

# Unsupervised Spatial, Temporal and Relational Models for Social Processes

George B. Davis

February 2012

CMU-ISR-11-117

School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA 15213

## Thesis Committee:

Kathleen M. Carley (CMU, ISR), Chair

Christos Faloutsos (CMU, CSD)

Javier F. Peña (CMU, Tepper)

Carter T. Butts (UCI, Sociology / MBS)

*Submitted in partial fulfillment of the requirements  
for the degree of Doctor of Philosophy*

This work was supported in part by the National Science Foundation under the IGERT program (DGE- 9972762) for training and research in CASOS, the Office of Naval Research under Dynamic Network Analysis program (N00014-02-1-0973, ONR N00014-06-1-0921, ONR N00014-06-1-0104) and ONR MURI N00014-08-1-1186, the Army Research Laboratory under ARL W911NF-08-R-0013, the Army Research Institute under ARI W91WAW-07-C-0063, and ARO-ERDC W911NF-07-1-0317. Additional support was provided by CASOS - the Center for Computational Analysis of Social and Organizational Systems at Carnegie Mellon University. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied of the National Science Foundation, the Office of Naval Research, or the U.S. government.

**Keywords:** Clustering, unsupervised learning, factor graphs, kernel density estimation

## Abstract

This thesis addresses two challenges in extracting patterns from social data generated by modern sensor systems and electronic mechanisms. First, that such data often combine spatial, temporal, and relational evidence, requiring models that properly utilize the regularities of each domain. Second, that data from open-ended systems often contain a mixture between entities and relationships that are known *a priori*, others that are explicitly detected, and still others that are latent but significant in interpreting the data. Identifying the final category requires unsupervised inference techniques that can detect certain structures without explicit examples.

I present new algorithms designed to address both issues within three frameworks: relational clustering, probabilistic graphical models, and kernel-conditional density estimation. These algorithms are applied to several datasets, including geospatial traces of international shipping traffic and a dynamic network of publicly declared supply relations between US companies. The inference tasks considered include community detection, path prediction, and link prediction. In each case, I present theoretical and empirical results regarding accuracy and complexity, and compare efficacy to previous techniques.



*For Mary, Lea, George and Bruce, who always looked forward.*



# Acknowledgments

I am deeply indebted to a small host whose collaboration and faith saw this work to completion. First thanks go to my advisor, Kathleen Carley, whose support and advice across every possible endeavor has broadened my world. It has been a pleasure studying under someone who refuses to limit their scope of interest, and encourages the same. My gratitude for the breadth of opportunity I've experienced at Carnegie Mellon extends to the rest of the Computations, Organizations and Society faculty, to the exceptional Computer Science Department faculty including Carlos Guestrin and Tuomas Sandholm, and to Javier, Carter and Christos, who continued to offer support and advice months outside and topics away from our original plan.

I also wish to thank Dannie Durand, Christian Burks, and Darren Platt, whose mentoring in Computational Biology formed the foundation of my academic interest in applied Computer Science.

I've benefited greatly from interactions with extraordinary collaborators. To Michael Benisch, for his contagious high standards, and to Jamie Olson, for his tenacity, I owe special thanks. I'm also very appreciative of the conversations I shared with Andrew Gilpin, Jesse St. Charles, Jana Deisner, Peter Landwehr, Brian Hirschman, Brad Malin, Edo Airoidi, and Ralph Gross, and many other exceptional students in the COS and CSD programs.

Many people outside the Computer Science community helped me situate this work in the context of real problems. Robert Martinez and Kevin O'Brien at Revere Data, LLC were invaluable in both the data they donated and their experience in interpreting declared business-to-business relations, as were Arif Inayatullah and Gerald Leitner for their perspectives on financial networks. Rebecca Goolsby at the Office of Naval Research was supportive in far more than funding, helping me understand how and where to look for data, and how government agencies process research. I'm also grateful to our collaborators at Singapore's DSTA, whose data and domain knowledge in shipping

traffic was crucial to our joint projects.

To yet unnamed friends, especially Drew, Leith, Sam, Cesar and both Bens, I say thank you for your help balancing my life during the first few years. To my partners at Hg Analytics, I say thank you for your moral support in persevering with academic work as entrepreneurial life grew inevitably more unbalanced.

My most stalwart supporters have always been my family, and without love and encouragement from my parents Ann and Bart, and my sister Stephanie, this would not even have begun. I also wish to thank the Ricous, Fonsecas, and Schattens in my life for their kind encouragement. However, my deepest gratitude is reserved for Joana, my true love, friend, and partner in this endeavor and all that follows.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Spatial, Temporal and Relational (STR) Data . . . . .	3
1.2	Unsupervised Learning . . . . .	4
<b>2</b>	<b>Background and Preliminaries</b>	<b>7</b>
2.1	STR Modeling . . . . .	7
2.1.1	Temporal Data . . . . .	7
2.1.2	Spatial Data . . . . .	9
2.1.3	Relational Data . . . . .	11
2.1.4	Frameworks for STR analysis . . . . .	12
2.2	Clustering . . . . .	14
2.2.1	Defining Similarity . . . . .	14
2.2.2	Clustering Algorithms . . . . .	15
2.2.3	How Many Groups? . . . . .	17
2.3	Probabilistic Graphical Models . . . . .	18
2.3.1	Factor graphs. . . . .	20
2.3.2	Parameter sharing. . . . .	21
2.3.3	Discriminative models. . . . .	22
2.3.4	Representation. . . . .	23
2.3.5	Inference. . . . .	25

2.3.6	The Bayesian view . . . . .	26
2.4	Kernel Conditional Density Estimation . . . . .	27
2.4.1	Regression . . . . .	27
2.4.2	Density Estimation . . . . .	29
2.4.3	Conditional and Categorical Density Estimation . . . . .	30
<b>3</b>	<b>Data</b>	<b>33</b>
3.1	AIS Data (AIS) . . . . .	33
3.2	Company Relations Data (B2B) . . . . .	34
3.3	Supplementary datasets . . . . .	41
3.3.1	Sampson's Monastery (SAMPSON) . . . . .	41
3.3.2	Davis' Southern Women (DAVIS) . . . . .	42
<b>4</b>	<b>Community Detection</b>	<b>45</b>
4.1	Problem Definition and Background . . . . .	45
4.1.1	Defining 'Group' . . . . .	46
4.1.2	Variations and Detection of Cohesive Groups . . . . .	47
4.1.3	Bipartite (Link) Data . . . . .	48
4.2	Fuzzy, Overlapping Grouping . . . . .	49
4.2.1	Link data from network data . . . . .	50
4.2.2	Stochastic model of evidence generation . . . . .	53
4.2.3	The H-FOG algorithm . . . . .	55
4.2.4	The $k$ -FOG algorithm . . . . .	58
4.2.5	The $\alpha$ -FOG algorithm . . . . .	58
4.3	Example analysis: fuzzy groups in the monastery . . . . .	59
4.3.1	Example analysis: fuzzy groups among the southern women . . . . .	62
4.4	Parameter recovery experiments . . . . .	65
4.4.1	Generation of parameters and data . . . . .	65

4.4.2	Evaluation of fit . . . . .	66
4.4.3	Convergence profiles . . . . .	67
4.5	Improving $\alpha$ -FOG performance with residual priority updates . . . . .	69
4.6	Performance comparison summary . . . . .	71
4.7	(Infinite) Residual Expectation Maximization (REM) . . . . .	80
4.8	Discussion . . . . .	85
4.8.1	Validation . . . . .	85
4.8.2	Interstitial Roles . . . . .	86
4.8.3	Generating link data from networks . . . . .	87
4.8.4	Analyzing and visualizing fuzzy relationships . . . . .	88
4.8.5	Efficient EM algorithms for an unknown number of clusters . . . . .	89
4.8.6	Final thoughts and remaining questions . . . . .	89
<b>5</b>	<b>Path Prediction</b> . . . . .	<b>91</b>
5.1	Problem Definition . . . . .	91
5.2	Factor Graphs for Path Prediction . . . . .	93
5.3	Unsupervised Learning for Factor Graphs . . . . .	95
5.4	Plan Projection Experiment . . . . .	99
5.4.1	Method . . . . .	99
5.4.2	Results and Analysis . . . . .	100
5.5	Residual Expectation Maximizing Belief Propagation (REM-BP) . . . . .	102
5.5.1	Multi-clustering factors and factor graphs . . . . .	102
5.5.2	Inference . . . . .	104
5.5.3	Empirical performance of REM-BP . . . . .	108
5.6	Discussion and Future Work . . . . .	110

<b>6</b>	<b>Spatially embedded link prediction</b>	<b>115</b>
6.1	Problem Definition . . . . .	115
6.2	Classifying spatial embedding models . . . . .	117
6.3	A kernel-based spatial correlation model . . . . .	123
6.3.1	Inference and training for spatial correlation . . . . .	124
6.3.2	Fast Approximation . . . . .	126
6.3.3	Distances . . . . .	127
6.4	Experimental Results . . . . .	128
6.4.1	Scoring link prediction . . . . .	128
6.4.2	Experimental Setup . . . . .	130
6.4.3	Impact of Approximation . . . . .	130
6.4.4	Robustness to Missing Data . . . . .	131
6.4.5	Comparative Performance . . . . .	134
6.5	Discussion . . . . .	135
<b>7</b>	<b>Conclusion: contributions, limitations and themes</b>	<b>139</b>
7.1	The human process that studies human processes . . . . .	139
7.2	Relaxed grouping in networks . . . . .	140
7.3	Plans and places as hidden entities . . . . .	142
7.4	Pairwise relations in abstract spaces . . . . .	143
7.5	Bringing the methods online . . . . .	145
7.6	Advice for practitioners . . . . .	145
7.7	Analysts and scientists . . . . .	146

# Chapter 1

## Introduction

Sensor systems and electronic mechanisms are changing the ways in which we learn about human behavior. Foundational work in the social and cognitive sciences analyzed small populations via controlled experiments, or large populations via coarse surveys. Today, we can passively record detailed behaviors of thousands of actors spread across the globe, along with spatial, temporal, and relational data about the contexts in which they interact.

The potential for this data impacts almost every field of human endeavor. Scientifically, mass data on human interactions have produced new insights into how we exchange information, transmit diseases, and find partners for everything from business to reproduction. In commercial applications, social data can be mined passively to improve decisions such as product design, or integrated directly into products such as social search, directed advertisement and recommendation systems, geographic contextualization, and socially aware communication platforms. Governments and other public policy institutions throughout the world have placed a high priority on analyzing social data to better target social programs, improve mechanisms that share public resources, detect criminal behavior such as tax-evasion, or inform national security efforts.

However, accessing this potential requires solving new challenges stemming from scale, structure, and origin. In this thesis I consider datasets defined by three qualities which complicate analysis.

1. *Size*. The datasets are large and complex enough that detailed human analysis and trivial algorithms are impractical.
2. *Multi-modality*. The datasets include spatial, temporal and relational information that must be integrated in analysis.

3. *Latent entities and relations.* Entities not described explicitly in the data may improve analysis.

The datasets I examine contain hundreds of thousands of records related to tens of thousands of observed entities, where recent studies have analyzed specific properties of social structure in systems with over a billion entities [57] [8]. However, size and scalability are nonetheless a significant concern. Studies at the largest scale make use of massively parallel architectures designed to answer specific questions, whereas the algorithms in this thesis are concerned with enabling exploratory analysis at its early stages. A primary objective of this thesis is to present algorithms which can extract as much insight as possible from medium-sized datasets using a single commodity workstation.

Multi-modality is a feature of the data because our activities are influenced and constrained by spatial, temporal, and relational factors. Information in all three domains may be relevant to patterns of interest. Specialized techniques abound for modeling the regularities found in each domain. For example, fast algorithms exist for detecting clusters or density estimates in geospatial data, cyclical behavior and autocorrelation in time series data, and communities or central nodes in relational data. However, when applied independently these models cannot detect patterns that span domains. In section 1.1, I introduce the goals and themes which guide the work on integrated models in the remainder of the thesis.

The presence of latent entities and relations arises from uncontrolled data collection. Outside of controlled experiments (and often within), human systems lack clear boundaries which isolate the effects being studied. This is compounded by the fact that data-driven studies often make use of evidence collected in a broad sweep or with another purpose in mind. In such data, entities and relationships that are known *a priori* or detected directly likely interact in significant ways with others that are visible only through their effects. In section 1.2, I discuss the objectives I attempt to meet regarding the identification of such latent entities.

This thesis introduces several new algorithms intended to address both challenges in an integrated way, and in doing so extends three prior threads of research in multidomain and unsupervised learning. *Clustering* is the unsupervised search for hidden labels or partitions that explain observed attributes of data. In section 2.2, I review methods based on hierarchical clustering, expectation maximization, and more recent multi-membership models as applied to spatial and relational data. For the integration of parametric models capturing regularities in various data domains, I rely primarily on probabilistic graphical models, which I provide background on in section 2.3. Finally, to model relations where a parametric form is not known in advance, I build on the nonparametric

kernel conditional density estimation techniques introduced in section 2.4.

Development of the algorithms produced in this thesis was motivated primarily by the analysis of two new and complex datasets. The first tracks cargo vessel movements from port to port over a period of several days, in conjunction with attributed data about the vessels and their ownership relations. The second is a dynamic network between US publicly traded companies, tracking business relationships disclosed under regulatory requirements. Where appropriate, I also compare to canonical “standard” datasets from prior literature. All datasets used are introduced in detail in chapter 3.

I organize exploration of the datasets and research threads listed above in relation to three specific analysis tasks. In chapter 4, I adapt clustering methods to detect fuzzy, overlapping communities within social network and event attendance data. In chapter 5, I introduce an expectation-maximization based message passing algorithm for probabilistic graphical models to detect probable destinations in geospatial path data. In chapter 6, I present a dyadic variant of kernel-conditional density estimation for predicting missing or future links in attributed network data.

Finally, in chapter 7, I review results and contextualize them with respect to ongoing research. By way of summary, I provide a brief manual for a practitioner looking to select among the methods introduced or otherwise touched upon in this thesis.

## 1.1 Spatial, Temporal and Relational (STR) Data

In a popular folk tale, a group of blind men encounter an elephant. Grappling with different parts of the beast, they name it variously a rope (for its trunk), a fan (for its ears), a column (for its legs), and a spear (for its tusks). Students of human interaction, or any complex system, should relate to this story. When analyzing systems too large or complex to model accurately in a controlled environment, we are often limited to data collection schemes that observe a single aspect of the system, such as a set of physical measurements, or records of a single kind of transaction or communication. In addition to being easier to collect, these datasets more plausibly admit useful assumptions (for example that they are independently and identically distributed) which would be difficult to stomach if connecting data were collected. As a result, domain-spanning interactions remain, like the elephant, undetected. The algorithms introduced in this thesis are intended to detect cross-domain patterns in datasets where spatial, temporal and relational (STR) data can be connected.

A spatial, temporal or relational setting is defined by its *regularity*, such as the ordering of a

timeline, the distance metric of a space, or the recurrence of relations in a network. Learning STR patterns in human systems means establishing a correspondence between this inherent regularity and statistical regularity found in the data. For example, we might expect that individuals living near each other speak the same language, or that ships belonging to the same company carry similar cargo. In section 2.1, I briefly overview the types of regularity associated with each data domain, and how they relate the mutual information structures in various models.

Our expectation of these correspondences is supported by our experience that constraints in one aspect of our location, schedule, affiliations or attributes influence our decisions in another. The correlation between space and language listed above might be driven by many choices of individuals desiring to communicate with those nearby to adopt a common language. In different circumstances, the same constraint might produce a similar pattern via an inverted process: an immigrant, needing to communicate and constrained by knowing only a foreign tongue, might choose to locate near other immigrants. These processes may be *homophilic*, as above, or *assortative*, such as the need of a corporation to do business with companies that occupy different roles in the supply chain.

Prior beliefs regarding the processes described above are a necessary part of domain modeling. However, when analyzing STR data from human systems it is very uncommon to enter with confidence regarding the exact structure of patterns produced, or the ways in which they interact. The selection of mixed-membership, graphical, and nonparametric models for this thesis is intended to relax the level of prior confidence necessary, by allowing a degree of model selection to be performed during the data fitting process. I refer to this process as *localization*, because the outcome is a mapping between the spatial, temporal and relational setting of the data and the structure of mutual information within the data. The primary goal of this thesis, as relates multi-domain analysis, is to shift part of the burden of multi-domain modeling away from an analyst's prior beliefs and toward a data-driven model selection process.

## 1.2 Unsupervised Learning

Any data analysis task involves consideration of boundary effects. Have we collected enough data to positively identify patterns? Has our collection process biased us toward detection of some patterns over others?

Human systems data are unusually high-risk in this regard. Almost nothing about individual behavior is actually individual: we are influenced by those we interact with, those we observe passively in our physical environment, and memories we carry over from past interactions. Our response to these interactions tends to sort us, so that it is almost impossible to observe groups



of individuals that don't have important factors linking their behavior [82]. Efforts to enumerate these linking factors are easily confounded by our tendencies to form and dissolve affiliations, change environments, and to forget or exchange memories [101].

The collection of data from sensor systems and electronic mechanisms further complicates these challenges. Technology and expense limit the types of information we can extract from sensor systems, and the need to attract participation limits the types of information we can extract from electronic mechanisms. Since data collection apparatus are difficult to deploy and take time to collect a critical mass, systems are increasingly designed with a "shotgun" approach aimed at adequately supporting many future analyses but specifically supporting none. Datasets such as the ones analyzed in this thesis often enjoy a "second life", in which they are analyzed for purposes having little to do with those for their original collection.

The net effect of these challenges is that it is rare to begin analysis on human STR data with confidence that a dataset explicitly records all of the entities that are relevant to understanding the patterns within. Inference of these entities is a desirable intermediate step to detecting patterns. In machine learning, this task is referred to as *unsupervised learning*, because the analyst cannot provide any ground truth regarding entities she herself does not know to exist.

In the previous section I discussed the process of *localization*, where the mutual information structure of a model is adapted to data. The introduction of latent entities is my primary means of accomplishing that (bandwidth estimation in nonparametric regression is a second one). At one extreme, when no latent entities are proposed, the initial model is entirely responsible for the mutual information structure of the data. At the other, proposal of infinite latent entities serves as a kind of *deus ex machina* in which all variation in a dataset can be explained by latent entities, analogous to overfitting in a parametric model. The main challenge in applying unsupervised techniques is achieving a balance on this spectrum by negotiating the tradeoff between the number of latent entities proposed and the accuracy of fit to the data. The guiding principle applied is Occam's razor, formally instantiated through criteria such as minimum description length (MDL) [15] [47] or a Bayesian priors regarding the population of latent entities [13].

Unsupervised learning is most often discussed in the context of clustering, which I review in section 2.2. The two unsupervised algorithms introduced in this thesis each relax a commonly found constraint in other clustering algorithms. In chapter 4, I consider multi-membership clusters within a relational setting, where clustering typically involves strict partitioning. In chapter 5, I perform strict clustering in a way that bridges spatial, temporal and relational domains. The goal, as usual, is to enable an analyst to precisely express the types of latent entities she suspects, and how they might interact with the rest of the data, while simultaneously providing the tools to achieve the

balance described above.

# Chapter 2

## Background and Preliminaries

### 2.1 STR Modeling

#### 2.1.1 Temporal Data

The weakest temporal setting is a *partial ordering*, in which an order is defined between some observations while others are ambiguous. Partial orderings often arise when the measurement or representational accuracy of time is limited, when complete orderings on two datasets are integrated (such as a merge of systems logs), or when orderings are recovered opportunistically from other data, such as references between papers.

*Full orderings* allow a complete indexing of events, permitting definition of simple regularities such as Markov assumptions [89]. A model based strictly on ordering makes minimal use of the continuity of time, relying instead on a well defined state space to define locality for each data point. This reliance might make it unusually sensitive to missing data or false ordering, and makes it impossible to detect effects that are mediated by the length of a time interval between events. Fully ordered time measurements are annotated in this thesis as variants of  $t \in \mathbb{N}$ .

The inclusion of *inter-arrival times*, which I annotate with variants of  $\delta$ , allow time-dependent relaxations of Markov assumptions, such as Gaussian processes or time decay models [5]. When all inter-arrival times are known, time measurements are placed on an *absolute timeline*, on which we annotate points with variations of  $t \in \mathbb{R}$ .

Distributions of temporal measurements can be studied independently (*e.g.* with point process models [50]), but this thesis considers only *temporal relations* which associate a time with additional labels or measurements. The simplest type of temporal relation is an *event*, meaning a pinpoint

observation or measurement, such as the timing of a transmission or a temperature reading. I annotate an event as a tuple  $e = \langle t, \mathbf{x} \rangle$ , where  $t$  is the temporal part of the relation and  $\mathbf{x}$  contains everything else.

Models of mutual information between events most commonly presume that events closer together are more closely related. One way of formalizing temporal closeness' impact on related data is by introducing a decay function.

**Definition 1** A **decay function**  $c : \mathbb{R}^+ \rightarrow [0, 1]$  is an indicator of mutual information such that given events  $\langle t, \mathbf{x} \rangle$  and  $\langle t', \mathbf{x}' \rangle$ ,  $I(\mathbf{x}, \mathbf{x}') = f(c(|t' - t|))$ . Furthermore,  $c$  must be monotonically decreasing in its parameter.

The function  $f$  which relates  $c$  to  $I$  is model specific. For example, in a Gaussian process  $c$  indicates the correlation between real-valued associated measurements  $x_1$  and  $x_2$ . In kernel regression,  $c$  is the coefficient on a weighted average, whose denominator is determined by the distribution of other observed points. Some classes of common decay functions include *window decay*, *polynomial decay* and *exponential decay*.

**Definition 2** The **window decay** with bandwidth  $\beta \in \mathbb{R}^+$  relates equally all points within a certain distance of each other.

$$c_{w=\beta}(\delta) = \begin{cases} 1 & : |\delta| \leq \beta \\ 0 & : |\delta| > \beta \end{cases}$$

**Definition 3** The **polynomial decay** with degree  $d \in \mathbb{N}$  relates points inversely proportional to the square of temporal distance.

$$(2.1) \quad c_{p=d}(\delta) = \delta^{-d}$$

**Definition 4** The **exponential decay** with scale  $\lambda \in \mathbb{R}^+$  states that mutual information between points decays with a half-life inversely proportional to  $\lambda$

$$(2.2) \quad c_{e=\lambda}(\delta) = e^{-\lambda\delta}$$

In cases where time has a seasonal effect, or where measured time is an imperfect proxy for some other ordered process, it can be useful to adopt a *temporal convolution function* in conjunction with some decay function to describe mutual information.

**Definition 5** A temporal convolution function  $v : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  in combination with decay function  $c$  indicates mutual information of  $\langle t, \mathbf{x} \rangle$  and  $\langle t', \mathbf{x}' \rangle$ , according to  $I(\mathbf{x}, \mathbf{x}') = f(c(v(t) - v(t')))$ .

Use of a temporal convolution function is analogous to adopting a non-standard distance metric, as described below.

Another way of using temporal structure is to use it to derive other relations. Event information may be collected in *interval summaries* describing, for example, the of events or average value of a measurement in a given time interval. Electronic mechanisms in which transactions update a persistent state often collect *state transition* data, such as the logs of writes to a database. *Hidden* or *partial* state models such as [95] relate observed events, interval summaries, or state transitions to a transition model which includes some latent, unobserved state.

### 2.1.2 Spatial Data

Many concepts regarding continuous time can be generalized for continuous spatial data. Events correspond to named spatial points, intervals to regions, and a state space with transition points to a full partitioning. Ordering and inter-arrival time must be replaced by more complex definitions of *adjacency* and *distance*, which are the subjects of most descriptions of spatial regularity.

A *distance function*  $d : \mathbb{R}^n, \mathbb{R}^{\times} \rightarrow \mathbb{R}$  measures path length between any two points in an  $n$ -dimensional space. A partial ordering on any set of points can be produced based on each of their distances to a common reference point. Many applications of distance functions rely on the additional constraints satisfied by a distance *metric*.

**Definition 6** A **distance metric** is function  $d : \mathbb{R}^n, \mathbb{R}^{\times} \rightarrow \mathbb{R}$  satisfying constraints of non-negativity ( $d(\mathbf{x}, \mathbf{y}) \geq 0$ ), discernability ( $d(\mathbf{x}, \mathbf{y}) = 0 \rightarrow \mathbf{x} = \mathbf{y}$ ), symmetry ( $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ ), and subadditivity ( $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{x}, \mathbf{z})$ ).

Some of the distance metrics used in this thesis include the Cartesian, Manhattan, and great circle distances.

**Definition 7** The **Cartesian distance** between two vectors in  $\mathbb{R}^n$  is given by

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

**Definition 8** *The Manhattan distance between two vectors in  $\mathbb{R}^n$  is given by*

$$d_m(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n |x_i - y_i|$$

**Definition 9** *The great circle distance approximates the length of the shortest path along the surface of the Earth between a pair of latitude-longitude coordinates.*

$$(2.3) \quad d_g(\mathbf{x}, \mathbf{y}) = 6371 \text{ km} * 2 \arcsin \left( \sqrt{\sin^2\left(\frac{x_{lat} - y_{lat}}{2}\right) + \cos x_{lng} * \cos y_{lng} * \sin^2\left(\frac{x_{lng} - y_{lng}}{2}\right)} \right)$$

Distance may also be defined in abstract spaces, such as coordinate spaces based on two or more measured attributes of an entity. A frequent approach in these spaces is to use a traditional metric (such as Cartesian) on normalized versions of these variables. The proper method of normalization is a domain and model-specific decision.

A set of points under a given distance metric possess a *convex hull*, the space enclosed by all minimum paths and surfaces between them. Two hulls or other regions which share a surface point are called *adjacent*, allowing a binary relationship to be extracted from spatial data. If a distance metric is present, adjacency among a set of points rather than regions can be defined using *Delauney triangulation*. Two points are adjacent under this definition if their *Voronoi cells* – the region, for each point, which is closer to it than any other point in its set – abutt. This definition has been used in models of shared information between irregularly distributed sensors [62].

Distance is often invoked with the expectation of similarity between nearby points. This can occur in data associated with processes where location indicates a common origin, where similar entities directly attract each other, or where some sorting influence pushes entities with certain characteristics to certain locations. One way to encode this concept is by pairing a distance function with a decay function as described in the previous sections. Alternatively, distance can be used to compute adjacency or common enclosure to achieve a binary relation associated with similarity. Concepts of distance, enclosure, and density are also important in spatial clustering (discussed further in later sections), another way in which relational data can be inferred from spatial configuration.

Spaces that are physically situated but exceptionally regular, such as a chess board, or especially convoluted distance functions, such as mass transit transportation time, can often be more efficiently represented in relational form.

### 2.1.3 Relational Data

In the sections above, I’ve described how associating data with a place or time implies relations such as “before”, “after”, “within”, and “adjacent” to data associated with other points or sets of points. In relational data, these kinds of associations are recorded explicitly. Rather than from temporal logic or the geometry of a distance function, regularity arises from a relational *schema* describing recurring patterns of association. These relations can be arbitrary in number and complexity, ranging from explicit representations of distance functions between points to complex relations involving dozens of entities.

Mathematically, relational algebra generalizes those applied to finite-dimensional spaces or time-lines, meaning that any kind of spatial or temporal regularity is in fact a special class of relational regularity. A useful property of these special cases is that they allow us to succinctly describe an infinite variety of entities and relations not observed in data. For example, the presence of a coordinate system implies the existence of an infinite space of points not explicitly listed in the data, and the presence of a distance function allows us to consider the space between them. I use *relational data* to refer to data in which entities and relations are explicitly enumerated, such as the entries in a relational database. The existence of additional entities or the values of unobserved relations might be inferred by a model (these are primary goals of the algorithms I present), but they are not a feature of the relational setting.

In this thesis I divide relations into three main categories. *Tabular relations* encode the association of an entity with its attributes - for example, the demographic characteristics of an agent. Each instance of a tabular relation describes a single entity. *Unimodal relations* encode relationships between like entities, such as pairwise friendships between individuals. The ability to string unimodal relationships together into paths, trees and other connected structures enables the search for certain kinds of regularity. *Bimodal relations* cover interactions between non-alike entities, such as the association of an individual with a group he or she belongs to. Multimodal relations of more than two entities are discussed in this thesis by treating each such association as a new entity.

Tabular relations are the most traditional subject of statistics. The most typical assumption regarding mutual information between instances of tabular relations is that it is mediated by some external parameters, conditioned on which the values observed for each entity are independent. The algorithms in this thesis presume that other relations also influence the structure of mutual information. I notate tabular relations by assigning a vector for each attribute whose index indicates the entity with which a value is associated. For example,  $a_i$  might indicate the age of the  $i$ ’th individual.

Unimodal relations have been studied in mathematics since the publication of Euler’s *Seven Bridges of Kronigsberg*. The relation considered in that publication was spatial adjacency, with the relational formalism adopted as a tool for considering possible paths. Graph theory has since been broadly applied to modeling of physical flows, optimal topologies, and formal logics.

Sociologists have long considered unimodal relationships in human systems under theory that social phenomena are neither emergent from the properties of individuals nor some abstract social entity, but from the patterns of relations among people [106]. This field is referred to as *social network analysis*. Early social network analysis focused primarily on graph theoretic definitions and measurements that could be used to summarize the position of an individual or group within a network [40], and relating these definitions to observed sociological effects. The recent availability of large scale, high quality data on relations between individuals as attracted interest from sociologists, physicists, and computer scientists in describing generative stochastic processes [107] [108] [69] which explain common patterns in network configurations observed “in the wild”.

Perhaps because their interpretation differs more from context to context, bimodal relationships have not attracted dedicated communities in the same way that unimodal relations have. A specific class which has been frequently studied are containment relations, such as those between an individual and group, or those between a document and a word [14].

I notate unimodal and bimodal relationships as matrices. For example, if  $\mathbf{R}$  is a bimodal relation between individuals and organizations,  $\mathbf{r}_{ij} \in \mathbf{0}, \mathbf{1}$  might indicate whether individual  $i$  is a member of organization  $j$ . When a relationship is binary, as above, compound relations can be achieved via linear algebra. For example,  $\mathbf{C} = \mathbf{R}\mathbf{R}^t$  would give the comembership relation between two individuals, such that  $c_i$  is the number of organizations in which they share membership. Unlike distance metric values, a higher value on a weighted relation conventionally indicates a greater association. When a relation has at most one value per column, we say that it is *many-to-one*, if its transpose has that property we say it is *one-to-many*, and if both are true then it is *one-to-one*. In this thesis, I do not consider relations with negative weightings, which complicate assumptions of transitivity built implicit in some algorithms.

#### 2.1.4 Frameworks for STR analysis

Unifying frameworks for the study of spatial, temporal and relational data have developed independently within several academic communities. Here I briefly profile and compare three communities on which the work in this thesis builds.



- **Dynamic network analysis** [22] (DNA) emerged from the social network analysis community as a way of extending that field’s unimodal analysis to incorporate multiple networks, entity types, change over time, and additional contextual information. A dynamic network dataset is termed a *metanetwork*, which may contain multiple *nodesets* representing different kind of entities. Tabular data about these entities is recorded as *attributes* associated with each *node*, and may contain spatial information such as latitude and longitude. Relations are termed *networks*, may be unimodal (also called *square* in reference to the matrix representation of the relation) or bimodal, and may be binary or *weighted* (conventionally a greater weight means a stronger connection rather than a greater distance). Temporal data is represented via a state transition model, where timestamped *deltas* describe changes observed in the network.

An aspect that distinguished DNA from the other disciplines described below are strong links to sociological literature, and in particular the social network analysis and agent-based modeling communities. To better standardize analysis between datasets, dynamic network analysis aimed at the sociological community often employs a standardized schema including entity categories such as “agent”, “organization”, “task”, “knowledge”, “place” [22]. Following from its origins, a significant topic in dynamic network analysis is the computation of node and network level measurements, and the extension of previous social network analysis measurements into the multimodal dynamic network setting. Work aimed at the agent-based modeling community includes simulations that use dynamic networks as input [23] and simulations that produce dynamic network data as output [105]. Other topics of study include the interplay between spatial clustering and network clustering [98] and the analysis of potential paths through a network based on those observed.

- **Statistical relational learning** [44] (SRL) emerged from the probabilistic graphical modeling of the statistical machine learning community, as a way of tackling common inference problems related to information in databases. A distinctive motivation within this community is the proper handling of a variety of types of erroneous data, such as duplicated entities [10] and missing relationships. SRL typically involves explicit modeling of mutual information in a dataset in the form of relational Markov networks [112] or Markov logic networks [102].
- **Data mining** has its origins in database management, and carries over that field’s focus on scale, as well as an emphasis on concepts defined algorithmically (as opposed to SRL’s probabilistic framework or DNA’s links to other fields of behavioral study). Major lines of research include analysis of self-similarity in networks [9], as well as the factors which may produce or confound it [69]. An ongoing challenge task within this field is the development

of generative models which replicate the properties observed in real world networks [67].

## 2.2 Clustering

Clustering is the inference of a hidden relation between an observed class of entities and a second, unobserved class, often a group entity indicating common origin or association. Occam’s razor argues against presuming entities beyond those necessary to explain experience; the necessity that motivates clustering is generally an unusual similarity between some subsets of observed entities relative to the general population. Since no instances of the cluster relation can be observed, its inference is an example of *unsupervised learning*.

Given  $n$  entities and  $k$  clusters, we can describe the clustering relation as an  $n \times k$  matrix,  $\mathbf{X}$ , where  $x_{ij}$  indicates the *member of* relation between observed entity  $i$  and latent cluster  $j$ . The number of clusters  $k$  may be known in advance or inferred during clustering, as we discuss later in this chapter. In general  $\mathbf{X}$  is nonnegative, but it may be either *binary* or *weighted* or represent of varying degrees of membership. If there is at most one nonzero entry per row,  $\mathbf{X}$  is a *strict partitioning* of the observed entities, and is sometimes represented as a vector  $\mathbf{x}$  where  $x_i$  indicates the cluster assignment (the index of the nonzero column). Otherwise we may say it is a *soft clustering* admitting multiple (if rows can have multiple entries) or overlapping (if columns can) memberships. In general, we seek clusters whose members demonstrate some kind of similarity in observable parameters. If there are  $m$  such parameters, we record them in an  $n \times m$  matrix  $\mathbf{Y}$ , where the row vector  $\mathbf{y}_i$  gives the observed parameters of the  $i$ ’th entity.

### 2.2.1 Defining Similarity

The similarity sought among members of the same cluster may be defined in various ways to identify different kinds of latent groups. Clustering based on relational data often seeks *cohesive groups* with an increased concentration of internal direct links or short paths. An alternative framework, *structural similarity*, groups nodes based on their relations to other points, such as set of network nodes with correlated relations [17]. I generally focus on the former pattern, but the assortative pattern is discussed further in the sociological context of structural roles in chapter 4.

In spatial or temporal clustering, a distance metric is typically employed to identify sets of nearby points in an otherwise sparsely populated space. Distance might be measured directly between co-clustered points, used to define a convex hull for density maximization [63], or applied between individual points and some summary point [80]. The concept of a summary point or

distribution may be generalized to allow clusters with adaptive shapes or distributions [43]. This “cluster-as-emitting-distribution” approach can be applied to detect many kinds of spatial, temporal and relational patterns, and is the primarily approach discussed in this thesis. I formalize this for strict partitionings in the following definition, which introduces a  $k \times d$  matrix  $\Theta$  containing  $d$  latent parameters for each of the  $k$  clusters.

**Definition 10** *A latent cluster likelihood  $L(\mathbf{Y} \mid \mathbf{x}, \Theta)$  relates the probability of observing entity attributes  $\mathbf{Y}$  given latent membership assignments  $\mathbf{x}$  and the unobserved cluster parameters  $\Theta$ . The likelihood function must have the properties that  $\mathbf{y}_i \perp \mathbf{x}, \Theta \mid \theta_{\mathbf{x}_i}$ . In other words, the likelihood of instance properties depends only on the parameters of the associated cluster.*

**Proposition 1** *Any latent cluster distribution can be rewritten in a factorized form*

$$L(\mathbf{Y} \mid \mathbf{x}, \Theta) = \prod_{i=1}^n l(\mathbf{y}_i \mid x_i, \theta_{\mathbf{x}_i})$$

### 2.2.2 Clustering Algorithms

The most constrained cluster analysis, classification of  $n$  entities into  $k$  discrete groups, admits  $n^k$  possible strict clustering relations, far too many to explore exhaustively in nontrivial datasets. Score based cluster objectives exhibiting near submodularity or supermodularity can be tractably approximately optimized [81] using variants of hierarchical clustering *e.g.* [76] or by recursive partitioning *e.g.* [78].

When a latent cluster distribution is introduced, clustering can be approached as the estimation of hidden cluster parameters and optimized using the expectation maximization (EM) algorithm. Dempster *et al.* define EM [32] as a very general method for determining maximum likelihood parameter estimates when some data are missing. Rather than performing often intractable integrations over the possible assignments to hidden variables, EM alternates between computing their expected values and computing likelihood maximizing parameters given those values.

Clustering is one of the exemplary problems analyzed in [32], where it is introduced as determining a finite mixture model in which unobserved categorical variables indicate the source component distribution for each observed point. The expectation step in this case amounts to determining the probability with which each point originates from each cluster given its current parameters, whereas the maximization step optimizes those parameters given the current expectations. The algorithm

terminates when it reaches a fixed point, detected by threshold of stability in the parameters. It is sketched out briefly below.

---

**Algorithm 1** Soft EM Clustering
 

---

**Require:**  $Y$  {observed entity properties}

**Ensure:**  $\Theta$  {optimized cluster parameters}

**Ensure:**  $\mathbf{X} \in \mathbb{R}^{n \times k}$  {cluster relation indicating probability of origin}

Initialize  $\mathbf{X}$  s.t.  $\forall_{i=1}^n \sum_{j=1}^k x_{i,j} = 1$

**repeat**

$\mathbf{X}' \leftarrow \mathbf{X}$

$\Theta \leftarrow \operatorname{argmax}_{\Theta} L(\mathbf{Y} \mid \mathbf{X}, \Theta)$

$\mathbf{X} \leftarrow \operatorname{argmax}_{\mathbf{X}} L(\mathbf{X} \mid \mathbf{Y}, \Theta)$

**until**  $\mathbf{X}' \approx \mathbf{X}$

---

Since each step increases the likelihood of observed data, each global maximum likelihood configuration is a fixed point of the algorithm [91]. However, there may be many other local maxima which halt the progress of the algorithm, making it sensitive to initial conditions. For applications where a local maximum is sufficiently informative, this may be irrelevant. A common property of latent cluster distributions is that hidden cluster attributes can be precisely tuned to give extremely high likelihood when responsible for only a small set of observed entities [79]. This propensity for overfitting can induce many poor-quality local minima for the EM algorithm, and must be handled by some combination of initialization procedures and cluster priors in a domain and model specific fashion.

Algorithm 1 is labeled “soft” because its output is a distribution over clusters for each entity. Converting this to a strict maximum *a posteriori* estimate of group assignments and parameters is nontrivial, as the joint probabilities of assignment may be different from the marginals estimate revealed by EM. An alternative “hard” EM clustering algorithm is given by replacing the expectation step with an assignment of each entity to its optimal cluster.

Because each step improves the overall likelihood of the data, algorithm 2 belongs to the *generalized expectation maximization* (GEM) algorithm class defined in [32], and because each iteration updates a subset of the hidden variables and cluster parameters it belongs to a subclass known as expected conditional maximization (ECM) [83]. ECM algorithms can achieve improved performance when calculating the maximum configuration of one or a few parameters is simpler than doing so for the entire set. Under a traditional EM algorithm, significant computation time is spent re-computing marginal distributions for variables that are relatively stationary between iter-

---

**Algorithm 2** Hard EM Clustering

---

**Require:**  $Y$  {observed entity properties}**Ensure:**  $\Theta$  {optimized cluster parameters}**Ensure:**  $\mathbf{x} \in \{1, 2 \dots k\}^n$  {cluster assignment vector} $\forall_{i=1}^n c_i \sim \text{Uniform}(\{1, 2 \dots k\})$  {random initialization}**repeat** $\mathbf{x}' \leftarrow \mathbf{x}$ **for**  $j = 1$  to  $k$  **do** $\theta_j \leftarrow \text{argmax}_{\theta} \prod_{i:x_i=j} l(\mathbf{y}_i | x_i, \theta_j)$ **for**  $i = 1$  to  $n$  **do** $x_i \leftarrow \text{argmax}_j l(\mathbf{y}_i | x_i = j, \Theta_j)$  {update entity assignments}**end for****end for****until**  $\mathbf{x}' = \mathbf{x}$ 

---

ations, but with the use of clever data structures ECM algorithms can concentrate computation on parameters whose maximal assignment is changing. Meng *et al.* show that ECM algorithms can achieve the same convergence guarantees as EM algorithms when updates are unconstrained, in the sense that all cluster parameters may be optimized freely within each iteration (i.e. between checks for convergence). However, hard-EM clustering as given above constrains parameter updates to those associated with a reassignment of an entity, weakening its convergence criteria and making it theoretically more vulnerable to “bad” local maxima.

In chapter 4 I compare performance on several EM variants, introduce an algorithm applying EM to an infinite mixture model, and consider the performance impact of an adaptive update schedule. This approach is generalized for use in graphical models in chapter 5.

### 2.2.3 How Many Groups?

In many clustering problems, measures of internal cluster cohesion decrease as additional entities are explained by a latent group. This creates a tension between the desire to summarize a dataset with the minimum number of groups and the desire for each group to be maximally representative of its members. In some cases the grouping objective is naturally maximized with a number of groups that can be determined algorithmically, as with Newman’s modularity score. In other cases, when group entities are intended as useful abstractions, group granularity is more a matter of preference or suitability to application than of correctness. For that reason, many clustering applications require

user input in determining group quantity or sizing. This might involve specifying an explicit number of groups (as in  $k$ -means clustering), selecting an “elbow point” in a cohesion scoring function, or via a selection process where a user dynamically navigates clusters at various granularities [98]. An alternative approach, discussed below and extended in this thesis, is to involve both data and prior preference in determining group granularity within a probabilistic framework.

In a latent cluster distribution, when a number of clusters  $k$  is known or presumed *a priori*,  $\mathbf{X}$  is an  $n \times k$  matrix, or in vector notation for a strict partitioning  $\mathbf{x} \in \{1 \dots k\}^n$ . If  $\Theta$  has a finite number of parameters per cluster, then the entire latent cluster distribution has a finite parametric form. In cases where an unknown number of hidden entities are present, the problem takes on nonparametric qualities or, more precisely, an infinite parametrization. The latter, due to Beal [11], involves infinite but sparsely populated  $\mathbf{X}$  and  $\Theta$ . Intuitively, specifying a prior distribution on the the degree of sparsity (i.e. the number of nonzero rows or columns) in this parameterization allows us to express prior beliefs or preferences regarding the number of groups.

One such prior is the Chinese restaurant process, or CRP. Briefly, the CRP considers a stream of entities being inserted into partitioned sets. Each entity is inserted into an existing set with a likelihood proportional to the set’s size, and begins a new set with likelihood proportional to an innovation parameter,  $\alpha$ .

$$(2.4) \quad P(c_i = k \mid x_{1 \dots k-1}; \alpha) = \begin{cases} \frac{\sum_{j=1}^{i-1} I(x_j=k)}{i-1+\alpha} & : 1 \leq k \leq \max(x_{1 \dots i-1}) \\ \frac{\alpha}{i-1+\alpha} & : k = 1 + \max(x_{1 \dots i-1}) \\ 0 & : else \end{cases}$$

The innovation parameter has a natural interpretation as the coefficient of logarithmic growth in the expected number of clusters as more data points are observed ( $E(\max(\mathbf{x})) = \alpha \log n$ ), and the tendency of individuals to prefer larger partitions mirrors preferential attachment effects observed in many datasets. CRPs have been adapted for use as priors in mixture models [13], as well as extended to allow for soft clustering, where it is sometimes referred to as the Indian buffet process [46]. Both are described in further detail in chapters 4 and 5 where I adapt them for use in community detection and for hidden variables with an unknown number of categories in probabilistic graphical models.

## 2.3 Probabilistic Graphical Models

Probabilistic graphical models (PGMs) have emerged as an effective framework for modeling patterns across spatial, temporal and relational domains, both individually and in conjunction. This

is due to their ability to concisely describe regular patterns of variable dependence induced by an STR environment. Examples include conditional random fields (CRFs) with topologies derived from the physical layout of sensors [66], dynamic Bayes nets [89] and CRFs [109] whose structure follows the flow of time, and relational Markov networks (RMNs) with structures templated on links in a relational database [112]. In this thesis, I build on existing work using PGMs to model these domains, and address a problem that reduces their applicability in some settings.

I will concentrate on discriminative PGMs, in which variables can be partitioned into those that will be observable at time of inference and those that will remain hidden. Discriminative PGMs are typically trained by optimizing model parameters against training data for which values of hidden variables have been provided (though some existing alternatives are discussed in section 2). For many real-world STR systems, however, proper training data can be impossible or cost prohibitive to obtain. This is particularly true in analyzing covert systems, where labels for hidden variables (for example, affiliation with a secret organization) cannot be accurately observed at any price. There also exist interesting tasks in which the objective is hidden by definition, such as identifying communities in a network. Rapidly evolving tasks, such as identification of trendy destinations in a busy city, pose another kind of challenge as labeled data may become obsolete soon after it is obtained. Finally, there are exploratory settings in which supervised training might be feasible in the final deployment of a learning system, but is cost prohibitive during development.

One goal of this thesis is to show that the modeling flexibility of PGMs can be practically applied to spatial, temporal and relational settings with important latent variables through the introduction of expectation maximizing belief propagation (EMBP) algorithms. Combining EM clustering with a message passing approach, these algorithms conduct unsupervised learning to produce a hypothesis that is consistent with the model structure and observed variables. In contrast with existing Bayesian approaches to estimating or sampling posterior distributions of individual variables or sets of interest, my algorithms are designed to identify only a single, consistent hypothesis regarding all variable values. While acknowledging limitations of this objective, I argue that it enables worthwhile performance improvements and is potentially more useful to analysts than a more thorough but difficult to interrogate result.

In chapter 5, I describe a class of undirected PGMs whose model parameters allow efficient updates and converge toward a fixed point when variable assignments and model parameters are optimized incrementally. I also define message passing algorithms that conduct these updates efficiently. It has been shown in other belief propagation schemes that the schedule by which messages are updated is an important factor in performance [38], and early experiments with EMBP have confirmed that an asynchronous schedule improves both performance and convergence [30]. I

will further explore this by testing an approximate-residual prioritized schedule following [110], as well as a novel online schedule which prioritizes updates on variables relevant to ongoing inference as new data arrives.

Consider an integer vector  $\mathbf{x} \in \mathbb{Z}^n$  consisting of random variables, each of which can take one of at most  $k$  discrete states. We will use the abbreviation  $\mathbb{Z}_k^n$  as a shorthand for the set of possible assignments to  $\mathbf{x}$ .  $P(\mathbf{x})$ , the joint PMF over all elements of  $\mathbf{x}$ , can be represented by a real vector  $\mathbf{w} \in \Delta^d$ , whose entries enumerate the probability of each combination of assignments to  $\mathbf{x}$ . This naive representation would require  $d = |\mathbb{Z}_k^n| = k^n$  parameters<sup>1</sup>, too many to store, learn from data, or conduct inference from unless  $n$  is trivially small. A key goal of any PGM is to provide a more scalable representation of  $P(\mathbf{x})$  by identifying a set of functions with smaller domains which can be reconstructed into the original PMF, or a useful approximation. In this section I describe several specific representational tricks on which my thesis builds. Since my primary goal is to introduce terms and notation, I defer to [61] for a more thorough discussion.

**Markov random fields.** Arguably the simplest graphical model is the Markov random field (MRF), in which each node is a variable and edges describe probabilistic dependencies. The structure of the field is described by a symmetric binary  $n \times n$  adjacency matrix  $\mathbf{N}$  indicating probabilistic dependence between variables. I will sometimes refer to  $\mathbf{N}$  through an equivalent function  $N$  that maps a variable to a set of adjacent variables, which constitute the variable's *Markov blanket*:

$$(2.5) \quad \forall_{x, x' \in \mathbf{x}} \quad x \perp x' \mid N(i)$$

As a consequence of these independences, the PMF can be rewritten as the product of a set of *potential functions* whose domains are maximal cliques of variables.

$$(2.6) \quad P(\mathbf{x}) = \prod_{C \in \text{cl}(\mathbf{N})} \phi_C(\mathbf{x}_C)$$

In a sparse network, these cliques can be parameterized much more efficiently than an arbitrary joint PMF. If the largest clique has size  $c$ , values of the function can be exhaustively enumerated in a table of  $k^c$  parameters, making the full representation of size  $|\mathbf{w}| \leq |C|k^c$ .

### 2.3.1 Factor graphs.

Further reduction in parameters can be achieved by identifying specific factors that are not maximal cliques in the MRF. A *factor graph* explicitly defines  $m$  factors by replacing  $\mathbf{N}$  with an  $m \times n$  binary

---

<sup>1</sup>In many models, there are several classes of discrete variables, with different number of states; I ignore this for simplicity. Similarly, I will not belabor the fact that, since likelihood functions are invariant to multiplication by a constant, enumerating values for every combination often creates a redundant parameter.



matrix  $\mathbf{M}_{\mathbf{x}}$  describing a bipartite graph between each factor  $\phi \in \Phi$  and the variables in its domain. As with  $\mathbf{N}$ , I define an equivalent function  $M_{\mathbf{x}}$  such that  $M_{\mathbf{x}}(\phi)$  returns the set of variables in  $\phi$ 's domain, allowing the updated factorization

$$(2.7) \quad P(\mathbf{x}) = \prod_{i=1}^m \phi_i(\mathbf{x}_{M_{\mathbf{x}}(\phi_i)})$$

Clearly we can produce a factor graph for any MRF by creating a factor for each clique; we can also produce an MRF for any factor graph by defining  $\mathbf{N} = \mathbf{M}_{\mathbf{x}}^t \mathbf{M}_{\mathbf{x}}$ . However, the latter transition is lossy, as a factor graph can represent additional structures – including, but not limited to those captured by directed graphical models such as Bayes nets [42]. Figure 2.1 shows an example where a factor graph's representational capacity allows a reduced parameterization. If we consider three random variables, each of which can take  $k = 4$  distinct states, and whose MRF is fully connected (a). Enumerating the probabilities of each of their joint states requires  $4^3 - 1 = 63$  parameters. A factor graph with equivalent independences (b) requires the same parameterization. However the factor graph in (c) shows that the PMF is separable into pairwise potentials, which can be represented with  $3 * (4^2 - 1) = 45$  parameters. The potential savings of the factor graph representation can increase dramatically with  $k$  and when  $\mathbf{M}_{\mathbf{x}}$  is significantly sparser than  $\mathbf{M}_{\mathbf{x}}^t \mathbf{M}_{\mathbf{x}}$

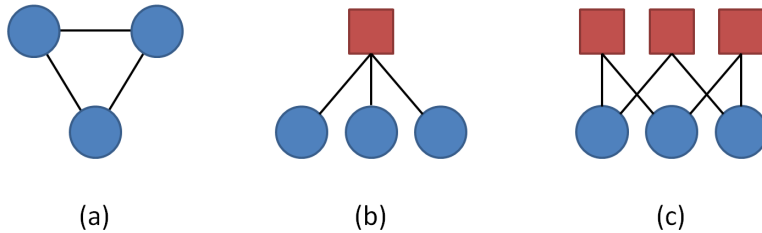


Figure 2.1: Blue circles are variables; red squares are factors. The Markov network (a) is shared by both factor graphs (b) and (c), but the maximum clique factor (b) cannot be parameterized as succinctly as three pairwise factors (c).

### 2.3.2 Parameter sharing.

So far we have considered only a naive parameterization for our PGMs, the exhaustive description of local probabilities for each factor. To describe abbreviated parameterizations, we introduce the real vector  $\mathbf{w}$ , which contains  $d$  parameters. An associated  $m \times d$  adjacency matrix  $\mathbf{M}_{\mathbf{w}}$  specifies which parameters are necessary for the computation with which factors (as usual we define  $M_{\mathbf{w}}$

as a function accessing the same information). Initially we will assume  $\mathbf{w}$  is externally determined and include it as a subscript on the probability function below.  $z(\mathbf{w})$  is a normalizer, also called the *partition function*, which ensures that probabilities sum to 1 over all possible assignments to  $\mathbf{x}$ .

$$(2.8) \quad P_{\mathbf{w}}(\mathbf{x}) = \frac{1}{z(\mathbf{w})} \prod_{i=1}^m \phi_i(\mathbf{x}_{M_{\mathbf{x}}(\phi_i)}; \mathbf{w}_{M_{\mathbf{w}}(\phi_i)})$$

Introducing  $\mathbf{w}$  and  $\mathbf{M}_{\mathbf{w}}$  allows us to establish *parameter sharing* between factors by giving them identical (or overlapping) adjacencies. Parameter sharing is the network generalization of stationary transition probabilities in a simple Markov chain. In a chain, it can be used to encode beliefs such as “alive rarely follows dead in any series of patient health states”. In a network an analogous statement might be “adjacent pixels are twice as likely to be the same as different”. In addition to making our representation more computationally tractable, these have an effect similar to IID assumptions in allowing us to use more information in estimating a variable, reducing the potential for overfitting.

Parameter sharing is also a crucial step in allowing the same model be *instanced* for application to different datasets. A common approach in procedurally defining factor graphs is to give generators which reproduce the same factor potential many times, referencing the same parameters, but with differing variable domains. This results in large blocks of identical rows in  $\mathbf{M}_{\mathbf{w}}$ , which we sometimes refer to as *instances* of the same factor. When the same generator is applied to different datasets, it is possible to create factor graphs of many different shapes that reference the same parameter vector  $\mathbf{w}$ . In supervised learning, this is used to fit parameters to a training dataset and apply them to a new problem instance, which may have a very different shape.

### 2.3.3 Discriminative models.

In some applications, there is a clear division between the *hidden* variables  $\mathbf{x}$ , whose values must be inferred, and a second vector  $\mathbf{y}$  of variables that will be observed during inference. In these cases we can often achieve better results by ignoring the process that generates  $\mathbf{y}$  and learning only the conditional distribution  $P(\mathbf{x} \mid \mathbf{y})$ . To simplify discussions of scalability we sometimes assume  $|\mathbf{x}| = |\mathbf{y}| = n$ . We incorporate  $\mathbf{y}$  into our factor graph by introducing an  $m \times n$  matrix  $\mathbf{M}_{\mathbf{y}}$  (and associated function  $M_{\mathbf{y}}$ ) indicating which observed variables affect the computation of which factors. The final PMF factorization is as follows.

$$(2.9) \quad P_{\mathbf{w}}(\mathbf{x} | \mathbf{y}) = \frac{1}{z(\mathbf{w}, \mathbf{y})} \prod_{i=1}^m \phi_i(\mathbf{x}_{M_{\mathbf{x}}(\phi_i)} | \mathbf{y}_{M_{\mathbf{y}}(\phi_i)}; \mathbf{w}_{M_{\mathbf{w}}(\phi_i)})$$

It is worth noting that while there can be factors with adjacencies in  $\mathbf{x}$  but none in  $\mathbf{y}$ , the inverse is not true - or at least is wasteful, as inclusion of  $\mathbf{y}$ -only factors will be irrelevant during inference. In the models we will be discussing,  $\mathbf{y}$  consists of the raw sensor data we have available to us, and might include both discrete and continuous variables. In the following section on previous approaches to spatial, temporal and relational PGMs I will discuss several forms of continuous factors, in which categorical variables depend on real valued observed variables as well as real valued parameters.

### 2.3.4 Representation.

As a final summary of our model, we define two tuples, the *factor graph* which contains structural information about the distribution in the form of factor definitions and adjacency matrices, and the *inference state* which contains factor parameters, observed and hidden variable values necessary to compute the PMF in equation 2.9.

$$(2.10) \quad \mathcal{FG} : \{\Phi, \mathbf{M}_{\mathbf{x}}, \mathbf{M}_{\mathbf{y}}, \mathbf{M}_{\mathbf{w}}\} \quad \mathcal{IS} : \{\mathbf{w}, \mathbf{y}, \mathbf{x}\}$$

Figure 2.2 gives a schematic illustrating the memory layout of a complete representation of both tuples. It is worth noting that for simplicity of notation and illustration, I've presumed that the unobserved variables  $\mathbf{x}$  can each take on  $k$  distinct values, and that the factors each compute local likelihoods based on  $l$  observed variables,  $l$  unobserved variables, and  $l$  parameters. These are meant to be taken as upper bounds. In practice  $\mathbf{x}$  would be partitioned into variable domains with their own category spaces of potentially varying cardinality.  $\Phi$  would be partitioned into regions of "instanced" factors representing the same relation among different sets of variables, with each partition having a specific dimensionality for the unobserved, observed, and parameter portions of its domain.

The spatial complexity of the upper bound representation is  $O(m(n+d))$ . In practice, however, it is more common to describe and implement factor graphs in terms of generators that exploit the sparsity of  $\mathbf{M}_{\mathbf{x}, \mathbf{y}, \mathbf{w}}$ , both in terms of blocks within matrices and the limited row ranks induced by finite dimensionality of factor domains. In addition to reducing memory consumption these implementations allow faster access to adjacencies. In the next section I give several examples of generators associated with spatial, temporal and relational regularities.

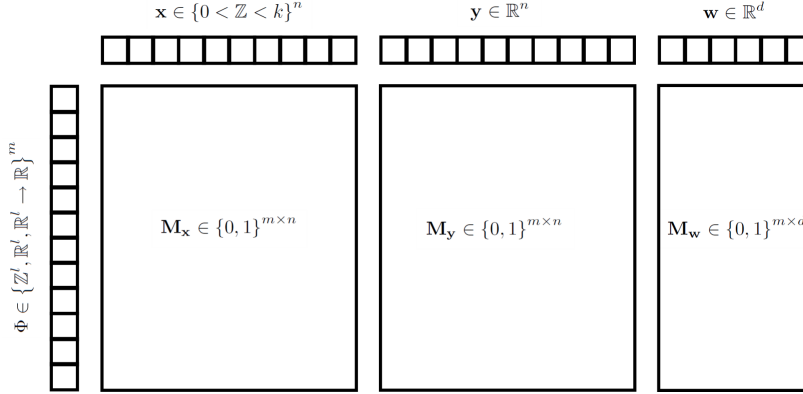


Figure 2.2: Schematic of a factor graph with associated inference state.

**Supervised learning.** Although my contributions are centered on *unsupervised* learning, it is useful to first review how the parameter vector  $\mathbf{w}$  is determined when training data are available. By far the most common objective in *supervised* learning is to find factor parameters which maximize the expectation of a training set for which  $\mathbf{x}$  is known.

$$(2.11) \quad \mathbf{w}^*(\mathbf{x}) = \underset{\mathbf{w}}{\operatorname{argmax}} P_{\mathbf{w}}(\mathbf{x} | \mathbf{y})$$

Factor parameterizations can often be chosen to be convex with respect to  $\mathbf{w}$ , making it attractive to optimize via gradient methods. The optimization is done in log-space, so the gradient with respect to  $\mathbf{w}$  can be written as follows.

$$(2.12) \quad \nabla \log P_{\mathbf{w}}(\mathbf{x} | \mathbf{y}) = -\frac{\nabla z(\mathbf{w}, \mathbf{y})}{z(\mathbf{w}, \mathbf{y})} + \sum_{\phi \in \Phi} \frac{\nabla \phi(\mathbf{x}_{M_{\mathbf{x}}(\phi)} | \mathbf{y}_{M_{\mathbf{y}}(\phi)}; \mathbf{w}_{M_{\mathbf{w}}(\phi)})}{\phi(\mathbf{x}_{M_{\mathbf{x}}(\phi)} | \mathbf{y}_{M_{\mathbf{y}}(\phi)}; \mathbf{w}_{M_{\mathbf{w}}(\phi)})}$$

Factors are generally parameterized in a way that makes the gradients in the rightmost term easily computable. For example, the most common representation for discrete factors is  $\phi(\mathbf{v}) = \exp w_{\mathbf{v}}$ , requiring one parameter for each configuration  $\mathbf{v}$  of the domain variables. The left hand term is potentially more difficult, although it is possible to approximate the partition function by conducting inference as discussed below at each step. A commonly used, easier to compute estimate is *the pseudolikelihood*, which approximates the probability of the data as the product of marginal probabilities for each variable given its Markov blanket.

$$\begin{aligned}
(2.13) \text{ PL} &= \prod_{x \in X} \frac{\prod_{\phi \in M^{-1}(x)} \phi(x|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w})}{\sum_{v \in \mathbf{Z}_k} \prod_{\phi \in M^{-1}(x)} \phi(v|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w})} \\
(2.14) \text{ PL} &= \sum_{x \in X} \left( \sum_{\phi \in M^{-1}(x)} \phi(x|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w}) - \log \sum_{x' \in \mathbf{Z}_k} \prod_{\phi \in M^{-1}(x)} \phi(x'|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w}) \right) \\
(2.15) \text{ PL} &= \sum_{x \in X} \left( \sum_{\phi \in M^{-1}(x)} \nabla \phi(x|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w}) - \frac{\sum_{x'} \left( \sum_{\phi} \frac{\nabla \phi(x'|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w})}{\phi(x'|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w})} \right) \left( \prod_{\phi} \phi(x'|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w}) \right)}{\sum_{x'} \prod_{\phi} \phi(x'|\mathbf{x} \setminus x, \mathbf{y}; \mathbf{w})} \right)
\end{aligned}$$

In the rightmost term of equation 2.14, we can see that in place of the global partition function we must calculate only a local partition function for each variable  $x$ . As long as both factors and factor gradients are available, the local partition function and its gradient can be computed very efficiently. In practice, efficient implementations can exploit sparsity and regularity in the factor graph, as well as regularity in parameterization, to avoid many redundant calculations implied by the summation and product iterations above.

### 2.3.5 Inference.

The most common inference operations in graphical models are *marginalization*, which computes a posterior distribution for each hidden variable independent of all others, and *maximum likelihood estimation* (MLE), which finds the single most probable configuration for all variable<sup>2</sup>. My contributions are more closely related to MLE, which can be summarized by the equation

$$(2.16) \quad \mathbf{x}^*(\mathbf{w}) = \underset{\mathbf{x}}{\operatorname{argmax}} P_{\mathbf{w}}(\mathbf{x})$$

Although MLE is NP-complete for general factor graphs with cycles, it can often be quickly approximated with *maximum likelihood belief propagation*. MLBP proceeds by iteratively updating a series of messages between pairs of adjacent variables and factors (in both directions). From variable to factor, the equation is:

$$(2.17) \quad m_{x \rightarrow \phi}(v) = \prod_{\phi' \in M_{\mathbf{x}}^{-1}(x) \setminus \phi} m_{\phi' \rightarrow x}(v)$$

---

<sup>2</sup>It is worth noting that the MLE may not result in a good estimate of any individual variable, as the single most probable joint configuration may have little in common with the majority of runners-up. Another popular inference task *marginalization*, estimates the *a posteriori* distribution of each individual hidden variable given those observed. Marginalization for general factor graphs belongs to the class P, but can be approximated quickly with belief propagation methods similar to those described for MLE.

In other words, the variable publishes an aggregation of what other factors are telling it about its likelihood of taking on each value  $v$ . From factor to variable, the equation is

$$(2.18) \quad m_{\phi \rightarrow x}(v) = \max_{\mathbf{v} \in \mathbb{Z}_k^{|\mathbf{M}_{\mathbf{x}}(\phi)|-1}} \left( \phi(\mathbf{v} \cup v) \prod_{x' \in \mathbf{M}_{\mathbf{x}}(\phi) \setminus x} m_{x' \rightarrow \phi}(v_{x'}) \right)$$

In other words, the factor considers each possible assignment to its domain variables and broadcasts, for each value, the likelihood of the most likely configuration which assigns that value.

Messages are initialized based on random assignments, and then iteratively updated based on the equations above. It can be shown that each update (weakly) reduces a free energy equation which has a minimum at the true maximum likelihood estimate. Although updates may occasionally cycle, and can come to rest at local maxima, empirical results in a wide variety of models have proved MLBP effective in practice.

### 2.3.6 The Bayesian view.

After discussing learning and inference as distinct problems, it is interesting to review equation 2.9 and note that the distinction between parameters  $\mathbf{w}$  and variables  $\mathbf{x}$  is somewhat arbitrary. Ignoring the mingling between continuous and categorical variables<sup>3</sup>, imagine we were to merge  $\mathbf{w}$  into  $\mathbf{x}$  and  $\mathbf{M}_{\mathbf{w}}$  into  $\mathbf{M}_{\mathbf{x}}$ . The result would be a valid factor graph taking the Bayesian perspective that parameters are themselves random variables. Instead of conducting learning and inference separately, we could apply a modified belief propagation which reconciled both at once.

Treating parameters in the same manner as other random variables has an appealing elegance and simplicity, and Bayesian inference is a proved approach to unsupervised learning in many domains. In this thesis, however, my approach is more closely related to EM-clustering than Bayesian inference, in that I treat parameters and variable assignments as quantities to be optimized in parallel, but with distinct processes. My reasons for maintaining this distinction are as follows.

- In the models I will introduce,  $\mathbf{w}$  are the only real values which must be inferred. Describing the procedures for optimizing them separately from those for optimizing  $\mathbf{x}$  is a natural way of taking advantage of the categorical or continuous nature as appropriate.

---

<sup>3</sup>Hybrid factor graphs with real-valued variables are common in some domains, and differ from discrete factor graphs primarily during inference. Well selected conjugate priors can be used to give the messages in equations 2.17 and 2.18 finite parameterizations.

- The sparsity structure of  $\mathbf{M}_{\mathbf{x},\mathbf{y},\mathbf{w}}$  is a crucial factor in the performance of message passing algorithms, but this structure differs significantly between  $\mathbf{M}_{\mathbf{x}}$  and  $\mathbf{M}_{\mathbf{w}}$ .  $\mathbf{M}_{\mathbf{w}}$  tends to feature large blocks of identical rows that are a function of parameter sharing. In  $\mathbf{M}_{\mathbf{x}}$ , regularities tend to be “diagonal”, in that variables participate in the same factor because they are adjacent in a spatial, temporal, or relational sense. The presence of these orthogonal forms of structure would create long range ties in the associated factor graph, which are associated with poor performance and non-convergence in belief propagation algorithms. I hope to show that these problems can be avoided with a more EM-like approach.
- I address situations in which the number of classes for a latent categorical variable, and therefore the number of associated parameters, are not known in advance. In the Bayesian factor graph this would involve introducing new variables and edges whenever a new class is considered; I view it as simpler to instead discuss manipulations of  $\mathbf{w}$ .

Those differences notwithstanding, the EM and Bayesian approaches are closely related. EM is often described as “semi-Bayesian”, and Beal [12] has shown that with very slight modifications, EM clustering such as  $k$ -means can be shown to be a special case of Bayesian inference.

## 2.4 Kernel Conditional Density Estimation

The clustering methods and probabilistic graphical models discussed in the previous sections address discrete unknowns in a dataset, such as missing values and hidden entities. The third framework I build on, Kernel Conditional Density Estimation (KCDE), addresses the problem of inferring from discrete observations a continuous function which describes their distribution. As I show in chapter 6, the use of kernel methods allows us to consider structures in continuous space which influence networks, and the use of density estimation allows us to consider relationships that are poorly described by a single, modal outcome. What follows is intended as a brief introduction to the topic and my notation, for greater detail I refer the reader to [113] (especially chapters 4-6, for introductory topics) and [71] (especially chapters 3-6, for multivariate and conditional density estimation).

### 2.4.1 Regression

An antecedent to KCDE, kernel regression, treats the problem of estimating a single-valued function in continuous space when a limited number of (potentially erroneous or distorted) observations are

available. Beginning with the one-dimensional case, we may consider two vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$  such that each  $(x_i, y_i)$  is an observation of the function we wish to estimate. As with many regression techniques, we seek to describe a function with minimum sum of squared errors in our sample size, as such an estimator would minimize variance of error as our number of samples grew to infinity. Rather than presuming a functional form and estimating its parameters, we can utilize the Nadaraya-Watson kernel estimator defined as follows [90] [115]<sup>4</sup>.

$$(2.19) \quad \hat{f}(x) = \sum_{i=1}^n y_i \frac{k(h^{-1}(x - x_i))}{\sum_{j=1}^n k(h^{-1}(x - x_j))}$$

The two symbols above not previously introduced are a kernel function  $k$  and a bandwidth  $h$ . Formally, a kernel must be symmetric ( $K(x) = K(-x)$ ) and a probability distribution function or PDF ( $K(x) \geq 0, \int_{-\infty}^{\infty} K(x)dx = 1$ ), some examples of which are given below. With the proper normalization factor, a kernel can be constructed from any decay function as described in section 1.1. Used in a weighted average as above, any kernel will produce the same estimate given enough data, but the shape of the kernel can determine properties such as the handling of sparse or highly clustered data, which I explore further in experiments in chapter 6.

$h$  is the *bandwidth*, and acts as a scaling parameter on the  $x$  axis. Large values of  $h$  create a smoother function: as all points become equally relevant, the function at all points converges toward the sample mean. Smaller values allow for greater local variation. In general, a higher bandwidth results in estimates with lower variance from the true value, but greater bias.  $h$  is chosen by balancing these traits via the minimization of some risk function, most commonly mean squared error. As a shorthand I generally refer to the scaled kernel  $K(x; h) = \frac{k(h^{-1}x)}{h}$ , which is itself a valid PDF.

The anatomy of our kernels implies certain assumptions about the regularity of the relationship between  $x$  and  $y$ . The symmetry condition implies that all directions in  $x$  are equally “lossy” with respect to mutual information about  $y$ , and the use of a constant bandwidth implies that the scale of this loss is constant throughout the space. Both conditions become irrelevant given enough samples, but these conditions must be close to true to get good results on a finite dataset. Situations where certain regions of  $x$  are scaled differently from others can be addressed using locally adaptive bandwidths [19]. If directional effects are known in advance, it is sometimes possible to perturb  $x$  to remove them. For example, if  $f \approx g(\log x)$ , where  $g$  is an unknown function believed to fit the

---

<sup>4</sup>Nadaraya Watson is one of several commonly used kernel estimators. I refer the reader to [56] for a comparison and synthesis of alternatives.



conditions above, a nonparametric estimator  $\hat{f}(\log x)$  may converge to better estimates with fewer samples.

### 2.4.2 Density Estimation

Above I motivated kernel regression as the search for the true value of a deterministic function observed through errored samples,  $y_i \sim f(x_i) + \epsilon_i$ , where  $\epsilon_i$  is some noise variable. The function and error model together can be viewed as a conditional distribution,  $P(y | x)$ , for which the estimator  $\hat{f}(x)$  gives local means – a meaningful summary for many applications. However, for other applications we may be interested in the entire density function - for example, to estimate sample variance at different values of  $x$ , or to detect a multimodal distribution suggesting a nondeterministic underlying relationship or a complex error process.

For now I set aside  $\mathbf{x}$  and consider only a vector of points  $\mathbf{y}$  (for continuity with the above, we may imagine that they are samples at a single  $x$  value for which we wish to estimate the distribution). Kernel density estimation estimates the PDF by placing a kernel-shaped probability mass centered at each observed value. Since each such mass is itself a valid PDF, normalizing by the number of samples ensures that the sum is also valid.

$$(2.20) \quad \hat{P}(y) = \sum_{i=1}^n \frac{K(y - y_i)}{n}$$

Since we have no direct observations of the PDF, bandwidth is typically optimized via *cross-validation*: we consider a series of subsets  $\mathbf{s} \in S[0, 1]^n$  of our data, and choose the bandwidth which minimizes the variance of the density estimate across these subsets, integrated over the variable domain. The most common choice for  $S$  is all sets omitting a single data point, also referred to as *leave-one-out cross-validation*. Stronger theoretical guarantees may be reached by considering the entire power set, and computational performance may be improved by sampling a smaller set for validation. Since the loss function given below is convex, it can generally be optimized with gradient-based numerical methods.

$$(2.21) \quad \hat{h} = \operatorname{argmin}_h \int \hat{f}(y; h)^2 dx - \frac{2}{|S|} \sum_{\mathbf{s} \in S} \{\forall_i \hat{f}(y_i; h)\} \cdot \mathbf{s}$$

Returning to a two variable setting, we might consider the relationship between  $x$  and  $y$  by modeling the joint distribution of our samples,  $P(x, y)$ . Density estimation for two or more variables can be accomplished using any symmetric multivariate density as a kernel. Generalizing

the notion of bandwidth to the multivariate case is nontrivial and an area of active research, as the parameterization can potentially include information about the orientation as well as degree of smoothing. In this thesis I consider only *product kernels* describable for a distribution of  $n$  variables in terms of a univariate kernel function as below.

$$(2.22) \quad W(\mathbf{x}; \mathbf{h}) = \prod_i^n K(x_i; h_i)$$

Equation 2.20 may be trivially updated for the multivariate case by considering a vector difference within the kernel function and normalizing by the product of marginal bandwidths. Bandwidth estimation may still be accomplished via least squared cross-validation, but computational difficulty of the optimization increases exponentially in greater dimensions. This is compounded by the fact that, due to the curse of dimensionality, the number of samples necessary to achieve comparable levels of estimation accuracy also increases exponentially. Kernel based estimation of joint distributions of more than 4 or 5 variables is not generally practical, methods for doing so by exploiting sparsity in variable dependencies are an area of active research, *i.e.* [77].

### 2.4.3 Conditional and Categorical Density Estimation

Armed with estimators for  $P(y)$  and  $P(x, y)$ , we may invoke Bayes law to estimate the conditional density,

$$(2.23) \quad P(y | x) = \frac{P(x, y)}{P(x)} \approx \frac{\hat{P}(x, y)}{\hat{P}(x)}$$

A complication with this approach is that bandwidths we selected to minimize estimation risk for the joint and marginal distributions do not necessarily match on the overlapping variables, and are not necessarily those which minimize risk in the conditional distribution. The complexity of the consistent estimator for mean-squared-error risk in the conditional case puts it beyond the scope of this introduction; it is furthermore computationally unattractive to optimize as a single evaluation of the risk is  $O(n^3)$  for sample size  $n$ . The alternative method I employ in this thesis is the maximum likelihood estimate below, whose objective can be evaluated in  $O(n^2)$ .

$$(2.24) \quad \mathbf{h}^* = \operatorname{argmax}_{\mathbf{h}} \sum_{i=1}^n \ln \frac{\hat{P}(x_i, y_i; \mathbf{h})}{\hat{P}(x_i; \mathbf{h})}$$

So far, I've discussed kernel estimation only for continuous variables. In chapter 6 I estimate binary variables, conditioned on both continuous and categorical variables. Univariate kernel techniques typically assume that all categories are equally "far apart" for smoothing processes. For a categorical variable with  $m$  categories, the discrete kernel is a PMF over the possible pairings between a fixed sample and a second one yet-to-be observed. The bandwidth parameter  $\lambda$  which specifies the amount of probability mass that is divided between all non-matched inputs.

$$C_m(x = x'; \lambda) = \begin{cases} 1 - \lambda & : x = x' \\ \frac{\lambda}{m-1} & : x \neq x' \end{cases}$$

As  $\lambda \rightarrow 0$ , each category is treated as a separate distribution, and no smoothing occurs. As  $\lambda \rightarrow 1$ , the variable is deemed irrelevant as all category assignments produce the same probability density.

Combining discrete and continuous kernels is somewhat involved when MSE bandwidth fitting or general multivariate kernels are involved. In this thesis I restrict myself to product kernels and maximum likelihood estimation of bandwidth parameters, the combination of which allow discrete kernels to be trivially substituted for continuous ones in the operations previously described.



# Chapter 3

## Data

### 3.1 AIS Data (AIS)

The Automated Identification System (AIS) is a communication standard for ocean vessels used by ships and ground stations to coordinate shipping traffic. AIS transponders on compliant vessels are integrated with the ship's radio, GPS, and navigational control systems. When pinged (via broadcast messages from other boats or ground stations), the transponder replies with a radio packet containing ship identity, current GPS coordinates, heading, speed, and various other fields describing navigational state, destination, and more. AIS compliance is required on ships over a certain size by most commercial ports, making it essential for most sizable merchant vessels operating worldwide.

For 5 days in June 2005, a sensor network queried Automated Identification System (AIS) transponders on merchant marine vessels navigating the English Channel. In total, the sensor sweep captured movements of over 1700 vessels were recorded, with activities ranging from simple shipping lane traversals to apparently complex itineraries with stops at multiple ports of call. The reasons for the collection of the data are primarily security related. The global shipping system plays a prominent role in a variety of terrorist attack scenarios, both in the United States and abroad: in any country, the ports are both the most likely means of entry for bombs and other weapons, and themselves a prime economic and symbolic target.

In addition to being an attractive target, ports are currently considered not secure – for example, it has been suggested that only 3% of shipping containers entering the United States are directly inspected by customs officials. The sheer volume of commerce conducted via international shipping makes naive attempts at greater security infeasible, as neither the direct costs associated

with detailed surveillance nor the indirect costs incurred by reducing industry efficiency are easily absorbed. If automated techniques such as those designed above can give insight into the behavioral patterns and structural features of the merchant marine population, then limited budgets for surveillance and interdictions can be more precisely targeted to have the greatest impact on overall security. The data under analysis here is especially promising as it represents the result of a relative inexpensive, passive, and consensual surveillance effort.

In many sensor datasets, physical limitations of the sensors are a primary source of error; for example, an error of 10m in a car-installed GPS system can introduce ambiguity as to which street the car is on. In the case of AIS data, the physical error of sensors is so small compared to the scale of navigation (some tankers are themselves 400m long) that modeling sensor error is less relevant. Instead, a primary source of error comes from creative utilization of user-input fields such as destination and navigational status. I chose to focus only on numeric fields that would be drawn directly from navigational computers. Even within this set, there cases likely due to misconfiguration which, for example, reported 0 latitude and 0 longitude for the study duration. I preprocessed to eliminate all ships with constant or out-of-range values for any numeric field.

AIS responses in the original dataset were intermittent with inconsistent inter-arrival times. Although work exists regarding the use of temporally irregular observations(e.g. [37]), I chose instead to standardize approximate interarrival times by filtering data to produce streams of observations in which at least 45 minutes and at most 180 minutes pass between observations. I also remove ships that make fewer than 5 consecutive reports, yielding a dataset of 10935 sequences of 576 ships. I also removed 140 erroneous responses sent by malfunctioning or otherwise corrupted responders. Figure 3.1 shows the final dataset as visualized using Google Earth [2].

## 3.2 Company Relations Data (B2B)

Coase's classical theory [27] defines a firm as a sort of bubble. Within, transactions are organized according to a charter, but between, all coordination is done through market mechanisms. Real firms engage in many activities that blur this (intentionally simplified) boundary. These include nested firm structures, deliberately segmented markets, and the subject of this section of my thesis: persistent inter-company alliances. These relations may be formed to ensure stable production, support new technologies, or compete against other coalitions.

The evolving network of such interactions is a natural subject for network analysis, but is challenging to operationalize due to the diversity and complexity of intercompany relations. One common approach is to connect companies which share one or more members on their board of

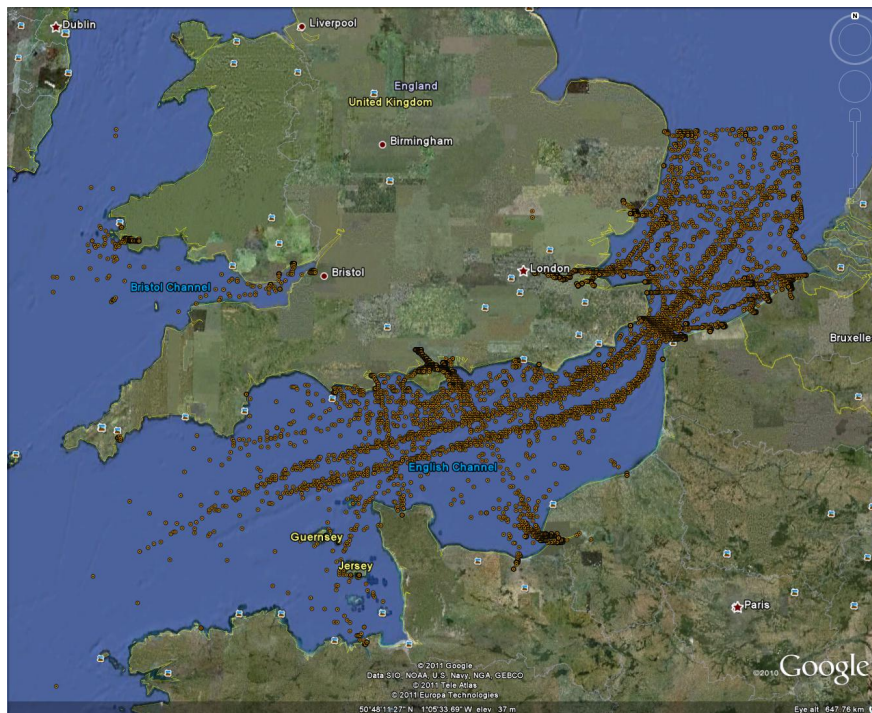


Figure 3.1: Observation points in AIS data

directors [86] [97]. Such “interlock networks” are attractive in being based on public information, and are unambiguous in providing a network structure. However, they are by definition relevant to understanding only those network phenomena which are governed by a small power-elite (much of the literature in this area is devoted to demonstrating exactly which phenomena fit this category).

In this thesis, I take an alternate approach based on public disclosures of supply contracts. Under most modern joint ownership arrangements, the managers of a publicly held company are required to disclose information necessary for an investor to make informed decisions regarding the sale or purchase of stock. In the United States, this release of information is regulated by the Securities and Exchange Commission, which mandates a series of public forms that constitute a standardized record of a company’s public activities. Among the activities that must be disclosed are “material definitive agreements not made in the ordinary course of business”, a category which has come to include a variety of inter-company relations.

The SEC publishes this information in machine readable formats via the EDGAR electronic database [1]. However, since the forms consist primarily of unstructured text fields, substantial pre-processing is necessary to extract data suitable for quantitative analysis. Many private firms resell data processed in particular ways for investors interested in particular aspects of company

activities. One in particular, Revere Data, LLC, has developed a specialty in extracting intercompany relationships. Using a mix of machine processing and human analysts, Revere processes SEC releases and other data sources on a daily basis, and consolidates relationship information in the Revere Relationships database. Their ontology classifies relationships as belonging to the following non-exclusive categories.

- A **competitor** participates in markets for a similar range of products. The markets and overlaps are defined in another Revere product, Revere Hierarchy, and numeric values regarding the amount of overlap are provided.
- A **customer** contracts another company for supplies necessary to provide their product or service. The SEC criterion for disclosure of such relationships is that they be of material interest to an investor; the typical interpretation is that the dollar amount of such transaction must equal some percentage of the revenue of the reporting party. When available this percentage is recorded with the relationship record.
- A **partner** is engaged in some type of shared enterprise with the source company. These relationships are binary and symmetric. Partnerships are divided into another layer of categories such as “joint venture”, “distribution”, or “technology”.

In this thesis I focus particularly on customer relationships, for two primary reasons. First, the presence of a normalized numerical weighting on relationships (the revenue share reported by the selling company) makes it easier to establish a threshold which omits some reported but insignificant links. Second, the relationship is transitive, in the sense that any long path through the network represents a potential flow of goods and services in one direction, and money in the other. Transitivity is important in the interpretation of network metrics such as Eigenvalue centrality, and is not a property of the other relationship types.

Taken together, these customer relationships describe the public business-to-business (B2B) economy of the United States. It is estimated that over 80% of dollars transacted in the US serve B2B commerce. Figure 3.2 shows a snapshot of this network from July, 2007. The view is limited to companies participating in relationships which were expected to constitute more than 10% of the revenue for the supplier. There exists a giant component containing the a large proportion of the companies, and for simplicity the view excludes many companies that were not a part of this group. Often these smaller components represented specific such as the airlines, or the members of a diversified conglomerate such as Berkshire Hatheway. Many companies not part of the giant component are excluded from the view. Nodes are colored according to a top level sector provided



by the company (examples include “energy”, “technology” and “health care”). Table 3.1 provides some vital statistics regarding the network.

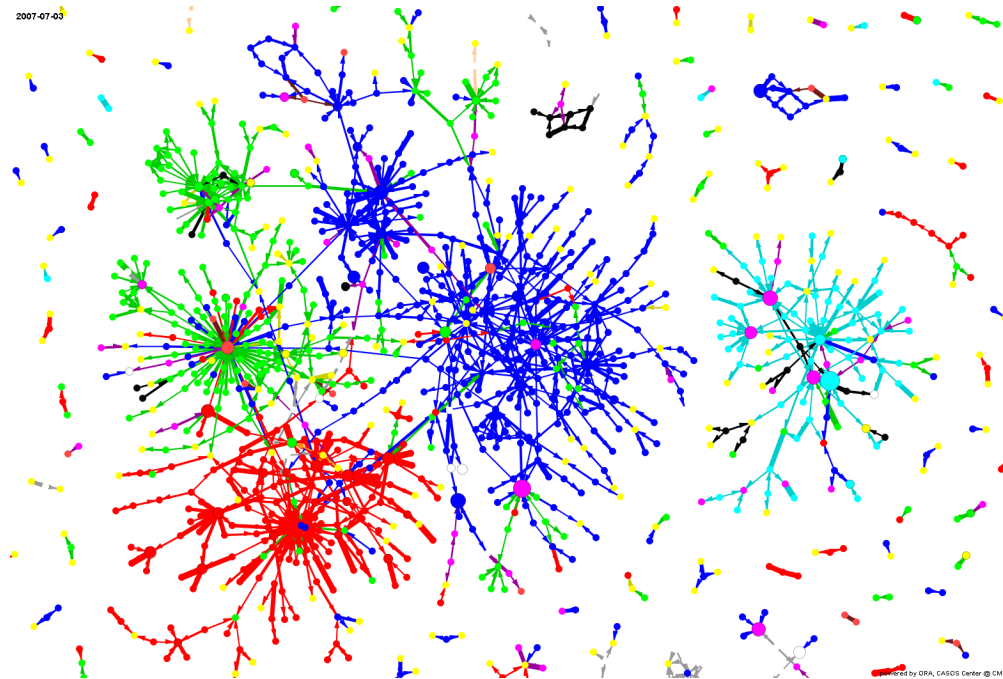


Figure 3.2: Business-to-business supply network extracted from the Revere Relationships database, July 7 2007.

Figure 3.2 illustrates a static network that is one snapshot of the Revere database. The true network is dynamic, changing constantly as new companies and relationships are formed and old ones dissolved. These changes are reported on a daily basis via SEC filings, but ambiguities in the exact timing of a change and a permitted 4 business day lag between an event and filing make it somewhat misleading to analyze network changes at this temporal resolution. I was given access to archives of Revere’s databases, and conducted exploratory analysis to determine whether to evaluate changes on a weekly, monthly, quarterly or yearly basis data. I elected to use quarterly snapshots in this thesis because this was the first interval at which average hamming distances between successive snapshots were significant, and because quarterly measurements mesh well with other readily available financial data. Table 3.1 shows vital statistics regarding the resulting network’s change over time. In each time frame I repeated the conditions under which the figure above was generated:

- Only supply chain relationships constituting 10% or more of the supplier’s revenue are considered.

Quarter	Companies	Links	GC Size	GC Links	GC Diameter
2006 Q1	1229	1437	785	1078	23
2006 Q2	1240	1463	785	1088	21
2006 Q3	1236	1452	776	1064	20
2006 Q4	1239	1484	768	1072	18
2007 Q1	1261	1510	776	1084	20
2007 Q2	1201	1437	738	1045	20
2007 Q3	1328	1607	826	1183	21
2007 Q4	1311	1587	801	1145	20
2008 Q1	1248	1504	880	1249	26
2008 Q2	1198	1438	839	1191	27
2008 Q3	1192	1423	848	1184	21

Table 3.1: Quarterly network measures for B2B network.

- Isolates in the remaining network are omitted from analysis.
- When network measures are defined only on a single component, they are evaluated only on the nodes in the giant component.

Table 3.1 reveals some interesting temporal properties of the B2B network. One point is that the size and composition of the larger network remains relatively stable over the 3 years, rather than growing alongside related factors such as US GDP or population. Another interesting factor is the relative density of the giant component compared to the overall graph. This may suggest a latent classification in companies, whereby they tend to either have many connections and participate in the main economic network, or have no connections at all. This is born out by inspection of smaller components, where we find conglomerate groups (such as Berkshire Hatheway) and specialized industries (such as the major airlines) operating in isolation from the rest of the economy. The sudden jump in giant component size in 2007 is due to connections initiated from the energy sector, previously the second largest component, into the rest of the network. This may be reflective of a true economic shift, but is more likely an artifact of my data cleaning process and an indicator of the inherent instability of some graph-theoretic concepts such as “largest component”. Either because of this merger of components or for an independent reason, there is not a clear trend of decreasing component diameter as had been observed in many other networks by [69].

Figure 3.3 provides summary information for several node-level attributes of the B2B network

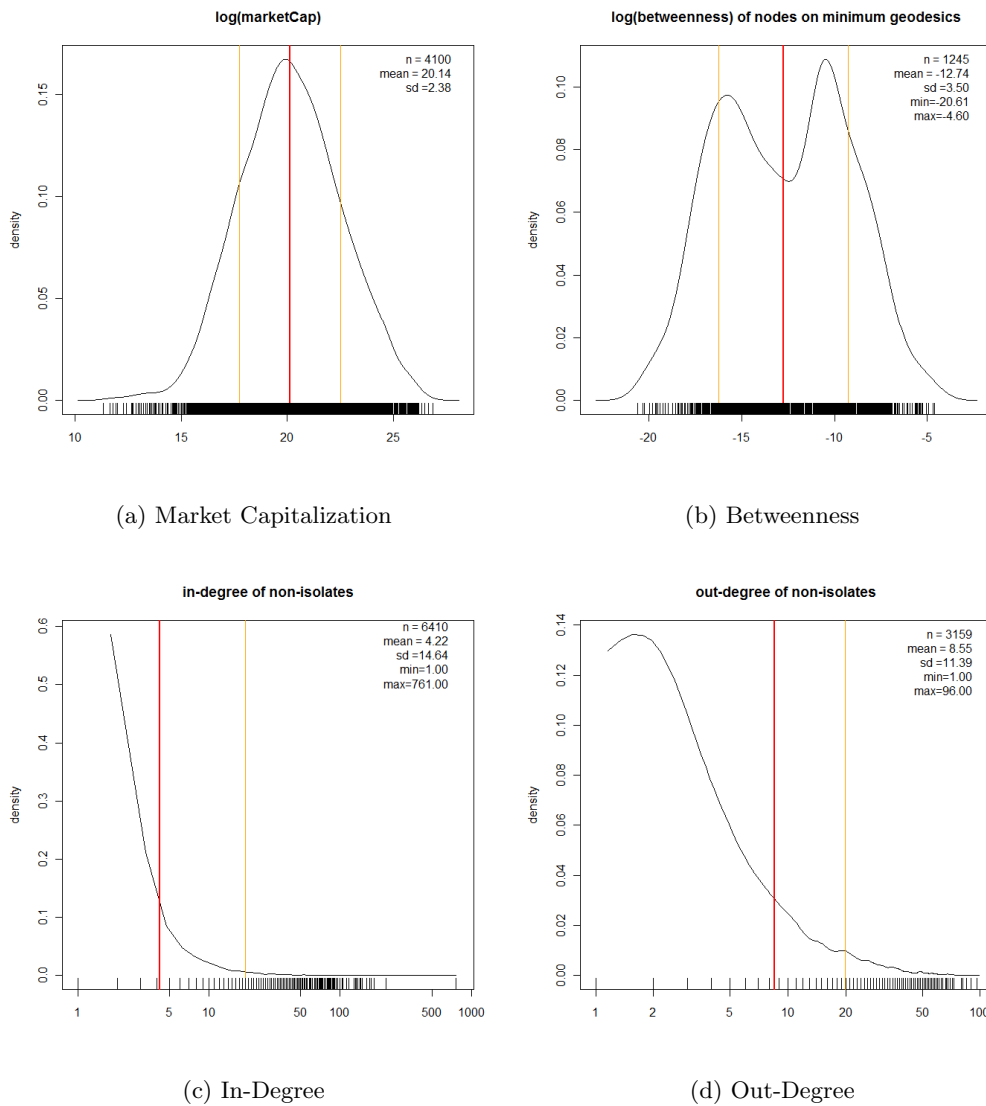


Figure 3.3: Node attribute distributions for Revere dataset.

as measured in the fourth quarter of 2008. Each plot shows a nonparametric estimate of the distribution of the variable in question, as well as the mean (in red), positive and negative standard deviations (in orange), and a rug indicating the actual observed values. They are shown here to contextualize data, and to illustrate a statistical distinction between the economic parameters and network measurements related to the data.

Market capitalization of those companies reporting an estimate appears to follow an approxi-

mately log-normal distribution (albeit with significant kurtosis), and is the only property visualized for which the mean can be interpreted as a reasonable expectation regarding the underlying data.

Betweenness is defined as in [114] as the proportion of shortest paths within the network on which a company lies. It is computed on an undirected (“or”-symmetrized) version of the network, and only for those nodes lying in the giant component (the measurement presupposes a connected graph). Since a large number of nodes are only on shortest paths between pairs involving themselves, I report only those nodes which are on additional paths. Betweenness is reported as a proportion of theoretical maximum path participation given the size of the resulting graph. I compared the goodness-of-fit of nonparametric estimates of the distribution based on betweenness and log-betweenness. The fit based on log-betweenness performed substantially better in cross-validation, suggesting that the data is more smoothly and symmetrically distributed in log-scale. The nonparameteric analysis of the resulting distribution is clearly bimodal, suggesting distinct “core” and “periphery” roles even among those nodes whose presence reduce the diameter of the giant component.

In-degree and out-degree distributions are displayed in logscale, although the plotted nonparametric density estimates were computed on absolute degrees. In each case, analysis was conducted only on nodes having degree of at least 1. In-degree - the count of companies that are customers of each supplier - appears to a first approximation to follow a power-law distribution as has been reported in numerous analyses of total degree (e.g. [69]). Out-degree, by contrast, has a modal value of 2, suggesting that those companies which are suppliers tend to have more than 1 customer of significance. The distribution appears to be fatter-tailed, with several plateaus perhaps corresponding to focal points in reporting (such as the tendency to report exactly 10 or 20 major customers).

Figure 3.4 shows the geographic distribution of company headquarters. The plot inadequately represents the extreme relative density of public corporations based in metropolitan areas, and especially the East coast. However, a few key features are clearly visible that are relevant to the geospatial analysis conducted in chapter 6. The distribution of company headquarters includes clusters (major cities), corridors (such as the Florida coasts or the strip between Washington, DC and New York), and density disparities such as the East and West of the Mississippi. This combination of factors makes it an attractive target for nonparametric and mixture based analysis rather than a single explicit model.

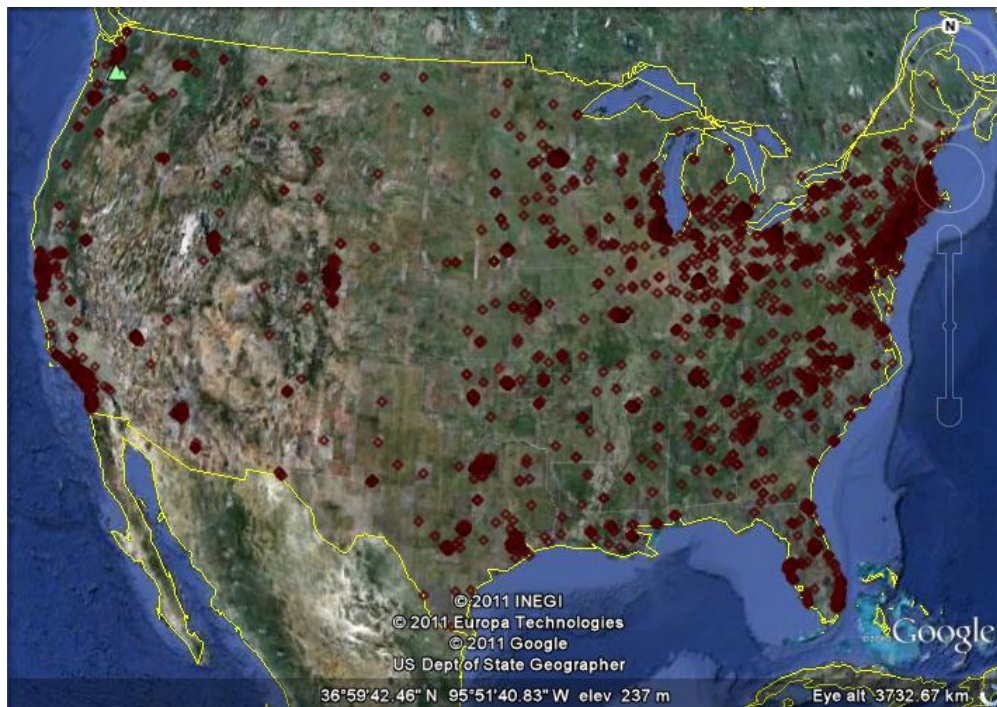


Figure 3.4: Listed addresses for public companies in Revere dataset.

### 3.3 Supplementary datasets

#### 3.3.1 Sampson’s Monastery (SAMPSON)

I chose Sampsons monastery dataset [104] as a testbed for the FOG framework because it is one of the datasets most widely discussed in social grouping literature. Sampson conducted a survey in which novice monks at a monastery ranked their compatriots according to four criteria: like/dislike, esteem, personal influence, and consistency with the creed of the monastic order. Sampson made strong arguments for several discrete social groups in the data based on direct anthropological observation. Events confirmed his observations when, during the study, novices of one group resigned or were expelled over religious differences. Sampsons surveys may be the dataset that comes closest to providing social data with a labeled ground truth for grouping research.

Sampsons monastery is discussed in greater depth in Sampsons original (1969) dissertation, and in the December 1988 issue of the journal *Social Networks*. I compare the groups discovered by FOG to Sampsons and those presented by Reitz in that issue in his introduction of a hierarchical clustering algorithm. Like that paper, I use Breiger et al.s [17] collation of Sampsons data: for each of the relations like, esteem, influence, and consistency, the top three positive selections by

	ROMUL 10	BONAVEN 5	AMBROSE 9	BERTH 6	PETER 4	LOUIS 11	VICTOR 8	WINF 12	JOHN 1	GREG 2	HUGH 14	BONI 15	MARK 7	ALBERT 16	AMAND 13	BASIL 3	ELIAS 17	SIMP 18
ROMUL	10	0	1	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0
BONAVEN	5	0	0	2	0	3	3	0	0	1	0	0	0	0	0	0	0	0
AMBROSE	9	0	1	0	0	2	0	2	1	2	1	0	0	0	0	0	0	0
BERTH	6	0	1	3	0	4	2	0	0	1	0	0	0	0	0	0	0	0
PETER	4	3	1	0	4	0	4	0	0	0	0	0	0	0	0	0	0	0
LOUIS	11	0	3	2	0	2	0	2	0	0	1	0	0	1	0	0	0	0
VICTOR	8	0	1	2	3	4	2	0	0	1	0	0	0	0	0	0	0	0
WINF	12	0	0	0	0	0	0	0	3	3	1	0	2	0	0	0	0	0
JOHN	1	0	1	0	0	0	0	1	4	0	1	2	0	1	0	0	1	1
GREG	2	0	1	0	0	0	0	1	3	4	0	0	0	3	0	0	0	0
HUGH	14	0	0	0	0	0	0	3	4	3	0	4	0	1	0	0	0	0
BONI	15	0	0	0	0	0	0	1	2	4	3	0	2	0	0	0	0	0
MARK	7	0	0	0	0	0	0	3	0	4	0	2	0	4	0	0	0	0
ALBERT	16	0	0	0	0	0	0	1	0	4	0	4	0	4	0	0	0	0
AMAND	13	0	4	0	0	0	3	0	0	0	0	0	3	0	0	0	0	1
BASIL	3	0	0	0	0	0	0	0	4	0	0	0	0	0	4	0	4	2
ELIAS	17	0	0	0	0	0	0	0	0	3	0	0	0	0	1	3	0	3
SIMP	18	0	0	0	0	0	0	0	1	4	0	0	0	0	0	3	4	0

Figure 3.5: Breiger (1975) collation of the Sampson monastery data

each individual at time three are recorded in a relation matrix. Negative selections are ignored, as negative relations are intransitive and thus cannot be positive evidence of an inherently transitive co-membership. These matrices are summed, yielding a single matrix summarizing the preferential data at that time period. The matrix in its entirety is shown below as Table 3.5. Because FOG analyzes link-based data, I then pre-process this matrix to generate links using the random tree technique described above.

### 3.3.2 Davis' Southern Women (DAVIS)

The southern women dataset due to Davis et al. [29] lists the attendance of 18 women and 14 parties. The parties in this network are precisely the type of linking observation which FOG is designed to analyze without pre-processing. As with the monastery dataset, there exists a labeling for groups based on direct observation rather than algorithmic analyses. Davis et al. used ethnographic analysis, including surveys, to distinguish not only between the two major cliques, but three tiers of centrality within them.

A wide variety of mathematical approaches have been used to reanalyze the data. Freeman [41] performed a comprehensive meta-analysis of 21 such studies, and I analyze our results in response to some of his conclusions. I have also accepted that paper's verdict on which of two conflicting figures in the original work was correct. I reproduce that figure as Table 3.6 for reference

Names of participants of group I		Code numbers and dates of social events reported in <i>Old City Herald</i>													
		(1) 6/27	(2) 3/2	(3) 4/12	(4) 9/26	(5) 2/25	(6) 5/19	(7) 3/15	(8) 9/16	(9) 4/8	(10) 6/10	(11) 3/23	(12) 4/7	(13) 11/21	(14) 8/3
1.	Mrs. Evelyn Jefferson	X	X	X	X	X	X	X	X	X					
2.	Miss Laura Mandeville	X	X	X		X	X	X	X						
3.	Miss Theresa Anderson		X	X	X	X	X	X	X	X					
4.	Miss Brenda Rogers	X		X	X	X	X	X	X						
5.	Miss Charlotte McDowd			X	X	X		X							
6.	Miss Frances Anderson			X		X		X							
7.	Miss Eleanor Nye			X		X	X	X							
8.	Miss Pearl Oglethorpe					X	X	X		X					
9.	Miss Ruth DeSand				X			X	X	X					
10.	Miss Verne Sanderson						X	X	X				X		
11.	Miss Myra Liddell							X	X	X			X		
12.	Miss Katherine Rogers							X	X	X			X		X
13.	Mrs. Sylvia Avondale						X	X	X	X			X	X	X
14.	Mrs. Nora Fayette					X		X	X	X	X		X	X	X
15.	Mrs. Helen Lloyd						X	X		X	X		X		
16.	Mrs. Dorothy Murchison							X	X				X		
17.	Mrs. Olivia Carleton								X		X				
18.	Mrs. Flora Price								X		X				

Figure 3.6: Davis' (1941) southern women party attendance





## Chapter 4

# Community Detection

### 4.1 Problem Definition and Background

Community detection is the sociological instantiation of the clustering problem. In a social context, groups often possess an entity beyond the descriptive and predictive criteria which clustering algorithms typically optimize. Individuals often name groups, describe their relationships to groups, and describe relationships groups have with each other. Since the earliest days of social network research, accurate detection of these group entities has been an attractive and elusive goal. Once identified, group structure can be used for high-level descriptions of complex networks, to support or contest theories about underlying processes influencing social interactions, and to detect strengths or vulnerabilities of social structures and individual positions in a variety of contexts. These goals have important applications in a wide range of fields, including anthropology, sociology, organization science, economics, management, and security and intelligence programs.

In social network settings, community detection has typically consisted of dividing nodes into discrete partitions indicating mutual association. However, common sense and empirical analysis [41] support the view that humans are capable of simultaneously filling many roles in many contexts, and that a strict partitioning may prevent detection of supported group entities in a graph. To better understand modular structure in networks, we must employ models which allow for multiple memberships and varied levels of membership.

In this chapter, I build off link analytic group detection methods due to Kubica *et al.* [65] and Battacharya and Getoor [10], which allow for relaxed partitioning by permitting individuals under certain conditions to participate in multiple groups. I refine the representation of group structure by permitting varying strengths of association from members to group entities, and present an

algorithm that generates such groupings from link data using a stochastic model of link emission from group entities and a maximum-likelihood clustering method.

To analyze the utility of the fuzzy overlapping group model, I make comparison to groupings by anthropological observations and prior algorithms. The results suggest that this approach is capable of identifying groups that are confirmed by existing quantitative methods as well as expert ethnographic analysis, while providing additional information about overlap between groups and individuals who play multiple roles. This additional information facilitates understanding emergent behavior in the groups.

As a more technical assessment of the fitting algorithms, I perform parameter recovery experiments on random data whose generative distribution matches the model. The fitting algorithms are evaluated and compared on the basis of computation time, accuracy in detecting the quantity of latent groups, and accuracy in detecting group composition.

#### 4.1.1 Defining 'Group'

A group is a set of entities which experience the same membership relation with respect to the same external entity, real or abstract. In the social sphere, this can take many forms. For example, a formal organization like a board of directors, an implicit organization like a circle of friends, a demographic quality such as hair color, or even the set of individuals uniquely affected by an external force, such as the victims of a flu epidemic.

Cohesive groups, which exhibit more frequent association within groups than between them, are sometimes contrasted with *structurally equivalent* groups [114], which are defined by matching relationships with other groups rather than with each other. Entities are grouped together as structurally similar if their interaction patterns are similar; that is, if they interact with the same other entities or classes of entities. The group they experience membership with in this case is called a structural role. An intuitive example of a structural role is the middle manager in a hierarchical organization, who interacts with both upper management and employees, but not necessarily other middle managers. Like cohesive groups, structural roles can be implicit and unnamed or encoded in formal relationships. Like cohesion, the concepts of structural similarity and roles have been operationalized in many ways, the most common discovery techniques including block modeling [39] and CONCOR [17].

The fuzzy, overlapping grouping (FOG) algorithms I develop in this chapter are designed to illuminate only cohesive groups, but some correlations exist between structurally similar and cohesive groups. Membership in a strongly cohesive group can constitute a structural role, as interaction

with other members dominates individuals interaction patterns. Roles whose members do not interact can in some cases nonetheless be detected as a cohesive group following a transformation in data. As I discuss in my analysis of Sampsons data, detecting overlapping cohesive groups permits detection of a type of structural role, the interstitial actor. Finally, many structural groups appear cohesive when we consider interactions mediated by common experience or co-occurrence in data. For example, members of boards of directors might occur together on the recipient list for formal memos and meeting announcements. Individuals afflicted by the same communicative disease might tend to be clustered in space and time in hospital records.

Measuring this definition of cohesion depends on being able to clearly measure both the presence and absence of links between entities a property inherent in social network data, but less obvious in link data, which I define and discuss in the next section. In link data, a stochastic model must fill the roll of defining what comprises a concentration of links. In the next section I describe several widely used algorithms to detect cohesive groups in both types of data.

#### 4.1.2 Variations and Detection of Cohesive Groups

Cohesion generally refers to a greater frequency of connections within groups than between them, but this umbrella supports many definitions. Most definitions would support that a set of disjoint cliques are clearly each cohesive groups, but more ambiguous data lead to questions such as: Are all individuals parts of groups? Can individuals be part of more than one group? If nested subgraphs contain increasing interaction density (such as in a core-periphery structure), are there several cohesive groups or one?

One major theme in formalizing cohesion has been an evolving series of graph theoretic group definitions, generally subgraphs satisfying internal connectivity requirement (e.g. cliques,  $k$ -clans,  $k$ -cores,  $k$ -plexes). These structures may overlap, leading to the possibility multiple community memberships. Palla et al. [100] give an iterative definition, related to  $k$ -cliques, designed specifically to examine overlapping communities. Because group membership under their technique is binary, all individuals in a community overlap have equivalent positions. One goal of the FOG algorithms is to reveal distinctions in these interstitial roles by detecting weights of membership.

An alternative graph theoretic approach due to Moody and White [87] emphasizes paths, defining cohesive communities as those supporting redundant communication threads. Interestingly, although the modularity heuristic of Girvan and Newman [78] is capable of evaluating any partitioning, the algorithm they give for approximately maximizing it takes a path-based perspective by iteratively removing of high-betweenness edges. Both techniques assign binary memberships,

and those using Newmans method lack capacity for overlapping or nested groups. Moody and Whites communities do not overlap at any given level (FOGs do), but their hierarchy provides nesting relationships and is in some ways more informative as it supports the pairwise query: at what level are two individuals grouped? Newmans algorithm has seen several extensions and applications [26] [94] [93] demonstrating its effectiveness on extremely large datasets which cannot be analyzed by other techniques (including FOG).

Another line of grouping research, block modeling, revolves around partitioning matrices such that subgroups have consistent relations. Block models are a natural setting for detection of structural equivalence [39], but have been extended to a variety of other settings including detection of cohesive groups [34]. Popular algorithms for block modeling include CONCOR [17] and FACTIONS search [16]. The stochastic model of Snijders and Nowicki [96], which presumes that likelihood of interaction is uniform within blocks corresponding to a discrete latent partitioning, is capable of finding both cohesive groups and intergroup relations.

Graph partitioning has received significant attention outside of sociological literature. The METIS [58] and spectral partitioning algorithms [51] are designed to find well connected subgraphs suitable for parallelization of graph algorithms, in which near-uniformity of partition size is a desirable quality. This is balanced with the presumption that the loss of link information resulting from edges destroyed by cuts between partitions will also degrade results. These methods differ significantly from FOG in their emphasis on partitions of equal size and on strict partitionings of the network.

### 4.1.3 Bipartite (Link) Data

Recently, Doreian *et al.* [36] have generalized block modeling for the analysis of 2-mode data, such as the relation of individuals and parties attended in Davis study described below. Such relations are represented as rectangular matrices, and are block modeled by providing a separate partitioning for rows and columns. This closely relates to H-FOGs method, which discretely partitions one mode (the events). Rather than partitioning the other mode (individuals), however, FOG optimizes and presents fuzzy groups induced by the first partitioning.

In this chapter, I often discuss 2-mode as described above, but refer to it as link data. This is to distinguish it from network data, and to emphasize FOGs perspective that we observe a sample of an infinite stream of links rather than the entirety of a finite matrix. I refer to our observed links as evidence, represented as an unordered set of links. Each link is an unordered set of entities in which each entity is assumed to have the same relation to an observation, such as “signed meeting roster”,

or “was observed in photograph”. Our set of links may carry redundant associations (i.e. two events with the same attendees) or simultaneous associations of more than two entities (one event with five attendees).

Data mining communities have produced several methods for extracting group entities from this type of data, including the GDA model/ $k$ -Groups algorithm [65] and Battacharya and Getoors [10] iterative deduplication method. These algorithms partition link data to infer groups which maximize the likelihood of observing the given data, according to a stochastic model. The fact that groups are built by partitioning links (not individuals) produces the advantage that, as with Palla *et al.*, individuals may belong to more than one group. The method I introduce in this chapter extends these methods by allowing varying levels of association from entities to groups. This relaxation is intended to allow group models to more tightly fit the data and to represent a wider variety of associative structures.

Another form of decomposition of bimodal data lies in factor analysis such as principle component analysis (PCA) and independent component analysis (ICA) [54]. Both methods are designed for dimensionality reduction in continuous vector spaces. PCA does so by identifying a given number of orathgonal vectors which capture the greatest degree of variance in the original high diensional set. ICA does so by identifying factors whose loadings are most independent (minimally correlated). The focus of both methods on continuous data and the usage of multiple factor loadings to explain each data point differentiate these algorithms from FOG, which operates on binary attendance data and presumes that a single group is responsible for each observation.

The existing technique with greatest similarity to the FOG framework is Latent Dirichlet Allocation (LDA) [14], a stochastic model for machine learning mixed memberships. Airoldi *et al.* [6] have adapted the model to examine single-mode network data, yielding novel clusterings in protein-protein interaction networks. The primary distinction between FOG and relational LDA models is that LDA allows a single observed link to be explained by a mixture of groups, whereas FOG assumes that a single social context is associated with a given observation, but hierarchically clusters such contexts to construct a restricted mixed-group structure.

## 4.2 Fuzzy, Overlapping Grouping

Grouping methodologies are often introduced as algorithms, although they encompass distinct models, measures, data translations, and validation schemes as well as the model-fitting algorithm itself. To minimize this confusion, I discuss FOG as a framework consisting of several components. The FOG generative model relates link interactions we observe to group entities, which are hidden.

The H-FOG algorithm is a simple approach to fitting groups of the type described in the model to data based on hierarchical clustering of links, with the advantage that various levels of group granularity can be explored following a single execution of the algorithm.  $k$ -FOG is an expectation-maximizing link clustering approach to the same task when a fixed number of groups is known *a priori*, and  $\alpha$ -FOG is a third algorithm that estimates the number of groups from data while also determining group composition. A separate link generation algorithm creates link data from social network data.

### 4.2.1 Link data from network data

Link analysis and network analysis have grown out of distinct communities, despite being frequently applied to the examination of the same interaction phenomena. In many ways, grouping research has become an intersection point in which practitioners of both fields are attempting to capitalize on the strengths of the other. Link analysis researchers approach group models as an opportunity to characterize structure and dependence in interaction data which is too often analyzed as though observations were independent.

Analysts who have traditionally used graph theoretic approaches to examine network data are incorporating statistical models and significance tests to improve their ability to reason about noisy data and support claims about the significance of structural characteristics in their networks. For frameworks such as FOG to see the widest use (and scrutiny), we must develop translation techniques that allow data in one format to be examined using algorithms for the other. These translations must account for disparate data qualities emphasized in the two branches of analysis.

Since small changes in network structure can have a large impact on the graph theoretic measures used in network analysis, translations from link data to network data are designed to reduce noise as much as possible. Many network datasets begin life as something more closely resembling link data. Lists of interactions or survey responses are “flattened” into a matrix of pairwise interactions using summation, cutoffs, or reciprocation criteria depending on the interaction being studied and the network type desired. Kubica *et al.* [64] have presented cGraph, an expectation maximization approach to detecting underlying networks based on such data.

The stochastic link analytic techniques I examined are intended to robustly handle noise given enough data. However, when our source data is limited to the information in an interaction matrix, we run the risk of amplifying any noise present when we generate additional links. We must also tackle the problem of reflecting the structural data contained in the network model in a way that link analytic algorithms can interpret.

The simplest approach would be to interpret each edge in the network data as a single link between two entities. Though it retains all of the original data, this naive method produces links that individually contain the minimal amount of structural information. Broader patterns such as paths and clusters can be revealed only by inspecting many links at once. This disadvantages greedy algorithms that examine individual links, such as H-FOG, because they have little basis on which to make their earliest (and most important) clustering decisions. For these algorithms, we must generate “richer” links that give more structural information while still only sampling the overall network.

I have adopted the “random tree” solution described by Kubica et al. [64], inverting its purpose to generate random interactions rather than extract graphs from observed interactions. Link data is constructed stochastically by iteratively adding to links entities which are randomly chosen from the neighbors of those already present. Fig. 4.1 illustrates this process, which is defined explicitly in algorithm 3. In the illustration, nodes A and B have been visited already, making the entire peripheral boundary of C, D, and E available as possible next additions. Note that C has a twice greater probability of selection than D or E, as there are two links proceeding to it from our already-visited structure. If our matrix were weighted, it would be the summed weights of those links rather than the quantity that determined relative likelihood. The intuition behind this process is to simulate chains of gossip or casual assemblies that would be directly measured as a source of 2-mode data if available. I use this algorithm on a weighted collation of Sampsons monastery data later in this chapter.<sup>1</sup>

Since algorithm 3 can operate on binary or weighted networks, with directed or symmetric edges, it is important to note how weights and directionality are interpreted. In the “viral process” metaphor for link generation, weights indicate relative probability of interaction. So this is most appropriate for weighted networks where weight indicates some kind of frequency statistic, rather than ranked qualitative assessments such as esteem. Directed links make the order in which nodes are added to the random tree significant, but this information is discarded as the links are considered to be unordered sets. However, if random trees are initiated from all nodes, directionality will manifest as an asymmetric conditional probability of appearing together. If node A has one edge, directed at B, than observation of A will always imply observation of B - but the reverse will not be true. It’s worth noting that the random tree algorithm could potentially be adapted to weighted

---

<sup>1</sup>A potential criticism of this method is that its stochasticity introduces uncertainty into results. In fact, since results will converge with a large sample, the user can define a preference between accuracy and speed by specifying a sample size. Reproducibility can be achieved by storing and reusing a random seed. Finally, random link production is consistent with FOGs modeling of uncertainty in all data, and is affirmed by both our own empirical results and the prior efforts below.

---

**Algorithm 3** Random Tree Sampling

---

**Require:**  $\mathbf{A} \in \mathbb{R}^{n \times n}$  {unimodal adjacency matrix, weighted or unweighted}**Require:**  $m \in \mathbb{Z}$  {the number of links to seed from each node}**Require:**  $c \in \mathbb{Z}$  {the number of entities included in each link}**Ensure:**  $\mathbf{L}$   $\{mn \times n$  binary link matrix with up to  $l$  nonzero memberships per row connected in  $bfA\}$  $\mathbf{L} \leftarrow \mathbf{0}^{kn \times n}$ **for**  $i = 1$  to  $n$  **do**  **for**  $j = 1$  to  $m$  **do**     $r \leftarrow m(i - 1) + j$      $l_{r,i} \leftarrow 1$     **for**  $k = 1$  to  $c - 1$  **do**       $\mathbf{p} \leftarrow \mathbf{1}_j \mathbf{A}$   $\{p_x$  is the summed adjacency from current link members to  $x\}$        $\forall_x p_x \leftarrow p_x I(l_x \neq 1)$  {do not repeat members}       $z \leftarrow \sum_x p_x$       **if**  $z = 0$  **then**         $j \leftarrow m$  {break if there are not enough adjacent members}      **else**         $y \sim \frac{\mathbf{p}}{z}$  {draw next link member from PMF given by normalized adjacency vector}         $l_{r,y} \leftarrow 1$       **end if**    **end for**  **end for****end for**

---

or unweighted multi-graphs, if a weighting scheme is adopted to determine relative significance of each edge type in the link generation process. This is effectively done by my use of Breiger's collated Sampson monastery data later in the chapter.

Prior work has examined relationships between networks and distributions of random processes on them. Kashima and Tsuboi [59] showed that random walks can be used as a kernel in classification of structural features of a graph. Random walks and trees in social networks have been used in simulations as analogs to real world processes, such as knowledge dissemination or the spread of a disease [24]. Page and Brin [18] note that eigenvalue centrality of each node in a network is proportionate to the fraction of an infinite length random walk it will occupy, which they have



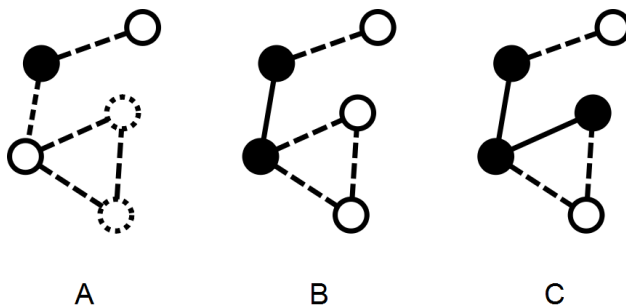


Figure 4.1: Three iterations of random tree generation. Solid nodes and the edges between them are included in the tree. Solid peripheral nodes are equiprobable candidates for inclusion in next iteration.

famously analogized to the search for information on the web.

In this chapter, I interpret groups found in random trees as sets of individuals who are likely to be exposed to the same experiences. Since we cannot directly calculate or represent the distribution of such groups, I sample them instead by computing a fixed number of groups of fixed size.

### 4.2.2 Stochastic model of evidence generation

Since we are trying to infer groups based on link evidence, I define our group membership relation as the tendency to be produced in observations associated with the group. We can alter the strength of the tendency to be included in observations without altering its fundamental character. I formulate the above mathematically as follows.

Consider a set of  $n$  entities organized into  $k$  groups. Entities are elementary objects whose presence or absence can be determined consistently within each of a set of  $m$  observations (called links), emitted by the groups. The links are described by a binary matrix  $\mathbf{L} \in [0, 1]^{m \times n}$ , and the groups that generated each one are enumerated in an *emission vector*  $\mathbf{e} \in [1 \dots k]^m$ . It is sometimes convenient to refer instead to the *emission matrix*  $\mathbf{E} \in [0, 1]^{m \times k}$  such that  $e_{i,j} = I(e_i = j)$ .

Groups emit with different frequencies, according to a frequency distribution  $\theta$  across groups such that  $\theta_i$ , for  $i \in 1 \dots k$ , is the probability a that a randomly selected link was emitted by the  $i$ th group ( $\sum_{i=1}^k \theta_i = 1$ ). Elsewhere in this paper I refer to  $\theta$  as the emission prior, since it represents our expectation that a piece of evidence will come from a specific group, prior to examining the members observed in the link. Each row of a membership matrix  $\mathbf{\Gamma} \in \mathbb{R}^{k \times n}$  is referred to as a membership vector  $\gamma_i$ , and its entities give a PMF over the  $n$  entities ( $\sum_{j=1}^n \gamma_{i,j} = 1$ ) such that  $\gamma_{ij}$  is

the probability that entity  $j$  will be observed in a link emitted by group  $i$ . The above relationships are enumerated in the equations below.

$$(4.1) \quad P(e_i = x) = \theta_x, \quad P(l_{i,j} | e_i = k) = \gamma_{l,j}$$

When considering the likelihood that a particular group would produce a specific link, we must consider not only the probability of observing the entities present in the link but also the probability of excluding those not present.

$$(4.2) \quad P(\mathbf{l}_i | e_i = j) = \prod_{x=1}^n l_{i,x} \gamma_{j,x} + (1 - l_{i,x})(1 - \gamma_{j,x})$$

The assumption that, in the emissions of a single group, members are emitted completely independently ( $l_{i,j} \perp l_{i,k}$ ) is important to maintaining that the membership relation differs only in intensity between entities, as a joint distribution would imply additional substructure. Similarly, I assume that links are generated independently given the groups and their emission priors ( $\mathbf{l}_i \perp \mathbf{l}_j | \mathbf{\Gamma}, \theta$ ), so that the only structure exists between the groups and the entities themselves, and in the relative frequency of emission of the groups. Combining these, we can derive a likelihood that an entire set of evidence would be produced given a grouping and an emission distribution vector.

$$(4.3) \quad P(\mathbf{L} | \mathbf{\Gamma}, \theta) \propto \prod_{i=1}^m \sum_{g=1}^k \theta_g \prod_{x=1}^n l_{i,x} \gamma_{j,x} + (1 - l_{i,x})(1 - \gamma_{j,x})$$

Performance and representation precision (probabilities involved can be extremely small) demand that the above likelihood function be calculated within a log-likelihood transformation. To enable this transformation, I add the restriction that  $\forall_{i,j} 0 < g_{i,j} < 1$  (in practice by including a weak Beta prior during optimization). This ensures that a group always has some nonzero probability of emitting its least related entity, or excluding from a link even its most significant member. Previous stochastic models of link generation have included an error term under which there is some small probability that a link will be emanated containing entities which do not cohabit any group. This was necessary to allow models to be fit to data without placing extreme penalties on groups which were forced to include outlying entities as equal members to more supported nodes. In FOG, a similar purpose is served by allowing weak memberships and assuming weak universal memberships.

$$(4.4) \quad \log P(\mathbf{L} \mid \mathbf{\Gamma}, \theta) = \log z + \sum_{i=1}^m \log \sum_{i=1}^k \theta_g \prod_{x=1}^n l_{i,x} \gamma_{j,x} + (1 - l_{i,x})(1 - \gamma_{j,x})$$

where  $z$  is a normalization constant not dependent on  $\mathbf{\Gamma}$  or  $\theta$ . With the relationship between groups and evidence described above, we can approach community detection via expectation maximization. However, the inner product in equation 4.4 makes the equation non-convex and intractable for gradient methods. In the following sections I give algorithms to instead approximate optimal parameter settings using local search and iterative expectation maximization.

### 4.2.3 The H-FOG algorithm

Since the FOG model holds a single group responsible for emanating each link, we can restrict our search by considering only groups which optimally represent some partitioning of the data. Given an emission vector  $\mathbf{e}$ , we call the set of links assigned to each group that group's *support*, and refer to that subset of the link data as  $\mathbf{L}_{\mathbf{e}=\mathbf{i}}$  (for group  $i$ ). The optimal membership value for each entity is equal to the proportion of the supporting links in which that entity occurs. However, to avoid overfitting and to prevent infinitely negative probabilities, we initialize each membership with a weak beta prior, resulting in the estimate below.

$$(4.5) \quad \hat{\gamma}_j(i; \alpha, \beta) = \frac{\alpha + \sum_{x=1}^m I(e_x = i) * l_{x,j}}{\alpha + \beta + \sum_{x=1}^m I(e_x = i)}$$

The parameters  $\alpha$  and  $\beta$  are the same for all memberships, and must be specified in the FOG algorithm. As the prior is not intended to be informative, it is sensible to set  $\alpha = \beta = \epsilon$ , where  $\epsilon$  is as small as can be used while retaining numerical stability in whichever algorithms are applied. To simplify notation, I omit them from most equations when utilizing  $\hat{\gamma}$ . An entire membership vector constructed this way is marked with a  $\hat{\gamma}(i)$ , and an entire membership matrix  $\hat{\mathbf{\Gamma}}(\mathbf{e})$ .

H-FOG builds groups of this sort by iteratively clustering link evidence in a way that ensures links with the greatest similarity are grouped together. For each pair of optimized group membership vectors  $\{\hat{\gamma}(i), \hat{\gamma}(j)\}$ , I consider a new group  $\hat{\gamma}(i, j)$  that would maximize probability of emitting the combined evidence supporting both groups. I then calculate the ratio indicating the relative increase in likelihood of the underlying links if they are considered the emissions of one merged group rather than two separate ones. Given a current link clustering  $\mathbf{e}$ , the H-FOG scoring function is as follows.

$$(4.6) \quad s_H(i, j) = \frac{(\sum_{x=1}^m I(e_x = i))(\sum_{x=1}^m I(e_x = j))P(\mathbf{L}_{\mathbf{e} \in \mathbf{i}, \mathbf{j}} | \hat{\gamma}(i, j))}{(\sum_{x=1}^k I(e_x = i))P(\mathbf{L}_{\mathbf{e} = \mathbf{i}} | \hat{\gamma}(i)) + (\sum_{x=1}^k I(e_x = j))P(\mathbf{L}_{\mathbf{e} = \mathbf{j}} | \hat{\gamma}(j))}$$

At initialization of the H-FOG algorithm, each observed link is assigned to its own group. On each iteration, the score function is computed for all pairs of groups, and the two groups with the highest merge compatibility are combined. Since the score function is independent of parameters for groups other than the two in question, if  $O(m^2)$  memory is available then a dramatic performance increase can be reached by memoizing scores and recomputing only those for the new, merged group on each iteration. I give the algorithm below as determining a set number of groups, but in practice we can store intermediate states to allow easy access to the hierarchy later.

**Claim 1** *The computational complexity of H-FOG is  $O(m^2 n \log m)$ .*

PROOF. If we choose a representation  $\mathbf{g}(i)$  for each  $\hat{\gamma}(i)$  such that  $g_j(i) = \sum_{x=1}^m I(e_x = i)l_{x,i}$  and  $\hat{g}_0(i) = \sum_{x=1}^m I(e_x = i)$ , such that  $\hat{\gamma}_j(i; \epsilon) = \frac{\epsilon + g_j(i)}{2 * \epsilon + g_0(i)}$ .  $\mathbf{g}(i, j)$  is trivially computable in  $O(n)$  as  $\mathbf{g}(i) + \mathbf{g}(j)$ , and the probability of a group emitting its own support can be computed in  $O(n)$  time using the following equation, allowing  $s_H$  to be computed for any previously computed groups in  $O(n)$  time as,

$$(4.7) \quad P(\mathbf{L}_{\mathbf{e} \in \mathbf{i}} | \hat{\mathbf{g}}(i)) = \prod_{j=1}^n \hat{\gamma}_j(i)^{g_j(i)} \hat{\gamma}_j(i)^{g_0(i) - g_j(i)}$$

If an ordered data structure with logarithmic insertion and deletion time, such as a heap, is used to maintain an ordered list of  $s_H$  scores, initialization step of H-FOG takes  $O(m^2 n \log m)$  time and each of the  $O(m)$  subsequent merge steps can be performed in  $O(m + \log m + n)$ , for a total bound of  $O(m^2 n \log m)$   $\square$

It is worth noting that the algorithm is embarassingly parallel, as merge score calculations could be trivially divided among a number of threads and reassembled between iterations.

The tree in Fig. 4.2, constructed from the southern women dataset, illustrates the hierarchical clustering of evidence. Each intermediate node corresponds to a group tuned to produce evidence of the types found in the leaves below. I define a horizon from this tree as a set of nodes such whose children span all of the evidence, for which none is the ancestor of another. A horizon, such as the circled nodes in Fig. 4.2, corresponds to an emission vector from groups which account

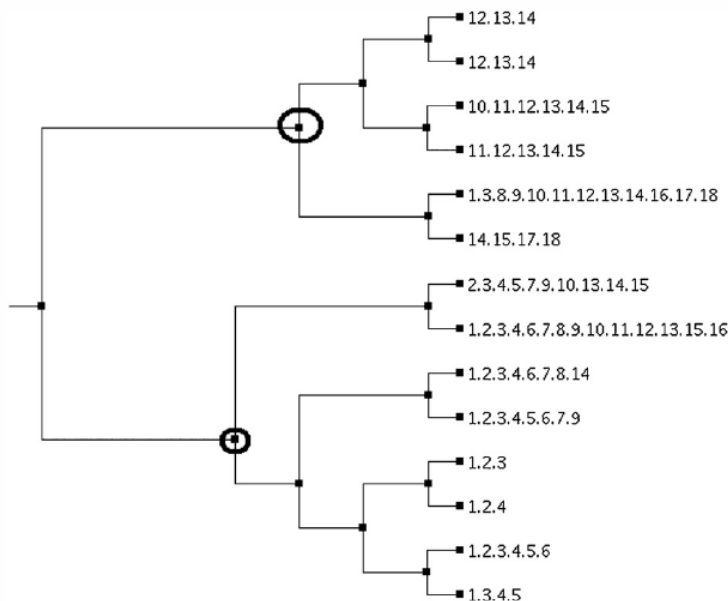


Figure 4.2: Clustering tree from the DGG dataset

collectively for all of the observed evidence. If we choose our horizon from the bottom level, groups are tuned to very specific profiles of evidence, so that they are expected to produce any of the few links below them with relatively high probability. As we move up the tree, membership rosters for groups become more complex and the distribution of links which they produce becomes more entropic, so that the probability of producing any particular link drops exponentially. At the same time, the average emission probability  $\bar{\theta}$  rises as we ascend the tree, since each group represents a greater proportion of the underlying evidence. Near the top, groups are overly general and fit the evidence underneath poorly, so that, even though  $\theta$ s are high, the total probability of producing the evidence set is quite low.

Unfortunately, reduced link likelihood  $P(\mathbf{L} \mid \hat{\mathbf{G}}(\mathbf{e}))$  generally outpaces increased emission probabilities over a climb of the tree, so that there is usually no optimal midpoint that would allow us to discern a most probable number of group entities. This conflict between the need for well-supported groups and ones that tightly fit the data must be resolved by a preference fitting the context of the analysis. As such, an operator must currently specify a number of groups,  $k$ , for which to search, effectively deciding on a tolerable tradeoff between a simple model with few groups and a model which most closely fits the evidence but may in fact be over-fit. Fortunately, due to its hierarchical nature, H-FOG needs be run only once to generate candidate groupings for each feasible  $k$ . An analyst can then explore different numbers of groups dynamically to determine subjectively which

is best supported.

#### 4.2.4 The $k$ -FOG algorithm

H-FOG’s clearly bounded running time and the fact that it creates a navigable hierarchy of groups make it attractive for data exploration in small datasets. However, the greediness of the algorithm can cause arbitrarily bad results in pathological cases, and the  $O(m^2n \log m)$  running time make it unattractive for large datasets.

The  $k$ -FOG algorithm, given below, is intended to detect a provided number of high-quality (though not guaranteed optimal) groups in much larger datasets in a reasonable amount of time. It is a FOG-specific application of the EM clustering algorithm discussed in section 2.2.

**Claim 2**  $k$ -FOG will halt within  $O(k^m)$  iterations.

PROOF. Each iteration which reassigns  $e_i$  for any  $i$  must strictly increase the overall likelihood of the data with respect to the current parameters. This means it cannot revisit any previous state of  $\mathbf{e}$ , allowing it to at most repeat for its maximum number of iterations.  $\square$

In practice,  $k$ -FOG tends to halt very quickly in a fixed point for the two parameter updates. Although the global maximum grouping is just such a fixed point, the one reached in practice depends unfortunately on the starting configuration.

#### 4.2.5 The $\alpha$ -FOG algorithm

Both H-FOG and  $k$ -FOG require the user to select the number of groups, either by exploration or as a starting parameter.  $\alpha$ -FOG infers the number of groups from data, but still depends on the user for an  $\alpha$  parameter which determines the tradeoff between *precision* in each group representing the distribution of links in its support and *summarization* in that each group is supported by numerous links.

When describing  $k$ -FOG, the need to update group parameters with each link reassignment brought us into the Bayesian territory of treating  $\mathbf{\Gamma}$  as a random variable.  $\alpha$ -FOG extends this to the  $\theta$  parameter by assigning it a Dirichlet process (DP) prior. For our purposes the DP is best described as an infinite dimensional version of the symmetric Dirichlet distribution, which is itself a distribution over discrete PDFs. The parameter  $\alpha$  of the symmetric Dirichlet specifies the expected concentration in the PDF. When  $\alpha = 1$ , all distributions are equally likely,  $\alpha > 1$  prefers

more even distributions, and  $\alpha < 1$  produces distributions whose probability mass is concentrated in a few values. In the Dirichlet process case,  $\alpha$  gains a convenient interpretation that  $\frac{\alpha}{m+\alpha}$  gives the probability that the  $(m + 1)$ st link “innovates”, *e.g.* belongs to a never-before seen group.

$$(4.8) \quad P(e_{m+1} = x \mid \mathbf{e}_{1\dots m}) = \frac{\alpha + \sum_{i=1}^m I(e_i = x)}{m + \alpha}$$

Although the equation above is written with respect to a particular ordering of our links, we are indifferent regarding the order and could calculate the same marginal probability for any single assignment conditioned on the others. Using it, we can reformulate the likelihood of a single link as follows.

$$(4.9) \quad P(l_i = x \mid \dots) = P(e_i = x \mid \mathbf{e}_{-i})P(\gamma_x \mid \mathbf{e})P(l_i \mid \gamma_{\mathbf{x}})$$

This likelihood becomes the objective in an expected conditional maximization (ECM) algorithm, where it increases as we optimize  $\mathbf{e}$  and  $\mathbf{\Gamma}$  in each successive iteration. There is no need to update  $\theta$  because it is marginalized out in the equation above; in implementation a representation is updated in the sense that counts for each group are maintained to make certain calculations below faster.

### 4.3 Example analysis: fuzzy groups in the monastery

Because Sampsons data consists of pairwise relations, I generated link data using the random-tree technique previously discussed. Ten trees were initiated at each node, each expanding to contain three nodes, and the set was clustered using the H-FOG algorithm. Results are shown as a two mode (agent  $\rightarrow$  group) network in Fig. 4. Line thickness indicates the degree of membership, normalized by group as it is not necessarily appropriate to compare association levels between groups. This is because our link generation method required exactly three individuals in each observation, artificially deflating the average frequencies of emission in large groups and inflating it in small ones. Nodes have been manually laid out to elucidate membership categories I discuss.

Sampson identified novice 2, Gregory, as the most significant leader of the young Turks, the liberal newcomers who would be expelled or resign in the coming drama. The members of that group are collected exactly as those affiliated with group A. Gregory’s position, as both the most affiliated to the Turks and the only novice with connections to all three groups, suggests a high

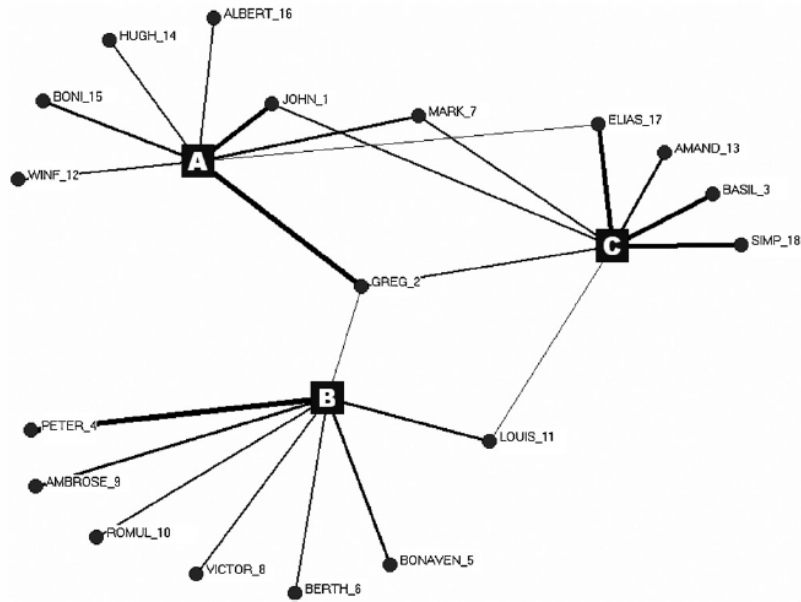


Figure 4.3: Fuzzy groups in Sampson’s monastery. Group A corresponds to the “young Turks”, group B to the “loyal opposition”, and group C to the “outcasts”.

degree of centrality both within the young Turks and in the network as a whole. This type of border-spanning centrality has been linked to iconoclasm, power, and stress, painting a vivid picture of factors which may have contributed to Gregory’s exit. The official reasons given for his expulsion were excessive independence and arrogance. Could he have been singled out as more dangerous precisely because he had captured the attention and esteem of individuals outside the clique? Rank of affiliation to group A turns out to be a good predictor for the order in which the young Turks left the abbey: the second individual most affiliated with the Turks is Sampson’s second identified leader, John (novice 1), who resigned shortly after Gregory’s exit, trailed by Mark (7) and the other novices.

Sampson’s third and lowest-ranking leader, Winfred (12), does not buck this trend. Although his relatively low association to group A belies his eventual identification as a leader of the Turks, it accurately reflects his placement as the last one to leave the Abbey. Winfred’s undistinguished position in our plot illustrates some biases of this analytic method, as well as some peculiarities of his situation. Winfred identified strongly enough with his group that he was completely embedded: all of his incoming and outgoing connections in the survey data lie within the Turks. The result is that, although the random trees in which he appears are exclusively tied to group A, he simply does not participate in nearly as many total evidence pieces as high-betweenness boundary spanners



like Gregory or John. As this shows, our evidence-generation technique could rightly be said to put a premium on individuals with high betweenness. However, this is defensible when paired with interpretation placing it in a social context. Recall that we generated random trees in order to model iterative interaction processes within the graph, such as the spread of a rumor or the slow accumulation of individuals to a casual gathering. It is easy to imagine an individual with more diverse ties, such as Gregory, being drawn into a wider variety of gatherings. By being a prolific interactor, Gregory may well have defined the Turks to the rest of the community, without necessarily intending to or even identifying exclusively with them.

Evidence supports the distinction between Gregory's celebrity and Winfred's poster child stances. In Sampson's study, Winfred's leadership was either absent or unobserved in the presence of the two higher-profile leaders, and became clear only after their exit. Winfred's embeddedness seems to reduce his significance at the time of our analysis, but as the split widened between the Turks and the opposition, making positions like Gregory and John's untenable, Winfred's exclusive loyalty became the crucial element of his ingroup leadership.

The membership and leadership of the loyal opposition party are similarly gathered around group B in our plot. Peter (4) and Bonaven (5), who were identified by Sampson as the leaders of the opposition, show the highest affinity for the group. Members described as less attached show less affinity, and one such novice shows a split allegiance to the outcast group. The absence of any links, save Gregory, between the opposition and the Turks serves to reflect the conflict between the two groups. By contrast, the outcasts in group C have several members associated with other groups as well. These cases show that fuzzy memberships can help elucidate not only the complexity of an individual's allegiances, but also the character of a group as exclusive or inclusive to interstitial members.

Sampson originally identified a fourth group, but I restricted our analysis to three clusters because the last was not a cohesive group fitting our definition. Sampson does not describe the waverers as a set of individuals allied or interacting with one another, but as being in similar positions of doubt between the two major groups, more akin to our interstitial roles. Additionally, previous analyses have questioned the distinction between the waverers and the loyal opposition. Our own analysis places two of them, Romul (10) and Victor (8), as weak members of the loyal opposition. From a purely structural perspective they are tied more to the loyal opposition; whatever their mental allegiance. Armand (13) is categorized as an outcast, owing less to his statements of affinity for those individuals than from Basil's (3) and Elias' (17) connections to him.

Our classification of Armand as an outcast is in line with the discrete partitioning provided by Doreian and Mrvar [36], who demonstrate that there was increasing evidence over time that

this foursome was a genuine group. Doreian and Mrvar used a block modeling approach optimizing structural balance, a measure of cohesion incorporating both positive and negative relations. Interestingly, their partitioning is perfectly correlated with the groups to which each individual is assigned maximal membership by FOG. I take these convergent results from different methodologies as encouraging validation in a setting for there is no known ground truth. The Doreian and Mrvar study also includes a temporal analysis suggesting that in the final period of Sampsons observation, two members of the Young Turks, Gregory and Mark, gave responses that fit better within the Outcasts partition. Both of these individuals are marked as interstitial in the FOG results, although Gregory's departure is somewhat surprising considering his very strong alignment with the Young Turks and weak connection to the Outcasts.

### 4.3.1 Example analysis: fuzzy groups among the southern women

Analyses of the DGG data, including the original, have generally partitioned the women into two cliques<sup>2</sup> that intersect on a few individuals or events. I use a spectrographic<sup>3</sup> visualization scheme in Fig. 4.4 to present the results of a 2-clustering of the southern women in greater detail than would have been readable in the Sampson analysis. Bars of each color indicate each womans affiliation with two groups derived from 8 and 6 of the party rosters respectively. Individuals are sorted along the X-axis according to the difference in their membership levels, which maximizes the visual distance between the cliques. I have also included a 2-mode network visualization for comparison to the one I presented for the Sampson data.

The results of our algorithmic approach correspond strongly to the intuitive conclusions of Davis *et al.* In group A, the core and primary periphery are reproduced precisely as plateaus in the membership levels. Someone attempting to fit our analysis to their mode might draw slightly different tiers for the group B, but the rough ordering of individual affiliations is the same. For both groups, the most peripheral members are seen in the center of our chart, with low levels of affiliation in both groups. Some of these members have been shown to be interstitial; for example Davis *et al.* report that Ruth (9) was claimed by both cliques in interviews with members. Others, such as Pearl (8) and Verne (10) were only claimed by members of the cliques to which our chart shows greater affiliation.

There are many mathematical studies of the DGG data to which the H-FOG clustering correlates. I will omit a pairwise comparison, as many of the results are significantly similar to Davis et

---

<sup>2</sup>I use *clique* here to maintain consistency with prior work, not to indicate a graph theoretic structure

<sup>3</sup>So named after similarity to overlaid graphs of element density used to differentiate substances in mass spectrometry.

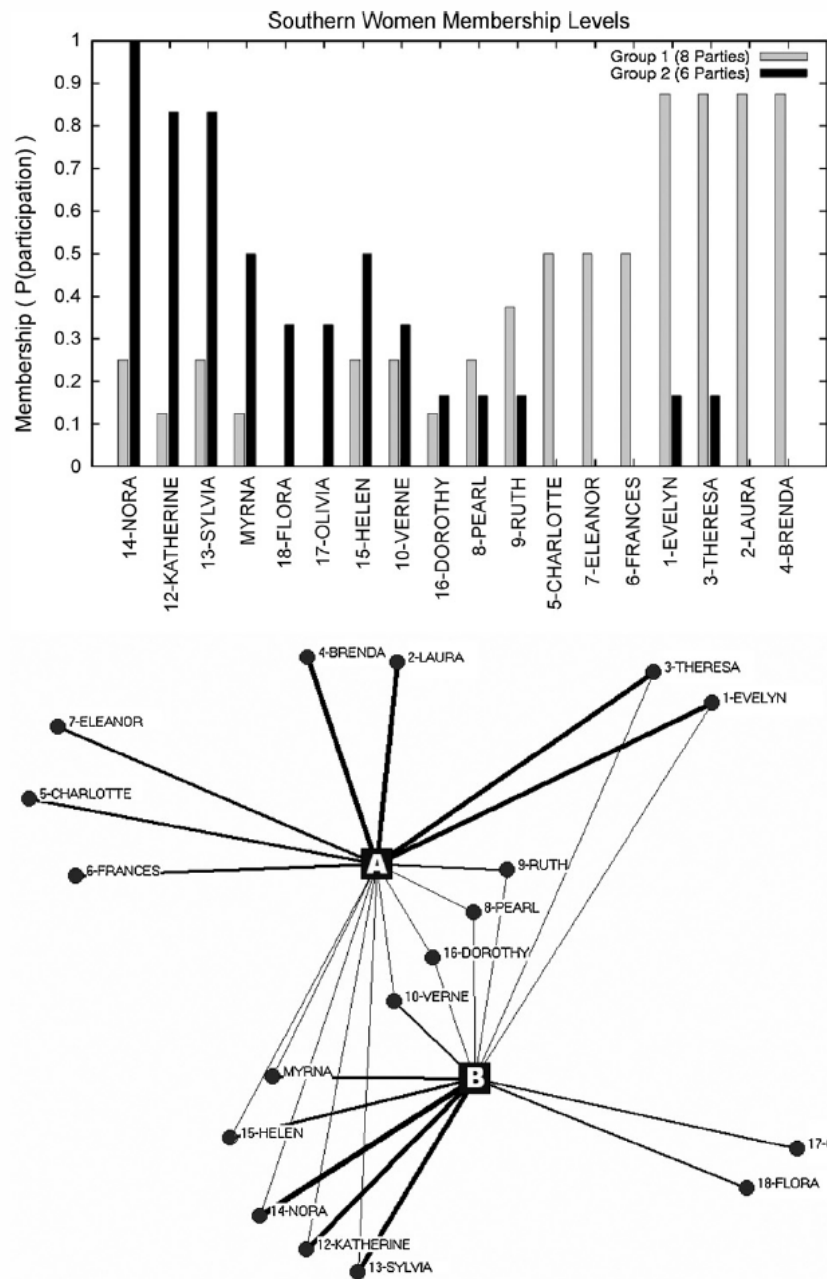


Figure 4.4: H-FOG 2-clustering of the DGG dataset, spectrographic (top) and network (bottom) representations.

al.s intuitive analysis described above, and a comprehensive metaanalysis has already been accomplished by Freeman. Instead, we focus on FOGs contribution to one prong of that analysis: the core-periphery structure of the two cliques.

Davis *et al.* describe as core the individuals that are seldom excluded from their cliques functions. I see that the most affiliated individuals in both groups demonstrate a propensity to appear with the other group as well. This supports the argument I proposed with the Sampson data<sup>4</sup>, that leaders of a group may either arise out of greater participation with other groups than do the less active members, such as those in DGG's primary and secondary members, or else experience more pressure to do so.

In his meta-analysis, Freeman treated core-periphery as an ordering of individuals for each group, without specifying that centrality in one group promoted distance from the other (although that was a side effect of many techniques compared). FOG results certainly fit that mode, but the juxtaposition of affiliations given above lends itself to an additional breakdown of several interactions. We can separate individuals into several modes of interaction. We have central leaders, such as the novices John or Gregory or the Southern women Nora and Katherine. There are embedded leaders such as Winfred, Laura, or Brenna. There is a loyal second tier in each of the groups I have analyzed, and finally a set of truly interstitial individuals who participate at low levels in both groups.

From our observations of these roles in Sampson data, we might issue the prediction that a thoroughly embedded member, such as Brenda or Flora, would flourish if there were a falling out between the two groups. On the other hand, if good relations continued between the groups, our profile of an emergent leader might better fit individuals such as Ruth or Helen: those with strong ties to one group, but some degree of participation with the other. Davis *et al.* do not examine conflict between these cliques and describe no events that would be telling regarding our first hypothesis. However, they completed a larger study of many cliques, in which they used interstitial members to examine relations between social classes associated with each clique. They describe a class of on the way up individuals, who participate in events outside their clique in order to socialize with those above them in social class.

The interpretation of interstitial members as a separate class is supported by Doreian *et al.* [34], wherein an error-averse block-model partitioning of events revealed that Pearl and Dorothy attended only events attended by members of both groups. For the block modeling approach, which considers extra-group connections when assigning groups, this was sufficient evidence to place them in a

---

<sup>4</sup>Since the DGG analysis is based on direct observations rather than synthetic observations from random trees, there is not the same concern about overemphasizing centrality that existed with the novices.

distinct group. FOG, which permits overlap but optimizes only for intra-group cohesion, places them instead in both groups.

By analyzing interstitial members, we can use FOG to capture some of the same structural insights provided by block modeling. However, some structural subtleties would not be captured by FOG. For example, we would be unable to distinguish between individuals like Pearl and Dorothy that attended only mixed events, and individuals who attended no mixed events but attended some from each group.

## 4.4 Parameter recovery experiments

As with all unsupervised inference tasks, the fact that underlying communities are never directly observed means that algorithms for detecting them must be evaluated either in relation to a secondary objective<sup>5</sup> or using artificial data in which the generating parameters are known. This section describes the latter. I generate group parameters from several underlying distributions, and for each set of parameters generate artificial link data. I apply each of the algorithms above to the resulting links, and evaluate them based on how quickly they can achieve what level of accuracy in recovering the originating distribution.

### 4.4.1 Generation of parameters and data

In section 4.2.5 I defined a prior  $\theta \sim \text{Dirichlet}(\{\alpha\}^k)$ , which can be directly sampled to generate artificial emission probabilities. An actual emission vector of any size is then drawn as independent samples from the resulting PMF.

The inclusion probability prior  $\gamma_i \sim \text{Beta}(\epsilon, \epsilon)$  is less suitable for generating the group matrix, because independent inclusion probabilities will result in diffuse, non-cohesive groups in which all entities have similar observation probabilities regardless of group. To generate cohesive groups, I first draw a group vector  $\mathbf{g} \sim \text{Dirichlet}(\{\alpha_\gamma\}^k)$  from a second Dirichlet prior, whose parameter  $\alpha_\gamma$  determines expected cohesion. The result is a normalized vector such that drawing  $P(l_i | e_i = j) = g_{j,i}$  would result in expectation in one entity per link. To generate links with an expected count of entities  $q$ , we use instead the probability that at least one of  $q$  draws from  $g$ , such that  $\gamma_{i,j} = 1 - (1 - g_{i,j})^q$ .

The full algorithm for generation of a random link matrix is given below.

---

<sup>5</sup>Examples of secondary objectives include prediction of events or replication of expert analysis; the qualitative analysis in section 4.3 is a bit of each.

#### 4.4.2 Evaluation of fit

Parameter values are the ultimate outputs of interest for each of our fitting values; however, they are difficult to compare directly because of the many symmetries possible in the parameterization. For example, any permutation on the group orderings amount to the exact same groups and will generate links with the same probability. Since FOG parameters ultimately correspond to a distribution over links, we can measure their similarity using the Kulback-Leibler divergence,

$$(4.10) \quad D_{KL}(\theta, \mathbf{\Gamma} \parallel \hat{\theta}, \hat{\mathbf{\Gamma}}) = \sum_{\mathbf{l} \in [0,1]^n} \log P(\mathbf{l} \mid \theta, \mathbf{\Gamma}) \frac{\log P(\mathbf{l} \mid \theta, \mathbf{\Gamma})}{P(\mathbf{l} \mid \hat{\theta}, \hat{\mathbf{\Gamma}})}$$

Even if equation 4.11 were efficiently computable (it involves an intractable summation over every possible link), it would be a noisy diagnostic of the performance of algorithms that train based on a very small sample of the space. Recent work on estimating intractable KL divergences has shown that monte-carlo sampling can be used effectively, especially in conjunction with importance sampling [52]. Based on this technique, the faster and more useful estimate below re-uses the random links on which groups were trained to compute a monte carlo estimation of  $D_{KL}$ .

$$(4.11) \quad \hat{D}_{KL}(\theta, \mathbf{\Gamma} \parallel \hat{\theta}, \hat{\mathbf{\Gamma}}) = z \sum_{i=1}^m P(\mathbf{l}_i \mid \theta, \mathbf{\Gamma}) \frac{\log z P(\mathbf{l}_i \mid \theta, \mathbf{\Gamma})}{\log z' P(\mathbf{l}_i \mid \hat{\theta}, \hat{\mathbf{\Gamma}})}$$

$$(4.12) \quad \text{where} \quad z = \left( \sum_{i=1}^m P(\mathbf{l}_i \mid \theta, \mathbf{\Gamma}) \right)^{-1}$$

$$(4.13) \quad z' = \left( \sum_{i=1}^m P(\mathbf{l}_i \mid \hat{\theta}, \hat{\mathbf{\Gamma}}) \right)^{-1}$$

A second challenge in creating a stable statistic for comparing goodness-of-fit is the fact that not all generator distributions, and not all samples from the same distribution, are equally learnable. Highly entropic distributions and samples that, by chance, poorly represent the probability of mass of the distribution will reduce performance for all parameter estimation algorithms. This makes it hard to compare performance across repeated experiments, since variations in performance may be amplified by problem difficulty. I control for this in two ways.

First, I estimate KL divergence not from the true generating distribution, but from new parameters fit directly to the emitted data with knowledge of the emission vector:  $\hat{D}_{KL}(\bar{\theta}, \bar{\mathbf{\Gamma}} \parallel \hat{\theta}, \hat{\mathbf{\Gamma}})$ , where  $\bar{\theta} = \frac{\sum_{j=1}^m I(e_j=i)}{m}$  and  $\bar{\mathbf{\Gamma}} = \hat{\mathbf{\Gamma}}(\mathbf{L} \mid \mathbf{e})$ . Since each of the algorithms is constrained to producing

parameters which are optimal given a latent emission vector, this distribution serves as a baseline “best fit possible given the data”, adjusting for some of the variance in the data.

Since this does not fully account for the varying difficulty in learning samples where the groups themselves are more entropic or overlapping, as a final step I normalize divergence scores by the best divergence score achieved by any of the algorithms. This gives each algorithm’s performance as a multiple of the winner’s.

### 4.4.3 Convergence profiles

To analyze convergence properties of the 4 FOG algorithms, I computed 100 FOG distributions using the method described in section 4.4.1. Each distribution involved  $k = 4$  groups of  $n = 20$  entities, and from each one  $m = 100$  links were sampled.  $h$ -FOG,  $k$ -FOG soft and hard variants, and  $\alpha$ -FOG were each used to reconstruct the distribution from samples. On each iteration of each algorithm, I recorded the computation time thus far and the normalized estimated KL divergence from the original distribution as described in section 4.4.2. The tests were performed in single threaded mode on an Intel Core i7 CPU running at 2.67 GHz, with all algorithms implemented in the R interpreted language.

The results of this analysis can be seen in figure 4.5, with the runs for each algorithm plotted in light grey. Overlaid on each graph is a kernel regression of normalized estimated KL-divergence with time, providing a rough guide to the convergence profile of the respective algorithm. It’s important to note that each of the FOG algorithms contain opportunities for system-specific optimizations and parallelization, as well as retrieval of anytime results, making a direct comparison of physical runtimes to convergence on the same machine is less useful in describing performance tradeoffs. Nevertheless, there are some general convergence features evident.

$h$ -FOG is both the slowest algorithm and achieves the worst parameter estimates overall (final fit performance is analyzed further in section 4.6). However, its deterministic nature means that running times are consistent, and visual inspection suggests that most runs converge nearly monotonically toward the source distribution as groups are merged. This suggests that our sample size is large enough to contain information to guide the algorithm toward the source distribution, and that our greedy agglomeration criterion extracts this information.

Exceptions to this generalization occur at very low numbers of groups, where some run graphs show large increases in KL-divergence in the last few iterations. Since  $h$ -FOG iterations correspond inversely with the number of inferred groups, the graphs show that the fit  $h$ -FOG achieves with a number of groups greater than the source distribution is often competitive with the other fitting

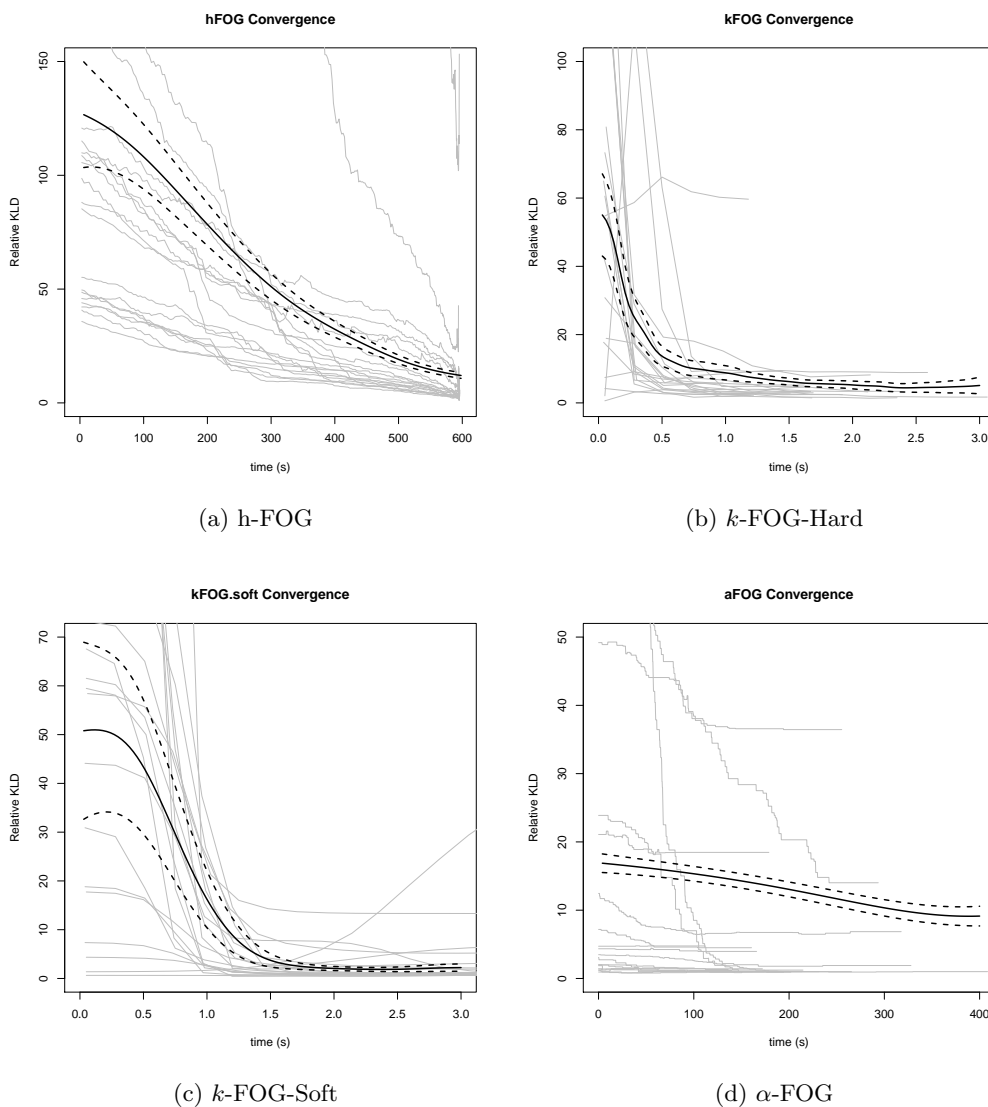


Figure 4.5: Convergence profiles for parameter recovery experiments.

algorithms, but the grouping at  $k = 4$  has twice the divergence or more. It may be that the final few merges in the algorithm effectively “pay the price” for the worst decisions made greedily in early iterations. As a practical matter, this suggests that, if the link distribution (rather than the groups themselves) is the object of interest and h-FOG must be used, it may be preferable for an analyst to specify a larger number of groups than they expect are actually present in the data.

The hard  $k$ -FOG algorithm shows the fastest convergence out of the set, usually reaching a



factor of 10 of the optimal convergence for any algorithm within one second. However figure 4.5b shows some degree of sensitivity to initial starting conditions and the potential for local maxima, as some plots converge early and with poor performance. By contrast, the soft variance of  $k$ -FOG shows only two test cases in which it was a factor of 10 from the optimum divergence, and only 2 more greater than a factor of 5. An interesting feature of the soft  $k$ -FOG convergence is the slow initial convergence, followed by a rapid period usually timed between the first and third half-second of computation. Both  $k$ -FOG algorithms tend not to detect convergence for a significant amount of time after nearing their optimal solutions. This is actually more pronounced than shown in figure 4.5 as the domain axes were truncated to show the most interesting part of the convergence curve; it is reported further in section 4.6. However rapid early convergence means that anytime variants of the algorithms could be useful.

Figure 4.5d shows that while  $\alpha$ -FOG often reached solutions competitive with those produced by the  $k$ -FOG algorithms, and is the most consistently monotonic in its convergence. However, it often took more than 100 times as much computation time, and frequently terminates at far from-optimal solutions. Long flat regions in the convergence paths, especially those with poor performance, suggest that many iterations are being spent by the algorithm without improving the solution. In the following section, I introduce an improvement leading to more consistent performance.

## 4.5 Improving $\alpha$ -FOG performance with residual priority updates

Although EM algorithms are traditionally defined in terms of synchronous expectation and maximization steps, Meng *et al.* [83] point out that for some problems other schedules interleaving partial expectation and maximization updates might be more tractable. This is the case with  $\alpha$ -FOG, where bookkeeping regarding new or retired groups is most convenient when only one link changes groups between updates.

The opportunity to define an arbitrary update schedule begs the question: which is best? And, is a single schedule appropriate for the duration of the algorithm? The slowdown in convergence rates on the EM-based algorithms in the prior section suggests that, especially as the algorithms progress, significant computation is spent on consideration of links that are already in their optimal configuration. Is it possible to spend more of our time adjusting links that are sub-optimally placed?

A similar problem was approached by Elidan *et al.* in [38] regarding the schedule on which messages are updated during belief propagation. I extend their work and explore the relationship

between EM and belief propagation further in the following chapter. In this section, I adapt their technique of prioritization by residual for FOG clustering.

The core idea in residual belief propagation is to compute all potential parameter updates and insert them into a priority queue according to a heuristic estimating their impact on the fitness of the parameter estimates. Since the likelihood of our fit is the product of link likelihoods, I take the change in log likelihood for each link as its residual, so that the maximum improvement is executed first.

After each update, other updates whose optimality or score might be affected by the just-executed one must be reconsidered, so performance is best when the distribution is factored such a way that each update affects a small subset of the others. In the context of a probabilistic graphical model, the graphical structure dictates this factorization. In our context of hard clustering, we can achieve this factorization by alternating assignment of a single link to an originating cluster with the optimization of the source and destination clusters' parameters. Doing so requires us to reconsider the likelihood of assignments of each link to just those two.

The  $\alpha$ -FOG-Residual algorithm is given below. For consistency I use a matrix based annotation similar to that for h-FOG to keep track of likelihood improvements for each link. In practice, both algorithms can be implemented more efficiently using a heap-based priority queue. In  $\alpha$ -FOG's case, queues would be used in two places: first to maintain the optimal assignment for each link, and second for each link the residual of its optimal assignment.

Maintenance of the priority queues for each variable makes each iteration of  $\alpha$ -FOG-Residual  $O(m \log k + mn)$ , more expensive than  $\alpha$ -FOG's  $O(mk)$ . Does the prioritization of more important updates make up for this? Figure 4.6 shows that it does, with residuals competitive with those of the  $k$ -FOG algorithms reached with greater consistency and within one minute, rather than the original  $\alpha$ -FOG's seven.

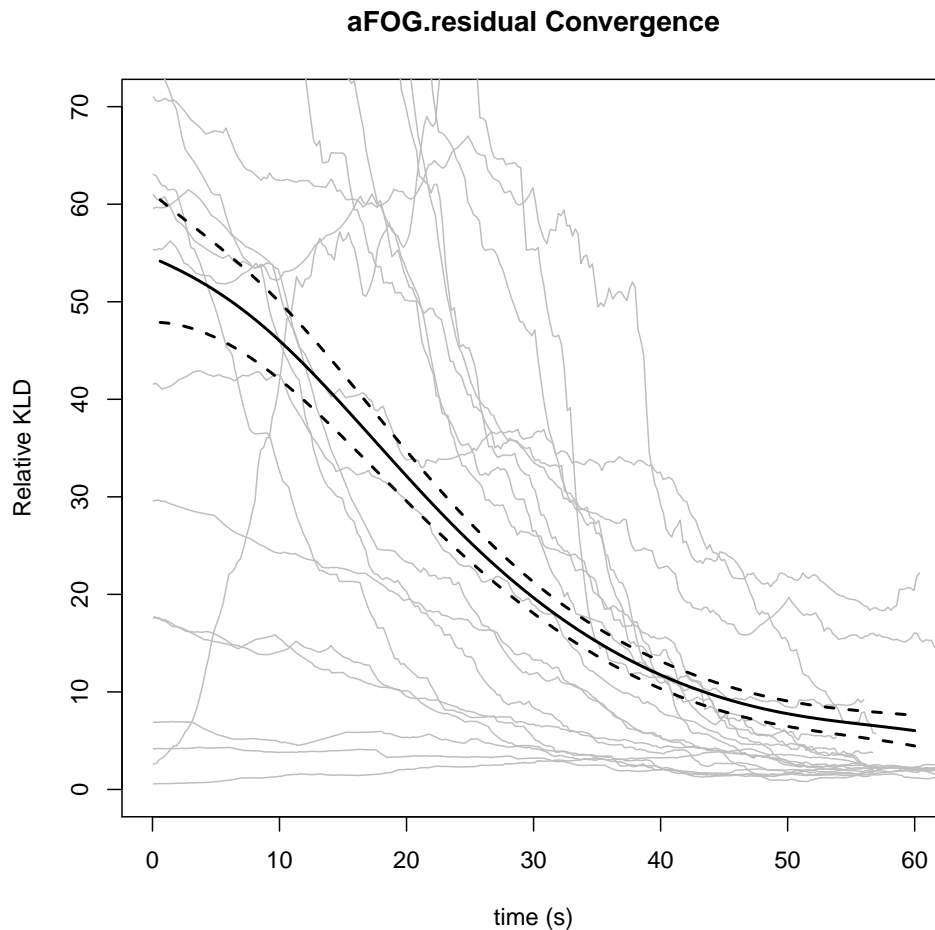


Figure 4.6: Convergence profile for  $\alpha$ -FOG-Residual.

## 4.6 Performance comparison summary

Table 4.1 summarizes performance data for each of the algorithms upon convergence on the 100 test runs. Time and KLD ratios are captured as in the previous sections, and the table includes counts

of *notional iterations* designed to count the number of times the outermost loop with inconstant frequency is called. Notional iterations are defined as follows.

- For H-FOG, I count each merge operation as  $k'$  iterations, where  $k'$  is the number of groups remaining.
- $k$ -FOG algorithms count complete cycles of both E and M steps and annotate each such pair as  $m$  notional iterations.
- In  $\alpha$ -FOG, each assignment of a link to a group counts as 1 notional iteration.

The summary statistics on h-FOG suggest little to recommend it relative to the other algorithms, as it takes longer and produces worse groupings with greater variance than all others besides  $\alpha$ -FOG. The intended motivation of producing groupings for all potential  $k$  might be better served by executing  $k$ -FOG-Hard for all the values within a certain range. In spite of these limitations, H-FOG might still be of interest in situations where hierarchical relationships between groups is itself hypothesized, as a way of exploring that structure in particular.

Comparing the  $k$ -FOG algorithms, hard and soft variants offer a pronounced tradeoff between speed and quality of fit. While the standard deviation of KLD ratios for both algorithms is high, a t-test statistic places the true mean for  $k$ -FOG-Soft higher than that for  $k$ -FOG-Hard with a p-value of 0.96%, while  $k$ -FOG-Hard is at least 5 times faster on average with confidence beyond machine precision.  $k$ -FOG-Hard may be the only algorithm applicable to large scale data or real-time applications requiring low latency. The result is unsurprising, given that  $k$ -FOG-Hard can be viewed as a constrained version of  $k$ -FOG-Soft in which emission distributions are constrained to point mass probabilities. The coarser configuration space can be searched more quickly, but a precise solution is impossible. In future work, it might be interesting to use hard clustering as a hot-starting technique, to be improved on by soft clustering.

One challenge in comparing convergence profiles between the algorithms is that none of the algorithms except H-FOG have a guaranteed time until convergence. Another is that qualities of data other than size might impact time until convergence. For example, data with more intermixing between groups might be relatively more challenging for  $\alpha$ -FOG variants, because determining the exact number of groups is more difficult. It is advisable for practitioners to try both  $k$ -FOG and  $\alpha$ -FOG variants on subsets of their own data to consider its properties, if running both on the whole dataset in infeasible.

$\alpha$ -FOG in its original form is impractical for a dataset of this size, taking almost as long as h-FOG but with worse and more variant fits. In contrast,  $\alpha$ -FOG-Residual offers performance

similar to  $k$ -FOG-Hard and only three times slower than  $k$ -FOG-Soft. This performance gap might be decreased further by an implementation with efficient priority queues (the one used matches the code provide in the previous section).

---

**Algorithm 4** H-FOG

---

**Require:**  $\mathbf{L} \in [0, 1]^{m \times n}$  {link matrix}**Require:**  $k \in \mathbb{Z} < m$  {number of groups to estimate}**Require:**  $\epsilon \in \mathbb{R}$  {membership prior parameter}**Ensure:**  $\mathbf{e}, \Gamma, \theta$  {estimated emission vector and optimized group parameters}**for**  $i = 1$  to  $m$  **do**

{initialize one group per link}

 $e_i \leftarrow i$      $\theta_i \leftarrow \frac{1}{m}$      $\gamma_i \leftarrow \hat{\gamma}(\mathbf{i}); \epsilon$ **end for** $\mathbf{S} \in \mathbb{R}^{m \times m} \leftarrow 0^{m \times m}$  {initialize score matrix}**for**  $i = 1$  to  $m - 1$  **do**    **for**  $j = i + 1$  to  $m$  **do**         $s_{i,j} \leftarrow s_H(i, j)$     **end for****end for****for**  $m - k$  iterations **do**    {merge until  $k$  groups remain}     $i, j \leftarrow \operatorname{argmax}_{i,j} s_{i,j}$     **for**  $x = 1$  to  $m$  **do**        **if**  $e_x = j$  **then**             $e_x \leftarrow i$         **end if**         $s_{j,x} \leftarrow 0$          $s_{x,j} \leftarrow 0$          $\theta_i \leftarrow \theta_i + \frac{1}{m}$     **end for****for**  $x = 1$  to  $n$  **do**     $\gamma_{j,x} \leftarrow 0$ **end for** $\theta_j \leftarrow 0$  $\gamma_i \leftarrow \hat{\gamma}(\mathbf{i}); \epsilon$ **end for**

---

---

**Algorithm 5**  $k$ -FOG

---

**Require:**  $\mathbf{L} \in [0, 1]^{m \times n}$  {link matrix}**Require:**  $k \in \mathbb{Z} < m$  {number of groups to estimate}**Require:**  $\epsilon \in \mathbb{R}$  {membership prior parameters}**Ensure:**  $\mathbf{e} \in [1 \dots k]^m, \mathbf{\Gamma} \in \mathbb{R}^{k \times n}, \theta \in \mathbb{R}^k$  {estimated emission vector and optimized group parameters} $\mathbf{e}' \leftarrow 0^m$ **for**  $i = 1$  to  $m$  **do** $e'_i \sim \text{Uniform}(1 \dots k)$  {assign each link to a random group}**end for****repeat** $\mathbf{e} \leftarrow \mathbf{e}'$ **for**  $i = 1$  to  $k$  **do**

{Maximize groups given link supports}

 $\gamma_i \leftarrow \hat{\gamma}(i; \epsilon)$  $\theta_i \leftarrow \frac{\sum_{j=1}^m I(e_j=i)}{m}$ **end for****for**  $i = 1$  to  $m$  **do**

{Maximize link assignments given group parameters}

 $e'_i \leftarrow \text{argmax}_j \theta_j P(\mathbf{l}_i | \gamma_j)$ **end for****until**  $\mathbf{e} = \mathbf{e}'$ 

---

---

**Algorithm 6**  $\alpha$ -FOG
 

---

**Require:**  $\mathbf{L} \in [0, 1]^{m \times n}$  {link matrix}

**Require:**  $k \in \mathbb{Z} < m$  {number of groups to estimate}

**Require:**  $\alpha \in \mathbb{R}$  {concentration parameter}

**Require:**  $\epsilon \in \mathbb{R}$  {membership prior parameter}

**Ensure:**  $\mathbf{e} \in [1 \dots k]^m, \mathbf{\Gamma} \in \mathbb{R}^{m \times n}, \theta \in \mathbb{R}^m$  {estimated emission vector and optimized group parameters}

$\mathbf{e}' \leftarrow 0^m$  {links initially unassigned}

**repeat**

$\mathbf{e} \leftarrow \mathbf{e}'$

**for**  $i = 1$  to  $m$  **do**

$e'_i \leftarrow \operatorname{argmax}_x P(e_i = x \mid \mathbf{e}_{-i}; \alpha) P(\gamma_x \mid \mathbf{e}) P(l_i \mid \gamma_{\mathbf{x}})$  {Assign link to optimal group}

$\mathbf{\Gamma} \leftarrow \hat{\gamma}(\mathbf{e}', \mathbf{L}; \epsilon)$  {Adjust group parameters}

**end for**

**until**  $\mathbf{e} = \mathbf{e}'$

---



---

**Algorithm 7** Generation of random parameters and links

---

**Require:**  $n \in \mathbb{Z}$  {number of entities}  
**Require:**  $m \in \mathbb{Z}$  {number of links}  
**Require:**  $k \in \mathbb{Z} < m$  {number of groups}  
**Require:**  $q \in \mathbb{Z} < n$  {expected entities per link}  
**Require:**  $\alpha_\theta \in \mathbb{R}$  {frequency concentration parameter}  
**Require:**  $\alpha_\gamma \in \mathbb{R}$  {cohesion parameter}  
**Require:**  $\epsilon \in \mathbb{R}$  {membership prior parameter}  
**Ensure:**  $\Gamma \in \mathbb{R}^{m \times n}, \theta \in \mathbb{R}^m$  {random FOG parameters}  
**Ensure:**  $\mathbf{e} \in [1 \dots \mathbf{k}]^m, \mathbf{L} \in [0, 1]^{m \times n}$  {emission vector and links}  
 $\mathbf{e}' \leftarrow 0^m$  {links initially unassigned}  
 {Draw  $\theta$ }  
 $\theta \sim \text{Dirichlet}(\{\alpha_\theta\}^k)$   
 {Draw relative likelihoods for groups}  
 $\mathbf{G} \in \mathbb{R}^{k \times n}$   
**for**  $i = 1$  to  $k$  **do**  
      $\theta \sim \text{Dirichlet}(\{\alpha_\gamma\}^n)$   
**end for**  
 {Convert relative likelihoods to observation probabilities}  
 $\Gamma \in \mathbb{R}^{k \times n}$   
**for**  $i = 1$  to  $k$  **do**  
     **for**  $j = 1$  to  $n$  **do**  
          $\Gamma_{i,j} \leftarrow 1 - (1 - g_{i,j})^q$   
     **end for**  
**end for**  
 {Monte carlo draw emissions and links}  
**for**  $i = 1$  to  $m$  **do**  
      $e_i \sim \theta$   
     **for**  $j = 1$  to  $n$  **do**  
          $L_{i,j} \sim \Gamma_{j,e_i}$   
     **end for**  
**end for**

---

---

**Algorithm 8**  $\alpha$ -FOG-Residual

---

**Require:**  $\mathbf{L} \in [0, 1]^{m \times n}$  {link matrix}**Require:**  $k \in \mathbb{Z} < m$  {number of groups to estimate}**Require:**  $\epsilon \in \mathbb{R}$  {membership prior parameter}**Ensure:**  $\mathbf{e}, \Gamma, \theta$  {estimated emission vector and optimized group parameters}**for**  $i = 1$  to  $m$  **do**

{initialize one group per link}

 $e_i \leftarrow i$      $\theta_i \leftarrow \frac{1}{m}$      $\gamma_i \leftarrow \hat{\gamma}(\mathbf{i}); \epsilon$ **end for** $\mathbf{S} \in \mathbb{R}^{m \times m} \leftarrow 0^{m \times m}$  {initialize score matrix}**for**  $i = 1$  to  $m - 1$  **do**    **for**  $j = i + 1$  to  $m$  **do**         $s_{i,j} \leftarrow s_H(i, j)$     **end for****end for****for**  $m - k$  iterations **do**    {merge until  $k$  groups remain}     $i, j \leftarrow \operatorname{argmax}_{i,j} s_{i,j}$     **for**  $x = 1$  to  $m$  **do**        **if**  $e_x = j$  **then**             $e_x \leftarrow i$         **end if**         $s_{j,x} \leftarrow 0$          $s_{x,j} \leftarrow 0$          $\theta_i \leftarrow \theta_i + \frac{1}{m}$     **end for**    **for**  $x = 1$  to  $n$  **do**         $\gamma_{j,x} \leftarrow 0$     **end for**     $\theta_j \leftarrow 0$      $\gamma_i \leftarrow \hat{\gamma}(\mathbf{i}); \epsilon$ **end for**

---

Algorithm	iterations				time (s)				KLD Ratio			
	$\mu$	$\sigma$	max	min	$\mu$	$\sigma$	max	min	$\mu$	$\sigma$	max	min
h-FOG	44840.00	0.00	44840	44840	596.29	0.89	598.06	593.16	9.21	17.15	153.26	1.42
$k$ -FOG-Hard	2880.00	1003.63	7200	1500	2.05	0.78	5.44	0.97	7.26	13.59	118.78	1
$k$ -FOG-Soft	26847.00	17608.75	110700	8700	20.55	13.61	85.69	6.60	3.23	7.20	69.18	1
$\alpha$ -FOG	1947.00	3620.65	20100	600	328.37	693.84	4758.76	44.25	13.97	24.19	107.08	1
$\alpha$ -FOG-Residual	265.12	20.34	326	228	60.52	4.70	74.66	52.23	6.24	16.58	162.67	1

Table 4.1: Performance summary of FOG algorithms at convergence.

## 4.7 (Infinite) Residual Expectation Maximization (REM)

As generalized in [32], expectation maximization algorithms proceed by alternating between optimization of *all* missing data expectations (the expectation step) and *all* distribution parameters (the maximization step). ECM algorithms [83] divide the latter step into a series of *conditional* maximizations, in each of which a subset of distribution parameters are optimized while the others are held constant.  $\alpha$ -FOG and  $\alpha$ -FOG-Residual operate on a similar but inverted principle, wherein the expectation step is instead subdivided into a series of conditional optimizations. The algorithm class defined below generalizes this strategy.

**Definition 11** A Conditional Expectation Maximization (CEM) algorithm is defined on a likelihood function  $L(\mathbf{x} \mid \theta)$ , where  $\mathbf{x}$  are missing data and  $\theta$  are unknown parameters, and partition sequence  $\Omega$ , of which each member  $\omega^{(i)}$  is a binary vector indicating a subsets of the variables  $\mathbf{x}$  which will be kept constant during a series of conditional maximization steps. Let the set-valued function  $N(\omega, \mathbf{x})$  yield those binary vectors  $\{\mathbf{x}' \mid \forall_i \omega_i = 1 \Rightarrow x_i = x'_i\}$ . A CEM is characterized by the following computed sequences.

$$(4.14) \quad \mathbf{x}^{(i+1)} = \underset{\mathbf{x} \in N(\mathbf{x}^{(i)}, \omega^{(i)})}{\operatorname{argmax}} L(\mathbf{x} \mid \theta^{(i)})$$

$$(4.15) \quad \theta^{(i+1)} = \underset{\theta}{\operatorname{argmax}} L(\mathbf{x}^{(i+1)} \mid \theta)$$

It should be clear from inspection that each iteration of a CEM nonstrictly increases the likelihood function, and that fixed points reached by this process are also fixed points of any maximum *a posteriori* EM or ECM algorithm. Like the EM or ECM algorithms, for most distributions a CEM will converge on different local maxima in a varying number of steps depending on the initial conditions of the algorithm. However, for distributions with specific factorizations, incremental updates to the expectations may permit easy computation of the M-step. As an example, a CEM for a latent cluster distribution which modifies individual cluster need only update parameters pertaining the original cluster and the newly assigned cluster on each iteration. If the clusters parameters can be incrementally updated, such as those in a mixture of exponential family distributions, then only the attributes of the reassigned data point need to be considered in making this update.

Both  $\alpha$ -FOG and  $\alpha$ -FOG-residual are CEM algorithms for the same distribution, but the performance achieved by the residual-prioritized version is significantly higher. This suggests that the schedule on which these partitions are updated can have a large effect on both convergence speed and the quality of local maximum converged upon. The policy of considering all incremental updates and executing the one which causes the greatest is formalized in the definition below.

**Definition 12** An Residual Expectation Maximization (REM) algorithm is defined on a likelihood function  $L(\mathbf{x} | \theta)$ , where  $\mathbf{x}$  are missing data and  $\theta$  are unknown parameters, and a partition set  $\mathbf{S}$ , of which each member  $\omega \in \mathbf{S}$  is a binary vector indicating a subsets of the variables  $\mathbf{x}$  which will be kept constant during a series of conditional maximization steps. The REM algorithm is the CEM algorithm whose partition sequence is determined by the following equation.

$$(4.16) \quad \omega^{(i)} = \operatorname{argmax}_{\omega \in \mathbf{S}} \frac{\max_{\mathbf{x} \in N(\mathbf{x}^{(i)}, \omega)} L(\mathbf{x} | \theta^{(i)})}{L(\mathbf{x}^{(i)} | \theta^{(i)})}$$

The double maximization involved in determining the next update makes REM intractable for many distributions, but for certain distributions an efficient data structure can facilitate this computation. In FOG and other latent cluster distributions, once the initial probabilities of membership for each entity in each cluster have been computed, updating the assignment of a single member to a new cluster necessitates the recomputation of only those assignment likelihoods involving the source and destination clusters. As a result, it is possible to perform the update and maintain a table allowing the determination of the next update in time linear in the number of entities being clustered. While this makes each update more expensive than those for a random schedule, the performance of  $\alpha$ -FOG-residual suggests that this is eclipsed by the increase in convergence rate.

Another benefit of maximizing expectation for one entity at a time is that it allows the incremental exploration of certain distributions with a theoretically infinite number of parameters. Under the Dirichlet-process priors used in the generative distribution for the  $\alpha$ -FOG algorithms, the likelihood associated with an observation's assignment to a previously unobserved group is independent of all other factor parameters. This allows for parameters of new groups to be initialized only when an entity is assigned to it, and decommissioned when that entity is done. The definition below generalizes a class of distributions for which this is possible.

**Definition 13** An infinite clustering likelihood is a latent clustering likelihood with the additional properties that

- The cluster parameter matrix  $\Theta$  is presumed to have infinite first dimension.
- The clusters are exchangeable, in that the likelihood is identical for any permutation of the cluster parameter matrix and the assignments in the clustering relation.

The exchangeability requirement ensures that the likelihood of emission by a previously unobserved cluster is unchanged by that cluster's index, allowing us to compute the likelihood at most

once for each entity in order to consider its assignment to a singleton cluster. We refer to this likelihood as  $l_0(y_i)$ .

Although not a requirement for our algorithms, it bears repetition that for most applications, interesting infinite clusterings arise from functions with a tension between the size of clusters and their cohesion. If cohesion is overemphasized, the distribution tends toward small, overfit clusters; if cluster size is overemphasized then giant clusters arise capturing only mean attributes of the population. In this thesis I return to the Dirichlet distribution, which elegantly specifies expectations regarding cluster sizes and allows the tradeoff to be adjusted by a single parameter. The following pseudocode illustrates the efficient application of an REM algorithm to an infinite clustering distribution.

---

**Algorithm 9** Infinite Residual Expectation Maximization (iREM) Clustering

---

**Require:**  $Y$  {observable attributes of entities}**Require:**  $L$  {infinite clustering likelihood} $k \leftarrow 0$  {number of inferred clusters} $\mathbf{x} \leftarrow \{-1\}^n$  $\Theta \leftarrow \{0\}^{\infty \times d}$  {latent cluster parameters} $\mathbf{l} \leftarrow \{0\}^n$  {likelihood of current assignments} $\mathbf{S}, \mathbf{r} \leftarrow \emptyset$  {keyed priority queues for scores and residuals} $\epsilon \leftarrow \frac{1}{2} \min_i l_0(\mathbf{y}_i)$  {minimum singleton cluster likelihood}**for**  $i = 1$  to  $n$  **do** $INSERT(\mathbf{s}_i, 0, l_0(\mathbf{y}_i)), INSERT(\mathbf{r}, i, \frac{l_0(\mathbf{y}_i)}{\epsilon})$  {initialize queues with assignments to singleton clusters}**end for****repeat** $i \leftarrow POP(\mathbf{r})$  {draw entity with greatest residual} $j \leftarrow x_i, j' \leftarrow PEEK(\mathbf{S}_i)$  {get old and new assignments}**if**  $j' = 0$  **then** $k \leftarrow k + 1, j' \leftarrow k$  {create a new cluster}**end if****if**  $j \neq x_i$  **then** $x_i \leftarrow j'$  $UPDATE(j)$  $UPDATE(j')$  $l_i \leftarrow l(\mathbf{y}_i | \theta_{j'})$ **end if****until**  $|\mathbf{r}| = 0$ 

---

---

**Algorithm 10** UPDATE Subroutine

---

**Require:**  $j$  {cluster to update}  
 $\mathbf{z} \leftarrow \{i' : x_{i'} = j\}$  {cluster members}  
**if**  $|\mathbf{z}| = 0$  **then**  
    $DELETE(j)$   
**else**  
    $\theta_j \leftarrow \operatorname{argmax}_{\theta} \prod_{i' \in \mathbf{z}} l(\mathbf{y}_{i'} | \theta)$  {update new parameters}  
   **for**  $i' = 1$  to  $n$  **do**  
       $l' \leftarrow l(\mathbf{y}_{i'} | \theta_{\mathbf{x}_i})$   
       $UPDATE(\mathbf{s}_{i'}, x_i, l')$  {update assignment likelihoods}  
       $UPDATE(\mathbf{r}, i', PEEK(\mathbf{s}_{i'})/l')$  {update residuals}  
   **end for**  
**end if**

---

The pseudocode above makes use of two nontrivial data structures and associated subroutines.

- The parameter matrix  $\Theta$  is given infinite dimension. The associated  $DELETE(j)$  subroutine has the symantics that (1) entries associated with  $j$  should be removed from  $\mathbf{S}$ , (2) all indices  $j' > j$  should be decremented for purposes of indexing  $\Theta$  and where used in  $\mathbf{x}$  and as keys in  $\mathbf{S}$ , and (3) that  $k$  should be decremented. In practice, this is easily implemented by maintaining  $\Theta$  as a hash map from cluster identifiers to parameter vectors. The space required for  $\Theta$  should not exceed a factor of the largest number of clusters populated during runtime, which cannot exceed the number of entities  $n$ .
- The keyed priority queues  $\mathbf{S}$  and  $\mathbf{r}$  store a given key no more than once, while associating a real priority value. The  $INSERT(\mathbf{q}, i, j)$  routine inserts key  $i$  into queue  $\mathbf{q}$  with associated priority  $j$ , overwriting any previous priority associated with  $i$ . The  $PEEK(q)$  operation retrieves the current key with top priority, and the  $POP(q)$  operation does the same while removing it from the queue. This can be implemented efficiently with a combination of a hash map and a heap, where the insert operation performs local rotations to rebalance when an update breaks heap invariants. All operations should be worst-case logarithmic in heap size, and the heaps themselves reach maximum sizes of  $n$  or  $k$ .

If we implement as above, assume constant evaluation time of  $L$ , and use the fact  $k \leq n$  to simplify our notation, we can see by inspection that the a single iteration's time complexity is determined by the time complexity of  $UPDATE$ , which is  $O(n \log n)$ . By contrast, an iteration of



a CEM algorithm with fixed update schedule requires only  $O(k+d)$  time, as we consider an entity’s emission likelihood from  $k$  clusters and then update  $d$  parameters. However, iterations of REM tend to be closer to linear in  $n$ , as frequently  $k \ll n$  and keyed priority queue updates require little or no changes to the structure of the heap. As the time results for  $\alpha$ -FOG-Residual show, the benefit of targeting updates can more than make up for their expense.

## 4.8 Discussion

I set out to introduce a new quantitative tool for inferring complex relationships between individuals and groups, allowing varied degrees of participation in multiple groups. I proposed the FOG stochastic model, which dictates relationships between individuals, groups, and observable interactions as a generative model for link data. To make FOG a useful analysis tool, I introduced the H-FOG,  $k$ -FOG, and  $\alpha$ -FOG algorithms intended to assist exploratory group discovery, fast composition estimation with a fixed number of groups, and estimation of an unknown number of groups respectively. To investigate single-mode network data, I implemented a simple method for generating rich multi-entity links from a pairwise network based on a simplistic simulation of interaction processes.

### 4.8.1 Validation

One approach to empirical validation of grouping algorithms is to compare our statistical analysis to that of anthropologists like Sampson, who were able to relate their intuitive observations to unforeseen events in the social group. Can fuzzy grouping rediscover social patterns that stood out to ethnographers in the field?

In the two datasets I studied, the answer is yes. The discrete groups identified by both Sampson and the DGG team were nearly identical to the list of individuals with greatest affiliation to each group in our analysis. Additionally, substructures and leadership roles identified by the original authors corresponded strongly to the levels of affiliation I discovered. FOG sits well among a variety of mathematical approaches which have supported the original intuitive analyses. However, these have usually relied on separate techniques to distinguish groups, leaders, and internal structures. One advantage of FOG is the ability to unify these multiple levels of analysis under a simple model.

A more quantitative test, which has been applied to link analytic methods including  $k$ -Groups and iterative deduplication, is the ability of a method to rediscover groups from artificially generated data. I performed such a parameter recovery analysis by first drawing group parameters from a

random distribution intended to produce realistic data, then estimating the KL divergence between the discovered groups and a distribution fit directly to the samples. While the results cannot be interpreted directly with respect to a general grouping task, all three algorithms showed significant convergence over time from a naive initial hypothesis toward the target distribution.

The  $k$ -FOG algorithms, which requires the most prior information (in the form of number of latent groups) and provides the minimal output (a single grouping of that order), converged faster and with lower divergence from the originating distributions on average. Interestingly, it also showed a lower variance than other methods in divergence, suggesting limited susceptibility to local maxima. The speed / performance tradeoff between the two algorithms motivates exploration of a hybrid approach, in which hard clustering is used as a hot-start for soft-clustering inference.

H-FOG displayed a deterministic runtime much slower than  $k$ -FOG's, and reached much lower levels of accuracy even when evaluated at the correct number of groups.  $\alpha$ -FOG was similarly impractical, showing worst-case time to convergence two orders of magnitude greater than the  $k$ -FOG algorithms in some pathological cases. However, the  $\alpha$ -FOG-Residual was surprisingly competitive in performance, given its additional challenge in estimating the number of latent groups, and could be practically applied to small datasets like the 300 link test cases examined. The performance improvement generated by residual based updates is promising and potentially applicable to other EM algorithms.

### 4.8.2 Interstitial Roles

The existence of interstitial roles, where an individual retains several group affiliations, was our principle motivation for developing a fuzzy grouper. I identified many such individuals in our analysis, fitting several profiles. With great frequency, the most apparent leaders of a group had weak ties to other groups as well, as did those members with the least affiliation to any group. The differentiation of these two roles, as well as the surprising result that most groups contained a well-embedded middle tier, would be difficult without FOGs novel properties: the combination of multiple memberships and degrees of membership. As FOG is applied to additional datasets I expect that a better understanding of individual roles based on multiple memberships will emerge. The FOG approach holds promise of providing a mathematical base for capturing and defining some critical types of social roles not heretofore measurable.

It is worth noting that FOG did not always identify as interstitial the individuals whom I would expect. In some cases, such as with Sampsons waverers, individuals who were considered interstitial by an observer were placed in single groups by FOG. Conversely, some of the secondary

clique members in the DGG dataset would appear to be interstitial on a reading of our charts, but were only claimed by a single clique in specific surveys conducted by Davis et al. The distinction between members who are simply weakly connected and those who fill an actively interstitial role may be beyond our level of analysis. Alternatively, noise may have been introduced in the specific data I examined, or results may have been misinterpreted by the original observers. Since analysis of interstitial roles is a vital component of FOG, future work should investigate in depth what factors in data affect our ability to differentiate roles.

### 4.8.3 Generating link data from networks

Although the theory underlying the FOG model requires link data that indicates a shared context between members, I am optimistic about the ability to examine single-mode network data by generating fake data from simulated interactions. In the Sampson data, I was able to affirm existing knowledge about the monastery social groups using this approach, while generating new theories.

A crucial aspect of this analysis was to connect the final results with the assumptions under which link data was generated. Since Breiger *et al.*'s matrix indicated relationships between novices that could lead to interaction, I built our link generator as a simulator of social contexts that spread infectively through iterative interactions. This type of link increased the observation frequency of high-betweenness individuals, but we might expect those individuals to be disproportionately represented in real data recording this type of interaction. Understanding this bias helped us interpret the difference between embedded and interstitial members when interpreting the role of novice Winfred (12), the last leader of the young Turks.

A potential criticism of random link generation is that it injects variance into our analysis. In this study, I approached the problem by generating larger sets of random trees until differences between runs were below the threshold of our qualitative analysis. However, increasing the number of samples comes at a significant computational cost. I attempted a similar process using a network from a cleaned corporate email corpus [33] containing 150 users, and were still experiencing visible variance between results when using samples of 450 links at an average runtime of over 8h. Notably, some networks drawn from 2-mode data may be easier to analyze in their original form. I was able to perform an informative analysis of the same emails in only 30min by multi-recipient emails rather than random walks on the incidence network as observations.

Another peculiarity of the random-tree link model is that it discards the directionality of links. Since FOG interprets only the presence or absence of an individual in a link, no distinction is drawn

between individuals originating a random observation and those added subsequently. This affects the placement of individuals like Amand(13), who appeared in many interactions with individuals whose admiration or affection he did not reciprocate. One could again argue that many types of real data would have similar confusion, but it is also possible that a link model could be adjusted to include this information.

Networks of different relations may require different link models. In a formal communication network, such as a corporate hierarchy, where messages pass along a fixed route from source to destination, a random, directed walk would be more appropriate than a random tree. It might be convenient to analyze 2-mode networks by simply interpreting one of the modes as links, but that decision should similarly depend on the type of relationship represented in the network. Link generation for multi-mode networks is another direction for potential improvement.

#### 4.8.4 Analyzing and visualizing fuzzy relationships

Social groups with binary memberships can be analyzed by common statistical techniques. For example, when Davis *et al.* introduced the southern women dataset as overlapping cliques, they were able to investigate the character of each clique by taking aggregate statistics over its members. The same analysis would be non-trivial for a FOG cluster. What is the mean income of the members of a fuzzy group? The question is especially difficult because our results are intended to denote a level of participation, and not necessarily the degree to which members are representative of their group. If fuzzy groupings become a useful analytic tool, new measurements will have to be developed or adapted to properly interpret the new information given.

I tried to uphold several principles in my qualitative analysis of derived clusters. First, membership values were not examined independently of the context of other memberships held by the same individual and to the same group. Groups or individuals may have different average memberships, for reasons that have less to do with the actual importance of those memberships than with the nature of group events or the way data was collected or generated. Secondly, the novel strength of grouping with multiple, variable memberships is the ability to compare several simultaneously occurring memberships in individuals. FOG is intended to define and investigate roles that are involve multiple memberships, rather than to rehash issues of internal group structure that have been examined by other algorithms.

Visualizations play an especially important role by influencing the types of patterns we can identify intuitively. I have presented two visualization paradigms in this chapter, one indicating individuals memberships to groups as a weighted two-mode network, and the other a spectrographic

view providing all membership levels explicitly in bar chart form. As with most visualizations of overlapping clusters, placement of individuals can be difficult as the page does not have enough dimensions to represent all association patterns. I had few enough groups in both of my analyses that we were able to position individuals for reasonable clarity, but this would not be true in more complicated datasets. I have experimented with several heuristics for laying out more than two groups in spectrographic figures, but more work needs to be done in this area.

#### 4.8.5 Efficient EM algorithms for an unknown number of clusters

The generalized EM algorithm class of [32] has seen a huge variety of variations, perhaps because the problems of creating structured likelihood functions and designing incremental update schedules which exploit that structure are so closely related. Each of the “innovations” in my most generally successful algorithm,  $\alpha$ -FOG-residual – optimizing the expectation of individual entities, using a residual prioritization schedule, and dynamically populating sparse clusters from an infinite set – have been used in other local search algorithms. For the more general problems of sampling from or estimating posterior probabilities in non- or infinite-parametric distributions, there exists a more robust literature under the hierarchical Bayesian paradigm. In defining the REM algorithm class, I have tried to incorporate each of these properties in an efficient and simple to implement structure applicable to a narrow but frequently encountered class of clustering problems. In the following chapter, I generalize REM for graphical structures that go beyond clustering and address temporal (and potentially spatial) regularities.

#### 4.8.6 Final thoughts and remaining questions

The stochastic model and algorithms presented in this chapter provide a starting point for detection of fuzzy, overlapping groups in a number of contexts. The input data may be a bimodal or unimodal relation, and by using H-FOG,  $k$ -FOG or  $\alpha$ -FOG respectively the analysis mode may be very exploratory or utilize prior knowledge of the analyst. However, there exist many desirable extensions, such as the inclusion of attributed data or simultaneous analysis of multiple relations. In this direction, it is possible that FOG would benefit from integration with ERGM models [107] [103] [53], which can already express distributions over graphs including these characteristics. The  $k$ -FOG and  $\alpha$ -FOG algorithms are themselves instantiations of the expected conditional maximization algorithm class; FOG could be generalized in such a way as to permit further instantiations itself.

In my review of previous literature, I mentioned that graph partitioning was of interest for the parallelization of certain algorithms. Since the input graphs to such algorithms often derive from “natural” phenomena, including social networks, recent work in this area has focused on identification of and adaptation to common properties in such networks, such as scale-free edge distributions [4]. Interestingly, one outcome of this research has been the realization that many naturally occurring networks may have no “good” cuts, in the sense that the subgraphs larger than a certain size preserve meaningful properties of the original network [70]. The FOG algorithms, because they do not rely on strict partitionings, may be interesting in describing substructure at this higher level. However, because FOG does not provide a strict partitioning, it is not an appropriate choice for finding subsets for parallel computation. The scalability properties of the current algorithms might also preclude use on networks large enough to exceed the approximate Dunbar number that seems to bound the size of meaningful partitions.

There are several promising directions by which to measurably improve the performance of the FOG algorithms. I did not explore their convergence properties as relates sample size, which might be informative toward possible subsampling approaches for dealing with large datasets. One direction for enhancement is parallelization, as each of the algorithms populate matrices full of computations which could be performed asynchronously. Another is the mixing of hard and soft clustering paradigms, and a final is the application of efficient queuing mechanisms within the h-FOG and  $\alpha$ -FOG-Residual algorithms.

## Chapter 5

# Path Prediction

### 5.1 Problem Definition

Many real world sensor networks are capable of simultaneously tracking many agents as they navigate a space. Examples include network-linked GPS devices in phones and vehicles, passive tracking mechanisms such as radar (when paired with technology to distinguish agent identities), and entry-point systems such as keycard and RFID scanners. (Many additional examples emerge when we consider agents navigating virtual spaces such as the World Wide Web, but this paper will focus on a physical system). As the volume and complexity of data produced by these systems has grown, human monitors are increasingly dependent on algorithms that efficiently extract relevant patterns for their analysis.

One successful approach to pattern mining in this domain has been to presume the existence of hidden variables which mediate the transitions observed by sensors. For example, there may be a hidden activity that explains the time an agent spends at a certain location, or an underlying plan that explains the choice to travel from one place to another. Probabilistic relationships between hidden variables and observed variables can be encoded with graphical models such as Conditional Random Fields (CRFs), which support efficient algorithms for inferring missing values. Previous work have used this general approach to predict future agent actions and detect surprising deviations from typical behavior [73]. Applications have been discussed in contexts ranging from robotic and human planning [7] to assistance of seniors and disabled individuals [72].

Inference from this type of model is generally preceded by a training phase, in which model parameters are optimized against a dataset for which “true” values of hidden variables have been provided. In previous experiments, training data were drawn from journals of experiment partici-

pants, hand-coded by human observers, or extracted from existing databases (*e.g.* maps of known locations or obstacles). In this paper, we examine a case where such data would be useful but is unavailable. Our dataset tracks the movement of 1700 merchant marine vessels servicing ports in the English channel over a 5 day period. Maritime navigation is organized around “soft” conventions, such as shipping lanes and known waypoints, rather than “hard” constraints, such as roads or walls. Because some conventions differ between nationalities, companies, and ship types, as well as changing over time, there is no single collation that could be encoded directly into our model. The same diversity of backgrounds and conventions would make conducting a survey with reasonable accuracy and breadth cost-prohibitive.

The maritime behavior modeling domain is one of many in which a graphical structure naturally describes the process, but expecting the existence of training data is unrealistic. Others include covert applications where subjects cannot be polled, large scale applications where surveys would be expensive or inaccurate, and evolving environments in which training data quickly becomes outdated. As a solution I describe in this section an unsupervised approach to graphical models, allowing the user to exploit knowledge of the structure of hidden variables in a system without observing them - even during training. I introduce 2 algorithms that concurrently assign variable values and optimize model parameters in a way that is self-consistent given the model structure. The model and algorithms are simpler than the most advanced supervised methods, but gives compelling results on destination prediction task, and demonstrates a variety of challenges involved in creating an unsupervised approach.

The rest of this chapter is organized as follows. In section 5.2 I describe the graphical model used to relate observed AIS data and latent behavioral parameters. In section 5.3, I define unsupervised learning for factor graphs and discuss its from the points of view of local search and expectation maximization. In sections 5.3 and 5.3 I give two algorithms which conduct unsupervised learning using different update schedules; their performance on a destination prediction task is compared in section 5.4. Finally, in 5.5, I address the challenge of prioritizing iterative updates which will have the greatest impact on model likelihood. I show that some unsupervised factors, including those associated with clustering distributions, have parameterizations that allow for efficient REM algorithms similar to  $\alpha$ -FOG-Residual. Exploiting this structure leads to faster and more accurate unsupervised inference on a simplified version of our factor graph.



## 5.2 Factor Graphs for Path Prediction

The trend in previous work has been to provide increasingly complex graphical models to incorporate additional sensor data (e.g. use of street maps in [73]) or knowledge regarding relationship structure (e.g. modeling of activity duration by [37]). In order to concentrate on unsupervised learning, we employed the relatively simple, two-layer model shown in figure 5.1. The variables in

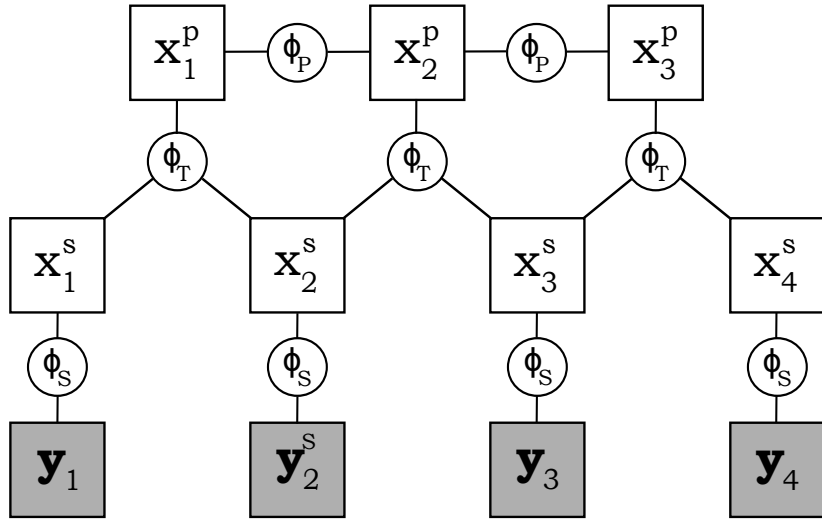


Figure 5.1: Plan Prediction Factor Graph

our factor graph include the following.

- Each  $\mathbf{y}_t$  is an *observed* vector in  $\mathbb{R}^3$  containing the latitude, longitude, and speed at time  $t$ .
- Each  $x_t^s$  is a hidden, discrete *state* variable representing the instantaneous state of an agent. It takes on integer values  $0 \leq x_t^s < n_s$ . These are intended to capture the notion of well-known waypoints a captain might route between when charting a course.
- Each  $x_t^p$  is a hidden, discrete *plan* variable capturing an *internal* agent state persisting over several time periods. It takes on integer values  $0 \leq x_t^p < n_p$ . This is intended to capture the notion of a route which might affect the planning of several sequential waypoints.

The following factors model relationships in our graphs.

- $\phi_S(x_t^s, \mathbf{y}_t)$  is a *state compatibility* factor which measures the likelihood of observation  $\mathbf{y}_t$  being generated when within state  $x_t^s$ . Since state variables are intended to capture the idea of a fixed “place”, the likelihood function corresponds to a Gaussian distribution on {latitude, longitude, speed} for each state.  $\phi_S$  is implemented by maintaining mean and covariance matrices for each of  $n_s$  Gaussians. To avoid overfitting the Gaussians corresponding to infrequently observed states, each one is initialized with a mean prior drawn from a uniform distribution over the range of latitudes, longitudes and speeds. The prior covariance is the covariance matrix for the same uniform distribution.
- $\phi_T(x_t^s, x_{t+1}^s, x_t^p)$  is a *state transition* factor which measures the likelihood of transitioning from  $x_t^s$  to  $x_{t+1}^s$  when plan  $x_t^p$  is active. The plan state  $x_t^p$  can represent (for example) the propensity of an agent to select a different route when targeting a different destination. This factor is parameterized as a likelihood table for all possible transitions, and is initialized with a uniform prior to ensure that a minimal probability remains for unobserved transitions.
- $\phi_P(x_t^p, x_{t+1}^p)$  is a *plan transition* factor which measures the likelihood of switching from  $x_t^p$  to  $x_{t+1}^p$ . Whereas the state transition factors capture physical constraints (the need to move between adjacent locations states) and tendencies in the context of the plan, the primary purpose of the plan transition factor is to model how frequently agents are expected to change longer term objectives. This factor has a single parameter, the change probability, which we initialize to .2 to indicate an expected time scale of plans being maintained for approximately 5 hours (the average time-at-sea we observed in ships that went to port). Although this parameter (and therefore the time-scale of a plan) can change during training, this initial setting plays an important role in determining which maximal labeling we will reach.

The figure above depicts the variables associated with the path of a single vessel. If we observed a single ship for an exceptionally long time, we might conceivably train a model using only this trace, since it would generate many observations of the same states and transitions wherever there was regular behavior. Since our data sample instead covers many ships over a small time, we must instead smooth between ships by placing all such traces within a single factor graph. Although the resulting graph has disconnected components (one for each ship), each component includes factors sharing the same parameters. During unsupervised learning, these parameters act as a conduit for information between paths, so that a single set of well traveled waypoints and routes is learned.

### 5.3 Unsupervised Learning for Factor Graphs

During *supervised* learning, factor parameters are generally found maximizing the expectation of some training set  $S = \mathbf{s}$  within which both  $\mathbf{x}$  and  $\mathbf{y}$  are known.

$$(5.1) \quad \mathbf{w}^*(\mathbf{s}) = \underset{\mathbf{w}}{\operatorname{argmax}} P(\mathbf{S})$$

Maximum likelihood estimation (MLE) can then be performed by finding the assignment to hidden variables  $X$  that has maximum likelihood under factor parameters  $\mathbf{w}^*$ .<sup>1</sup>

$$(5.2) \quad \mathbf{x}^*(\mathbf{w}) = \underset{\mathbf{x}}{\operatorname{argmax}} P_{\mathbf{w}^*(\mathbf{t})}(X = \mathbf{x})$$

In the unsupervised case, no true values are provided, preventing sequential learning and inference. As an alternative goal, we seek to find an assignment satisfying the fixed point of (5.1) and (6.7):

$$(5.3) \quad \mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmax}} P_{\mathbf{w}^*(\mathbf{x}^*)}(X = \mathbf{x})$$

To compare the many possible solutions to (5.3), we introduce the *self-consistency likelihood*, a scoring function favoring assignments which receive high probability under their own optimal parameter values:

$$(5.4) \quad \bar{L}(\mathbf{x}) = P_{\mathbf{w}^*(\mathbf{x})}(X = \mathbf{x})$$

The global maximum of  $\bar{L}$  is the fixed point with maximum self-consistency. However, finding it is challenging on several levels. First, the space of possible assignment vectors is far too large (size  $n^{|X|}$ ) to enumerate or sample meaningfully. Second, evaluating  $\bar{L}(\mathbf{x})$  is expensive: one must first compute the parameter values  $\mathbf{w}^*(\mathbf{x})$ , and then the partition constant  $z$  for the corresponding distribution.

---

<sup>1</sup>In most applications the training sets  $X$  and  $T$  may be different sizes or even “shapes” in terms of relations between variables. However, if a generator is provided for instantiating the same factors on both sets, parameter sharing allows us to reuse a single parameter vector.

Algorithms for supervised inference on PGMs face the same challenges above, and most overcome them using local search informed by the graphical structure. For example, the max-residual belief propagation (MRBP) algorithm maintains a set of messages corresponding to graphical ties, and incrementally update message values in a way that it is guaranteed to reduce a free energy quantity. Unfortunately, these methods cannot be directly applied to maximize our target,  $\bar{L}$ . Whereas changing the value of a variable  $x$  in a graph with fixed factor parameters affects only local likelihoods, it can potentially effect *all* factor instances used to calculate  $\bar{L}$ . This is because the change may affect the optimal parameter settings for all factors for which  $x$  participates in an instance. An alternate way to describe this effect is that the distribution  $\bar{P}$  achieved by normalizing  $\bar{L}$  no longer induces the independences given in (2.5) – the Markov blanket for  $x$  under  $\bar{P}$  includes all variables with which it shares a *factor*, not an instance.

However, under certain circumstances these “long range effects” may be limited. Let  $w^{f*}(\mathbf{x})$  be the optimal parameter assignments for a single factor under assignments  $\mathbf{x}$ , and let  $\mathbf{x} \leftarrow (x, c)$  be an operator returning an updated assignment vector with variable  $x$  set to state  $c$ . Now consider the condition

$$(5.5) \quad \forall_{x,c} \lim_{|I(f)| \rightarrow \infty} w^{f*}(\mathbf{x}) - w^{f*}(\mathbf{x} \leftarrow (x, c)) = 0$$

In other words, as the number of instances of a factor grows, the incremental change to optimal parameters caused by changing the value of a single variable approaches zero. Many common factor parameterizations satisfy this condition, including those we use and list in section 5.2 (modulo the assumption that we observe all Gaussians and state transitions a sufficient number of times). Under this condition, the effect under  $\bar{P}$  that changing  $x$  has on local likelihoods outside  $N(x)$  becomes negligible as our graph becomes larger.

Armed with this intuition, we define a local search with an operator  $\delta : \mathbb{Z}^{|X|} \rightarrow \mathbb{Z}^{|X|}$ , which produces a sequence of assignment vectors following  $\mathbf{x}^{(i)} = \delta(\mathbf{x}^{(i-1)})$ . If  $\delta$  is such that

$$(5.6) \quad P_{\mathbf{w}^*(\mathbf{x})}(\delta(\mathbf{x})) \geq P_{\mathbf{w}^*(\mathbf{x})}(\mathbf{x})$$

then its fixed point must satisfy (5.3) as well (assuming that it does not trivially self-cycle). In the following subsections we introduce two operators that satisfy this condition, but have different properties in terms of convergence rate and susceptibility to local maxima while maximizing  $\bar{L}$ .

The  $\delta$ -operator notation in this section has been adopted for compatibility with the notation of [38] relating to asynchronous belief propagation schemes. However, we can also see that a valid

$\delta$ -operator can be constructed by combining the update steps of any CEM algorithm (definition 11). In section 5.5 I explore this further by introducing additional properties of a factor graph which permit tractable application of a full REM algorithm.

### Asynchronous EM

My first local search operator improves an assignment vector incrementally by setting one variable at a time to the value with maximum expectation under the current state. The successor is

$$\delta_A(\mathbf{x}^t)_i = \begin{cases} i \neq j^{(t)} : & x_i^{(t)} \\ i = j^{(t)} : & \operatorname{argmax}_{x_i} P_{\mathbf{w}^*(\mathbf{x}^t)}(x_i | \mathbf{x}_{-i}^{(t)}) \end{cases}$$

where  $x^t$  is drawn from a round robin schedule established in advance. This operator is easy to implement for our graph because our factors support incremental updates: changing the value of  $x$  changes only factor instances  $I^{-1}(x)$ , and each of our factors can be readjusted to give maximum expectation to a new instance assignment in constant time. When describing an iteration of the algorithm we include one update for each variable, in order to standardize the unit of work by graph size. Pseudocode for an implementation of  $\delta_A$  can be found as Algorithm 11.

The initial assignments  $\mathbf{x}^0$  are selected in the following way. First, a vector of random assignments  $\mathbf{x}'$  is established. Then, each variable is set to its maximum likelihood value with neighbor variables assigned according to  $\mathbf{x}'$  using the prior factor parameters. This “stacking” of the initial state assures that initial factor parameters fully explore the range of possible values they can take on. In testing, we found that making sure that initial parameters were distributed was essential to avoiding bad local maxima. For example, maximizing initial factor parameters against a random allocation vector tended to initialize all Gaussians in state factors to have means near the actual mean coordinates for the data. This initial clustering resulted in poor exploration of the state space, with most clusters remaining near the map center even after many iterations.

### Synchronous EM

My second operator synchronously updates all variable values to a maximum likelihood estimate under the current factor parameters:

$$(5.7) \quad \delta_S(\mathbf{x}^t) = \operatorname{argmax}_{\mathbf{x}} P_{\mathbf{w}^*(\mathbf{x}^t)}(\mathbf{x})$$

This is analogous to a standard EM algorithm, in which cluster assignments and cluster parameters are updated in an alternating fashion. We hypothesized that taking larger steps in the

---

**Algorithm 11** ASYNCHRONOUS

---

$\mathbf{w}^0 \leftarrow$  Draws from prior  
 $\mathbf{x}^0 \leftarrow$  Random allocation  
 $t \leftarrow 1$   
**loop**  
 $\mathbf{w}^{(t)} \leftarrow \mathbf{w}^{(t-1)}$   
 $\mathbf{x}^{(t)} \leftarrow \mathbf{x}^{(t-1)}$   
**for all**  $x \in X$  **do**  
 $\mathbf{x}^{(t)} \leftarrow (\mathbf{x}^{(t)} \leftarrow x, \operatorname{argmax}_c P_{\mathbf{w}^{(t)}}(X = \mathbf{x}^{(t)} \leftarrow (x^{(t)}, c)))$   
 $\mathbf{w}^{(t)} \leftarrow \operatorname{argmax}_{\mathbf{w}} P_{\mathbf{w}}(X = \mathbf{x}^{(t)})$  (local updates)  
**end for**  
 $t \leftarrow t + 1$   
**end loop**

---



---

**Algorithm 12** SYNCHRONOUS

---

$\mathbf{w}^0 \leftarrow$  Draws from prior  
 $t \leftarrow 0$   
**loop**  
 $\mathbf{x}^{(t)} \leftarrow \operatorname{MLE}_{\mathbf{w}}(\mathbf{x})$  (calls MLBP)  
 $\mathbf{w}^{(t+1)} \leftarrow \operatorname{argmax}_{\mathbf{w}} P_{\mathbf{w}}(X = \mathbf{x}^{(t)})$   
 $t \leftarrow t + 1$   
**end loop**

---

space of assignment vectors might make us less susceptible to local minima. However, by changing assignments to many variables at once, we may be less protected by the guarantee in (5.5).

Pseudocode for this method is listed as Algorithm 12. We initialize cluster parameters with priors as we did for the asynchronous method, but it is unnecessary to initiate the first state as we will be using maximum likelihood belief propagation, which depends only on observed variables and factor parameters. Then, at each step, we conduct inference with the current factor parameters using MLBP. Finally, we re-optimize factor parameters to the new assignment.

## 5.4 Plan Projection Experiment

### 5.4.1 Method

We designed an experiment to simulate our system’s performance at a fundamental task: using the estimated plan of an agent at sea to predict where it will next make port. Our experiment proceeds in two phases. First, we perform unsupervised learning on a test set representing sequences that would have occurred prior to some test sequences, as well as on the first portion of the test sequences themselves. Then, using the labels and factor parameters assigned during the first phase, we sample a distribution of future states for the test set, in order to estimate its next stop.

To create a dataset of sequences appropriate to this task, we developed the following test. First, we included only observations from ships with five consecutive observations “in motion” (reported velocity over 1 km / h) to eliminate a large percentage of ships that did not move often enough to assist training of transition probabilities. Since our model does not explicitly address the duration between observations, we standardized this quantity by eliminating sequences whose inter-observational interval was outlying (over 3 hours). A total of 13715 individual observations fell into sequences in this category. Then, for the test set, we isolated the 457 subsequences within the test set that consisted of routes beginning in motion and ending stopped, with at least 5 segments in between. The criterion on test sequence length is the only one of these filter that could not be applied without full knowledge of the test set, but was necessary to ensure that each test sequence A) was long enough for us to instantiate a factor graph with all factors on, and B) had a buffer beyond this so that we would be forced to predict multiple state transitions.

To calculate a maximum likelihood estimate for the next portfall of a particular test ship, we appended 100 additional hidden plans and states (along with associated factors) to the section of the ship’s factor graph which was optimized during training. We then ran Gibbs a sampler on these hidden states using the factor parameters learned during training. Every 1000 iterations we

would extract a prediction from the sampler by recording the mean position of the first state whose expected velocity was under 1 km / h.

### 5.4.2 Results and Analysis

Visual inspection of the locations and transition probabilities learned by our algorithm confirms that it produces a coarse but credible summary of traffic flow in the channel. Figure 5.2 shows probable transitions within one plan, trained with our asynchronous algorithm and visualized using Google Earth. Vertexes are placed at the Gaussian mean for each state, with edges placed between transitions with high probability. Transition edges do not necessarily indicate a path (for example, many extend over land), but simply a start and end point. These transition could potentially be post-processed with known land locations to produce more realistic paths. It is also worth noting that no single ship likely visited all of the paths on the plan. Because the plan only affects transitions conditioned on the starting location, there is no penalty for a single plan being “overloaded” with paths from dissociated end points. For interpretable plans associated with a single start and endpoint, we might need to add an additional factor favoring a sparse adjacency relation between plans and state variables.

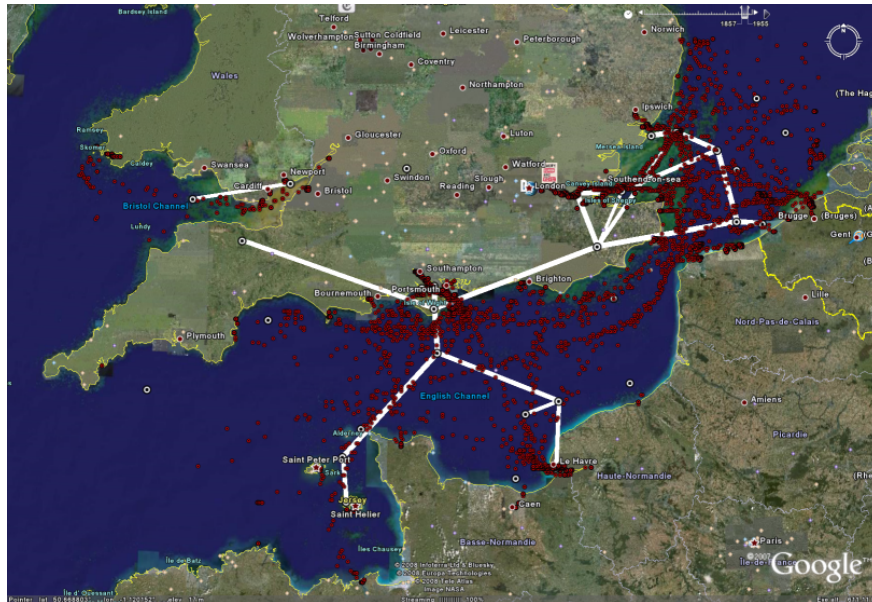


Figure 5.2: Probable locations and transitions associated one learned plan.

To measure accuracy on our portfall prediction task, we computed the surface distance between the predicted destination and the actual portfall associated with each prediction and plotted the



inverse cumulative density for this figure as Figure 5.3. The curve summarizes a set of probably approximately correct (PAC) bounds for the estimator. For example, models trained with the asynchronous algorithm achieved accuracy under 100km 71% of the time. Synchronous models had only a 53% chance of achieving this accuracy.

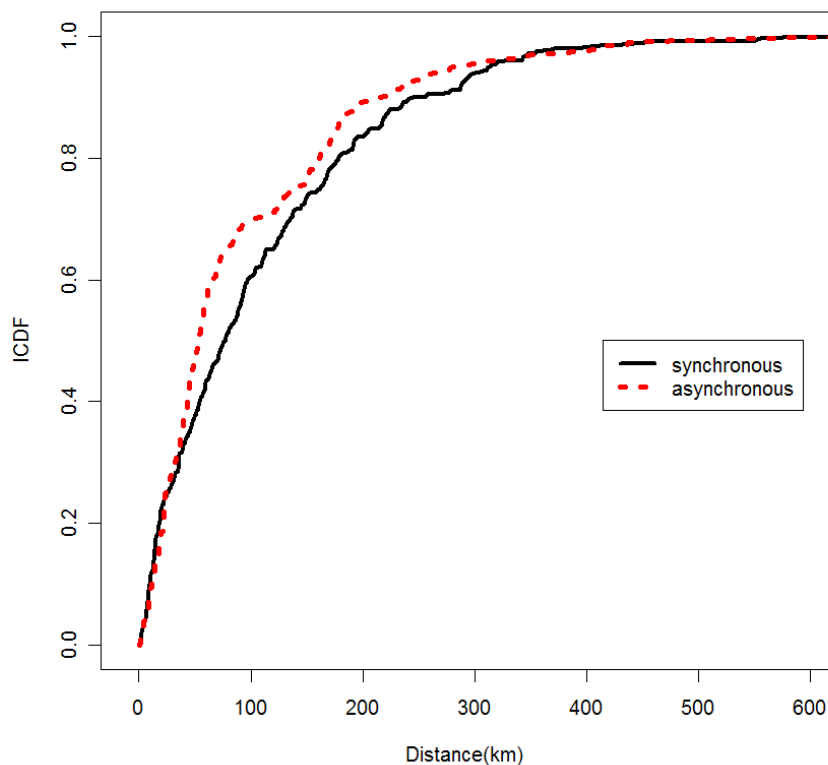


Figure 5.3: Inverse cumulative density function for error

Another important factor in algorithm choice for probabilistic graphical models is time to convergence. We measured this by counting the number of variables updated in each iteration of the algorithm. To minimize the impact of random starting configuration, we ran 5 trials to 20 iterations with each algorithm, producing the mean updates and error bars shown in figure 5.4.

Overall, the predictions made by the model were well short of the accuracy needed for most real world applications of this system. For example, if the goal was to meet the ship upon portfall, then in many parts of the English channel there would be several potential ports within the 100km radius mentioned above. However, the results do show that asynchronous updates dominate synchronous

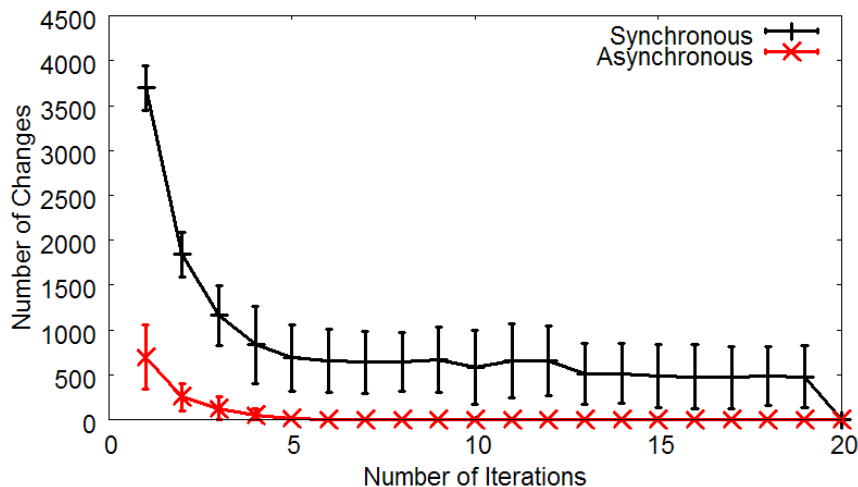


Figure 5.4: Convergence rates for the two learning algorithms

updates in terms of both probable approximate correctness and convergence rate. We were surprised to find that after only 4 cycles of updates the asynchronous algorithm reached a fixed point in most cases. In contrast, the synchronous algorithm seemed prone to cycles in which each iteration toggled significant number of predictions even after 20 iterations.

## 5.5 Residual Expectation Maximizing Belief Propagation (REM-BP)

Having established that even an arbitrary asynchronous update schedule can result in faster and more accurate unsupervised learning in factor graphs, I now consider whether additional performance can be achieved by prioritizing updates which will have a large impact on the consistency of the model.

### 5.5.1 Multi-clustering factors and factor graphs

To apply the same principle to factor graphs, I adapt the definition of a latent clustering distribution to apply to factors. Since some factors, such as the state transition factor in our AIS graph, involve more than one unobserved variable, our “cluster parameters” must relate to a configuration of such variables rather than a single value. To discuss this, I introduce a *configuration function*

$$\zeta : \{1 \dots k\}^l \rightarrow 1 \dots k^l,$$

which associates a categorical label with each configuration of the unobserved domain of a multi-clustering factor. As with the representation illustrated in figure 2.2, I initially presume a single configuration function with associated  $k$  and  $l$ ; in practice there would be a distinct function for each class of factors that is repeatedly instanced in a graph. The following definition gives conditions on the way factors are parameterized which will make computations introduced later more tractable.

**Definition 14** A factor  $\phi(\mathbf{x}, \mathbf{y}; \mathbf{w}) : \mathbb{Z}^k, \mathbb{R}^*; \mathbb{R}^l \rightarrow \mathbb{R}^+$ , is a multi-clustering factor (MCF) if it can be rewritten as  $\phi(\zeta(\mathbf{x}), \mathbf{y}; \mathbf{w}_{\zeta(\mathbf{x})})$ , where  $\mathbf{w}_{\zeta(\mathbf{x})}$  is a distinct (non-overlapping) set of parameters associated with configuration  $\zeta(\mathbf{x})$ .

We may also define an *infinite multi-clustering factor* analogous to definition 13 for cases in which one or more of the configuration dimensions (and therefore the configuration space itself) can take a potentially infinite number of valuables. To do so, we impose an exchangeability requirement that any permutation of the labels on unobserved variables, and any permutation on the the labeling imposed by the configuration function, give the same likelihood so long as the parameter matrix is identically permuted. In other words, the label itself cannot be part of the likelihood function. This ensures that the maximum likelihood estimate for parameters of all configurations which have not been observed will be identical, allowing us to address them with a single computation in later algorithms. I reserve 0 as a placeholder index for as-yet unobserved variables and configurations, so that  $\exists i \mid x_i = 0 \rightarrow \zeta(\mathbf{x}) = 0$ , and the likelihood of a novel configuration is always  $\phi(0 \mid \mathbf{y})$ .

A single (infinite) multi-clustering factor is essentially an (infinite) latent cluster likelihood for a configuration space, but they can be joined in a graph to represent co-clustering problems with a wide range of independence structures. The structure captured by multi-clustering factors is not addressed by normal factor graphs because it factors the distribution according to subsets of unobserved *values*, rather than by subsets of *variables*. To represent this, I define an augmented factor graph where configurations of factor domains intermediate the parameters used in calculating potentials.

**Definition 15** An multi-clustering factor graph (MCFG) for  $n$  unobserved variables assuming  $k$  distinct values, with  $m$  factors, each of which relating at most  $d$  unobserved variables which can participate in up to  $d$  distinct factors,  $d$  observed variables, and  $d$  factor parameters per configuration, consists of a tuple,

$$\mathcal{MCFG} : \langle \Phi, \mathbf{M}, \mathbf{Y} \rangle$$

where

- $\Phi \in \{\mathbb{Z}^k, \mathbb{R}^d; \mathbb{R}^d \rightarrow \mathbb{R}^+\}^m$  is the factor list consisting of (infinite) multi-clustering factors.
- $\mathbf{M} \in \{0 \dots d\}^{m \times n}$  is the configuration matrix, where  $m_{ij}$  indicates the participation (if nonzero) and position of unobserved variable  $j$  within the configuration associated with factor potential  $\phi_i$ . Each row and column contains at most  $d$  nonzero entries.
- $\mathbf{Y} \in \mathbb{R}^{m \times d}$  is the observed matrix containing  $d$  observed variables associated with each factor instance. In practice observed variables may be used in more than one factor; for simplified notation they would be indexed redundantly in this matrix.

**Definition 16** A multi-clustering inference state associated with an MCFG structure as above is a tuple

$$\mathcal{MCS} : \langle \mathbf{x}, \mathbf{c}, \mathbf{W} \rangle$$

consisting of,

- $\mathbf{x} \in \{1 \dots k\}^n$  is the assignment state designating current cluster memberships for each variable.
- $\mathbf{c} \in \{1 \dots k^d\}^m = \zeta(\mathbf{x})$  is the configuration state enumerating the configuration of variables associated with each factor.
- $\mathbf{W} \in \mathbb{R}^{k^d \times d}$  is the parameter state containing  $d$  parameters for each possible factor domain configuration.

Figure 5.5 provides a visual representation of the graph and inference state of an MCFG. Note that, as in section 2.3.4, I have assumed for simplicity uniform variable domains sized  $k$  and factor domains (observed, unobserved and parameter) sized  $d$ . In real applications we can partition each of the matrices and vectors above to allow several types of variables and factors. Additionally,  $k$  may be infinite in theory, and vary in practice during inference, as in iREM (algorithm 9). I address representation of  $\mathbf{W}$  in the following section.

### 5.5.2 Inference

The generalized REM algorithm (9) relies on the property of clustering distributions that the parameters associated with one cluster affect only the likelihood of belonging to that cluster. As a result, it is possible to maintain a priority queue of the most important updates by updating only the likelihoods associated with transitions to the previous and new cluster assignment. In factor

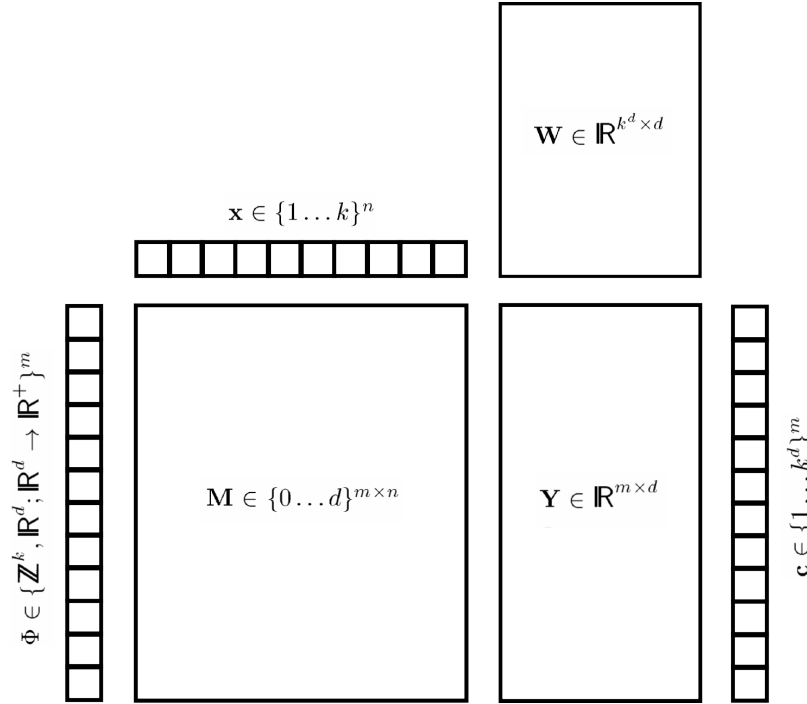


Figure 5.5: Schematic of a multi-clustering factor graph with associated inference state.

graphs, the situation is complicated by the fact that each variable can participate in more than one factor distribution, and that the factor parameters themselves can relate multiple unobserved variables. However, the basic principle that an update requires only sparse recomputation of other queued updates can still apply when the conditions we placed on multi-clustering factors hold.

The Residual Expectation Maximization Belief Propagation (REM-BP) algorithm for multi-clustering factor graphs works, similarly to REM, by maintaining a list of priority queues  $\mathbf{S}$  that help determine the best assignment for each variable, as well as a cross-variable residual queue  $\mathbf{r}$  that speeds determination of the most beneficial update available. At each iteration, the update providing the maximum improvement is executed, factor parameters are updated to maximize the likelihood of the new assignments, and queues must be updated to permit selection of the following update. During this process, it is possible to exploit graph and factor structures by performing updates in the following phases.

- *Parameter Phase.* Due to factor structure, the only parameters which must be updated are those associated with the former configuration and new configuration for each factor with the reassigned variable in its domain.

- *Graphical Phase.* Due to graphical structure, only those score queues associated with variables in the same factor domains as the updated variable must be updated in their entirety.
- *Configuration Phase.* Due to parameter updates, it is necessary to also update each entry in  $\mathbf{S}$  associated with an assignment that would lead to either the original or new configurations in which the updated variable participates.

In the pathological scenario where a variable participates in every factor in the graph, the updates above could involve a recomputation of all assignment likelihoods. However, when the degree of each variable is bounded, and especially if the configurations are varied, it is possible to maintain  $\mathbf{S}$  with very few updates.

To enumerate the variable updates itemized above, I define some helper functions that explore graph and configuration structure. Let the set-valued *neighbor* function

$$N(M, i) = \text{SORTED}(\{j : M_{ij} \neq 0\})$$

return the indices of columns in which a matrix has nonzero entries in ascending order by value. In most sparse matrix representations, and particularly in graphs with regular generators,  $N$  can be iterated through very efficiently.

Next, let the *substitution* function

$$B(c, i, x) = \zeta(\zeta^{-1}(c)_{<i}, x, \zeta^{-1}(c)_{>i})$$

return the index of the configuration achieved by substituting  $x$  as the value for the  $i$ 'th variable in configuration  $c$ . Finally, let the integer *adjacency* function be defined:

$$A(c, i) = \{c' : \exists x \in \{1 \dots k\} \mid B(c', i, x) = c\}$$

In other words, given a configuration and a position index, return all configurations reachable by substitution of a single variable value.

This notation is used in the pseudocode for the REM-BP algorithm (13), given below, with implementation discussed following.

---

**Algorithm 13** Residual Expectation Maximization Belief Propagation (REM-BP)

---

**Require:**  $\Phi, \mathbf{M}, \mathbf{Y}$  {Multi-clustering factor graph}**Ensure:**  $\mathbf{x}, \mathbf{c}, \mathbf{W}$  {locally maximal multi-clustering inference state} $k \leftarrow 0$  {number of inferred clusters} $\mathbf{x} \leftarrow \{0\}^n, \mathbf{W} \leftarrow \{0\}^{\infty \times d}, \mathbf{c} \leftarrow \{0\}^m$  $\mathbf{l} \leftarrow \{0\}^n$  {current variable likelihoods} $\mathbf{S}, \mathbf{r} \leftarrow \emptyset$  {keyed priority queues for scores and residuals} $\epsilon \leftarrow \min_j \prod_{i \in N(\mathbf{M}^t, j)} \phi_i(0 \mid \mathbf{y}_j)$  {minimum initial assignment likelihood}**for**  $i = 1$  to  $n$  **do** $l_i \leftarrow \epsilon$  $UPDATE(i, 0)$  {queue assignment to novel value}**end for****repeat** $i \leftarrow POP(\mathbf{r})$  {draw variable with greatest update priority} $v \leftarrow x_j, v' \leftarrow PEEK(\mathbf{S}_j)$  {get old and new values}**if**  $v' = 0$  **then** $k \leftarrow k + 1, v' \leftarrow k$  {create a novel value}**end if****if**  $v \neq v'$  **then** $x_i \leftarrow v'$  $\mathbf{u} \leftarrow \emptyset$  {set of variable, assignment pairs needing likelihood recomputation}**for**  $j \in N(\mathbf{M}^t, i)$  **do** $p \leftarrow c_j, p' \leftarrow B(c_j, M_{ji}, v'), c_j \leftarrow p'$  {parameter updates} $\mathbf{w}_p \leftarrow \operatorname{argmax}_{\mathbf{w}} \prod_{j': c_j = p} \phi_{j'}(p \mid \mathbf{y}_{j'}, \mathbf{w}), \mathbf{w}_{p'} \leftarrow \operatorname{argmax}_{\mathbf{w}} \prod_{j': c_j = p'} \phi_{j'}(p' \mid \mathbf{y}_{j'}, \mathbf{w})$  $\mathbf{u} \leftarrow \mathbf{u} \cup \{\forall_{v'' \in \{1 \dots k\}, i' \in N(\mathbf{M}, j)} (i', v'')\}$  {graphical updates needed} $\mathbf{u} \leftarrow \mathbf{u} \cup \{\forall_{p'' \in \{p, p'\}, b \in \{1 \dots d\}, c \in A(p'', c), j': c_{j'} = p''} (N(\mathbf{M}, j')_b, \zeta^{-1}(p'')_b)\}$  {configuration updates needed}**end for****for**  $(i', v) \in \mathbf{u}$  **do** $UPDATE(i', v)$  {recompute likelihood of assignment to  $v$ }**end for****end if****until**  $|\mathbf{r}| = 0$ 

---

---

**Algorithm 14** UPDATE Subroutine

---

**Require:**  $i, v$  {variable and value to update}

$$\hat{l} \leftarrow \prod_{j \in N(\mathbf{M}^t, i)} \phi_j(v \mid \mathbf{y}_j, \mathbf{w}_v)$$

 $INSERT(\mathbf{s}_i, v, \hat{l})$  {update likelihood of assignment} $INSERT(\mathbf{r}, i, \frac{PEEK(\mathbf{s}_i)}{l_i})$  {update priority of assignment to most likely value}

---

Many of the implementation details for REM-BP are similar to those for the original iREM algorithm (*e.g.* the use of keyed heap structures). The multi-level iteration required to enumerate configuration-based updates can be assisted by maintaining a hash from configuration indices to factor indices whose domains are so configured. If factor parameters admit incremental updates (as, for example, exponential family likelihoods do), parameters can be optimized without considering observed variables other than those associated with the factor instance being updated. Variable likelihood updates could be performed with fewer evaluations of potential functions by memoizing factor potentials and only updating those whose configurations or parameters changed in this round. It also improves numerical stability and speed to perform likelihood products in logspace.

### 5.5.3 Empirical performance of REM-BP

To compare REM-BP to asynchronous EMBP with an arbitrary schedule, I considered a simplified map building task, in which the goal was to compute a set of locations and transition probabilities between them rather than infer longer range paths. Figure 5.6 illustrates the associated graph, which is essentially a subset of the path prediction graph used in the previous set of experiments.

The observed variables  $\mathbf{y}$  and hidden variables  $\mathbf{x}$  represent location data and latent location centroids, as in the previous section, with the exception that the observed ship location  $\mathbf{y}_t$  is comprised of only its latitude and longitude. We define the factors as infinite multi-clustering factors as follows.

- $\phi_S(x, \mathbf{y}; \mathbf{w}_x)$ , gives the likelihood of the latitude, longitude pair  $\mathbf{y}$  being emitted by the centroid associated with  $\mathbf{x}_t$ , whose latitude and longitude are the configuration-associated parameters  $\mathbf{w}_x$ . The formula for this factor potential is the Gaussian

$$\phi_S = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{D(\mathbf{y}, \mathbf{w}_x)^2}{2\sigma^2}},$$

where  $\sigma$  is a meta parameter input to the algorithm indicating scale, and  $D$  is the distance between coordinates in a planar projection tangent to the centroid.



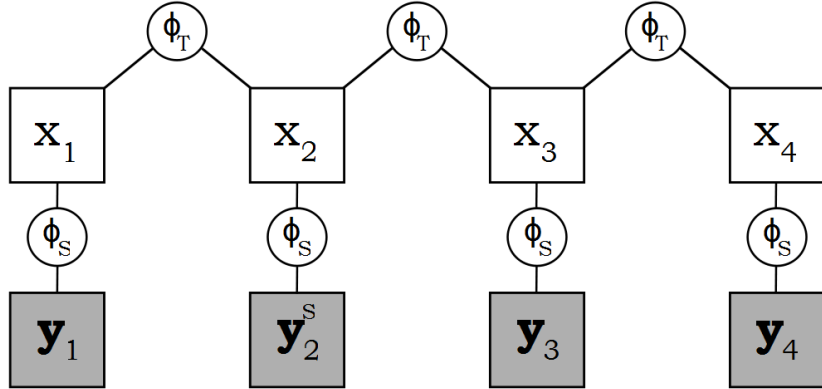


Figure 5.6: REM-BP Experiment Factor Graph

Using a planar projection is highly accurate for short distances, allows the centroid position to be optimized by simple averaging, and allows distance computations to use only a single trigonometric calculation. Using a constant scale on the distances, rather than a centroid specific one, permits a more regular spacing of centroids which creates a more regular transition matrix. The potential associated with a novel cluster is always  $\frac{1}{\sqrt{2\pi\sigma^2}}$ , the probability when a centroid exactly matches a data point. One way to interpret this is to imagine that every possible centroid exists, but there is no need to consider those that are neither optimal for a data point nor associated with other data points.

- $\phi_T(x, x'; w_{\zeta(x,x')})$ , gives the likelihood of a transition from  $x$  to  $x'$ . A single parameter is associated with each  $(x, x')$  configuration, equalling the number of such configurations currently observed in the data. The likelihood returned is the Dirichelet likelihood

$$\phi_T = \frac{1 + \alpha + w_{\zeta(x,x')}}{1 + \alpha + n},$$

where  $\alpha$  is a meta-parameter indicating the expected sparseness of the transition matrix, which indirectly enforces the sparseness of the state space as well.

Instantiating the above factor graph for the AIS data set, I trained factor parameters using the REMBP algorithm, and using the round robin expectation maximization algorithm from the

previous section. Figure 5.7 compares the convergence profile in terms of wall-clock time respectively by graphing the cumulative in log-likelihood achieved on sampled iterations of the algorithm. REM-BP far exceeds a round-robin schedule in quickly coming close, in terms of overall likelihood, to its final configuration. The initial rapid increase in likelihood in the round-robin schedule is due to initial assignments being cataloged for each variable. Visual inspection of the learned transition graphs figures 5.8, shows that the solution reached by REM-BP also appears much richer than that reached previously by the round-robin schedule (depicted in the previous section).

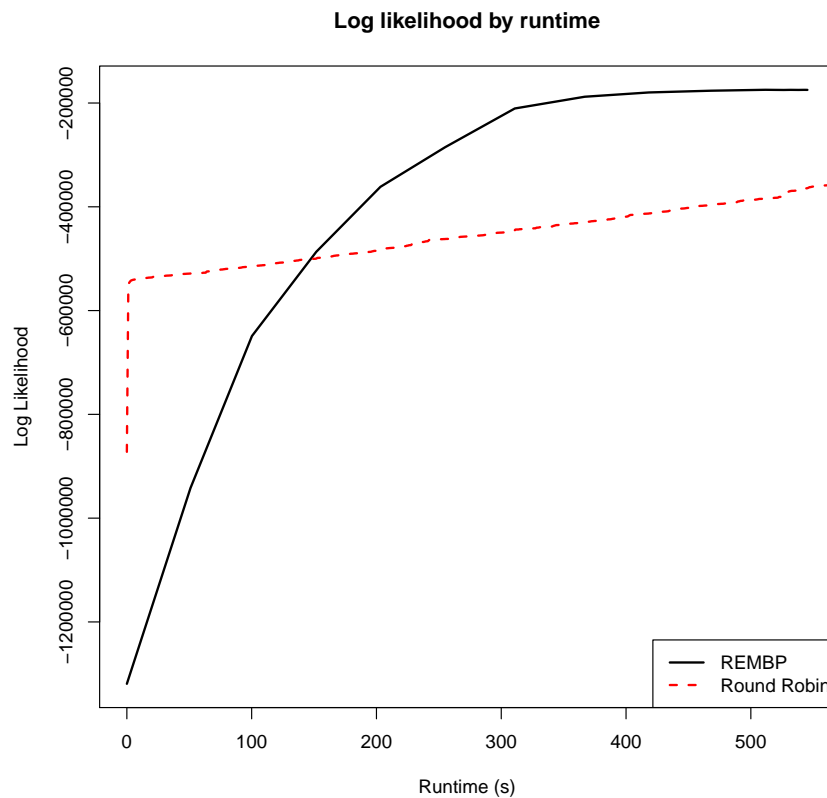


Figure 5.7: Convergence profiles for REM-BP and round robin algorithms.

## 5.6 Discussion and Future Work

I first presented synchronous and asynchronous expectation maximization algorithms for unsupervised learning in factor graphs. I used these algorithms with a factor graph interpreting AIS data in order to simultaneously detect a map, hidden plans, and transition frequencies between plans.

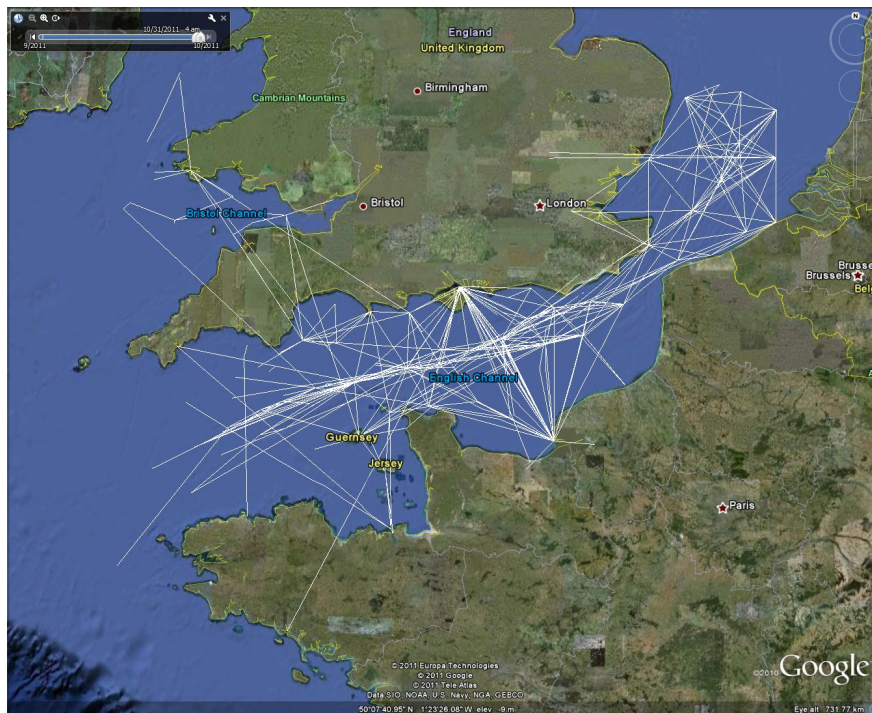


Figure 5.8: REM-BP learned model

To my knowledge, this was the first work applying general purpose unsupervised algorithms in graphical models to conduct learning with real data. I used the learned models to make projections of portfalls for ships in motion. Although these preliminary results were not accurate enough for real-world application, both prediction accuracy and direct inspection of learned locations and transition probabilities suggested that a reasonable model was being inferred. The more significant result is that the asynchronous method significantly outperformed our synchronous method in terms of both convergence rate and probability of achieving high accuracy in portfall prediction.

By animating the algorithmic iterations within Google Earth, I was able to see that that, consistently, some regions of the map stabilized quickly in terms of cluster locations and transition probabilities while others were left to slowly improve over many iterations. As a result, the pre-computed schedule for variable updates caused significant computation. In the  $\alpha$ -FOG experiments of the previous chapter, and in previous work on supervised PGMs by Elidan *et. al* [38], it has been shown that algorithms which prioritize updates that strongly improve the model converge faster and often to better local maxima. To translate this concept into unsupervised inference for factor graphs, I proposed a subclass of *multi-clustering factor graphs* with additional restrictions on factors. The additional restrictions permit efficient implementation of a residual prioritizing

algorithm, REM-BP. Experiments on a simplified graph for AIS data confirmed the hypothesis that a residual-prioritized schedule would significantly improve performance, and even made it possible to conduct learning in a graph with a large and initially undetermined number of latent categories.

I foresee two major challenges in improving performance on the AIS graph. The first is the need to provide greater precision in modeling of traffic patterns in more populated areas. For example, accurately predicting behavior of ships may require associating them with paths and waypoints with effective diameters of a few kilometers, while 10 or even 100km resolution may be sufficient to identify the trajectory of those in the open sea. One way to accomplish this would be via multi-scale models, in which locations and trajectories were organized into a hierarchy of locations. Olson *et al.* [98] has begun exploring a version of this problem in which waypoints and arcs identified at a high spatial resolution must be summarized at a lower one. In practice, it may be more scalable to attempt the problem in reverse, as with the progressive precision created as a spatial *kd*-tree [99] is populated.

Solving the variable-granularity problem might actually exacerbate the second challenge, which is scalability. Experiments conducted with an unknown number of waypoints quickly generated an intractable quantity when high resolutions were used. One major reason for this is the poor fit between a Gaussian surface centroid and the abstract navigational coordinates of a ship. On long shipping lane stretches, our algorithm detected a series of discrete states, rather than the single transition between one waypoint and a distant one which was probably determined by the human process we are trying to model. In addition to requiring unnecessary computation, this complicates the prediction task and most likely degrades performance. Identifying the long-distance transition made by ships going all the way through the channel as a single decision would allow us to recognize it as quite distinct from the path taken by a ferry going from one side to the other, though they may both cross the same point.

Although it is not reported in this chapter, I attempted to solve this problem by adapting the filament-based method of [43] to the the MCFG framework. The challenge I proved unable to solve was how to sensibly initialize the filament network within my unsupervised context. Gordon *et al.* identified this as a difficulty within their domain as well, and reported experiments based on a hand-drawn initialization of this graph. A model of this kind would work well hand-in hand with future work incorporating an inflow of data over time, coupled with human input to aide the comprehension process.

Ultimately, unsupervised algorithms and future improvements have the potential to support a variety of applications where labeled data is unavailable, but they do not replace the need for *a*

*priori* reasoning. The structure of the model, determined in advance, encodes beliefs that constrain the learning problem, opening the door to incorporate expert knowledge but also to erroneous assumptions. Proper model selection in this context may take on a special importance, indicating another line future research.



## Chapter 6

# Spatially embedded link prediction

### 6.1 Problem Definition

Describing and predicting change in networks over time has been an active topic within the social network analysis community since [92]. Doing so in the context of multi-mode and attributed data are the primary goals of the dynamic network analysis [22] and SIENA modeling frameworks [108]. *Link prediction* refers to a specific task within this topic: given prior snapshots of a network relation and an observed set of nodes at a subsequent timeframe, can we predict which new links have been created in the interval?

In this section, I approach this link prediction task informed by a spatial embedding for a network. When Stanley Milgram set strangers searching for each other in his 1967 small world experiment [84], he armed them with only two pieces of information regarding the name they sought: its city of residence and profession. His design presumed, and results affirmed, that spatial and community cues were sufficient to guide information through a social network, in spite of the exponential variety of structures which the network could theoretically take. Since then, studies have confirmed that spatial models usefully predict network relations, not only for searching individuals in a static social network [60], but for observers predicting missing or nascent links in a sampled or dynamic network [20] [25].

As described in section 2.1, a natural explanation for the spatial and community structure of our interpersonal ties is that they reflect our limitations as individuals. Distance complicates our interactions, so we either move near those we work with and care about, or learn to work with and care about those who are near. A broad class of *inverse-distance models* have been proposed operationalizing this insight [20].

One drawback of inverse-distance models is that they are sensitive to the choice of distance function, which is not always clear even when a clear spatial context exists. Should we sooner expect a New York City resident to have a business partner in Los Angeles or in Cincinnati? One could digress over whether spatial, demographic, or jet-travel time distances are better predictors for this relation, but doing so would ignore that each of these measures can be approximated from the geographic locales we referred to by city name. *Spatial clustering models* utilize this insight by pre-processing data for clusters based on some embedding distance, then deriving cluster-cluster link probabilities that approximate the most appropriate distance measure [98]. This method is less sensitive to the embedding distance, but requires that nodes fall into spatially separable, homogeneous clusters. We might not, for example, be able to distinguish one borough of New York from another, with the different linkage likelihoods that doing so might incur.

To extract value from spatial data in spite of the challenges above, I define a nonparametric *spatial correlation* approach which revolves around the self referential definition, “nodes nearby each other tend to link with other nodes that are nearby each other”. Spatial correlation generalizes inverse distance and spatial clustering models, and so can be considered without knowing *a priori* whether one of the more specific models might apply. To reduce dependence on the spatial distribution of nodes, I propose a nonparametric framework which adapts to overlapping populations with varying densities. Since the algorithms involved in inferring link probability scale poorly with increased data, we provide an estimation algorithm which can be tailored to the data size and computational power available.

Spatial correlation can be generalized to non-geospatial aspects of network data, including continuous embeddings in any dimension and “spaces” described by categorical variables. To validate the approach I examine an evolving network of supply chain relations between publicly traded U.S. companies (described in section 3.2). The company nodes have several rich spatial attributes, including geospatial coordinates of headquarters, the one dimensional space of market capitalizations, and the categorical space of sector relations. I show that inverse distance and spatial clustering models do not fit the dataset, potentially due to the distributed nature of the entities or the potential for complementarity of assets across distant / different firms. I then test the predictive capability of our nonparametric spatial correlation algorithms by conducting experiments in link prediction and missing link detection following [74] and [85].



## 6.2 Classifying spatial embedding models

Network models have been proposed in many forms, from distributions over summary statistics [9] [116] to generative algorithms and simulations [69] [88] [108]. In this paper we focus on models that, for a set of  $n$  nodes, can be expressed as a distribution  $P(\mathbf{M})$  over  $n \times n$  binary matrices  $\mathbf{M}$  for which each entry  $m_{ij}$  indicates the presence or absence of a directed link from node  $i$  to node  $j$ . As prior work on exponential random graph (ERG) models have demonstrated [103] [107], adopting a probabilistic framework allows a single model to be conveniently applied to a variety of tasks, including missing data detection and link prediction.

Since the number of possible realizations of  $\mathbf{M}$  grows exponentially with the number of nodes, models must make simplifying assumptions to keep training and inference tractable. *Attribute-mediated* models accomplish this by introducing a secondary  $n \times m$  matrix  $\mathbf{X}$  containing attribute data. If we assume (or accept the approximation) that each edge is independent of all others given the row-vector attributes of the source and target nodes, we can derive  $P(M)$  by modeling the simpler attribute-dependent distribution.

$$\forall_{i,j,k,l} \quad m_{ij} \perp m_{kl} \mid \mathbf{x}_i, \mathbf{x}_j \quad \rightarrow \quad P(\mathbf{M}|\mathbf{X}) = \prod_{i,j} P(m_{ij}|\mathbf{x}_i, \mathbf{x}_j)$$

The spatial models I consider are a subclass of attribute-based models, in which certain attributes are node coordinates in some space in which the network is embedded. Typically these would be geospatial coordinates such as a latitude / longitude pair, but I also consider more abstract spaces, such as demographic distance or group co-membership. The defining characteristic of an embedding space is its distance function  $D(\mathbf{x}_i, \mathbf{x}_j)$ . So long as  $D$  adheres to the triangle inequality<sup>1</sup>, we can define models that exploit spatial concepts such as density and convexity to simplify link prediction.

Geospatial models for interpersonal social networks are attractive because there exists a convincing web of justifications for their use. First, there is the process argument: most kinds of relationship are more easily maintained at close distance, so bounded rationality will lead them to be developed preferentially. Second, there is the pervasive phenomenon of homophily. The similar perspectives induced by sharing an environment may make closeby relationships more attractive in addition to being simpler. Finally, space is often highly correlated with, and so can act as a proxy for, other kinds of node attributes for which the above arguments (or others) may apply.

The phenomena above are causally agnostic (we can interpret space as either cause or effect), and can perpetuate each other via positive feedback. For example, groups of nodes that form ties

---

<sup>1</sup>  $\forall_{i,j,k} D(\mathbf{x}_i, \mathbf{x}_j) \leq D(\mathbf{x}_i, \mathbf{x}_k) + D(\mathbf{x}_k, \mathbf{x}_j)$

due to some form of demographic homophily may move nearby each other to decrease maintenance costs of those relations, leading to further similarity and greater homophily in the future. Geospatial information is particularly convenient given that it can be easily and cleanly operationalized in many environments, but it may not always be the space for which the above arguments best apply. I show this is the case for our interfirm network, in which a better model is produced by an embedding based on non-physical attributes.

Having adopted homophily as a motivation, I consider only spatial models that are *distance-negative*: nodes at greater distance are less likely to connect, less similar in linking behavior, *et cetera*. This bias is rooted deeply in the models and algorithms compared here, which can be seen by considering an inverse distance ( $D' = \frac{1}{D}$  or  $D' = -D$ ) as an approach at creating a distance-positive model. The inverted function must break the triangle inequality on which my model implicitly relies. Similar assumptions of link transitivity are built into many relational network models, though they are not always explicitly discussed.

Table 6.1 illustrates archtypical networks predicted by three increasingly general classes of spatial model: inverse distance, spatial clustering, and spatial correlation. The remainder of this section introduces the three categories and discusses previous work on the first inverse distance and spatial models. In section 6.3 I give a concrete instantiation of the spatial correlation class.

*Inverse distance* models operationalize the stylized fact, “nodes at greater distance are less likely to connect”. When applicable, this deceptively simple intuition is sufficient to make spatial information an incredibly potent description of the network. [21] shows that even in a near trivial model, in which link probability falls off abruptly at some critical distance, entropy of the distribution  $P(\mathbf{M})$  is drastically reduced when conditioned on the spatial data  $\mathbf{X}$ . Similarly, [60] showed that when agent nodes with only local knowledge of graph structure are linked on a spatial lattice with few “long” links, spatial information about a target recipient is sufficient to permit routing of information along a near-optimal path.

The example equation given in table 6.1 is a slightly more practical model, which can be fit to data. Link likelihood decreases exponentially with distance at a rate determined by the single parameter  $\lambda$ , which can be trained using maximum likelihood estimation. A trained inverse distance model turned out to be capture the majority of relational information in a prediction similarity score computed in [28].

[20] explored the predictive power of a variety of inverse distance models on both real and simulated data, and in doing so identified the following consideration when selecting distance functions. Inverse distance models are not robust to factors which might distort the effect of distance in various scales and locales. For example, from the point of view of many social processes, London and New

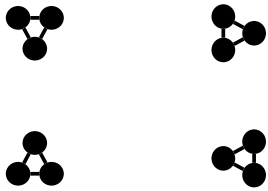

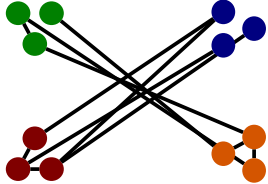

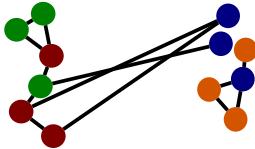

Category	$L(m_{ij} \mathbf{x}_i, \mathbf{x}_j)$	2d Example	1d Example
Inverse Distance	$e^{-\frac{D(\mathbf{x}_i, \mathbf{x}_j)}{\lambda}}$		
Spatial Cluster	$\frac{\sum_{k \in C(\mathbf{x}_i)} \sum_{l \in C(\mathbf{x}_j)} m_{kl}}{ C(\mathbf{x}_i)   C(\mathbf{x}_j) }$		
Spatial Correlation	$\frac{\sum_{k, l \neq i, j} K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right) m_{kl}}{\sum_{k, l \neq i, j} K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)}$		

Table 6.1: Typical likelihood functions and graphs of spatial network model categories.

York are significantly “closer” to each other than either are to the islands that lie between them on the flight route. This phenomenon is demonstrated in figure 6.1, which estimates the distribution of distances between linked companies’ headquarters in our interfirm network.

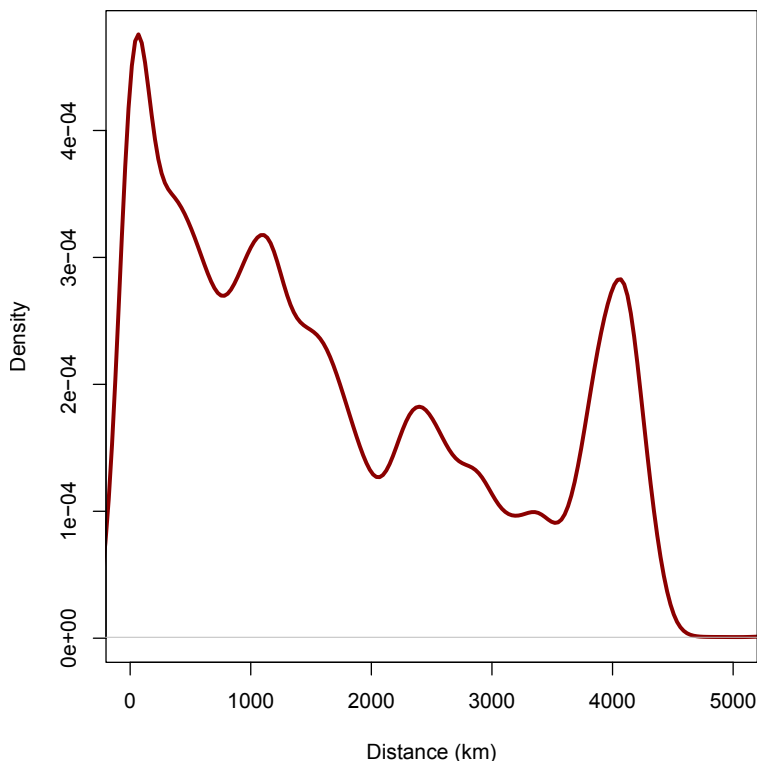


Figure 6.1: Kernel estimation of density of distances between linked companies in the Revere dataset ( $n = 112985$ , bandwidth = 122.1 km). The various maxima correspond to distances between major cities, demonstrating how social context can convolute the generally accurate notion that more distant links are less probable.

There are several interesting ways to interpret the distance-conditioned density function shown by the figure. On one hand, it demonstrates clearly that no monotonic function of distance captures the density relation visible in the data. While there is clearly a distance correlated effect, it is clearly confounded with a coastal or clustering effect, with the distances between major metropolitan centers causing clear peaks on the graph. However, it would be no solution to adopt the nonparametric density function (or any parametric representation thereof) for generalized prediction, as we cannot sanely believe that a 4000km separation would dramatically increase the link probability for two companies not located in New York and Los Angeles. Instead, the graph shows us that distance functions alone do not capture the spatial effects in our data, and that efforts to

regress a distance-based function alone would be highly biased by the unmodeled effects.

With domain knowledge and experimentation it may be possible to select a distance function well-suited to a specific network. However, if an appropriate function is not known *a priori* or is difficult to compute, the utility of inverse distance models can be compromised. In general, if the distribution of dyad density over distance seems non-monotonic, inverse distance is inappropriate. As a rule of thumb, if there is evidence that PDF of link distance is multimodal, either spatial clustering models or spatial correlation models as defined below may be a more appropriate fit.

*Spatial clustering* models introduce a pre-processing clustering step, represented by the  $C$  function in the sample equation in table 6.1, which mediates between the distance function and link probabilities. Cluster assignments can be thought of latent group attribute which influences both the link matrix  $\mathbf{M}$  and spatial distribution of nodes  $\mathbf{X}$ . As a result, the assumption that “nearby” nodes have similar linking behavior need only be true as far as cluster boundaries extend. This allows the model to adapt to more complex spatial behavior or a distorted distance function, for example correctly encoding the New York - Los Angeles relationship pattern described above.

Spatial clustering is a challenging problem with its own literature outside our framing purpose of predicting network structure. Some clustering methodologies are described in sections 2.2 and 4.1. Among the challenges involved is determining a tradeoff between cluster size and cluster cohesion. If cluster cohesion is maximized, clusters will be too small to be useful as a summary of the underlying spatial data. At the other extreme, including overly distant points in the same cluster diminishes the validity of that cluster as a summary. [98] proposed a framework for selecting an optimized tradeoff between these extremes in the context of spatially embedded networks, using entropy-reduction criteria from information theory.

Regardless of the clustering method and parameters selected, results depend heavily on  $\mathbf{X}$  having a distribution that is well represented by the cluster model under the distance function selected. Nodes that are on boundaries between clusters may have the wrong information applied when predicting adjacencies. If two highly predictive clusters overlap to the point of being merged, the resulting cluster may have reduced utility in predicting node behavior. Finally, if the distribution of nodes is more uniform than clustered in some regions, cluster boundaries may become extremely arbitrary (in these areas an inverse distance model might be much more appropriate).

Figure 6.2 illustrates how these problems might arise in our interfirm network data. The disparity in density of company headquarters between the northeast and the rest of the country is so great that it may be difficult for an algorithm to find clusters with similar interpretations across the region. Additionally, there exist a large number of “isolated” headquarters which would be grouped by various clustering algorithms.

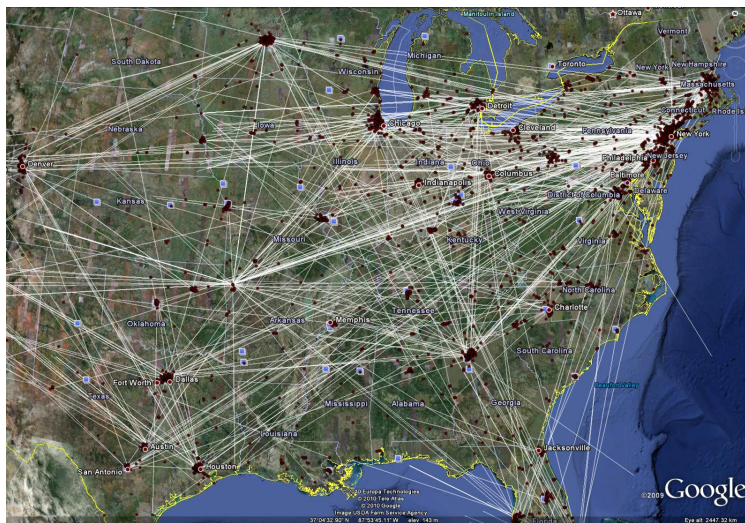


Figure 6.2: Locations and significant supply chain relations between public company headquarters in the Eastern United States.

*Spatial correlation* is my term for a new class of model that integrates the continuity of inverse distance models with the flexibility of clustering models. Rather than individual nodes and their properties being the unit of analysis, I assume that each dyad is similar to other nearby dyads in likelihood of a link observed. ‘Nearby’ in the preceding sentence must be operationalized for pairs of dyads; I propose one such instantiation in section 6.3 and explore it experimentally in section 6.4.

Table 6.1 shows two networks whose properties motivate the spatial correlation model. In the two-dimensional example we show a network where link behavior transitions smoothly as the location of source or target nodes moves, consistent with spatially overlapping or mixed membership node classes. In this network, distance alone is a poor predictor of link probability, and a clustering model would capture only a portion of the predictive power of spatial positioning. In the one-dimensional example, we show that even on a single axis, different spatial regions might be better captured by distance based or cluster based models. Because spatial correlation generalizes both model types and is adaptive by region, it can be applied across the entire space without knowing in advance which model class is preferable.

Spatial correlation joins previous relational models targeting increased expressiveness. [35], [31] and [6] are examples of efforts to accomplish this in the group detection domain, and [111] introduces a general framework for conducting inference of attributes and relations in a wide variety of structured data. More recently, nonparametric methods have emerged as a way of reducing requirements for *a priori* knowledge. One example related to the approach found in this paper is

that of [85], which attempts to identify a hidden set of features which underly node attributes and predict link behavior. The method could be compared to identifying an optimized distance function for a space of categorical attributes, and may make a powerful complement for the continuous-space focused approach in this paper.

Nonparametric methods offer an alternative to practitioners that, during and exploratory analysis, might be expected to mis-specify a parametric model, or adopt one with unwanted bias. They also provide a practical solution for scenarios where analysis where the proper parameterization is not known in advance, and there will be no opportunity to determine one. The cost of this is a much higher data (and thus computational) requirement, as we must algorithmically determine the entire shapes of functions rather than a few control parameters.

### 6.3 A kernel-based spatial correlation model

One way to describe a spatially correlated relation is to say that, as the locations of source and target nodes in one dyad approach those of another, the probabilities of observing a link on each dyad converge.

$$(6.1) \quad \forall_{i,j,k,l} \lim_{D(\mathbf{x}_i, \mathbf{x}_k) \rightarrow 0} \lim_{D(\mathbf{x}_j, \mathbf{x}_l) \rightarrow 0} P(m_{ij}) = P(m_{kl})$$

What distinguishes one model from another is the way in which the above convergence occurs. I begin with the Bayesian approach that link probabilities in both training and test data are themselves random variables whose prior expectation is the overall density of the training graph

$$m_{ij}, m'_{ij} \sim \text{Bernoulli}(p_{ij})$$

$$p_{ij} \sim \text{Beta} \left( \frac{\sum_{k,l} m'_{kl}}{n * (n - 1)}, 1 \right)$$

To describe the approach to the convergence given in equation 6.1, I adapt the concept of a kernel function from nonparametric density estimation techniques. A *dyadic spatial kernel* function  $K : \mathbb{R}^2 \rightarrow \mathbb{R}$  gives the mutual of information of linkage probabilities on two dyads as a function of the spatial distances between the corresponding source and target nodes from each pair. A bandwidth parameter  $\lambda$  is used to adjust the scale at which the kernel operates.

$$(6.2) \quad I(p_{ij}; p_{kl}) = K \left( \frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda} \right)$$

The methods in this paper require the following kernel properties, which simplify inference of  $p_{ij}$  and estimation of the bandwidth  $\lambda$ .

- *Monotonicity:*  $K(0, 0) = 1$ ,  $K(\infty, \infty) = 0$ ,  $\frac{\delta}{\delta d_1} K(d_1, d_2) < 0$ ,  $\frac{\delta}{\delta d_2} K(d_1, d_2) < 0$
- *Symmetry:*  $K(d_1, d_2) = K(d_2, d_1)$
- *Differentiability:*  $\frac{\delta}{\delta \lambda} K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)$  has a closed, readily computable form.

I compare three kernel functions fitting these criteria. Under the *Gaussian dyadic kernel*, mutual information of link probabilities is inversely proportional to the sum of squared distances between source nodes and target nodes in the two dyads.

$$(6.3) \quad K_G\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right) = \exp - \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)^2 + D(\mathbf{x}_j, \mathbf{x}_l)^2}{2\lambda^2}\right)$$

$$(6.4) \quad \frac{\delta}{\delta \lambda} K_G = \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)^2 + D(\mathbf{x}_j, \mathbf{x}_l)^2}{\lambda^3}\right) \exp - \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)^2 + D(\mathbf{x}_j, \mathbf{x}_l)^2}{2\lambda^2}\right)$$

Under the *exponential dyadic kernel* mutual information is inversely proportional to the product of source and target distance.

$$(6.5) \quad K_E\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right) = \exp - \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k) * D(\mathbf{x}_j, \mathbf{x}_l)}{2\lambda^2}\right)$$

$$(6.6) \quad \frac{\delta}{\delta \lambda} K_E = \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k) * D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda^3}\right) \exp - \left(\frac{D(\mathbf{x}_i, \mathbf{x}_k) * D(\mathbf{x}_j, \mathbf{x}_l)}{2\lambda^2}\right)$$

Finally, I consider a kernel whose role is to combine information from multiple kernels and distance functions. A *product dyadic kernel* is constructed by multiplying mutual information values returned by a set of factor kernels, each with their own associated distance functions and  $\lambda$  values. In section 6.4 I present results showing that a product kernel provides better prediction accuracy than any singular kernel in our interfirm network dataset.

### 6.3.1 Inference and training for spatial correlation

Having specified a spatial structure of mutual information, we can view each training sample  $m'_{ij}$  as an outcome containing information about not only  $p_{ij}$  but all other (and especially nearby) link likelihoods. We can then estimate marginal probabilities for each  $m_{ij}$  in the test data by iterating over the data in the training set. I chose the conjugate prior Beta distribution for  $p_{ij}$  so that its



posterior expectation after observing training data is essentially a weighted average of training link values, where the weights are determined by the kernel function.

$$(6.7) \quad P(m_{ij} \mid \mathbf{x}_i, \mathbf{x}_j) = E(p_{ij} \mid M', \mathbf{x}_i, \mathbf{x}_j, \lambda) = \frac{\frac{\sum_{k,l} m'_{kl}}{n*(n-1)} + \sum_{k,l} m'_{kl} * K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)}{1 + \sum_{k,l} K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)}$$

Because we directly inspect training data, our inference equation contains only one unknown parameter, the bandwidth  $\lambda$ . Bandwidth can be intuitively interpreted as the scale of the distance function, but in my model it plays an important secondary role in determining relative strength of the prior. As bandwidth approaches zero, the density prior and the presence or absence of a spatially identical dyad in the training data become the sole determinants of link probability. As bandwidth approaches infinity, the weighting of all dyads in the training dataset become equal, and the link probability converges once again toward the training graph's density. At each of these extremes we have ignored almost all spatial information, so we must instead search for a value of  $\lambda$  which utilizes  $\mathbf{X}$  optimally.

Optimizing  $\lambda$  can be accomplished by maximizing the likelihood of the training data. Since the probability of any particular graph realization  $\mathbf{M}'$  is vanishingly small, the optimization is best conducted in log-space for numerical stability.

$$(6.8) \quad \bar{\lambda} = \underset{\lambda}{\operatorname{argmax}} P(\mathbf{M}' \mid \mathbf{X}, \lambda)$$

$$(6.9) \quad = \underset{\lambda}{\operatorname{argmax}} \log P(\mathbf{M}' \mid \mathbf{X}, \lambda)$$

$$(6.10) \quad 0 = \frac{\delta}{\delta \lambda} \log P(\mathbf{M}' \mid \mathbf{X}, \bar{\lambda})$$

$$(6.11) \quad = \frac{\delta}{\delta \lambda} \sum_{i,j} \log P(m_{ij} \mid \mathbf{X}, \bar{\lambda})$$

$$(6.12) \quad = \sum_{i,j} \frac{P'(m_{ij} \mid \mathbf{X}, \bar{\lambda})}{P(m_{ij} \mid \mathbf{X}, \bar{\lambda})}$$

In general there is no closed form solution for  $\bar{\lambda}$  satisfying the above, but I have chosen kernels such that both the marginal distributions and their derivatives with respect to  $\lambda$  are available and convex. As a result, we can choose from a variety numerical methods to approximate  $\bar{\lambda}$ . For the experiments reported in this paper, we conducted an initial bound search to find an interval on  $\lambda$  that enclosed the optimum, and then used binary search to reach an arbitrary precision. We chose this to ensure a consistent level of accuracy while bounding the number of search iterations

in our experiments. Gradient descent methods might provide a more efficient solution in practical applications.

### 6.3.2 Fast Approximation

From equation 6.7, we can see that computing the marginal probability of even a single link requires iterating through every dyad in the training data, resulting in temporal complexity  $O(n^2)$ . To calculate AUC, we must compute the marginals for every dyad, at a complexity of  $O(n^4)$ . Estimation of  $\lambda$  is more expensive still, since it must complete the entire  $O(n^4)$  inference loop each time the derivative is calculated on an unknown number of iterations. Poor scalability is a common drawback to nonparametric models, and this one is no exception. Each time period of our interfirm network contain over 1000 nodes, making exact inference to slow for us to compute a wide range of experimental configurations.

One potentially wasteful aspect of the precise inference of  $P(m_{ij})$  is that it requires computation time incorporating information from dyads between nodes very distant from  $i$  and  $j$ , which contribute minutely to the local distribution. A sensible approximation would iterate only over nearby dyads for which  $K$  is high. We cannot actually consider all prospective dyads without incurring the same  $O(n^2)$  cost as the precise method. However, the monotonicity of our kernel functions allows us to find a population of high- $K$  dyads by pairing nodes in the local neighborhoods of  $i$  and  $j$  respectively. Choosing a size  $s$ , we can determine local neighborhoods  $N(i)$  and  $N(j)$  by sorting potential neighbors in  $O(n \log n)$  time. Approximate inference is then accomplished by iterating over neighborhood pairs only, for a total cost  $O(n \log n + s^2)$

$$(6.13) \quad \bar{P}(m_{ij} \mid \mathbf{x}_i, \mathbf{x}_j) = \frac{\frac{\sum_{k,l} m'_{kl}}{n*(n-1)} + \sum_{k \in N(i)} \sum_{l \in N(j)} m'_{kl} * K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)}{1 + \sum_{k \in N(i)} \sum_{l \in N(j)} K\left(\frac{D(\mathbf{x}_i, \mathbf{x}_k)}{\lambda}, \frac{D(\mathbf{x}_j, \mathbf{x}_l)}{\lambda}\right)}$$

In practice, I calculate the entire neighborhood function  $N$  prior to training ( $O(n^2 \log n)$ ), and store it at a spatial cost  $O(ns)$ . As a result the total cost of training is reduced to  $O(n^2 \log n + in^2 s^2)$ , where  $i$  is the number of iterations required to maximize  $\lambda$ . Afterwards, the marginal cost of inference of a link probability between two known nodes is  $O(s^2)$ . A query regarding a node not in the training set, such as that of a company entering the network in the second timeframe of the link prediction task, must still compute the neighborhood of that new node.

The product kernel described in the preceding section poses a special challenge with respect to this optimization technique, because each factor kernel may apply a different distance function and therefore a different local neighborhood for each node. To select a single neighborhood, I union the

neighborhoods for each specific distance function. If the result is larger than  $s$ , we select the top  $s$  as those with the smallest bandwidth adjusted sum of distances,  $\sum_D \frac{D(i,k)}{\lambda_D}$ .

Several more trivial implementation tricks had significant impact on our algorithm's speed. Memoizing graph density and expensive distance calculations such as great circle distance between latitude and longitude performance greatly accelerated both neighborhood computation and kernel computation. Inference can also be parallelized trivially by accumulating different blocks of the summations in the numerator and denominator of equation 6.7 in different threads.

### 6.3.3 Distances

In our experiments I consider three distances derived from the company attributes of our interfirm network. They are selected in part with our expectation that they contain information regarding the linkage patterns in the network, and in part because they demonstrate three types of distance function which can be applied via the method above.

The first, *geospatial distance* ( $D_{\text{GEO}}$ ), is derived from the addresses of company headquarters provided in company filings<sup>2</sup>. I applied a geocoder [3] to determine the latitude and longitude of the address. The distance between two points is the great circle difference, measuring the length of the minimal path along the surface of the Earth. The bandwidth computed for this distance function for either kernel function corresponds to the distance at which correlation in link likelihood drops by a certain amount.

$$D_{\text{GEO}}(\mathbf{x}, \mathbf{y}) = 6371 \text{ km} * 2 \arcsin \left( \sqrt{\sin^2\left(\frac{x_{\text{lat}} - y_{\text{lat}}}{2}\right) + \cos x_{\text{lng}} * \cos y_{\text{lng}} * \sin^2\left(\frac{x_{\text{lng}} - y_{\text{lng}}}{2}\right)} \right)$$

The *capitalization distance*  $D_{\text{CAP}}$  measures separation in an abstract one-dimensional space where each company's coordinate is its market capitalization. In the dataset market capitalization follows roughly a power law distribution, with a large percentage of total market capitalization being possessed by a few companies whose values are several orders of magnitude larger than the average. This prompted some consideration of the distance function. Since our kernel function imposes symmetry in information content across the distance function, adopting a simple arithmetic distance would imply that linking behavior is as similar between a \$1 million company and a \$100

---

<sup>2</sup>Headquarters location is a far from perfect proxy for the geographic activities of firms, which may have varying levels of geographic diffuseness. For example, saying that US Steel's business activities are concentrated in Pittsburgh, PA may be more accurate than suggesting that McDonald's Corporation's are focused in Oak Brook, IL. However, in spite of this diversity, many regulatory and economic analysis regimes resort to a pinpoint specification of location when analyzing them.

million company as it is between a \$1 billion company and a \$1.1 billion dollar company. I chose instead to adopt a logistic distance, which assumes equal similarity between companies separated by the same multiple.

$$D_{\text{CAP}}(\mathbf{x}, \mathbf{y}) = \left| \log \frac{y_{\text{cap}}}{x_{\text{cap}}} \right|$$

The *sector distance*  $D_{\text{SEC}}$  is a categorical distance based on the top-level market sector attribute identified in the Revere dataset. Examples correspond to standard industry classifications such as “energy”, “telecommunications”, or “information technology”. The distance function itself is a simple binary check on whether the two companies share a sector.

$$D_{\text{SEC}}(\mathbf{x}, \mathbf{y}) = I(x_{\text{sec}} = y_{\text{sec}})$$

Sector distance demonstrates the degree to which our nonparametric approach can derive an expressive model from relatively simple assumptions.  $D_{\text{SEC}}$  can provide only 3 distinct inputs to a dyadic kernel function.

1. If source sectors and target sectors match for both dyads, shared information will be highest, corresponding to the model “link probability is determined by source and target sector”. In this way the nonparametric model can simulate a block modeling approach.
2. If only source or only target sectors match, sharing will be lower, corresponding to the model “average out degree in the source sector and average in degree on the target sector determine link probability”.
3. The lowest information sharing occurs when neither source or target match, corresponding to the model “link probability is determined by network density”.

The ordering of these models is fixed by the distance function, but their relative weight is determined by the kernel function and the learned bandwidth  $\lambda$ : a low bandwidth gives greatest significant to the pseudo-block modeling performed in the first case, whereas a high one weights the models equally.

## 6.4 Experimental Results

### 6.4.1 Scoring link prediction

[74] introduced a convenient framework for comparing approaches to this problem, in which each is represented as a scoring function responsible for outputting potential of links in order of

descending likelihood. Using the order rather than the scores themselves in evaluating a scoring function avoids issues of comparing the normalization and prediction confidence of various approaches. Under this operationalization, the data mining community has evaluated a wide variety of scoring functions based on attributed data and relational structures [68] [75] [45] [48]<sup>3</sup>.

The probabilistic framework adopted above for spatial models fits naturally into this paradigm, since the likelihood  $L(m_{ij})$  is itself a scoring function. For notational convenience, we assume that the nodes and inter-node distances remain constant across all time periods, and represent changing network structure with the matrix valued, time domain function  $\mathbf{M}(t)$ . The scoring function takes as input a relation matrix  $\mathbf{M}(t)$  and outputs a list of node pairs in descending order of  $P(m_{ij}(t+1))$ .

$$S : \{0, 1\}^{n \times n} \rightarrow \{(i_1, j_1), (i_2, j_2), \dots, (i_{n \times n}, j_{n \times n})\}$$

In some cases scoring functions are evaluated by drawing a fixed number of predictions and measuring prediction accuracy. We instead follow [85] in measuring prediction accuracy via the area under the response curve (AUC), which can be calculated via the formula above. The most intuitive description of this statistic is that it measures how many pairs of dyads in which the first dyad is linked at  $t+1$  and the second is not are output in the correct order by the scoring function.

$$\text{AUC}(S) = \frac{\sum_{\{(i,j), \dots, (k,l)\} \in S(\mathbf{M}(t))} m_{ij}(t+1) * (1 - m_{kl}(t+1))}{|i, j : m_{ij}(t+1) = 1 \mid \mid k, l : m_{kl}(t+1) = 0 \mid}$$

A closely related problem to link prediction is the identification and repair of missing data. Perfectly measuring relations is impossible in many settings and for many relations of interest, and many measures used in analysis of network data are sensitive to even relatively small error rates in sampling. I consider a specific error model in which real links are omitted at a fixed rate  $r$ . We can approach missing data detection via the framework above by interpreting the observed data as a noisy representation of the real data, in the same way that today's network is a noisy conveyor of information about tomorrow's. In the experiments in the following section, I select a rate parameter and construct a sample of the network that randomly omits links with that probability (for convenience we leave the nodeset constant). I then conduct model training with the sampled graph, and calculate AUC by predicting links in the full graph.

---

<sup>3</sup>A direct comparison between these methods is impossible, as they use different types of data in predicting link formation. For example, Gilbert and Karahalios use frequency data for a variety of tie interactions in predicting future tie strength, while Leskovec's method is designed to predict both positive and negative associations. They are grouped here for their similarity in scoring criteria rather than comparable input / output.

To unify notation for link prediction and missing data detection in the following section, I adopt the notation that we are modeling the distribution  $P(\mathbf{M} \mid \mathbf{M}')$ , where  $\mathbf{M}'$  is the training network and  $\mathbf{M}$  the test network.

### 6.4.2 Experimental Setup

I followed a box design, conducting experiments in every combination of the following parameters.

- *Distance* - either geospatial, capitalization or sector, plus product indicating the use of a product kernel to apply all 3.
- *Kernel* - either Gaussian or exponential (in the case of a product distance, all factors use the same kernel function).
- *Task* - Missing data detection with omission rate parameters  $r \in \{5\%, 10\%, 15\%, 20\%\}$ , or link prediction.
- *Neighborhood size* of  $s \in \{15, 20, 25, 30\}$

I ran at least 5 samples of each of the combinations above, keeping track of run times, AUC accuracy, and  $\lambda$  values. We distributed runs evenly between the 12 quarterly snapshots in the missing data task, or 11 adjacent pairs in the case of the link prediction task. All simulations were run on the same desktop PC, which utilized a 2.67 GHz Intel i7 CPU (experiments were single-threaded for consistent runtime analysis) with 6 gb of RAM. The following sections present the results in various aggregations to demonstrate the impacts of various parameters.

### 6.4.3 Impact of Approximation

In general, I expected use of larger neighborhoods to increase both runtime and performance. My experiments confirmed both hypotheses. Figure 6.3 summarizes the distribution of training times for a geospatial distance model. The mean training time grows quadratically with neighborhood size. The two outliers at neighborhood sizes 25 and 30 may be the result of experimental error, caused by an external process on the experimental machine.

Figure 6.4 paints a slightly more complex picture regarding the effect of neighborhood size on prediction accuracy in the missing data experiment. Increasing neighborhood size always improves performance, but to varying degrees depending on the distance function. The market capitalization model was most sensitive to neighborhood size, producing an AUC significantly lower than

the sector based model at small neighborhood sizes, but superior performance with large neighborhoods. This suggests that researchers applying our nonparametric prediction technique should test sensitivity for their distance function before selecting a neighborhood size.

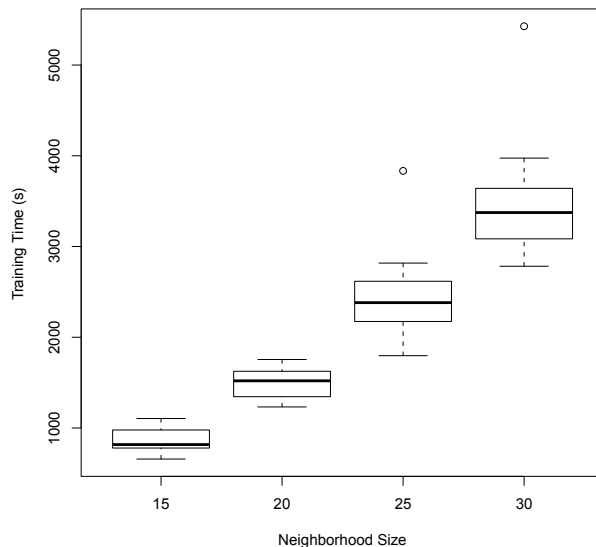


Figure 6.3: Training time by neighborhood size. Samples include exponential and Gaussian kernel functions for Geospatial distance measure on missing data experiments.

#### 6.4.4 Robustness to Missing Data

One would expect that increasing the rate at which edges are deleted from the data would reduce our accuracy in detecting those deletions, since less information is available to train the model. Figure 6.5 illustrates that there is in fact a surprising resilience to such increases. Although all distance functions had a lower mean AUC at 20% than they did at 5%, along the rest of the interval accuracy differentials were within even optimistic error margins ( $\frac{1}{2}$  standard deviation). This stability to omitted data raises hopes that a nonparametric model trained on noisy data might be useful for applications beyond missing data detection. I view this as one of most attractive features of the technique.

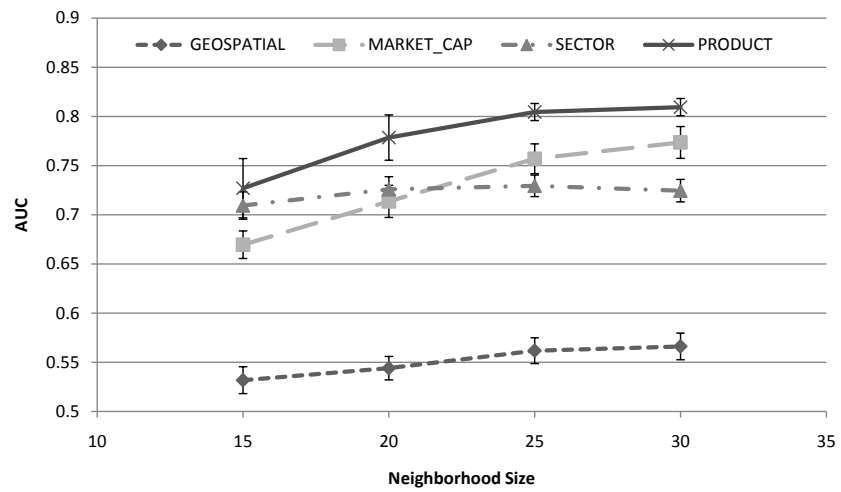


Figure 6.4: Average AUC by neighborhood size on missing data experiment, using 3 distance functions with exponential kernel, and product kernel of same. Error bars are  $\frac{1}{2}$  standard deviation for clear visibility.



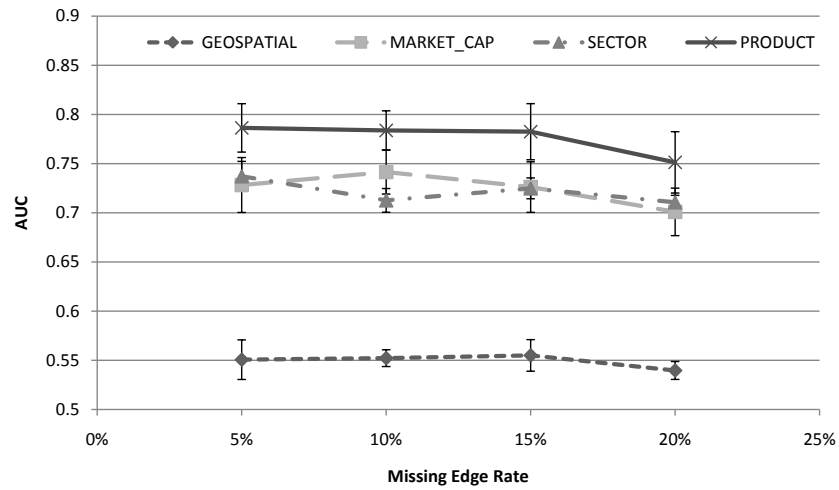


Figure 6.5: Average AUC by edge deletion rate on missing data experiment, using 3 distance functions with exponential kernel, and product kernel of same. Error bars are  $\frac{1}{2}$  standard deviation for clear visibility.

### 6.4.5 Comparative Performance

Table 6.2 summarizes the performance of each distance function and kernel at each of performance task. The first and most general observation is that the average AUC for most methods is competitive with other reported link prediction techniques such as those mentioned in [74] and [49]. The outlying exception is that weak performance of  $D_{\text{GEO}}$  based models on the missing data detection task. This is made more interesting by the same distance function’s strong performance on link prediction task. One possible explanation is that stable core of edges which exist in all time periods is highly entropic with respect to location of source and target nodes, but that new link additions are clustered in the areas best described by the geospatial model.

The choice of kernel seemed to make no difference whatsoever in performance accuracy of each individual distance function, in spite of choosing substantially different optimal bandwidths. This result is consistent with the literature on nonparametric density estimation, where it is known that different kernel functions tend to produce substantially different predictions only for points near the edge of the sampled space. However, usage of the Gaussian kernels proved slightly preferable as the basis for a product kernel, performing almost one standard deviation better than product kernels constructed from exponential factors.

Of the three individual distance functions,  $D_{\text{CAP}}$  performed best across tasks (although, as we noted earlier, this performance declines significantly when smaller neighborhoods are used). However, all three of the individual distances were dominated by the product kernel, which produced an AUC exceeding .8 on every task. Since the product kernel can only aggregate the structural insights encoded in the factors, this suggests that the information garnered by these distance functions does not fully overlap. Overall, this result suggests that a product factor with a “kitchen sink” of distance functions might be an acceptable approach to exploring a new link dataset if sufficient computing power were available.

The relative accuracy of these techniques provides some interesting insights into the structure of our interfirm network. For example, the high AUC scores associated with  $D_{\text{CAP}}$  support the statement “companies of certain specific sizes tend to do business with companies of certain other specific sizes”. Intuitively, we might imagine this relationship exists between the largest and smallest companies in our dataset, as small companies often either require the services of or sell services to a small set of large corporations. We tested this intuition by querying a trained model for the probability of a relationship between a US\$ 20 M company and a US\$ 500 M and received a link probability just under 0.5% – significantly higher than the density which would be the null model.

Distance	Gaussian					Exponential				
	MD		LP		$\bar{\lambda}$	MD		LP		$\bar{\lambda}$
	AUC	$\sigma^2$	AUC	$\sigma^2$		AUC	$\sigma^2$	AUC	$\sigma^2$	
$D_{\text{GEO}}$	.572	.030	.817	.074	2706.054	.571	.029	.817	.074	2435.872
$D_{\text{CAP}}$	.774	.019	.835	.029	1.302	.773	.020	.835	.029	0.167
$D_{\text{SEC}}$	.731	.021	.792	.014	0.396	.729	.016	.788	.014	0.323
<i>Product</i>	.831	.019	.877	.020	N/A	.815	.020	.863	.007	N/A

Table 6.2: Prediction performance by distance and kernel. *Product* refers to product kernel whose factors include each of the above distance / kernel pairings. MD and LP refer to missing data and link prediction experiments respectively. AUC,  $\sigma^2$  and  $\bar{\lambda}$  report average and standard deviation of AUC and average bandwidth parameter in at least 5 trials with the specified distance and kernel. All experiments were performed with neighborhood size  $s = 30$ . Missing data experiments performed with rate  $r = .15$ .

## 6.5 Discussion

This chapter presented a nonparametric method for discovering a range of structural patterns in a graph associated with an embedding space on which a metric distance function can be defined. Models based on this method can be queried for the marginal probability of any particular dyad being linked, which can be the basis for applications where missing links are detected or nascent links are predicted. The method generalizes two previously explored classes of spatial model, those based on inverse distance and those based on spatial clustering of nodes. By defining distance functions based on scalar or categorical attributes, one can also apply the technique to attribute-based linked prediction, an active area of research in several communities.

The motivating setting for the method was analysis of an interfirm network in which I was unsure whether previous results regarding spatially embedded interpersonal social networks could be directly applied. I hypothesized that space, market capitalization, and industrial sector played roles in determining network structure, but wanted to make minimal assumptions about the exact nature of their effects. In addition to interfirm networks, there exist many other “networks of social origin” whose structure may or may not conform to structural models that have proven useful in other settings (*i.e.* homophily, power law or preferential attachment models). Examples include structured data produced by online social mechanisms or distributed sensor mechanisms, which capture previously difficult to detect types of social behavior. In settings like this, where prior assumptions should be minimized, nonparametric methods like this one can play a significant roll.

In the context of our interfirm network, I was able to use geospatial, market capitalization and industrial sector data to produce an ordering on probabilities for missing or predicted links that far exceeded in accuracy one produced by chance or by trivial prediction methods, and was competitive with results from other methods and datasets on the same task. The ability to predict nascent links in the economic network is of potential interest to companies making strategic decisions and investors trying to understand the risk profiles that will be created by new relationships. However, it is doubtful whether the results presented here are accurate enough to be useful in that context.

By comparing results from different distance functions, we can draw some interesting conclusions about the structure and dynamics of the economic network. The fact that a distance measure based on market capitalization outperformed other distance metrics suggests that firm size may be a more important in determining relationships than industrial sector or geographic location. One possible explanation for poor performance of geospatial distance is that globalization has reduced the relevance of spatial distance for US countries. If a similar network were available for American companies during the industrial revolution, or for a modern-day country whose economy is based around physical goods, it would make for an interesting comparative analysis. The significant improvement in performance of the geospatial distance when conducting link prediction rather than missing link detection suggests that the links being formed in the last 6 years are much more spatially structured than the network as a whole. Finally, the fact that a product kernel was able to combine information from each of the distance functions into a superior prediction suggests that at least two of the three distances contain valuable, distinct information about the network structure.

Scalability to large networks remains a major challenge for my algorithm, but which may be overcome by further study. One potential innovation would be to adapt the neighborhood size  $s$  to distribution of nearby nodes, stopping when a certain confidence was reached. The search for local neighborhoods could be made more efficient by application of KD-trees, as is routinely done for  $k$ -nearest-neighbor algorithms [55]. Finally, link probability could be interpolated from a few sampled points rather than being computed from scratch for each new point. This technique would require knowing an appropriate sampling scale in advance, but could detect unusual spatial topologies.

With enough data and computation time, dyadic kernel estimation can fit any pattern that can be described by an inverse distance or spatial clustering model. However, that does not mean it is preferable for all datasets. In addition to the scalability issues described in the preceding paragraph, the dyadic kernel method is dependent on having data points nearby to both sides of each dyad being predicted.

If attempting to generalize characteristics of a relationship from one region to another, it may be

that only inverse density models can be applied. As a process for selecting between the three classes of spatial models I discuss, a practitioner might begin by plotting the frequency of dyads at various distances. If the plot is dominated by a monotonic effect, inverse distance might be an appropriate model. If the plot is multimodal, the data should be further investigated for susceptibility to clustering. This could be done by a visual assessment, or more consistently by training a spatial clustering model and assessing the entropy of the cross-cluster relationship matrix. If neither of these models is a good fit, dyadic kernel density can be used as a more general final step. However, if computation time is not an issue and the region of interest is well covered by sample data, dyadic kernel density (potentially with a product kernel if there are several embedding spaces for the kernel) can be used as a robust first step.

One line of potential furtherance of dyadic kernel methods would be to study the breadth of its applicability, by analyzing a wider range of datasets and comparing results to those produced by parametric methods, particularly those that use relational statistics such as triad completion. A second would be to adapt innovations from other nonparametric techniques to our dyadic setting. One example of this is the concept of an adaptive kernel, where the bandwidth varies by region of the embedding space. An adaptive kernel might reduce the need for *a priori* judgements regarding the distance function, such as our decision to perform the market capitalization differencing in log-space. That type of improvement – one that increases flexibility of a model and diminishes need for prior assumptions regarding a domain, while remaining computationally tractable – is the primary goal of this line of inquiry.



## Chapter 7

# Conclusion: contributions, limitations and themes

### 7.1 The human process that studies human processes

Early in the work that would become this thesis, an analyst at a government agency told me, “Behind every well-posed question I am asked, there are a host of poorly defined ones which are far more important.” A query to predict the destination of a particular ship might be highly relevant for an hour or a day. That *class* of query might be deemed one of many priorities for years. But the long term goal of an organization invested in a human system is to maintain a more general situational awareness, recognizing new entities and phenomena as they become important for understanding dynamics to which the stakeholder is sensitive.

Most human processes that accomplish this adapt a variation of the scientific method to their goal. Individuals or organizations must persistently identify phenomena of interest, reduce them to questions that can be operationalized in data, propose models that relate them, and assess those models for consistency. However, while a natural scientist hopes to arrive at intransient rules that describe their environment – including sociologists, who seek constants across the range of human behaviors – most analysts of human systems must accept that the patterns they discover will continuously be subsumed by new dynamics. These may be due to boundlessness, as entities enter and leave the system and new relationships evolve. It can be caused by competition, as entities adapt to changes brought about by their previous behavior (including interventions by the analyst). Or the change can take place within the interlocutor, whose evolving needs and concerns demand an understanding of different aspects of a system.

Whatever its cause, transience exacts a cost on model development, as the modeller must reserve capacity for adaptation that would otherwise be spent refining established models. In most organizations, adaptability rests, expensively, in the human parts of the process. Computational resources are brought to bear mainly in the final stages of analysis, where entities and relationships of interest have already been identified, and the data within which to operationalize them has been established. The algorithms presented in this thesis were intended to enable computational analysis incrementally earlier in the process, where the basic structural regularities of a system and data of interest have been established, but the exact entities and relationships of interest have yet to be determined (or must be consistently re-determined).

## 7.2 Relaxed grouping in networks

Many human behaviors are difficult to explain without appealing to the notion of communities, indirect associations that can influence patterns of interaction as strongly as pairwise relationships. Communities and their effects have been often verified in targeted experiments and surveys, but detecting them consistently in more general data remains an area of active research. In many ways, communities exemplify the human system challenges listed above: they form, mutate, and devolve without announcement, sometimes exist without the conscious knowledge even of participants, and in some cases react to attempts to label and characterize them.

In chapter 4, I argued that while sociological literature shows that individuals participate in many communities with varying strength of association, and clustering literature has presented a variety of methods for finding overlapping groups, the latter are rarely applied to detection of cohesive communities within social networks. I introduced a latent mixture model, FOG, whose parameters describe the memberships of Fuzzy, Overlapping Groups. The instances generated by the model are multi-entity observations, such as the attendance lists of events. To adapt them to network data, I proposed a random-tree based preprocessor, interpretable as a contagion process simulation, which generates multi-entity relations that sample the structure of the overall graph. I showed that fitting the model with a simple hierarchical clustering algorithm produced interpretable groups consistent with ethnographic observations on two seminal data sets from sociological studies on communities.

The hierarchical clustering method used in my initial validation of FOG had undesirable properties including poor scalability, sub-optimal results due to greedy clustering, and the need to specify the number of latent groups exogenously. I introduced a series of expectation maximization algorithms ameliorating each of the above. Because the algorithms admit no formal guarantees about



runtime or goodness-of-fit, I compared their performance empirically on simulated data in which the true underlying grouping was known. While both the best runtimes and the most optimal fits were achieved by algorithms with foreknowledge of the true number of groups, competitive performance on both criteria was reached by the most advanced algorithm,  $\alpha$ -FOG-residual, which inferred the number of groups from data.

Outside of the community detection task, the algorithmic results explored several general topics in clustering. I compared hard and soft variants of expectation maximization. Based on the hard variant, I formally defined the CEM update schedule for expectation maximization algorithms, a play on the previous ECM classification which updates subsets of the expectations rather than maximizing subsets of the parameters. I defined a class of distributions with an unknown number of latent entities in which CEM updates could be updated efficiently. Finally, I introduced the REM algorithm, which prioritizes important updates in such distributions, and is the basis of the relatively high performance of the  $\alpha$ -FOG-Residual algorithm.

The principle limitations in my work on community detection come from the difficulty in identifying a ground truth against which to validate results. In my experiments I relied on detailed ethnographic surveys and synthetic data, meaning that my work risks participating in confirmation bias or not generalizing to structures in the real world that differ from my synthetic data generation techniques. “True” community structure is inherently hidden, but the FOG algorithms could potentially be further validated in relation to specific prediction tasks with observable results, in which fuzzy, overlapping groups are an intermediate variable. A second area needing further exploration is the detailed impact of network sampling method on discovered groups.

A final limitation is the non-treatment of graphs with multiple edge and node-types. As discussed in text, if the graph is unimodal but multi-relational, it may be sufficient to adapt the link-generation algorithm to generate bimodal link data for multigraphs. In a multi-modal graph, it might be more interesting to consider the construction of multi-modal links, with groups for each type of node contributing to each link.

Notwithstanding these concerns, the empirical results regarding residual-prioritized clustering and the ease with which it can be adapted to distributions for specific domains should make it an appealing approach for a variety of latent entity detection problems.

### 7.3 Plans and places as hidden entities

A second challenging category of human behavior is the organization of their movement patterns through space. A variety of constraints and objectives push us towards consistent habits when it comes to the schedules of our destinations and the routes we take to get there, but like community structure these habits are often not explicitly enumerated, and change over time. Detecting these patterns is useful for goal prediction and deviation detection, with all their related applications. In chapter 5, I studied this problem in the context of data capturing the paths of shipping vessels between ports in the English Channel.

In my initial effort, I modeled each ship's trajectory as a three layer hidden markov model. The bottom layer contained observed locations, the middle contained hidden categorical state variables which effectively discretized the space, and the third contained hidden plan variables intended to carry information about long term objectives across many time steps. I proposed an expectation-maximizing belief propagation (EMBP) algorithm for inferring the hidden states, which combined the usually distinct steps of model training and inference within a factor graph. Drawing on the results of the previous chapter, I considered both synchronous and asynchronous update schedules for the variable and parameter assignments within the graph. Comparing the techniques in their ability to predict the location of a ships next landfall, the asynchronous algorithm converged much faster and produced much more accurate estimates. However, neither algorithm produced a degree of accuracy suitable for most applications. This was in part because each was initialized with a fixed number of latent locations and, due to greedy iteration, ended up using a small subset and producing a model with very low spatial resolution.

To improve performance and explore further algorithmic problems, I considered a simplified model with a fixed spatial resolution, but where the number of latent locations was inferred from data. To make the model tractable, I proposed a new class of factor graph specifying structure not only in the mutual information between variables but between factor parameters as respects variable values. The additional structure, which is based on the CEM-tractable clustering distributions from the previous chapters, admits the use of a new belief propagation algorithm, REM-BP, based on the residual clustering of the previous chapter. I compared REM-BP with a similar algorithm performing random updates, and verified that, in spite of the added expense involved in maintaining a prioritized update schedule, REM-BP converged faster and to more compelling models than either the random schedule or the previous algorithms.

Within the path prediction task, and more generally the domain of understanding shipping patterns, the least satisfying aspect of the models I presented is the discretization of space using

centroids. This partitions space into Voronoi diagram, in that were we to exclude other factors observations would be assigned to states based slowly on their closeness to the state's center. While this might be appropriate for processes involving spatial diffusion, it's my belief that traffic patterns are better modeled as a latent graph, in that ships may be located either at particular waypoints or on thin filaments in between. This is a non-strict partitioning of space, admitting crossing filaments such as those clearly observed in shipping traffic. I spent significant unreported effort adapting the approach of [43] to the factor graph model, but was ultimately unable to overcome difficulty in initializing the model brought about by a proliferation of trivial local maxima. This underscores a remaining challenge in the generalizability of these EM-based techniques, namely that they are far more robust for some distributions of latent entities than others. A profitable area for future work would be further classifications of which distributions are "well-behaved".

Notwithstanding these limitations, REM-BP is to my knowledge the first algorithm aimed at tractable unsupervised inference within general factor graphs where variables hold an unknown number of latent states. It's my hope that it may be applicable in other domains where factor graphs are prevalent, such as image segmentation.

## 7.4 Pairwise relations in abstract spaces

A tenant of social network analysis is that relationships persist beyond individual interactions, forming structures that shape many or all social phenomena. However, the observation that under almost any operationalization social networks are *dynamic*, with links strengthening and weakening over time, begs the question of what external structure dictates those dynamics. Communities are one answer to this question, as are concepts of homophily, preferential attachment, structural holes, and many others. Some of these approaches are general theories which must be instantiated with care from application to application, but others, such as most spatial propinquity models, can be defined in terms of specific parameters.

In chapter 6 I attempted to generalize many of these models by introducing a method which related the location of *both* sides of a potential dyadic interaction within a physical or conceptual space with the probability of that interaction taking place. In contrast to methods based purely on distance across the dyad or clustering pre-processors, this kind of spatial correlation can detect overlapping effects and those which promote interaction between discontinuous regions. Because the structure of the model can better adapt to the structure of the data, it is potentially more robust to a suboptimal choice of coordinate systems or distance functions.

To train the model, I introduced *dyadic kernel density estimation*, based on previous non-parametric techniques, which predicts propinquity based on the density of similar interactions in observed data. Similarity is defined in terms of a dyadic kernel, which measures the shared information between the interaction patterns of any pair of individuals with that of a second pair. I show how to construct kernels from several kinds of distance functions, including those for geographic, ordinal, and categorical spaces.

I validated the approach in a link prediction task, where I attempted to predict missing data and future interactions within a business-to-business network of companies linked by supply chain relationships. After showing that the dataset likely contained effects necessitating a spatial correlation model, I compared models built around several kernels and several types of metadata about each company, including spatial coordinates of headquarters, market capitalization, and an categorical industrial sector classification. The prediction accuracy of each model was better than random, but capitalization and sector proved significantly more predictive of propinquity than headquarters location.

Given the presence of at least two significant effects, it seemed wasteful to choose only one of them in constructing a model. To remedy this, I showed how to construct a dyadic product kernel that would make use of dyadic distance information in several different spaces to predict propinquity. As expected, this model outperformed any of the individual models, generating accuracy very competitive with link prediction results using different algorithms (and, unfortunately, datasets).

Like other nonparametric methods, dyadic kernel density estimation suffers from a tension between requiring significant data for accuracy and scaling very poorly with additional data. While I was able to partially ameliorate this with an approximation method using only nearby nodes during estimation, the algorithm as given would remain intractable for many datasets. One direction for further improving this performance would be the integration of the modeling algorithm with efficient data structures for spatial proximity queries, such as the spatial *kd*-tree or indexing methods based space-filling curves. These could be trivially applied to speeding the approximation method presented in this paper, but could be further integrated by associating propinquity estimates with more abstracted levels of the data structure.

The main strength of dyadic kernel density estimation is that, when tractable, it can be applied to networks with several associated spatial variables with minimal model adaptation or assumed parameters. While some exploratory analysis is recommended to select a distance function or projection (for example, logistic) most aligned with kernel assumptions of symmetry and locality are most appropriate, the bandwidths of each kernel and of the product kernel can be learned automatically.

## 7.5 Bringing the methods online

A major area for future research in all three problem areas is the adaptation of the algorithms I've presented to giving online results for streaming input. In the case of FOG, this could mean tracking the fission and fusion of overlapping groups as new link observations come in. In the case of the path prediction algorithms, this could mean adapting to new traffic patterns as they are observed for the first time. And in the case of the supply chain link prediction, this could mean incorporating new link observations to the prediction of future contract initiations.

The iREM and iREM-BP algorithms I used for inference in  $\alpha$ -FOG and unsupervised factor graphs should be especially adaptable to online data. New latent variables could be initialized as the associated observed variables come in, and placed on the same queue to receive updates as previous variables. The prioritization of the queue will naturally favor updates to new data over old data that is relatively near equilibrium. For the dyadic kernel methods, it seems likely intractable to dynamically adjust kernel bandwidth to account for new data. However, if bandwidth is constant, adding new data is simply a matter of updating the structure that identifies nearest neighbors for a proposed point.

Two related challenges for online variants of the iREM and iREM-BP algorithms are (1) the propensity to get stuck in local minima, and (2) the assumption of stationarity in latent variables. Over time, both problems could make the algorithms less and less sensitive to new data as time went on. One potential solution for this would be some sort of time boxing or time decay, which would remove or underweight factors associated with observed and latent variables from long-past observations. Another solution would be temporal validation, continually making predictions for observed variables as they come in, and explicitly throwing away old data and re-learning the model when accuracy dropped below a certain point. The former solution would account well for situations in which there was a continuous drift in model parameters, and the latter would be preferable if there were punctuated regime shifts.

## 7.6 Advice for practitioners

For a practitioner trying to apply the algorithms in this thesis to a set of problems, the key question to ask is: *what are the hidden entities in my data?* If a single type of hidden classification is expected to explain observed data, a simple application of the iREM or iREM-infinite algorithms described at the end of chapter 4 can be employed. If the classifications are expected to be fuzzy

and overlapping, it may be worth applying the FOG stochastic model using that chapter's  $k$ -FOG or  $\alpha$ -FOG algorithms, respectively, if the number of latent groups is known or unknown.

If there are several kinds of hidden variables, and more complex relationships between them and the observed data, than the development of a factor graph and application of the iREM-BP algorithms might be more appropriate. This is a relatively complex process because of the need to model parametrically each of the relationships expected in the dataset. However, a virtue of that process is that the enumeration forces the researcher to explicitly state the assumptions being incorporated in the model.

Finally, if the relationships involved in the dataset are difficult to model parametrically, such as the impact of a spatial embedding on interactions, than the dyadic kernel methods of chapter 6 may be in order. If computation time is readily available, the product kernel methodology might be a good first step to determining which features influence link propensity in a new dataset.

## 7.7 Analysts and scientists

A natural tension exists between those seeking generalizable, permanent natural laws and those attempting to extract useful summaries – however transient and contextual – from a system in which they are a stakeholder. This is especially true when the two groups use similar terms and technologies, and frequently involve the same people. On the analysis side, techniques which, in pursuit of guarantees regarding truth, demand significant control of the data created or the environment producing the data, are of little use. Analysts frequently must resort to ad-hoc methods of pre-processing their data or make arbitrary decisions about model structure to shoehorn their problem into existing models. On the scientific side, the threat of admitting a summary produced less rigorously as “true” leads to understandable taboos. Modern researchers inevitably feel these tensions both internally and externally.

New data, computational resources, and social mechanisms are tilting the balance of scientific effort expended on transient versus intransient phenomena farther and farther toward the transient. Much as the next generation of English usage will be defined significantly by English speakers in countries with another national language (whose numbers now far exceed those within the English-primary world), this constitutes an irresistible force which will change the organization of the scientific community and the ways in which the scientific method is applied. This can be viewed as a risk to effective progress toward determining more fundamental aspects of human behavior, but is certainly an extraordinary opportunity to capture new data and ideas at all levels.

A determinant of the productivity of this change will be our ability to effectively manage and distinguish between levels of generalizability and transience. Nonparametric and unsupervised methods have a special roll in aiding this distinction, as they extend the vocabulary of the types of unknown or transient phenomena we can incorporate in our models in a principled, data driven way. Frameworks, such as probabilistic graphical models and product kernels, that can be adapted quickly and easily to represent a variety of structural assumptions, play a significant role in the discourse as they potentially decrease the cacophony created by method rediscovery across domains. The algorithms presented in this thesis are an incremental step in improving the integration of these techniques. It's my hope that as necessity drives further developments in this area, we will find a world of human systems data larger, but better understood and better organized than the one we experience today.





# Bibliography

- [1] Edgar database. <http://www.sec.gov/edgar.shtml>".
- [2] Google earth. <http://earth.google.com/>.
- [3] Google maps web api. <http://code.google.com/apis/maps/documentation/services.html>.
- [4] Amine Abou-rjeili and George Karypis. Multilevel algorithms for partitioning power-law graphs. In *IEEE International Parallel Distributed Processing Symposium (IPDPS)*. In, pages 16–575. press, 2006.
- [5] Ryan Prescott Adams. *Kernel Methods for Nonparametric Bayesian Inference of Probability Densities and Point Processes*. PhD thesis, Department of Physics, University of Cambridge, 2009.
- [6] E. Airoldi, D. Blei, S. Fienberg, and E.P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 4:1981–2018, September 2008.
- [7] D. Ashbrook and T. Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.
- [8] Lars Backstrom, Eric Sun, and Cameron Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th international conference on World wide web, WWW '10*, pages 61–70, New York, NY, USA, 2010. ACM.
- [9] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
- [10] I. Battacharya and L. Getoor. Deduplication and group detection using links. In *Proceedings of the tenth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Seattle, Washington, 2004.

- [11] M. J. Beal, Z. Ghahramani, and C. E. Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.
- [12] M.J. Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College, London, UK, 2003.
- [13] D. Blei, T. Gri, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process, 2003.
- [14] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [15] Christian Böhm, Christos Faloutsos, Jia-Yu Pan, and Claudia Plant. Robust information-theoretic clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 65–75, New York, NY, USA, 2006. ACM.
- [16] Stephen P Borgatti, M G Everett, and L C Freeman. Ucinet for windows: Software for social network analysis. *Harvard Analytic Technologies*, 2006, 2002.
- [17] Ronald Breiger, Scott Boorman, and Phipps Arabie. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *Journal of Mathematical Psychology*, 12:328–383, 1975.
- [18] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30:107–117, April 1998.
- [19] M. Brockmann, T. Gasser, and E. Hermann. Locally Adaptive Bandwidth Choice for Kernel Estimation. *Journal of American Statistical Association*, 88:1302–1309, 1993.
- [20] Carter T. Butts. *Spatial Models Large-Scale Interpersonal Networks*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2002.
- [21] Carter T. Butts. Predictability of large-scale spatially embedded networks. In *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*. Washington, D.C.: National Academies Press, 2003.
- [22] Kathleen M. Carley. Dynamic network analysis. In *Committee on Human Factors*, pages 133–145. National Research Council, 2004.

- [23] Kathleen M. Carley, Neal Altman, Douglas Fridsma, Elizabeth Casman, Alex Yahja, Li-Chou Chen, Boris Kaminski, and Damian Neve. Biowar: Scalable agent-based model of bioattacks. *IEEE Transactions on Systems, Man, and Cybernetics*, 36:252–265, 2006.
- [24] R. M. Christley, G. L. Pinchbeck, R. G. Bowers, D. Clancy, N. P. French, R. Bennett, and J. Turner. Infection in Social Networks: Using Network Analysis to Identify High-Risk Individuals. *Am J Epidemiol*, 162(10):1024–1031, September 2005.
- [25] Aaron Clauset, Cristopher Moore, and M.E.J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453, 2008.
- [26] Aaron Clauset, M. E. J. Newman, , and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, pages 1– 6, 2004.
- [27] Ronald H. Coase. The nature of the firm. *Economica*, 16:386–407, 1937.
- [28] B. Cowgill, J. Wolfers, and E. Zitewitz. Using prediction markets to track information flows. *Working paper*, 2008.
- [29] A. Davis, B.B. Gardner, and M.R. Gardner. *Deep South: A sociological and anthropological case study of caste and class*. 1941.
- [30] G.B. Davis, J. Olson, and K. M. Carley. Unsupervised plan detection with factor graphs. In *Proceedings of Sensor-KDD Workshop at 2008 ACM-SigKDD*, 2008.
- [31] George B. Davis and Kathleen M. Carley. Clearing the fog: Fuzzy, overlapping groups for social networks. *Social Networks*, 30:201–212, 2008.
- [32] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):pp. 1–38, 1977.
- [33] Jana Diesner, Terrill L. Frantz, and Kathleen M. Carley. Communication networks from the enron email corpus "it's always about the people. enron is no different". *Computational & Mathematical Organization Theory*, 11(3):201–228, 2005.
- [34] P. Doreian, J. Batageli, and A. Ferligoj. *Generalized Blockmodeling*. Cambridge University Press, Cambridge, 2005.
- [35] Patrick Doreian, Vladimir Batagelj, and Anuška Ferligoj. Generalized blockmodeling of two-mode network data. *Social Networks*, 26(1):29–53, 2004.

- [36] Patrick Doreian and Andrej Mrvar. A partitioning approach to structural balance. *Social Networks*, 18(2):149 – 168, 1996.
- [37] Thi V. Duong, Hung H. Bui, Dinh Q. Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 838–845, Washington, DC, USA, 2005. IEEE Computer Society.
- [38] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI)*, Boston, Massachusetts, July 2006.
- [39] F.Lorrain and H.C. White. Structural equivalence of individuals in social networks. *Journal of Mathematical Sociology*, 1, 1971.
- [40] Linton C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1(3):215–239, 1979.
- [41] Linton C. Freeman. The sociological concept of "group": An empirical test of two models. *The American Journal of Sociology*, 98(1):152–166, 1992.
- [42] Brendan Frey. Extending factor graphs so as to unify directed and undirected graphical models. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 257–26, San Francisco, CA, 2003. Morgan Kaufmann.
- [43] Andrew Moore Geoff Gordon. Learning filaments. In *Proceedings of the International Conference on Machine Learning*, pages 335–342, 340 Pine Street, 6th Fl., San Francisco, CA 94104, 2000. Morgan Kaufmann.
- [44] Lise Getoor and Ben Taskar, editors. *Introduction to Statistical Relational Learning*. The MIT Press, Boston, 1 edition, 2007.
- [45] Eric Gilbert and Karrie Karahalios. Predicting tie strength with social media. In *In Proceedings of the Conference on Human Factors in Computing Systems (CHI09)*, page 220, 2009.
- [46] Thomas Griffiths and Zoubin Ghahramani. Infinite latent feature models and the indian buffet process. *Advances in Neural Information Processing Systems*, 18(GCNU TR 2005-001):475, 2005.

- [47] Peter D. Grnwald. *The Minimum Description Length Principle*, volume 1 of *MIT Press Books*. The MIT Press, 2007.
- [48] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.
- [49] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.
- [50] A. G. Hawkes and D. Oakes. A cluster representation of a self - exciting process. 1974.
- [51] Bruce Hendrickson and Robert Leland. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Sci. Comput.*, 16:452–469, March 1995.
- [52] John R. Hershey and Peder A. Olsen. Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007.*, volume 4, pages IV–317–IV–320, 2007.
- [53] David R. Hunter, Mark S. Handcock, Carter T. Butts, Steven M. Goodreau, and Martina Morris. ergm: A package to fit, simulate and diagnose Exponential-Family models for networks. *Journal of Statistical Software*, 24(3):1–29, 2007.
- [54] Aapo Hyvrinen, Patrik O. Hoyer, and Mika Inki. Independent component analysis. *Neural Computing Surveys*, 2, 2001.
- [55] Piotr Indyk. Nearest neighbors in high-dimensional spaces. In J. E. Goodman and J. O’Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 877–892. CRC Press, second edition, 2004.
- [56] M.C. Jones, S.J. Davis, and B. U. Park. Versions of kernel-type regression estimators. *Journal of the American Statistical Association*, 89(427):625–832, 1994.
- [57] U. Kang, Charalampos E. Tsourakakis, Ana Paula Appel, Christos Faloutsos, and Jure Leskovec. Hadi: Mining radii of large graphs. *ACM Trans. Knowl. Discov. Data*, 5:8:1–8:24, February 2011.
- [58] George Karypis and Vipin Kumar. Metis - unstructured graph partitioning and sparse matrix ordering system, version 2.0. Technical report, 1995.

- [59] Hisashi Kashima and Yuta Tsuboi. Kernel-based discriminative learning algorithms for labeling sequences, trees, and graphs. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 58–, New York, NY, USA, 2004. ACM.
- [60] Jon Kleinberg. Navigation in a small world. *Nature*, 406, 2000.
- [61] D. Koller, N. Friedman, L. Getoor, and B. Taskar. Graphical models in a nutshell. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
- [62] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research (JMLR)*, 9:235–284, February 2008.
- [63] Marzena Kryszkiewicz and Łukasz Skonieczny. Faster Clustering with DBSCAN. pages 605–614. 2005.
- [64] Jeremy Kubica, Andrew Moore, David Cohn, and Jeff Schneider. cgraph: A fast graph-based method for link analysis and queries. In *In Proc. IJCAI Text-Mining and Link-Analysis Workshop*, 2003.
- [65] Jeremy Kubica, Andrew W. Moore, and Jeff G. Schneider. Tractable group detection on large link data sets. In *ICDM*, pages 573–576, 2003.
- [66] Sanjiv Kumar and Martial Hebert. Discriminative random fields. *International Journal of Computer Vision*, 68(2):179–201, 2006.
- [67] Jure Leskovec. *Dynamics of large networks*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2008.
- [68] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.
- [69] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Laws of graph evolution: Densification and shrinking diameters. *ACM Transactions on knowledge discovery from data*, 1:1–40, 2006.
- [70] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW*, pages 695–704, 2008.

- [71] Qi Li and Jeffrey S. Racine. *Nonparametric Econometrics: Theory and Practice*. Princeton University Press, November 2006.
- [72] Lin Liao, Dieter Fox, and Henry Kautz. Extracting places and activities from gps traces using hierarchical conditional random fields. *Int. J. Rob. Res.*, 26(1):119–134, 2007.
- [73] Lin Liao, Donald J. Patterson, Dieter Fox, and Henry Kautz. Learning and inferring transportation routines. *Artif. Intell.*, 171(5-6):311–331, 2007.
- [74] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of the 12th International Conference on Information and Knowledge Management*, 2003.
- [75] Ryan Lichtenwalter, Jake T. Lussier, and Nitesh V. Chawla. New perspectives and methods in link prediction. In *Knowledge Discovery and Data Mining*, pages 243–252, 2010.
- [76] Marek Lipczak and Evangelos Milios. Agglomerative genetic algorithm for clustering in social networks. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, GECCO '09, pages 1243–1250, New York, NY, USA, 2009. ACM.
- [77] Han Liu, John D. Lafferty, and Larry A. Wasserman. Sparse nonparametric density estimation in high dimensions using the rodeo. *Journal of Machine Learning Research - Proceedings Track*, 2:283–290, 2007.
- [78] Girvan M. and Newman M.E.J. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 99, 2002.
- [79] David J. C. Mackay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 1st edition, June 2003.
- [80] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.
- [81] Maurizio Maravalle, Bruno Simeone, and Rosella Naldini. Clustering on trees. *Computational Statistics Data Analysis*, 24(2):217–234, 1997.
- [82] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

- [83] Xiao-Li Meng and Donald B Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993.
- [84] Stanley Milgram. The small world problem. *Psychology Today*, 1, 1967.
- [85] K. Miller, T. Griffiths, and M.I. Jordan. Nonparametric latent feature models for link prediction. In *Advances in Neural Information Processing Systems 22*, 2000.
- [86] Mark S. Mizruchi. What do interlocks do? an analysis, critique, and assessment of research on interlocking directorates. *Annual Review of Sociology*, 22:271–298, 1996.
- [87] J. Moody and D.R. White. Structural cohesion and embeddedness: A hierarchical concept of social groups. *American Sociological Review*, pages 103–127, 2003.
- [88] Il-Chul Moon and Kathleen M. Carley. Modeling and simulating terrorist networks in social and geospatial dimensions. *IEEE Intelligent Systems*, 22(5):40–49, 2007.
- [89] Kevin Murphy. *Dynamic Bayesian Networks*. PhD thesis, UC Berkeley, Berkeley, CA, 2002.
- [90] E.A. Nadaraya. On estimating regression. *Theory of Probability and its Application*, 10:186–190, 1964.
- [91] Radford Neal and Geoffrey E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.
- [92] T. Newcomb. *The acquaintance process*. Reinhardt & Winston, 1961.
- [93] M. E. J. Newman. Analysis of weighted networks. *PHYS.REV.E*, 70:056131, 2004.
- [94] M. E. J. Newman. Coauthorship networks and patterns of scientific collaboration. In *Proceedings of the National Academy of Sciences*, pages 5200–5205, 2004.
- [95] N. Nguyen, D. Phung, S. Venkatesh, and H. Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden markov model. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2005.
- [96] K. Nowicki and T. A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, September 2001.
- [97] Sean B. O’Hagan and Milford B. Green. Corporate knowledge transfer via interlocking directorates: a network analysis approach. *Geoforum*, 35(1):127–139, 2004.



- [98] Jamie Olson and Kathleen M. Carley. Summarization and information loss in network analysis. In *Workshop on Link Analysis, Counterterrorism and Security, SIAM International Conference on Data Mining*, 2008.
- [99] Ooi, K. J. Mcdonell, and Sacks R. Davis. Spatial kd-tree: An Indexing Mechanism for Spatial Database. *COMPSAC conf.*, pages 433–438, 1987.
- [100] Gergely Palla, Imre Derenyi, Illes Farkas, and Tamas Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, jun 2005.
- [101] Yuqing Ren, Kathleen M. Carley, and Linda Argote. The contingent effects of transactive memory: When is it more beneficial to know what others know? *Management Science*, 52(5):671–682, 2006.
- [102] Matthew Richardson and Pedro Domingos. Markov logic networks. *Mach. Learn.*, 62:107–136, February 2006.
- [103] Garry Robbins, Pip Pattison, Yuval Kalish, and Dean Lusher. An introduction to exponential random graph ( $p^*$ ) models for social networks. *Social Networks*, 29, 2007.
- [104] F.S. Sampson. *A Novitiate in a Period of Change: An Experimental and Case Study of Social Relationships*. PhD thesis, Cornell, Ithaca, NY, 1968.
- [105] Craig Schreiber and Kathleen M. Carley. Construct - a multi-agent network model for the co-evolution of agents and socio-cultural environments. Technical Report CMU-ISRI-04-109, Carnegie Mellon University, School of Computer Science, Institute for Software Research International, Pittsburgh, PA, 2004.
- [106] Georg Simmel. *Sociology: investigations on the forms of sociation*. Duncker Humblot, Leipzig, 1908.
- [107] Tom Snijders, Philippa Pattison, Garry Robins, and Mark Handcock. New specifications for exponential random graph models. *Sociological Methodology*, 36, 2008.
- [108] Tom A. B. Snijders, Gerhard G. van de Bunt, and Christian E. G. Steglich. Introduction to stochastic actor-based models for network dynamics. *Social Networks*, March 2009.
- [109] Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *Journal of Machine Learning Research*, 8:693–723, 2007.

- [110] R. Sutton and A. McCallum. Improved dynamic schedules for belief propagation. In *Proceedings of the Twenty-third Conference on Uncertainty in AI (UAI)*, Vancouver, BC, July 2007.
- [111] B. Taskar, P. Abbeel, M.-F. Wong, and D. Koller. Relational markov networks. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.
- [112] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2002.
- [113] Larry Wasserman. *All of Nonparametric Statistics (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [114] S Wasserman and K Faust. *Social Network Analysis*, chapter Social Network Analysis and the Behavioral Sciences. Cambridge University Press, New York, 1 edition, 1994.
- [115] G.S. Watson. Smooth regression analysis. *Sankhya Series A*, 26:359–372, 1964.
- [116] Duncan J. Watts and Steven H. Strogatz. *Collective dynamics of 'small world' networks*, chapter 4.2, pages 301–303. Princeton University Press, 2006.