terry 2 model although I believe similar comments may be made about the generalised model. So, we let $a = 0$, $d = 1$, $n \in \mathbb{N}$, $m \in \mathbb{N} \cup \{0\}$, and $0 \leqslant m \leqslant n-1$. Further, let $y = e^{\theta d} = e^{\theta}$. Then we may rewrite the expectation constraint (3.7) as:

$$m = \frac{y - (m+1)y^{m+1} - (n-m)y^{n-m}}{y + 1 - y^{m+1} - y^{n-m}} + \frac{y}{1-y}. \tag{3.8}$$

We rearrange this and define a new function $f(y)$:

$$f(y) = \frac{y - (m+1)y^{m+1} - (n-m)y^{n-m}}{y + 1 - y^{m+1} - y^{n-m}} + \frac{y}{1-y} - m = 0. \tag{3.9}$$

Now if there is a solution y^* to $f(y) = 0$ on the interval $0 \leqslant y < 1$, then there will be a solution $\theta < 0$ since $\theta < 0 \Leftrightarrow 0 \leqslant y = e^{\theta} < 1$. In figure 3.2 we plot $f(y)$ against y on $y \in [0, 1)$ for $n = 1000$ and for a range of values of m. Clearly solutions can exist! However, even for m small compared to n, these solutions y^* are extremely close to 1 and they get closer to 1 as m increases. This causes two problems. First, it makes the solutions difficult to find. And second, the corresponding θ values, though of course negative, will get very close to zero. But if θ is close to zero then all Hamiltonians will be close to zero and the resulting probability distribution will be rather flat and arguably not very interesting. In figure 3.3 further pictorial examples demonstrating the possibility of negative θ solutions are shown next to the 'nice' distributions they give. Notice how, as m increases, the distribution does indeed become flatter.

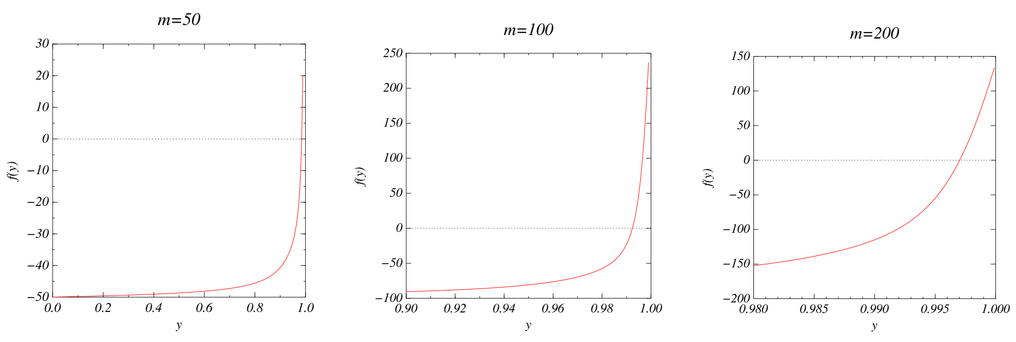

Figure 3.2: Terry 2 model: Demonstration that negative θ solutions may exist (corresponding to solutions for $y = e^\theta$ between 0 and 1). Here $n = 1000$.

The reader may be tempted to believe that the story ends there. (The author certainly was!). But it does not. The observant reader may also wonder why, in figure 3.3, the sequence of m values considered is 10, 20, 29. At first sight it looks pretty odd. For $m = 30$ I made many attempts to find the place where $f(y)$ crosses the horizontal axis, all in vain. For some time I was convinced that there was a vertical asymptote at $y = 1$ but eventually came to believe otherwise. Finally I managed to establish the following:

Theorem 3.1. *The number of solutions to the expectation constraint of the Terry 2 model for $\theta < 0$ depends on the value of m. In fact:*

A *If $m = 0$ then $\theta = -\infty$ is a solution*

B *If $1 \leqslant m < n-1$ and $n \geqslant 5$ there exists a real number $k_n \in \left(\frac{1}{n}, \frac{n-1}{n}\right)$, dependent on n, such that for all $m < k_n n$ there is at least one finite solution $\theta < 0$ to the expectation constraint. For large n, $k_n \approx \frac{2-\sqrt{2}}{2} = 0.293$ to 3 d.p.*

C *If $m = n-1$ then there are no solutions for $\theta < 0$*

The proof of this is somewhat long and is deferred to appendix A. Within the proof it is seen that there is *not* a vertical asymptote at $y = 1$. The limit $\lim_{y\to 1} f(y)$ exists and is finite. It is positive for $m < k_n n$ and negative for $m > k_n n$. Moreover, the function $f(y)$ appears to be strictly increasing on $(0, 1)$. Therefore:

Conjecture 3.1. *For $n \geqslant 5$, a solution $\theta < 0$ exists only for $m < k_n n$. When it does exist it is unique.*

Some comments on this conjecture may be found in appendix A. If it *is* true then it would explain why I was unable to find a solution for $m = 30$ when I was creating figure 3.3. Indeed, for $n = 100$ (as it does in figure 3.3), it turns out that

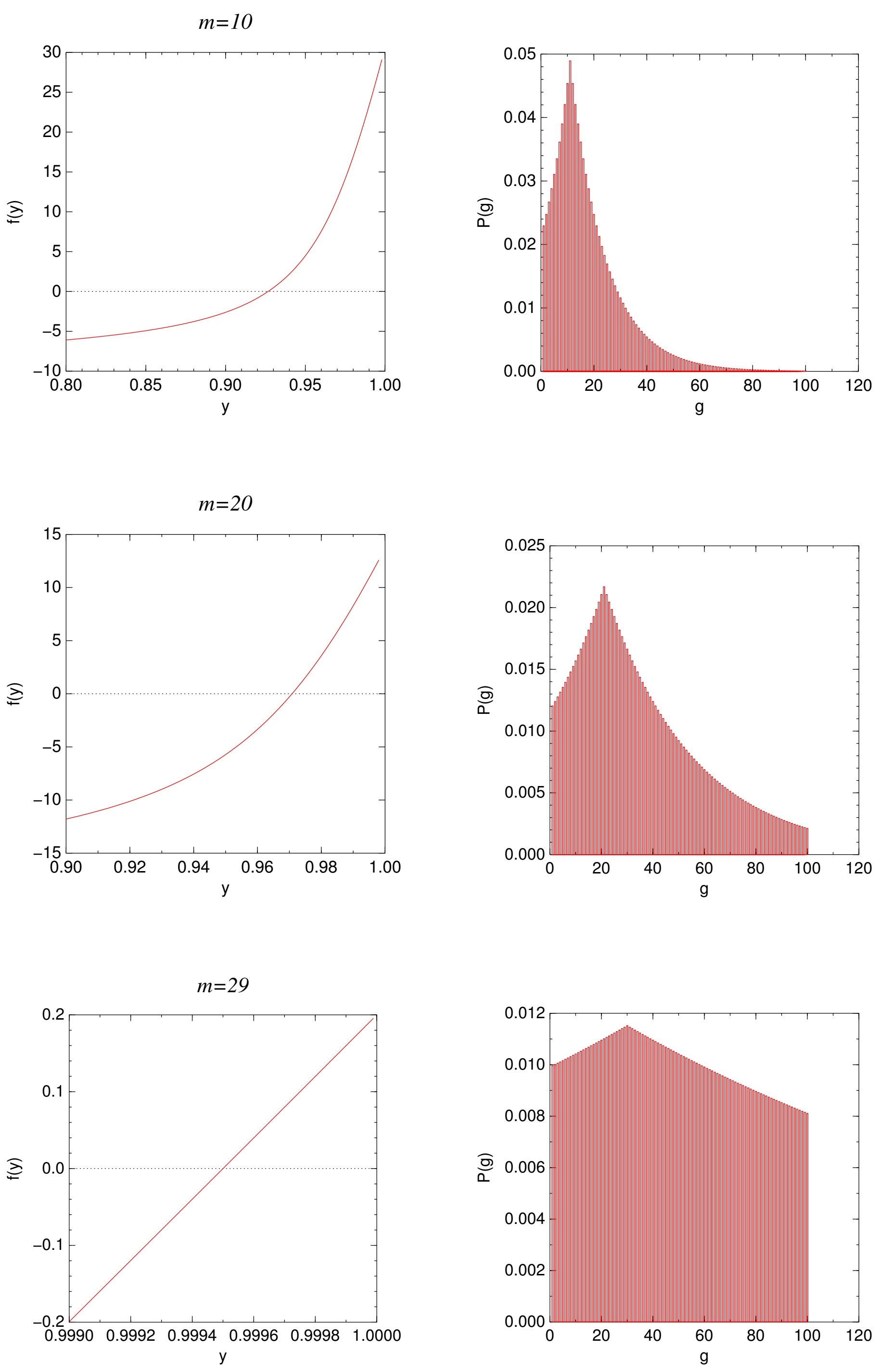

Figure 3.3: Terry 2 model: Negative θ solutions (corresponding to solutions for $y = e^{\theta}$ between 0 and 1) and the 'nice' distributions they give. Here $n = 100$. (The graphs on the horizontal axis are in the order in which they take the observable values as those values are ordered in equation (3.5) with $d = 1$.)

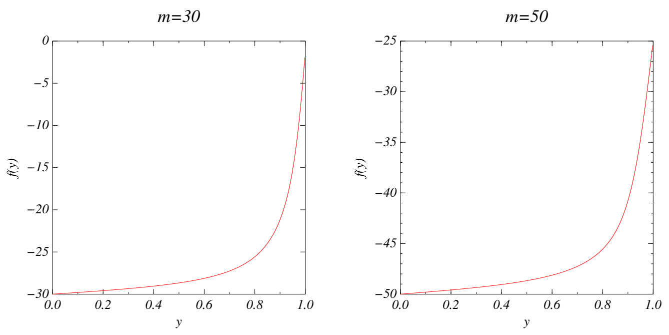

Figure 3.4: Terry 2 model: Demonstration that negative θ solutions may *not* exist (they would correspond to solutions for $y = e^{\theta}$ between 0 and 1). Here $n = 100$.

$k_n = 0.291$ to 3 d.p., so that for $m > k_n n \approx (0.291)(100) = 29.1$, solutions are not guaranteed by theorem 3.1. Graphical evidence that there are no solutions when $n = 100$ and $m = 30$ or $m = 50$ is given in figure 3.4.

3.4 Average properties

Like the Bernoulli model, the Terry 1 and Terry 2 models may be solved for many of their average properties. Some of these solutions are trivial because the ensemble is highly restricted but other results are more involved. For example, to find the expected number of twostars requires a clever trick, as we now demonstrate for the Terry 1 model. Let $t(g)$ denote the number of twostars of a graph g and recall that $x(g)$ is the length of the walk on g. Let g_i be the graph with a walk of length $i-1$. Then

$$\begin{aligned} \mathrm{E}_\theta[\text{twostars}] &= \sum_{g\in G} t(g)P_\theta(g) \\ &= \frac{1}{Z_\theta}\left\{t(g_1)e^{\theta x(g_1)} + t(g_2)e^{\theta x(g_2)} + \cdots + t(g_n)e^{\theta x(g_n)}\right\}. \end{aligned} \tag{3.10}$$

But for $i \geqslant 2$, the number of twostars in a graph g_i with a non-intersecting walk of length $x(g_i) = i-1$ (and otherwise isolated nodes) is $t(g_i) = x(g_i)-1 = i-2$, as is easily seen either by induction or a few simple pictures. If $i = 1$ then $t(g_1) = 0$, since g_1 has no edges. Letting $y = e^{\theta}$ we may therefore write

$$\mathrm{E}_\theta[\text{twostars}] = \frac{1}{Z_\theta}\left\{y^2 + 2y^3 + 3y^4 + \cdots + (n-2)y^{n-1}\right\}. \tag{3.11}$$

We can now use the clever trick. (Note in passing that more than one trick is possible. There is a trick that involves differentiation but we use another option here). Denote by S_n the series in y in this last equation (3.11). Multiply it by y to get yS_n and line up the two expressions as follows:

$$\begin{aligned} S_n &= y^2 + 2y^3 + 3y^4 + \cdots + (n-3)y^{n-2} + (n-2)y^{n-1} \\ yS_n &= \qquad\; y^3 + 2y^4 + \cdots + (n-4)y^{n-2} + (n-3)y^{n-1} + (n-2)y^n. \end{aligned}$$

Now subtract yS_n from S_n:

$$S_n - yS_n = y^2 + y^3 + y^4 + \cdots + y^{n-1} - (n-2)y^n. \tag{3.12}$$

The right hand side of this equation contains a geometric progression in y, so we can simplify it:

$$S_n(1-y) = \frac{y^2(1-y^{n-2})}{1-y} - (n-2)y^n. \tag{3.13}$$

Using the fact that $y = e^\theta$, we may then write

$$S_n = \frac{e^{2\theta}(1-e^{\theta(n-2)})}{(1-e^\theta)^2} - \frac{(n-2)e^{\theta n}}{1-e^\theta}, \tag{3.14}$$

and using the formula for the partition function Z_θ for the Terry 1 model (equation (3.2) with $a = 0$, $d = 1$), we finally have (from equation (3.11))

$$\mathrm{E}_\theta[\text{twostars}] = \left(\frac{1}{1-e^{\theta n}}\right)\left\{\frac{e^{2\theta}(1-e^{\theta(n-2)})}{1-e^\theta} - (n-2)e^{\theta n}\right\}. \tag{3.15}$$

The same ideas may be used to find $\mathrm{E}_\theta[\text{twostars}]$ for the Terry 2 model, although the end result is even more complex. Tables 3.1 and 3.2 give details of various average properties for the Terry 1 and Terry 2 models.

3.5 Generalisations and other patterns

We can extend the generalised Terry 1 model to have r observables where r is any natural number and each observable takes values in arithmetic progression (not necessarily the same arithmetic progressions of values for all observables). Suppose the progressions 'line up' over the ensemble, in the sense that for some graph g_1 all observables take their first value in their respective progressions, for some graph g_2 they all take their second value, and so on. Then the model is again exactly soluble because, in evaluating the partition function, we once again get a geometric progression.

Similarly we may extend the generalised Terry 2 model and the model remains soluble, though somewhat algebraically cumbersome. We have already seen (section 3.3) that an observable in the model could be height. Two observables could be called height and weight, and our producer of the Snow White pantomime could potentially use the model in a quest for plump dwarfs.

I considered other types of pattern for observables, namely geometric progressions and sequences of squares. Both of these led to prohibitive algebra and a headache.

length of walk	$\mathrm{E}_\theta[\text{length of walk}] = \frac{e^\theta}{1-e^\theta} - \frac{ne^{\theta n}}{1-e^{\theta n}}$
edges	$\mathrm{E}_\theta[\text{edges}] = \mathrm{E}_\theta[\text{length of walk}]$, since the number of edges on a graph *is* the length of the walk
twostars	$\mathrm{E}_\theta[\text{twostars}] = \left(\frac{1}{1-e^{\theta n}}\right)\left\{\frac{e^{2\theta}(1-e^{\theta(n-2)})}{1-e^\theta} - (n-2)e^{\theta n}\right\}$ (equation (3.15))
cycle of length k	$\mathrm{E}_\theta[\text{cycle of length} k] = 0$ for any $k \geqslant 3$, since no graph in the ensemble contains a cycle
clustering	$\mathrm{E}_\theta[\text{clustering coefficient}] = 0$, since two neighbours of a given node cannot be neighbours of each other for this ensemble
girth	$\mathrm{E}_\theta[\text{girth}] = 0$ since no graph in the ensemble contains a cycle

Table 3.1: Terry 1 model: Average properties

length of walk	$\mathrm{E}_\theta[\text{length of walk}] = \frac{e^\theta-(m+1)e^{\theta(m+1)}-(n-m)e^{\theta(n-m)}}{e^\theta+1-e^{\theta(m+1)}-e^{\theta(n-m)}} + \frac{e^\theta}{1-e^\theta}$ (equation (3.7) with $a = 0$, $d = 1$)
edges	$\mathrm{E}_\theta[\text{edges}] = \mathrm{E}_\theta[\text{length of walk}]$, since the number of edges on a graph *is* the length of the walk
twostars	$\mathrm{E}_\theta[\text{twostars}] = \frac{1}{KZ_\theta}[S_1 + S_2]$ where $K = \frac{1}{(1-e^\theta)^2}$ $Z_\theta = \frac{e^\theta+1-e^{\theta(m+1)}-e^{\theta(n-m)}}{1-e^\theta}$ $S_1 = \left[(m-2)e^\theta - e^{\theta(m-1)}\right](1-e^\theta) - e^{2\theta}(1-e^{\theta(m-3)})$ $S_2 = \left[(m-1) - (n-2)e^{\theta(n-m)}\right](1-e^\theta) - e^\theta(1-e^{\theta(n-m-1)})$
other observables	For cycles of length k (any $k \geqslant 3$), clustering, and girth, the average properties are the same as for the Terry 1 model and for the same reasons (see table 3.1).

Table 3.2: Terry 2 model: Average properties

3.6 Discussion

Our motivation at the start of this chapter was to find new ways of constructing exactly soluble ERG models. We achieved this by considering an observable taking values in arithmetic progression and the resulting model was called the generalised Terry 1 model. We took a specific case of this and found an observable–ensemble interpretation to be the length of a walk on the set of all non-isomorphic undirected graphs on n nodes containing a non–intersecting walk and otherwise isolated nodes, and we called this the Terry 1 model. In finding these models we became aware that the ERG machinery may actually be used in more general contexts than network modelling.

Like the Bernoulli model, we noted that the generalised Terry 1 model always makes the extreme 'graphs' or situations the most likely. We saw this as a limitation and to overcome it we considered a new observable, namely the absolute difference of terms in an arithmetic progression with an estimated expected value for these terms. We called this the generalised Terry 2 model. A specific case of this gave us the Terry 2 model with the same ensemble as in the Terry 1 model. We discussed applications of the generalised Terry 2 model and conditions under which the Terry 2 model produces a 'nice' distribution, finding that the latter only appears to happen when the estimated expected value is towards the lower end of the possible values.

At no stage did we claim to have constructed a new model for networks, although for networks with a very restricted range of behaviour (containing only a non-intersecting walk) we have done this. Non-intersecting walks, also called self-avoiding walks, arise in the study of linear polymer collapse [44], which may suggest a real-world application of the models. A number of average properties for the generalised Terry 1 model and generalised Terry 2 model were found, indicating that there remains some scope for further exploration.

Approximate methods exist for solving ERG models, such as perturbation techniques for estimating the partition function [72], and there are ad hoc ways of exactly solving particular examples, but there are no general methods in the current literature for *constructing* exactly soluble models. I have introduced such a method here, namely the restriction of observable values to certain types of pattern. My constructions lead me to believe that a fruitful area of research would be the consideration of a wide variety of patterns, notwithstanding the somewhat complex algebra this may entail. At the same time it would be worth contemplating which kinds of real-world networks could give rise to such patterns in the observables. The ultimate goal would of course be to find under what conditions an ERG model is exactly soluble. I have outlined a place to start.

Chapter 4

The likelihood function

"An undefined problem has an infinite number of solutions."
Robert A. Humphrey

4.1 Solving the constraints

Earlier (section 1.3) we raised the question of how many solutions may exist for the constraint equations (1.10), and how to find them if there are any. We settle the matter in this chapter by recourse to a function known as the *likelihood*.

I was reminded of the likelihood concept by my supervisor at BT, Keith Briggs. He stated the definition and told me that the value of its argument that maximised it (the "maximum likelihood estimator") was often used in statistics to solve the kind of problem I was working on. After experimenting with the function computationally (next section), I decided that only rigorous proofs would satisfy me of anything. I was able to establish a strong theorem (theorem 4.2), which turns out to be valuable in computational solutions of ERGs. The results and proofs in this section were independently found and constructed by me. It was only after constructing them that I became aware that they yield a special case of a general result on likelihood for exponential families. See [58] or [82] for more details. Other useful references for likelihood include [78] and [28].

We begin, then, with some definitions.

Definition 4.1. *Suppose we have an ERG model with r graph observables. Denote them by $x_1, x_2, \ldots, x_r$. We define the vector of observables for a graph g to be $x(g) = (x_1(g), x_2(g), \ldots, x_r(g))$.*

Definition 4.2. *An ERG model with r graph observables will have r parameters. Denote the parameters by $\theta_1, \theta_2, \ldots, \theta_r$. We define the vector of parameters to be $\theta = (\theta_1, \theta_2, \ldots, \theta_r)$. The parameter space is the set $\{\theta\}$ where each of the components θ_i of a particular θ may be any finite real number or $\pm\infty$.*

Notice that we may now write the Hamiltonian (1.7) for a graph g as

$$H_\theta(g) = \sum_{i=1}^{r} \theta_i x_i(g) = \theta \cdot x(g), \tag{4.1}$$

where the $\cdot$ denotes the dot product.

Definition 4.3. *For a random variable Y, dependent on some parameter θ, the likelihood of θ, given a particular subset $(y_1, y_2, \ldots, y_k)$ of the sample space, may be defined as*

$$L(\theta) = L(\theta|y_1, y_2, \ldots, y_k) = \text{Prob}[y_1, y_2, \ldots, y_k|\theta],$$

where Prob[] denotes the probability operator.

The loglikelihood $l(\theta)$ is defined to be simply the natural logarithm of the likelihood, $\ln L(\theta)$. (Technically the base of the logarithm is unimportant but we shall use the natural logarithm for consistency.) If the events $y_1, y_2, \ldots, y_k$ are independent then we may write

$$L(\theta) = \text{Prob}(y_1|\theta)\text{Prob}(y_2|\theta)\ldots\text{Prob}(y_r|\theta) \tag{4.2}$$

and

$$l(\theta) = \ln \text{Prob}(y_1|\theta) + \ln \text{Prob}(y_2|\theta) + \cdots + \ln \text{Prob}(y_r|\theta). \tag{4.3}$$

It may be trivially seen that the likelihood is bounded below by 0 and above by 1, being a probability, and that the loglikelihood is therefore not bounded below and bounded above by 0. It is also trivially seen that the likelihood and loglikelihood will be globally maximised at the same value(s) of θ, since the natural logarithm function $\ln$ is monotonic increasing if its argument is.

Definition 4.4. *We define the maximum likelihood estimator(s) to be the value(s) of θ that globally maximise the likelihood and loglikelihood.*

Henceforth we suppose that the random variable Y is our familiar ensemble G and that the probability operator Prob[] refers to an ERG distribution P, as defined in equations (1.6) to (1.8). Moreover we assume that the events y_1 to y_k are each observed configurations of a real-world network and, perhaps unrealistically, that these observations are independent. (An example will show why this is unrealistic. Observations of the internet on consecutive days will not be independent - what the internet does tomorrow very much depends on what it does today. But observations that are farther apart in time will probably be less dependent.) We make the assumption of independence for convenience. In many situations it may be difficult to know how else to begin the fitting of a parametric model.

Let us denote by $g_{\text{obs}}^{(1)}, g_{\text{obs}}^{(2)}, \ldots, g_{\text{obs}}^{(k)}$ the observed configurations of the real-world network. Then, using the probability formula (1.6), as well as (4.3) and (4.1), and recalling the definition of the free energy (section 1.3) as $F_\theta = \ln Z_\theta$, we may rewrite the loglikelihood $l(\theta)$ as follows

$$\begin{aligned} l(\theta) &= \ln\left\{\frac{e^{\theta\cdot x(g_{obs}^{(1)})}}{Z_\theta}\right\} + \cdots + \left\{\frac{e^{\theta\cdot x(g_{obs}^{(k)})}}{Z_\theta}\right\} \\ &= \theta\cdot\left[x(g_{\text{obs}}^{(1)}) + \cdots + x(g_{\text{obs}}^{(k)})\right] - kF_\theta. \end{aligned} \tag{4.4}$$

Now, from our observed graphs $g_{\text{obs}}^{(1)}, g_{\text{obs}}^{(2)}, \ldots, g_{\text{obs}}^{(k)}$, we may express the estimated expectation value $\langle x_i\rangle$ for the observable x_i as

$$\langle x_i\rangle = \frac{x(g_{\text{obs}}^{(1)}) + \cdots + x(g_{\text{obs}}^{(k)})}{k}. \tag{4.5}$$

Theorem 4.1. *For any i from 1 to r,*

$$\frac{\partial}{\partial\theta_i}(l(\theta)) = k\left[\langle x_i\rangle - \frac{\partial}{\partial\theta_i}(F_\theta)\right].$$

Proof. Using (4.4) and (4.5) we have

$$\begin{aligned} \frac{\partial}{\partial\theta_i}(l(\theta)) &= \frac{\partial}{\partial\theta_i}\left(\theta\cdot\left[x(g_{\text{obs}}^{(1)}) + \cdots + x(g_{\text{obs}}^{(k)})\right]\right) - k\frac{\partial}{\partial\theta_i}(F_\theta) \\ &= x_i(g_{\text{obs}}^{(1)}) + \cdots + x_i(g_{\text{obs}}^{(k)}) - k\frac{\partial}{\partial\theta_i}(F_\theta) \\ &= k\langle x_i\rangle - k\frac{\partial}{\partial\theta_i}(F_\theta) = k\left[\langle x_i\rangle - \frac{\partial}{\partial\theta_i}(F_\theta)\right]. \end{aligned}$$

□

Corollary 4.1. *The values of θ that make $l(\theta)$ stationary are precisely those θ that satisfy the expectation constraints* (1.10).

Proof. This amounts to showing that $\frac{\partial}{\partial\theta_i}(l(\theta)) = 0 \Leftrightarrow \langle x_i\rangle = \frac{\partial}{\partial\theta_i}(F_\theta)$. But this trivially follows from theorem 4.1.

□

Theorem 4.2. *For all θ in the parameter space and all i from 1 to r,*

$$\frac{\partial^2}{\partial\theta_i^2}(l(\theta)) \leqslant 0.$$

Proof. Theorem 4.1 tells us that $\frac{\partial}{\partial\theta_i}(l(\theta)) = k\left[\langle x_i\rangle - \frac{\partial}{\partial\theta_i}(F_\theta)\right]$. Therefore

$$\frac{\partial^2}{\partial\theta_i^2}(l(\theta)) = -k\frac{\partial^2}{\partial\theta_i^2}(F_\theta).$$

Now note that

$$F_\theta = \ln Z_\theta = \ln \sum_{g \in G} e^{\theta \cdot x(g)},$$

from which we find

$$\frac{\partial}{\partial \theta_i}(F_\theta) = \frac{\sum_{g\in G} x_i(g) e^{\theta\cdot x(g)}}{\sum_{g\in G} e^{\theta\cdot x(g)}} = \sum_{g\in G} x_i(g) P_\theta(g) = \mathrm{E}_\theta[x_i],$$

and hence

$$\begin{aligned}\frac{\partial^2}{\partial \theta_i^2}(F_\theta) &= \frac{\partial}{\partial \theta_i}\left[\frac{\partial}{\partial \theta_i}(F_\theta)\right] = \frac{\partial}{\partial \theta_i}\left[\frac{\sum_{g\in G} x_i(g) e^{\theta\cdot x(g)}}{\sum_{g\in G} e^{\theta\cdot x(g)}}\right] \\ &= \frac{\left(\sum_{g\in G} e^{\theta\cdot x(g)}\right)\left(\sum_{g\in G} (x_i(g))^2 e^{\theta\cdot x(g)}\right) - \left(\sum_{g\in G} x_i(g) e^{\theta\cdot x(g)}\right)^2}{\left(\sum_{g\in G} e^{\theta\cdot x(g)}\right)^2},\end{aligned}$$

where we have used the formula for differentiating a quotient. This simplifies:

$$\begin{aligned}\frac{\partial^2}{\partial \theta_i^2}(F_\theta) &= \frac{\sum_{g\in G} (x_i(g))^2 e^{\theta\cdot x(g)}}{\sum_{g\in G} e^{\theta\cdot x(g)}} - \left(\frac{\sum_{g\in G} x_i(g) e^{\theta\cdot x(g)}}{\sum_{g\in G} e^{\theta\cdot x(g)}}\right)^2 \\ &= \sum_{g\in G} (x_i(g))^2 P_\theta(g) - \left[\sum_{g\in G} x_i(g) P_\theta(g)\right]^2 \\ &= \mathrm{E}_\theta[{x_i}^2] - (\mathrm{E}_\theta[x_i])^2 \\ &= \sum_{g\in G} (x_i(g) - \mathrm{E}_\theta[x_i])^2 P_\theta(g) \\ &= \text{mean of squares} - \text{square of mean} \propto \text{variance}.\end{aligned}$$

Therefore

$$\frac{\partial^2}{\partial \theta_i^2}(l(\theta)) = -k \sum_{g\in G} (x_i(g) - \mathrm{E}_\theta[x_i])^2 P_\theta(g) \leqslant 0$$

for all θ, since $0 \leqslant P_\theta \leqslant 1$ for all θ, $(x_i(g) - \mathrm{E}_\theta[x_i])^2 \geqslant 0$ for all θ, and $k \in \mathbb{N}$.

$\square$

Corollary 4.2. *There always exists a solution to the expectation constraints* (1.10) *and all solutions are precisely equal to the maximum likelihood estimator(s).*

Proof. By corollary 4.1 this amounts to showing that the loglikelihood is stationary in all dimensions θ_i at the maximum likelihood estimator(s) and nowhere else. (By a dimension θ_i we mean the dimension corresponding to the parameter θ_i. We mention this since it may be considered a minor abuse of notation.)

The loglikelihood is defined over all θ_i for $-\infty \leqslant \theta_i \leqslant \infty$ and is everywhere differentiable. As noted in the comment just before definition 4.4, the loglikelihood is bounded above. But in a dimension θ_i a function that is defined everywhere, is everywhere differentiable, and is bounded above must have zero partial derivative (with respect to θ_i) where it is biggest (which may be at $\pm\infty$). If it didn't, then to at least one side of where it was biggest, it would be bigger!

Then, where the loglikelihood $l(\theta)$ is biggest in any dimension θ_i, it must be stationary. In other words, at the maximum likelihood estimator(s) the loglikelihood is stationary. It remains for us to show that the loglikelihood is not stationary anywhere else.

We show that in a dimension θ_i all stationary points must be at the same height (the same $l(\theta)$ value), namely the height at the maximum likelihood estimator(s). So, suppose there is a stationary point in a dimension θ_i which is at a maximum likelihood estimator (the existence of such a point is guaranteed by the previous paragraph). Suppose further that there exists another stationary point. The two stationary points may either be at the same height (the same $l(\theta)$ value), in which case there is nothing to prove, or one is higher than the other (different $l(\theta)$ values). If one is higher than the other, then the curve, in going from the higher point to the lower, must change somewhere from being concave down to concave up, since it passes from one point with zero derivative to another. But the latter (concave up) is not allowed since it would imply $\frac{\partial^2}{\partial{\theta_i}^2}(l(\theta)) > 0$ somewhere between the two points, which would contradict theorem 4.2.

Hence any stationary point in any dimension must be at a maximum likelihood estimator for that dimension. This completes the proof.

□

Corollary 4.3. *If a graph observable x_i is constant there are infinitely many solutions to the expectation constraints* (1.10) *for the parameter θ_i.*

Proof. Let the graph observable x_i be constant. Then $x_i(g) - \mathrm{E}_\theta[x_i] = 0$ for any graph g and all θ. It follows from theorem 4.2 that $\frac{\partial^2}{\partial\theta_i^2}(l(\theta)) = 0$ for all values of θ_i. In other words, $l(\theta)$ is flat in dimension θ_i. But then all values of θ_i, from $-\infty$ to ∞, are maximum likelihood estimators in this dimension. By this and corollary 4.2 the result follows.

□

Corollary 4.4. *If a graph observable x_i is not constant and $P_\theta(g) \neq 0$ for any θ, then there is a unique solution for θ_i in the expectation constraints* (1.10).

Proof. Let the graph observable x_i be non-constant. Then for any θ there is a graph g such that $x_i(g) - \mathrm{E}_\theta[x_i] \neq 0$. Because of this, and since $P_\theta(g) \neq 0$ for any

θ, theorem 4.2 tells us that $\frac{\partial^2}{\partial\theta_i^2}(l(\theta)) < 0$ for all values of θ_i. But the loglikelihood is a function defined everywhere and which is everywhere differentiable (in any dimension θ_j, including θ_i), so if it has a second derivative that is everywhere negative in a dimension θ_i it must have a unique maximum, that is, $l(\theta)$ has a unique maximum likelihood estimator in dimension θ_i. By this and corollary 4.2 the result follows.

□

4.2 Graphical examples of likelihood

With the ensemble of figure 2.1 and taking as an observable the number of edges, I plotted the loglikelihood for each graph in the ensemble (figure 4.1, top). Notice that the maximum likelihood estimator (MLE) is unique in all cases. The observable is non-constant over the ensemble yet for $\theta = -\infty$ only the null graph has a non-zero probability, which suggests that the condition on the probabilities in corollary 4.4 may sometimes be relaxed. For $\theta = +\infty$ only the complete graph has a non-zero probability.

Using the same ensemble but now choosing two observables - edges and twostars - I plotted the likelihoods of the graphs (figure 4.1, bottom). (When I plotted the loglikelihoods the pictures were much 'flatter' and harder to interpret.) The curious amoeba objects of figure 4.1 (bottom) are stretched in the vertical direction, suggesting greater dependence on the parameter that represents the horizontal axis, which is the twostar parameter. An appropriate re-scaling of the axes would yield more rounded amoebae (for most of the graphs) and give more accurate estimations of the MLE, but even these pictures are sound evidence that the MLE is unique in most and perhaps all of the graphs in the ensemble.

Finally, with the ensemble of figure 4.2 and taking as observables the number of nodes of degree 1 and the number of nodes of degree 2, I once again plotted likelihoods to obtain figure 4.3. The amoeba objects now are nicely rounded and suggest, along with brief reflection of extreme cases such as the null graph and the complete graph, that the likelihood is well behaved and gives a unique MLE for each graph. In fact notice that the null graph, the complete graph, and a few others, give exactly the same likelihood plots, with the amoeba going off the top left corner. These graphs all have no nodes of degree 1 or 2, so that, from the probability formula (1.6), we can say they become most likely when the parameters coupled to the observables are both $-\infty$, at which 'point' they are all equally likely with probability $\frac{1}{5}$. It seems we may observe then that the MLE can be unique for each graph where different graphs can have the same MLE.

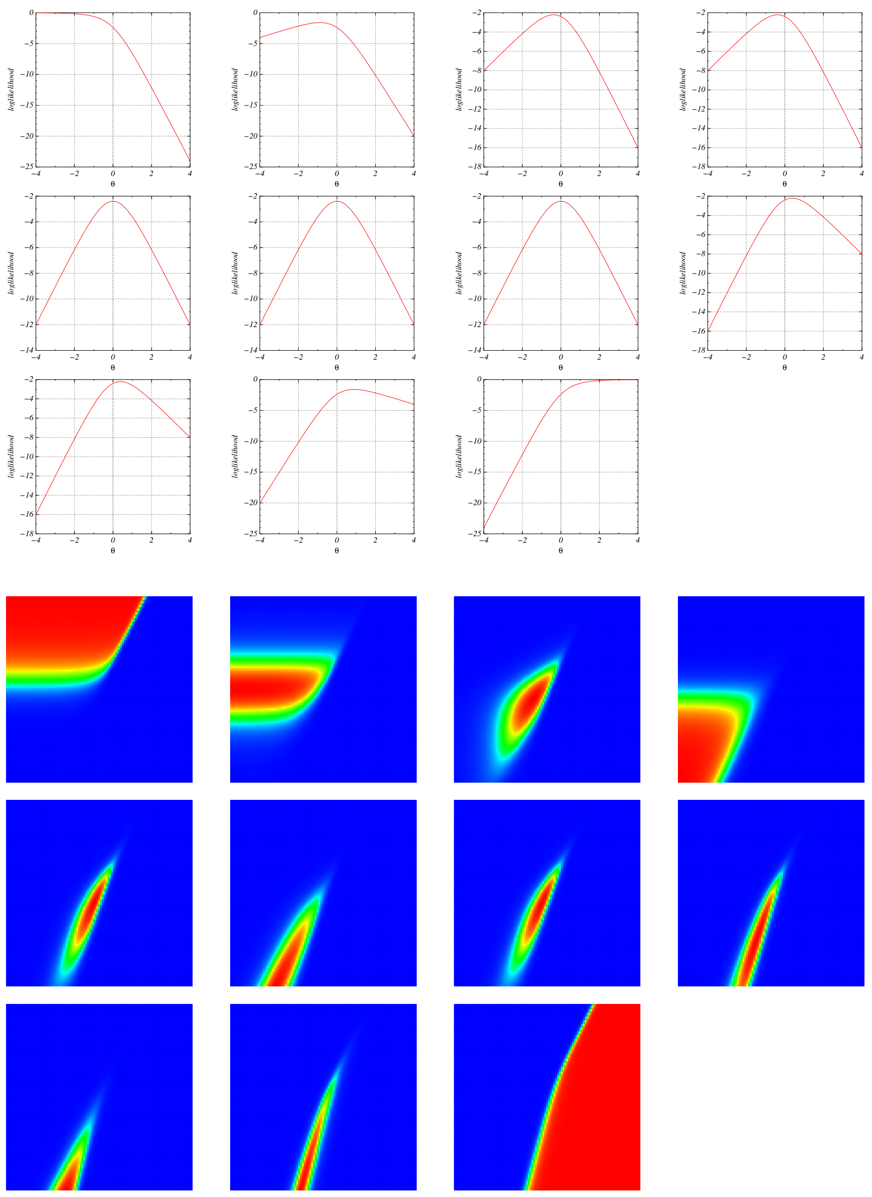

Figure 4.1: **Top**: Plots of loglikelihood. Observable – edges. Ensemble – the graphs in figure 2.1. Each picture here shows the loglikelihood plotted against its argument, θ. The graph used to make each picture is the graph in the corresponding position in figure 2.1. **Bottom**: Plots of likelihood. Observables – edges and twostars. Ensemble – the graphs in figure 2.1. Each picture here shows the likelihood plotted against its arguments, θ (the edge parameter) and α (the twostar parameter). Vertical axis is θ. Horizontal axis is from -5.0 to 5.0 left to right, vertical from -5.0 to 5.0 top to bottom. The graph used to make each picture is the graph in the corresponding position in figure 2.1. Red areas show the highest values; blue the lowest.

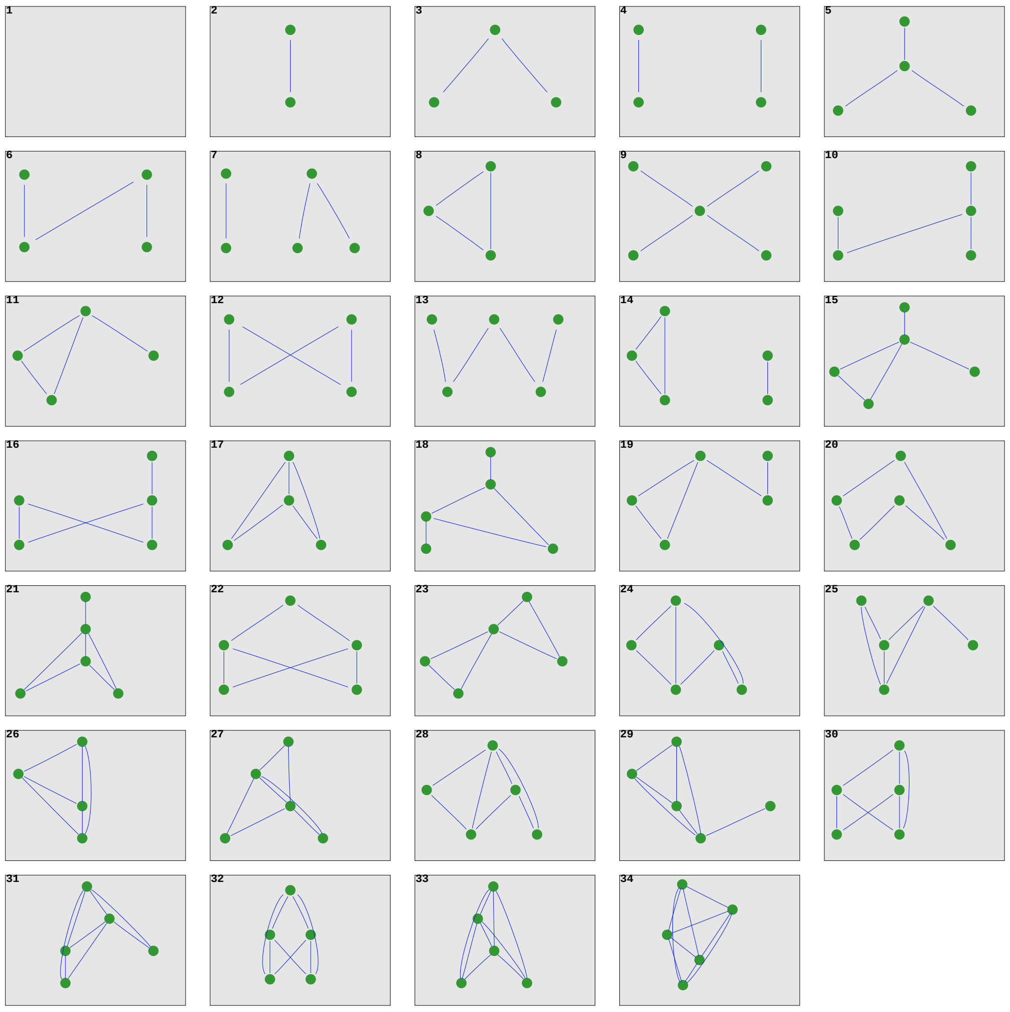

Figure 4.2: All non-isomorphic undirected graphs on 5 nodes (isolated nodes not shown, hence null graph produces an empty box).

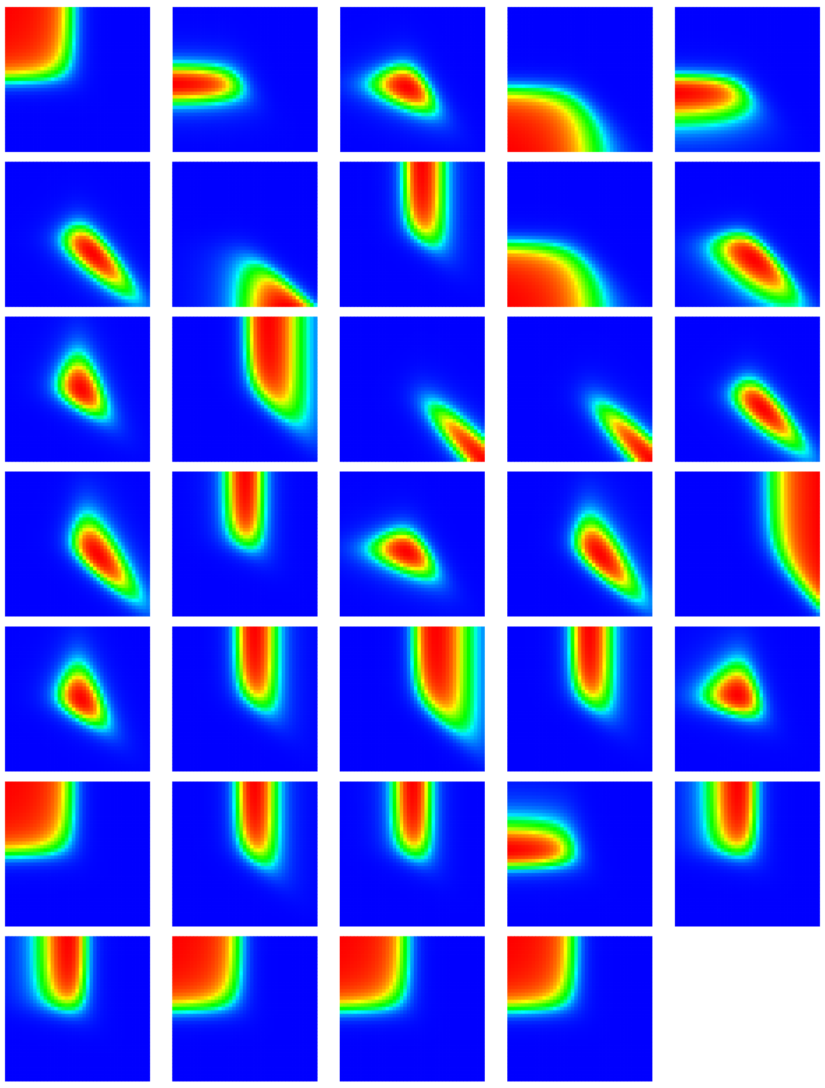

Figure 4.3: Plots of likelihood. Observables – nodes of degree 1 and nodes of degree 2. Ensemble – the graphs in figure 4.2. Each picture here shows the likelihood plotted against its arguments, θ (the degree 1 parameter) and α (the degree 2 parameter). Vertical axis is θ. Horizontal axis goes from -3.0 to 3.0 left to right, vertical from -3.0 to 3.0 top to bottom. The graph used to make each picture is the graph in the corresponding position in figure 4.2. Red areas show the highest values; blue the lowest.

Chapter 5

Markov Chain Monte Carlo simulations

"Pause you who read this, and think for a moment of the long chain of iron or gold, of thorns or flowers, that would never have bound you, but for the formation of the first link on one memorable day."

Charles Dickens

5.1 Introduction

Most ERG models do not have known exact solutions. Of course, to use an ERG model, it is always possible to work directly from its defining equations ((1.6) to (1.9)), but often this is not practical. If, for example, the ensemble is all undirected graphs on n nodes, then the number of terms that we need to sum to find the partition function (1.8) is $2^{\binom{n}{2}}$. If n is only 30, this sum will involve more than 10^{130} terms!

An alternative is required, and thankfully one exists. It is called the *Markov Chain Monte Carlo (MCMC) simulation* and will be the subject of the rest of this work.

5.2 The MCMC simulation

A formal description of Markov Chains would be complicated and is anyway unnecessary to give an understanding of MCMC simulations. Informally we may think of a (discrete) Markov Chain as being some sort of process where the outcome of the process X_{t+1} at a time $t + 1$ depends only on the outcome of the process X_t at the previous time t. Under certain conditions the chain will converge to an equilibrium, in the sense that the probabilities of the possible outcomes converge to a distribution known as the equilibrium distribution π. A careful treatment of Markov Chains may be found in [73].

Similarly we need but an intuitive approach to Monte Carlo simulations. A Monte Carlo simulation is simply an algorithm that uses randomly generated numbers. (Technically they are almost always pseudo-random numbers but we won't go into that here.) More details may be found in [9].

Informally, then, we may say:

Definition 5.1. *A* Markov Chain Monte Carlo (MCMC) simulation *is a process in which the outcome X_{t+1} at time $t+1$ depends only on the outcome X_t at time t and the comparison of a quantity dependent on X_t with a randomly generated number.*

5.3 Some background

We have seen the importance of the maximum likelihood estimator(s) (MLE) in the previous chapter. Aware of this importance, not merely to ERG models but to a host of other statistical problems, Geyer and Thompson [40] (1992) constructed Monte Carlo-based algorithms for approximating the MLE. In 1993 Dahmström and Dahmström [21] proposed an MCMC simulation for the estimation of a single parameter of a particular kind of ERG model, the Markov graph. (It would take too long to adequately describe Markov graphs here, so I shall just refer the reader to [35].) Then, in 1998, Corander, Dahmström, and Dahmström [20] extended this MCMC simulation to the estimation of a multi-dimensional parameter, using ideas from Geyer and Thompson's 1992 paper.

More recently Snijders [82] (2002) has drawn attention to convergence problems in MCMC simulations for ERG models. Handcock [45] (2003) and Snijders et al [83] (2004) have subsequently wrestled with the question of what makes a 'good model', but the question remains far from settled. We will speak of convergence problems again in subsection 5.4.5.

5.4 Theory

Recall that before we can use an ERG model we must initialise it by solving the expectation constraints (1.10) and by finding the partition function (1.8). In chapter 4 we saw that the MLE always yields the solutions to the expectation constraints. If we can therefore somehow approximate the loglikelihood function $l(\theta)$, it should be straightforward to roughly calculate the MLE. We shall describe how MCMC simulations may be used to make such an approximation, and may further be used to approximate the partition function.

5.4.1 Approximating the loglikelihood

We need a number of observations and results before we can arrive at our approximation for the loglikelihood $l(\theta)$. The ideas used in this and the next two subsections are adapted from [50].

To begin, then, suppose we have a real-world network, an appropriate ensemble G, and appropriate vector of observables x. Further, let θ be any vector of parameters in the parameter space, and let θ_0 be some fixed vector of parameters in this space. (Theoretically θ_0 can be any such vector, but we will see later (subsection 5.4.5) that for computational purposes it may be sensible to choose θ_0 close to the MLE, assuming we know how to do this.) Finally, let P_θ be the ERG model subject to these assumptions and let g denote any graph in G. We may write

$$\begin{aligned} E_{\theta_0}\left[e^{(\theta-\theta_0)\cdot x(g)}\right] &= \sum_{g\in G} e^{(\theta-\theta_0)\cdot x(g)} P_{\theta_0}(g) = \sum_{g\in G} e^{(\theta-\theta_0)\cdot x(g)} \frac{e^{\theta_0\cdot x(g)}}{Z_{\theta_0}} \\ &= \frac{1}{Z_{\theta_0}} \sum_{g\in G} e^{\theta\cdot x(g)} = \frac{Z_\theta}{Z_{\theta_0}}, \end{aligned} \tag{5.1}$$

where, on the last step, we have used equation (1.8).

Thus $\frac{Z_\theta}{Z_{\theta_0}}$ is an expectation with respect to θ_0. But we can use the law of large numbers to estimate an expectation by a sample mean. Let $g_1, g_2, \ldots, g_m$ be a random sample of graphs picked from the distribution P_{θ_0} defined by the ERG model with parameter θ_0. (Naturally a graph is more likely to be picked from this model if it has a higher probability *in* the model.) Then we may estimate

$$\frac{Z_\theta}{Z_{\theta_0}} = E_{\theta_0}\left[e^{(\theta-\theta_0)\cdot x(g)}\right]$$

by the sample mean

$$\frac{1}{m}\sum_{i=1}^{m} e^{(\theta-\theta_0)\cdot x(g_i)}. \tag{5.2}$$

Assume for the moment that we have such a sample of graphs $g_1, g_2, \ldots, g_m$. (We discuss how to get this in subsection 5.4.2.) Suppose in addition that we have an observation g_{obs} of our real-world network. Then, using the definition of the likelihood function L_θ (definition 4.3), we can express the loglikelihood $l(\theta)$ as

$$l(\theta) = \ln L(\theta) = \ln \text{Prob}[g_{\text{obs}}|\theta] = \ln P_\theta(g_{\text{obs}}) = \ln\left\{\frac{e^{\theta\cdot x(g_{\text{obs}})}}{Z_\theta}\right\}. \tag{5.3}$$

where Prob[] denotes the probability operator. We can express $l(\theta_0)$ similarly. It now follows that

$$\begin{aligned} l(\theta) - l(\theta_0) &= \ln\left\{\frac{e^{\theta\cdot x(g_{\text{obs}})}}{Z_\theta}\right\} - \ln\left\{\frac{e^{\theta_0\cdot x(g_{\text{obs}})}}{Z_{\theta_0}}\right\} \\ &= -\ln\left\{e^{(\theta_0-\theta)\cdot x(g_{\text{obs}})}\frac{Z_\theta}{Z_{\theta_0}}\right\}. \end{aligned} \tag{5.4}$$

But $\frac{Z_\theta}{Z_{\theta_0}}$ may be approximated by $\frac{1}{m}\sum_{i=1}^{m} e^{(\theta-\theta_0)\cdot x(g_i)}$ (equation (5.2)). Hence

$$\begin{aligned} l(\theta) - l(\theta_0) &\approx -\ln\left\{ e^{(\theta_0-\theta)\cdot x(g_{\text{obs}})} \frac{1}{m}\sum_{i=1}^{m} e^{(\theta-\theta_0)\cdot x(g_i)} \right\} \\ &= -\ln\left\{ \frac{1}{m}\sum_{i=1}^{m} e^{(\theta-\theta_0)\cdot[-x(g_{\text{obs}})]} e^{(\theta-\theta_0)\cdot x(g_i)} \right\} \\ &= -\ln\left\{ \frac{1}{m}\sum_{i=1}^{m} e^{(\theta-\theta_0)\cdot[x(g_i)-x(g_{\text{obs}})]} \right\}. \qquad (5.5) \end{aligned}$$

If we now maximise (5.5) as a function of θ, we will (approximately) maximise $l(\theta) - l(\theta_0)$. However, since θ_0 is fixed, $l(\theta_0)$ is fixed, and we therefore (approximately) maximise $l(\theta)$. But the θ value that does this is, of course, our MLE! So we have found what we want!

Two questions arise. First, how easy is it in practice to calculate (5.5)? Second, what if there is more than one observed graph? We deal with the first question in subsection 5.4.3 and answer the second here.

Suppose the observed graphs are $g_{\text{obs}}^{(1)}, g_{\text{obs}}^{(2)}, \ldots, g_{\text{obs}}^{(k)}$. The loglikelihood is, by definition (see definition 4.3 and subsequent discussion),

$$l(\theta) = \ln \text{Prob}\left[g_{\text{obs}}^{(1)}, \ldots, g_{\text{obs}}^{(k)} | \theta\right],$$

where Prob[] denotes the probability operator. If we assume all observed graphs are independent, and that the operator Prob[] refers to our ERG distribution P, then

$$\begin{aligned} l(\theta) &= \ln\left\{\text{Prob}[g_{\text{obs}}^{(1)}|\theta] \ldots \text{Prob}[g_{\text{obs}}^{(k)}|\theta]\right\} = \ln\left\{P_\theta(g_{\text{obs}}^{(1)}) \ldots P_\theta(g_{\text{obs}}^{(k)})\right\} \\ &= \ln P_\theta(g_{\text{obs}}^{(1)}) + \cdots + \ln P_\theta(g_{\text{obs}}^{(k)}). \qquad (5.6) \end{aligned}$$

But $\ln P_\theta(g_{\text{obs}}^{(1)})$ is the loglikelihood for the observed graph $g_{\text{obs}}^{(1)}$. So, in fact (5.6) is a sum of loglikelihoods, each of which may be estimated up to an additive constant and for a given θ by (5.5). Then, to find the MLE, we simply find the θ that maximises this sum of loglikelihoods. As noted after definition 4.4 it may be unreasonable to suppose the observed graphs are independent, but it can be difficult to do the maths without this assumption. The issue of multiple observed graphs for MCMC methods is worth exploring as I have not seen it appear anywhere in the literature.

5.4.2 Using a simulation to get a sample

How, then, do we find a random sample of graphs $g_1, g_2, \ldots, g_m$ from the distribution defined by the ERG model with parameter θ_0? Simple! We run an MCMC simulation! We begin with our observed graph g_{obs} (although theoretically we could begin with *any* graph in the ensemble) and, on each step of the simulation, change (or not) the graph that we currently have, according to some straightforward rule (examples of which will be given very shortly), where the rule is dependent on θ_0, and the chain will converge in theory to the desired distribution.

One such rule is known as the *Gibbs Sampler*, which we describe here for undirected graphs (the directed case is very similar). On each step of the simulation we randomly pick from the current graph g a pair of nodes (i, j) where $i \neq j$. If there is an edge (i, j) in g (or (j, i), which is the same thing since g is undirected), we let g_{ij}^{+} be the graph g and g_{ij}^{-} be the graph g missing the edge (i, j). Otherwise we let g_{ij}^{+} be the graph g plus the edge (i, j) and let g_{ij}^{-} be the graph g. We form the new graph g_{new} by first letting it be the same as g, and then, whether or not it already has the edge (i, j), we allow there to be the edge (i, j) with probability

$$\frac{e^{\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}}{1 + e^{\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}}. \tag{5.7}$$

In other words, if this quantity is bigger than a randomly chosen number between 0 and 1, we allow the edge (i, j) (and therefore (j, i) as well since the graph is undirected) in g_{new} and otherwise do not.

The observant reader may wonder if changing a graph in this way may lead to a graph outside the ensemble and consequently to an inaccurate chain. There is no theoretical problem if the ensemble is all directed or undirected graphs on a fixed number of nodes n (see [82]), but the Terry 1 and Terry 2 models (chapter 3) may produce strange results. There has not been the time to investigate and in any case those latter two models are exactly soluble.

Obviously the graph g_{new} may not actually be different to g. In any case we always use g_{new} at the start of the next step as our current graph. If we run a chain of, say, 30 million steps, we could take the last 10^5 graphs as a sample.

There are many alternatives to the Gibbs Sampler. A small list of updating steps is given in table 5.1.

5.4.3 Efficient storage

Crucially the only data we need to store as we run the simulation is the running total of the changes in the vector of graph observables. To help explain this, remember that the chain starts at g_{obs} and imagine that at the end of step t the graph becomes $g_{\text{chain}}^{(t)}$. Then on the first step the change in the vector of observables is $x(g_{\text{chain}}^{(1)}) - x(g_{\text{obs}})$. On the second step the change is $x(g_{\text{chain}}^{(2)}) - x(g_{\text{chain}}^{(1)})$, and the running total of the changes is

$$\left[x(g_{\text{chain}}^{(2)}) - x(g_{\text{chain}}^{(1)})\right] + \left[x(g_{\text{chain}}^{(1)}) - x(g_{\text{obs}})\right] = x(g_{\text{chain}}^{(2)}) - x(g_{\text{obs}}).$$

Gibbs Sampler	Let the current graph be g. Pick a random pair of nodes (i,j) where $i \neq j$. If (i,j) is an edge in g, set $g_{ij}^{+} = g$ and $g_{ij}^{-} = g$ minus edge (i,j). Otherwise set $g_{ij}^{+} = g$ plus edge (i,j) and $g_{ij}^{-} = g$. Form g_{new} to be identical to g and then, whether g_{new} has the edge (i,j) or not, let g_{new} have edge (i,j) with probability $\frac{e^{\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}}{1+e^{\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}}$.
Metropolis-Hastings	Same as Gibbs Sampler except that, instead of deciding whether or not to allow there to be an edge (i,j) in g_{new}, we decide whether or not to *add* the edge if it's not already there or *remove* it if it is. If it's *not* already there, add it with probability $\min(1, \pi_1)$ where $\pi_1 = e^{\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}$. If it *is* there, remove it with probability $\min(1, \pi_2)$ where $\pi_2 = e^{-\theta_0 \cdot \left(x(g_{ij}^{+}) - x(g_{ij}^{-})\right)}$.
Groups of node-pairs	A number of pairs of nodes may be considered in one updating step. See [82].
Inversion	The current graph is changed to its complement i.e. an edge is added where there isn't one and taken away where there is. This updating step is used to supplement a simulation that normally has a different kind of update on a particular step, since if we only ever inverted we would only ever flip between two graphs.

Table 5.1: Examples of updating steps in MCMC simulations

Likewise, after three steps, the running total is $x(g^{(3)}_{\text{chain}}) - x(g_{\text{obs}})$. It is now easy to see that after t steps the running total will be $x(g^{(t)}_{\text{chain}}) - x(g_{\text{obs}})$. But in (5.5), the only information we need from our sample of graphs is $x(g_i) - x(g_{\text{obs}})$ for each graph g_i in the sample. So just by keeping a running total of the changes in the vector of observables, we will be placing ourselves in a position to calculate (5.5).

Keeping such a running total is *very efficient computationally*. This is because, to find the *change* in the vector of observables, $x(g^{(t)}_{\text{chain}}) - x(g^{(t-1)}_{\text{chain}})$, on a particular step t, may not (depending on the choice of observables) require us to calculate the *individual* vectors of observables, $x(g^{(t)}_{\text{chain}})$ or $x(g^{(t-1)}_{\text{chain}})$. Moreover, on any step, we only need to store the current graph.

An example will make this clear. Let the vector of observables consist of only one observable, namely the total number of edges. Also, let the randomly chosen pair of nodes on step t be (i, j). And suppose we use the Gibbs Sampler. We may easily determine in a computer program if (i, j) is an edge in $g^{(t-1)}_{\text{chain}}$ by asking for the value of the element in row i and column j of its adjacency matrix. Suppose there is not an edge (i, j) in $g^{(t-1)}_{\text{chain}}$. Then if on step t it is decided that we do *not* allow there to be an edge (i, j) in $g^{(t)}_{\text{chain}}$, the change in the total number of edges is clearly zero, and if it is decided that we *do* allow there to be an edge (i, j) in $g^{(t)}_{\text{chain}}$, the change is clearly $+1$. We may reason in a similar way if $g^{(t-1)}_{\text{chain}}$ *does* have an edge (i, j). Once we know if we are to have the edge or not, we can change $g^{(t-1)}_{\text{chain}}$ into $g^{(t)}_{\text{chain}}$ quite trivially. Writing a program to keep track of the correct change and the current graph on each step, and the running total of the changes, is therefore not complicated (see the second program in appendix B).

5.4.4 The partition function

We now have a method for estimating the vector of parameters that satisfies equation (1.10) but to be able to properly use our ERG model we still need an estimate for the partition function Z_θ. Fortunately such an estimate may be found using ideas with which we are already familiar. Indeed recall by equation (5.2) that we may write

$$\frac{Z_\theta}{Z_{\theta_0}} \approx \frac{1}{m} \sum_{i=1}^{m} e^{(\theta - \theta_0) \cdot x(g_i)} \tag{5.8}$$

where $g_1, g_2, \ldots, g_m$ is a random sample of graphs drawn from the distribution defined by the ERG model with parameter θ_0. If we set $\theta_0 = 0$ we can use equation (1.8) to find that

$$Z_\theta = Z_{\theta=0} = \sum_{g \in G} e^0 = \sum_{g \in G} 1 = \text{number of graphs in ensemble.}$$

If the ensemble is all undirected graphs on n nodes, say, we get $Z_{\theta=0} = 2^{\binom{n}{2}}$. Then we can rearrange (5.8) to obtain

$$Z_{\theta_0} \approx \binom{n}{2} \left\{ \frac{1}{m} \sum_{i=1}^{m} e^{-\theta_0 \cdot x(g_i)} \right\}^{-1} \tag{5.9}$$

which may be written

$$Z_{\theta_0} \approx \binom{n}{2} e^{-\theta_0 \cdot x(g_{\text{obs}})} \left\{ \frac{1}{m} \sum_{i=1}^{m} e^{-\theta_0 \cdot [x(g_i) - x(g_{\text{obs}})]} \right\}^{-1} \tag{5.10}$$

where g_{obs} is an observed graph. We have already discussed (section 5.4.3) how it is possible to efficiently compute quantities such as the expression enclosed in parentheses in this last equation (5.10). Calculating the other factors in (5.10) presents no significant challenge. For large n it is straightforward to approximate $2^{\binom{n}{2}}$ and for a reasonably small number of observables, and as long as g_{obs} is not a really gigantic graph, there should be no problems in computing $x(g_{\text{obs}})$. Thus if we set θ_0 to be our MLE, found by the method outlined in the last few subsections, then equation (5.10) becomes an approximation for the partition function for the model we want to use.

I have not seen anyone else derive (5.10). Nor have I had time to experiment with it, so I recommend it as an area to be investigated.

5.4.5 Convergence

If it is possible to go from any graph to any other graph in a finite number of steps (i.e., assuming all graphs 'communicate'), then the only requirement for convergence of an MCMC simulation is the satisfaction of the *detailed balance equation* [73]. Let us denote by $P_t(g_a, g_b)$ the probability, or transition probability, of going from a graph g_a to a graph g_b on a time step t of the simulation. Then if there exists a distribution π over the ensemble such that, for all g_a and g_b in the ensemble,

$$\pi(g_a)P_t(g_a, g_b) = \pi(g_b)P_t(g_b, g_a) \tag{5.11}$$

then P and π are said to be in detailed balance and π is a stationary distribution of the Markov Chain with transition probabilities $P_t(g_a, g_b)$. It may be shown fairly easily [82] that the updating steps in table 5.1 satisfy the detailed balance equation, so these steps are the transition probabilities of a convergent chain (assuming all graphs 'communicate').

Of course knowing only that a chain is convergent tells us nothing about how quickly it converges. In practice MCMC simulations may take a hideously long time to converge. As an example, suppose the ensemble is all directed graphs on n nodes and that the observables are the total number of edges, the reciprocity, and the number of out-twostars (defined in table 1.3). Snijders [82] has found that for certain ranges of the vector of parameters the equilibrium distribution is bimodal, in the sense that most of the probability mass is distributed over two clearly separated subsets of the ensemble, one containing low-density graphs (few edges) and the other high-density graphs (many edges). The separation between these two subsets can be so extreme that steps using the Gibbs Sampler, or other updating steps that change only a small number of edges, have a negligible probability of taking the chain from one subset to the other. For such models simulation results are, practically speaking, determined by the initial graph, and they give completely misleading information about expectation values for the ensemble.

According to Hunter [50], if the vector of parameters θ_0 for which we run a simulation is not close to MLE then convergence may be "agonisingly slow". But Snijders [82] points out that in many cases of interest the MLE is in a range that gives the kind of hopeless situation mentioned in the last paragraph. Hunter also claims that a choice of θ_0 that "often works" is the maximum pseudolikelihood (MPLE) (see [50] for a description of it). But he admits that little is known of its theoretical value. Snijders suggests that the MPLE no longer be used until substantial theory can justify its choice. As far as I can see, any θ_0 that gives a quick convergence is worth considering, if we can find it.

To find a good choice for θ_0, then, my approach has been as follows. Always beginning my chains with the observed graph g_{obs}, I print out, for a range of θ_0, the running totals of the changes in the vector of observables for a small sample of graphs, after, say, chains of 10, 20, and 30 million steps. When I find a θ_0 that leads to small values in the running totals, I assume that this is a good choice because few changes appear to be needed to go from the observed graph to the graphs that occur many millions of steps later. Then I investigate values close to θ_0. Eventually, if I have a θ_0 that consistently gives very small values in the running totals of the changes in the vector of observables, I turn my attention to estimating the MLE and may make further adjustments to θ_0 depending on what happens there. Snijders has written some programs that automatically adjust θ_0 whilst estimating the MLE [82] but by his own admission they give mixed results.

5.5 A real-world application

We consider a data set that has been well studied in the literature, the Sampson monastery data. A PhD student, Sampson spent some time in a monastery observing the interactions of 18 monks. He gathered friendship data and analysed it [79], and various authors have subsequently manipulated and dissected it [46, 33, 35, 82]. We shall look at the symmetrised version used by Frank and Strauss [35]. The original data was converted into a symmetric adjacency matrix with an edge between two nodes (or monks, which the nodes represent) indicating, essentially, that at least one of the monks considers the other to be a friend (see table 5.2). We have here, then, an undirected social network and we take as our ensemble all undirected graphs on 18 nodes.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	1	1		1										1			
2		0	1	1	1	1	1		1	1					1			
3			0		1	1	1	1	1	1								
4				0	1	1	1			1								
5					0	1	1											
6						0	1				1			1	1			
7							0		1	1								
8								0	1	1	1	1	1	1				
9									0	1	1	1	1			1	1	1
10										0	1	1	1	1			1	1
11											0	1		1				
12												0	1	1				
13													0	1	1			
14														0				
15															0	1	1	1
16																0	1	1
17																	0	1
18																		0

Table 5.2: The symmetrised version of the adjacency matrix of Sampson's 1969 data used in [82, 35]. In the upper triangle blank spaces correspond to zeroes.

I began my exploration of the data by choosing to have one observable, the number of edges. I wrote a C program (see appendix B) to carry out an MCMC simulation by Metropolis-Hastings updating steps, with the possibility of inversion on every hundredth step. Very rapid convergence was seen for various θ_0. In fact the graphs in the chain appeared to settle into equilibrium immediately, with the number of edges fluctuating about their expected value (see figure 5.1, top left and right). (The expected value is known exactly here since we have the ensemble-observable combination of the Bernoulli model on 18 nodes.) Choosing $\theta_0 = 0.0$, I then used equation (5.5) to plot an approximation of $l(\theta) - l(\theta_0)$ against θ. Thus I was able to find the MLE to be $\theta \approx -0.4$ (figure 5.1, bottom right).

Frank and Strauss [35] and more recently Snijders [82] took as their observables the number of edges, twostars, and triangles (cycles of length three). Frank and Strauss maximised the pseudolikelihood as a rough guess at the MLE and Snijders did the same in order to compare the result to an estimate for the MLE that he calculated by MCMC simulations. Snijders drew attention to the fact that they were completely different! He also mentioned that his simulations had only 60 thousand steps in them, gave evidence that the parameters were highly correlated, and said that quite different estimates for the MLE may be found. The results of Frank and Strauss, and of Snijders, are given in table 5.3.

I added a function to my C program to deal with these three observables - edges, twostars, triangles - and proceeded to estimate a good θ_0 for an MCMC simulation, with Metropolis-Hastings updating steps, in the way that I outlined at the end of subsection 5.4.5. Denote by $(\theta_1, \theta_2, \theta_3)$ the parameters for (edges,

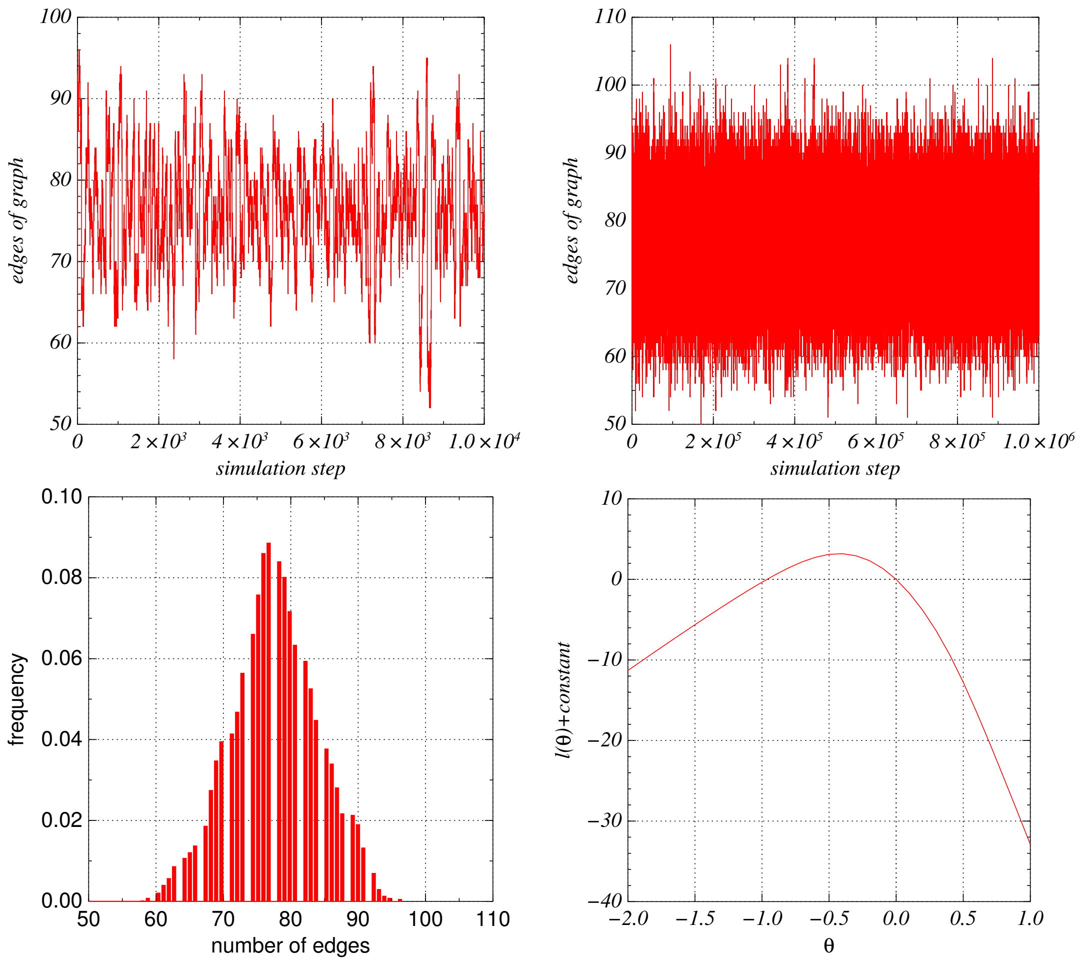

Figure 5.1: Observable: edges. **Top left and right**: The edges of the current graph in an MCMC simulation (with $\theta_0 = 0.0$) are plotted against the simulation (time) step. This model is exactly soluble and the expected average for this θ_0 is 76.5. Notice that there is no significant transient. **Bottom left**: The number of edges of the graphs in the last 10^5 steps of a ten million step simulation (again with $\theta_0 = 0.0$) are plotted against their frequencies. **Bottom right**: The approximation to $l(\theta) - l(\theta_0)$ of equation (5.5) is plotted against the edge parameter θ. The sample of graphs used to do this was the last 10^5 of a 30 million step simulation with $\theta_0 = 0.0$.

Observable	MCMCMLE (this work)	MCMCMLE (Snijders [82])	pseudolikelihood (Snijders [82])	pseudolikelihood (Frank and Strauss [35])
Edges	3.5	-0.77476	-0.0810	-0.10
Twostars	-0.6	-0.04875	-0.2285	-0.23
Triangles	1.1	0.34520	0.9754	0.97

Table 5.3: MLE parameter estimation for: symmetrised Sampson data (table 5.2).

twostars, triangles). I discovered that for $\theta_0 = (4.6, -0.8, 1.5)$, the chain seemed to always give relatively small values in the running totals of the changes in the vector of observables after 10, 20, and 30 million steps. The running totals in the changes of edges, for instance, usually had magnitude less than 15. I decided to consider the cubic grid of points satisfying $-2.0 \leqslant \theta_1 \leqslant 6.0$, $-2.0 \leqslant \theta_2 \leqslant 1.0$, $-1.0 \leqslant \theta_3 \leqslant 2.9$, with the distances between values of θ_1 being 0.5, between θ_2 being 0.2, and between θ_3 being 0.3. Then, for $\theta_0 = (4.6, -0.8, 1.5)$ I ran a simulation of 30 million steps, took the next 10^5 graphs as a sample, and calculated the approximation of $l(\theta) - l(\theta_0)$ of equation (5.5) at each of the points of the cubic grid. I estimated the MLE as the point θ on the cubic grid at which the approximation of $l(\theta) - l(\theta_0)$ was biggest. (This brute force approach was adopted for two reasons: first, due to random noise effects there may be local maxima of the approximation of $l(\theta) - l(\theta_0)$ that are nowhere near the true MLE, so that search direction methods could be hampered or complicated; second, I didn't have time to do anything else.)

On the first simulation I got an MLE of $(4.5, -0.6, 0.8)$. Repeated simulations gave a fairly varied series of MLEs but one in particular kept cropping up, namely $(3.5, -0.6, 1.1)$. So I took this as my θ_0 and found that it gave, after 10, 20, and 30 million steps, even smaller values in the running totals of the changes in the vector of observables than for the previous θ_0. The running totals in the changes of edges typically had magnitude less than 10, intuitively indicating good convergence. For $\theta_0 = (3.5, -0.6, 1.1)$ I now carried out my brute force cubic grid MLE search and a varied sequence of MLEs once more resulted. But whenever I tried these MLEs as my θ_0, then, unless they were close to $(3.5, -0.6, 1.1)$, they gave large running totals of changes, which made the evaluation of the approximation to $l(\theta) - l(\theta_0)$ of equation (5.5) essentially impossible. A sum of many thousands of exponentials to powers of more than a thousand is not evaluated as a finite number in C. On the other hand, for $\theta_0 = (3.5, -0.6, 1.1)$, I found the approximation to $l(\theta) - l(\theta_0)$ to range from about -220 to 0 or 1 over my entire cubic grid. The MLE was not always close to θ_0 but at $\theta = \theta_0$ the value of the approximation to $l(\theta) - l(\theta_0)$ was always very close to the biggest value it took over the grid. (In fact I reassuringly found that the approximation to $l(\theta) - l(\theta_0)$ was 0.0 at $\theta = \theta_0$.) Thus this θ_0 appears to give good convergence and produces a fairly 'flat' loglikelihood function around θ_0 (flat in the sense of a small range of values for the estimate of $l(\theta) - l(\theta_0)$). Yet the loglikelihood will generally only be flat in the vicinity of its MLE (see, for instance, figure 5.1, bottom right).

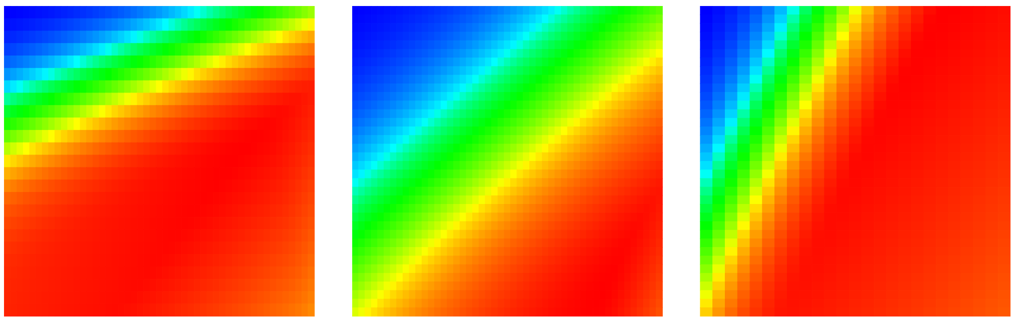

Figure 5.2: Observables: edges (parameter θ_1), twostars (θ_2), triangles (θ_3). A simulation of 30 million steps is run with $\theta_0 = (3.5, -0.6, 1.1)$. Last 100000 graphs taken as a sample. Plots here are projections of the approximation to $l(\theta) - l(\theta_0)$ of equation (5.5). Left: $\theta_1 = 3.5$, $-2.0 \leqslant \theta_2 \leqslant 1.0$, $-1.0 \leqslant \theta_3 \leqslant 2.9$, θ_2 vertical axis, increasing top to bottom, θ_3 horizontal, increasing left to right. Middle: $\theta_2 = -0.6$, $-2.0 \leqslant \theta_1 \leqslant 6.0$, $-1.0 \leqslant \theta_3 \leqslant 2.9$, θ_1 vertical. Right: $\theta_3 = 1.1$, $-2.0 \leqslant \theta_1 \leqslant 6.0$, $-2.0 \leqslant \theta_2 \leqslant 1.0$, θ_1 vertical. Red areas show biggest values of approximation of $l(\theta) - l(\theta_0)$, blue smallest.

All of this mounting circumstantial evidence has given me cause to consider $(3.5, -0.6, 1.1)$ to be a fair candidate for the MLE. I include it in table 5.3. For a graphical appreciation of the situation, I made three plots on the cubic grid, each holding one of either θ_1, θ_2, or θ_3 constant (see figure 5.2). It is difficult to deduce uniqueness of the MLE from the plots alone. I have experimented with grids of various different sizes (but still holding one of the parameters fixed at the values just stated), but seem to repeatedly get pictures like those in figure 5.2. If these pictures are to give the kind of amoebae we saw in figure 4.3, the amoebae would have to be extremely large.

Notice that my MLE estimate is wildly different from that of Snijders (table 5.3). I have to question how Snijders felt the confidence to state his results to five decimal places after simulations of only 60 thousand steps when the ensemble has $2^{\binom{18}{2}} > 10^{45}$ graphs and when he freely admits that quite different estimates for the MLE may be found. In any case I have given pause for thought on this well studied data set.

Concluding remarks

We began with the intention of laying down a broad framework for the modelling of all real-world networks. A brief review of network modelling to date revealed that only one suitably general candidate had so far emerged, the *exponential random graph* (ERG) model. We described the model, defining the *ensemble* to be the set of graphs that our real-world network could reasonably be expected to become. Just how reasonable it was that our network could become a particular graph in the ensemble was estimated by maximising Gibbs' entropy subject to constraints involving measurements of several properties or *observables* of the network. This gave us a *probability distribution* over the ensemble.

The rest of the project was devoted to assessing how appropriate the model was as a practical tool. For some straightforward examples it was discovered that the equations governing the model could be simplified drastically and we called such models *soluble*. One such example was the well-known Bernoulli model with the ensemble of all undirected graphs on a fixed number of nodes and a single observable, namely the total number of edges. An attempt was made, with some success, to introduce a method by which further soluble models may be found. The method was to restrict the values of the observables to well-behaved patterns, and with the pattern being an arithmetic progression, or a related sequence, two new models - Terry 1 and Terry 2 - were born. Despite their solubility it became clear that some complicated maths lay beneath the surface.

We then sought general conditions under which the governing equations of the ERG model had solutions. This was not the same as finding soluble models, for of course the existence of solutions is not the same as the possession of them. We found, by recourse to the *likelihood* function, that a solution *always* exists to the ERG model and is provided through the maximising of this function, or its logarithm.

Finally we discussed *Markov Chain Monte Carlo simulations* as a tool for maximising the loglikelihood in models not known to be soluble. Despite a firm theoretical basis for this approach and the knowledge that in some ways it is computationally efficient, it was observed that in certain cases the simulations give poor or unreliable results. We considered one particular example, the Sampson monastery data. We found that with a single observable, the total number of edges, the simulations converged very rapidly and yielded a well-behaved approximation to the loglikelihood. But for three observables - the number of edges, twostars, and triangles - it was difficult to draw any exact conclusions, even after 30 million steps.

Various suggestions for future research were made throughout the work. The restriction of observable values to predictable patterns should be explored as a means of constructing soluble models. Precise conditions on the observables alone that guarantee a unique solution to the governing equations of the model could be investigated. Above all the applicability of Markov Chain Monte Carlo simulations needs to be understood. Very recent research into soluble models has been conducted by Newman and Park [72] and the appropriateness of Monte Carlo simulations is being actively pursued by Snijders, Handcock, and others [83].

As a general device for network analysis the ERG model is in principle sensible but in practice often limited. Even extensive computation may yield little of value. Until more research has been carried out into the practical aspects of the model, it will not be possible to give it a ringing endorsement.

Appendix A

Proof of theorem 3.1

Theorem 3. 1. *The number of solutions to the expectation constraint of the Terry 2 model for $\theta < 0$ depends on the value of m. In fact:*

A *If $m = 0$ then $\theta = -\infty$ is a solution*

B *If $1 \leqslant m < n-1$ and $n \geqslant 5$ there exists a real number $k_n \in \left(\frac{1}{n}, \frac{n-1}{n}\right)$, dependent on n, such that for all $m < k_n n$ there is at least one finite solution $\theta < 0$ to the expectation constraint. For large n, $k_n \approx \frac{2-\sqrt{2}}{2} = 0.293$ to 3 d.p.*

C *If $m = n-1$ then there are no solutions for $\theta < 0$*

As we saw in chapter 3, the expectation constraint of the Terry 2 model may be written as

$$f(y) = \frac{y - (m+1)y^{m+1} - (n-m)y^{n-m}}{y + 1 - y^{m+1} - y^{n-m}} + \frac{y}{1-y} - m = 0 \tag{A.1}$$

where $y = e^{\theta}$, $m \in \mathbb{N} \cup \{0\}$, $n \in \mathbb{N}$, and $0 \leqslant m \leqslant n-1$. A finite solution for $\theta < 0$ is equivalent to a solution for $y \in (0,1)$ and the solution $\theta = -\infty$ is equivalent to the solution $y = 0$.

Proof of part A. Notice that when $m = 0$, $f(0) = 0$, so the first part of the theorem is trivially true.

□

Proof of part B. Assume $1 \leqslant m < n-1$ and $n \geqslant 5$. We shall show

1. $f(0) < 0$
2. $f(1) > 0$ if $\frac{m}{n} < k_n = 1 - \frac{1}{2n} - \frac{1}{2n}\sqrt{2n^2 - 2n + 1}$
3. $k_n \in \left(\frac{1}{n}, 1\right)$
4. $f(y)$ is continuous on $[0,1]$
5. $k_n \approx \frac{2-\sqrt{2}}{2}$ for large n

By the intermediate-value theorem (see [7]) results 1, 2, and 4 will guarantee, for m in the range $1 \leqslant m < k_n n$, the existence of a $y^* \in (0,1)$ such that $f(y^*) = 0$. Hence a finite solution for $\theta < 0$. Result 3 will guarantee that it makes sense to speak of the range $1 \leqslant m < k_n n$, since it will ensure $k_n n > 1$ and $k_n n < n-1$. In other words, this range will be non-empty (it will include $m = 1$) and will not include any m outside the range we are considering ($1 \leqslant m < n-1$). Result 5 will establish the final detail of part B of the theorem.

Proof of result 1

We have $f(0) = -m$, so that $m \geqslant 1 \Rightarrow f(0) < 0$.

Proof of result 2

We rewrite equation (A.1) as a single fraction:

$$f(y) = \frac{\text{num}(y)}{\text{den}(y)} \tag{A.2}$$

where

$$\text{num}(y) = -m + 2y + my^2 - y^{m+1} - (n-2m)y^{n-m} + (n-2m-1)y^{n-m+1} \tag{A.3}$$

and

$$\text{den}(y) = 1 - y^2 - y^{m+1} + y^{m+2} - y^{n-m} + y^{n-m+1}. \tag{A.4}$$

Now observe that $\text{num}(1) = \text{den}(1) = 0$, so that $f(1)$ is an indeterminate form of type $\frac{0}{0}$. We may evaluate it by l'Hôpital's rule (see [7]). We shall use l'Hôpital's rule (or one version of it) to find $\lim_{y\to 1} \frac{\text{num}(y)}{\text{den}(y)}$. To be allowed to use the rule there has to exist an open interval (a,b) containing 1 such that

(i) The derivatives of $\text{num}(y)$ and $\text{den}(y)$ exist on the interval except possibly at 1

(ii) The derivative of $\text{den}(y)$ is non-zero on the interval except possibly at 1

We obviously have (i) since $\text{num}(y)$ and $\text{den}(y)$ are both polynomials in y. We can show that (ii) is true by differentiating $\text{den}(y)$ and expressing the result as

$$\frac{d}{dy}\text{den}(y) = (y^{m+1}-y)+(m+1)(y^{m+1}-y^m)+(n-m)(y^{n-m}-y^{n-m-1})+(y^{n-m}-y). \tag{A.5}$$

We are assuming that $1 \leqslant m < n-1$, so we can say that $n-m > 0$, $n-m-1 > 0$, and $m+1 > 0$ in equation (A.5). Now if $y < 1$, $y^{t_1} > y^{t_2}$ for any real t_1, t_2 such that $0 \leqslant t_1 < t_2$. But then each bracket of y terms in (A.5) is negative. So $\frac{d}{dy}\text{den}(y) < 0$. Further, if $y > 1$, $y^{t_1} < y^{t_2}$ for $0 \leqslant t_1 < t_2$. But then each bracket of y terms in (A.5) is positive. So $\frac{d}{dy}\text{den}(y) > 0$. Hence $\frac{d}{dy}\text{den}(y) \neq 0$ whenever $y \neq 1$. This ensures we have (ii) for *any* open interval containing 1.

We may therefore use l'Hôpital's rule. So let us differentiate $\text{num}(y)$ (equation (A.3)) to get

$$\frac{d}{dy}\text{num}(y) = 2+2my-(m+1)y^m-(n-2m)(n-m)y^{n-m-1}+(n-2m-1)(n-m+1)y^{n-m}. \tag{A.6}$$

By equation (A.5) we find $\frac{d}{dy}\text{den}(y)\Big|_{y=1} = 0$ and by (A.6) we get $\frac{d}{dy}\text{num}(y)\Big|_{y=1} = 0$. So we have another indeterminate form of type $\frac{0}{0}$! We require *another* application of l'Hôpital's rule, provided this is allowable. Once again, condition (i) (for $\frac{d}{dy}\text{num}(y)$ and $\frac{d}{dy}\text{den}(y)$) will be satisfied since they are both polynomials, and we shall argue condition (ii) (for $\frac{d^2}{dy^2}\text{den}(y)$) retrospectively. Let us, then, find the second derivatives of $\text{num}(y)$ and $\text{den}(y)$. From equations (A.6) and (A.5), and using our assumption that $m < n-1$ (so that $n-m-2 \geqslant 0$), we may write

$$\begin{aligned}\frac{d^2}{dy^2}\text{num}(y) &= 2m - (m+1)my^{m-1} - (n-2m)(n-m)(n-m-1)y^{n-m-2} \\ &\quad +(n-2m-1)(n-m+1)(n-m)y^{n-m-1} \qquad \text{(A.7)}\end{aligned}$$

$$\begin{aligned}\frac{d^2}{dy^2}\text{den}(y) &= -2 - (m+1)my^{m-1} + (m+2)(m+1)y^m \\ &\quad -(n-m)(n-m-1)y^{n-m-2} + (n-m+1)(n-m)y^{n-m-1}. \qquad \text{(A.8)}\end{aligned}$$

It follows that

$$\frac{d^2}{dy^2}\text{num}(y) = 2m^2 + m(2-4n) + (n^2-n) \qquad \text{(A.9)}$$

and that

$$\frac{d^2}{dy^2}\text{den}(y) = 2n. \qquad \text{(A.10)}$$

Equation (A.8) shows that $\frac{d^2}{dy^2}\text{den}(y)$ is a polynomial in y, so it must be continuous. By (A.10) we see that it takes the value $2n$, which is > 0, at $y = 1$. It must then be that there is an open interval of y containing 1 on which the function is non-zero. If there were not, then there would exist an open interval containing 1 in which points arbitrarily close to 1 would be zero, so that the function would be discontinuous at 1. But as we just noted, the function is continuous. Hence condition (ii) (for this function) is satisfied.

Thus l'Hôpital's rule is again valid and in fact we've already taken the appropriate limits in (A.9) and (A.10). So we have

$$\lim_{y\to 1} f(y) = \frac{2m^2 + m(2-4n) + (n^2-n)}{2n}. \qquad \text{(A.11)}$$

So $f(1)$ exists and we want to know when it is positive. Since the denominator in (A.11) is necessarily positive, we will have $f(1) > 0$ when the numerator is positive:

$$2m^2 + m(2-4n) + (n^2-n) > 0. \qquad \text{(A.12)}$$

We treat this as a quadratic in m. Since the coefficient of the leading term is positive, this quadratic will be positive whenever m is either less than the smaller root or greater than the larger root of the following equation:

$$2m^2 + m(2-4n) + (n^2-n) = 0. \qquad \text{(A.13)}$$

In other words we find the quadratic to be positive when either

$$m < n - \frac{1}{2} - \frac{1}{2}\sqrt{2n^2-2n+1} \qquad \text{(A.14)}$$

or

$$m > n - \frac{1}{2} + \frac{1}{2}\sqrt{2n^2 - 2n + 1}. \tag{A.15}$$

Now, since we assumed at the start that $n \geqslant 5$ we certainly have $n \geqslant 2$. But then $2n^2 \geqslant 2n$ or $2n^2 - 2n + 1 \geqslant 1$, so that inequalities (A.14) and (A.15) reassuringly do not involve the square root of a negative number. In fact since $2n^2 - 2n + 1 \geqslant 1$, the positive square root of $2n^2 - 2n + 1$ is also $\geqslant 1$, so that

$$n - \frac{1}{2} + \frac{1}{2}\sqrt{2n^2 - 2n + 1} \geqslant n - \frac{1}{2} + \frac{1}{2} = n > m. \tag{A.16}$$

Thus inequality (A.15) cannot possibly hold.

Finally inequality (A.14) may be re-expressed as

$$\frac{m}{n} < k_n = 1 - \frac{1}{2n} - \frac{1}{2n}\sqrt{2n^2 - 2n + 1}. \tag{A.17}$$

This proves result 2.

Proof of result 3

Assume $n \geqslant 5$. Then $(n-1)(n-4) > 0$. This is equivalent to

$$(2n-3)^2 > 2n^2 - 2n + 1. \tag{A.18}$$

As we saw just before inequality (A.16), $2n^2 - 2n + 1 \geqslant 1$ for $n \geqslant 2$, so this must be true for $n \geqslant 5$. Also $2n - 3 > 0$ for $n \geqslant 5$. So we may take the positive square root in (A.18) i.e. we can write

$$2n - 3 > \sqrt{2n^2 - 2n + 1}. \tag{A.19}$$

This is easily rearranged:

$$1 - \frac{1}{2n} - \frac{1}{2n}\sqrt{2n^2 - 2n + 1} > \frac{1}{n}. \tag{A.20}$$

But the left hand side of (A.20) is k_n. So $k_n > \frac{1}{n}$. If we again use the fact that $2n^2 - 2n + 1 \geqslant 1$ for $n \geqslant 5$ we then trivially see that

$$-\frac{1}{2}\sqrt{2n^2 - 2n + 1} < -\frac{1}{2} \tag{A.21}$$

or

$$nk_n = n - \frac{1}{2} - \frac{1}{2}\sqrt{2n^2 - 2n + 1} < n - 1. \tag{A.22}$$

Hence $k_n < \frac{n-1}{n}$, so that $k_n \in \left(\frac{1}{n}, \frac{n-1}{n}\right)$ as required.

Proof of result 4

Earlier we wrote $f(y)$ as a function $\frac{\text{num}(y)}{\text{den}(y)}$ of two polynomials (equations (A.2) to (A.4)). In particular it follows that f(y) is continuous on $[0, 1]$ if $\text{den}(y) \neq 0$ on $[0, 1)$ and $\lim_{y \to 1} f(y)$ exists and is finite. But we have seen in the proof of result 2 that $\lim_{y \to 1} f(y)$ exists and is finite. The proof here will certainly be complete then if we show that $\text{den}(y) > 0$ on $[0, 1)$.

Recall equation (A.4):

$$\mathrm{den}(y) = 1 - y^2 - y^{m+1} + y^{m+2} - y^{n-m} + y^{n-m+1}.$$

In other words:

$$\mathrm{den}(y) = (1 - y^2) + (y^{m+2} - y^{m+1}) + (y^{n-m+1} - y^{n-m}). \tag{A.23}$$

But notice that for $0 \leqslant y < 1$, $1 - y^2 > 0$, $y^{m+2} - y^{m+1} > 0$, and $y^{n-m+1} - y^{n-m} > 0$ since $m + 1 > 0$ and $n - m > 0$. Hence the right hand side of (A.23) is positive and the proof is complete.

Proof of result 5

We have

$$k_n = 1 - \frac{1}{2n} - \frac{1}{2n}\sqrt{2n^2 - 2n + 1} = 1 - \frac{1}{2n} - \frac{1}{2n}\sqrt{2(n - \frac{1}{2})^2 + \frac{1}{2}}. \tag{A.24}$$

So for large n:

$$\begin{aligned} k_n &\approx 1 - \frac{1}{2n}\sqrt{2(n - \frac{1}{2})^2} = 1 - \frac{1}{2n}\sqrt{2}(n - \frac{1}{2}) \\ &\approx 1 - \frac{n\sqrt{2}}{2n} = 1 - \frac{\sqrt{2}}{2} = \frac{2 - \sqrt{2}}{2}. \end{aligned} \tag{A.25}$$

□

Proof of part C. If $m = n - 1$ then, using equation (A.3), the numerator in $f(y) = \frac{\mathrm{num}(y)}{\mathrm{den}(y)}$ becomes

$$\mathrm{num}(y) = 1 - n + ny - y^n. \tag{A.26}$$

We rewrite this as follows:

$$\begin{aligned} \mathrm{num}(y) &= -n(1 - y) + (1 - y^n) \\ &= -n(1 - y) + (1 - y)(1 + y + y^2 + \cdots + y^{n-1}) \\ &= (1 - y)(1 + y + y^2 + \cdots + y^{n-1} - n). \end{aligned} \tag{A.27}$$

Now recall that a solution $\theta < 0$ is equivalent to a solution to $f(y) = 0$ for $y \in [0, 1)$. Recall also by result 4 of part B that $\mathrm{den}(y) > 0$ for $y \in [0, 1)$. Then, for a solution $\theta < 0$ to exist, there must be a $y \in [0, 1)$ such that $\mathrm{num}(y) = 0$. But for this range of y, $1 - y > 0$ and

$$1 + y + y^2 + \cdots + y^{n-1} < \underbrace{1 + 1 + 1 + \cdots + 1}_{n\text{times}} = n,$$

so that the product of the brackets in (A.27) is necessarily negative. So there are no solutions.

□

Conjecture 3. 1. *For $n \geqslant 5$, a solution $\theta < 0$ exists only for $m < k_n n$. When it does exist it is unique.*

A possible proof of this would involve showing firstly that $f(0) < f(1)$ and secondly that the function $f(y)$ is strictly increasing on $(0, 1)$. It is in fact relatively straightforward to show that the first of these assuming $m < n - 1$. We know after all that $f(0) = -m$ and (by equations (A.9) and (A.10)) that

$$f(1) = \frac{2m^2 + m(2 - 4n) + (n^2 - n)}{2n}.$$

Hence by assuming the reverse of what we want, namely $f(0) > f(1)$ we get, after a little rearrangement

$$(n - m - 1)(n - m) < -m,$$

which must be false for $n - m - 1 > 0$ or (as we just mentioned) $m < n - 1$.

To prove that $f(y)$ is strictly increasing would be thoroughly unpleasant, however. To do this would involve demonstrating that the second derivative of $f(y)$ is positive on $(0, 1)$. I actually found this second derivative using the computer algebra system maxima but the expression was over a page and a half long! When I factorised it, it was more than three pages long! Hence I leave this as an exercise for the reader.

Appendix B

Program listings

I wrote dozens of programs for this project but due to considerations of space I can only present streamlined versions of the two most important here.

```
#!/usr/local/bin/python
# AJT 2005 June,July,August
# mcopy -o -v ./Chapters2and4pics.py a:
# to make bar charts in chapter 2, use linux command:
# geng 4 | showg -e -l0 | ./rearrangeg.py |
# ./Chapters2and4pics.py | ./bars3.py | dm "x1+1.0" "x2" | graph -Tx -C
# to make colour grids in chapter 4:
# geng 5 | showg -e -l0 | ./rearrangeg.py | ./Chapters2and4pics.py | matrix2pdf3.py

import re
from sys import stdin,stderr,exit,argv
from math import exp,log
graph=re.compile(r'^Graph (\d+),')
nm=re.compile(r'(\d+)\s+(\d+)')
ed=re.compile(r'\s*(?P<a>\d+)\s+(?P<b>\d+)')
ng=0
maxng=10000
graphs=[]

class G:
  '''
  Represents a graph
  '''
  def __init__(s,ng,n,m,adjdict,adjmatrix):
    s.ng=ng       # graph number
    s.n=n         # number of nodes
    s.m=m         # number of edges
    s.adjdict=adjdict       # adjacency dictionary
    s.adjmatrix=adjmatrix     # adjacency matrix
  def __repr__(s):
    # return 'G(%d,%d,%g)'%(s.ng,s.n,s.m)
    return 'Graph %d has %d nodes, %d edges'%(s.ng,s.n,s.m)
    # return 'Graph has adjacency dictionary', s.adj
  def get_nnodes(s): # this gives security apparently
    return s.n
  def get_nedges(s):
    return s.m
  def get_adjacencydict(s):
    return s.adjdict
  def get_adjacencymatrix(s):
    return s.adjmatrix
  def __iter__(s):
    return iter(s.adjdict.keys())

def Ham1 (theta, graph): # calculates Bernoulli Hamiltonian
  Hamiltonian = theta*(graph.m)
  return Hamiltonian

def Ham4 (theta, graph): # calculates twostar Hamiltonian
  Hamiltonian = 0
  for i in range(graph.n):
    temp = len(graph.adjdict[i])
    Hamiltonian+=theta*temp*(temp-1)
  return Hamiltonian/2

def Ham5 (theta1, theta2, graph): # "edges-twostars" Hamiltonian
 # theta1 edge parameter, theta2 twostar parameter
  Hamiltonian=0
  for i in range(graph.n):
    temp = len(graph.adjdict[i])
    Hamiltonian+=theta2*temp*(temp-1)
  Hamiltonian= Hamiltonian/2
  Hamiltonian += theta1*(graph.m)
  return Hamiltonian

def Ham10 (theta1,theta2, graph): # "nodes of degree1 and degree2" Hamiltonian
  nodes_of_degree1=0
  for i in range(graph.n):
    if (len(graph.adjdict[i])==1):
      nodes_of_degree1+=1
  nodes_of_degree2=0
  for i in range(graph.n):
    if (len(graph.adjdict[i])==2):
      nodes_of_degree2+=1
  Hamiltonian = theta1*nodes_of_degree1 + theta2*nodes_of_degree2
  return Hamiltonian

while ng<maxng:
  line=stdin.readline()
  if not line: break
  x=graph.match(line)
  if x: # 'Graph' matched
    ng+=1
    line=stdin.readline()
    y=nm.match(line)
    if not y:
      print >>stderr,'nm match failed'
      exit(1)
    n,m=map(int,y.groups()) # new graph, # vertices and # edges
    # print 'graph %d n=%d m=%d'%(ng,n,m)
    adj=dict([(i,[ ]) for i in range(n)])
    line=stdin.readline()
    for e in re.finditer(ed,line):
      a,b=int(e.group('a')),int(e.group('b'))
      adj[a].append(b)
      adj[b].append(a)
    # print '   ',len(adj[0]),'edges at node 0'
    # print '    adjacency dictionary: ',adj
    # next we define adjacency matrix. we zero everything initially
    matrix=dict([(i,[0]*n) for i in range(n)])
    # now we can fill in the 1's by using the adjacency dictionary
    for i in range(n):
      for j in range(n):
        if j in adj[i]:
          matrix[i][j]=matrix[j][i]=1
    # print 'adjacency matrix is', matrix
    graphs.append(G(ng,n,m,adj,matrix))

#print 'Using Hamiltonian to control the number of edges only: \n'
Z=0.0
theta=0.3
for graph in graphs:
  H = Ham1(theta,graph) # Replace by Ham4 to control twostars
  Z+= exp(H)
if verbosity>1: print 'For Bernoulli model Z is ', Z
for k,graph in enumerate(graphs):
  H = Ham1(theta,graph) # Replace by Ham4 to control twostars
  P = exp(H)/Z
  #print k,P   # printing this alone as output can be used to plot them

# next we shall combine some observables
#print 'Using Hamiltonian to control total edges AND number of 2-stars: \n'
Z=0
theta=0.1   # edge parameter
alpha=0.1  # two-stars parameter
for graph in graphs:
  H = Ham5(theta,alpha,graph)
  Z+= exp(H)
for k,graph in enumerate(graphs):
  H = Ham5(theta,alpha,graph)
  P = exp(H)/Z
  #print k,P   # printing this alone as output can be used to plot them
```

```
# Next we plot loglikelihood function for a given graph where observable is edges
theta=-4.0
while theta<4.0:
  Z=0.0
  for graph in graphs:
    H = Ham1(theta,graph)
    Z+= exp(H)
  # now Z is partition function for theta value in current step of the loop
  loglikely = Ham1(theta,graphs[1]) - log(Z)
  #print theta, loglikely
  theta+=0.01

# Next we plot likelihood function (2d colour grid) for a given graph
# Ham5 - edges and two-stars
# Ham10 - nodes of degree1 (theta) and nodes of degree2 (alpha)
theta=-3.0
while theta<3.01:
  alpha=-3.0
  while alpha<3.01:
    Z=0.0
    for graph in graphs:
      H = Ham5(theta,alpha,graph)
      Z+= exp(H)
      # now Z is partition function for theta, alpha in current step of the loop
    likely = exp(Ham10(theta,alpha,graphs[0])) / Z
    print likely,
    alpha+=0.15
  print ""
  theta+=0.15
```

```
// AJT July and August 2005
// gcc -Wall MCMCMLE08.c -o MCMCMLE08 -lm && ./MCMCMLE08
// gcc -Wall MCMCMLE08.c -o MCMCMLE08 -lm && time ./MCMCMLE08
// mcopy -o -v MCMCMLE08.c a:
// Program to run an MCMC simulation of an ERG model to find MLE
// The ensemble will be all undirected graphs on n nodes

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

// prototypes next
double minof2(double,double);
int edges(char**,int);    // finds no. of edges of undirected graph
int* MCMCBernoulliMetHastInv(char**,int,int,int,double);
// performs MCMC simulation where observable is edges
// uses Metropolis-Hastings sampling plus inversions
int** MCMCEdgesTwostarsTrianglesMetHast(char** observed_graph, int n,
int length_of_chain, int sample_size, double theta_chain1, double theta_chain2,
double theta_chain3);  // performs MCMC simulation where graph observables are
// edges, twostars, and triangles, uses Metropolis-Hastings sampling
void invert_graph(char**, int);
double MLEfunction1(int*,int,double);
// finds MLE using output from MCMCBernoulliMetHastInv
double* MLEfunctionEdgesTwostarsTriangles(int** vector_of_change_statistics,
int sample_size, double theta_chain1, double theta_chain2, double theta_chain3);
char** Sampson_monastery_example_graph();

// next - globals!
const int n=18;  // no. of edges of graphs in ensemble
const double theta_chain=0.0;  // parameter value to be
// used in MCMCBernoulliMetHastInv simulation.
const double theta_chain1=3.5;
const double theta_chain2=−0.6;
const double theta_chain3=1.1;
const int length_of_chain=30000000;
const int sample_size=100000;

int main() {
  int* vector_of_change_statistics_Sampson;
  vector_of_change_statistics_Sampson=MCMCBernoulliMetHastInv(
  Sampson_monastery_example_graph(),18,length_of_chain,sample_size,theta_chain);
  double MLE_Sampson;
  MLE_Sampson=MLEfunction1(vector_of_change_statistics_Sampson,sample_size,theta_chain);
  //printf("MLE_Sampson is %f\n",MLE_Sampson);
  free(vector_of_change_statistics_Sampson);
  int i;

  int** vector_of_change_statistics_SampsonNew;
  vector_of_change_statistics_SampsonNew = MCMCEdgesTwostarsTrianglesMetHast(
  Sampson_monastery_example_graph(), n, length_of_chain, sample_size, theta_chain1,
  theta_chain2, theta_chain3);
  for (i=0; i<sample_size; i++) {
    //printf("vector_of_change_statistics_SampsonNew[0] value (edges)
    //%d\n",vector_of_change_statistics_SampsonNew[0][i]);
    //printf("vector_of_change_statistics_SampsonNew[1] value (twostars)
    //%d\n",vector_of_change_statistics_SampsonNew[1][i]);
    //printf("vector_of_change_statistics_SampsonNew[2] value (triangles)
    //%d\n",vector_of_change_statistics_SampsonNew[2][i]);
  }
  double* optimum_theta;
  optimum_theta = MLEfunctionEdgesTwostarsTriangles
  (vector_of_change_statistics_SampsonNew,
  sample_size, theta_chain1, theta_chain2, theta_chain3);

  for (i=0; i<3; i++) { free(vector_of_change_statistics_SampsonNew[i]); }
  free(optimum_theta);
  return 0;
}
```

```
// function definitions next

// Sampson monastery example.  Frank & Strauss Markov graphs page 839
#define AE(i,j)   example_graph[i−1][j−1]=1;
char** Sampson_monastery_example_graph() {
  char** example_graph=graph(18,0,0);
  AE( 1, 2) AE( 1, 3) AE( 1, 5) AE( 1,15)
  AE( 2, 3) AE( 2, 4) AE( 2, 5) AE( 2, 6) AE( 2, 7) AE( 2, 9) AE( 2,10)AE( 2,15)
  AE( 3, 5) AE( 3, 6) AE( 3, 7) AE( 3, 8) AE( 3, 9) AE( 3,10)
  AE( 4, 5) AE( 4, 6) AE( 4, 7) AE( 4,10)
  AE( 5, 6) AE( 5, 7)
  AE( 6, 7) AE( 6,11) AE( 6,14) AE( 6,15)
  AE( 7, 9) AE( 7,10)
  AE( 8, 9) AE( 8,10) AE( 8,11) AE( 8,12) AE( 8,13) AE( 8,14)
  AE( 9,10) AE( 9,11) AE( 9,12) AE( 9,13) AE( 9,16) AE( 9,17) AE( 9,18)
  AE(10,11) AE(10,12) AE(10,13) AE(10,14) AE(10,17) AE(10,18)
  AE(11,12) AE(11,14)
  AE(12,13) AE(12,14)
  AE(13,14) AE(13,15)
  AE(15,16) AE(15,17) AE(15,18)
  AE(16,17) AE(16,18)
  AE(17,18)
  int i,j;
  for (i=0; i<18; i++) {
    for (j=0; j<18; j++) {
      if (j<i) { example_graph[i][j]=example_graph[j][i]; }
    }
  }
  return example_graph;
}
#undef AE

inline double minof2(double x,double y) {
  return x<y?x:y;
}

int edges(char** graph, int n) {
  int m=0;  // will be returned as no. of edges
  int i,j;    // loop variables
  for (i=0; i<n; i++) {
    for (j=i+1; j<n; j++) {
      m+=graph[i][j];
    }
  }
  return m;
}

int** MCMCEdgesTwostarsTrianglesMetHast(char** graph, int n, int length_of_chain,
int sample_size, double theta_chain1, double theta_chain2, double theta_chain3) {
  int i,k;    // loop variables
  int** vector_of_change_statistics;        // we will return this
  int node1, node2;  // a randomly chosen dyad
  double exponent, pi1, pi2;  // used in finding probabilities
  double probability_of_change;
  int running_total_change_in_edges=0;
  int running_total_change_in_twostars=0;
  int running_total_change_in_triangles=0;
  int no_neighbours_node1, no_neighbours_node2;
  int no_neighbours_common_node1node2;
  int edge_changed;
  int no_twostars_changed; // used to find no. twostars changed on a simulation step
  int no_triangles_changed;
  int definite_edge_change;    // no. of edge changes on each step (+1 if edge added,
                               // -1 if edge removed)
  int definite_twostar_change;
  int definite_triangle_change;
  double x;  // will be a random no. from 0 to 1
  int counter=0;  // will be used as a sort of loop variable
                  // in defining vector_of_change_statistics
  vector_of_change_statistics=(int**)malloc(3*sizeof(int*));
  if (vector_of_change_statistics==NULL) {
    fprintf(stderr,"Out of memory, location X1\n");
    exit(−1);
  }
  for (i=0; i<3; i++) {
    vector_of_change_statistics[i]=(int*)malloc(sample_size*sizeof(int));
    if (vector_of_change_statistics[i]==NULL) {
      fprintf(stderr,"Out of memory, location X2\n");
      exit(−1);
    }
  }
//vector_of_change_statistics[0] will be edges
//vector_of_change_statistics[1] will be twostars
//vector_of_change_statistics[2] will be triangles
  srand(time(NULL)); // "seed" the random number generator

  for (i=0; i<length_of_chain; i++) {      // Here goes Chain!!!
    // first we randomly select a dyad
    node1=rand()%n; // node1 is now a random int from 0 to n-1
    node2=node1;
    while (node2==node1) {
      node2=rand()%n;
    }
    // Now node1 and node2 are 2 DIFFERENT random ints from 0 to n-1
    definite_edge_change=0;
    definite_twostar_change=0;
    definite_triangle_change=0;
    no_neighbours_node1=0;
    no_neighbours_node2=0;
    no_neighbours_common_node1node2=0;
```

```
for (k=0; k<n; k++) {
  no_neighbours_node1+=graph[node1][k];
  no_neighbours_node2+=graph[node2][k];
  if (graph[node1][k]==1) {
    if (graph[node2][k]==1) {
      no_neighbours_common_node1node2+=1;
    }
  }
}
// Now no_neighbours_node1 is exactly that for the graph on current step!! etc..
// Now we use Metropolis-Hastings method to decide
// if (node1,node2) should be an edge or not
// Notice that it may already be an edge in fact

x=(double)rand()/RAND_MAX; // a random number from 0 to 1

if (graph[node1][node2]==1) {  // we want to know if we should REMOVE the edge
  edge_changed=1;
  no_twostars_changed=no_neighbours_node1 + no_neighbours_node2 − 2;
  no_triangles_changed=no_neighbours_common_node1node2;

  exponent= − (theta_chain1*edge_changed + theta_chain2*no_twostars_changed
  + theta_chain3*no_triangles_changed);
  pi2=exp(exponent);
  probability_of_change=minof2(1,pi2);

  if (probability_of_change>x) {   // we remove edge (node1,node2)
    graph[node1][node2]=0;
    graph[node2][node1]=0;
    definite_edge_change= − edge_changed;
    definite_twostar_change= − no_twostars_changed;
    definite_triangle_change= − no_triangles_changed;
  }
} else {  // we want to know if we should ADD the edge

  edge_changed=1;
  no_twostars_changed=no_neighbours_node1 + no_neighbours_node2;
  no_triangles_changed=no_neighbours_common_node1node2;

  exponent=theta_chain1*edge_changed + theta_chain2*no_twostars_changed
  + theta_chain3*no_triangles_changed;
  pi1=exp(exponent);
  probability_of_change=minof2(1,pi1);

  if (probability_of_change>x) {   // we add edge (node1,node2)
    graph[node1][node2]=1;
    graph[node2][node1]=1;
    definite_edge_change= edge_changed;
    definite_twostar_change= no_twostars_changed;
    definite_triangle_change= no_triangles_changed;
  }
```

```
    }
    running_total_change_in_edges+=definite_edge_change;
    running_total_change_in_twostars+=definite_twostar_change;
    running_total_change_in_triangles+=definite_triangle_change;

    if (i>=(length_of_chain−sample_size)) {
      vector_of_change_statistics[0][counter]=running_total_change_in_edges;
      vector_of_change_statistics[1][counter]=running_total_change_in_twostars;
      vector_of_change_statistics[2][counter]=running_total_change_in_triangles;
      counter+=1;
    }
  } // Here ends the Chain
  return vector_of_change_statistics;
} // Here ends the function

int* MCMCBernoulliMetHastInv(char** observed_graph, int n, int length_of_chain,
int sample_size, double theta_chain) {
  int* vector_of_change_statistics;     // we will return this
  double nchoose2=n*(n−1)*0.5;
  int current_edges=edges(observed_graph,n);
  double temp, prob_of_inversion;
  int i;    // loop variable
  int node1, node2; // a randomly chosen dyad
  double exp1,exp2; // temporary variables
  double p1, p2; // probabilities that will help determine
                        // the step in the Chain
  int running_total_change_in_edges=0;
  int definite_edge_change;   // no. of edge changes on each step
                              // of the simulation (+1 if edge added,
                              // -1 if edge removed)
  double x; // will be a random no. from 0 to 1
  int counter=0; // will be used as a sort of loop variable
                 // in defining vector_of_change_statistics
  srand(time(NULL)); // "seed" the random number generator
  char** graph=observed_graph;
  // we allocate memory
  vector_of_change_statistics=(int*)malloc(sample_size*sizeof(int));
  if (vector_of_change_statistics==NULL) {
    fprintf(stderr,"Out of memory, location 3\n");
    exit(−1);
  }

  for (i=0; i<length_of_chain; i++) {     // Here goes Chain!!!
    // first we ask if we should try an inversion step
    if (i%100==0) { // replace with i<0 to run chain with no inversions
      temp=theta_chain*(nchoose2−(2.0*current_edges));
      temp=exp(temp);
      prob_of_inversion=minof2(1,temp);
      srand(i);
      x=(double)rand()/RAND_MAX;
      if (prob_of_inversion>x) {
        invert_graph(graph,n);
        running_total_change_in_edges+=nchoose2−2*current_edges;
        current_edges=nchoose2−current_edges;
      }
      if (i>=(length_of_chain−sample_size)) {
        vector_of_change_statistics[counter]=running_total_change_in_edges;
        counter+=1;
      }
      continue;  // if we do inversion, we go to
                 // next time step of chain
    }
    // now we randomly select a dyad
    node1=rand()%n; // node1 is now a random int from 0 to n-1
    node2=node1;
    while (node2==node1) {
      node2=rand()%n;
    }
    // Now node1 and node2 are 2 DIFFERENT random ints from 0 to n-1
    //printf("node1,node2 are %d,%d\n",node1,node2);
    definite_edge_change=0;
    exp1=exp(−theta_chain);
    exp2=1.0/exp1;
    p1=minof2(1,exp1);
    p2=minof2(1,exp2);

    // Now we use Metropolis-Hastings method to decide
    // if (node1,node2) should be an edge or not
    // Notice that it may already be an edge in fact
    if (observed_graph[node1][node2]==1) {
      x=(double)rand()/RAND_MAX;
      if (p1>x) {  // we remove edge (node1,node2)
        graph[node1][node2]=0;
        graph[node2][node1]=0;
        definite_edge_change=−1;
      }
    } else {
      x=(double)rand()/RAND_MAX;
      if (p2>x) {  // we add edge (node1,node2)
        graph[node1][node2]=1;
        graph[node2][node1]=1;
        definite_edge_change=1;
      }
    }
    current_edges+=definite_edge_change;
    running_total_change_in_edges+=definite_edge_change;
    if (i>=(length_of_chain−sample_size)) {
      vector_of_change_statistics[counter]=running_total_change_in_edges;
      counter+=1;
    }
  }  // Here ends the Chain
  return vector_of_change_statistics;
```

```
} // Here ends the function

void invert_graph(char** graph, int n) {
  int i,j;    // loop variables
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++) {
      if (i!=j) {
        graph[i][j]=1−graph[i][j];
      }
    }
  }
}

double MLEfunction1(int* vector_of_change_statistics, int sample_size,
double theta_chain) {
  double biggest=−1e308; // a small number to start with
  double estimate, temp1,temp2,temp3; // estimate will be our estimate
                                      // of log likelihood
  int i;   // loop variable
  double theta, optimum_theta=0.0; // optimum_theta will be
                                   // returned as our MLE
  for (theta=−1.0; theta<1.0; theta+=0.1) {
    estimate=0.0;
    for (i=0; i<sample_size; i++) {
      temp1=theta−theta_chain;
      temp2=vector_of_change_statistics[i];
      temp3=temp1*temp2;
      estimate+=(exp(temp3)−estimate)/(i+1);
    }
    estimate=−log(estimate);
    if (estimate>biggest) {
      optimum_theta=theta;
      biggest=estimate;
    }
    printf("%f %f\n",theta,estimate); // can plot theta vs. loglikely approx
  }
  return optimum_theta;
}

double* MLEfunctionEdgesTwostarsTriangles(int** vector_of_change_statistics,
int sample_size, double theta_chain1, double theta_chain2, double theta_chain3) {
  double biggest=−1e308; // a small number to start with
  double* optimum_theta;
  optimum_theta=(double*)malloc(3*sizeof(double));
  // optimum_theta[0] will be optimum value of parameter for edges
  // optimum_theta[1] for twostars, optimum_theta[2] for triangles
  int i;   // loop variable
  double estimate;  // estimate will be our estimate of loglikelihood
  double theta1, theta2, theta3; // parameter space, over which we maximise loglikelihood
  double temp1, temp2, temp3, tempsufficient1, tempsufficient2, tempsufficient3, tempfinal;
  if (optimum_theta==NULL) {
      fprintf(stderr,"Out of memory, location Z\n");
      exit(−1);
  }
  for (theta1=−2.0; theta1<5.01; theta1+=0.5) {
    for (theta2=−1.0; theta2<0.21; theta2+=0.2) {
      for (theta3=−1.0; theta3<1.41; theta3+=0.3) {
        estimate=0.0;
        for (i=0; i<sample_size; i++) {
          temp1=theta1−theta_chain1;
          temp2=theta2−theta_chain2;
          temp3=theta3−theta_chain3;
          tempsufficient1=vector_of_change_statistics[0][i];
          tempsufficient2=vector_of_change_statistics[1][i];
          tempsufficient3=vector_of_change_statistics[2][i];
          tempfinal=temp1*tempsufficient1+temp2*tempsufficient2+temp3*tempsufficient3;
          estimate+=(exp(tempfinal)−estimate)/(i+1);
        }
        estimate=−log(estimate);
        if (estimate>biggest) {
          optimum_theta[0]=theta1;
          optimum_theta[1]=theta2;
          optimum_theta[2]=theta3;
          biggest=estimate;
        }
        printf("theta1,theta2,theta3,loglikelihood are %f,%f,%f,%f\n",
        theta1,theta2,theta3,estimate);
      }
    }
  }
  printf("MLE is optimum_theta[0], optimum_theta[1], optimum_theta[2] is
  %f,%f,%f\n",
  optimum_theta[0], optimum_theta[1], optimum_theta[2]);
  return optimum_theta; //remember to free memory for this after function call in main
}
```

Bibliography

[1] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 171–180, New York, 2000. Association of Computing Machinery.

[2] R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:113–126, 1973.

[3] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Review of Modern Physics*, 74:47–97, 2002.

[4] R. Albert, H. Jeong, and A.-L. Barabási. Diameter of the world-wide web. *Nature*, 401:130–131, 1999.

[5] L. A. N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Science USA*, 97:11149–11152, 2000.

[6] C. Anderson, S. Wasserman, and B. Crouch. A p* primer: Logit models for social networks. *Social Networks*, 21:37–66, 1999.

[7] H. Anton. *Calculus*. John Wiley & sons, New York, 5th edition, 1995.

[8] Various authors. Internet statistics: Distribution of languages on the internet. http://www.netz-tipp.de/languages.html.

[9] Various authors. Monte carlo methods. http://en.wikipedia.org/wiki/Monte_Carlo_method.

[10] D. L. Banks and K. M. Carley. Models for network evolution. *Journal of Mathematical Sociology*, 21:173–196, 1996.

[11] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[12] A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: The topology of the World Wide Web. *Physica*, A281:69–77, 2000.

[13] J. A. Barnes. Graph theory and social networks: A technical comment on connectedness and connectivity. *Sociology*, 3:215–232, 1966.

[14] J. Berg and M. Lässig. Connected random networks. *Physics Review Letters*, 89(228701), 2002.

[15] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, B36:192–225, 1974.

[16] S. A. Boorman and H. C. White. Social structure from multiple networks ii: Role structures. *Amercian Journal of Sociology*, 81:1384–1446, 1976.

[17] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tompkins, and J. Wiener. Graph structure in the web. *Computer Networks*, 33:309–320, 2000.

[18] Z. Burda, J. Jurkiewicz, and A. Krzywicki. Network transitivity and matrix models. *Physics Review*, E69(026106), 2004.

[19] J. E. Cohen, F. Briand, and C. M. Newman. *Community Food Webs: Data and Theory*. Springer, New York, 1990.

[20] J. Corander, K. Dahmström, and P. Dahmström. Maximum likelihood estimation for Markov graphs. Technical report, University of Stockholm, 1998. Research report, Department of Statistics.

[21] K. Dahmström and P. Dahmström. Ml-estimation of the clustering parameter in a Markov graph model. Technical report, University of Stockholm, 1993. Research report, Department of Statistics.

[22] J. A. Davis. Statistical analysis of pair relationships: Symmetry, subjective consistency, and reciprocity. *Sociometry*, 31:102–119, 1968.

[23] J. A. Davis. Clustering and hierarchy in interpersonal relations: Testing two theoretical models on 742 sociograms. *American Sociological Review*, 35:843–852, 1970.

[24] S. N. Dorogovtsev and J. F. F. Mendes. Effect of the accelerating growth of communication networks on their structure. *Physics Review*, E63(025101), 2001.

[25] S. N. Dorogovtsev and J. F. F. Mendes. Evolution of networks. *Advances in Physics*, 51:1079–1187, 2002.

[26] S. N. Dorogovtsev, J. F. F. Mendes, and A. M. Samukhin. Principles of statistical mechanics of random networks. http://arxiv.org/abs/cond-mat/0204111, 2002.

[27] S. N. Dorogovtsev, J. F. F. Mendes, and A. N. Samukhin. Structure of growing networks with preferential linking. *Physics Review Letters*, 85:4633–4636, 2000.

[28] A. W. F. Edwards. *Likelihood*. Cambridge University Press, Cambridge, 1972.

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

[30] P. Erdős and A. Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–61, 1960.

[31] P. Erdős and A. Rényi. On the strength of connectedness of a random graph. *Acta Mathematica Scientia Hungary*, 12:261–267, 1961.

[32] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *Computer Communications Review*, 29:251–262, 1999.

[33] S. Feinberg, M. Meyer, and S. Wasserman. Statistical analysis of multiple sociometric relations. *Journal of the American Statistical Association*, 80:51–67, 1985.

[34] D. A. Fell and A. Wagner. The small world of metabolism. *Nature Biotechnology*, 18:1121–1122, 2000.

[35] O. Frank and D. Strauss. Markov graphs. *Journal of the American Statistical Association*, 81:832–842, 1986.

[36] L. C. Freeman. A set of measures of centrality based on betweeness. *Sociometry*, 40:35–41, 1977.

[37] L. C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1:215–239, 1979.

[38] J. R. P. French. A formal theory of social power. *Psychological Review*, 63:181–195, 1956.

[39] M. F. Friedell. Organizations as semilattices. *American Sociological Review*, 32:46–54, 1967.

[40] C. J. Geyer and E. A. Thomson. Constrained Monte Carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society*, B54:657–699, 1992.

[41] I. M. Google. Google search: More, more, more. http://www.google.com/options/.

[42] W. T. Grandy. *Foundations of Statistical Mechanics: Equilibrium Theory*. Reidel, Dordrecht, 1987.

[43] S. Gupta, R. M. Anderson, and R. M. May. Network of sexual contacts: Implications for the pattern of spread of hiv. *AIDS*, 3:807–817, 1989.

[44] T. Guttmann. Research interests. http:www.ms.unimelb.edu.au/˜tonyg/.

[45] M. S. Handcock. Assessing degeneracy in statistical models of social networks. Technical report, University of Washington, 2003. Working paper 39, Center for Statistics and the Social Sciences.

[46] P. W. Holland and S. Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of Mathematical Statistical Association*, 76:33–50, 1981.

[47] P. Holme and B. J. Kim. Growing scale-free networks with tunable clustering. *Physics Review*, E65(026107), 2002.

[48] T. K. Hopkins. *The Exercise of Influence in Small Groups*. Bedminster, Totowa, New Jersey, 1964.

[49] B. A. Huberman. *The Laws of the Web*. MIT Press, Cambridge, MA, 2001.

[50] D. R. Hunter. Exponential random graph models for network data. http://www.ima.umn.edu/talks/workshops/11-17-21.2003/hunter/ergmtalk.pdf, 2003.

[51] E. T. Jaynes. *Papers on Probability, Statistics, and Statistical Physics*. Reidel, Dordrecht, 1983.

[52] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási. The large-scale organization of metabolic networks. *Nature*, 407:651–654, 2000.

[53] E. M. Jin, M. Girvan, and M. E. J. Newman. The structure of growing social networks. *Physics Review*, E64(046132), 2001.

[54] E. E. Jones and H. B. Gerard. *Foundations of Social Psychology*. John Wiley & sons, New York, 1967.

[55] P. Jordano, J. Bascompte, and J. M. Olesen. Invariant properties in coevolutionary networks of plant-animal interactions. *Ecology Letters*, 6:69–81, 2003.

[56] V. K. Kalapala, V. Sanwalani, and C. Moore. The struture of the United States road network. Technical report, University of New Mexico, 2003.

[57] V. Latora and M. Marchiori. Is the Boston subway a small-world network? *Physica*, A314:109–113, 2002.

[58] E. L. Lehmann. *Theory of Point Estimation*. Wiley, New York, 1983.

[59] J. H. Levine. The sphere of influence. *American Sociological Review*, 37:14–27, 1972.

[60] F. Liljeors, C. R. Edling, L. A. N. Amaral, H. E. Stanley, and Y. øAberg. The web of human sexual contacts. *Nature*, 411:907=908, 2001.

[61] R. D. Luce. Connectivity and generalised cliques in sociometric structure. *Psychometrika*, 15:169–190, 1950.

[62] P. Mariolis. Interlocking directorates and control of corporations: The theory of bank control. *Social Science Quarterly*, 56:425–439, 1975.

[63] M. S. Mizruchi. *The American Corporate Network, 1904-1974*. Sage, Beverley Hills, California, 1982.

[64] M. Molloy and B. Reed. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms*, 6:161–179, 1995.

[65] J. M. Montoya and R. V. Solé. Small world patterns in food webs. *Journal of Theoretical Biology*, 214:405–412, 2002.

[66] J. L. Moreno. *Who Shall Survive?* Beacon House, New York, 1934.

[67] R. L. Moxley and N. F. Moxley. Determining point-centrality in uncontrived social networks. *Sociometry*, 37:122–130, 1974.

[68] T. M. Newcomb. Reciprocity of interpersonal attraction: A nonconfirmation of a plausible hypothesis. *Social Psychology Quarterly*, 42:299–306, 1979.

[69] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences in the USA*, 98:404–409, 2001.

[70] M. E. J. Newman. Random graphs as models of networks. In S. Bornholdt and H. G. Schuster, editors, *Handbook of Graphs and Networks*, pages 35–68. Wiley-VCH, Berlin, 2003.

[71] M. E. J. Newman. The structure and function of complex networks. http://arxiv.org/abs/cond-mat/0303516v1, 2003.

[72] M. E. J. Newman and J. Park. Statistical mechanics of networks. *Physical Review*, E70(066117), 2004.

[73] J. R. Norris. *Markov Chains*. Cambridge University Press, Cambridge, 1997.

[74] T. Nosanchuk. A comparison of several sociometric partitioning techniques. *Sociometry*, 26:112–124, 1963.

[75] J. Park and M. E. J. Newman. Solution of the 2-star model of a network. http://arxiv.org/abs/cond-mat/0405457, 2004.

[76] S. L. Pimm. *Food Webs*. Unviersity of Chicago Press, Chicago, 2nd edition, 2002.

[77] A. Rapoport and W. J. Horvath. A study of a large sociogram. *Behavioral Science*, 6:279–291, 1961.

[78] J. A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, Belmont, California, 2nd edition, 1995.

[79] F. Sampson. *Crisis in a Cloister*. PhD thesis, Department of Sociology, Cornell University, 1969.

[80] P. Sen, S. Dasgupta, A. Chatterjee, P. A. Sreeram, G. Mukherjee, and S. S. Manna. Samll-world properties of the indian railway network. http://arxiv.org/abs/cond-mat/0208535, 2002.

[81] I. Sloane. On-line encyclopedia of integer sequences. http://www.research.att.com/cgi-bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=A000088.

[82] T. A. B. Snijders. Markov chain monte carlo estimation for exponential random graphs. *Journal of Social Structure*, 2(2), 2002. http://stat.gamma.rug.nl/snijders/index.html.

[83] T. A. B. Snijders, P. E. Pattison, G. L. Robins, and M. S. Handcock. New specifications for exponential random graph models. http://stat.gamma.rug.nl/snijders/index.html, 2004.

[84] R. Solomonoff and A. Rapoport. Connectivity of random nets. *Bulletin of Mathematical Biophysics*, 13:107–117, 1951.

[85] J. Stelling, S. Klamt, K. Bettenbrock, S. Schuster, and E. D. Gilles. Metabolic network structure determines the key aspects of functionality and regulation. *Nature*, 420:190–193, 2002.

[86] S. H. Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2001.

[87] A. Vázquez, R. Pastor-Satorras, and A. Vespignani. Large-scale topological and dynamical properties of the Internet. *Physics Review*, E65(066130), 2002.

[88] S. Wasserman and P. Pattison. Logit models and logistic regression for social networks: 1. An introduction to Markov grpahs and p*. *Psychometrika*, 61:401–425, 1996.

[89] D. J. Watts. *Small Worlds*. Princeton University Press, Princeton, 1999.

[90] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[91] H. C. White. Probabilities of homomorphic mappings from multiple graphs. *Journal of Mathematical Psychology*, 16:121–134, 1977.

[92] R. J. Wilson. *Graph Theory*. Longman, Harlow, Essex, 4th edition, 1996.