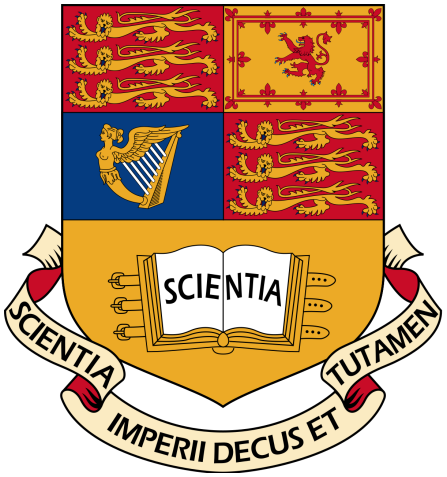
# Structural Controllability of Random Geometric Graphs




**Spencer R. Wilson**

**CID No. 01268121**

*Supervisor: Dr. Nick Jones*

2017

I would like to dedicate this thesis to my awesome parents and friends. Specifically, Georgia, Peter, Lawrence, Nic, John, Saul, Rahul, Sam, Mom, and Dad: thanks for putting up with me.

# Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this report are original and have not been submitted in whole or in part for consideration for any other degree or qualification at this or any other university. The work presented here is my own and contains nothing which is the outcome of work done in collaboration with others, unless specified explicitly or by citation in the text or Acknowledgements.

Spencer R. Wilson
CID No. 01268121
2017

# Acknowledgements

I am grateful to have had the patience of Dr. Nick Jones in guiding me through a previously unfamiliar research method: exploratory data analysis. Thanks to Matt Garrod for his prior work on many of the subjects touched upon in this project, specifically simulation programming and graph statistics. Thanks to Dr. Yang-Yu Liu for his helpful email correspondence about interpretations of structural controllability. Finally, thanks to the Foreign and Commonwealth Office and the Marshall Aid Commemoration Commission for funding this work.

# Abstract

In recent years, there has been considerable interest in the control theoretic properties of large dynamical systems. Specifically, structural controllability is a measure that determines how many components of a dynamical system need to be controlled and is found using graph theoretic algorithms. In this work, we analyze the structural controllability of random networks with a spatial embedding, as these models are unstudied in the context of control. Random geometric graphs (RGGs) are one class of such spatially embedded random network models where the positions of nodes in a $d$-dimensional metric space, sampled from a chosen distribution, determine the graph's topology. We draw samples from RGG ensembles with an underlying uniform distribution in $[0, 1]^d$ for solid (SRGGs) and periodic (PRGGs) boundaries, and from RGG ensembles with an underlying Gaussian distribution (GRGGs). We measure the fraction of nodes needed to control a graph, $n_D$, for each sample's largest connected component (LCC) over a range of mean degrees, $\langle\kappa\rangle$, and dimensions, $d$. We compute the decay rate of $n_D$ over $\langle\kappa\rangle$ for each value of $d$ and compare the results of each RGG type to that of Erdös-Rényi graphs generated of the same size and range of $\langle\kappa\rangle$. We find that the boundary conditions and underlying distributions imposed on the RGG models have a dominant effect on the $n_D$ decay rate. Specifically, SRGGs are less controllable compared to PRGGs and ER graphs in dimensions greater than 2, and GRGGs are less controllable than all other studied RGG models for all dimensions. We report a positive correlation between $n_D$ and the fraction of low-degree nodes for SRGGs and GRGGs connecting structural controllability to the degree distribution of graph samples. We discuss how our findings inform the design of control policies for spatially embedded networks and conclude with suggestions for future research directions.

# Table of contents

# Chapter 1

# Introduction

*One of the chief duties of the mathematician is acting as an advisor... to discourage from expecting too much from mathematics.*

Norbert Wiener

While graph theory took root, so the story goes, with Euler and his Königsburg bridge predicament, modern, random graph theory was conceived by Erdos, Renyi, and Gilbert in 1959 [1, 2]. However, the network metaphor fully permeated the collective consciousness only in the later 20th century. This theoretical diffusion is, perhaps, symbolized most apparently with the work of the French "schizoanalyst" philosophers Gilles Deleuze and Felix Guattari who, in *L'Anti Œdipe*, appropriated the concept of the "rhizome" to describe the tangled nature of a postmodern, post-industrial, hyper-technological society [3].

As Deleuze suggested, we can find the network concept anywhere we look, that is, in every situation where we wish to apply network models – models with individual units connected to one another in a certain fashion. Indeed, we find the network concept everywhere: power networks, communication networks (such as the one shown in Fig. 1.1), production networks, and more recently in protein interaction networks, general functional biology, financial markets, urban transportation, and even flavor profiles for culinary purposes [4–10]. We find network models applied across such a variety of length scales and domains that they probe the most inaccessible corners of the physical universe: the stuff of our brains and the fabric of the cosmos [11, 12]. In fact, understanding control and feedback in the brain is a scientific frontier that many believe is well-suited to network modeling, especially in the subfield of psychopathology [13–15]. To meet network science's increased demand, there exists a plethora of reviews detailing the uses of network modeling, as well as a definitive introductory textbook on the subject [16–18].

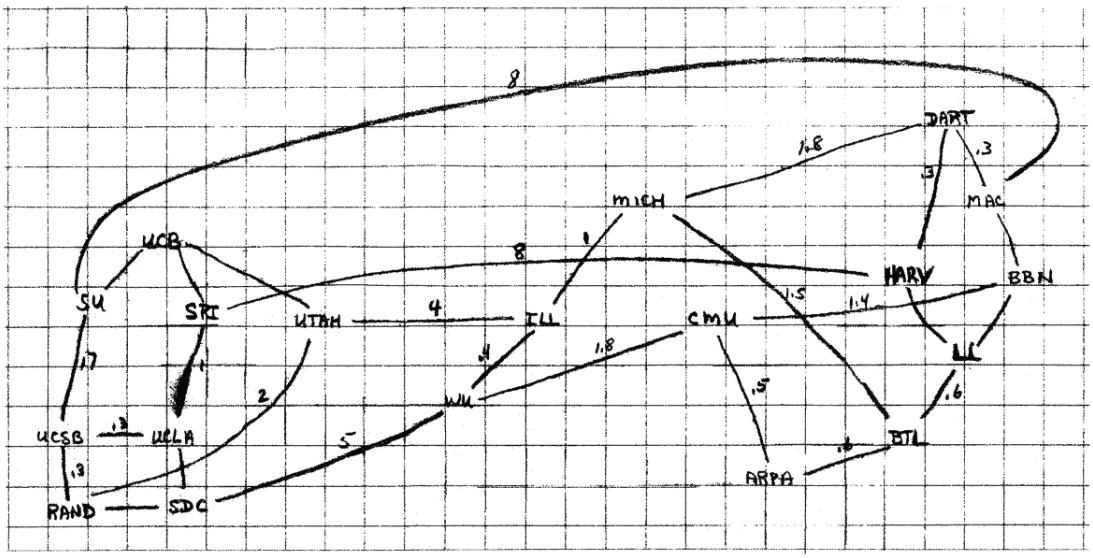



Fig. 1.1 Network sketch of the precursor to the Internet called ARPANET (Advanced Research Projects Agency, U.S. Department of Defense) from 1969, shown here as an undirected network with nodes as research institutions and edges labeled with weights [19]

As humans are wont to do, we currently yearn to move beyond describing physical systems using networks. For the last half-decade, the field of networks and the field of control theory have interacted in order to better understand how we might control "complex systems" [20, 21]. The scope of this work is to conduct an empirical study using techniques from network control theory on random geometric graphs (RGGs), a specific type of network model which have not been studied in the context of control thus far.

The aim of this project is to understand the parameters that influence the structural controllability, a measure of how many unique input signals are required to obtain a desired system-level output, of RGG models. In other words, what factors influence the control of RGG models, and thus of systems modeled by RGGs? We seek to relate RGG model parameters – node density, dimension, boundary condition, and underlying distribution – to the structural controllability of RGGs. Additionally, we contextualize our results for RGGs by comparing them with results generated for Erdös-Rényi (ER) random graph models over the same parameter ranges. This comparison highlights how the spatial embedding of RGG models change their controllability, if at all.

The second chapter intends to provide the knowledge necessary to interpret our results such that the report is as self-contained as possible. We give a basic background in the relationship between dynamical systems, networks, and their controllability, along with a mathematical description of the RGG and ER models used in the study. The third chapter details the key methods used to generate

the graph data and produce results. This includes details about software written for the study as well as statistical tools used to analyze the data.

The results of our study are presented in the fourth chapter, where we offer brief descriptions of the findings. In essence, we find that there is a significant difference in controllability among different types of RGG models as well as in comparison to the ER model. Specifically, we find that the choice of boundary conditions and underlying probability distributions used for the RGG models has a dominant effect on the type of control policy required. The fifth chapter provides a detailed discussion of these and other key findings as well as their implications in modeling and control of complex systems. This discussion is followed by suggestions for future directions in the study of spatially embedded network control theory.

# Chapter 2

# Background

*What most experimenters take for granted before they begin their experiments is infinitely more interesting than any results to which their experiments lead.*

Norbert Wiener

## 2.1 Network Control Theory

As we are interested in understanding the controllability of network models, we begin with an explanation of control theory and its relationship to networks. A control theory of systems that can be modeled as networks begins with linear control theory – models using some number of linear differential equations with $N$ state variables. Such systems have matrices of coefficients for their state and vectors for their inputs and outputs. We briefly discuss linear dynamical systems and their relationship to networks, and we define the notion of controllability for such systems. As a note on nomenclature, we refer to graphs and networks interchangeably, as there is no mathematical difference apart from a network being a graph with many nodes.

### 2.1.1 Linear Dynamical Systems

Linear Time-Invariant (LTI) dynamical systems are described in state space by a system of linear equations of the form

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) \tag{2.1}$$

where $x(t) \in \mathcal{R}^N$ is called the state vector, $A \in \mathcal{R}^{N \times N}$ is the state matrix, $B \in \mathcal{R}^{N \times M}$ is the input matrix, and $u \in \mathcal{R}^M$ is the control input vector. This is the most basic form of a feedback control system. This system can be represented

by a graph, where nonzero matrix entries of $A$, $a_{i,j}$, represent weighted edges from $i$ to $j$ in a graph with $N$ nodes. Thus, an edge of the corresponding graph represents two interacting state variables in the system, such that if $a_{i,j}$ is zero, these two variables do not share a state equation for $\dot{x}(t)$. In this way, the dynamical coefficient matrix encodes the topology, or structure, of the graph. The time invariance of LTI systems is due to the matrices not being functions of time. That is, the corresponding graph structure does not change over time, but only the input and state vectors, $u$ and $x$.

In general, we call the system $(A, B)$ "controllable" if it meets the Kalman criterion:

$$\mathbf{rank}(C) = N \tag{2.2}$$

where $C$ is the controllability matrix

$$C = (B, AB, A^2B, ..., A^{N-1}B). \tag{2.3}$$

In Kalman's words, "we find that a linear dynamical system is an irreducible realization of an impulse-response matrix if and only if the system is completely controllable and completely observable" [22]. The controllability question is based on the row-rank condition: is it possible to choose the non-zero entries in A and B such that the system satisfies this criterion? The answer to this question has a geometric interpretation: controllability is the ability of the state space to be populated with reachable states. If there is a rank deficiency in $C$, then all states are not reachable, as their are an insufficient number of independent basis vectors required to access the state space. Note that the Kalman criterion is a binary one: the system can be said to be controllable or not.

This criterion can be expanded upon in at least two ways: the *controllability Gramian* and *structural controllability*. The difference in these methods is their generality. The Gramian is used to assess systems where the dynamical coefficients, the elements of the $A$ matrix, are known exactly. Structural controllability assesses the topology of the system only, and does not require specific coefficients. We discuss structural controllability in the next section, and provide a brief explanation of the Gramian here for completeness.

The Gramian for LTI systems is defined as

$$\mathbf{W}(t) = \int_0^t e^{A\tau}\mathbf{B}\mathbf{B}^{\mathbf{T}}e^{A^T\tau}d\tau \tag{2.4}$$

and is found using an ansatz for the Lyapunov stability equation. Using the singular value decomposition (SVD) of $C$, the singular values can be interpreted as controllability components of the system. This finite-time Gramian version of controllability was used by Gu et. al. with success in the analysis of brain networks

to determine not only *if* a system (a network) is controllable, but *how* controllable it is [23]. Additionally, it has been shown that optimizing controllers can be computed using the SVD of the Gramian [24]. The Gramian is an elaboration on the Kalman condition for determining degrees of controllability through the eigenvalues of $C$ for specific system parameters. We will return to the Gramian in the last chapter of this work, and we present the basic theory behind the controllability Gramian to highlight the differences between this approach, for which a system's dynamical coefficients must be exactly known, and *structural controllability* which requires knowledge of the presence or absence of connections in the graphical representation of the system to quantify controllability.

### 2.1.2 Structural Controllability

Beginning with Lin's work in 1974 on the concept of *structural controllability*, we take a broader look at systems $(A, B)$ represented by graphs $G(A, B)$. This move to the underlying *structure* of the system in question is an effort to circumvent the usual case of unknown dynamical coefficients [25]. A baseline criterion that establishes the possibility for control given the structure of a system is useful in reality where system parameters are difficult to determine but the structure of the system is readily available. This is the case in many real-world networks, where the system's topology is known exactly, but the dynamical parameters are elusive [20]. Here, we assume that the system topology is known, and the precise weights of the coefficients are set to unity. We note, however, that even this assumption may be generous in some cases and will be discussed in the final chapter.

We refrain from exhaustively rewriting the detailed proofs of the structural controllability concept and refer the reader to Lin's wonderfully lucid paper. Here we reproduce the main conclusions of Lin's work used by Liu et. al., namely that the following three statements are equivalent:

1. A linear control system (A,B) is structurally controllable.
2. (a) The digraph G(A, B) contains no inaccessible nodes.
   (b) The digraph G(A, B) contains no dilation.
3. G(A, B) is spanned by cacti.

This is known in the literature as Lin's Structural Controllability Theorem. Inaccessible nodes are referred to in this work as *isolates*, as we do not allow self-loops in our graphs. Dilations occur when the size, or cardinality, of a topological neighborhood of a subset of nodes in a graph is smaller than the cardinality of the subset itself. A cactus is an important concept in the graph theory of network control, and is intuitive: a cactus is made up of stems and buds, where a stem

is a progression of directed edges without cycles, that is, a tree (in the graph theoretical sense) with a single stem – or trunk, as it were; a bud is a cycle with a single outgoing edge. To create a cactus, one can think of a tree with each branch ending with a cycle. The cactus is a minimum structure, meaning if a single edge is removed it is no longer a cactus.

We can imagine how this structure is controlled by a single *origin*, the root of the tree allowing a cascade through the stem and branches, much like a cactus taking up groundwater and distributing it to its leaves. The important point is that each node has its own *unique* input. The beauty of Lin's results lies in our ability to intuit them. Quite simply, as discussed by Ruths and Ruths, if a node has no incoming edges, we are unable to control that node, for it is a "source". If more than one node has incoming edges from a single node, we cannot control that subset of nodes, for it is either an "external" or "internal" dilation [26]. External dilations can be thought of as surplus sink nodes, and internal dilations as sink nodes within a neighborhood. By identifying these nodes, we create a minimum subset of nodes to make structural control of the network viable. Note that isolated nodes are trivially uncontrolled, and require a unique input.

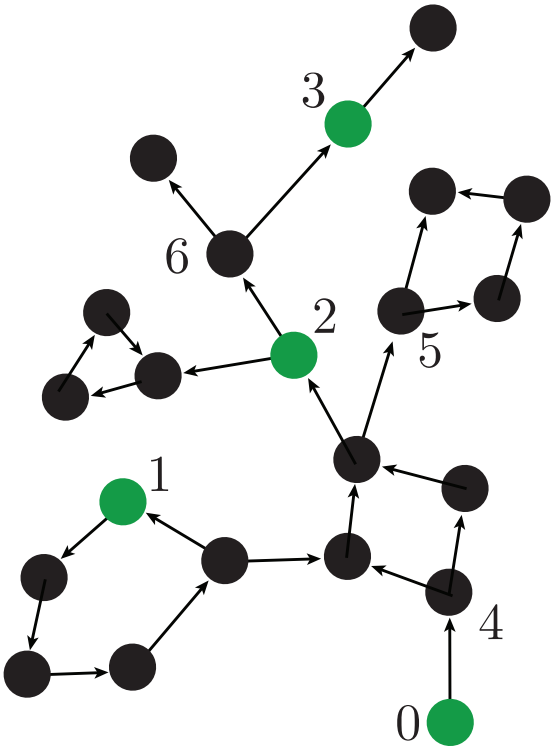



Fig. 2.1 A typical random graph with number of driver nodes, $N_D$, equal to 4. Nodes $0, 1, 2$ and 3 are one possible set of driver nodes, as the maximal matching is not unique. Nodes 4 and 5 and their neighbors are internal dilations, while node 6 is a point of external dilation due to surplus sink nodes at the end of the stem beginning with node 0.

It is important to note that these driver nodes establish the minimum set of *unique* control signal inputs, but it may be the case that to access parts of the state space a certain number of these unique inputs should be applied to more than one node, and there may be a multitude of viable configurations for the control inputs. Specifically, every cycle is self-controlled by design, though depending on the direction of its connection to the stem, it may require a control input that isn't necessarily unique. Similar ternary schemes for the categorization of nodes has been developed by classifying nodes as always, never, and sometimes necessary for complete control of the network [27]. These subtleties are shown in Fig. 2.1.

It is worth noting here that structural controllability only makes sense in the context of directed networks, and we give details on our implementation of directed networks in the next chapter. Additionally, we forbid the presence of self-loops in our systems, or what some sources refer to as "instrinsic nodal dynamics" [28]. Indeed, if every node included a self-loop, structural controllability would demand only a single unique control input. The inclusion of self-loops is a subtle point that is returned to in the last chapter.

### 2.1.3 Maximal Matchings

Real-world networks will include inaccessible nodes, dilations, and elements of cacti in some mixture. Thus, we need algorithms to classify nodes given a general graph. This is the extension of Lin's work by Liu et. al., where they have realized that the concept of the maximal matchings in graph theory applied to directed networks is coincident with Lin's connections between LTI control and the cactus graph topology. A maximal matching is simply the number of nodes without unique inputs. One can immediately see how this relates to source and dilation nodes discussed in the previous section.

In order to identify these unmatched nodes, which we will call the "driver nodes", we utilize the Hopcroft-Karp maximum matching algorithm starting from a bipartite graph representation [29]. In order to convert a directed graph into a bipartite graph, we simply duplicate each node as shown in Fig. 2.2, such that the left-hand side of the bipartite graph lists the node copies with outgoing edges, and the right-hand lists the incoming copies. The Hopcroft-Karp algorithm runs in $\mathcal{O}(\sqrt{V}E)$, where $V$ is the number of vertices and $E$ is the number of edges, by iteratively breadth-first searching through alternating matched and unmatched nodes to create what they call "augmenting paths". These paths alternate between matched and unmatched nodes since a maximal matching is defined as an edge set in which no two edges are coincident on one node. These augmenting paths are depth-first searched and the matching is updated until there are no more augmenting paths.

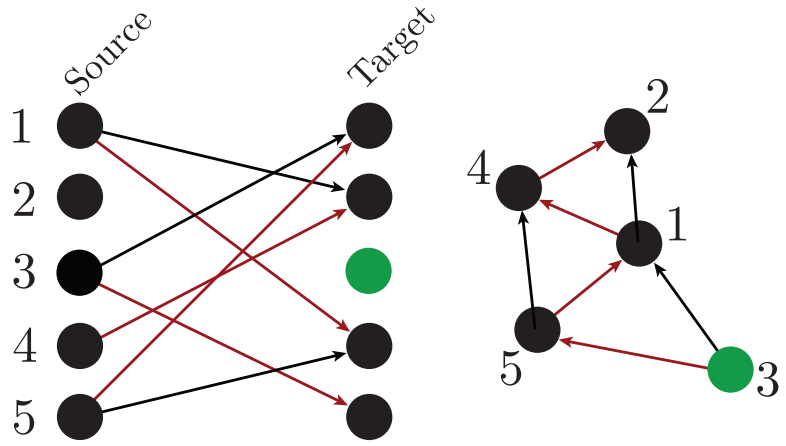



Fig. 2.2 a) The bipartite representation of a graph with five nodes where the lefthand side shows the outgoing edges and the righthand side shows the incoming edges. The unmatched node is shown in green and the matching is shown in red. Note that nodes may be unmatched and have incoming links, though here this is not the case, as the node is a source node. b) The usual graph representation with the driver node shown in green and the matching shown in red. Note that every node has a unique input given by the matched edges.

While a maximal matching is not unique, the number of nodes in the matching is constant over the set of possible maximal matchings. The number of driver nodes is thus the nodes that are left unmatched for any maximal matching. This coincides, in one sense, with our earlier discussion of control input multiplicity. Given our understanding of structural controllability, unmatched nodes are intuitive: any nodes that are sources have no incoming edges and are left unmatched, and any nodes in a dilation have the chance of being left out of maximal matching.

Following Ruths and Ruths and their notion of the "control profile", we note here that we can easily identify source nodes that are unmatched, for they have no incoming links. We call the number of this node set $n_D^s$. Similar, $n_D^e$ is the size of the set of external dilation nodes, or driver nodes with no outgoing links. As we know the total number of driver nodes, $n_D$, from the matching we can use the equation

$$n_D = n_D^s + n_D^e + n_D^i \tag{2.5}$$

to find the internal driver nodes, which have no demand for in- or out-degree.

### 2.1.4 Scaling of the Driver Node Fraction

We are interested in tracking the scaling of the driver node fraction for RGG models compared to ER graphs. We offer some background here on the prior research towards analytical methods for the scaling of the driver node fraction over mean degree. As spatially embedded networks are previously unstudied, the literature focuses ER. The following discussion is meant to motivate the

comparison between RGGs and ER graphs highlighted by our results in the fourth chapter.

Liu et al. use the concept of the maximal matching combined with the cavity method from statistical physics to derive a set of self-consistent equations describing energy arguments for random networks [20]. Essentially, this method ascribes an energy function to the matchings of a given graph, and determines the average minimum number of unmatched nodes, or driver nodes, given the type, size, and degree distribution for the ensemble of interest. They note that the method confirms the trivial result that regular directed graphs have perfect matchings with zero driver nodes. They provide equations to solve for $n_D$ of ER graphs where the degree distribution is Poisson. In the large mean degree limit, they find that

$$n_D \sim e^{-\frac{\langle \kappa \rangle}{2}} \tag{2.6}$$

where $\langle \kappa \rangle$ is the mean degree of the sample. We point out that the coefficient of the exponential is unity. This includes the assumption that at $\langle \kappa \rangle = 0$ all nodes are isolated. Thus, all nodes are driver nodes. We refer the reader to the original paper for details on the derivation of Liu's method. In this work, we implement these findings using a simple Newton's method routine in order to validate our own data for ER graphs, and compare these results to the data collected for RGGs.

Previously and in this section, we have built a strong foundation for the concept of controllability as applied to systems modeled using networks. So far, however, we have not mentioned any specific network model. In this work, we are interested in tracking the number and type of driver nodes over a range of parameters to explain the factors that effect controllability in network models with a spatial embedding. In the next section we introduce our specific model of interest, the RGG.

## 2.2 Random Geometric Graphs

Shortly after the initial conception of a theory of random graphs, Gilbert proposed a spatially embedded version of this theory known as the "Gilbert disc model" [30]. In this model, the sample graph is drawn from an ensemble for which the nodes are distributed within a $d$-dimensional Euclidean space according to a probability density function. Nodes are connected if they lie within a certain radius of each other.

This type of model and its variants are known as random geometric graphs (RGGs) or, more recently, spatially embedded networks. They are often used to model mobile ad-hoc networks (MANETS), such as sensor arrays, due to the intuition that these processes are limited in their connection power to a geometric

range centered on a node's current position in space [31]. Additionally, RGGs are used to model city and brain networks [32, 33]. In the context of brain networks, much of the literature argues that the models of neural networks should factor in "wiring costs" based on spatial distance, thus making the RGG a prime model candidate [34]. It has been argued that classical random graphs, such as ER graphs, and lattices are too dissimilar to many real world networks to be make useful predictions, as in the case of disease spread where the contagion infects individuals who are geographically adjacent [35].

The fecundity of the RGG model for real-world phenomena is also due to its flexibility with respect to dimension, boundary conditions, density function used for node placement, and function used to establish connections between nodes. For instance, if we want to simulate connections between people of a similar metric, say age and income, we might use a Gaussian kernel for our node placement process. As in the usual study of random graphs without spatial embedding, the parameters of the model are associated with an ensemble of random graphs, where a particular random graph is a sample drawn from this ensemble. Thus, in using random graphs to represent LTI systems as discussed in the previous chapter, each sample is drawn from an ensemble of LTI systems with a range of topologies. In this way, sampling random graphs and computing statistics tells us about the attributes of the ensemble of LTI systems that we are interested in modeling, or any other such complex system of interest.

Here, we draw samples from RGG ensembles with an underlying $d$-dimensional uniform distribution in $[0,1]^d$ for two for solid (SRGGs) and periodic (PRGGs) boundaries, and from RGG ensembles with an underlying $d$-dimensional Gaussian distribution (GRGGs) in $\mathcal{R}^d$. For SRGGs, PRGGs, and GRGGs, we impose the hard radius cutoff for connection as in the Gilbert disc model. That is, we use a connection function $p$ for the probability that two nodes, $i$ and $j$, with separation $r_{i,j}$ are connected where

$$p(i \text{ connected to } j) = \begin{cases} 1 & r_{i,j} < r_c \\ 0 & r_{i,j} \geq r_c \end{cases}$$

given a known connection radius, $r_c$. We show examples of the three types of RGG studied in this report in Fig. 2.3. In this work, we are interested in directed graphs, and thus each ensemble includes choices of direction for each edge.

Following Dall, we give the basic definitions of the RGG beginning with terminology for the ER graph that carries over to RGGs [35]. For ER graphs, we say that two nodes are connected with probability $p$, and the graph has mean degree $\langle\kappa\rangle$ over $N$ nodes where $\langle\kappa\rangle = \frac{pN(N-1)}{2}$. The graph connectivity is defined

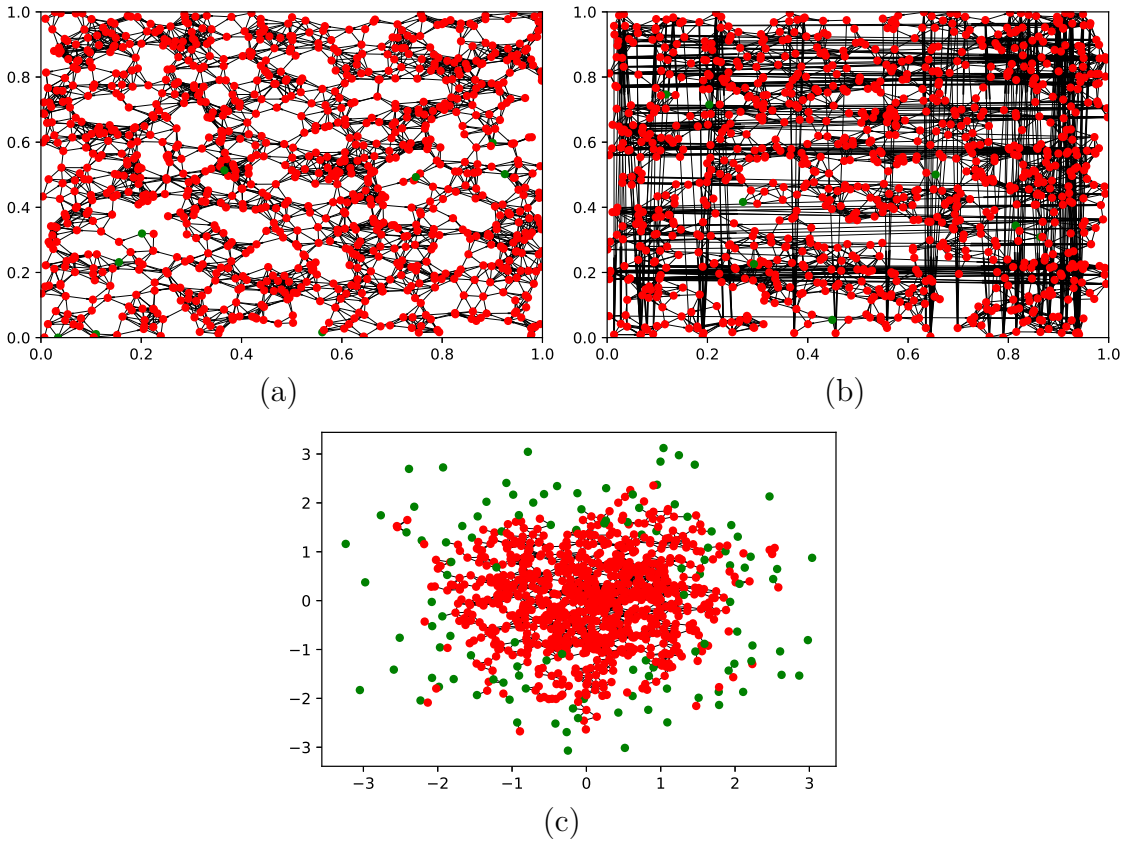



Fig. 2.3 RGG samples for N=1000 nodes, $\langle\kappa\rangle = 10$, and $d = 2$ with a) solid boundary (SRGG) b) periodic boundary (PRGG) (c) Gaussian kernel (GRGG).

as

$$\alpha = \frac{2\langle\kappa\rangle}{N} \tag{2.8}$$

$$= pN. \tag{2.9}$$

Since the probability of edges is binomial, we can approximate it as a Poisson distribution in the large $N$ limit

$$p(k) = \binom{N}{k} p^k (1-p)^{N-k} \approx \frac{\alpha^k e^{-\alpha}}{k!} \tag{2.10}$$

For SRGGs and PRGGs in the $d$-dimensional space $[0,1]^d$ with nodes added according to a uniform distribution, probability $p$ of two nodes being within radius $r$ is the *excluded volume* that they share, in this case a sphere with radius $2r$. Since the total volume of the space is 1, the excluded volume $V_{ex}$ is $2^d V$ where $V$ is the volume of a sphere surrounding each node of radius $r$. Connectivity is defined as $Np$ and is thus equal to $NV_{ex}$. Using the formula for the volume of a

sphere in $d$ dimensions, we can solve for $r$ to find

$$r = \frac{1}{2\sqrt{\pi}} \left[ \frac{\alpha}{N} \Gamma \left( \frac{d+2}{2} \right) \right]^{\frac{1}{d}}. \tag{2.11}$$

There currently exists no such relationship for GRGGs in $\mathcal{R}^d$. It is important to note that as $d \to \infty$, typical graph characteristics for PRGGs, such as connectivity, are known to converge to those of ER graphs [35]. However, spatially embedded graphs, where connectivity is driven by the choice of node distribution and radius connection function, lead to very different proofs for network characteristics than typical random graph models. Additionally, it is known that as dimension increases, effects on connectivity for random graphs confined in geometries are exacerbated [36]. This mixture of results for RGGs gives us the idea that the control properties of RGGs may be different than those of ER graphs. We hypothesize that the spatial embedding of RGGs will lead to a different structural controllability profile compared to their non-embedded counterparts.

In the next chapter we will describe how we implement the generation of RGG and ER graph samples in practice, as well as the methods used to extract useful statistics for analyzing the RGG ER ensembles.

# Chapter 3

# Methods

*The best material model of a cat is another, or preferably the same, cat.*

Norbert Wiener

## 3.1 Data Generation

As this is an experimental study, we provide details here of how samples were drawn from each ensemble, and the details of how each ensemble is defined programmatically. The necessary parameters to encode a sample from an ER ensemble are $\langle\kappa\rangle$ and $N$. For RGGs, we must include dimension $d$, boundary condition, node placement probability distribution, and connection function. In this study, we use a hard connection function identical to that of Gilbert's original model for all simulations. We are interested in three boundary types: periodic (or toroidal), solid (or bounded), and open. We will simulate periodic and solid boundary conditions for RGGs with a uniform placement, and open boundaries for a Gaussian Normal placement distribution. All links are chosen independently, and each link direction is chosen with probability $\frac{1}{2}$.

After generating the node locations, we store the graph in a k-d tree and perform efficient nearest-neighbor lookups using a binary search algorithm to generate the edges and store them in an edge list of two-tuples [37]. The k-d tree is of average $\mathcal{O}(N)$ space and time complexity on average, a significant speed up over brute search costing $\mathcal{O}(N^2)$ . We implement k-d trees using the `scipy` data structure `cKDTRee` in our library and perform fast nearest-neighbor searches using their optimized function `query_nearest`, though we note that efficiency could be gained by assimilating these algorithms into our codebase. Despite its efficacy, this data structure has not been implemented in any major network package as of yet since no major network package offers spatially embedded graph functionality. We discuss challenges associated with adding this capability in the final chapter.

Listing 3.1 Calculating the Dictionary Representation

```
1
2  def calculate_dictionary_representation(self):
3          # build dictionary from the edge list
4          nums = xrange(self.n)
5          # left-hand side of the graph, the sources/keys
6          nodelist = (str(num) + 's' for num in nums)
7          # empty dict of lists for the targets/values
8          D = {k: [] for k in nodelist}
9          # link sources/keys to targets/values
10         for idx, val in enumerate(self.source):
11                 D[str(val)+'s'].append(self.target[idx])
12         # update sample object attribute
13         self.bipartite_dict = D
```

From the source and target list data, we generate the adjacency matrix and the bipartite dictionary data structure for each graph sample and store it within its ensemble. The dictionary is created by storing $2N$ nodes with sources as keys and targets as values. The main function is shown in Listing 3.1 within the scope of the `RGGSample` object as shown in Fig. 3.1.

For each boundary condition we sweep over the parameter values of $\langle\kappa\rangle$ and $d$ and store this data on the hard disk for later processing to save time. For each ensemble, we use $N = 1000$ nodes and simulate 100 samples to compute statistics unless otherwise noted. The object-oriented structure of the python code is shown in Fig. 3.1. Using this structure, we are able to write more general functions that are more testable and easier to implement. All RGG ensembles share the same functionality and hold the same attributes. In this way, we can act on a wide range of ensembles using a lightweight codebase. Additionally, we created an `Experiment` class in order to hold a range of ensembles over $\langle\kappa\rangle$ necessary to extract statistics associated with the scaling of $n_D$ in a straightforward manner.

In order to simulate graphs with the required mean degree according to Eqn (2.11), we use a two-step process building from work by Garrod [38]. First, after the node positions are generated for a sample, the distance vector, the list of pairwise distances between every node in the sample, is computed and sorted. This can be thought of as the list of all possible edges and their length. Since the number of edges given $\langle\kappa\rangle$ is $\frac{N\langle\kappa\rangle}{2}$, the second step is to store the $E$'th element in the sorted distance vector as an approximation of the the radius, $r$. This ensures that the correct number of edges is formed when the hard connection function is applied to the graph. To average out errors in this radius computation scheme, we run this routine a number of times and take the average radius to be the critical radius, $r_c$, for a given ensemble. In the the case GRGGs, this method is required as a critical radius formula does not exist. We implement this method for all three RGG types to validate our results. Since the mean degree of ER graphs is

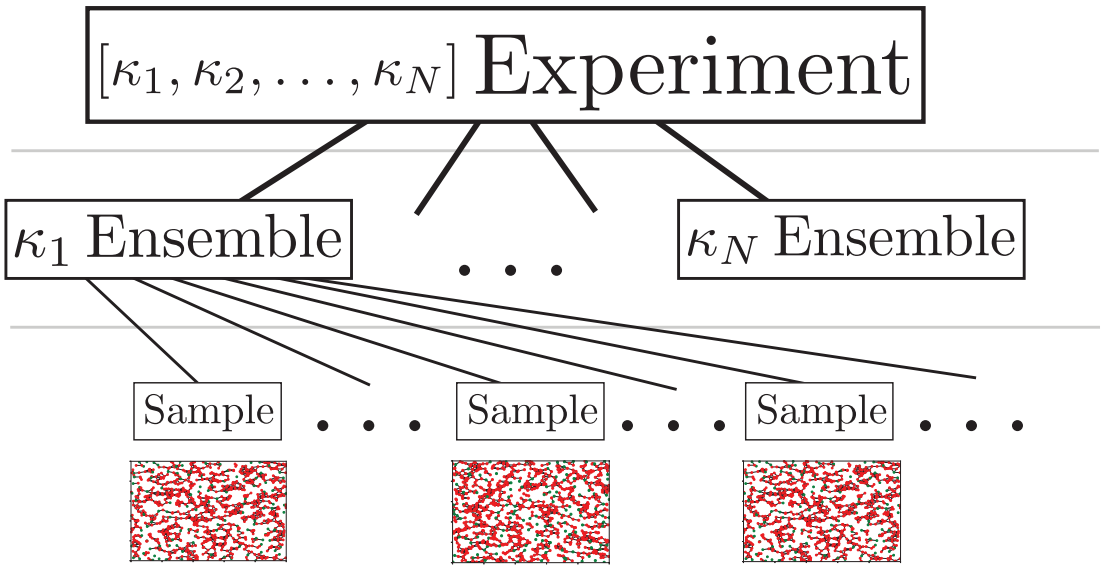



Fig. 3.1 A flowchart depicting the objects used in the python generation code. Experiment objects hold a range of $\langle\kappa\rangle$ values, with functions that pass to ensembles and samples, such as `compute_mean_degree()`. Samples are drawn and stored in the Ensemble objects, which are in turn stored in Experiment objects. This work flow makes the code much more readable and modular and applies to all graph types.

analytical, we confirm that our graph generation scheme produces samples with the correct value for $\langle\kappa\rangle$ by plotting the mean degree of ER graphs against each RGG graph type as shown in Fig. 3.2.

As discussed in the previous chapter, we use the Hopcroft-Karp algorithm for directed graphs to identify the unmatched nodes in the graph, which we call the *driver nodes*. This is implemented using the open-source `HopcroftKarp` package in `python` taking as input a bipartite matrix in dictionary representation for all graph types. In Fig. 3.3, we show the driver nodes highlighted in green for two small graphs. Our testing confirms that the unmatched nodes are identified and counted correctly.

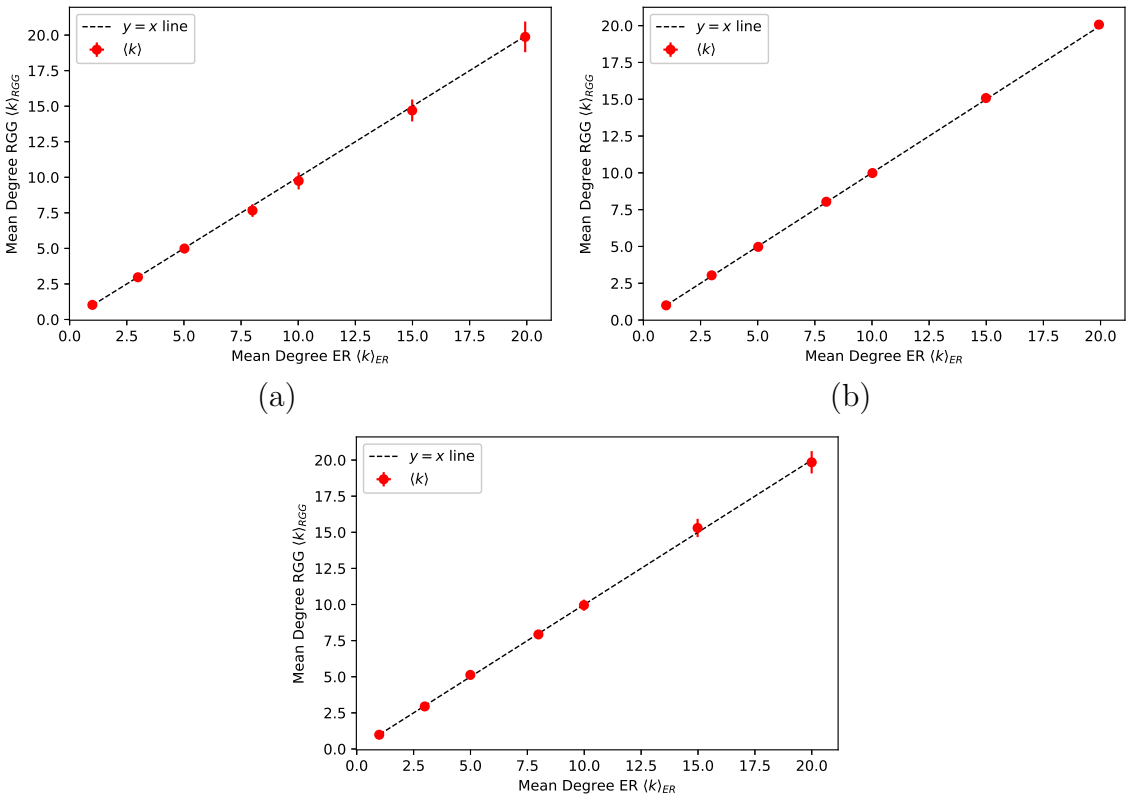



Fig. 3.2 All ensemble values are averaged over 100 samples of size $N = 1000$ with $d = 9$. All expected values for $\langle\kappa\rangle$ match that of the ER ensemble with the same parameters. This test validates our work flow for producing graph ensembles. The RGG types plotted are (a) SRGG (b) PRGG (c) GRGG.

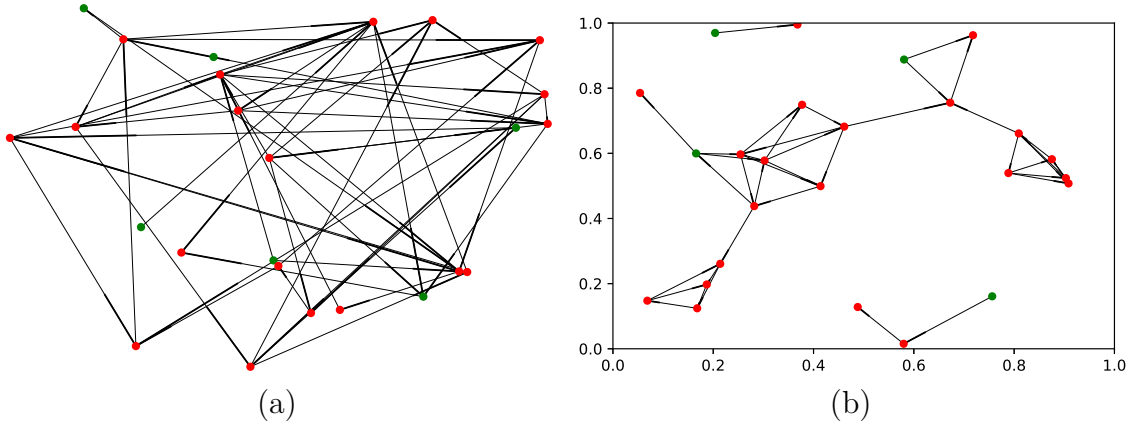



Fig. 3.3 (a) ER and (b) SRGG each with $\langle\kappa\rangle = 4$ and $N = 25$. Driver nodes are shown in green. We see that the unmatched nodes agree with our theoretical understanding of driver nodes for directed graphs.

## 3.2 LCC Extraction

For our purposes in this study, we find it necessary to deal with the largest connected components (LCCs) of our simulated graphs. We implement a `Cython`

function developed by Hoffman which utilizes a standard node walking algorithm to find all of the connected components in the graph and return them in order of size [39]. A comparison of what is tracked before and after applying the extraction is shown in Fig. 3.4. In Fig. 3.5, the calculation of $\langle\kappa\rangle$ is confirmed for the three graph types. For ensembles with low values of $\langle\kappa\rangle$, the empirical average value of $\langle\kappa\rangle$ is higher when extracting the LCC, as the subcomponents are of lower average $\langle\kappa\rangle$ and make up a significant portion of the sample. Thus, below a certain value for $\langle\kappa\rangle$, samples are composed of a large number of small components rather than a single network as we see for greater values of $\langle\kappa\rangle$. There is a minimum value for $\langle\kappa\rangle$ such that a large majority of the sample is connected in a single network. In this work, we are interested in studying properties of large RGGs rather than a disconnected group of small RGGs, and thus we need to choose values of $\langle\kappa\rangle$ appropriately to both generate samples with the desired average $\langle\kappa\rangle$ for the ensemble and contain a large network for study. These subtleties are discussed further in the next chapter.

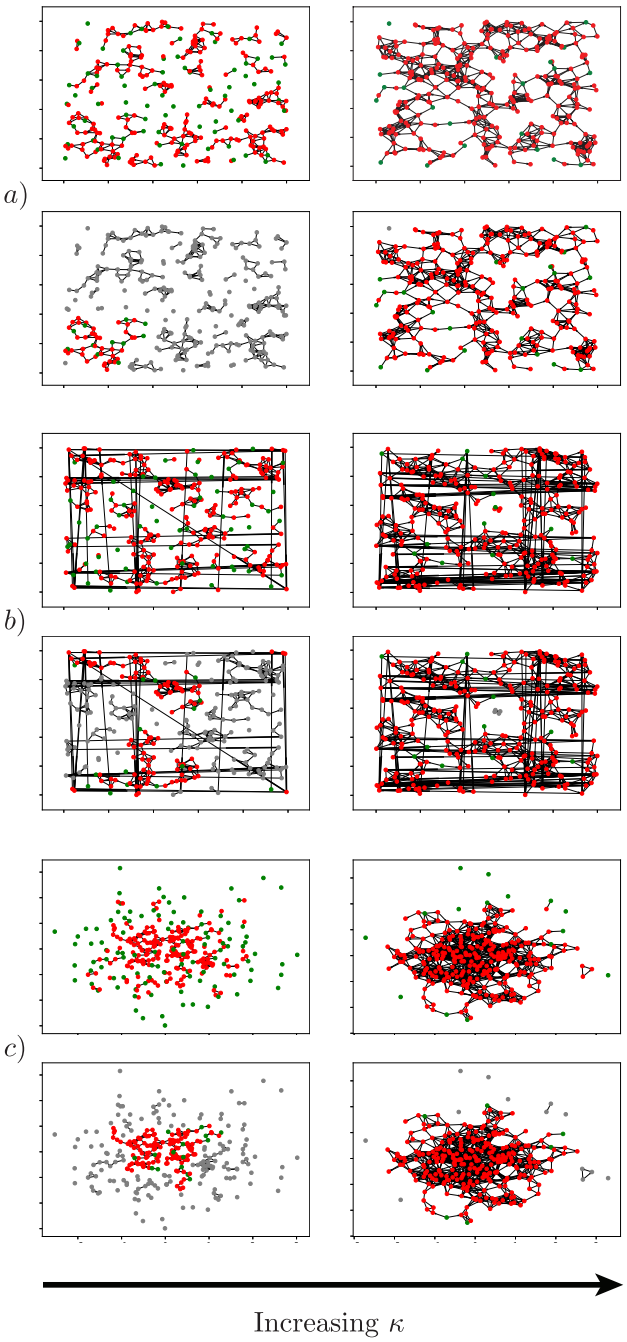



Fig. 3.4 The graphs in the left column are for $\langle\kappa\rangle$ values where there is not a large component in the sample, while the graphs in the right column are near a value of $\langle\kappa\rangle$ such that largest connected component (LCC) of the sample comprises at least 90% of the sample nodes. For (a) SRGGs (b) PRGGs and (c) GRGGs, the lower row shows nodes colored gray to depict which nodes would not be included in analysis after applying LCC extraction for the sample. That is, the nodes in color are the LCC, with driver nodes shown in green.

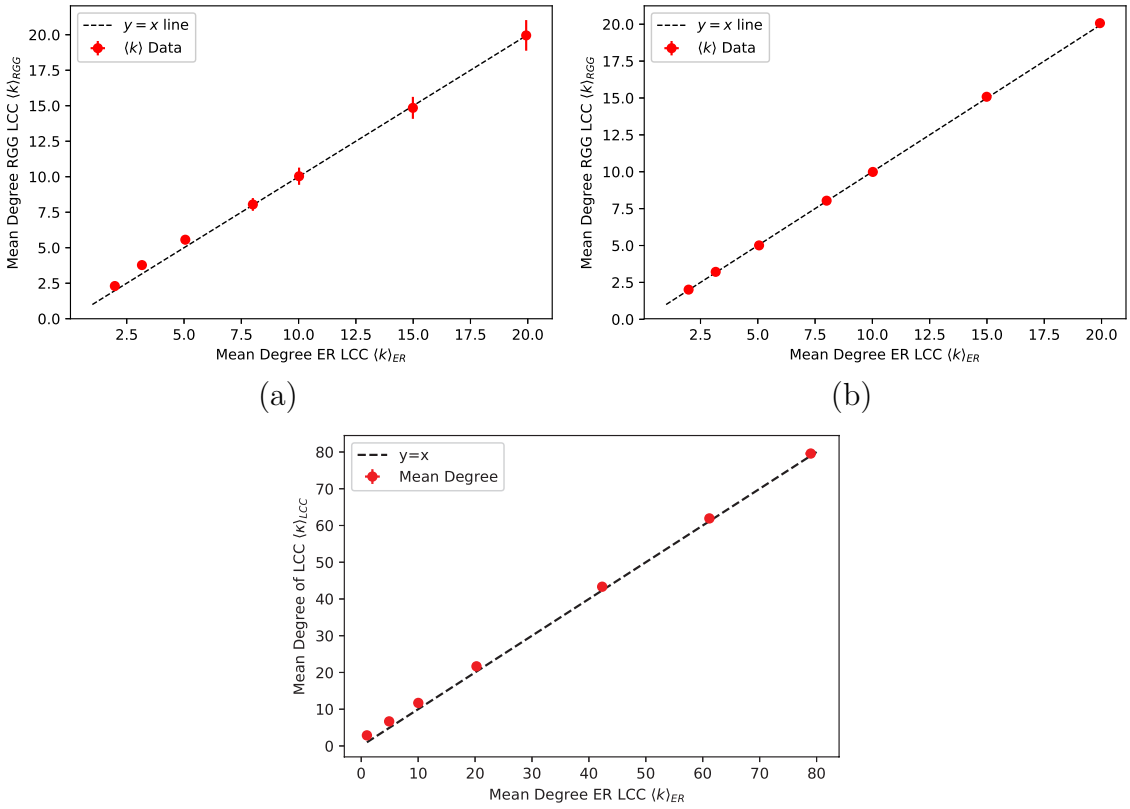



Fig. 3.5 All ensemble values are averaged over 100 samples of size $N = 1000$ with $d = 9$. The RGG types plotted are (a) SRGG (b) PRGG (c) GRGG. Note that for low values of $\langle\kappa\rangle$ across all ensemble types, the $\langle\kappa\rangle_{LCC}$ overestimates the corresponding target value. This is expected, as for low mean degree samples the LCC will not comprise a majority of the graph. Note that in (c) the values for GRGGs are much higher due to the low number of nodes within the LCC for $\langle\kappa\rangle$ below 20.

## 3.3 Orthogonal Distance Regression

We are interested in investigating the scaling of the driver node fraction $n_D$ with $\langle\kappa\rangle$ over a range of dimensions $d$ for RGGs. We compare these results to samples drawn from ER ensembles with the same parameter ranges. To extract scaling parameters, we identify relevant models and use a sound regression procedure. To choose a method of regression, we take into account the dispersion of our data over the parameter ranges.

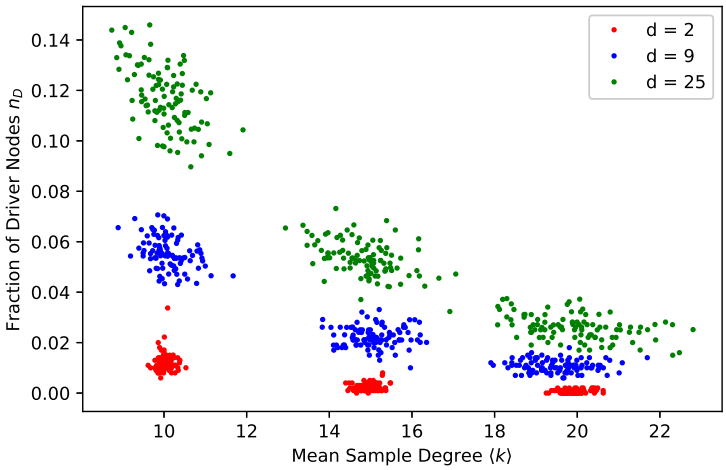



Fig. 3.6 Samples drawn from an SRGG ensemble for three dimensions $d = [2, 9, 25]$ and three values of $\langle\kappa\rangle = [10, 15, 20]$. The dispersion in values for $n_D$ do not vary greatly over $\langle\kappa\rangle$, while dispersion in $\langle\kappa\rangle$ varies in an approximately linear fashion. We use this behavior to choose the nonlinear orthogonal distance regression scheme for fitting models to data. The dispersion of the data increases with dimension, and this is discussed in the main text.

As shown in Fig. 3.6, when plotting the fraction of driver nodes $n_D$ over many graph samples at increasing $\langle\kappa\rangle$ the dispersion of the data points is additive in both variables, and the shape of the data is nonlinear. That is, we do not identify a large spread in the data associated with with our explanatory or response variables. This implies that our fitting problem is well-suited to Orthogonal Distance Regression (ODR) as opposed to a linear fit on a semilog scale [40]. ODR allows us to take into account variation in both $\langle\kappa\rangle$ and $n_D$, the explanatory and response variables, respectively. This is also known as an "errors-in-variables" method, and is a type of nonlinear least-squares regression. In this study, we fit two exponential models to the data we generate for decreasing driver node fraction $n_D$ as a function of $\langle\kappa\rangle$ . To implement this routine, we use the `FORTRAN` LINPACK version of ODR ported to a `scipy` library in `python` [41].

In this work we implement an exponential model,

$$\hat{n}_D = \beta e^{\gamma\langle\kappa\rangle}, \tag{3.1}$$

the motivations for which are discussed in the next chapter. The setup of the regression problem is such that

$$(X_i, Y_i), i = 1, \ldots, n \tag{3.2}$$

are random variables representing samples for $(\kappa, n_D)$ with true values

$$(x_i, y_i), i = 1, \ldots, n \tag{3.3}$$

and errors defined by

$$X_i = x_i - \delta_i \tag{3.4}$$

$$Y_i = y_i - \epsilon_i. \tag{3.5}$$

We identify our proposed model as a functional relationship $f$ with parameters $\beta$ and $\gamma$ such that

$$y_i = f(x_i, \beta, \gamma) \tag{3.6}$$

which can be written

$$Y_i = f(X_i + \delta_i, \beta, \gamma) - \epsilon_i. \tag{3.7}$$

Because we identify that our input and output variables have associated errors $\delta$ and $\epsilon$, respectively, we seek to minimize the orthogonal distance of these errors from our model, namely

$$r_i^2 = \min_{\delta_i, \epsilon_i} \{\delta_i^2 + \epsilon_i^2\} \tag{3.8}$$

subject to our proposed model $f$. In order to give more importance to the observations in our data that have higher precision in the sample set, we can weight these observations using the inverse of their sample standard deviations, or equivalently the inverse of their covariance matrices. Thus, our optimization problem is written

$$\min_{\delta_i, \epsilon_i} \sum_{i=1}^{n} w_i^2 \left\{ [f(X_i + \delta_i, \beta, \gamma) - Y_i]^2 + d_i^2 \delta_i^2 \right\} \tag{3.9}$$

where we have replaced $\epsilon_i$ using the model function definition in Eqn (3.7) and inserted weights, $w_i$ and $d_i$.

To extract the parameters $\beta$ and $\gamma$ in our analysis, we utilize bootstrapping to produce an estimate of the distribution over the fit parameters [42]. For each sample of driver node fraction, $n_D$, over the range of mean degree, $\langle\kappa\rangle$, we

resample the same number of points as the original sample with replacement. This resample is a uniform random sample drawn from the sample data. We then fit this resample using the ODR routine with weights set to unity. This process of resampling and fitting is repeated for 1000 iterations to produce a bootstrapped distribution over the fit parameters, $\beta$ and $\gamma$. These results are reported in the next section.

## 3.4 Coefficient of Variation

We use the coefficient of variation (CV) as a measure of statistical dispersion for the distribution of $n_D$ found when drawing samples from each of our ensembles of interest and is defined as

$$CV = \frac{\sigma}{\mu} \tag{3.10}$$

where $\sigma$ is the sample standard deviation and $\mu$ is the sample mean. We study the CV over dimension to compare the variation of $n_D$ over dimension for various graph types, where the mean values of each ensemble are different. Finding ensembles with low values for CV implies that the distribution is focused around a particular value. That is, low dispersion implies having more information about $n_D$ knowing only the ensemble parameters as the distribution is focused around a particular value for that ensemble. On the RGG sample level, a low CV indicates that by knowing the node placement in the embedding space, we have information about other quantities of interest such as $n_D$. We quantify the standard error of the CV to visualize its uncertainty as an estimate in the section. Here we derive the standard error of the CV using propagation of errors. In general, we have

$$R = R(X, Y, ...) \tag{3.11}$$

$$\delta R = \sqrt{\left(\frac{\partial R}{\partial X}\delta X\right)^2 + \left(\frac{\partial R}{\partial Y}\delta Y\right)^2 + ...} \tag{3.12}$$

$$\tag{3.13}$$

where $R$ is the function for which the error is of interest, and $\delta R$ is its error. Applied to the formula for $CV$,

$$\mathrm{CV} = \frac{\sigma}{\mu} = \frac{\mathrm{Var}^{\frac{1}{2}}}{\mu}, \tag{3.14}$$

where Var and $\sigma$ refer to the sample of interest. Thus, we use Eqn (3.13) to compute

$$\delta CV = \sqrt{\left(\frac{1}{2}\frac{\mathrm{Var}^{-\frac{1}{2}}}{\mu}\partial\mathrm{Var}\right)^2 + \left(\frac{\mathrm{Var}^{\frac{1}{2}}}{\mu^2}\partial\mu\right)^2} \tag{3.15}$$

$$\tag{3.16}$$

where $\delta$Var is the standard error of the variance and $\delta\mu$ is the standard error of the mean.

# Chapter 4

# Results

*Scientific discovery consists in the interpretation for our own convenience of a system of existence which has been made with no eye to our convenience at all.*

Norbert Wiener

## 4.1 Non-LCC Scaling

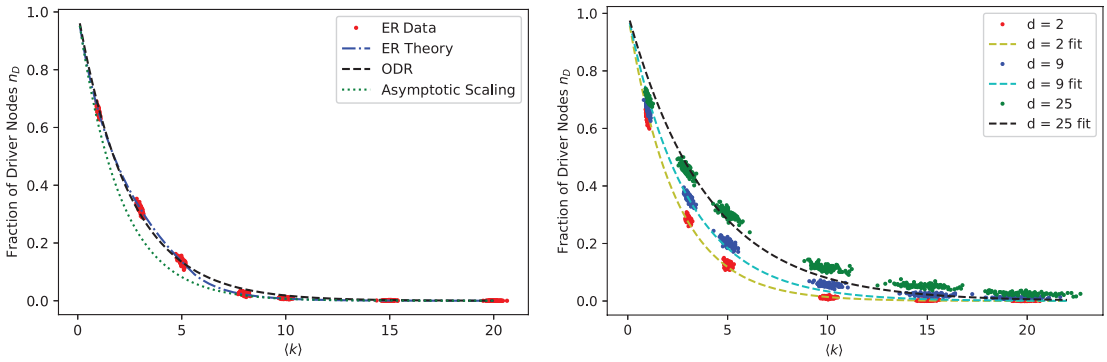
In order to validate our graph generation procedure, we compare our simulation results for SRGGs, PRGGs, and ER graphs to the self-consistent equations derived by Liu and Barabasi discussed in the second chapter. Following their analytical treatment, we use the single-parameter exponential model

$$\hat{n}_d = e^{\gamma\langle\hat{\kappa}\rangle}, \tag{4.1}$$

and fit using the ODR routine. We call this approach "Non-LCC Scaling" as it takes into account every node in the sample drawn from the graph ensemble. This includes all isolated nodes and small components if they exist, and thus is a measure not of the properties of a connected network but of the draw from the ensemble in its entirety. We display these results to highlight the difference in what is tracked using the single-parameter model and what is tracked using the model we present in the next section.

We see from these plots that our implementation of the Liu-Barabasi theoretical results match the ER data well for a range of $\langle\kappa\rangle$. This validates our methods for generating graphs and identifying driver nodes. However, the model fit does not capture the data at or above $\langle\kappa\rangle = 8$. Similarly, the single-parameter fit does not capture generated SRGG data at low or high $\langle\kappa\rangle$ for a range of dimensions. These results are shown in (a) of Fig. 4.1.

Despite the mismatch between the fits and the generated graph data, we extract the scaling parameter $\gamma$ for a range of dimension as shown in Fig. 4.2. Our findings suggest that both SRGGs and PRGGs have a faster driver node fraction decay than ER graphs of the same size. However, we will show that while there is a similar trend in the rates of decay over dimension, this result is erroneous. We now change our focus to analyze the largest connected components (LCCs) of graphs where this component is comprised of at least 90% of the nodes in the entire sample.




(a) (b)

Fig. 4.1 a) The raw ER graph data for $\langle\kappa\rangle = [1, 3, 5, 8, 10, 15, 20]$ are shown in red scatter, the Liu-Barabasi solution for their self-consistent equation solutions are shown in blue, the ODR fit for the ER data is shown in black, and the large mean degree limit for the exponential scaling ($\gamma = 0.5$) is shown in green. Notice that the ODR fit does not capture the data above $\langle\kappa\rangle = 5$ while the Liu-Barabasi result does. b) SRGG sample data for $N = 1000$ with 100 samples for each $\langle\kappa\rangle$ as used in (a) over $d = [2, 9, 25]$. ODR fits for the single-parameter model are shown for each dimension. Note that the single-parameter model underestimates the scaling parameter, $\gamma$, of $n_D$ over $\langle\kappa\rangle$ for each dimension.

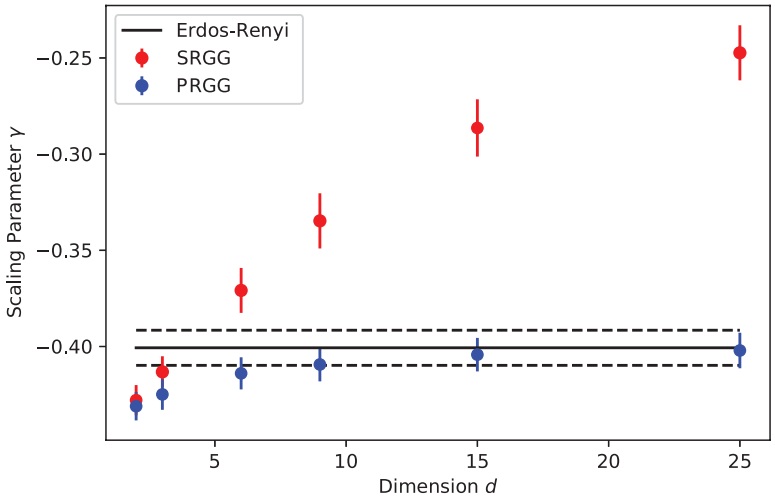



Fig. 4.2 The fit parameter $\gamma$ of the single-parameter model plotted over dimensions $d = [2, 3, 6, 9, 15, 25]$ for SRGGs (red), PRGGs (blue), and ER graphs (black). The fits were produced using the ODR routine for mean values over 100 samples with graph size N=1000 in all cases. Error bars report standard error for the fit parameter. Note that $n_D$ for RGGs seems to decay faster than ER graphs for low dimensions here, though this is due to the single-parameter fit used here to compare with prior literature for ER graphs. We clarify this result in the main text.

## 4.2 LCC Scaling

As far as we are aware, the prior literature has not made the distinction between the study of driver node fraction for the LCC and for the entire graph sample, that is, between LCC and Non-LCC scaling. Most studies follow Liu and Barabasi's treatment and track $n_D$ from $\langle\kappa\rangle = 0$. In this work, we are concerned with properties of a connected network in an ensemble, and thus extract the LCC from each graph sample after it is drawn from its ensemble.

First, we find the $\langle\kappa\rangle$ at which our graphs transition to having at least 90% of its nodes within the LCC in order to maintain the intended graph size attributed to the ensemble. Next, we investigate the threshold for this $\langle\kappa\rangle$ by drawing graph samples and computing the fraction of nodes in the LCC for a range of $\langle\kappa\rangle$ as shown in Fig. 4.3. We find that the graphs are well above the 90% point at $\langle\kappa\rangle = 10$ for SRGGs and PRGGs, while this point is approximately $\langle\kappa\rangle = 20$ for GRGGs. We will only use values of $\langle\kappa\rangle$ at or above these points when computing the scaling.

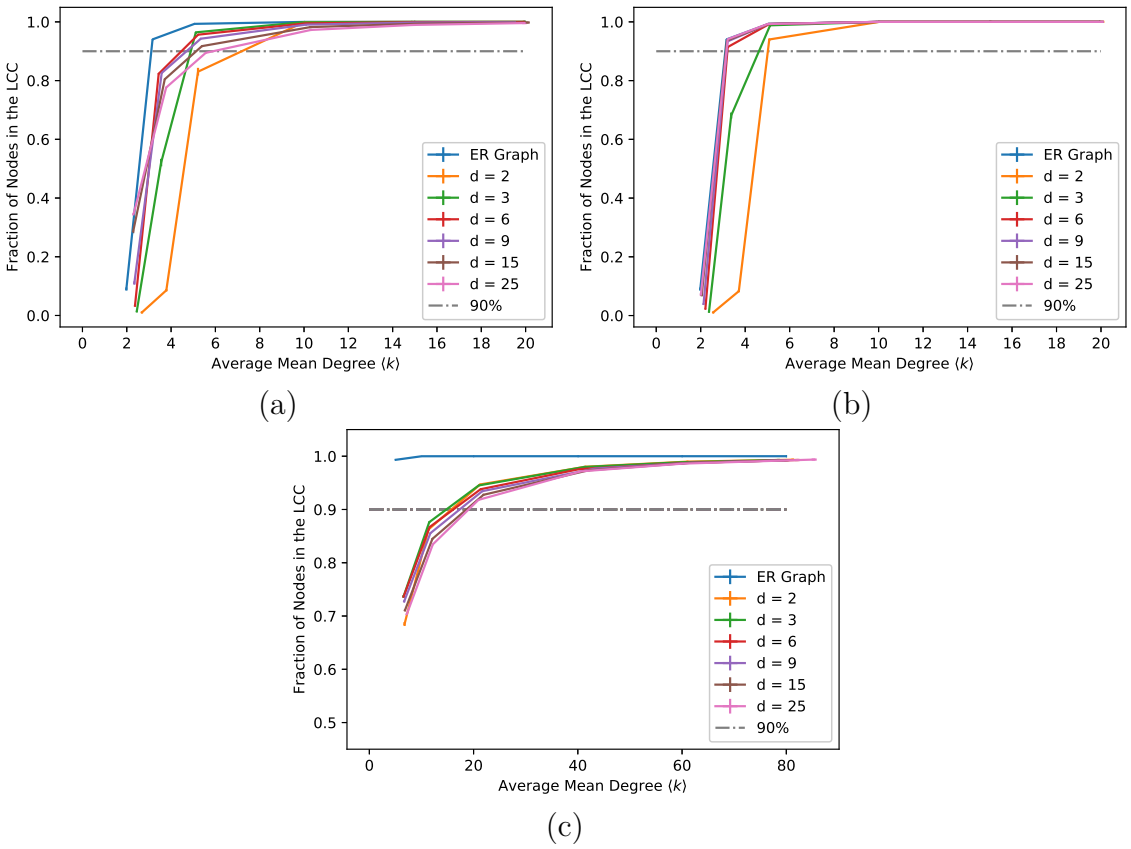
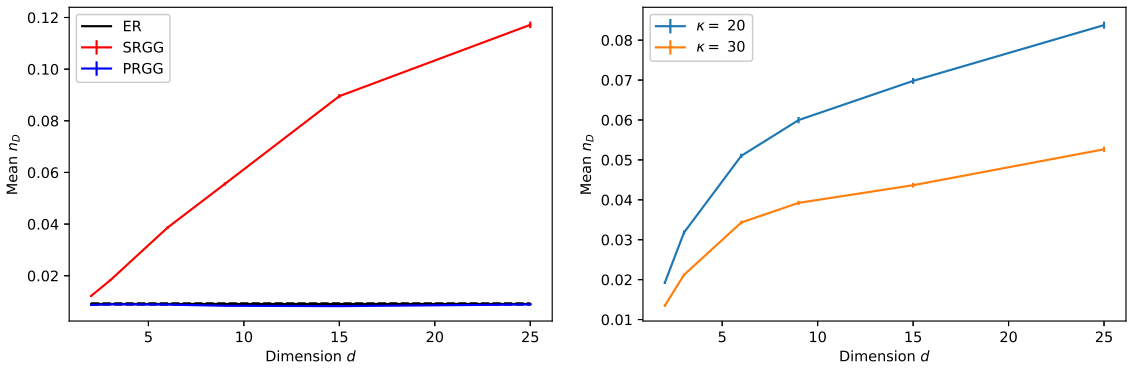



Fig. 4.3 We plot the average over 100 samples of the number of nodes in the LCC divided by the graph size for a range of dimensions $d$ for (a) SRGGs (b) PRGGs and (c) GRGGs. Note that the $\langle\kappa\rangle$ range for GRGGs is much higher than the uniform distribution ensembles. Error bars report standard error of the mean and are smaller than the thickness of the lines. We see that the point at which the LCC fraction is well above 90% is approximately $\langle\kappa\rangle = 10$ for SRGGs and PRGGs, and $\langle\kappa\rangle = 20$ for GRGGs. These are the lowest $\langle\kappa\rangle$ values we will use in our analysis.

### 4.2.1 Driver Node Fraction v. Dimension

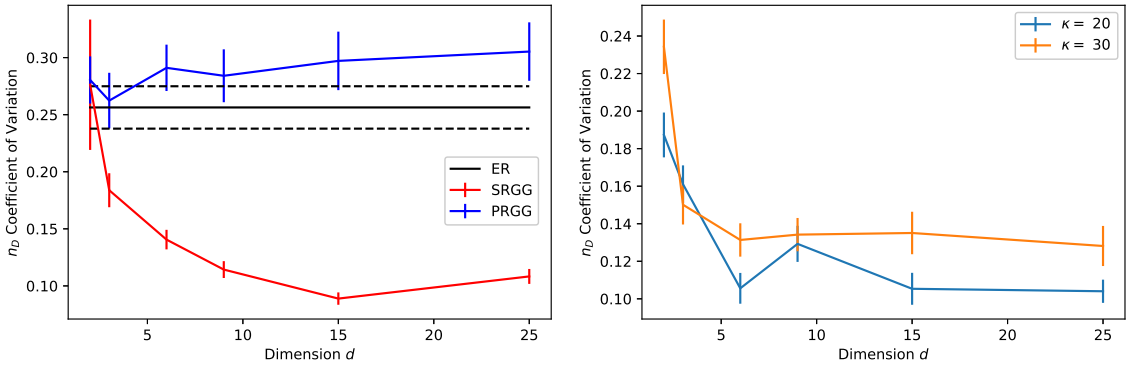
To begin our analysis of the driver node fraction $n_D$, the fraction of driver nodes for the LCCs of RGGs and ER graphs, we track $n_D$ for the lowest allowable $\langle\kappa\rangle$ value we identified in the previous section over dimension. We find that $n_D$ for SRGGs increases with dimension while $n_D$ for PRGGs is very close to that of ER graphs for all dimensions. For the same analysis of GRGGs, we find behavior similar to that of SRGGs, an increase of $n_D$ with dimension. However, GRGGs seem to increase more rapidly from low dimensions to higher dimensions. Both SRGGs and GRGGs have an appreciably higher number of driver nodes than both PRGGs and ER graphs at high dimensions, while at $d = 2$ all graph types are roughly 1.5% driver nodes on average for $\langle\kappa\rangle = 10$. These results are shown in Fig. 4.4.




(a) (b)

Fig. 4.4 All points shown are averaged over 100 samples from ensembles of size N=1000 with error bars reporting the standard error of the mean. a) The mean driver node fraction at $\langle\kappa\rangle = 10$ for SRGGs (red), PRGGs (blue), and ER graphs (black) over $d = [2, 3, 6, 9, 15, 25]$. We see that PRGGs covary with ER graphs strongly while SRGGs are close to PRGGs and ER graphs at $d = 2$ but quickly increase with dimension. b) The mean driver node fraction for GRGGs over the same dimension range as (a) for $\langle\kappa\rangle = 20$ (blue) and $\langle\kappa\rangle = 30$ (orange). We see similar behavior as that of SRGGs over dimension, and a decrease in $n_D$ as $\langle\kappa\rangle$ increases, as we would expect: when the samples become more well-connected, they are easier to control.

In addition to studying the mean $n_D$ over $d$ for three types of RGGs, we are interested in the dispersion of the $n_D$ distribution, created by taking draws from each RGG ensemble. We track the dispersion of the $n_D$ distribution using the coefficient of variation (CV) as discussed in the second chapter. We find that SRGGs have a lower CV at higher dimensions that PRGGs or ER graphs, thus knowing the node locations of the SRGG gives more information about the $n_D$ of these high-dimensional graph ensembles that the same PRGG or ER ensemble. We find that PRGGs have a higher CV than ER graphs with dimension, thus in high dimension the node locations give us less information about the $n_D$ for PRGGs than ER or SRGG graphs. GRGGs exhibit similar behavior as SRGGs. We will discuss these findings further in the next chapter and relate them to the degree distribution of driver nodes.




(a) (b)

Fig. 4.5 All points shown are averaged over 100 samples from ensembles of size N=1000 over $d = [2, 3, 6, 9, 15, 25]$. Error bars report the standard error of the mean. a) The coefficient of variation (CV) for SRGGs (red), PRGGS (blue), and ER graphs (black) at $\langle\kappa\rangle = 10$. The $n_D$ distribution of SRGGs becomes less dispersed as dimension increases, implying that at higher dimension we have better knowledge of $n_D$ than for lower dimensions. PRGGs show the inverse behavior: at higher dimensions we know less about $n_D$. Note that at $d = 2$ SRGGs, PRGGs, and ER graphs have similar CVs. b) The CV for GRGGs is reported for the same parameters as in (a) at $\langle\kappa\rangle = 20$ (blue) and $\langle\kappa\rangle = 30$ (orange). GRGGs display similar behavior as SRGGs. Note that as $\langle\kappa\rangle$ increases, the $n_D$ distribution becomes more dispersed.

### 4.2.2 Driver Node Fraction v. Mean Degree

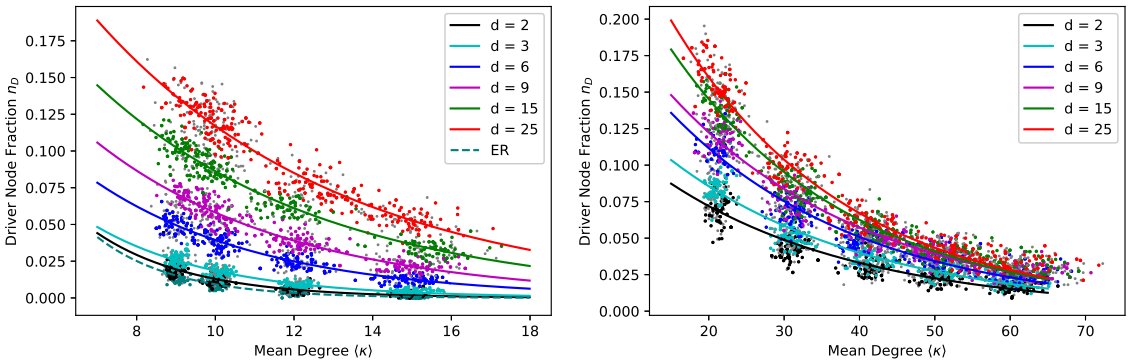
To extract useful parameters describing the driver node fraction $n_D$ over ranges of $\langle\kappa\rangle$ and $d$ for comparison, we use the ODR with bootstrapping procedure described in the third chapter to extract scaling parameters for $n_D$ over $\langle\kappa\rangle$ using the two-parameter exponential model

$$\hat{n}_D = \beta e^{\gamma\langle\hat{\kappa}\rangle}. \tag{4.2}$$

The model fits and raw data is shown in Fig. 4.6. By extracting these parameters, we compare a point value for each range of ensembles over $\langle\kappa\rangle$. We plot the decay rate parameter $\gamma$ in Fig. 4.7 and the intercept parameter $\beta$ in Fig. 4.8. Using this model, we are not held to the assumption that at $\langle\kappa\rangle = 0$ the driver node fraction, $n_D$, must equal unity, for this is below of our range for $\langle\kappa\rangle$. While we report the intercept parameter for completeness, we note that it does not have a great amount of efficacy in interpretation, and is mostly a byproduct of the fitted model. With two parameters, we are better able to extract an accurate decay rate $\gamma$, which in turn allows for more useful interpretation of trends.

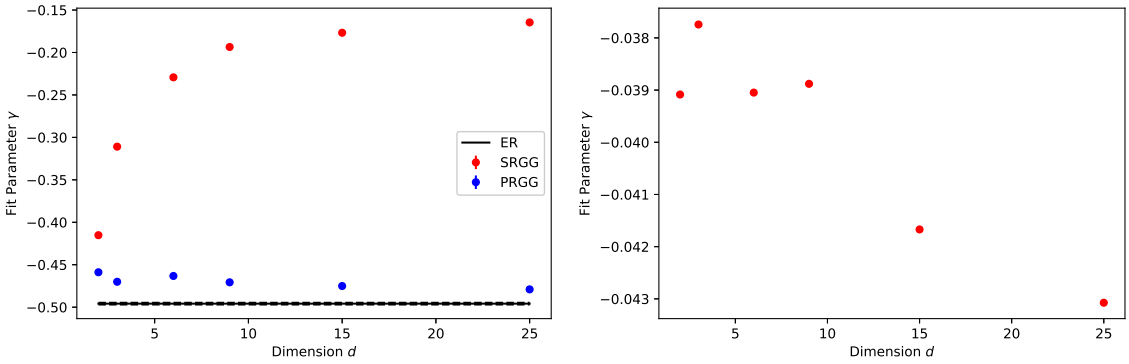
The general trends we find in $\gamma$ are similar in shape to that of the Non-LCC Scaling analysis. However, the relative values of SRGGs, PRGGs, and ER graphs for the scaling parameter are different. We see that SRGGs decay less quickly

as dimension is increased, and thus we expect high dimensional SRGGs to have more driver nodes than PRGGs and ER graphs of similar parameters over $\langle\kappa\rangle$. We find an opposite trend for the decay of $n_D$ for GRGGs, however. As dimension increases above $d = 10$, $n_D$ decays more quickly over $\langle\kappa\rangle$ with dimension. Overall, RGGs show slower scaling than SRGGs, PRGGs, and ER graphs by an order of magnitude.



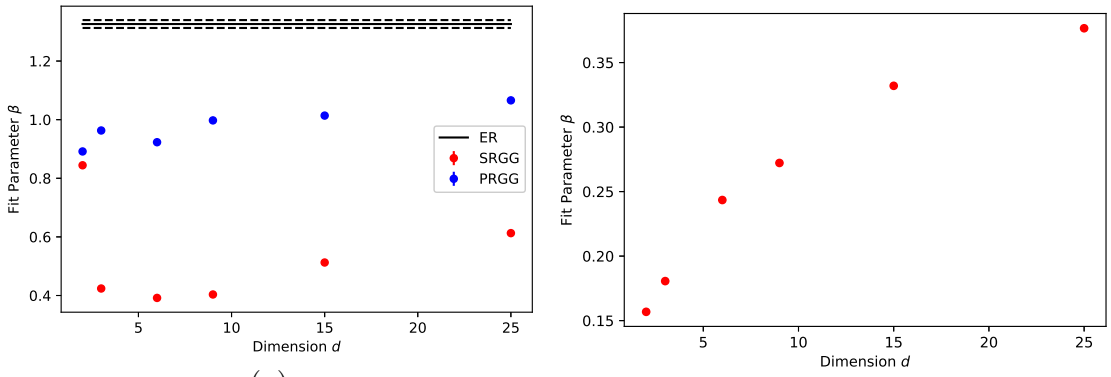

(a) (b)

Fig. 4.6 (a) The raw $n_D$ data for SRGGs and ER graphs with 100 samples for each ensemble is plotted where colored markers are an example of a single bootstrap resample set, and unused data points from the original sample are shown in gray. The solid lines are two-parameter exponential fits using the mean parameter values found by bootstrap resampling for 1000 iterations. Note that as dimension increases, the $n_D$ increases over the range of $\langle\kappa\rangle$. PRGGs are not shown because they covary strongly with ER and thus are indistinguishable in the plot. (b) The same procedure as in (a) is used for GRGGs, though with an increased $\langle\kappa\rangle$ range. Note that as dimension increases, the decay of $n_D$ becomes steeper, and the curves become very close at high values of $\langle\kappa\rangle$.




(a) (b)

Fig. 4.7 The scaling parameter $\gamma$ extracted through the two-parameter model fitting procedure is plotted over dimension $d$ for SRGG, PRGG, GRGG, and ER graph ensembles all with N=1000 over 100 samples. The values correspond to the curves shown in Fig. 4.6. (a) We see a marked difference in the parameter values for SRGGs and PRGGs. The value of $\gamma$ for SRGGs moves away from the PRGG and ER value, increasing rapidly from low dimension. At high dimensions, the scaling parameter begins to asymptote to a value near -0.15. The value of $\gamma$ for PRGGs, however, decreases slightly over dimension, moving closer to the value for ER graphs which is very near the scaling predicted by Liu and Barabasi in their analytical treatment for Non-LCC Scaling. (b) The decay of $n_D$ described by $\gamma$ for GRGGs decreases towards higher dimension, and all values are an order of magnitude smaller than that of SRGGs.




(a) (b)

Fig. 4.8 The intercept parameter $\beta$ for the model in Eqn (4.2) shown fitted to the data in Fig. 4.6. Error bars for standard error of the mean are smaller than the marker size. (a) Due to the slower decay rate of $n_D$ for SRGGs, we expect the intercept to be lower than that of PRGGs and ER graphs as shown. For $d = 2$ the intercept for SRGGs and PRGGs is nearly the same, coinciding with $\gamma$. ER graphs have a higher $\beta$ as we expect given their fast decay rate. (b) GRGGs, as with $\gamma$, show the inverse trend. As dimension increases, the intercept increases since the decay rate becomes faster. Note that the scale of the $\beta$ axis is lower than that of even SRGGs, which we expect given the slow decay of $n_D$ for GRGGs compared to the other graph types studied.

### 4.2.3 Finite Size Effects

After producing results for the scaling of $n_D$ for RGGs over dimension, we would like to investigate how the finite size of the networks might affect the measured scaling parameters. To do so, we sampled three SRGG ensembles an order of magnitude larger than the SRGG ensembles used previously in our analysis, $N = 10^4$ opposed to $N = 10^3$. We repeated the same ODR bootstrapping routine shown in Fig. 4.6 for the larger ensembles, and the results are shown in Fig. 4.9. We find that the increase in the size of the samples for SRGGs leads to more pronounced scaling behavior, but the overall trend does not change. We find that for low dimensions, the SRGG $\gamma$ is closer to that of the PRGG and ER scaling parameters for the larger ensemble, and in high dimension the rate of decay for $n_D$ is seen to be slower than the smaller samples.

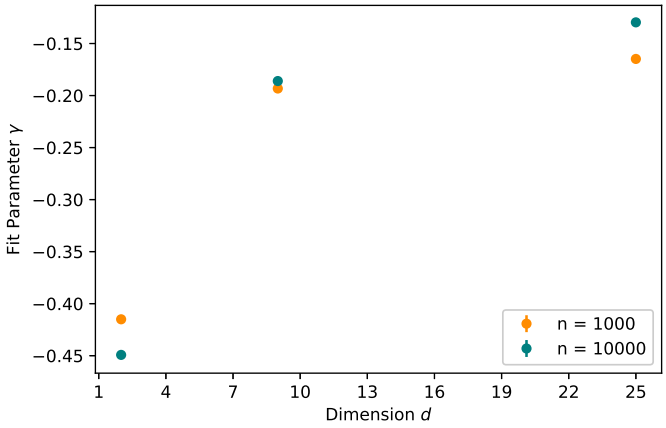


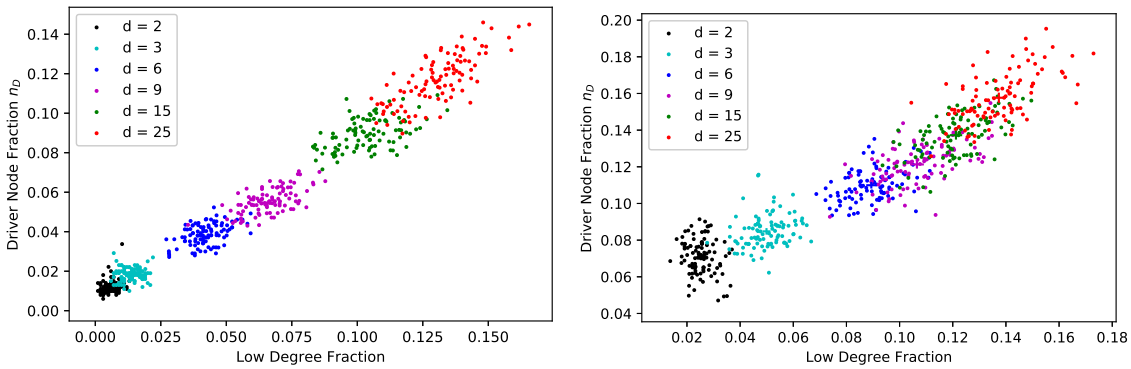

Fig. 4.9 We repeat the scaling procedure for larger SRGG ensembles to investigate finite size effects. Each ensemble value used in the scaling was averaged over 100 samples and bootstrapped for 1000 iterations. Error bars for standard error of the mean over $\gamma$ are smaller than the marker. Note that as $N$ grows larger in an SRGG, it tends to depress the $n_D$ decay rate $\gamma$ for lower dimensions, and elevate it for higher dimensions. This implies that increasing $N$ makes the scaling behavior more extreme across dimension. Plots shown over two orders of magnitude, $N = 10^3$ (orange) and $N = 10^4$ (blue).

### 4.2.4 Low Degree Nodes

There is evidence in the literature that 1- and 2-degree nodes play a key role in the number of driver nodes for networks with power-law degree distributions [43]. To investigate the possibility of similar behavior for SRGGs and GRGGs, we compute the fraction of these "low-degree" nodes in the LCCs of SRGGs and GRGGs at $\langle\kappa\rangle = 10$ and $\langle\kappa\rangle = 20$, respectively. Since we are interested in whether this fraction is correlated with the driver node fraction, we record $n_D$ for the same samples and plot them together as shown in Fig. 4.10. We find strong

linear correlations between low-degree node fraction and the driver node fraction for SRGGs and GRGGs with Pearson correlation coefficients 0.984 and 0.937, respectively, with two-sided p-values lower than machine precision.

These correlations suggest that as dimension increases, the degree distribution of driver nodes shifts to lower degree. To investigate this suggestion, we compute degree distributions for SRGG and GRGG ensembles with the same parameters used in the correlations by recording the mean degree of each driver node for 500 samples. The histograms are shown in Fig. 4.11. Indeed, we see that as dimension increases, the driver node degree distribution shifts towards values of lower degrees, leaving a long tail towards higher degrees. At lower dimensions, most of the mass is centered around the mean degree of the graph LCC, while in higher dimension a considerable amount of mass has shifted to lower degrees and a long tail has formed.




(a) (b)

Fig. 4.10 (a) The driver node fraction $n_D$ for SRGG ensembles with 100 samples over a range of dimensions at $\langle\kappa\rangle = 10$ plotted against the fraction of low-degree nodes in the samples: nodes with degree one or two. There is a positive correlation between $n_D$ and the low-degree fraction with Pearson correlation coefficient 0.984. Note that the values of each fraction are similar in magnitude across $d$ as well. (b) The same plot for GRGGs at $\langle\kappa\rangle = 20$ with Pearson coefficient 0.937. Note that the magnitude of the low-degree node fraction is less than that of the driver node fraction across dimension.

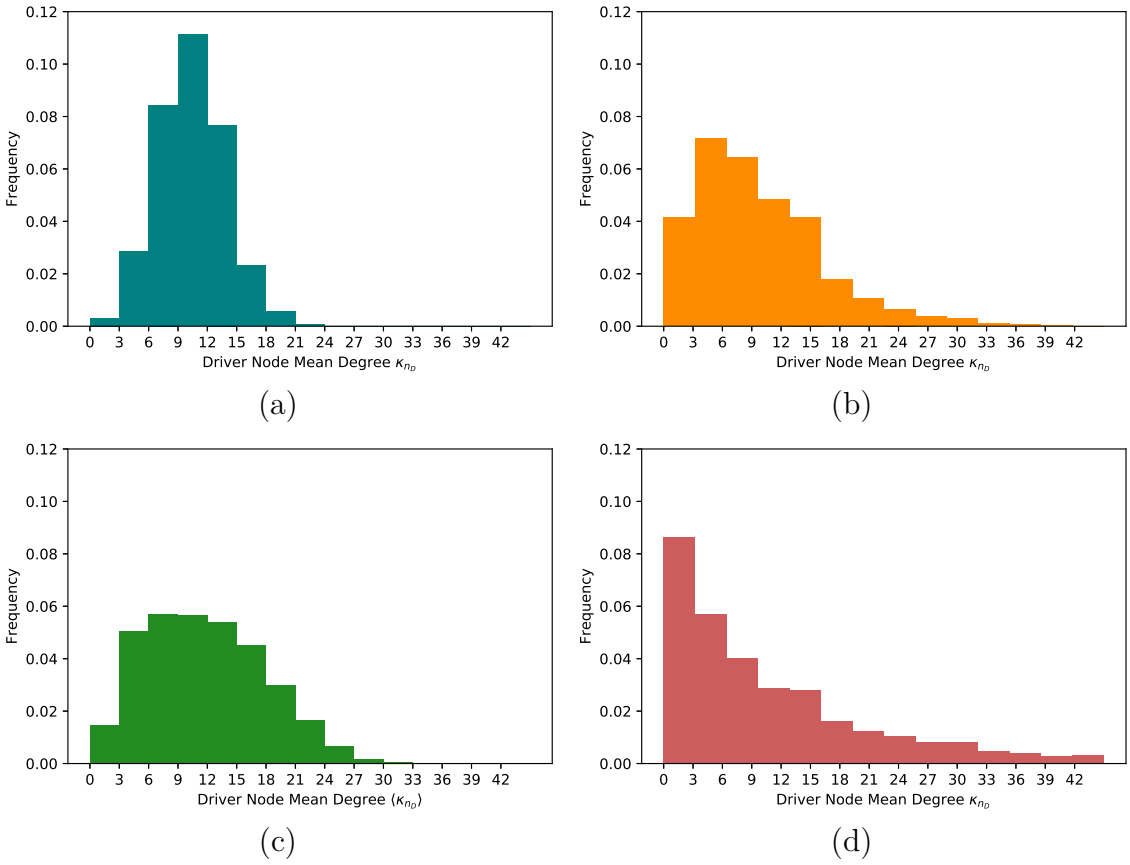



Fig. 4.11 Histograms for the driver node degree distributions over 500 samples of each ensemble. (a) SRGG ensemble for $d = 2$ and $\langle\kappa\rangle = 10$ with a mean value of 9.93±0.04. The distribution has a tight Gaussian shape around its mean, without tails. b) SRGG ensemble for $d = 9$ and $\langle\kappa\rangle = 10$, for which we see a tail develop towards higher degree nodes and more mass around low-degree driver nodes with mean 9.80±0.04 (c) GRGG ensemble for $d = 2$ and $\langle\kappa\rangle = 20$ with a slight tail towards higher degree driver nodes and mean 11.31±0.02 (d) GRGG ensemble for $d = 9$ and $\langle\kappa\rangle = 20$ with a long tail towards higher degree and a large amount of low-degree driver nodes. The mean is 11.76±0.038. Note that we see similar behavior moving from low to high dimension, but that the GRGGs are more pronounced in the development of long tails.

### 4.2.5 Control Profiles

As discussed in the second chapter, we can categorize the control nodes as sources, external dilations, and internal dilations. The fractions of nodes in the graph LCC for these categories are $n_D^s$, $n_D^e$ and $n_D^i$, respectively, and together are called the control profile [26]. In order to further investigate how the controllability of RGGs is related to dimension, we plot the control profile using the Zen `Python` library as shown in Fig. 4.12 [44]. By normalizing each plot by the number of driver nodes and mapping to a color spectrum, we can visualize the density of the control profile in a heatmap as shown in Fig. 4.13. For all graphs reported, SRGG, PRGG, GRGG, and ER, we see from these two sets of plots that the majority of

the nodes are source nodes, though an appreciable number of driver nodes are internal dilations for SRGGs, PRGGs, and ER ensembles. Interestingly, many of the graphs show zero external dilation nodes. This array of results will take further investigation to fully explain, though we offer a basic interpretation in the concluding chapter.

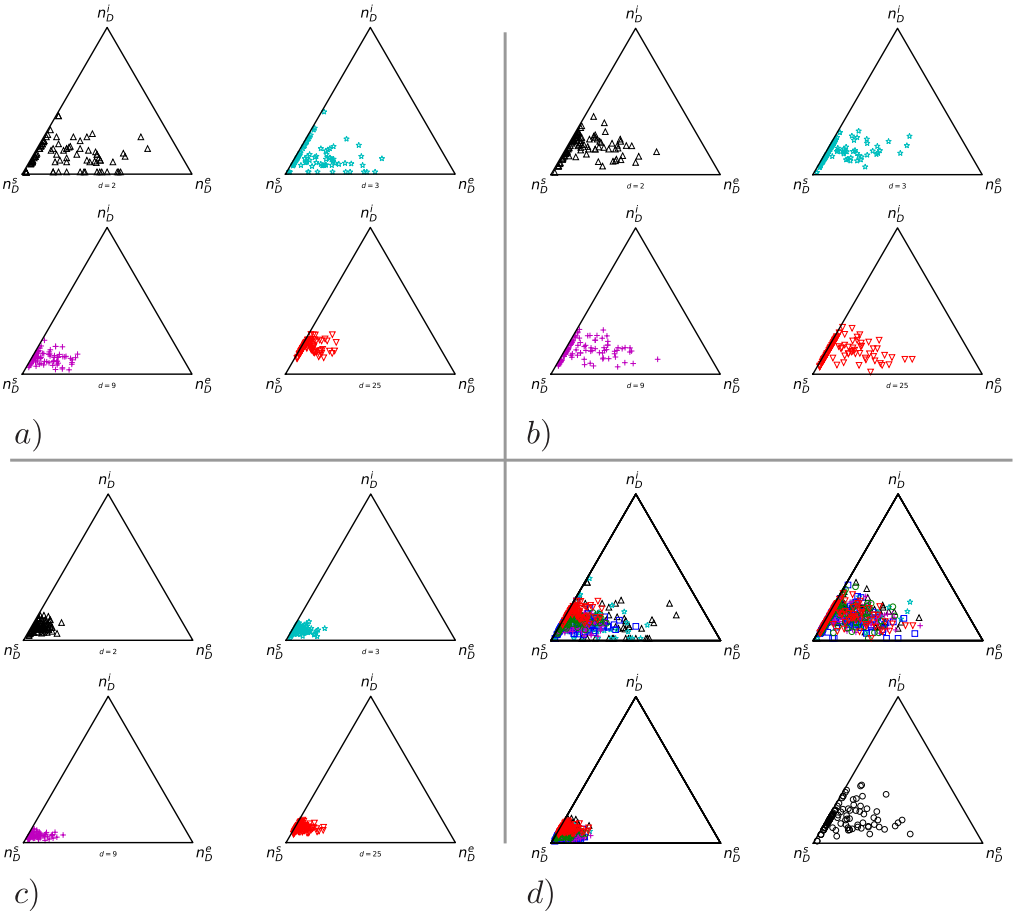



Fig. 4.12 Ternary plots of the raw control profile data for (a) SRGGs (b) PRGGs and (c) GRGGs. Each of the four triangles within a square is a different dimension. Reading top to bottom, left to right, the triangles show $d = 2$, $d = 3$, $d = 9$, and $d = 25$. (d) Reading top to bottom, left to right, these four triangles show SRRGs, PRGGs, GRGGs, and ER graph ensembles with the data for all dimensions plotted together. While there is a range of samples for each graph, all types tend to be source- and internal dilation-dominated.

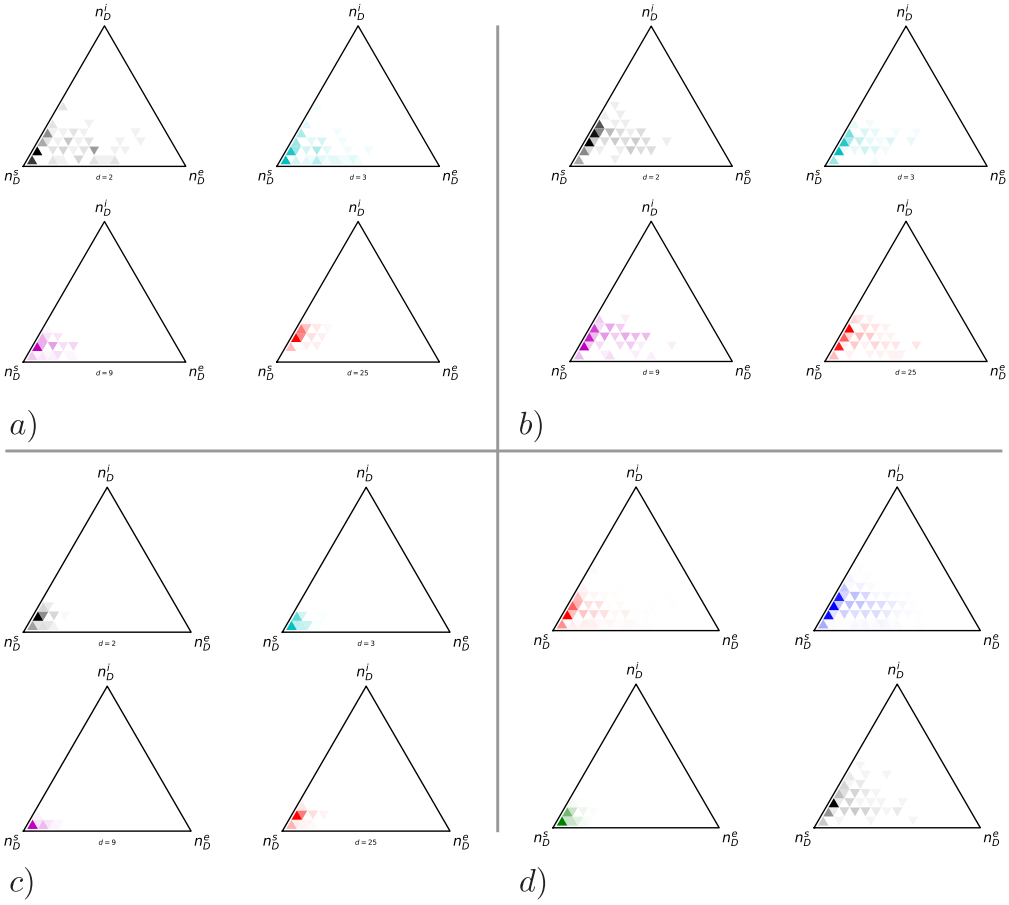



Fig. 4.13 Ternary heat maps of the control profile data for (a) SRGGs (b) PRGGs and (c) GRGGs. Each of the four triangles within a square is a different dimension. Reading top to bottom, left to right, the triangles show $d = 2$, $d = 3$, $d = 9$, and $d = 25$. (d) Reading top to bottom, left to right, these four triangles show SRRGs, PRGGs, GRGGs, and ER graph ensembles with the data for all dimensions plotted together.The heatmap allows us to assess where the majority of the samples lie, from which it is more clear that all graphs are source- and internal dilation-dominated, while GRGGs in particular are almost exclusively dominated by source nodes.

# Chapter 5

# Conclusions

*There are no answers, only cross references.*
Norbert Wiener

## 5.1 Discussion of Key Findings

In our comparison with the Liu-Barabasi model for controllability, we highlight the fact that computing $n_D$ for the entire graph sample from each ensemble leads to different scaling behavior over dimension as opposed to tracking $n_D$ for the LCC only with values of $\langle\kappa\rangle$ over the 90% point. Neither case can be said to be correct or incorrect, but here we are interested in the controllability of a connected network, not a disconnected set of graphs which. Our results show that when tracking the LCC, $n_D$ decays less rapidly over $\langle\kappa\rangle$ for all dimensions when compared to tracking entire graph samples. This is not, however, a result interpretable using intuitions about the graphs themselves, but evidence of the differences in the one- and two-parameter models used to capture the scaling behavior. When studying the entire graph sample over $\langle\kappa\rangle$, we see $n_D$ decay more rapidly as $\langle\kappa\rangle$ increases from low values, as the network becomes more connected. When tracking $n_D$ for the LCC at or above a $\langle\kappa\rangle$ for which the graph is 90%-connected using the two-parameter model, the fit parameters track the fraction of nodes in the LCC without isolates and small connected components. Thus, the latter model is more informative about the scaling for the connected network itself rather than the decay of isolates and small components. That said, our reported scaling for the LCCs of ER graphs ($-0.496 \pm 0.001$) is very close to that of the Liu-Barabasi theoretical scaling for ER graphs in the large mean degree limit (-0.5). This implies that our two-parameter model accurately captures the scaling behavior of $n_D$ for adequate values of $\langle\kappa\rangle$, which we take to mean above the 90% point.

Looking at the LCCs of SRGGs, we find that their average $n_D$ at $\langle\kappa\rangle = 10$ is greater than that of PRGGs and ER graphs and increases with dimension. This holds for $\langle\kappa\rangle$ greater than 10 until $n_D$ goes to zero. We find that the decay rate of $n_D$ for the LCCs of SRGGs is less rapid for all dimensions compared to the decay rate of $n_D$ for PRGGs and ER graphs over $\langle\kappa\rangle$. However, for $d = 2$ SRGGs are close to PRGGs and ER graphs in both mean $n_D$ and $\gamma$. Based on these results, we expect to find more driver nodes for SRGGs than for ER graphs, and expect that number to decay more slowly as the graphs become more connected. In other words, SRGGs demand a greater number of control inputs as dimension increases. We hypothesized that this is related to the degree distribution, specifically the number of low-degree nodes in the SRGG increasing with dimension due to boundary effect, as other research has suggested [36, 20]. Indeed, we find a strong positive correlation between the fraction of degree one and two nodes and $n_D$ with Pearson coefficient 0.984. As the literature has suggested for scale-free networks, it seems that these low-degree nodes are a good estimator for the driver node fraction of SRGGs [43]. We note that this result is true even for three dimensions, which may play a role in physical systems well-modeled by RGGs with solid boundaries.

To investigate the driver nodes further, we compute the control profile of the SRGGs to determine the breakdown of driver node type across dimension. We find that the driver nodes of SRGGs are dominated by source and internal dilation nodes for all dimensions, and that this behavior becomes more concentrated as dimension increases. This is in accordance with intuition and the prior results: source nodes and internal dilations will be greater than degree one, but as dimension more nodes are on the boundary of the hypercube, and the degree of these source and internal dilations will decrease as the degree distribution of all nodes shifts towards lower degree. Thus, control policies for SRGGs should be similar to PRGGs and ER graphs in two dimensions, but should focus more on nodes with degree one and two as dimension increases. We note that based on the decrease in the CV with dimension for the $n_D$ distribution of SRGGs, there is significant information relayed by the node locations with respect to the driver node fraction as dimension increases. However, this information can be estimated by the low-degree nodes.

For PRGGs, the value of $n_D$ at $\langle\kappa\rangle = 10$ is nearly indistinguishable from that of ER graphs for all dimensions, and is only slightly above that of ER graphs for greater values of $\langle\kappa\rangle$ until $n_D$ goes to zero. Thus, $\gamma$ is slightly greater than that of ER graphs and does not vary with dimension. We argue that control policies for PRGGs, as such, would be similar to that of ER graphs for all dimensions. The periodic boundary condition leads to uniform behavior for $n_D$ over dimension and allows PRGGs to be controlled more easily that SRGGs, supporting our claim

that boundary effects play a crucial role in the number of driver nodes in RGG models.

For GRGGs, we find a very slow decay of $n_D$ over dimension, nearly an order of magnitude less than that of ER graphs. As opposed to SRGGs, PRGGs, and ER graphs, the decay rate for GRGGs decreases with dimension, though to a lesser degree. Looking a the raw data, we see that for GRGGs increasing dimension does not appreciably change the value of $\langle\kappa\rangle$ for which $n_D$ goes to zero. Thus, as dimension increases the rate of decay for $n_D$ increases as the network becomes more connected. We claim that this difference from SRGGs is due to the outer ring of the GRGGs being low degree, and support this claim by reporting a strong positive correlation between $n_D$ and the fraction of low-degree nodes, with Pearson coefficient 0.937. However, the fraction of low-degree nodes is less than the fraction of driver node in low dimensions, implying that the number of driver nodes is mostly low-degree nodes only in high dimensions. We see this behavior for SRGGs as well, but it is more pronounced in the GRGGs due to the ring of low-degree nodes around the boundary compounded by effects due high dimension. Accordingly, studying the control profile for GRGGs we find that for all dimensions they are strongly source-dominated. This agrees with our intuition about low-degree nodes, as well as the degree distribution of the driver nodes. A feasible control policy for GRGGs would be to control the outer ring nodes for low dimensions, as these are more likely to be drivers. In high dimensions the number of driver nodes increases, but low-degree nodes become a good estimate for the driver nodes and should be controlled first.

To summarize, we find that boundary conditions and the underlying distributions are extremely important when considering control policies for RGGs. We provide supporting evidence for a link between the degree distribution, specifically the number of nodes with degree one and two, and the controllability of the LCC for SRGGs and GRGGs, though with different explanations. We find that PRGGs do not deviate appreciably from ER graphs with respect to controllability for any dimension.

## 5.2 Future Research Directions

Our plan for further work on the control theoretic properties of RGGs is threefold. We would like to develop a more robust assessment of control for RGGs to provide more exact control policies, create and compare new types of spatially embedded models stemming from our result presented here, and improve the scaling of our algorithms to generate larger graph samples. We elaborate on these directions below.

While tracking $n_D$ is the most appropriate first step in understanding a network's structural controllability, in order to develop a more holistic control assessment we need to define controllability using multiple metrics. As prior research has shown using biological networks, structure is not the only important characteristic for controllability [45]. The fraction of driver nodes $n_D$ is an underestimate, as it measures the number of unique inputs needed to control the entire network. We need to expand this notion of control to address accessibility and take the energetic cost of control into account, perhaps computing the controllability Gramian given the weighted coefficients of a system [46, 47]. In this work, we focus on the structure of RGGs and its relationship to controllability as this is the most general metric to establish a control policy before looking at specific systems. We hope to expand our notion of controllability in the future to account for more factors.

In order to assess which factors are important to track, the creation and comparison of new models with spatial embedding is important to better understand what contributes to the controllability of real-world systems. At the risk of adding the proverbial "kitchen sink" to network modeling, we suggest that the thoughtful addition of dimensionality and boundary conditions in models of spatially embedded systems will prove fruitful for predicting real-world phenomena. Some important factors in addition to spatial embedding that we plan to assess include the simulation of nodal dynamics by randomly assigning self-loops, the addition of long-range connections in spatially embedded graphs, and the use of general connection functions opposed to the hard cutoff used in this work [48–50]. We feel that these are the most important parameters, in this order, to study in addition to the ones studied here to further analyze the structural controllability of spatially embedded networks.

Additionally, we think it is important to focus attention on the internal dilation nodes, as they make up a significant fraction of the driver nodes for RGGs, and may play a role in the accessibility of the modeled system as others have suggested [26]. In this work, we are assigning edge direction randomly with probability $\frac{1}{2}$, but it is worth noting that perhaps by creating a redirection algorithm we might improve the structural controllability of the network. This technique may be of use in physical systems such as wireless sensor networks and should be studied [51].

That said, it is difficult to interpret the usefulness of such ideas for future study unless we can identify discrepancies between our models and the systems we are intending to simulate. For this reason, it is important that we work towards analyzing real-world network data with spatial embedding, and compare this analysis to the RGG models we have used here as well as the future models we have suggested above. Some researchers, specifically in the swarm robotics community,

have created artificial systems against which to compare their random graph models, and we suggest this might be an opportunity for fruitful collaboration, as it may prove useful for the control of autonomous devices [24, 52, 53].

Lastly, to experimentally study large networks we need to devise fast algorithms to compute meaningful statistics. The algorithms we used in this project are a first attempt at quickly sampling hundreds of graphs of size $N^3$, and worked well for our purposes here to make steps towards quantifying RGG controllability. Going forward, we would like to use a strongly typed language such as `C` or `Fortran` as these languages handle memory usage much more efficiently than `python`, though they do sacrifice readability. A middle ground is to use the `Cython` library for the entirety of our module, rather than small components. We would like to contribute our RGG sampling framework to major network research packages such as `Networkx` or `Zen` libraries in order to expand the number of tools we have available for use [44]. This is not out of reach, and will be a fruitful first step.

# References


[1] Bela Bollobas. The Evolution of Random Graphs. *Trans. Am. Math. Soc.*, 286(1):257, 1984. ISSN 00029947. doi: 10.2307/1999405.

[2] E. N. Gilbert. Random Graphs. *Ann. Math. Stat.*, 30(4):1141–1144, 1959. ISSN 0003-4851. doi: 10.1214/aoms/1177706098.

[3] Gilles Deleuze and Felix Guattari. *L'Anti-Oedipe*. Les Editions de Minuit, Paris, 1972. ISBN 2707300675.

[4] Albert-Laszlo Barabasi and Reka Albert. Emergence of scaling in random networks. *Science (80-. ).*, 286(October):509–512, oct 1999. ISSN 00368075. doi: 10.1126/science.286.5439.509.

[5] Sumeet Agarwal, Charlotte M. Deane, Mason A. Porter, and Nick S. Jones. Revisiting date and party hubs: Novel approaches to role assignment in protein interaction networks. *PLoS Comput. Biol.*, 6(6):1–12, 2010. ISSN 1553734X. doi: 10.1371/journal.pcbi.1000817.

[6] Albert-László Barabási and Zoltán N. Oltvai. Network biology: understanding the cell's functional organization. *Nat. Rev. Genet.*, 5(2):101–113, 2004. ISSN 1471-0056. doi: 10.1038/nrg1272.

[7] Stefano Battiston, J. Doyne Farmer, Andreas Flache, Diego Garlaschelli, Andrew G. Haldane, Hans Heesterbeek, Cars Hommes, Carlo Jaeger, Robert May, and Marten Scheffer. Complexity theory and financial regulation. *Science (80-. ).*, 351(6275):818–819, 2016. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aad0299.

[8] Yuri Mansury and J. K. Shin. Size, connectivity, and tipping in spatial networks: Theory and empirics. *Comput. Environ. Urban Syst.*, 54:428–437, 2015. ISSN 01989715. doi: 10.1016/j.compenvurbsys.2015.08.004.

[9] Aaron Sim, Sophia N. Yaliraki, Mauricio Barahona, and Michael P. H. Stumpf. Great cities look small. *J. R. Soc. Interface*, 12(109):20150315, 2015. ISSN 1742-5689. doi: 10.1098/rsif.2015.0315.

[10] Yong-Yeol Ahn, Sebastian E. Ahnert, James P. Bagrow, and Albert-László Barabási. Flavor network and the principles of food pairing. *Sci. Rep.*, 1 (196):1–7, 2011. ISSN 2045-2322. doi: 10.1038/srep00196.

[11] Ed Bullmore and Olaf Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nat. Rev. Neurosci.*, 10(4): 312–312, 2009. ISSN 1471-003X. doi: 10.1038/nrn2618.

[12] B. C. Coutinho, Sungryong Hong, Kim Albrecht, Arjun Dey, Albert-László Barabási, Paul Torrey, Mark Vogelsberger, and Lars Hernquist. The Network Behind the Cosmic Web. *arXiv.org*, pages 1–5, 2016.

[13] Richard F. Betzel, Shi Gu, John D. Medaglia, Fabio Pasqualetti, and Danielle S. Bassett. Optimally controlling the human connectome: the role of network topology. *Nat. Publ. Gr.*, pages 1–14, 2016. ISSN 2045-2322. doi: 10.1038/srep30770.

[14] Sarah Feldt Muldoon, Fabio Pasqualetti, Shi Gu, Matthew Cieslak, Scott T. Grafton, Jean M. Vettel, and Danielle S. Bassett. Stimulation-Based Control of Dynamic Brain Networks. *PLoS Comput. Biol.*, 12(9), 2016. ISSN 15537358. doi: 10.1371/journal.pcbi.1005076.

[15] Rutger Goekoop and Jaap G. Goekoop. A network view on psychiatric disorders:Network clusters of symptoms as elementary syndromes ofpsychopathology. *PLoS One*, 9(11):1–47, 2014. ISSN 19326203. doi: 10.1371/journal.pone.0112734.

[16] Evelyn Tang and Danielle S. Bassett. Control of Dynamics in Brain Networks. *arXiv.org*, 2017.

[17] Réka Albert and Albert László Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74(1):47–97, 2002. ISSN 00346861. doi: 10.1103/RevModPhys.74.47.

[18] Olaf Sporns. Structure and function of complex brain networks. *Dialogues Clin. Neurosci.*, 15(3):247–262, 2013. ISSN 12948322. doi: 10.1137/S003614450342480.

[19] Larry Roberts. ARPANET Sketch, 1969.

[20] Yang-Yu Liu, Jean-Jacques Slotine, and Albert-László Barabási. Controllability of complex networks. *Nature*, 473(7346):167–173, 2011. ISSN 0028-0836. doi: 10.1038/nature10011.

[21] Yang Yu Liu and Albert László Barabási. Control principles of complex systems. *Rev. Mod. Phys.*, 88(3):1–58, 2016. ISSN 15390756. doi: 10.1103/RevModPhys.88.035006.

[22] R E Kalman. Mathematical Description of Linear Dynamical Systems. *J.S.I.A.M. Control*, 1(2):152–192, jan 1963. ISSN 0887-4603. doi: 10.1137/0301010.

[23] Shi Gu, Fabio Pasqualetti, Matthew Cieslak, Qawi K. Telesford, Alfred B. Yu, Ari E. Kahn, John D. Medaglia, Jean M. Vettel, Michael B. Miller, Scott T. Grafton, and Danielle S. Bassett. Controllability of structural brain networks. *Nat. Commun.*, 6:8414, 2015. ISSN 2041-1723. doi: 10.1038/ncomms9414.

[24] Tyler H. Summers and John Lygeros. *Optimal Sensor and Actuator Placement in Complex Dynamical Networks*, volume 47. IFAC, 2014. ISBN 9783902823625. doi: 10.3182/20140824-6-ZA-1003.00226.

[25] Ching Tal Lin. Structural Controllability. *IEEE Trans. Automat. Contr.*, 19 (3):201–208, 1974. ISSN 15582523. doi: 10.1109/TAC.1974.1100557.

[26] J. Ruths and D. Ruths. Control Profiles of Complex Networks. *Science (80-. ).*, 343(6177):1373–1376, 2014. ISSN 0036-8075. doi: 10.1126/science.1242063.

[27] Tao Jia, Yang-Yu Liu, Endre Csóka, Márton Pósfai, Jean-Jacques Slotine, and Albert-László Barabási. Emergence of bimodality in controlling complex networks. (May), 2015. ISSN 2041-1723. doi: 10.1038/ncomms3002.

[28] R. Wollman. Counting the Ways to Decode Dynamic Signals. *Science (80-. ).*, 343(6177):1326–1327, 2014. ISSN 0036-8075. doi: 10.1126/science.1252247.

[29] JE Hopcroft and RM Karp. An n^(5/2) Algorithm for Maximum Matching in Bipartite Graphs. *SIAM J. Comput.*, 2(4):225–231, 1973. doi: https://doi.org/10.1137/0202019.

[30] E. N. Gilbert. Random Plane Networks. *J. Soc. Ind. Appl. Math.*, 9(6): 533–543, 1961. ISSN 0368-4245. doi: 10.1137/0109045.

[31] Maziar Nekovee. Worm epidemics in wireless ad hoc networks. *New J. Phys.*, 9(6):189–189, 2007. ISSN 13672630. doi: 10.1088/1367-2630/9/6/189.

[32] Renaud Lambiotte. Multi-scale Modularity in Complex Networks. *Proc. Natl. Acad. Sci. U. S. A.*, 108(19):7663–7668, apr 2010. ISSN 0027-8424. doi: 10.1073/pnas.1018962108.

[33] Sid Henriksen, Rich Pang, and Mark Wronkiewicz. A simple generative model of the mouse mesoscale connectome. *Elife*, 5(MARCH2016):1–19, 2016. ISSN 2050084X. doi: 10.7554/eLife.12366.

[34] D. Papo, J. M. Buldu, S. Boccaletti, and E. T. Bullmore. Complex network theory and the brain. *Philos. Trans. R. Soc. B Biol. Sci.*, 369(1653):20130520–20130520, 2014. ISSN 0962-8436. doi: 10.1098/rstb.2013.0520.

[35] Jesper Dall and Michael Christensen. Random geometric graphs. *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.*, 66(1), 2002. ISSN 15393755. doi: 10.1103/PhysRevE.66.016121.

[36] Justin Coon, Carl P. Dettmann, and Orestis Georgiou. Full Connectivity: Corners, Edges and Faces. *J. Stat. Phys.*, 147(4):758–778, 2012. ISSN 00224715. doi: 10.1007/s10955-012-0493-y.

[37] Songrit Maneewongvatana and David M Mount. On the Efficiency of Nearest Neighbor Searching with Data Clustered in Lower Dimensions. *Comput. Sci. - ICCS 2001*, pages 842–851, 2001. ISSN 16113349. doi: 10.1007/3-540-45545-0_96.

[38] Matthew Garrod and Nick S. Jones. Large fluctuations in the algebraic connectivity of Random Geometric Graphs (Working Paper). 2017.

[39] Till Hoffmann. C++ Containers In Cython (Blog Post), 2017. URL http://tillahoffmann.github.io/2016/04/18/Cpp-containers-in-cython.html.

[40] Xiao Xiao, E White, M Hooten, and S Durham. On the use of log-transformation vs. nonlinear regression for analyzing biological power-laws. *Ecology*, 92(10):110606103516051, 2011. ISSN 00129658. doi: 10.1890/11-0538.1.

[41] P.T. Boggs and J.E. Rogers. Orthogonal distance regression. *Contemp. Math.*, 112:183–194, 1990.

[42] Bradley Efron and Robert Tibshirani. *An Introduction to the Bootstrap*. CRC Press LLC, 1998. ISBN ISBN 0-412-04231-2.

[43] Giulia Menichetti, Luca Dall'Asta, and Ginestra Bianconi. Network controllability is determined by the density of low in-degree and out-degree nodes. *Phys. Rev. Lett.*, 113(7):1–5, 2014. ISSN 10797114. doi: 10.1103/PhysRevLett.113.078701.

[44] Derek Ruths. ZEN: Zero Effort Networks Library, 2012. URL http://zen.networkdynamics.org/.

[45] Alexander J. Gates and Luis M. Rocha. Control of complex networks requires both structure and dynamics. *Sci. Rep.*, 6(1):24456, 2016. ISSN 2045-2322. doi: 10.1038/srep24456.

[46] Jianxi Gao, Yang-Yu Liu, Raissa M. D'Souza, and Albert-László Barabási. Target control of complex networks. *Nat. Commun.*, 5:5415, 2014. ISSN 2041-1723. doi: 10.1038/ncomms6415.

[47] Gang Yan, Georgios Tsekenis, Baruch Barzel, Jean-Jacques Slotine, Yang-Yu Liu, and Albert-László Barabási. Spectrum of controlling and observing complexnetworks. *Nat. Phys.*, 11(9):779–786, 2015. ISSN 1745-2473. doi: 10.1038/nphys3422.

[48] Noah J. Cowan, Erick J. Chastain, Daril A. Vilhena, James S. Freudenberg, and Carl T. Bergstrom. Nodal dynamics, not degree distributions, determine the structural controllability of complex networks. *PLoS One*, 7(6), 2012. ISSN 19326203. doi: 10.1371/journal.pone.0038398.

[49] Isaac Klickstein, Afroza Shirin, and Francesco Sorrentino. Energy scaling of targeted optimal control of complex networks. *Nat. Commun.*, 8:15145, 2017. ISSN 2041-1723. doi: 10.1038/ncomms15145.

[50] Carl P. Dettmann and Orestis Georgiou. Random geometric graphs with general connection functions. *Phys. Rev. E*, 93(3):1–16, 2016. ISSN 24700053. doi: 10.1103/PhysRevE.93.032313.

[51] Srikanth K. Iyer and Debleena Thacker. Nonuniform random geometric graphs with location-dependent radii. *Ann. Appl. Probab.*, 22(5):2048–2066, 2012. ISSN 10505164. doi: 10.1214/11-AAP823.

[52] Seoung Kyou Lee, Sándor P Fekete, and James McLurkin. Structured triangulation in multi-robot systems: Coverage, patrolling, Voronoi partitions, and geodesic centers. *Int. J. Rob. Res.*, 35(10):1234–1260, 2016. ISSN 0278-3649. doi: 10.1177/0278364915624974.

[53] Alexander L Fradkov. Horizons of cybernetical physics. *Phil. Trans. R. Soc. A*, 375, 2017. ISSN 1860-0832. doi: 10.1098/rsta.2016.0223.