The behaviour of modularity-optimizing community detection algorithms

Sally Hutchings
St Hugh's College
University of Oxford

A thesis submitted for the degree of
Mathematics and Foundations of Computer Science
1 September 2011

Acknowledgements

I am very grateful to my supervisors, Dr Mason Porter and Dr Raphael Hauser, for their guidance and encouragement. They have tirelessly provided me with their time and resources, and most importantly their insights into this vast field. Thank you also to Peter Mucha, Roger Guimerà and Renaud Lambiotte for helping me to understand their algorithms and write MATLAB programs for them, to Professors Colin McDiarmid and Oliver Riordan for their influential lectures on Graph Theory and Probabilistic Combinatorics which introduced me to the fields of graph theory and network science, and to my supervisor Jonathan Pila for his support. I would not be here without the love and support of my family and friends, and a special thank you goes to Dominic Bowe, Lindsay Munroe, and Anne Hillebrand for a fantastic time this year!

Abstract

Networks can represent many relationships between collections of objects - for example, friendships or geographical proximity between people, or neural connections between brain cells. Communities are a mesoscopic property of a network representing the intuitive concept of a community, such as friendship groups in a friendship network, or towns and villages in a geographical network. Community detection algorithms aim to divide a network into communities using only knowledge of the nodes and the links between them. The results of such algorithms can therefore tell us who belongs to the same community, how many communities there are in a network and even in some cases whether someone belongs to multiple communities. Dividing a network into its underlying communities can also indicate the roles of nodes in the network, such as whether a highly-connected node is connected to many communities, or only to nodes within its own community. Thus community detection can provide deep insights into the structure and resilience of the network, that local properties such as node degree alone cannot.

Results like this motivate the study of communities in networks - but conversely, the strong ties to applications have resulted in fewer mathematical results than in graph theory. In particular, there is little mathematical understanding of the behaviour of community detection algorithms and the statistical significance of their results. This dissertation will focus on the popular method of community detection called modularity-optimization. Currently, modularity-optimizing community detection algorithms are assessed based on their performance on networks with known community structure. However, there are serious concerns about the results of modularity-optimization that suggest that what is more important is knowledge of the *behaviour* of such algorithms. Therefore we will discuss and also address these issues by providing both an up-to-date review of theoretical work on the behaviour of modularity-optimizing community detection algorithms as well as new theoretical results on the subject. The aim is to show that with these theoretical results come new insights into the behaviour of these algorithms and a deeper understanding of the significance of modularity-optimization results.

Contents

Chapter 1

Introduction

Networks are graphical representations of a collection of objects and a type of relationship that holds between them.

Examples of networks are students at Oxford University connected by friendships, or train stations connected if they are only a stop apart. In certain ways, networks are the same as graphs, in the basic sense that they consist of nodes, pairs of which are connected by edges. In this sense one can apply the results of graph theory to gain insight into the network. However, there are vast differences in the structures and properties found in networks and graphs, and the significance of them [23]. These differences have led to the development of different methods to study networks in order to better understand the structures and properties that arise in them. Communities in networks are a prime example of such a structure. An example of a community is a friendship group in a friendship network, for example, or a city in a railway network. That is, the groups we classify as communities in networks all represent communities in the *intuitive* sense of the word: a group of people sharing religion, race, profession, the area in which they live or other characteristics. In a network of people with links representing one of these attributes, such communities will appear as densely connected areas. Thus, to complement this intuitive idea of a community there is a corresponding characterisation of the communities found in networks:

Communities are groups of nodes more densely connected to each other than they are to nodes in the rest of the network.

This is not a *definition* of what we signify as a true community in a network, for there may be both groups we would call communities that are not dense enough to be picked out by this characterisation and groups of nodes we would not call communities (such as cliques arising in random graphs) that would be picked out by this characterisation. Yet groups of nodes more densely connected to each other than they are to nodes in the rest of the network represent groups of induividuals sharing the relationship represented by the edges and thus represent a community in the intuitive sense, therefore in general the two coincide. The aim of community detection in network science is to find such communities.

Note Providing a definition of a community is an issue and the subject of an intense discussion

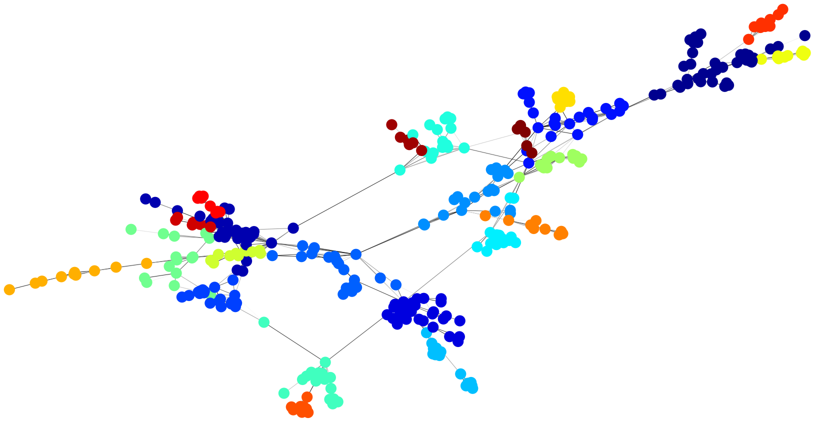

Figure 1.1: Coauthorship network [22] visualised using the Fruchtermann-Reingold algorithm [30, 11], where colours represent communities as determined by Blondel et al's modularity-optimizing community detection algorithm [3].

that we shall not broach here, as the above characterisation is sufficient for the purposes of this dissertation (for more information see Fortunato's paper [9]).

There are many reasons to divide a network into communities. One might want to know how segregated (or integrated) the French and Flemish-speaking people in Belgium are [3, 8], perhaps in order to advise the government on how to address various issues. Or one might need to know how a terrorist network is made up of cells, in order to determine whether an attack was an isolated plan or part of a larger sequence of attacks [17]. Or one might use community detection in conflict resolution to detect nodes that belong to multiple communities and can therefore act as effective negotiators [9]. A very versatile use with many important applications is the classification of node roles, and this we shall consider in more detail.

Example 1. Guimerà and Amaral [15] have developed a node classification scheme based on the results of community detection algorithms. It allows one to say how highly connected or isolated a node is in a network and to determine the sort of function it and others like it may have. While their results can be applied in principle in many areas, Guimerà and Amaral focused on biological networks, and in particular they investigated the classification of nodes in metabolic networks into seven functional roles. The important element is that these classifications are based purely on how the node is connected to nodes in its own community and how it is connected to nodes in other communities.

The roles are defined as follows. The first distinction is between "hubs" and "non-hubs", where a hub is a node highly connected within its community, and the second is between between three different types of hub, "connector" hubs that connect their community to nodes in many other communities, "provincial" hubs that mostly connect to nodes in their own community, and "kinless" hubs whose links are homogenously distributed between communities. This distinction alone has a notable impact on the study of structure and resilience, for the removal of a provincial hub will have much less of an impact on network connectedness than the removal of a connector or kinless hub, even though the nodes may have the same degree. The last distinction is between four non-hub node types, “connector” and “kinless”, which are similar to the above in being connected to other communities or being homogenously connected, and two other roles, “peripheral” and “ultra-peripheral” correspond to being connected mostly or only to nodes in their own community.

In various empirical tests, it was found that these classifications correspond to functional metabolic groups, thus supporting the use of community detection to define node roles. Such a method can be extended to other areas, for example it was noted by Guardiola et al’s [14] that due to the intensely modular structure of the trust network they consider, many links could be removed before intra-community communication was affected; furthermore, the entire network’s communications could be re-established by rebuilding just a few links. Being able to determine which nodes are connector hubs would be very useful in this case.

A consequence of our characterisation of a community in a network is that it suggests how we can go about finding such groups algorithmically. For example, the pioneering article on community detection by Girvan and Newman in 2002 [13] suggested using a *betweenness centrality*, a diagnostic that measures how many shortest paths pass through an edge, since intercommunity edges tend to have higher betweenness values. Having brought community detection to the attention of mathematicians and statistical physicists with this paper, many community detection methods have sprung up in its wake, but few theoretical results to explain their behaviour, justify their use, or increase our understanding of community structure in networks [23]. Modularity-optimization is a particularly popular community detection method, frequently used, and yet it too lacks theoretical support and understanding. Modularity is a partition quality function, measuring how “good” a partition is; that is, how much community structure it suggests a network has compared to what is expected at random.

In Chapter 2, we introduce and define the modularity function, and the principles of its optimization. In Chapter 3, we shall discuss some interesting types of computational heuristics popularly used to optimize it and discuss their performance. Then, in Chapter 4, we shall introduce and explain the problems that beset modularity-optimization, which brings us to the study of the behaviour of modularity-optimizing community detection algorithms. Following this review, we shall discuss two particular problems of interest and present some results of my own. The first of these problems relates to properties of the modularity function and the behaviour of the heuristics on a random graph model, and this is discussed in Chapter 5. The second problem, discussed in Chapter 6, relates to the theoretical properties and behaviour of simulated annealing, one of the heuristics, on a random graph model. Thus my own work on the subject can be found in Chapter 5 and Chapter 6. Lastly, a summary and conclusion can be found in Chapter 7.

Note In this dissertation we consider only undirected, unweighted networks, although many results can be extended to directed, weighted networks. See [9] for more information.

Chapter 2

The modularity function

Modularity is a partition quality function that measures how "good" a network partition is, where in this case the better a partition is the more community structure the network has, when compared with a chosen random graph model called the *null model.* In this section, we follow the development of this intuition into a definition of the partition quality function *modularity*, first introduced and defined by Mark Newman [21].

2.1 Preliminaries

Definition 2. Given a network A, let M be the number of edges, N the number of nodes, $A = (A_{ij})$ the adjacency matrix where $A_{ij} = 1$ if the edge ij is present in the network and 0 otherwise, and d_i the degree of the i^{th} node and k the mean node degree.

Definition 3. By a *partition* σ of a network A we mean an assignment function $\sigma : \{1, \ldots, N\} \longrightarrow \{1, \ldots, r\}$ where $\sigma(i)$ is the class assigned to node i and r is the number of classes in the partition. For ease of notation we can write partitions as vectors of length N, for example, $\sigma = 1, \ldots, N$ for the singleton partition where $\sigma(i) = i$ for each node i.

Definition 4. Given a community of a network A, define k_i to be the number of intra-community edges and k_o the number of edges it has to other communities.

Definition 5. Also define $\delta(i, j) = 1$ if $\sigma(i) = \sigma(j)$, and 0 otherwise.

Definition 6. Let $\mathcal{G}(N, p)$ be the class of *Erdös-Rényi (ER) random graphs* on N nodes with edge probability p. This means that any graph $G(N, p) \in \mathcal{G}(N, p)$ has N nodes and each edge in that graph has probability p of being present, independently of all the other edges.

Definition 7. A *blockmodel* A is a computer-generated network with predefined communities, it was introduced in [13] and as such is sometimes called a Girvan-Newman model. Each edge has probability p_i of being present if it is an intraclass edge or probability p_o otherwise. One can generate them by defining an $N \times N$ probability matrix $P = (P_{ij})$ where each P_{ij} is the probability that the edge ij occurs in the graph and comparing this to a random $N \times N$ matrix $U = (U_{ij})$ whose entries are sampled from the uniform distribution, if $P_{ij} > U_{ij}$ then $A_{ij} = 1$, otherwise $A_{ij} = 0$.

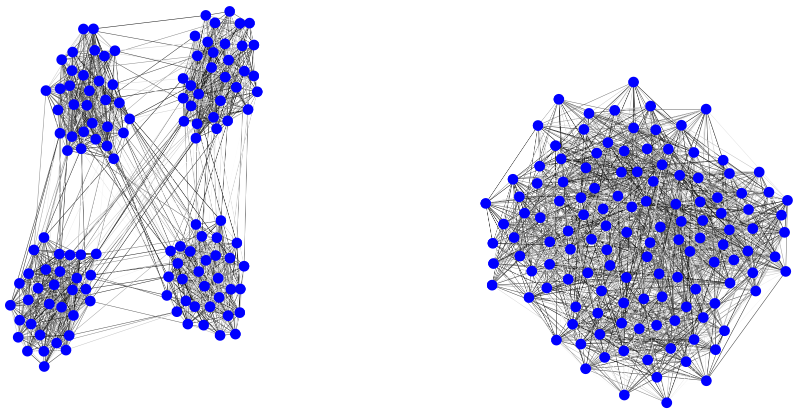

Figure 2.1: Two blockmodels generated using the method given in definition 7. In the network on the left, any intra-community edge has probability $\frac{1}{2}$ and any inter-community has probability $\frac{1}{96}$, wheras in the network on the right every edge has uniform probability of $\frac{16}{128} = \frac{1}{8}$.

Definition 8. The *null model* with respect to a network A is the random graph G on N nodes such that each edge ij has probability $\frac{d_i d_j}{2M}$ of occurring. Thus G has expected degree sequence $d_1, \ldots, d_N$ identical to the network A, but each edge is placed randomly.

Definition 9. Finally, to fix notation, we shall use $<>$ for the *mean* and $\approx$ for *approximately.*

2.2 The mathematical definition

Definition 10. Given a partition σ of a network A , its modularity (as defined in [21] is:

$$Q_\sigma = \frac{1}{2M} \sum_{i,j} \left(A_{ij} - \frac{d_i d_j}{2M} \right) \delta\left(i, j\right) \tag{2.1}$$

where $\sigma, M, N, A_{ij}, d_i$ and $\delta\left(i, j\right)$ are as defined in Definitions 2, 3 and 5.

Definition 11. The matrix $\mathbf{B} = (B_{ij})$ where $B_{ij} = A_{ij} - \frac{d_i d_j}{2M}$ is called the *modularity matrix* [21].

The expectation of an edge can be calculated locally or globally and Definition 10 uses the local expectation, $\frac{d_i d_j}{2M}$, the ratio between the number of ways for such an edge to occur and the total number of edges. Thus modularity compares the presense (or lack thereof) of an edge, A_{ij}, to the expectation $\frac{d_i d_j}{2M}$ of that link being present, for each intraclass edge (each i, j for which $\delta\left(i, j\right) = 1$). Or, equivalently, it compares the intraclass structure of network A with respect to the partition σ to that of the null model (see Definition 8). Thus the null model gets its name from statistics, as it represents what the network would be expected to look like if only randomness was at play in the development of its structure. Therefore modularity represents how “good” a given partition is, where the better a partition is the more unexpected intraclass edges the network has, with respect to the null model. This means that the higher the modularity score, the more a network has unexpectedly high numbers of intraclass edges with respect to that partition. Thus given a network, the partition(s) with the largest modularity scores are the ones with respect to which the network A has the most unexpectedly dense intraclass structure.

Note The factor $\frac{1}{2M}$ scales the modularity score so that it is between -1 and 1 and so that it is 0 when the adjacency matrix is identical to that expected.

The definition of modularity is based on the idea of a community given in Chapter 1, namely that it has more links between its members than expected at random and fewer than expected between those members and the rest of the network. Consider the networks shown in Figure 2.1. The network on the left was generated randomly with each intraclass edge having probability $\frac{1}{2}$ and each interclass edge having probability $\frac{1}{96}$, thus having a mean expected node degree of 16. The network on the right has the same node degrees (on average), but each edge was placed with uniform probability $\frac{1}{8}$.

Evidently any partition of the network on the right will have many more links between groups than the intuitive partition of the network on the left, displaying our intuition that the network on the left has community structure. Furthermore many partitions of the lefthand network will have a larger number of links between groups than the intuitive partition, demonstrating our idea that the intuitive partition is a *good* partition of this network. Modularity allows us to say which partitions are good and bad in this sense of representing the underlying community structure, as well as its maximum value giving us an idea of how *much* community structure a network has.

2.3 Optimization

The aim of modularity-optimizing community detection algorithms, then, is to determine the partition(s) with maximum modularity. However, because we are looking for a maximum over all partitions of a network, the number of which is exponential in N, maximising modularity is equivalent to comparing the modularity of each and every partition of the network to determine the best one, and decision problems like this are often NP-hard. Therefore, this problem could be NP-hard, and in fact it is, as has been shown by Brandes et al [4]. We shall not discuss complexity here, as it only affects my dissertation in as much as it means that there is no polynomial time algorithm that can reliably find $Q_{\max}$ (unless $P = NP$). There are, however, approximation algorithms, and these are the topic of my next chapter.

Chapter 3

Computational heuristics for optimizing modularity

There are a variety of algorithms that attempt to approximate $Q_{\max}$, and to do this reliably and accurately (and quickly, although this is not our concern as we are looking at theoretical properties and for this we will sacrifice concerns for speed and resources) [23, 9]. Four popular classes of modularity-optimizing community detection algorithms are greedy, extremal optimization, simulated annealing, and spectral algorithms. We shall consider one or two of the main competitors from these types. In the rest of this chapter, we discuss their methods, and summarise their achievements assessed according to common practice.

3.1 Methods

3.1.1 Louvain greedy method

Blondel et al's greedy algorithm [3] works by starting with the singleton partition (each node alone in its own class) and assessing potential moves for each node in turn, always choosing the move with the largest increase in modularity. More specifically, one starts with the adjacency matrix A of the network, the singleton partition $G = 1, \ldots, N$ where $G(i)$ is the class of node i, and an ordering of the nodes $\alpha_1, \ldots, \alpha_N$. Then, one by one, each node is assessed by moving it from its current class to a neighbour's class and calculating the change in modularity of each move, selecting as the permanent move the move with maximal increase in modularity, or leaving it in its current class if no such move exists. This last step is then repeated until no further increases in modularity are possible. Note that the result of this algorithm depends on the chosen ordering of the nodes, Blondel et al claim in [3] that test cases suggest the dependence is not significant.

3.1.2 Extremal optimization

Duch & Arenas' extremal optimization algorithm [7] works by recursive bipartitioning, beginning with a random bipartition and using each node's contribution to the modularity as a fitness test,

Algorithm 1 Simulated annealing algorithm [15]

```
$K = 1$
$\sigma = 1 : N$
WHILE $K \leq 25$ OR $Q\left((K-25):K\right) \neq Q\left(K\right)$
|   FOR $i = 1 : fN^2$
|   |   $old = modularity(A, \sigma)$
|   |   Choose a node $x$ uniformly at random
|   |   Choose a different class $a$
|   |   $\sigma' = \sigma$
|   |   $\sigma'(x) = a$
|   |   $new = modularity(A, \sigma')$
|   |   IF $new > old$
|   |   |   $\sigma = \sigma'$
|   |   ELSE $\sigma = \sigma'$ with probability $\exp(-(new - old)/T)$
|   |   ENDIF
|   ENDFOR
|   FOR $i = 1 : fN$
|   |   $old = modularity(A, \sigma)$
|   |   Choose merge or split uniformly at random
|   |   IF merge
|   |   |   Choose a two classes $a$ and $b$
|   |   |   $\sigma' = \sigma$
|   |   |   FOR $x = 1 : N$
|   |   |   |   IF $\sigma'(x) = a$
|   |   |   |   |   $\sigma'(x) = b$
|   |   |   |   ENDIF
|   |   |   ENDFOR
|   |   ELSE Choose a class $a$
|   |   |   $\sigma' = \sigma$
|   |   |   Randomly bipartition $a$ into $a$ and $b$
|   |   ENDIF
|   |   $new = modularity(A, \sigma')$
|   |   IF $new > old$
|   |   |   $\sigma = \sigma'$
|   |   ELSE $\sigma = \sigma'$ with probability $\exp(-(new - old)/T)$
|   |   ENDIF
|   ENDFOR
|   $Q(K) = modularity(A, \sigma)$
|   $T = cT$
|   $K = K + 1$
ENDWHILE
RETURN $Q(K), \sigma$
```

at each step moving the node with the lowest fitness. The fitness function of node i is defined as $q_i = \kappa_{\sigma(i)} - d_i a_{\sigma(i)}$ where $\kappa_{\sigma(i)}$ is the number of neighbours i has in its current class $\sigma(i)$, d_i is the degree of node i and $a_{\sigma(i)}$ is the fraction of edges in the network with at least one end in i's class, so that $Q = \frac{1}{2M}\sum_i q_i$. Thus the q_i are local variables, whose joint optimization results in the optimization of the global variable Q. These local variables q_i can be normalised by dividing through by d_i to get corresponding local variables $\lambda_i = \frac{\kappa_{\sigma(i)}}{d_i} - a_{\sigma(i)}$, where $-1 \leq \lambda_i \leq 1$. Therefore to optimize the global variable Q is to optimize over the local variables λ_i.

The algorithm is based on the process of calculating the λ_i's for every node i and moving the node with the lowest fitness to the other class in the bipartition. Each such move results in an increase in modularity and this process is repeated until no further increase in modularity is possible, at which point the links between the two classes are removed and the whole process is repeated for each class as if it were a network itself. This step is then also repeated also until no increase in modularity is possible.

Note There is also a version of this algorithm where other nodes are selected with a small probability, which aids the algorithm in escaping local maxima, see [7] for more details, we will consider only the original version in order to isolate the study of the principle of using the fitness test as a basis for selecting a node move.

3.1.3 Simulated annealing

Guimerà and Amaral's simulated annealing algorithm [15] is an iterative procedure relying on a temperature T, which decreases with each iteration by a factor c. Each iteration involves $fN^2 + fN$ updates, each of which is accepted with probability 1 if it results in an increase in modularity or otherwise with a small probability, $\exp\left(\frac{\triangle Q}{T}\right)$, where $\triangle Q$ is the old modularity minus the new modularity. This small probability of accepting the move that results in a decrease in modularity is again in order to increase the chance of finding the global maxima. The updates include fN^2 *individual steps* followed by fN *collective steps*. An individual step involves choosing a node and a community to move it to randomly, and a collective step involves randomly choosing two communities to merge, or one community to split. The factor f for the number of updates can be varied depending on results, but $f = 1$ is usually chosen [15]. Parts of this algorithm will be considered in great detail in Chapter 6, so let us set it out more formally. See Algorithm 1 above, with reference to the definitions below.

Definition 12. Let *modularity*(A, σ) be the modularity of the input network A with respect to partition σ, and Q a vector where $Q(K)$ is the modularity at the Kth iteration.

3.1.4 Spectral methods

Newman's spectral algorithm [22] calculates the largest eigenvector of the modularity matrix $\mathbf{B}$ (see Definition 11) and use the entries' signs to bipartition the nodes. Richardson, Mucha and Porter's version [29] uses the *two* largest eigenvectors of $\mathbf{B}$ to bipartition *and* tripartition the nodes, choosing the split with the largest modularity increase.

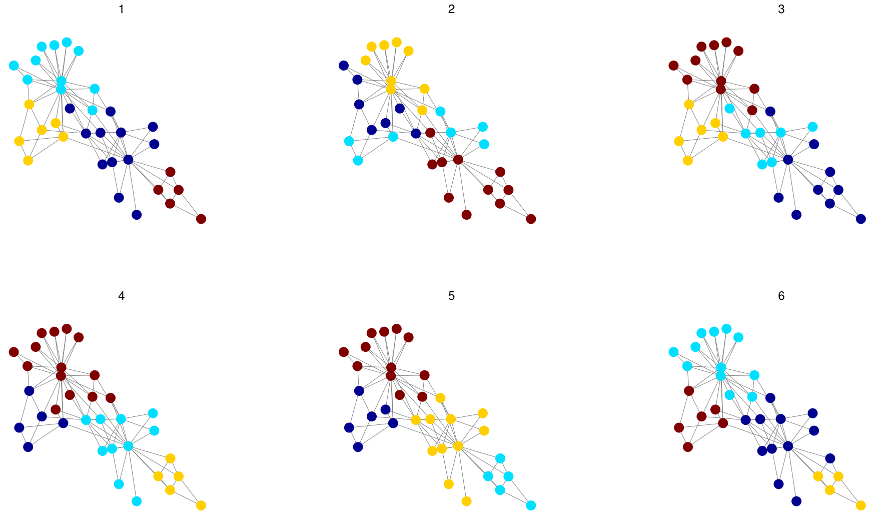

Figure 3.1: The karate club network, visualised using [11] with colours indicating partitions given by (1) the Louvain greedy method [3], (2) extremal optimization [7], (3) simulated annealing [15], (4) spectral method (Newman) [22], (5) spectral method (Richardson et al) [29] and (6) the observed community structure [31] (6).

Let's look at Newman's version first. Let $\mathbf{s} = (s_i)$ where $s_i = 1$ if node i is in class 1 and $s_i = -1$ otherwise (that is, if node i is in class 2). Then we can write modularity as $\mathbf{Q} = \frac{1}{4M}\mathbf{s}^T\mathbf{B}\mathbf{s}$. Alternatively, writing $\mathbf{u}_j$ for the j^{th} eigenvector of $\mathbf{B}$ and β_j for the corresponding eigenvalue so that $\mathbf{s} = \sum_j \left(\mathbf{u}_j^T \cdot \mathbf{s}\right)\mathbf{u}_j$, we can write $\mathbf{Q} = \frac{1}{4M}\sum_j \left(\mathbf{u}_j^T \cdot \mathbf{s}\right)^2 \beta_j$. Maximizing Q is now equivalent to choosing $\mathbf{s}$ so as to concentrate as much weight as possible on the terms involving the largest eigenvalues. Therefore, if β_1 is the largest eigenvalue, we want to choose $\mathbf{s} \propto \mathbf{u}_1$. Of course, because $s_i \in \{-1, 1\}$ this is not always possible, but by choosing the sign of s_i to match that of the corresponding element of $\mathbf{u}_1$, we find the bipartition with maximal modularity.

Richardson et al's spectral algorithm [29] is based on this method but considers the first *two* eigenvectors in order to both bipartition and *tripartition* the nodes, selecting the partition which most increases modularity at each stage.

3.2 Performance

Let us now assess their performance according to the most common practice (not necessarily the best practice), which is to run the algorithm on networks with known community structure, or *benchmarks*, and compare the results of the algorithm with the actual community structure. Such benchmarks include empirical networks as well as artificially generated networks.

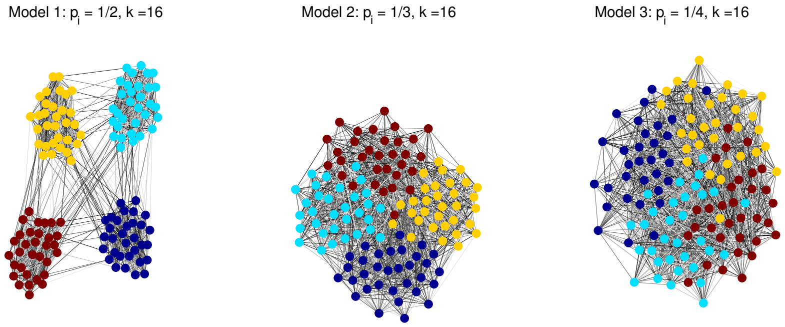
Model 1: p_i = 1/2, k =16
Model 2: p_i = 1/3, k =16
Model 3: p_i = 1/4, k =16

Figure 3.2: Blockmodels for various p_i, all with $N = 128$, mean expected degree $k = 16$ and $r = 4$ communities, giving $p_o = \frac{16-32p_i}{96}$. Note that for model 3 we have $k_i \approx k_o$.

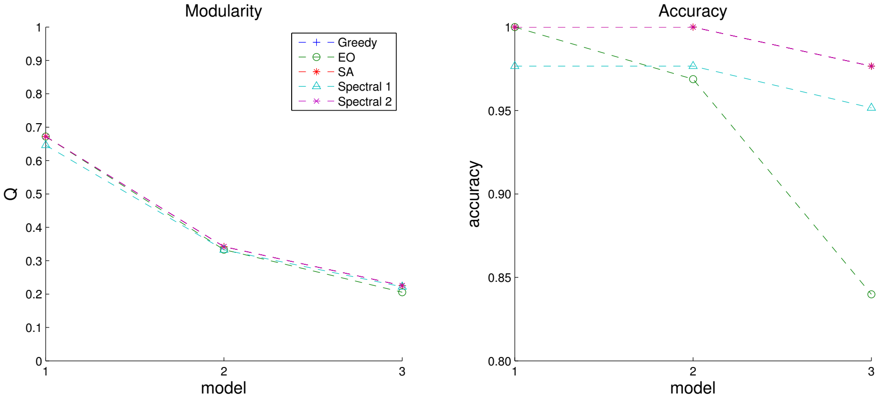

Figure 3.3: Performance of the algorithms on the blockmodels depicted in Figure 3.2.

3.2.1 Benchmarks

Of the many empirical networks used to test algorithms, the coauthorship network [22] in Figure 1.1 and a network called the karate club network [31] have become standard [23]. The karate club network is a very small network of 34 nodes representing alliances between members of a karate club, as observed by a sociologist Zachary [31]. Zachary was observing and recording allegiances in the group when the two leaders argued and the club broke up into communities. Zachary realised that he could have predicted the divisions using the data he had collected, as it provided him with information about the community structure, thus the karate club network has become a standard test of the performance of a community detection algorithm. The coauthorship network depicted in Figure 1.1 is another standard benchmark, as the underlying structure is that of working groups, making it too a good test.

Artificial networks with community structure can also be generated in various ways. A popular method is the generation of blockmodels, proposed by Newman and Girvan [13], where one specifies the groups vector or optimal partition and then sets a higher probability of intraclass links than interclass (see Definition 7). For example, one can define a network on 128 nodes with 4 classes each of 32 nodes and specify an intraclass link probability of $\frac{1}{3}$ and interclass link probability of $\frac{1}{16}$ to get a network with mean expected degree 16 and an instance like that depicted in Figure 3.2. Because in all of these cases the community structure is known, one can compare not only the modularity scores of the results but also the *accuracy.*

Definition 13. Given a partition σ of a network A with known community assignments, the *accuracy* of σ is the proportion of correctly placed nodes.

3.2.2 Comparison of performance

Consider Figure 3.1, which depicts the partitions of the karate club network given by each algorithm compared to the observed community structure. What it shows is that even on such a small network the resulting partitions aren't necessarily going to be alike. While the two spectral algorithms produce partitions very similar to each other and to the optimal one, some disagree with the observed assignment for a large proportion of nodes, such as the extremal optimization partition. For a comparison of modularity scores, consider Figures 3.2 and 3.3, which show that for $< k_i > \leq < k_o >$ (see Definitions 4 and 9) all the algorithms achieve similar modularity scores and perform well (over 80%) in terms of accuracy. Note also that it is hard to differentiate between the results of greedy, spectral (Richardson et al) and simulated annealing, despite their having produced very different partitions in the case of the karate club. If we now look at Figure 3.4, which depicts the mean modularity scores achieved by running each algorithm on ER random graphs (see Definition 6) with $N = 100$ and mean expected degree between 3 and 20, we can see more of a distinction (each line is distinguishable from the others) but again the results are very close.

It is hard to believe, especially after witnessing the differences in partitions in the case of the karate club network, that our conclusion should be that we can use any algorithm we choose and get fairly accurate, reliable results, but what more can this sort of computation tell us? There is good cause to be sceptical, for the algorithms each work very differently to each other, and little is

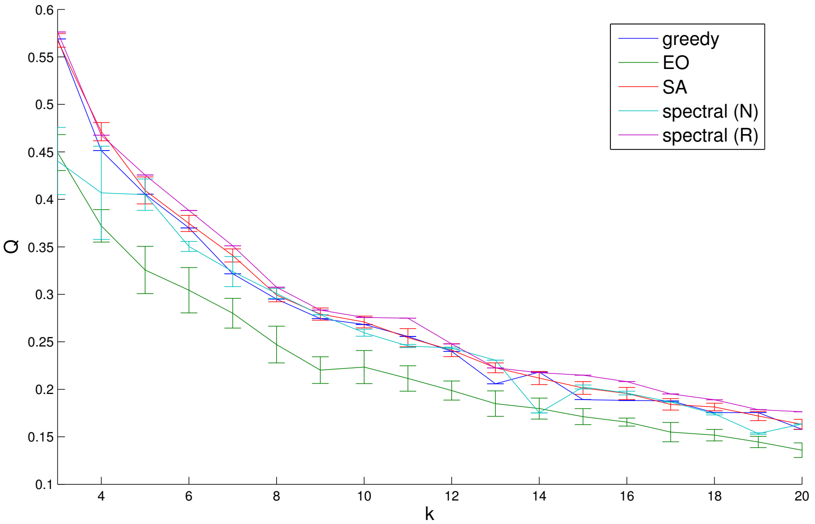

Figure 3.4: Results of each algorithm on ER random graphs with mean degree $3 \leq k \leq 20$ for $N = 100$, averaged over 20 runs. Bars indicate standard deviation.

known theoretically about the impact of these methods on the resulting partitions [23]. Of course, we could make many more such investigations in this manner, comparing resulting partitions for whole swathes of blockmodels and random graphs and empirical benchmarks, but what such an investigation would be searching for is theories on the *behaviour* of these algorithms, for that is what will tell us which algorithms are doing what we want them to do and as a consequence whether a given algorithm truly is reliable and accurate. Therefore it is the behaviour of these algorithms that is the better subject of investigation, and the subject to which we now turn.

Chapter 4

Issues with modularity optimization

It might seem from the tests performed above that the algorithms are consistently performing “well”; that is, on the small benchmarks discussed above they are providing partitions and modularity scores close to those of the optimal ones. However, we also know that optimizing modularity is NP-hard [4], so we know that all these algorithms can do is *approximate* the optimal solution - they are unlikely to be consistently and reliably providing optimal results. In fact, unless $P = NP$, they *cannot* be doing this, for they are polynomial time algorithms approximating a solution to an NP-hard problem.

Because we are dealing with such a hard problem, it might seem that the only way to assess these algorithms is to analyse their performance on benchmarks, but this is not the case, and there have been several successful investigations into the behaviour of modularity-optimization algorithms [12, 16, 24, 29, 9]. Some of these are negative results and some positive, but all provide deeper understanding of the modularity function, the optimization algorithms, and their results. First we shall discuss some very worrying theoretical results that have been presented recently, such as the *extreme near-degeneracy* and *inconsistencies* of the modularity function as well as the existence of a *resolution limit*. Then we shall discuss a positive result that has shone through the negative, namely Richardon et al’s investigation of the behaviour of spectral modularity-optimization algorithms.

4.1 Extreme near-degeneracy

Good et al [12] have shown that not only does the number of partitions increase exponentially with the size of the network, but the number of *near-optimal solutions* (solutions within just a few percentage points of the optimal solution) grows exponentially with the number of communities in the graph. This means, counter-intuitively, that the more modular a network is (the more communities it has), the harder it is to find the optimal partition (the partition into these communities). The reason for this is that the more communities there are, the more ways there are to merge just two of them.

In the class of networks consisting of r sparsely interconnected modules with edge-densities of roughly $\frac{2M}{r}$, the resulting loss in modularity from merging two of these communities is just

$\triangle Q_r = -2r^{-2}$ where r is the number of communities in the network. This value is very small for even moderate r; as Good et al point out, for a network with just 20 communities, the penalty for merging two of them is just 0.005. This means that even partitions with modularity within just 1% of optimal, ones we would certainly call *near optimal*, can be far from optimal in terms of accuracy (recall Definition 13) as they have merged two communities. That is, they are not sub-optimal due merely to the misplacement of a single node, but perhaps as much as a whole misplaced *community.*

The number of such near-optimal solutions is bounded *below* by 2^{r-1}, a huge value in the case above where $r = 20$. Thus even for moderate r our confidence in near-optimal modularity scores must be greatly reduced. For even if they are just 0.005 short of the optimal modularity score, the partition giving them may be quite different from the optimal one. Clearly, then, this is a serious problem that greatly affects our treatment of the modularity scores and partitions that modularity-optimizing algorithms provide us with. Although a negative result, it is therefore a very significant one in the field, and one that should be taken very seriously. That is, unless it can be *shown* that the partition that goes with the near-optimal score is also near-optimal, then it should not be trusted to be all that similar.

4.2 Inconsistencies

Bickel and Chen [2] investigated the asymptotic behaviour of modularity on a random graph model defined as follows. There are K unknown communities, a potentially infinite number of nodes, and a $K \times K$ edge probability matrix $\mathbf{P} = (P_{ab})$ such that P_{ab} is the probability of an edge ij given that i is in community a and j is in community b. Thus the case for which $K = 1$ is the class of ER random graphs of Definition 6. Defining *consistent* to mean identifying the members of each community perfectly, Bickel and Chen show that on this model the modularity function, as maximised in [21], is not always consistent. In fact they show via counterexample that even in the subset of cases that have more intracommunity edges than intercommunity edges ($k_i > k_o$) it is not always consistent. The problem in the counterexample they give is that two small communities are merged, and this is a problem with the modularity function that we shall proceed to discuss in the next section.

4.3 Resolution limit

From our characterisation of a community in a network and our definition of modularity as seeming to pick out exactly these structures, it is worrying to learn from Fortunato and Barthélemy [10] that communities smaller than a certain size, depending on the number of edges in the network, M, will not be distinguished. This is called a *resolution limit.* As Good et al [12] explain, it arises due to the fact that the change in modularity from merging two classes in a partition is given by the equation:

$$\triangle Q = \frac{E_{ij}}{M} - 2\frac{D_i}{2M}\frac{D_j}{2M} \tag{4.1}$$

where E_{ij} is the number of edges with one end in class i and one in class j and D_i is the sum of the degrees of nodes in class i. This means that two classes are merged if and only if $E_{ij} > \frac{D_i D_j}{2M}$.

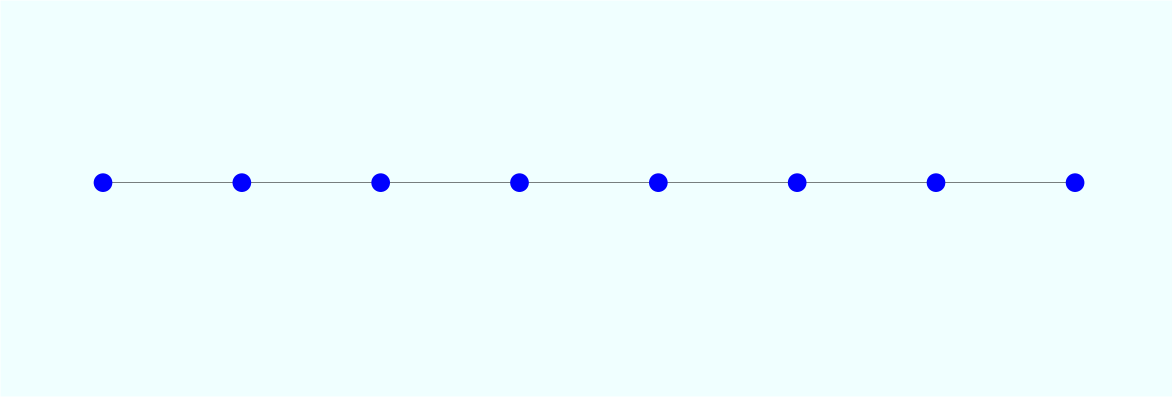

Figure 4.1: Bucket brigade network with 8 nodes.

Problems arise because properties of the null model (recall Definition 8) mean that modularity tends to *expect* E_{ij} to be less than 1, making even a single edge between two modules unexpected, thus merging two communities even though there is just one edge between them.

The good news is that certain sorts of algorithms can circumvent this problem, ones involving recursive partitioning in particular, such as the spectral algorithms described in Chapter 3 and multi-resolution models (see [9]). The reason the resolution limit is circumvented is that at each stage the network is divided into smaller "sub-networks", thus reducing the resolution limit at each stage. Therefore this negative result on the behaviour of modularity has shone light on the advantages of certain methods, such as spectral algorithms, showing them to be behaving in a desirable way and thus giving us results closer to those we desire (the optimal partition).

4.4 Spectral behaviour

A development in research on the behaviour of modularity-optimizing community detection algorithms is Richardon et al's work on bucket brigade networks [29]. When Newman introduced his spectral method [22] he noted that it had a fault, a fault which can be seen in the consideration of a theoretical case called a "bucket brigade network". A bucket brigade network is a chain of nodes each connected only to the one before it and the one after, such as the 8-node one pictured in Figure 4.1. Due to the size of the network, the optimal partition can be determined by exhaustive comparison of all partitions, and it is found that it consists in 3 classes. The recursive bipartitioning of Newman's spectral method partitions this network into 4 groups and therefore fails to determine the optimimal partition [29]. The solution it provides is rather different to that of the optimal partition, since it divides the bucket brigade into 4 equally sized groups of 2, whereas the optimal partition consists in two groups of 3 and one of 2 (the central 2 nodes) [22, 29].

To combat this issue, Richardson et al suggest consideration of the first *two* eigenvectors in order to consider both bipartitions and tripartitions, which extends the options considered and avoids this problem by finding the tripartition in the initial stage, and choosing it over the bipartition. This extension goes a long way in opening up the options for the spectral method, but even this version has limitations. For example, at each stage the algorithm has to decide whether to go with the bipartition or the tripartition so there are cases it too will miss. The example considered in [29] is a 20-node bucket brigade, where the optimal partition consists of 4 groups. At the initial stage, the bipartition has lower modularity than the tripartition, so the tripartition is chosen and the option to split the network into 4 groups at the next stage is missed. The results of Richardon et al's extension of Newman's algorithm is actually inferior to those of Newman's in this case, even though Richardon et al's has more tools and considers more options.

4.5 Progress

In summary, there are several serious problems in the field, including extreme near-degeneracy, a resolution limit, inconsistencies, and limitations of recursive bipartitioning (and tripartitioning). These issues create a certain amount of distrust in the results of modularity-optimizing community detection algorithms, but so far little progress in the way of deepening theoretical understanding has been made. Instead, current best practice in the use of such algorithms is to run different types and compare the results for consistent structural properties. This is not to say that progress has not been made, as we have seen investigations have been made into the *behaviour* of the function and algorithms optimizing it, results including that recursive algorithms avoid the resolution limit and that including tripartitioning in spectral algorithms takes us a lot closer to optimal results. These results show how much there is to be gained from a theoretical investigation into the behaviour of modularity and modularity-optimizing algorithms.

This consideration of the theoretical properties, and theoretically interesting networks such as the bucket brigade, results in a deeper understanding of the limitations and restrictions on spectral methods, and could provide similar insight in other cases. Results like this create interest in finding further ways to develop this algorithm in ways that help it to avoid these pitfalls rather than merely run multiple algorithms to compare results. Theoretical results take us beyond the practical results, such as the occassional case of a surprisingly low modularity score, to bigger developments in the field, such as a theoretically justified improvement in the case of Richardson et al's work. More than this, to present trustworthy results one needs to understand that the new version also has limitations and Richardson et al have provided us with an understanding of these also.

Moving forward, we turn to a discussion of my own results in this area, in particular, work on the modularity of ER random graphs, presented next in Chapter 5, and work on simulated annealing's behaviour on ER random graphs, presented in Chapter 6.

Chapter 5

Modularity of random graphs

In this chapter, we investigate modularity in random graphs. From the definition of modularity, given in Section 2.2, one can quite reasonably expect the modularity of a random graph to be zero, but natural fluctuations in the distribution of the edges mean that many particular instances of a random graph will have strictly positive modularity scores [9]. This means that random graphs can appear to have significant community structure by achieving strictly positive modularity scores. To counter this, we need to know what maximum modularity scores random graphs are likely to produce, and then adjust our rubric for what we call *significant* accordingly. The typical maximum modularity of a random graph was independently investigated by Reichardt and Bornholdt using spin-glass theory [24, 25, 26, 27, 28], and Guimerà, Sales-Pardo, and Amaral using combinatorial arguments [16]. Both did this by looking at the sort of *partitions* one can expect in an ER random graph and then looking within that group of partitions for the one with the largest modularity.

Knowing the expected maximum modularity of a random graph tells us that any networks whose modularity scores are equal to or lower than this value are not likely to contain structure that couldn't arise randomly; that is, any seeming community structure contained within could have arisen randomly and is therefore not significant. What such results about expectation cannot tell us is what *does* count as significant, for without knowing the variance, we do not know how frequently partitions with larger modularities than expected occur. If they occur very frequently, then a network achieving modularities larger than expected may still not have significant enough structure for one to conclude that it is unlikely for it to have occured at random. The work of Reichardt and Bornholdt, and Guimerà, Sales-Pardo and Amaral on the expected maximum modularity of null models is the extent of work done in this area, so there is an issue here that has not been addressed in the literature to date. That is, there has not as of yet been an investigation into the variance, only a consideration of the expectation, and this is an issue that should be addressed.

The structure of this chapter is as follows. First, there is an assessment the work of both Reichardt and Bornholdt and Guimerà et al and a comparison of their results. Following this, a presentation of my own work, involving an extension of Guimerà et al's work to consider also the variance, thus addressing the issue of being able to say what *is* significant.

Both Guimerà et al and Reichardt and Bornholdt chose to consider the Erdös-Rényi (ER) random graph model, so let $\mathcal{G}(N, p)$ be the class of Erdös-Rényi (ER) random graphs as in Definition

6. On average graphs in this class have $M = p\binom{N}{2}$ edges and each node has on average $k = p(N-1)$ neighbours, or pN asymptotically. We will look in particular at ER random graphs for which the mean degree k is fixed, giving $p = \frac{k}{N}$.

5.1 Spin-glass models

Reichardt and Bornholdt showed that optimizing modularity is equivalent to finding the ground state of an equivalent spin-glass model [25]. They use this equivalence, together with spin-glass theory, to show various properties of modularity maxima of ER random graphs. For example, they show that the partition of an ER random graph with maximal modularity will be an equipartition [25], so that only the modularity of equipartitions of random graphs need be considered in order to predict the typical modularity maximum. They then use these results to provide theoretical results on the typical modularity of a random graph.

Reichardt and Bornholdt conclude that the maximum modularity of an ER random graph is typically $Q_{ER}(N,p) = 0.97\sqrt{\frac{1-p}{pN}}$. This appears to be a fairly accurate prediction when compared with the modularity score results of running simulated annealing on sample ER random graphs [25]. An interesting conclusion of Reichardt and Bornholdt [26] is that modularity scores of a certain value being expected at random explains why algorithms are notably less accurate when there are less intraclass edges than interclass (see Figure 3.3). For, they explained, this limit represents the point at which the mean number of edges a node has into its own community equals that it has to other communities. The reason that the algorithms are less accurate is that only when there are more intraclass than interclass links do we have modularity greater than that expected in a random graph with similar size and number of edges. That is, only when there are more edges from a node into its community than between it and other communities is there more structure than expected in a random graph with similar size and number of edges.

Thus being able to say what is not a significant modularity score is is not the only result of considering the typical modularity of ER random graphs, it can also explain other phenomena, such as the cut-off point observed in Figure 3.3. However, Reichardt and Bornholdt have not gone as far as to consider the variance. We now consider Guimerà et al's work, which uses some of Reichardt and Bornholdt's results (namely the one that shows the optimal partition to be an equipartition), but works in a combinatorial environment that allows one to extend their work to consider the variance.

5.2 Combinatorial arguments

Guimerà et al study the behaviour of the number of equipartitions as a function of the number of nodes N, the edge probability p, the number of classes r, and the number of intraclass edges k_i. The number of such equipartitions is a random variable and as such one can look at its expected value *and* its variance. This makes it very useful when considering the modularity of a random graph, as the modularity of such an equipartition can be written in terms of N, p, r and k_i as $Q(N,p,r,k_i) = \frac{2rk_i}{N^2p} - \frac{1}{r}$ [16]. Thus, to find the *maximum* expected modularity with respect to N

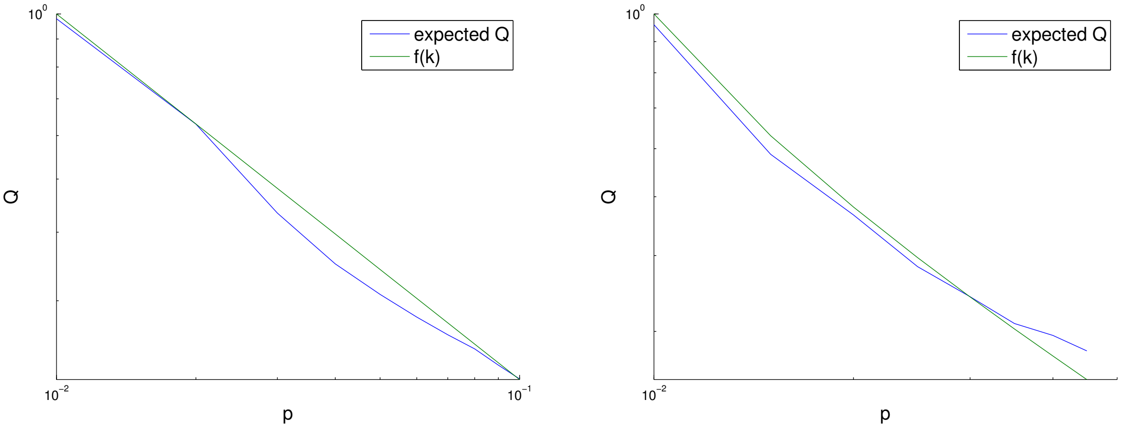

Figure 5.1: Logarithmic scale plot showing that $\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]$ is proportional to $f(k)=k^{-\frac{2}{3}}$ for $N=100$ (left) and $N=150$ (right)

and p only, denote it $Q_{max}(N,p)$, one wants to optimize $Q(N,p,r,k_i)$ with respect to r and k_i. Let us consider a more formal presentation of their work. A presentation that follows the traditional methods of calculating the expectation of a random variable, while preserving their results and equations, will be beneficial for the purposes of clarity and consistency.

Definition 14. Let $\mathcal{Z}\left(N,p,r,k_i\right)$ be the number of equipartitions of an ER random graph $G(N,p)$ with r classes of $n=\frac{N}{r}$ nodes and exactly k_i intraclass edges. Thus $\mathcal{Z}\left(N,p,r,k_i\right)$ is a random variable.

Definition 15. Given an equipartition partition σ of $G(N,p)$ (with r classes of roughly $n=\frac{N}{r}$ nodes), call it *valid* if each class has exactly k_i intraclass edges.

Note Each class in a valid equipartition will have roughly $k_o=\frac{N^2p}{r}-2k_i$ interclass edges, as each class has k_i intraclass edges and $\frac{N^2p}{r}$ is the expected number of edges shared by nodes in two classes.

To find the typical modularity, one would find the most typical equipartition (given r), but to find the *maximum* expected modularity, one wants to refine this partition, achieving larger and larger k_i and also larger and larger Q. Therefore, one must find an equipartition that is still typically expected but also has the largest possible value of k_i. This involves solving $\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]=1$, as such an equipartition is typically expected but also the least expected of the typically expected partitions, and thus the most refined (that is has the largest k_i, and thus largest modularity). This value of k_i will be a function of N, p and r, so one then wants to maximise over the pairs r and k_i using the equation $Q\left(N,p,r,k_i\right)=\frac{2rk_i}{N^2p}-\frac{1}{r}$ to find the optimal pair, call them r^* and k_i^*.

The first step to finding the typical modularity of an ER random graph, then, is to calculate the expected number of equipartitions $\mathcal{Z}\left(N,p,r,k_i\right)$. Now $\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]=\sum_\sigma \mathbb{P}\left(\sigma \text{ valid}\right)$ where σ is an equipartition as defined above. Given σ, by considering each class one at a time, we have

$$\mathbb{P}\left(\sigma \text{ valid}\right)=\prod_{t=1}^{r}\binom{\binom{n}{2}}{k_i}p^{k_i}\left(1-p\right)^{\binom{n}{2}-k_i}\binom{n\left(N-n\right)}{k_o\alpha_t}p^{k_o\alpha_t}\left(1-p\right)^{n(N-n)-k_o\alpha_t}. \tag{5.1}$$

Therefore,

$$\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]=\prod_{t=1}^{r}\binom{N-(t-1)\,n}{n}P_i\left(N,p,r,k_i\right)P_o\left(N,p,r,k_o\alpha_t\right), \tag{5.2}$$

where

$$P_i\left(N,p,r,k_i\right)=\binom{\binom{n}{2}}{k_i}p^{k_i}\left(1-p\right)^{\binom{n}{2}-k_i}, \tag{5.3}$$

$$P_o\left(N,p,r,k_o\alpha_t\right)=\binom{n\left(N-n\right)}{k_o\alpha_t}p^{k_o\alpha_t}\left(1-p\right)^{n(N-n)-k_o\alpha_t} \tag{5.4}$$

and lastly $\alpha_t=\frac{r-t}{r-1}$ is the proportion of classes left to be considered.

Thus, given N and $p=\frac{k}{N}$ where k is the mean degree, one can compute the expected maximum modularity of an ER random graph by taking the above equation, solving it equal to 1 and maximising over r to find r^* and k_i^*. This will give us an upper bound for the modularity that occurs on average at random, which tells us *not* to consider modularity scores less than or equal to this value to be significant.

Guimerà et al solved numerically $\mathbb{E}\mathcal{Z}\left(N,p,r,k_i\right)=1$ and maximised over r for various N and p. See Appendix A for an analytical evaluation of this equation. Using logarithmic scale plots they showed that $Q_{max}(N,p)$ is proportional to $f(k)=k^{-\frac{2}{3}}=(pN)^{-\frac{2}{3}}$ [16]. See figure 1 in their article for a demonstration of these findings, as well as Figure 5.1, which is plotted from the results of my own programmes based on these equations. Taking into account the percolation point at $Np=k=2$ they conclude that the function that best predicts $Q_{max}(N,p)$ is $F(N,p)=\left(1-\frac{2}{\sqrt{N}}\right)\left(\frac{2}{Np}\right)^{\frac{2}{3}}$ [16]. These results suggest that any network with N nodes and mean degree k (and similar degree distribution i.e. not power law) that has modularity less than or equal to $F\left(N,\frac{k}{N}\right)$ has no more a modular structure than the average ER random graph, thus saving the large amount of computational time required to directly compute $Q\left(N,p,r,k_i\right)$, and optimize over r and k_i, providing us with a convenient theoretical result about modularity and random graphs.

Furthermore, this value is large for k small, which means that even the seeming community structure of networks with large modularity scores could have arisen at random in cases where the mean degree of the network is small. My own work included generating sample ER random graphs and running the selection of algorithms discussed in Chapter 3 on them to determine their modularity, to compare with this prediction. The results of this work can be seen in Figure 5.2, which shows that the prediction is so close to the actual modularity scores as to almost underestimate the modularity for larger k.

Note that Reichardt and Bornholdt's prediction differs from Guimerà et al's, not merely in coefficient, but in the function of N and p. For Reichardt and Bornholdt claim the expected modularity maxima to be $Q_{ER}=0.97\sqrt{\frac{1-p}{pN}}$ and Guimerà et al claim that $Q_{max}(N,p)$ is best predicted by $F(N,p)=\left(1-\frac{2}{\sqrt{N}}\right)\left(\frac{2}{Np}\right)^{\frac{2}{3}}$. The reason for the difference is that Guimerà et al derived their equation using a logarithmic scale plot and measuring the gradient to determine $Q_{max}(N,p)$ as a function of N and p. This results in a fairly accurate prediction but it doesn't mean that they will arrive at the same function as an analytical investigation or one that imports

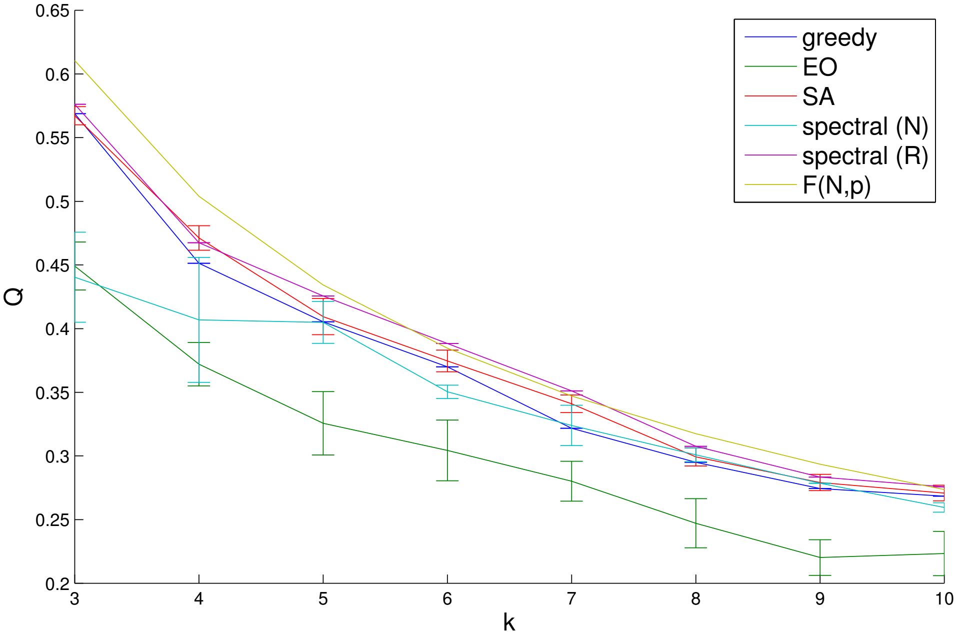

Figure 5.2: Comparison of the prediction $F(N, p)$ and the modularity results of the algorithms. Bars indicate standard deviation.

from the results of another field, such as spin-glass theory. Both functions are fairly accurate in predicting expected modularity maxima in ER random graphs [25, 16], and the reason this can happen is that they are in fact very similar to each other. Consider Figure 5.3 which compares these functions and shows them to be within 0.02 of each other for $N = 100$ and $N = 150$.

5.3 Variance

As discussed above, what we can't say without knowing the variance in modularity is how much above this value a result needs to be in order to say that the network does have significant community structure. In order to determine the variance in modularity, I calculated the variance in the underlying random variable $\mathcal{Z}(N, p, r, k_i)$. The reason for this is that it is the variance in $\mathcal{Z}(N, p, r, k_i)$ that will determine the possible choices of k_i^* and as a consequence perhaps even the choice of r^*. If there is large variance in the most refined partition present in a random graph instances $G(N, p)$, then there will be large variance in occuring modularities. Once we have an equation for the variance in $\mathcal{Z}(N, p, r, k_i)$, we can consider the largest *frequently-occurring* k_i^* as opposed to the typical one and thus consider the largest frequently-occurring modularities. This will allow us to say, given a network with a modularity score Q, whether such scores frequently occur in ER random graphs (and are thus likely to occur at random) or whether they do not. If they do not then the score is unlikely to have been caused by randomness and thus indicates significant community structure. Hence, calculating an equation for the variance in $\mathcal{Z}(N, p, r, k_i)$ will lead us

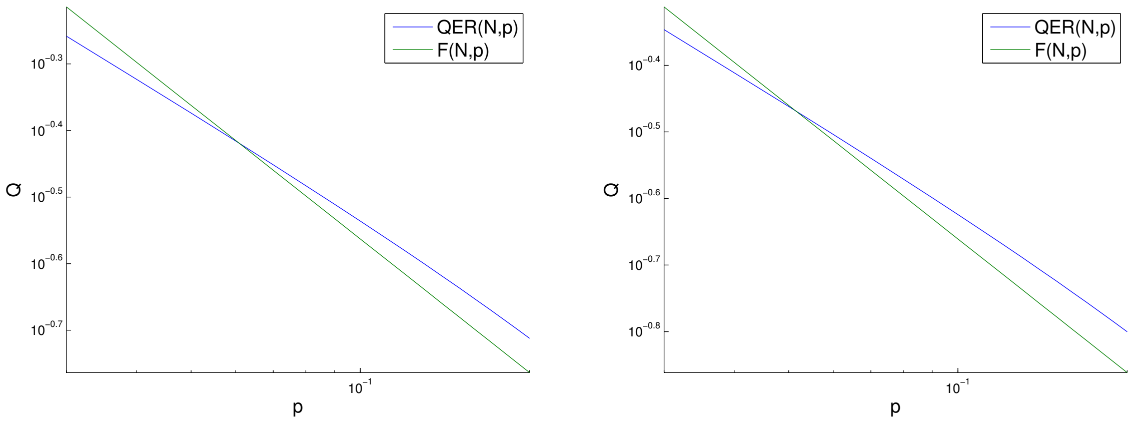

Figure 5.3: Graphs comparing $Q_{ER} = 0.97\sqrt{\frac{1-p}{pN}}$ to the $Q_{max}(N,p)$ prediction $F(N,p) = \left(1 - \frac{2}{\sqrt{N}}\right)\left(\frac{2}{Np}\right)^{\frac{2}{3}}$ for $N = 100$ and $N = 150$ respectively.

to a solution to the problem of being able to say what *is* a significant modularity score, as opposed to only what is not significant. Thus in this section we shall perform the first calculation of the variance of the random variable and make the first assessment of the most frequently-occurring maximum modularity in ER random graphs, as opposed to considering only the mean maximum modularity.

The variance of the random variable $\mathcal{Z}(N,p,r,k_i)$ is given by

$$\mathrm{var}\left[\mathcal{Z}(N,p,r,k_i)\right] = \mathbb{E}\left[\mathcal{Z}(N,p,r,k_i)^2\right] - \mathbb{E}\left[\mathcal{Z}(N,p,r,k_i)\right]^2 \tag{5.5}$$

where for σ, τ equipartitions we have

$$\mathbb{E}\left[\mathcal{Z}(N,p,r,k_i)^2\right] = \sum_{\sigma}\sum_{\tau}\mathbb{P}\left(\sigma,\tau\ \textbf{both}\ \text{valid}\right) = \sum_{\sigma}\sum_{\tau}\mathbb{P}\left(\sigma\ \text{valid}\right)\mathbb{P}\left(\tau\ \text{valid}|\sigma\ \text{valid}\right) \tag{5.6}$$

Now $\mathbb{P}(\sigma\ \text{valid})$ will be as above, but $\mathbb{P}(\tau\ \text{valid}|\sigma\ \text{valid})$ will depend on how the partitions overlap. Hence to calculate $\mathrm{var}\left[\mathcal{Z}(N,p,r,k_i)\right]$, we need to calculate $\mathbb{P}(\tau\ \text{valid}|\sigma\ \text{valid})$.

All we need to derive a formula for the variance, then, is to derive a formula for $\mathbb{P}(\tau\ \text{valid}|\sigma\ \text{valid})$. For this we need to consider how knowing that the first partition σ is valid affects the probability that the second partition τ is also valid. Being valid is a global property of the graph (it says something about every node and edge, not only a small number of them) which makes the process of calculation quite complex. First we shall need a way of talking about partition overlap, and for this we shall extend the notation Achlioptas uses in his discussion of equipartitions with respect to k-coloring [1].

Definition 16. Let the matrix $\mathbf{L} = (l_{ij})$ where $1 \leq i,j \leq r$ and l_{ij} is the number of vertices in class i in partition σ and class j in partition τ [1]. Also define the matrix $\mathbf{P} = (P_{uv})$ for each pair of nodes u, v such that P_{uv} is the probability of edge uv occuring, *given that partition σ is valid.*

Then:

$$P_{uv} = \begin{cases} \frac{k_i}{\binom{n}{2}} & \text{if } \sigma(u) = \sigma(v), \\ \frac{k_o}{n(N-n)} & \text{otherwise,} \end{cases} \tag{5.7}$$

because edges are uniformly distributed within classes and between classes in σ, as we are in the graph class $\mathcal{G}(N, p)$.

Definition 17. For ease of notation, let $p_1 = \frac{k_i}{\binom{n}{2}}$ and $p_2 = \frac{k_o}{n(N-n)}$.

To make sure our intuition is correct at each stage of development of our formula, let us consider a specific case and create general formulas from our observations of this case.

5.3.1 Deriving the variance

For the example, let us take $N = 8, r = 2, n = 4, k_i = 2, k_o = 2$ and equipartitions σ and τ as illustrated in Figure 5.4. We know that $\mathbb{P}(\tau \text{ valid}|\sigma \text{ valid})$ is the probability that each class in τ has k_i intraclass edges and k_o interclass edges, given that we know that each class in σ has k_i intraclass edges and k_o interclass edges. In this case, $l_{ij} = 2$ for all $1 \leq i, j \leq 2$, so $\mathbf{L} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$.

Intraclass edges Let us consider class 1 of τ first and the probability it has k_i intraclass edges. When we choose k_i intraclass edges of the $\binom{n}{2}$ possible, the probability that exactly those k_i edges are there isn't $p^{k_i}(1-p)^{\binom{n}{2}-k_i}$ any more, because the probability of each edge uv isn't uniformly p but depends on whether it's an interclass or intraclass edge in the partition σ. Thus, we can characterise the probability of there being exactly k_i edges by summing over choices of k_i potential edges the number of times that scenario occurs multiplied by the probability of it occuring. The total number of possible edges that are intraclass edges in class 1 of τ is $\binom{n}{2}$, and each edge is either intraclass or interclass in σ. We want to consider the various scenarios as to whether the edges are intraclass or interclass in σ. Of the intraclass edges in class 1 of τ, there are $\sum_j \binom{l_{1j}}{2}$ edges that are intraclass in σ, because this is the number of ways to choose pairs of nodes that are in the same class in σ. Similarly, there are $\sum_j \sum_{k \neq j} \frac{l_{1j} l_{1k}}{2}$ edges that are interclass in σ, because this is the number of ways to choose pairs of nodes that are in different classes in σ.

Because $k_i = 2$, our only options are that either both edges are intraclass edges in σ, both are interclass in σ edges, or there is one of each. Let x be the number of ways the first scenario can occur, y the second, and z the third. From observation, $x = 1$, as there is only one way to choose 2 edges such that both edges are within classes in σ. We can generalise these observations by characterising them in terms of the matrix $\mathbf{L}$. This gives

$$x = \binom{\sum_j \binom{l_{1j}}{2}}{2}. \tag{5.8}$$

The probability of an instance of the first scenario occuring (i.e. the probability that the k_i intraclass

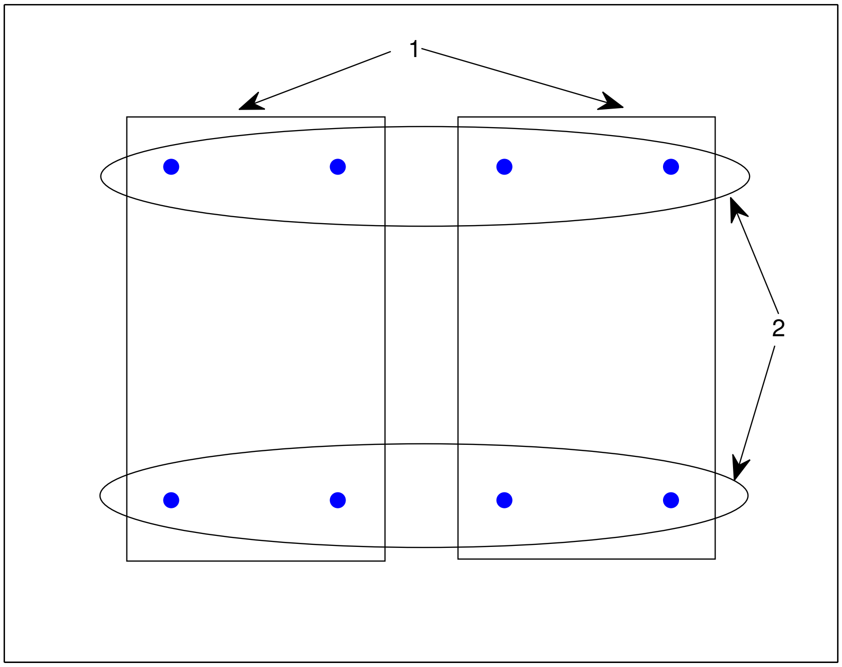

Figure 5.4: Example of two 8-node partitions, σ and τ (partitions 1 and 2, respectively).

edges in σ chosen are exactly the edges that are present) is

$$\left(\frac{k_i}{\binom{n}{2}}\right)^2 \left(1-\frac{k_i}{\binom{n}{2}}\right)^{\sum_j \binom{l_{1j}}{2}-2} \left(1-\frac{k_o}{n\,(N-n)}\right)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}}, \tag{5.9}$$

because there are 2 edges for which $P_{uv} = \frac{k_i}{\binom{n}{2}}$ and the rest are such that $P_{uv} = \frac{k_o}{n(N-n)}$. To put this more simply Equation 5.9 is equal to

$$p_1^2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}-2}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}}\,. \tag{5.10}$$

Similarly, from observation, $y = 4$ and $z = 8$. In terms of the matrix $\mathcal{L}$ we have

$$y = \binom{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}}{2}, \tag{5.11}$$

and

$$z = \left(\sum_j \binom{l_{1j}}{2}\right)\left(\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}\right). \tag{5.12}$$

Similarly the probability of the second scenario is

$$p_2^2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}-2}, \tag{5.13}$$

and the probability of the third scenario is

$$p_1p_2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}-1}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}-1}\,. \tag{5.14}$$

Hence, the probability that there are exactly k_i edges in class 1 of τ, given the fact that σ is valid, is

$$\begin{aligned} x\left(p_1^2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}-2}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}}\right) + y\left(p_2^2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}-2}\right) \\ +z\left(p_1p_2\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}-1}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}-1}\right). \end{aligned} \tag{5.15}$$

More generally, the probability that there are exactly k_i edges in class 1 of τ, given the fact that σ is valid, is

$$\sum_{s=0}^{k_i} \binom{\sum_j \binom{l_{1j}}{2}}{s}\binom{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}}{k_i-s} p_1^s p_2^{k_i-s}\,(1-p_1)^{\sum_j \binom{l_{1j}}{2}-s}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{1j}l_{1k}}{2}-(k_i-s)}. \tag{5.16}$$

Similarly, the probability that there are exactly k_i edges in class 2 of τ, given the fact that σ is valid, is

$$\sum_{s=0}^{k_i} \binom{\sum_j \binom{l_{2j}}{2}}{s}\binom{\sum_j \sum_{k\neq j} \frac{l_{2j}l_{2k}}{2}}{k_i-s} p_1^s p_2^{k_i-s}\,(1-p_1)^{\sum_j \binom{l_{2j}}{2}-s}\,(1-p_2)^{\sum_j \sum_{k\neq j} \frac{l_{2j}l_{2k}}{2}-(k_i-s)}. \tag{5.17}$$

That is, instead of $P_i\left(N,p,r,k_i\right) = \binom{\binom{n}{2}}{k_i} p^{k_i}\left(1-p\right)^{\binom{n}{2}-k_i}$, we have that for each class t in τ, the probability that that class has exactly k_i intraclass edges is

$$\begin{aligned}&\widetilde{P}_i\left(N,p,r,k_i\right)\\&=\sum_{s=0}^{k_i}\binom{\sum_j \binom{l_{tj}}{2}}{s}\binom{\sum_j\sum_{k\neq j}\frac{1}{2}l_{tj}l_{tk}}{k_i-s}p_1^s p_2^{k_i-s}\left(1-p_1\right)^{\sum_j\binom{l_{tj}}{2}-s}\left(1-p_2\right)^{\sum_j\sum_{k\neq j}\frac{1}{2}l_{tj}l_{tk}-(k_o-s)}.\end{aligned} \tag{5.18}$$

Interclass edges Now we need to consider the probability that each class has k_o interclass edges. The situation is different here, for we only have to look at probability of k_o edges out of class 1 and this entails that there are k_o edges out of class 2 as well. Of course, if there were 3 classes, there being k_o edges out of class 1 would mean we still had to look for $\frac{k_o}{2}$ edges between classes 2 and 3, but at least part of the work is done in this case too. In fact, for each class t we need only account for edges between that class and classes yet to be considered. This means that only $k_o\alpha_t$ edges between class t and classes $u > t$ need to be sought.

In the case we are considering, with $r = 2$, class 1 of τ is made up of 2 nodes from class 1 of σ and 2 nodes from class 2 of σ, and so is class 2 of τ. Therefore, with $k_o = 2$, the only possibilities for the interclass edges in τ are that both edges are intraclass edges in σ, both are interclass in σ, or that there's one of each. There are $\sum_j l_{1j}l_{2j}$ possible edges out of class 1 that are intraclass edges in σ in this specific case, or for any r, $\sum_{u>t}\sum_j l_{tj}l_{uj}$. So there are $\binom{\sum_j l_{1j}l_{2j}}{2}$ ways to get the first scenario. The probability of the first scenario is similar to before:

$$\left(\frac{k_i}{\binom{n}{2}}\right)^2\left(1-\frac{k_i}{\binom{n}{2}}\right)^{\sum_j l_{1j}l_{2j}-2}\left(1-\frac{k_o}{n\left(N-n\right)}\right)^{\sum_j\sum_{k\neq j}\frac{l_{1j}l_{2k}}{2}}, \tag{5.19}$$

which can also be written in terms of p_1 and p_2 as

$$p_1^2\left(1-p_1\right)^{\sum_{u>t}\sum_j l_{tj}l_{uj}-2}\left(1-p_2\right)^{\sum_{u>t}\sum_j\sum_{k\neq j}\frac{l_{tj}l_{uk}}{2}}. \tag{5.20}$$

There are also $\sum_j\sum_{k\neq j}\frac{l_{1j}l_{2k}}{2}$ possible edges that are interclass in σ, or more generally $\sum_{u>t}\sum_j\sum_{k\neq j}\frac{l_{tj}l_{uk}}{2}$. Thus similarly there are $\binom{\sum_j\sum_{k\neq j}\frac{l_{1j}l_{2k}}{2}}{2}$ ways to get the second scenario, and $\left(\sum_j l_{1j}l_{2j}\right)\left(\sum_j\sum_{k\neq j}\frac{l_{1j}l_{2k}}{2}\right)$ to get the third. The probabilities of the second and third scenarios occuring are

$$p_2^2\left(1-p_1\right)^{\sum_{u>t}\sum_j l_{tj}l_{uj}}\left(1-p_2\right)^{\sum_{u>t}\sum_j\sum_{k\neq j}\frac{l_{tj}l_{uk}}{2}-2}, \tag{5.21}$$

and

$$p_1p_2\left(1-p_1\right)^{\sum_{u>t}\sum_j l_{tj}l_{uj}-1}\left(1-p_2\right)^{\sum_{u>t}\sum_j\sum_{k\neq j}\frac{l_{tj}l_{uk}}{2}-1}. \tag{5.22}$$

Thus, instead of $P_o\left(N,p,r,k_o\alpha_t\right)$, the probability that there are exactly k_o edges between class t

and all of the other classes is

$$\begin{aligned}
&\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right) \\
&\quad = \sum_{s=0}^{k_o\alpha_t} \binom{\sum_{u>t}\sum_j l_{tj}l_{uk}}{s} \binom{\sum_{u>t}\sum_j\sum_{k\neq j} l_{tj}l_{uk}}{k_o\alpha_t - s} p_1^s p_2^{k_o\alpha_t - s} \left(1-p_1\right)^{\sum_{u>t}\sum_j l_{tj}l_{uk}-s} \\
&\qquad\qquad \left(1-p_2\right)^{\sum_{u>t}\sum_j\sum_{k\neq j} l_{tj}l_{uk}-\left(k_o\alpha_t - s\right)}. \quad (5.23)
\end{aligned}$$

Thus

$$\mathbb{P}\left(\tau \text{ valid}|\sigma \text{ valid}\right) = \prod_{t=1}^{r} \widetilde{P}_i\left(N,p,r,k_i\right)\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right), \quad (5.24)$$

where $\widetilde{P}_i\left(N,p,r,k_i\right)$ and $\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right)$ are as defined in Equations 5.3 and 5.4.

5.3.2 A formula for the variance

Now, using Equation 5.24, we can say that:

$$\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)^2\right] = \sum_{\sigma,\tau}\left[\prod_{t=1}^{r} P_i\left(N,p,r,k_i\right)P_o\left(N,p,r,k_o\alpha_t\right)\right]\left[\prod_{t=1}^{r} \widetilde{P}_i\left(N,p,r,k_i\right)\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right)\right], \quad (5.25)$$

where where $\widetilde{P}_i\left(N,p,r,k_i\right)$ and $\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right)$ are defined in Equations 5.3 and 5.4.

Also, because the partitions are chosen randomly, the l_{ij}'s are approximately $\frac{N}{r^2}$. This means that we can approximate $\widetilde{P}_i\left(N,p,r,k_i\right)$ and $\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right)$ too.

$$\begin{aligned}
&\widetilde{P}_i\left(N,p,r,k_i\right) \\
&\quad \approx \sum_{s=0}^{k_i}\binom{r\frac{1}{2}\left(\frac{N}{r^2}\right)^2}{s}\binom{r^2\frac{1}{2}\left(\frac{N}{r^2}\right)^2}{k_i - s} p_1^s p_2^{k_i-s}\left(1-p_1\right)^{r\frac{1}{2}\left(\frac{N}{r^2}\right)^2-s}\left(1-p_2\right)^{r^2\frac{1}{2}\left(\frac{N}{r^2}\right)^2-\left(k_o-s\right)} \\
&\qquad \approx \sum_{s=0}^{k_i}\binom{\frac{N^2}{2r^3}}{s}\binom{\frac{N^2}{2r^2}}{k_i-s} p_1^s p_2^{k_i-s}\left(1-p_1\right)^{\frac{N^2}{2r^3}-s}\left(1-p_2\right)^{\frac{N^2}{2r^2}-\left(k_o-s\right)} \\
&\quad \approx \left(1-p_1\right)^{\frac{N^2}{2r^3}}\left(1-p_2\right)^{\frac{N^2}{2r^2}}\sum_{s=0}^{k_i}\binom{\frac{N^2}{2r^3}}{s}\binom{\frac{N^2}{2r^2}}{k_i-s}\left(\frac{k_i}{\binom{n}{2}-k_i}\right)^s\left(\frac{k_o}{n\left(N-n\right)-k_o}\right)^{k_i-s} \\
&\qquad = \overline{P}_i\left(N,p,r,k_i\right). \quad (5.26)
\end{aligned}$$

$$
\begin{aligned}
&\widetilde{P}_o\left(N,p,r,k_o\alpha_t\right)\\
&\approx \sum_{s=0}^{k_o\alpha_t}\binom{r\left(r-t\right)\left(\frac{N}{r^2}\right)^2}{s}\binom{r^2\left(r-t\right)\left(\frac{N}{r^2}\right)^2}{k_o\alpha_t-s}p_1^s p_2^{k_o\alpha_t-s}\left(1-p_1\right)^{r(r-t)\left(\frac{N}{r^2}\right)^2-s}\left(1-p_2\right)^{r^2(r-t)\left(\frac{N}{r^2}\right)^2-(k_o\alpha_t-s)}\\
&\qquad\approx \sum_{s=0}^{k_o\alpha_t}\binom{\frac{N^2(r-t)}{r^3}}{s}\binom{\frac{N^2(r-t)}{r^2}}{k_o\alpha_t-s}p_1^s p_2^{k_o\alpha_t-s}\left(1-p_1\right)^{\frac{N^2(r-t)}{r^3}-s}\left(1-p_2\right)^{\frac{N^2(r-t)}{r^2}-(k_o\alpha_t-s)}\\
&\approx \left(1-p_1\right)^{\frac{N^2(r-t)}{r^3}}\left(1-p_2\right)^{\frac{N^2(r-t)}{r^2}}\sum_{s=0}^{k_o\alpha_t}\binom{\frac{N^2(r-t)}{r^3}}{s}\binom{\frac{N^2(r-t)}{r^2}}{k_o\alpha_t-s}\left(\frac{k_i}{\binom{n}{2}-k_i}\right)^s\left(\frac{k_o}{n\left(N-n\right)-k_o}\right)^{k_o\alpha_t-s}\\
&\qquad\qquad\qquad\qquad= \overline{P}_o\left(N,p,r,k_o\alpha_t\right). \qquad (5.27)
\end{aligned}
$$

Thus the variance is approximately

$$
\begin{aligned}
&\operatorname{var}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]\\
&\quad\approx \mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]\left[\prod_{t=1}^{r}\binom{N-(t-1)\,n}{n}\overline{P}_i\left(N,p,r,k_i\right)\overline{P}_o\left(N,p,r,k_o\alpha_t\right)-\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]\right].
\end{aligned}
\qquad (5.28)
$$

This means that we can numerically approximate the variance and therefore consider the variance at points such as k_i^* and r^*. We want to look at the variance for $k_i \geq k_i^*$, as we want to determine whether partitions with larger k_i than k_i^* (and consequently larger modularity) occur frequently or not.

5.3.3 Computation

Thus using these equations for the variance in number of partitions, I computed the partition that is both *frequently occurring* and has largest $k_i \geq k_i^*$, call k_i' the largest such value. By frequently occurring I mean that the expectation plus one standard deviation is greater than or equal to 1, so a partition with that number of intraclass edges frequently exists. By the largest such partition I mean that partitions with $k_i > k_i'$ do not fit this criteria and thus partitions. On the one hand, $Q\left(N,p,r,k_i'\right)$ is larger than $Q\left(N,p,r,k_i^*\right)$, and is frequently the maximum modularity of ER random graphs, so modularity results larger than $Q\left(N,p,r,k_i^*\right)$ are not necessarily significant. On the other hand, because $Q\left(N,p,r,k_i'\right)$ is the largest such frequently occurring maximum modularity, any modularity result larger than this *can* be said to be unlikely to be caused by randomness. Therefore $Q\left(N,p,r,k_i'\right)$ provides a measure for the point at which we can call the structure found by modularity-optimization significant. It is important to consider the variance because the maximum frequently occuring modularity can be significantly larger than that given by the expectation. This can be seen in Figure 5.5, which shows the frequently occurring maximum modularity ("maximum Q") versus the expected or typical maximum modularity ("expected Q"). For all but very small p (less than 0.02) there is at least a 0.05 difference between the frequently occuring and typical maximum modularities.

Note Because the equations for the expectation and variance of $\mathcal{Z}\left(N,p,r,k_i\right)$ involved multiplying enormous numbers by miniscule numbers, this required matching up numbers of order $p =$

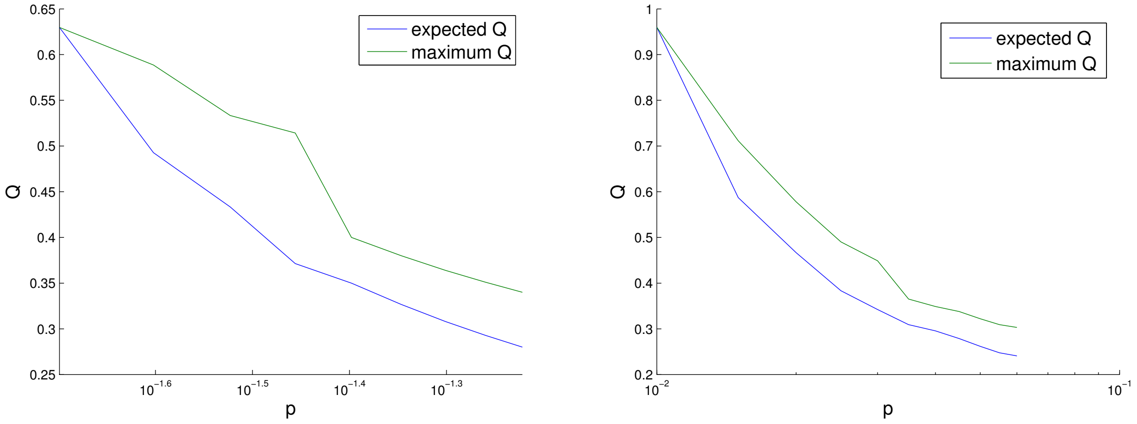

Figure 5.5: Plots showing results taking into account the variance for $N = 100$ (left) and $N = 150$ (right).

$O\left(\frac{1}{N}\right)$ with numbers of order N.

5.4 Conclusions

As we argued at the beginning of this chapter, while considering the typical modularity of an ER random graph allows us to show the insignificance of a modularity score, considering the variance in the number of equipartitions gives a better bound for showing that a network modularity score is significant. For to say that the structure it represents is unlikely to have occurred at random, one needs to be able to say that modularity scores that high do not frequently occur at random, not merely that they do not typically occur at random. That is, whilst showing what a typical modularity score on a random graph tells us that anything up to and including that score is likely to be random, to say that a score is unlikely to be caused at random requires more than merely being above the typical value; it requires that the score be above *all* frequently occuring random graph modularity scores (not just the *most* frequently occurring one). In this chapter we presented a derivation of a formula for the variance in the the random variable $\mathcal{Z}\left(N, p, r, k_i\right)$ and from it I computed for various N and p the largest frequently occuring modularity for a range of ER random graphs. From the results, presented in Figure 5.5, we saw that the largest frequently occuring maximum modularity was in general larger than the typical modularity, therefore the variance should be considered in this manner when making claims regarding a network having significant structure.

Chapter 6

Convergence in simulated annealing

6.1 Introduction

Definition 18. To fix notation, *with high probability* means with probability almost or tending to 1 as N tends to infinity.

Definition 19. By *convergence* we mean the point at which with high probability no further increase (in modularity) is possible through repetition of the same process.

In this section we investigate the convergence of individual node moves in Guimerà and Amaral's simulated annealing algorithm. Guimerà and Amaral's simulated annealing algorithm [15] runs individual node moves in blocks of N^2, or fN^2 for some specified constant f, with the aim of maximising modularity. However, there is no theoretical understanding behind this choice. The choice was made based on computational findings that suggest it might be a good choice. To see that they do indeed suggest this, consider Figure 6.1, which depicts modularity versus number of steps for $N = 100$ and mean degree $k \in \{3, \ldots, 20\}$. As you can see, all but the case where $k = 3$ have fully converged by N^2 moves, and even the anomalous case clearly converge in $O(N^2)$. In this section we show that simulated annealing's individual node moves do indeed converge within $O(N^2)$ moves. Before we begin, let us fix our notation.

6.2 Notation

Definition 20. For simplicity, in this chapter, an *individual node move* (or a *step)* involves choosing both a node and a different class of the current partition, and moving that node to the new class if *and only if* the modularity increases.

Note Node moves that result in a decrease in modularity also occur in the algorithm, with a small probability decreasing with the number of iterations performed. That probability, $\exp\left(-\frac{\triangle Q}{T}\right)$, is designed to be so small as to occur only a few times in each run of the algorithm. Thus, the consequence of these moves would be the multiplication of any result by

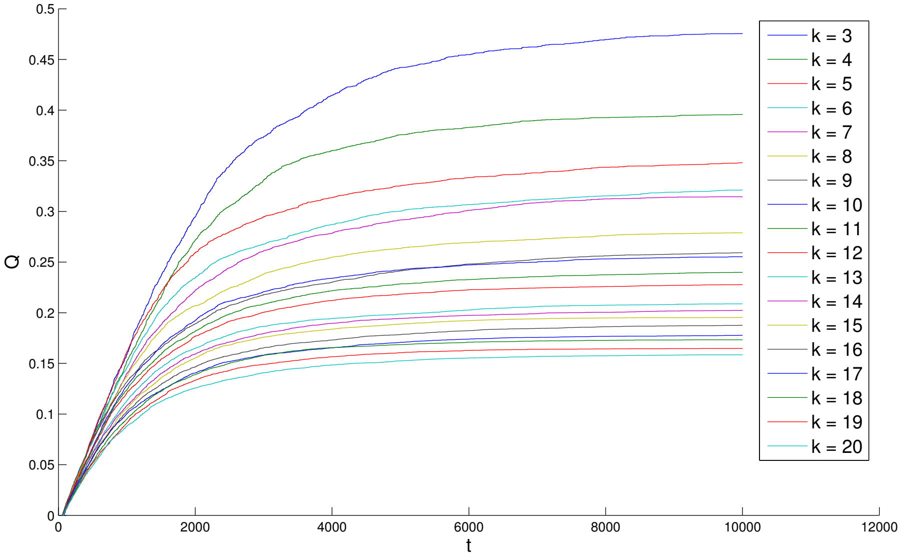

Figure 6.1: Convergence of individual node moves in a run of simulated annealing on $N = 100$, $3 \leq k \leq 20$.

a small factor, and thus we can ignore this situation for the following theory, where we shall be less exact than that.

Definition 21. We shall call a step a *success* if the move is completed and merely a *try* otherwise. Let us also say that a new *stage* begins after each success.

Definition 22. Finally, recall that $G = G(N, p)$ is an ER random graph as in Definition 6, where throughout this chapter $p = \frac{k}{N}$ so that the mean node degree is k for $1 \leq k \leq N - 1$.

Definition 23. To fix notation let *dense* mean having a lot of edges, more specifically in the limit of large N a dense graph has $O\left(N^2\right)$ edges. Let *sparse* mean having few edges, and in the limit of large N having $O\left(N\right)$ edges. [6]

6.3 Hypothesis

For fixed N, let us think about the extreme cases: k small and k large. Call them the sparse case and the dense case in accordance with Definition 23.

Sparse case: In a sparse graph, the nodes have so few options (there are so few classes with neighbours of theirs in) that they won't move to a different class very often, even if there's a large number of classes. This means that each node moves very few times, say $O(1)$, so there are only $O(N)$ successful moves before convergence. Also, because the mean degree is small, each try has a very small chance of finding a neighbour, and thus at least as small a chance of success. This

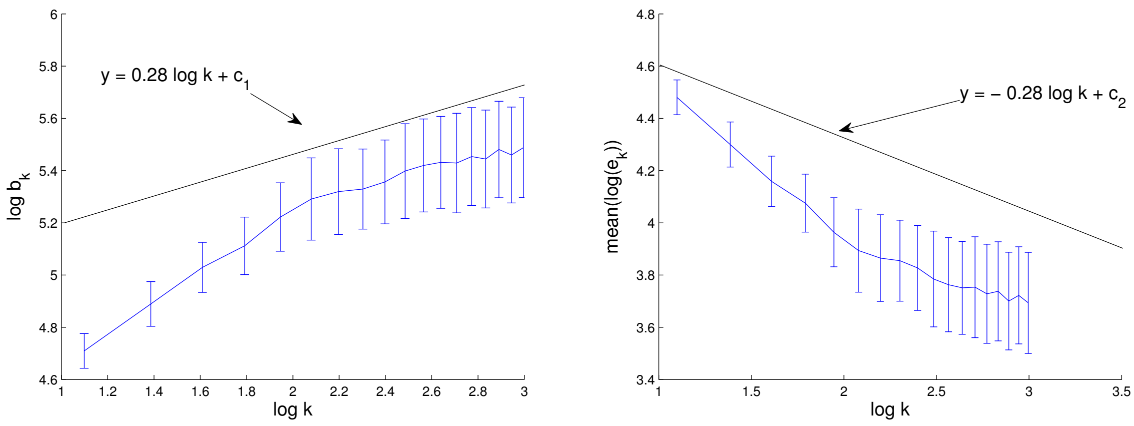

Figure 6.2: *Left:* Logarithmic scale plot of the mean number of successes versus the mean degree k, showing at most linear growth with gradient $\alpha \leq 1$. Bars indicate standard deviation. *Right:* Logarithmic scale plot of the mean number of tries per success versus the mean degree k, showing at least linear growth with gradient $-\alpha$. Bars indicate standard deviation.

means that we might have to try a lot of moves before there is a successful one. There are $O(N)$ successful moves out of a maximum of N^2, so on average there might be $O(N)$ tries per success. Thus, as the number of steps to convergence is the number of successful node moves multiplied by the number of tries per success, we have $O(N^2)$ steps to convergence in the sparse case.

Dense case: In a dense graph, the mean degree is large, so each node is connected to a significant fraction of the other nodes. This means that there are so many successful moves possible that practically any move is a success, so there are only $O(1)$ tries per success on average. Conversely, each node having a large number of neighbours means that they are unlikely to settle in a community. This means that the nodes will move around a lot more than in the dense case - they could potentially move up to $O(N)$ times (if their neighbours are divided into many classes). This means that there are up to $O(N^2)$ successful moves, but only $O(1)$ tries per success and therefore $O(N^2)$ moves to convergence in the dense case too.

Computation: I ran simulated annealing individual node moves on ER random graphs of size $N = 100$ with varying mean degree k, to see what relation k had to the mean number of successes and the mean number of tries per success. My results are presented in Figure 6.2, which shows that the mean number of successes increases at most at a rate identical one upper bounding the rate at which the mean number of tries per success decreases.

Hypothesis: Let k be the mean degree in a network of N nodes. Then the number of tries per success is at most on average $O(\frac{N}{k})$ and the number of successful moves to convergence is at most on average $O(kN)$, therefore the number of tries to convergence is $O(\frac{N}{k}) \cdot O(kN) = O(N^2)$.

6.4 Showing convergence

If we can show that the hypothesis holds, it would *guarantee* the numerically observed convergence of Figure 6.1, explaining simulated annealing's behaviour and lending substance to the algorithm's popularity. What we shall see in this section is that on the one hand the number of tries per success, T, depends on the number of successes possible, which is bounded, and on the other hand, with small variance in class size and number of neighbours, nodes almost always move to classes with more neighbours, and this gives an upper bound on the number of possible successes, S. Note that we consider the behaviour for large N, in particular as $N \longrightarrow \infty$. All the work in this section is my own.

I begin by showing that the number of tries per success is bounded. I want to show that the expected number of successes in $O(\frac{N}{k})$ steps is at least 1. The expected number of successes, given a number of tries, is the probability of each try being a success, summed over all the tries. At a given stage, the probability of success of a following step is the ratio of the mean number of successful moves possible at this stage to the total number of possible moves. I want a lower bound for this, at an arbitrary stage, to give an upper bound on the number of tries per success, T.

Lemma 24. *There are, on average, at most $O(\frac{N}{k})$ tries per success.*

Proof. The mean number of successes per node is at most k, the mean number of neighbours, and it is not much smaller, for if it is *much* smaller, say $O(1)$, then very few moves are possible and we would converge almost instantly in all cases, which we would have seen in our computation. The total number of moves is the number of communities which is at most N, making the probability of success on average at least $\frac{k}{N}$ and we're done, because for $\sum_{\text{tries}} \mathbb{P}\,(\text{success})$ to be at least 1, we need at most $O(\frac{N}{k})$ tries, so $T \leq O(\frac{N}{k})$. □

Now we show that the variance is mean degree and class size is small, and that a node x will only move to a smaller or sparser class than its current one, or a class with more of its neighbours in. These propositions together are used in Lemma 29 to show that the number of successes to convergence is bounded.

Proposition 25. *The expected number of shared neighbours for any two nodes x and y is $\frac{k^2}{N}$.*

Proof. Expected number of neighbours of x shared with neighbour y is $(N-2)\,p^2 \sim \frac{k^2}{N}$ as edges are independent and $p = \frac{k}{N}$. □

Proposition 26. *The variance in mean degree tends to 0 as $N \longrightarrow \infty$.*

Proof. The mean degree of any given node x is $\sum_{y \neq x} \mathbb{P}(xy) = (N-1)\,p \sim k$ as $N \longrightarrow \infty$ and variance is $\sum_y \sum_z \mathbb{P}(xy \;\&\; xz) - \left(\sum_{y \neq x} \mathbb{P}(xy)\right)^2 = (N-1)^2\,p^2 - k^2 \longrightarrow 0$ as $N \longrightarrow \infty$ because edges are independent. □

Proposition 27. *A node x will only move from one class in the partition σ, say a, to another class in the new partition σ', say b, if the new class b contains more of x's neighbours than a and/or if b is smaller and/or sparser than a.*

Proof. Take a step with partition σ, where node x and class b are chosen, where x is currently in class a. By definition of a successful move, x will move from a to b (that is, the new partition σ' is chosen over σ) if and only if the change in modularity is strictly positive. Let $\delta(i,j) = 1$ if i and j are in the same class in σ and $\delta(i,j) = 0$ otherwise, and let $\delta'(i,j) = 1$ if i and j are in the same class in σ' and $\delta'(i,j) = 0$ otherwise. Then the change in modularity is $\triangle Q_{ab} = Q_\sigma - Q_{\sigma'} = \frac{1}{2M}\sum_y \left(A_{xy} - \frac{d_x d_y}{2M}\right)\delta(x,y) - \frac{1}{2M}\sum_z \left(A_{xz} - \frac{d_x d_z}{2M}\right)\delta'(x,y)$. This equation depends only on the number of non-zero A_{xy}'s (i.e. the neighbours of x) and the number and degrees of the members of the classes a and b, as all other contributions are identical in partitions σ and σ'. Hence, $\triangle Q_{ab} > 0$ if and only if b contains more of x's neighbours than a and/or if a is smaller and/or sparser than b. □

For the next proposition recall that the optimal partition of an ER random graph comes in the form of an equipartition [25].

Proposition 28. *With high probability, at any stage, the difference in class size is O(1).*

Proof. We start with an equipartition, the singleton partition. Suppose that we are at an equipartition (could be the singleton, or later on) with r classes. Suppose b gains a node at a stage. The probability that a class other than b is next to gain a node is:

$$\left(\frac{R-1}{R}.\frac{N-\frac{N}{R}}{N}\right) \sim \left(\frac{R-1}{R}\right)^2 \geq \left(\frac{N-1}{N}\right)^2 \longrightarrow 1$$

Furthermore the probability of any one class, b, growing twice within C stages for some constant integer C is:

$$\left(\frac{1}{R}\frac{N-\frac{N}{R}}{N}\right)^2 \left(\frac{R-1}{R}\right)^{C-2} \longrightarrow 0$$

Essentially, due to the uniform and independent nature of the choices of node and class, the chance of something happening to any particular class as opposed to any of the others tends to 0 and N tends to infinity, so for large N the probability of any stage containing classes more than $O(1)$ apart in size tends to 0. □

Lemma 29. *With high probability $O(Nk)$ successful moves to convergence.*

Proof. Propositions 26 and 28 imply that nodes rarely move to sparser or smaller communities, so they (almost always) move only to communities with more neighbours. Letting S be the number of successful steps to convergence, we know that S is at most the sum over each node of the mean number of times each node moves. Thus, with high probability,

$$S \leq \sum_x \sum_{d=1}^{k} O(1) = O(Nk).$$

□

Theorem 30. *With high probability simulated annealing's individual node moves converge within $O(N^2)$ steps.*

Proof. This follows from Lemmas 24 and 29, as the number of steps to convergence is given by $S \times T = O(N^2)$. □

6.5 Conclusions

In this chapter, we saw that with high probability simulated annealing's individual node move iterations converge on ER random graphs in $O\left(N^2\right)$ steps. This helps in the understanding of the behaviour of the simulated annealing algorithm. Furthermore, this result increases understanding of modularity optimization, as it helps to characterise the factors at play behind each sequence of moves. It provides insights into the upper bounds of the number of modularity increases. That they are, for example, a function of the number of times a node can move class with an increase in modularity. It also increases our understanding of the simulated annealing algorithm and allows us to make some guarantees about convergence in certain classes of networks, such as those relating to random graphs, for example the null model.

The proof above was only possible due to factors such as nodes only moving to classes that are smaller and/or sparser and/or with more neighbours in them, and the uniformity in choice of node and class. These factors are general to the algorithm, that is they are not specific to the network or graph it is run on. However, some factors, such as the uniformly distributed edges resulting in small deviation in node degree, are not necessarily properties found in empirical networks, so there is more work to be done to be able to show convergence on empirical networks.

Chapter 7

Conclusions and further research

The aim of this dissertation was to show the importance of studying the behaviour of modularity and modularity-optimizing community detection algorithms. The aim of community detection is to find communities *with significance*; that is, to find communities in the formal sense given in Chapter 1, with statistically significant results showing that this structure is not the result of chance, thereby giving us communities in the intuitive sense as opposed to groups of people that randomly happen to be connected (see Chapter 1). There are concerns about the ability of a polynomial time algorithm to do this, despite the near-optimal solutions modularity-optimizing community detection algorithms seem to provide in benchmark tests. These concerns precipitate from various results, including that providing optimal solutions is an NP-hard problem (see Chapter 2) and that modularity exhibits extreme near-degeneracy for optimal solutions and suffers from a resolution limit, discussed in Chapter 4. On top of this, as also discussed in Chapter 4, properties of the algorithms themselves, such as the use of recursive bipartitioning, can limit ones ability to discern the optimal solution.

However, there is much that can be done to abate these concerns by investigating the *behaviour* of modularity and modularity-optimizing community detection algorithms. A little bit of work in this area has already been done, including Good et al's investigation of the behaviour of the modularity function [12], Richardson et al's investigation of the behaviour of spectral algorithms [29] (Chapter 4), and Reichardt and Bornholdt's [24, 25, 26, 27, 28] and Guimerà et al's [16] work on the modularity of ER random graphs (Chapter 5). My own work in this area (Chapters 5 and 6), involved investigating the convergence of individual node moves in simulated annealing, and taking into account the variance in modularity of ER random graphs to allow one to make claims about significant modularity results. In the work presented in Chapter 6 I showed that simulated annealing's individual node moves converge in $O(N^2)$ moves, justifying Guimerà and Amaral's choice of fN^2 moves for some positive integer f. In the work on the modularity of ER random graphs I presented a derivation of a formula for the variance in partitions with a specific modularity and used it to compute the largest frequently occuring modularity in ER random graphs $G(N, p)$. To our knowledge such a computation for modularity has never previously been done. It showed that the largest *likely* modularity was notably larger than the largest *typical* modularity and thus that the variance should be taken into account when making claims of significance.

If I were to extend this work in the future, I would do so in a number of ways. Firstly, I would have liked more time and computational resources in order to collect more results for the testing of hypotheses and I would have liked to have looked at larger networks. In terms of theoretical results, I would like to investigate the number of merges, splits, and decreasing modularity moves needed when running simulated annealing on ER random graphs to (with high probability) converge on a modularity score (that is, to get as close as possible to the optimal solution). An area I would particularly like to extend my investigation to is the expected and frequently occurring maximum modularity scores of other random graph models, such as the null model defined in Chapter 2, which is closely related to the *configuration model* [20], a random graph model for networks.

Definition 31. The *configuration model* is a random graph model with fixed degree sequence $\{d_1, \ldots, d_N\}$. Edges are placed using the "stub-matching" method of [19], where each node i is given d_i "stubs" (half-edges) and pairs of stubs are chosen randomly with equal probability to be "matched" (make an edge). [20]

The close connection to the null model arises in the limit of large N, where the expectation of there being an edge between nodes i and j becomes $\frac{d_i d_j}{2M}$, the same as for the null model [20]. The configuration model is a useful model of networks as it has many solvable properties [20]. These properties also make it useful in studying the community structure of empirical networks, therefore to be able to extend the results of this dissertation to the null model, and from there to the configuration model, would greatly help us to understand community structure in networks.

Appendix A

Analytical evaluation

In this section I present my work on solving $\mathbb{E}\mathcal{Z}\left(N,p,r,k_i\right)=1$ analytically. Guimerà et al [16] solve this equation numerically to obtain k_i^* given N,p and r, but this is very case-specific and more understanding could be gained from understanding how k_i^* behaves in response to variations in N,p and r. The best way to gain this understanding is to analytically manipulate the function $\mathbb{E}\mathcal{Z}\left(N,p,r,k_i\right)=1$ in order to obtain a function for k_i^* in terms of N,p and r.

First note that numerical solution of this equation suggests that k_i^* is $O(N)$ for r small, and that r^* is typically small (usually as small as 2 or 3 for $N\leq 200$, except for especially small p). Thus we may assume that $k_i \ll N^2$ and $r \ll N$. We shall also use Stirling's approximation, $\ln\left(n!\right)\approx n\ln\left(n\right)-n$. and the fact that for $x\ll 1$, $\ln\left(1-x\right)\approx -x$.

Recall from Equation 5.2 that

$$\mathbb{E}\left[\mathcal{Z}\left(N,p,r,k_i\right)\right]=\prod_{t=1}^{r}\binom{N-\left(t-1\right)n}{n}P_i\left(N,p,r,k_i\right)P_o\left(N,p,r,k_o\alpha_t\right). \tag{A.1}$$

Thus we want to solve

$$1=\prod_{t=1}^{r}\binom{N-\left(t-1\right)n}{n}P_i\left(N,p,r,k_i\right)P_o\left(N,p,r,k_o\alpha_t\right), \tag{A.2}$$

for k_i, given N,p and r. Note that for $t=r$, $\alpha_t=0$ and so $\binom{N-(t-1)n}{n}=1$ and $P_o\left(N,p,r,k_o\alpha_t\right)=1$. Thus solving Equation A.2 is equivalent to solving

$$0=\sum_{t=1}^{r-1}\ln\left[\binom{N-\left(t-1\right)n}{n}P_i\left(N,p,r,k_i\right)P_o\left(N,p,r,k_o\alpha_t\right)\right]+P_i\left(N,p,r,k_i\right) \tag{A.3}$$

Using the definitions of $P_i\left(N,p,r,k_i\right)$ and $P_o\left(N,p,r,k_o\alpha_t\right)$ from Equations 5.3 and 5.4, we have

$$0=\sum_{t=1}^{r-1}\ln\left[\binom{N-(t-1)\,n}{n}\binom{\binom{n}{2}}{k_i}p^{k_i}\,(1-p)^{\binom{n}{2}-k_i}\binom{n\,(N-n)}{k_o\alpha_t}p^{k_o\alpha_t}\,(1-p)^{n(N-n)-k_o\alpha_t}\right]$$
$$+\binom{\binom{n}{2}}{k_i}p^{k_i}\,(1-p)^{\binom{n}{2}-k_i} \tag{A.4}$$

and

$$\ln\left[\binom{N-(t-1)\,n}{n}\binom{\binom{n}{2}}{k_i}p^{k_i}\,(1-p)^{\binom{n}{2}-k_i}\binom{n\,(N-n)}{k_o\alpha_t}p^{k_o\alpha_t}\,(1-p)^{n(N-n)-k_o\alpha_t}\right]$$
$$=\ln\binom{N-(t-1)\,n}{n}+\ln\binom{\binom{n}{2}}{k_i}+k_i\ln(p)+\left(\binom{n}{2}-k_i\right)\ln(1-p)+\ln\binom{n\,(N-n)}{k_o\alpha_t}$$
$$+\,k_o\alpha_t\ln(p)+\left(n\,(N-n)-k_o\alpha_t\right)\ln(1-p)\,. \tag{A.5}$$

Expanding this and using the fact that $k_o=\frac{N^2p}{r}-2k_i$, and moving k_i terms to the left hand side of the equation we get

$$\sum_{t=1}^{r-1}[k_i\ln k_i+\left(\binom{n}{2}-k_i\right)\ln\left(\binom{n}{2}-k_i\right)+\left(\frac{N^2p}{r}-2k_i\right)\left(\frac{r-t}{r-1}\right)\ln\left(\left(\frac{N^2p}{r}-2k_i\right)\left(\frac{r-t}{r-1}\right)\right)$$
$$-k_i\ln p+k_i\ln(1-p)+\left(n\,(N-n)-\left(\frac{N^2p}{r}-2k_i\right)\left(\frac{r-t}{r-1}\right)\right)\ln\left(n\,(N-n)-\left(\frac{N^2p}{r}-2k_i\right)\left(\frac{r-t}{r-1}\right)\right)$$
$$+\,2k_i\left(\frac{r-t}{r-1}\right)\ln p+2k_i\left(\frac{r-t}{r-1}\right)\ln(1-p)]-k_i\ln k_i+k_i+k_i\ln\binom{n}{2}-\frac{k_i^2}{\binom{n}{2}}+k_i\ln p-k_i\ln(1-p)$$
$$=\sum_{t=1}^{r-1}[(N-(t-1)\,n)\ln(N-(t-1)\,n)-(N-(t-1)\,n)-(N-(t-2)\,n)\ln(N-(t-2)\,n)$$
$$+\,(N-(t-2)\,n)+\binom{n}{2}\ln\binom{n}{2}+n\,(N-n)\ln(n\,(N-n))+\binom{n}{2}\ln(1-p)+\frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\ln p$$
$$+\,n\,(N-n)\ln(1-p)-\frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\ln(1-p)-n\ln(n)+n]-\binom{n}{2}\ln(1-p)\,. \tag{A.6}$$

Thus using the approximations above we have

$$\begin{aligned}
&\sum_{t=1}^{r}[k_i \ln k_i + k_i(-1 - \ln \binom{n}{2} - \ln p - p - 2\left(\frac{r-t}{r-1}\right)\ln\left(\frac{N^2p}{r}\right) - 2\left(\frac{r-t}{r-1}\right)\ln\left(\frac{r-t}{r-1}\right) \\
&\quad - \frac{4N^2p}{rn\left(N-n\right)}\left(\frac{r-t}{r-1}\right) + 2\left(\frac{r-t}{r-1}\right)\ln\left(n\left(N-n\right)\right) + 2\left(\frac{r-t}{r-1}\right)\ln p - 2\left(\frac{r-t}{r-1}\right)\ln p) \\
&\quad + k_i^2\left(\frac{2}{n\left(n-1\right)} + \frac{4r}{N^2p} + \frac{4}{n\left(N-n\right)}\left(\frac{r-t}{r-1}\right)\right)] - k_i \ln k_i + k_i + k_i \ln\binom{n}{2} - \frac{k_i^2}{\binom{n}{2}} + k_i \ln p \\
&\quad - k_i \ln\left(1-p\right) = \sum_{t=1}^{r}[\left(N-\left(t-1\right)n\right)\ln\left(N-\left(t-1\right)n\right) - \left(N-\left(t-2\right)n\right)\ln\left(N-\left(t-2\right)n\right) \\
&\quad + \frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\ln\left(n\left(N-n\right)\right) + \frac{N^2p}{r} - \binom{n}{2}p - \frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\ln\left(\frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\right) \\
&\quad - \frac{1}{n\left(N-n\right)}\left(\frac{N^2p}{r}\right)^2\left(\frac{r-t}{r-1}\right) + 2n - n\ln\left(n\right)] - \binom{n}{2}\ln\left(1-p\right).
\end{aligned} \tag{A.7}$$

Now by observing that only some terms depend on t, we can rearrange this using standard partial sums such as

$$\sum_{t=1}^{r-1}\left(\frac{r-t}{r-1}\right) = \frac{1}{r-1}\left[\sum_{t=1}^{r-1} r - \sum_{t=1}^{r-1} t\right] = \frac{1}{r-1}\left[r\left(r-1\right) - \frac{r\left(r-1\right)}{2}\right] = \frac{r}{2}, \tag{A.8}$$

and

$$\begin{aligned}
&\sum_{t=1}^{r-1}\left(\frac{r-t}{r-1}\right)^2 = \frac{1}{\left(r-1\right)^2}\left[\sum_{t=1}^{r-1} r^2 - \sum_{t=1}^{r-1} 2t + \sum_{t=1}^{r-1} t^2\right] \\
&= \frac{1}{\left(r-1\right)^2}\left[r^2\left(r-1\right) - r\left(r-1\right) + \frac{1}{6}r\left(r-1\right)\left(2r-1\right)\right] = \frac{1}{\left(r-1\right)}\left[r^2 - r + \frac{1}{6}r\left(2r-1\right)\right]
\end{aligned} \tag{A.9}$$

Therefore we have

$$\begin{aligned}
&k_i(\ln p - \ln\left(1-p\right) + 2p - \left(r-2\right) - 4pr - \left(r-2\right)\ln\binom{n}{2} - \left(r-1\right)\ln\left(\frac{N^2p}{r}\right) - \left(r-1\right)\left(\sum_{t=1}^{r-1}\ln\left(\frac{r-t}{r-1}\right)\right) \\
&+ \left(r-1\right)\ln\left(n\left(N-n\right)\right)) + k_i^2\left(\frac{2\left(r-1\right)}{n\left(n-1\right)} + \frac{4r\left(r-1\right)}{N^2p} + \frac{4\left(r^2 - r + \frac{1}{6}r\left(2r-1\right)\right)}{\left(r-1\right)n\left(N-n\right)} - \frac{1}{\binom{n}{2}}\right) + \left(r-2\right)\left(k_i \ln k_i\right) \\
&= \frac{\left(r-1\right)N^2p}{2r}\ln p + 2\left(r-1\right)n + \sum_{t=1}^{r-1}[\left(N-\left(t-1\right)n\right)\ln\left(N-\left(t-1\right)n\right) - \left(N-\left(t-2\right)n\right)\ln\left(N-\left(t-2\right)n\right)] \\
&- \left(r-1\right)n\left(N-n\right)p + \frac{N^2p^2\left(r-1\right)}{2r} + \left(r-1\right)n\ln\left(n\right) - \left(r-1\right)\binom{n}{2}p + \frac{N^2p\left(r-1\right)}{2r}\ln\left(n\left(N-n\right)\right) \\
&\quad - \frac{N^2p\left(r-1\right)}{2r}\left(\sum_{t=1}^{r-1}\ln\left(\frac{N^2p}{r}\left(\frac{r-t}{r-1}\right)\right)\right) + \frac{N^2p\left(r-1\right)}{r} - \frac{N^4p^2\left(r-1\right)}{2r^2n\left(N-n\right)} - \binom{n}{2}\ln\left(1-p\right)
\end{aligned} \tag{A.10}$$

I would have liked to have developed this further, and if I were given the time I would. With more time it may be possible to isolate k_i on the left hand side, or to solve the quadratic to get k_i as a function of N, p and r, but the $\ln k_i$ term on the left hand side will cause problems with doing this in any standard way.

Bibliography

[1] Achlioptas, D., Naor, A., 2005, Annals of Mathematics, 162(3), pp. 1335 - 1351.

[2] Bickel, P.J., Chen, A., 2009, PNAS, 106(50), pp. 21068 - 21073.

[3] Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, 2008, Journal of Statistical Mechanics, P10008.

[4] Brandes, U., Delling, D., Gaertler, M., Goerke, R., Hoefer, M., Nikoloski, Z., and Wagner, D., 2008, IEEE Transactions on Knowledge and Data Engineering, 20(2), pp. 172 - 188.

[5] Danon, L., A. Díaz-Guilera, J. Duch, and A. Arenas, 2005, Journal of Statistical Mechanics, P09008.

[6] Diestel, R., 2006, Graph Theory, Birkhäuser.

[7] Duch, J., and A. Arenas, 2005, Physical Review E, 72(2), 027104.

[8] Expert, P., Evans, T. S., Blondel, V. D., and Lambiotte, R., 2011, PNAS, 108(19), pp. 7663 - 7668.

[9] Fortunato, S., 2010, Physics Reports, 486(3-5), pp. 75 - 174.

[10] Fortunato, S., and Barthélemy, M., 2007 Proceedings of the National Academy of Sciences USA, 104(1), pp. 36 - 41.

[11] Fruchtermann, T.M.J., Reingold, E.M., 1991, Software, Practice and Experience, 21(11), pp. 1129 - 1164.

[12] Good, B. H., de Montjoye, Y.-A., and Clauset, A., 2010, Physical Review E, 81(4), 046106.

[13] Girvan, M., and Newman, M. E. J., 2002, Proceedings of the National Academy of Sciences USA, 99(12), pp. 7821 - 7826.

[14] Guardiola, X., Guimerà, R., Arenas, A., Diaz-Guilera, A., Streib, D., & Amaral, L. A. N., 2002, eprint arXiv:0206240v1.

[15] Guimerà, R., and L. A. N. Amaral, 2005, Nature, 433(7028), pp. 895 - 900.

[16] Guimerà, R., M. Sales-Pardo, and L. A. N. Amaral, 2004, Physical Review E, 70(2), 025101 (R).

[17] Irons, L., 2008, Homeland Security Affairs, IV(1).

[18] Lancichinetti, A., S. Fortunato, and F. Radicchi, 2008, Physical Review E 78(4), 046110.

[19] Molloy, M., and B. Reed, 1995, Random Structures and Algorithms, 6(2-3), pp. 161 - 180.

[20] Newman, M., 2010, Networks: An introduction, Oxford University Press.

[21] Newman, M. E. J., 2004b, Physical Review E, 69(6), 066133.

[22] Newman, M. E. J., 2006a, Physical Review E, 74(3), 036104.

[23] Porter, M. A., J.-P. Onnela, and P. J. Mucha, 2009, Notices of the AMS 56(9), pp. 1082 - 1097, 1164 - 1166.

[24] Reichardt, J., and S. Bornholdt, 2004, Physical Review Letters, 93(21), 218701.

[25] Reichardt, J., and S. Bornholdt, 2006a, Physical Review E, 74(1), 016110.

[26] Reichardt, J., and S. Bornholdt, 2006b, Physica D: Nonlinear Phenomena, 224(1-2), pp. 20 - 26.

[27] Reichardt, J., and S. Bornholdt, 2007, Journal of Statistical Mechanics, P06016.

[28] Reichardt, J., and S. Bornholdt, 2007, Physical Review E, 76(1), 015102 (R).

[29] Richardson, T., P. J. Mucha, and M. A. Porter, 2009, Physical Review E, 80(3), 036111.

[30] Traud, A. L., Frost, C., Mucha, P. J. and Porter, M. A., 2009, Chaos 19(4), 041104.

[31] Zachary, W. W., 1977, Journal of Anthropological Research, 33(4), pp. 452 - 473.