

ABSTRACT

Title of Dissertation:

MEASURING NETWORK DEPENDENCIES
FROM NODE ACTIVATIONS
Rachael T.B. Sexton
Doctor of Philosophy,

Dissertation Directed by:

Professor Mark D. Fuge
Department of Mechanical Engineering

My abstract for this dissertation.

MEASURING NETWORK DEPENDENCIES FROM NODE ACTIVATIONS

by

Rachael T.B. Sexton

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

Advisory Committee:

Professor Mark D. Fuge, Chair/Advisor

Professor Jordan L. Boyd-Graber

Professor Maria K. Cameron

Professor Michelle Girvan

Professor Vincent P. Lyzinski

Preface

Foreward

Acknowledgements

Table of Contents

Preface	i
Foreward	ii
Acknowledgements	iii
Table of Contents	iii
List of Tables	vii
List of Figures	viii
1 Introduction	1
1.1 Ambiguous Metrology	1
1.2 Indirect Network Measurement	3
1.3 Scope of this work	7
I A Practitioner’s Guide to Network Recovery	9
2 Metrology with matrices	10
2.1 Observation and feature “spaces”	11
2.2 Models & linear operators	12
2.3 Measurement quantification & error	15
2.3.1 Additive Smoothing	15
2.3.2 Conditional Probabilities & Contingencies	17
2.4 Distance vs. Incidence	19
2.4.1 Kernels & distances	20
2.4.2 Incidence structures & dependency	22
3 Vector representations of incidence	25
3.1 Graphs as incidence structures	26
3.1.1 Embedding incidences in vector space	28
3.1.2 Inner products on B	31
3.1.3 Edge Metrology, Edge Vectors	32
3.2 Node activation, bipartite graphs, and hypergraphs	33

3.2.1	Inner product on Hyperedges	35
3.2.2	Combining Occurrence & Dependence	35
4	Roads to Network Recovery	37
4.1	Organizing Recovery Methods	37
4.1.1	Local Structure & Additivity Assumptions	39
4.1.2	Resource and Information Flow	42
4.1.3	Global Structural Assumptions	44
4.2	A Path Forward	47
4.2.1	Observation space assumptions	47
4.2.2	Filling the local+data “gap”	49
II	Nonparametric Network Recovery With Random Spanning Forests	52
5	Latent Graphs with Desire Paths	53
5.1	The Gambit of the Inner Product	54
5.1.1	Gambit of the Group	54
5.1.2	Inner-Product Projections and “Clique Bias”	55
5.2	Networks as Desire Path Density Estimates	58
5.2.1	Subgraph Distributions	59
5.2.2	Graph Unions as Desire Paths	61
6	Approximate Recovery in Near-linear Time by <i>Forest Pursuit</i>	65
6.1	Random Walks as Spanning Forests	66
6.1.1	Random Walk Activations	66
6.1.2	Activations in a Forest	67
6.1.3	Spreading Dependencies as Trees	69
6.2	Sparse Approximation	71
6.2.1	Problem Specification	71
6.2.2	Max. Spanning (Steiner) Trees	73
6.3	Forest Pursuit	75
6.3.1	Algorithm Summary	75
6.3.2	Approximate Complexity	76
6.4	Simulation Study	79
6.4.1	Experimental Method	79
6.4.2	Metrics	81
6.4.3	Results - Scoring	83
6.4.4	Results - Runtime Performance	88
7	Modifications & Extensions	91
7.1	Forest Pursuit Interaction Probability	91
7.1.1	Simulation Study Revisited	92
7.1.2	Simulation Case Study	92

7.2	Generative Model	94
7.2.1	Marked Random Spanning Forest (RSFm) distribution	94
7.2.2	Model Specification	98
7.3	Expected Forest Maximization	99
7.3.1	Factorization & Dictionary Learning	100
7.3.2	EFM Simulation Study	102
III	Applications & Case Studies	105
8	Qualitative Application of Relationship Recovery	106
8.1	Network Science Collaboration Network	107
8.2	Les Misérables Character Network	112
9	Recovery from Working Memory & Partial Orders	118
9.1	Technical Language Processing with INVITE	119
9.1.1	Optimizing absorbing-state probabilities	121
9.1.2	Application: Mining Excavator MWOs	123
9.1.3	Results	124
9.2	Forest Pursuit Animal Network	127
9.2.1	Domain-aware preprocessing	128
9.2.2	Edge Connective Efficiency and Diversity	130
9.2.3	Thresholded Structure Preservation	138
9.2.4	Forest Pursuit as Preprocessing	138
10	Conclusion & Future Work	142
10.1	Modifications and Extensions to Forest Pursuit	142
10.1.1	Multi-source Spreading	142
10.1.2	Hidden Node Discovery	142
10.1.3	MCMC and Gibbs Sampling	142
10.2	Generalizing Inner Products on Incidences	142
10.2.1	special orthogonal group	142
10.2.2	\mathcal{H}_2 embedding and beltrami voronoi	143
10.3	Potential Applications and Case Studies	143
10.3.1	Topology-Dependent Graph Application	143
10.3.2	Human-in-the-loop for Exploratory Relational Analysis	143

List of Tables

4.1	Organizing recovery methods by representation space and level	50
6.1	Experiment Settings (MENDR Dataset)	79
6.2	Summary of algorithms compared	80
7.1	Comparing scores for FP, FPi and GLASSO	93
9.1	Maintenance Work Order as categorized vs. tagged data	120

List of Figures

1.1	Zachary’s Karate Club, with ambiguously extant edge 78 highlighted.	2
1.2	Observations as activation sets	3
1.3	Co-authorship vs. collaborator network	4
1.4	Recovering underlying dependency networks from node-cooccurrences.	6
2.1	Spring system as a network of conditional dependencies	23
5.1	Gram matrix as sum of observation outer products	56
5.2	Inner-product projections as sums of cliques illustrating “clique bias”.	57
6.1	Edge Measurements with true (tree) dependencies known	69
6.2	Comparison of MENDR recovery scores	84
6.3	Comparison of MENDR Recovery Scores by Graph Type	85
6.4	Score trends vs problem scaling	86
6.5	Partial Residuals (regression on $E[MCC]$)	87
6.6	Runtime Scaling (Forest-Pursuit vs GLASSO)	89
6.7	Partial Residuals (regression on computation time)	89
7.1	FPi shows best APS, lower MCC,F-M	93
7.2	P-R curves for two experiments	95
7.3	Dissemination plan as rooted RST on augmented graph	96
7.4	Change in Expected MCC (EFM vs FP)	102
7.5	Logistic Regression Coef. (EFM - FP) vs. (Ground Truth)	103
7.6	Runtime Scaling (Forest-Pursuit vs GLASSO)	104
7.7	Partial Residuals (regression on computation time)	104
8.1	134 Network scientists connected by co-authorship	108
8.2	Chow-Liu tree of NetSci collaborator dependency relationships	109
8.3	Forest Pursuit estimate of NetSci collaborator dependency relationships	111
8.4	Degree distributions of FP vs co-occurrence social networks	111
8.5	Les Misérables character co-occurrence network	113
8.6	Les Misérables character dependency network (Forest Pursuit)	115
8.7	Changes in character centrality ranking for FP vs co-occurrence	116
9.1	Partial-order edge measurements with Markov assumption	121
9.2	Semisupervised MWO Tag Classification with Network Label Propagation	126
9.3	Effects of rolling-window activations on observation data	129

9.4	Verbal Fluency (animals) Network Backbone (Doubly-Stochastic) . . .	131
9.5	Verbal Fluency (animals) Dependency Network (Chow-Liu Tree) . . .	133
9.6	Verbal Fluency (animals) Dependency Network (GLASSO)	134
9.7	Verbal Fluency (animals) Dependency Network (Forest Pursuit) . . .	135
9.8	Changes in animal centrality ranking for FP vs co-occurrence,GLASSO	137
9.9	Differences in structural preservation with over-thresholding.	139
9.10	Forest Pursuit preprocessing for Doubly-Stochastic and GLASSO recovered networks	141

Chapter 1: Introduction

A wide variety of fields show consistent interest in inferring latent network structure from observed interactions, from human cognition and social infection networks, to marketing, traffic, finance, and many others. [50] However, an increasing number of authors are noting a lack of agreement in how to approach the metrology of this problem. This includes rampant disconnects between the theoretical and methodological network analysis sub-communities[6], treatment of error as purely aleatory, rather than epistemic [55], or simply ignoring measurement error in network reconstruction entirely[28].

1.1 Ambiguous Metrology

Networks in the “wild” rarely exist of and by themselves. Rather, they are a model of interaction or relation *between* things that were observed. One of the most beloved examples of a network, the famed *Zachary’s Karate Club*[86], is in fact reported as a list of pairwise interactions: every time a club member interacted with another (outside of the club), Zachary recorded it as two integers (the IDs of the members). The final list of pairs can be *interpreted* as an “edge list”, which can be modeled with a network: a simple graph. This was famously used to show natural community

structure that nicely matches the group separation that eventually took place when the club split into two.[\[73\]](#)

Note, however, that we could have just as easily taken note of the instigating student for each interaction (i.e. which student initiated conversation, or invited the other to socialize, etc.). If that relational asymmetry is available, our “edges” are now *directed*, and we might be able to ask questions about the rates that certain students are asked vs. do the asking, and what that implies about group cohesion. Additionally, the time span is assumed to be “for the duration of observation” (did the students ever interact), but if observation time was significantly longer, say, multiple years, we might question the credulity of treating a social interaction 2 years ago as equally important to an interaction immediately preceding the split. This is now a “dynamic” graph; or, if we only measure relative to the time of separation, at the very least a “weighted” one.

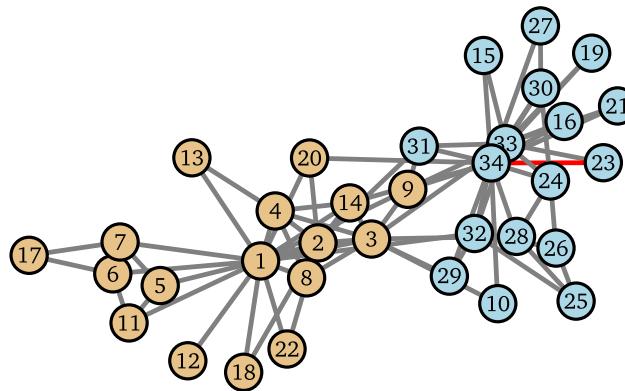


Figure 1.1: Zachary’s Karate Club, with ambiguously extant edge 78 highlighted.

We do not know if any of these are true. In fact, as illustrated in Figure [1.1](#), we do not know if the network being described from the original edge data even has

77 or 78 edges, due to ambiguous reporting in the original work. Lacking a precise definition of what the graph’s components (i.e. it’s edges) are, *as measurable entities*, means we cannot estimate the measurement error in the graph.

1.2 Indirect Network Measurement

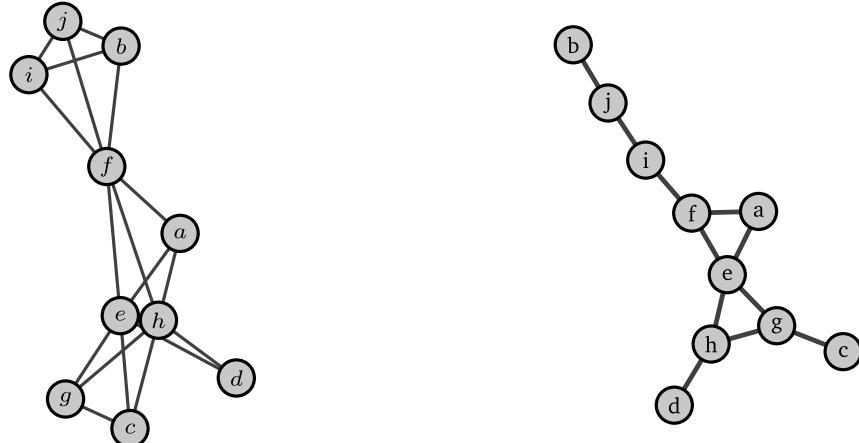
While the karate club graph has unquantified edge uncertainty derived from ambiguous edge measurements, we are fortunate that we *have edge measurements*. Regardless of how the data was collected, it is de facto reported as a list of pairs. In many cases, we simply do not have such luxury. Instead, our edges are only measured *indirectly*, and instead we are left with lists of node co-occurrences. Networks connecting movies as being “similar” might be derived from data that lists sets of movies watched by each user; networks of disease spread pathways might be implied from patient infection records; famously, we might build a network of collaboration strength between academic authors by mining datasets of the papers they co-author together.

Such networks are derived from what we will call *node activation* data, i.e., records of what entities happened “together”, whether contemporaneously, or in some other context or artifact.

$$\begin{aligned} \{ & \quad d \quad h \quad e \quad \} = x_1 \\ \{ & g \quad c \quad e \quad h \quad \} = x_2 \\ \{ & f \quad e \quad a \quad h \quad \} = x_3 \\ \{ & i \quad j \quad f \quad b \quad \} = x_4 \end{aligned}$$

Figure 1.2: Observations as activation sets

These are naturally represented as “bipartite” networks, having separate entities for, say, “papers” and “authors”, and connecting them with edges (paper 1 is “connected” to its authors E,H,C, etc.). But analysts are typically seeking the collaboration network connecting authors (or papers) themselves! Networks of relationships in this situation are not directly observed, but which *if recovered* could provide estimates for community structure, importances of individual authors (e.g. as controlling flow of information), and the “distances” that separate authors from each other, in their respective domains. [75] Common practice assumes that co-authorship in any paper is sufficient evidence of at least some level of social “acquaintance”, where more papers shared means more “connected”.



(a) Network based solely on co-authorship observations
 (b) graph of mutual collaboration relationships
 i.e. the “ground truth” social network

Figure 1.3: Co-authorship vs. collaborator network

Thus our social collaboration network in Figure 1.3a is borne out of indirect measurements: author connection is implied through “occasions when co-authorship occurred”. However, authors of papers may recall times that others were added, not by their choice, but by someone else already involved. In fact, the final author list

of most papers is reasonably a result of individuals choosing to invite others, not a unanimous, simultaneous decision by all members. Let's imagine we wished to study the social network of collaboration more directly: if we had the luxury of being in situ as, say, a sociologist performing an academic ethnography, we might have been more strict with our definition of "connection". If the goal is a meaningful social network reflecting the strength of interaction between colleagues, perhaps we prefer our edges represent "mutual willingness to collaborate". Edge "measurement", then, could involve records of events that show willingness to seek or participate in collaboration event, such as:

- *author (g) asked (e), (h), and (c) to co-author a paper, all of whom agreed*
- *(i) asked (f) and (j), but (j) wanted to add (b)'s expertise before writing one of the sections*

and so on. Each time two colleagues had an opportunity to work together *and it was seized upon* we might conclude that evidence of their relationship strengthened. With data like this, we could be more confident in claiming our collaboration network can serve as "ground truth," as far as empirically confirmed collaborations go. However, even if the underlying "activations" are identical, our new, directly measured graph looks very different.

Fundamentally, the network in Figure 1.3b shows which relationships the authors *depend on* to accomplish their publishing activity. When causal relations between nodes are being modeled as edges, we call such a graph a *dependency network*. We will investigate this idea further later on, but ultimately, if a network of dependencies

is desired (or implied, based on analysis needs), then the critical problem remaining is *how do we recover dependency networks from node activations?* Additionally, what goes wrong when we use co-occurrence/activation data to estimate the dependency network, especially when we wish to use it for metrics like centrality, shortest path distances, and community belonging?

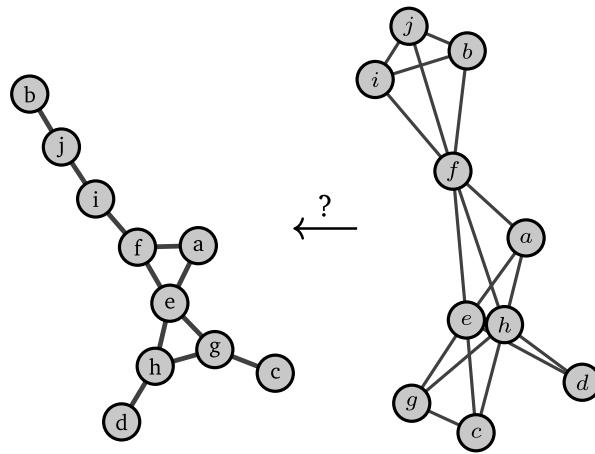


Figure 1.4: Recovering underlying dependency networks from node-cooccurrences.

Even more practically, networks created directly from bipartite-style data are notorious for quickly becoming far too dense for useful analysis, earning them the (not-so-)loving moniker “hairballs”. Network “backboning,” as it has come to be called tries to find a subset of edges in this hairball that still captures its core topology in a way that’s easier to visualize.[42, 59] Meanwhile, underlying networks of dependencies that *cause* node activation patterns can provide this: they are almost always more sparse than their hairballs. Accessing the dependency *backbone* in a principled way is difficult, but doing so in a rapid, scalable manner is critical for practitioners to be able to make use of it to trim their hairballs.

1.3 Scope of this work

The purpose of this thesis is to provide a solid foundation for basic edge metrology when our data consists of binary node activations, by framing network analysis as a problem of *inference*, as suggested by Peel et al. [6]. We give special focus to binary activations that occur due to spreading processes, such as random walks or cascades on an underlying carrier graph. Recovering the carrier, or, “dependency” network from node activations is of great interest to the network backboning and causal modeling communities, but often involves either unspoken sources of epistemic and aleatory error, or high computation costs (or both). To begin addressing these issues, we present a guide to current practices, pitfalls, and how common statistical tools apply to the network recovery problem: a *Practitioner’s Guide to Network Recovery*. We cover what “measurement” means in our context, and specifically the ways we encode observations, operations, and uncertainties numerically. Clarifying what different versions of what “relation” means (whether proximity or incidence) is critical, since network structure is intended to encode such relations as mathematical objects, despite common ambiguities and confusion around what practitioners intend on communicating through them. Then we use this structure to present a cohesive framework for selecting a useful network recovery technique, based on the available data and where in the data processing pipeline is acceptable to admit either extra modeling assumptions or information loss.

Next, building on a gap found in the first part, we present a novel method, *Forest Pursuit*, to extract dependency networks when we know a *spreading process*

causes node activation (e.g. paper co-authorship caused by collaboration requests). We create a new reference dataset to enable community benchmarking of network recovery techniques, and use it show greatly improved accuracy over many other widely-used methods. Forest Pursuit in its simplest form scales linearly with the size of active-node sets, being trivially parallelizable and streamable over dataset size, and agnostic to network size overall. We then expand our analysis to re-imagine Forest Pursuit as a Bayesian probabilistic model, *Latent Forest Allocation*, which has an easily-implemented Expectation Maximization scheme for posterior estimation. This significantly improves upon the accuracy results of Forest Pursuit, at the cost of some speed and scalability, giving analysts multiple options to adapt to their needs.

Last, we apply Forest Pursuit to several qualitative case-studies, including a scientific collaboration network, and the verbal fluency “animals” network recovery problem, which dramatically change interpretation under use of our method. We investigate its use as a low-cost preprocessor for other methods of network recovery, like GLASSO, improving their stability and interpretability. Finally we discuss the special case when node activations are reported as an ordered set, where accounting for cascade-like effects becomes crucial to balance false positive and false-negative edge prediction. Along with application of this idea to knowledge-graph creation from technical language in the form maintenance work-order data, we discuss more broadly the future needs of network recovery, specifically in the context of embeddings and gradient-based machine learning toolkits.

Part I

A Practitioner's Guide to Network Recovery

Chapter 2: Metrology with matrices

Where metrology is concerned, the actual unit of observation and how it is encoded for us is critical to how analysts may proceed with quantifying, modeling, and measuring uncertainty around observed phenomena. Experiment and observation tends to be organized as inputs and outputs, or, independent variables and dependent variables, specifically. Independent variables are observed, multiple times (“observations”), and changes in outcome for each can be compared to the varying values associated with the independent variable input (“features”). For generality, say a practitioner records their measurements as scalar values, i.e. $x \in \mathbb{S} \in \{\mathbb{R}, \mathbb{Z}, \mathbb{N}, \dots\}$. The structure most often used to record scalar values of n independent/input variable features over the course of m observations is called a design matrix $X : \mathbb{S}^{m \times n}$.¹

¹Not all observations are scalar, but they can become so. If individual measurements are higher-dimensional (e.g. images are 2D), X is a tensor, which can be transformed through unrolling or embedding into a lower dimensional representation before proceeding. There are other techniques for dealing with e.g. categorical data, such as one-hot encoding (where the features are binary for each possible category, with boolean entries for each observation).

2.1 Observation and feature “spaces”

If we index a set of observations and features, respectively, as

$$i \in I = \{1, \dots, m\}, \quad j \in J = \{1, \dots, n\}, \quad I, J : \mathbb{N}$$

then the design matrix can map the index of an observation and a feature to the corresponding measurement.

$$x = X(i, j) \quad X : I \times J \rightarrow \mathbb{S} \quad (2.1)$$

i.e. the measured value of the j th independent variable from the i th observation.² In this scheme, an “observation” is a single row vector of features in $\mathbb{S}^{n \times 1}$ (or simply \mathbb{S}^n), such that each observation encodes a position in the space defined by the features, i.e. the *feature space*, and extracting a specific observation vector i from the entire matrix can be denoted as

$$\mathbf{x}_i = X(i, \cdot), \quad \mathbf{x} : J \rightarrow \mathbb{S}$$

Similarly, every “feature” is associated with a single column vector in $\mathbb{S}^{1 \times m}$, which can likewise be interpreted as a position in the space of observations (the *data space*):

$$\mathbf{x}'_j = X(\cdot, j), \quad \mathbf{x}' : I \rightarrow \mathbb{S}$$

²This notation is adapted from the sparse linear algebraic treatment of graphs in Kepner and Gilbert [57] and Kepner et al. [38].

Note that this definition could be swapped without loss of generality. In other words, \mathbf{x} and \mathbf{x}' being in row and column spaces is somewhat arbitrary, having more to do with the logistics of experiment design and data collection. We could have measured our feature vectors one-at-a-time, measuring their values over an entire “population”, in effect treating that as the independent variable set.³

To illustrate this formalism in a relevant domain, let’s take another look at co-citation networks. For m papers we might be aware of n total authors. For a given paper, we are able to see which authors are involved, and we say those authors “activated” for that paper. It makes sense that our observations are individual papers, while the features might be the set of possible authors. However, we are not given information about which author was invited by which other one, or when each author signed on. In other words, the measured values are strictly boolean, and we can structure our dataset as a design matrix $X : \mathbb{B}^{m \times n}$. We can then think of the i^{th} paper as being represented by a vector $\mathbf{x}_i : \mathbb{B}^n$, and proceed using it in our various statistical models. If we desired to analyze the set of authors, say, in order to determine their relative neighborhoods or latent author communities, we could equally use the feature vectors for each paper, this time represented in a vector $\mathbf{x}'_i : \mathbb{B}^{1 \times m}$.

2.2 Models & linear operators

Another powerful tool an analyst has is *modeling* the observation process. This is relevant when the observed data is hypothesized to be generated by a process we

³In fact, vectors are often said to be in the column-space of a matrix, especially when using them as transformations in physics or deep learning layers. We generally follow a one-observation-per-row rule, unless otherwise stated.

can represent mathematically, but we do not know the parameter values to best represent the observations (or the observations are “noisy” and we want to find a “best” parameters that account for this noise). This is applicable to much of scientific inquiry, though one common use-case is the de-blurring of observed images (or denoising of signals), since we might have a model for how blurring “operated” on the original image to give us the blurred one. We call this “blurring” a *linear operator* if it can be represented as a matrix⁴, and applying it to a model with l parameters is called the *forward map*:

$$\mathbf{x} = F\mathbf{p} \quad F : \mathbb{R}^l \rightarrow \mathbb{R}^n$$

where P is the space of possible parameter vectors, i.e. the *model space*. The forward map takes a modeled vector and predicts a location in data space.

Of critical importance, then, is our ability to recover some model parameters from our observed data, e.g. if our images were blurred through convolution with a blurring kernel, then we are interested in *deconvolution*. If F is invertible, the most direct solution might be to apply the operator to the data, as the *adjoint map*:

$$\mathbf{p} = F^H \mathbf{x} \quad F^H : \mathbb{R}^n \rightarrow \mathbb{R}^l$$

which removes the effect of F from the data \mathbf{x} to recover the desired model \mathbf{p} .

Trivially we might have an orthogonal matrix F , so $F^H = F^{-1}$ is available directly. In practice, other approaches are used to minimize the *residual*: $\hat{\mathbf{p}} = \min_{\mathbf{p}} F\mathbf{p} - \mathbf{x}$.

⁴in the finite-dimensional case

Setting the gradient to 0 yields the normal equation, such that

$$\hat{\mathbf{p}} = (F^T F)^{-1} F^T \mathbf{x}$$

This should be familiar to readers as equivalent to solving ordinary least-squares (OLS). However, in that case it is more often shown as having the *design matrix* X in place of the operator F .

This is a critical distinction to make: OLS as a “supervised” learning method treats some of the observed data we represented as a design matrix previously as a target to be modeled, $y = X(\cdot, j)$, and the rest maps parameters into data space, $F = X(\cdot, J/j)$. With this paradigm, only the target is being “modeled” and the rest of the data is used to create the operator. In the citation network example, it would be equivalent to trying to predict one variable, like citation count or a specific author’s participation in every paper, *given* every other author’s participation in them.

For simplicity, most work in the supervised setting treats the reduced data matrix as X , opting to treat y as a separate *dependent variable*. However, our setting will remain *unsupervised*, since no single target variable is of specific interest—all observations are “data”. In this, we more closely align with the deconvolution literature, such that we are seeking a model and an operation on it that will produce the observed behavior in an “optimal” way.

2.3 Measurement quantification & error

In binary data, such as what we have been considering, it is common to model observables as so-called “Bernoulli trials”: events with two possible outcomes (on, off; yes, no; true, false), and one outcome has probability p . These can be thought of as weighted coin-flips: “heads” with probability p , and “tails” $1 - p$. If k trials are performed (the “exposure”), we say the number of successes s (the “count”) is distributed as a binomial distribution $s \sim \text{Bin}(p, k)$. The empirical estimate for the success probability is $\hat{p} = \frac{s}{k}$.

Note that this naturally resembles marginal sums on our design matrix X , if we treat columns (or rows!) as an array of samples from independent Bernoulli trials: $\hat{p}_j = \frac{\sum_{i \in I} X(i, j)}{m}$. Many probability estimates involving repeated measurements of binary variables (not simply the row/column variables) have this sort of $\frac{\text{count}}{\text{exposure}}$ structure, as will become useful in later sections.

However, if we are “measuring” a probability, we run into issues when we need to quantify our uncertainty about it. For instance, an event might be quite rare, but if in our specific sample we *never* see it, we still do not generally accept a probability of zero.

2.3.1 Additive Smoothing

One approach to dealing with this involves adding *pseudocounts* that smooth out our estimates for count/exposure, from which we get the name “additive smoothing”.

ing".[CITE?]

$$\hat{p} = \frac{s + \alpha}{k + 2\alpha}$$

Adding 1 success and 1 failure ($\alpha = 1$) as pseudocounts to our observations is called *Laplace's Rule of Succession*, or simply “Laplace smoothing,”⁵ while adding $\alpha = 0.5$ successes and failures is called using *Jeffrey’s Prior*. It’s so-called because this pseudocount turns out to be a special case of selecting a Beta prior on the Bernoulli probability $p \sim \text{Beta}(\alpha, \beta)$, such that the posterior distribution for p after our observations is $\text{Beta}(s + \alpha, k - s + \beta)$, which has the mean:

$$E[p | s, k] = \frac{s + \alpha}{k - \alpha + \beta} \tag{2.2}$$

This exactly recovers additive smoothing with a Jeffrey’s prior for $\alpha = \beta = 0.5$.⁶ This generalization allows us to be more flexible, and specify our prior expectations on counts or exposure with more precision. Such models provide both an estimate of the aleatory uncertainty (via the posterior distribution), and a form of “shrinkage” that prevents sampling noise from unduly affecting parameter estimates (via the prior distribution). Despite being a simple foundation, this treatment of “counts” and “exposure” can be built upon in many ways.

⁵derived when Laplace desired estimates of probability for unobserved phenomena, such as the sun (not) rising tomorrow.

⁶A useful comparison of the two priors (1, 0.5) is to ask, given all of the trials we have seen so far, whether we believe we are near the “end” or “middle” of an average run of trials. For $\alpha = 1$, we believe nearly all evidence has been collected, but for $\alpha = 0.5$, only half of expected evidence has been observed.

2.3.2 Conditional Probabilities & Contingencies

In dependency/structure recovery, since our goal involves estimating (at least) pairwise relationships, the independence assumption required to estimate node occurrences as Beta-Binomial is clearly violated.⁷

However, it's common to estimate how similar two random variables A, B are, e.g. if samples of each correspond to columns of binary X . For instance, the joint probabilities $P(A \cap B)$ answer "how often does A happen with B, out of all data?"

Conditional probabilities $P(A | B) = \frac{P(A \cap B)}{P(B)}$ measure how often A occurs given B happened. Once again, we can estimate the base probabilities $P(A)$ and $P(B)$ with methods like Equation 2.2 for each marginal sums $X(\cdot, A)$ or $X(\cdot, B)$, but the joint and conditional probabilities can instead be estimated using matrix multiplication using the Gram matrix, discussed below. It encodes pair-wise co-occurrence counts, such that $G(i, i') : \mathbb{Z}^{n \times n}$ has the co-occurrence count for node i with i' .

For binary data, we typically create association measures from values on a 2×2 contingency table, with counts and marginals. A shorthand notation for these values is:

⁷In fact, a recent method from [46] models probabilistic binary observations, *with dependencies*, by generalizing the mechanics overviewed here to a fully multivariate Bernoulli distribution, capable of including 3rd- and higher-order interactions, not just pairwise.

	$B = 1$	$B = 0$	Σ_B	
$A = 1$	p_{11}	p_{10}	$p_{1\bullet}$	(2.3)
$A = 0$	p_{01}	p_{00}	$p_{0\bullet}$	
Σ_A	$p_{\bullet 1}$	$p_{\bullet 0}$	p_{\bullet}	

The co-occurrence probability $p_{11} = P(A \cap B)$ for each pair can also be approximated with the beta-binomial scheme mentioned above, but care must be taken not to confuse this with the edge strength connecting two nodes. First, nodes that rarely activate (low node probability) may nonetheless reliably connect to others when they do occur (high edge probability). In fact, without direct observation of edges, we are not able to estimate their count, *or* their exposure, which can be a source of systemic error from *epistemic uncertainty*. We don't know when edges are used, directly, and we also don't have a reliable way to estimate the opportunities each edge had to activate (their exposure), either. This is especially true when we wish to know whether an edge even *can* be traversed, i.e. the edge *support*. Support, as used in this sense, is the set of inputs for which we expect a non-zero output. Intuitively, this idea captures the sense that we might care more about *whether* an edge/dependency exists, not *how important* it is. For that, we have to re-assess our simple model: even if we could count the number of times an edge might have been traversed, how do we estimate the opportunities it had to be available for traversal (it's “exposure”)?

Assuming this kind of epistemic uncertainty can be adequately addressed through modeling—attempts at which will be discussed in more detail in Chapter 4—conditional probability/contingency tables will again be useful for validation.

When comparing estimated edge probability to some known “true” edge existence (if we have that), we can count the number of correct predictions, as well as type I (false positive) and type II (false negative) errors. We can do this at *every probability/weight threshold value*, as well, and we will return to ways to aggregate all of these values into useful scoring metrics in Section [6.4](#).

2.4 Distance vs. Incidence

As we have already seen, operations from linear algebra make many counting and combinatoric tasks easier, while unifying disparate concepts to a common set of mechanics. In addition to having a map from integer indices to sets of interest, these design matrices/vectors are implicitly assumed to have entries that exist in a field $F = (\mathbb{S}, \oplus, \otimes)$. equipped with operators analogous to addition (\oplus) and multiplication (\otimes).⁸ With this, we are able to define generalized inner products that take pairs vectors in a vector space $\mathbf{x} \in V$, such that $\langle \cdot, \cdot \rangle_F : \mathbb{S}^n \times \mathbb{S}^n \rightarrow \mathbb{S}$.

$$\langle \mathbf{x}_a, \mathbf{x}_b \rangle_F = \bigoplus_{j=1}^n \mathbf{x}_a(j) \otimes \mathbf{x}_b(j)$$

We can use this to perform “contractions” along any matching dimensions of matrices as well, since the sum index is well-defined.

$$X \in \mathbb{S}^{m \times n} \quad Y \in \mathbb{S}^{n \times m}$$

$$Z(i,j) = X \oplus, \otimes Y = \bigoplus_{j=1}^n X(i,j) \otimes Y(j,k) = XY$$

⁸Or, more generally, a semiring if inverse operations for \oplus, \otimes don't exist.

For ease-of-use, we will assume the standard field for any given set $(\mathbb{S}, +, \times)$ if not specified otherwise, which recovers standard inner products $\langle \cdot, \cdot \rangle$. However, Kepner et al. [38] illustrates the usefulness of various fields (or semirings). They allow linear-algebraic representation of many graph operations, such as shortest paths through inner products over $(\mathbb{R} \cup -\infty, \min, +)$. This works because discrete/boolean edge weights will not accumulate extra strength beyond 1 under contraction over observations.

2.4.1 Kernels & distances

As alluded to in the previous section, co-occurrence have a deep connection to a Gram matrix, which is a matrix of all pairwise inner products over a set of vectors.

$$X^T X = G(j, j') = \langle \mathbf{x}'_j, \mathbf{x}'_{j'} \rangle = \sum_{i=1}^m X(i, j) X(i, j') \quad (2.4)$$

Matrices that can be decomposed into another matrix and its transpose are symmetric, and positive semidefinite (PSD), making every gram matrix PSD. They are directly related to (squared) euclidean distances through the polarization identity[CITE?]:⁹

$$d^2(j, j') = \|\mathbf{x}'_j - \mathbf{x}'_{j'}\|^2 = G(j, j) - 2G(j, j') + G(j', j') \quad (2.5)$$

⁹**Important:** these definitions are all using the \mathbf{x}' notation to indicate that these measurements are almost exclusively being done in the *data space*, i.e. on column vectors. While most definitions work on distances in terms of the measurements/objects/data, for *inverse problems* (like network recovery, structure learning, etc.) they are more often applied in terms of the features (here, the nodes). This can be seen in statistics as well, where covariance and correlation matrices (which are related to the gram and distance matrix definitions above), are defined as relationships between features/dimensions, not individual samples.

In our example from before, the gram matrix will have entries showing the number of papers shared by two authors (or total papers by each, on the diagonal). This is because an inner product between two author (column) vectors will add 1 for each paper in the sum only if it has both authors in common. This is called a *bipartite projection*[CITE] into the authors “mode”, and is illustrated visually in Figure 1.3a.

Due to [CITE Shoenberg/Mercer], we can generalize Equation 2.5 such that *any* function “kernel” function $\kappa(x, y)$ that creates PSD matrix $K(j, j') \in \mathbb{S}^{n \times n}$. It says that such a PSD matrix can always be decomposed into a form $K = R^T R$ for any matrix $R(i, j) \in \mathbb{S}^{m \times n}$, thus letting us use the polarization identity to create arbitrary distance metrics. on \mathbb{S}^n [22]¹⁰

$$d_K(j, j') = \frac{1}{2} (K(j, j) + K(j', j')) - K(j, j') \quad (2.6)$$

This ability to create valid distance measures from arbitrary kernel functions is the core of a vast area of machine learning and statistics that employs the so-called *kernel trick*. [CITE?] Different kernels yield different properties useful for distinguishing points having specific properties. One class of kernels are normalized to the range $[0, 1]$, such that we ensure that equality along any one dimension is given a weight of $\tilde{K}(j, j) = 1$. Such a kernel matrix can be derived from any other kernel, and is often

¹⁰Distance metric, here, means that $d(x, y)$ satisfies the triangle inequality for all x, y .

combined with a logarithmic similarity measure $\tilde{\kappa}(x, y) = \log s(x, y)$.

$$\begin{aligned}\tilde{K}(j, j') &= \frac{K(j, j')}{\sqrt{K(j, j)^2 K(j', j')^2}} \\ d_{\tilde{K}}(j, j') &= -\log \tilde{K}(j, j')\end{aligned}\tag{2.7}$$

Since this is equivalent to applying Equation 2.6 to s directly. This normalization should be familiar as the way cosine similarities and correlation matrices are made as well (also having 1s along their diagonal), and illustrates how non-metric similarities can be potentially made into (pseudo)metrics.

2.4.2 Incidence structures & dependency

Rather than how “close” or “far” to points are in vector space, which is described with the kernels and distances above, whether something “touches”—or, is incident to—something else is usually described abstractly as an *incidence structure*. This is an abstract way to describe how things “touch”, such as when a set of points lie on a line/plane, or nodes touch an edge. We say an incidence structure is a triple of sets called (for historical reasons) *points* \mathfrak{P} , *lines* \mathfrak{L} , and *flags* \mathfrak{I} .[62]

$$(\mathfrak{P}, \mathfrak{L}, \mathfrak{I}) \quad \mathfrak{I} \subseteq \mathfrak{P} \times \mathfrak{L}$$

Representing these as matrices will be further explored in Section 3.1. But, the discrete nature of these incidence sets makes it clear that estimating the size and elements of \mathfrak{I} , is a very different question from estimating the similarity/distance

between two entities in a vector space.

In statistics, such discrete structures usually arise when we are concerned with direct dependence from indirect. Take as an example a set of masses connected together by springs. If we shake one mass, all masses will also shake some amount, depending on the spring constants of the springs each mass is “transmitted” force through, and the losses due to friction or air resistance. While the amount of movement over time depends on how “close” in this spring network two masses are, the movement itself can only be transmitted through springs that the masses are *incident* to. Movement could be modeled through similarity/distance measurements like correlation, since none of the masses are independent (all move when any do), but incidence in terms of spring force transmission is modeled in terms of conditional (in)dependence. If we hold all but two masses still, and moving one doesn’t move the other, then we know they are conditionally independent: no spring connects them!

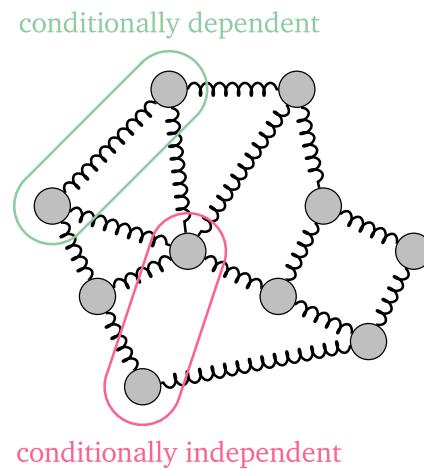


Figure 2.1: Spring system as a network of conditional dependencies

This idea gets formalized as probabilistic graphical models, which are networks that define “incidence” between two variables as conditional dependence. Letting

the random variables in the column-space of X be A, B and the remaining columns be $C = X(\cdot, J \setminus \{A, B\})$, then

$$P(A \cap B | C) = P(A | C)P(B | C) \implies (A \perp B | C) \implies (A, B) \notin \mathfrak{I} \quad (2.8)$$

for a set of incidences \mathfrak{I} defining a PGM that was sampled as X .

Chapter 3: Vector representations of incidence

To provide a sufficiently rigorous foundation for network recovery from binary occurrence, we will need a rigorous way to represent networks and occurrences that lends itself to building structured ways both connect to each other. We build on the incidence structure and matrix product formalism from the previous chapter, introducing various ways to build graphs as incidence structures that have direct representations as matrices. This can be extended to representing occurrences as matrices of *hyperedge vectors*. This view allows us to interpret different representations of graphs (or hypergraphs) as connected by simple matrix operations.

Traditionally[14, 38], we might say a graph on nodes (or, “vertices”) $v \in V = \{1, \dots, n\}$ and “edges” E is a tuple:

$$G = (V, E) \quad \text{s.t.} \quad E \subseteq V \times V$$

In incidence geometry terms, this would be similar to making two duplicate sets of the same nodes, and defining a graph as the set of incidences between nodes. The *adjacency matrix* A of G , degree matrix D , and graph/discrete Laplacian L are then

defined as:¹

$$A(u, v) = \mathbf{1}_E((u, v)) \quad A : V \times V \rightarrow \mathbb{B}$$

$$D(u, v) = \text{diag}(\sum_V A(u, v)) \quad D : V \times V \rightarrow \mathbb{N}$$

$$L(u, v) = D(u, v) - A(u, v) \quad L : V \times V \rightarrow \mathbb{Z}$$

However, if edges and their recovery is so important to us, defining them explicitly as nodes-node incidences can be problematic when we wish to estimate edge existence (or not), given noisy pairs of node co-occurrences. Additionally, we have to be very careful to distinguish whether our graph is *(un)directed, weighted, simple*, etc., and hope that the edge set has been filtered to a subset of $N \times N$ for each case. Instead, we propose a less ambiguous framework for vectorizing graphs, based on their underlying incidence structure.

3.1 Graphs as incidence structures

Instead, we give edges their own set of identifiers, $e \in E = \{1, \dots, \omega\}$. Now, define graphs as incidence structures between nodes and edges such that every edge is incident to either two nodes, or none:

$$G = (V, E, \mathcal{I}) \quad s.t. \quad \mathcal{I} \subseteq E \times V \tag{3.1}$$

Variations on graphs can often be conveniently defined as constraints on \mathcal{I} :

- Self loops can be prohibited by only allowing unique flags for a given relation²

¹The *indicator function* $\mathbf{1}_A(x)$ is 1 for values of x in the set A , and 0 otherwise.

²never two flags with the same pair, i.e. \mathcal{I} is a set, not a multiset.

- Multigraphs are similarly described by whether we allow pairs of vertices to appear with more than one edge³

Together, these constraints define “simple” graphs. Similarly, we can equip Equation 3.1 with a function B that allows \mathcal{I} to encode information about the specific kinds of incidence relations under discussion. We give B a range of possible flag values S :

$$G = (V, E, \mathcal{I}, B) \quad s.t. \quad \mathcal{I} \subseteq E \times V \quad B : \mathcal{I} \rightarrow S \quad (3.2)$$

- Undirected, unweighted graphs only need single elements: “incidence exists” i.e. $S = \{1\}$
- Directed graphs can use two elements e.g. a “spin” for $S = \{-1, 1\}$
- Weighted, undirected graphs are supported on positive scalars e.g. $S = \mathbb{R}^+$
- Weighted, directed graphs are supported on any scalar e.g. $S = \mathbb{R}_{\neq 0}$

If the “absence” of incidence needs to be modeled explicitly, a “null” stand-in (0, False) can be added to each S , which is useful for representing each structure as arrays for use with linear algebra (i.e. $\{0, 1\}$, $\{-1, 0, -1\}$, \mathbb{R}_0^+ , and \mathbb{R} , respectively). By doing so, we can also place an exact limit on the maximum possible size of $\omega = \|E\|$ in the simple graph case, and indicate edges by their unique ID, such that $\mathcal{I} = E \times V$ is no longer a subset relation for $E = \{1, \dots, \binom{n}{2}\}$. Instead of existence in \mathcal{I} , we explicitly use incidence relation S to tell us whether each possible edge “exists” or not, simplifying our graph definition further⁴:

³in the set of flags containing nodes u or v , only one e may be incident to both of them.

⁴if we allow multi-edges , then

$$G = (V, E, B) \quad s.t. \quad B : E \times V \rightarrow S$$

$$v \in V = \{1, \dots, n\} \tag{3.3}$$

$$e \in E = \left\{ 1, \dots, \binom{n}{2} \right\}$$

The representation of B in Equation 3.3 bears a remarkable similarity to our original description of design matrices in Equation 2.1. In fact, as a matrix, $B(e, v)$ is called the *incidence* matrix: every row has two non-zero entries, with every column containing a number of non-zero entries equal to that corresponding node's degree in G . Traditionally, we use an *oriented* incidence matrix, such that each row has exactly one positive (non-zero) value, and one negative (non-zero) value.⁵ Even for undirected graphs, the selection of *which entry* is positive or negative is left to be ambiguous, since much of the math used later is symmetric w.r.t. direction⁶.

3.1.1 Embedding incidences in vector space

A formalism for graphs that starts with incidence matrices would benefit from a *canonical* oriented incidence matrix, rather than the family that is ambiguous w.r.t. edge orientation. To start, we can be more precise by selecting each row(edge) vector, and partitioning it into two: one for each non-zero column (node) that edge is incident to. Every incidence can be represented individually as standard basis vector e in the feature space of B , scaled by the corresponding value of S .

Let V_e be the set of nodes with (non-zero) incidence to edge e . Then the incidence

⁵In fact, this would make $B \wedge^*(v, e)$ equivalent to a *graphical matroid*, another common formalism that generalizes graph-like structures to vector space representations.

⁶though not always!

vectors are:

$$\delta_e(v) = B(e, v)\mathbf{e}_v \quad \forall v \in V_e \quad (3.4)$$

And the (unoriented) incidence matrix vectors are recovered as sums over the incidence vectors for each edge:

$$\mathbf{b}_e = \sum_{v \in V_e} \delta_e(v) \quad (3.5)$$

A traditional approach might then define undirected graphs as equivalent, in some sense, to multidigraphs, where each edge is really two directed edges, in opposing directions. This does allow the matrix B to have the correct range for its entries in this formalism (the directed graph range, $S = \{-1, 0, 1\}$), and the edge identity based on sums would hold. However, the resulting set of incidences would have twice the number of edges than our combinatoric limit for simple graphs, and prevent the more elegant definition of graph types through the set \mathbf{S} . Plus, it would necessitate averaging of weights over different edge ID's to arrive at a single undirected “edge weight”, and many other implementation details that make keeping track of specifics difficult for practitioners.

Instead, we would like a canonical oriented distance matrix, which can be derived from the vectorized incidences in the undirected range of B (the standard basis vectors). Without loss of generality, let $u_e, v_e \in V_e$ be nodes such that $u < v$.⁷ Using this, we can unambiguously define a *partition* $B(e, \cdot) = B(e, u_e) + B(e, v_e)$ between incidences on e , along with a new derived incidence, B_o , which has oriented rows

⁷the inequality is strict because self-loops are not allowed.

like:

$$B_o(e, \cdot) = \mathbf{b}_e^o = \delta_e(u) - \delta_e(v)$$

In other words, while the unoriented incidence matrix is the “foundational” representation for graphs in our formalism, the (canonical) oriented one can be derived, even if negative incidence values are not in \mathbb{S} .⁸

But, now that we have removed the information on “which nodes an edge connects” from our definition of edges (since every edge is a scalar ID), how do we construct V_e without a circular dependency on B to find non-zero entries? Because of our unique identification of edges up to the combinatoric limit, we can still actually provide a unique ordering of the nodes in V_e , without searching over the entirety of B ’s domain. Using an identity from Angeletti et al. [21], we have a closed-form equation both to retrieve the IDs of nodes u, v (given an edge e), and an edge e (given two nodes u, v), for any simple graph with n vertices.

$$\begin{aligned} u_n(e) &= n - 2 - \left\lfloor \frac{\sqrt{-8e + 4n(n-1) - 7} - 1}{2} \right\rfloor \\ v_n(e) &= e + u_n(e) + 1 - \frac{1}{2} \left(n(n-1) + (n - u_n(e))^2 - n + u_n(e) \right) \\ e_n(u, v) &= \frac{1}{2} \left(n(n-1) - ((n-u)^2 - n + u) \right) + v - u - 1 \end{aligned} \tag{3.6}$$

Our ease-of-calculation lets us drop some of the excess notation and refer to our (un)oriented incidence matrices in terms of the incidences of each edge on their u or

⁸This works as long as we are in at least a ring, since semirings in general do not need to define additive inverse operations. In this case we would limit ourselves to the oriented incidence.

v , directly.

$$B = B_u + B_e \quad B_o \equiv B_u - B_v$$

3.1.2 Inner products on B

With all of this background, the other representations of graphs can seen as derivations from the canonical incidence matrices. The Laplacian, which is usually introduced either in terms of adjacency/degree, or as the gram matrix for oriented edge vectors, is also related to the gram matrix between all pairs of incidences on (u, v) . The other identities are simply consequences of the polarization identity:

$$\begin{aligned} L &= B_o^T B_o \\ &= \|B_u - B_v\|^2 \\ &= 2\|B_u\|^2 + 2\|B_v\|^2 - \|B_u + B_v\|^2 \\ &= 2D - B^T B = D - A \end{aligned} \tag{3.7}$$

The Laplacian is often used in defining random-walks and markov chains, such that the degree of each node should be normalized to 1, which can be accomplished either by row- or column-normalizing it: $L^{rw} = D^{-1}L$ or LD^{-1} , respectively. If this degree-normalization is desired without de-symmetrizing L , we can still perform an operation similar to normed kernels in Equation 2.7, called the symmetric normalized Laplacian.

$$\tilde{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = \frac{L(u, v)}{\sqrt{L(u, u)L(v, v)}} \tag{3.8}$$

3.1.3 Edge Metrology, Edge Vectors

Implicitly in the use of B with design matrix mechanics from the previous chapter is a treatment of edges as “observations” (the data space), and nodes as features. If an edge *is* an observation, then unfortunately we cannot really quantify uncertainty over repeated measurements of edges with the simple mechanics from Section 2.3.1 (because that edge *is* that corresponding observation, and IDs are not duplicated).

So far we have seen two ways of representing the entire graph in matrix form: Incidence matrix B (or B_o), and the inner-product matrices derived from it (L, A). Since we can recover node IDs from edge IDs by Equation 3.6, we can use a single vector to represent an entire graph structure by its edges alone. Then a data dimension for vectors in “edge space” can once again represent observations, with nodes implied by Equation 3.6. This is either done by contracting B along the nodes (columns) dimension, or by *unrolling* the upper triangle of L or A according to Equation 3.6.⁹ If each vector represents a value of B associated with a corresponding edge, then m of these vectors would be equivalent to m observations of $\binom{n}{2}$ edges on the same set of

⁹Equation 3.9 uses an averaging operation to accomplish the contraction, but any reduction over the two nodes shared by an edge would accomplish the same, especially since we rarely see separate values for edge weight per-node, the way incidences can.

n nodes. Formally, for the i th observed structure on n nodes:

$$\begin{aligned}
R(i, e) &= \min(\{B_i(e, u_n(e)), B_i(e, v_n(e))\}) \\
\text{s.t. } R : I \times E &\rightarrow \mathbb{S} \\
i \in I = \{1, \dots, m\} \quad e \in E = \{1, \dots, \omega\} \\
n &= \frac{1}{2}(1 + \sqrt{8\omega + 1})
\end{aligned} \tag{3.9}$$

This representation formalizes what practitioners call “edgelists” into a data structure that can unambiguously distinguish directed, undirected, and weighted graphs. In addition, it allows for repeated measurements of edges over the same set of nodes, while flexibly growing when new nodes arrive.¹⁰

3.2 Node activation, bipartite graphs, and hypergraphs

What if an incidence structure allows for more than 2 incidences for the “line” set? In our binary design matrix, we might consider each observation its own “line”, such that it is incidence to all the activated nodes. This is no longer a graph of edges and nodes, but rather a more general object.

Every incidence structure can be seen as an incidence matrix, which additionally can be thought of as a *bipartite* graph. In this sense, the incidence matrix is thought of as a bi-adjacency matrix, which is a subset of a larger adjacency having two off-diagonal non-zero blocks

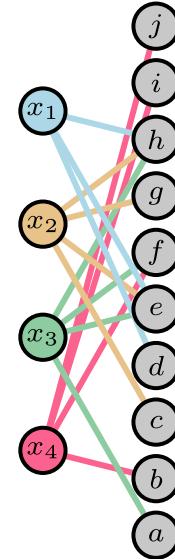
¹⁰For instance, say observations are stored as sparse entries via R , and a new node arrives. First, the participating nodes can be recovered in a vectorized manner through Equation 3.6. Then, a new node id increases n , followed by reassignment of the edge IDs with $e_n(u, v)$.

$$A_{BP} = \begin{pmatrix} 0_{n,n} & X^T \\ X & 0_{m,m} \end{pmatrix}$$

The graph having this adjacency structure has two sets of nodes that do not intraconnect (ergo, “bipartite”). The resulting structure for our toy example is shown next to the incidence matrix in fig Figure ??.

$$\begin{array}{c} \begin{matrix} & a & b & c & d & e & f & g & h & i & j \\ x_1 & \left[\begin{matrix} 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ x_2 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ x_3 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ x_4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ \vdots & & & & & \vdots & & & & \end{matrix} \right] \end{matrix} \end{array}$$

(a) : X as a (bipartite) incidence matrix.



(b) Bipartite representation of node “activation” data

When the set of lines is a family of subsets of points, incidences on points and lines form a *hypergraph*. Hypergraphs are usually thought of as graphs where edges can connect more than two nodes, which again can be made into an incidence structure in the same way our graphs are. This isomorphism lets many of the familiar ideas on graphs (e.g. walks, paths, laplacians, etc.) be reformulated in terms of hypergraphs. For a more in-depth elaboration on algebraic graph theory on graphs, see Reff and Rusnak [54].

3.2.1 Inner product on Hyperedges

Rather than go into similar detail about hypergraphs, we want to focus on the implication of performing inner product calculations with vectors of the hypergraph. $\mathbf{x}_i^T \mathbf{x}_{i'}$ will not necessarily result in a binary value, but a sum over all shared nodes in each edge. As mentioned previously, the Gram matrix for X will count co-occurrences in the off-diagonals, with marginal counts on the diagonal. This means that the relation between edge-vector entries and node-activations is not so straight forward: each entry has a “magnitude”, so forcing a binary edge activation from the adjacency or laplacian forms (using Equation 3.6) will necessarily lose information. This information loss is one of the key precautions practitioners are advised to consider in Torres et al. [14], since transformation between any of the representations for complex systems will change the information encoded by them.

3.2.2 Combining Occurrence & Dependence

As an aside, there are several ways that known graph structures can be exploited to regularize distance or association measures for hypergraphic data. Graphs of dependency information can provide useful kernels *on the graph*.¹¹ For an in-depth analysis of common kernels on graphs, including the now-famous PageRank, Heat, and Commute-time kernels, see Avrachenkov et al. [22]. In Coscia [16] the “commute time” kernel is said to generate a “generalized euclidean distance”, and using an adjacency matrix kernel before normalization by marginals gives what is known

¹¹Note that a “kernel on a graph” works by defining distances between weighted vectors of nodes. A “graph kernel,” on the other hand, describes a kernel that defines distances *between different graphs*.

as “soft cosine” measures, which incorporate prior knowledge e.g. for semantic similarities between words.

Importantly, each of these methods assume that a network structure is already available. What can often happen in practice, however, is the use of a filtered/thresholded version of the hypergraphic Gram matrix *as the network*, because no a priori network is available. This runs the risk of over-estimating the proximity of nodes in similar regions of node-space, reinforcing local correlations at the expense of long-distance path estimates. A more formalized analysis of this problem, which we term “clique bias,” can be found in Section 5.1.

As covered in the introduction, finding the underlying dependency network from the hypergraphic data is the core concern of network recovery. Reliably recovering this structure would then allow for less biased use of these graph kernels in real-life application.

Chapter 4: Roads to Network Recovery

Here we give a brief overview of the key approaches to backboning and dependency recovery for networks through binary activations. We organize the literature into categories based on the kinds of constraints that have been applied to make the network reconstruction “inverse problem” tractable: local structure, information-flow/resource constraints, or global structure. Finally, we assess patterns in the assumptions made by the presented algorithms, and motivate the need for a new approach to fill a perceived gap in the network recovery space.

4.1 Organizing Recovery Methods

All recovery methods will require assumptions in addition to data to accomplish their task. As discussed in Torres et al. [14], there are fundamental difference in dependency capability between the network and hypergraphic/bipartite representations of complex systems. Necessarily, some information will be lost in translation between the two forms.

As a (hopefully) helpful way to organize the various *kinds* of assumptions that are taking bipartite observations to simple graphs, we present an organization of common modeling assumptions into three loosely-defined groups:

- Local Structure & Additivity
- Information Flow & Resource Constraints
- Global Structure Models

In truth, this classification should be viewed as more of a sliding scale, with approaches falling somewhere within. Some approaches make very few assumptions about the shape a resulting network “should” take, but do so by making strong assumptions about how individual observations relate to a desired quantity, and especially how those observations are able to be combined to result in an “answer”, however that looks. Others instead provide clear normative constraints on the overall network topology, or emission mechanism, but this allows for flexibility in how data is individually handled.

This idea could be thought of as similar to pooling in bayesian inference. Do individuals all have fundamentally separate distributions, so that global behavior (and by extension, uncertainty) is an aggregate phenomena? Or do individuals inherit parameters from a global distribution shared by all, and anything outside that structural assumption must be “noise”? In between the extremes, some other assumption as to how the global and local scales mitigate information between them is required, i.e. *partial pooling*. In this domain, what we often see are attempts to perform noise corrections through the way information is thought to travel between nodes, generally.

For each, we provide examples to illustrate modeling patterns and highlight common practice, though for a deeper assessment of the broad space of backboning

and edge prediction in general a reader may see the overview in Coscia [8].

4.1.1 Local Structure & Additivity Assumptions

These are, together, typically called “association” measures. While they are sometimes presented as functions of the entire dataset, they nearly always find a basis in the inner-product operation, and have definitions in terms of *contractions* along the data/observation dimension. By relying on the (Euclidean) inner product, even with various re-weighting or normalization schemes, an analyst is making strong assumptions about their ability to reliably take measurements from linear combinations of observed activation vectors.

Essentially, if a measure relies on marginal counts or summation over the data axis, then the main assumptions are at the *local* level, about whether what we are adding together estimates our target correctly. The most basic would be to count co-occurrences, and consequently the co-occurrence probability $p_{11} = P(A, B)$. However, for very rare co-occurrences, we need to correct for rate-imbalance of the nodes in much the same way correlation normalizes covariance. This idea leads to “cosine similarity”

Note 1: Ochiai Coefficient (Cosine)

Effectively an uncentered correlation, but for binary observations the “cosine similarity” is also called the *Ochiai Coefficient* between two sets A, B , where binary “1” stands for an element belonging to the set.[84] In our use case, it is

measured as

$$\frac{|A \cap B|}{\sqrt{|A||B|}} = \sqrt{p_{1\bullet}p_{\bullet 1}} \rightarrow \frac{X^T X}{\sqrt{\mathbf{s}_i \mathbf{s}_i^T}} \quad \mathbf{s}_i = \sum_i \mathbf{x}_i$$

This interpretation of cosine similarity as the geometric mean of conditional probabilities is particularly useful when trying to approximate interaction rates. The geometric mean as a pooling operator is conserved through Bayesian updates [49], so the use of a prior with co-occurrences as base counts is possible for additive smoothing. To do this, the geometric mean of marginal counts acts as a “psedovariable” for exposure somewhere between A and B. Empirically, this is a powerful approximation with good performance characteristics, for relatively little effort.

Note 2: Odds Ratios

Along with others derived from it, the Odds ratio is based on the ratio of conditional probabilities, and takes the form

$$OR = \frac{p_{11}p_{00}}{p_{01}p_{10}}$$

Yule's Q and Y [92] are mobius transforms of the (inverse) OR and \sqrt{OR} , respectively, that map association values to $[-1, 1]$.

$$Q = \frac{OR - 1}{OR + 1} \quad Y = \frac{\sqrt{OR} - 1}{\sqrt{OR} + 1}$$

Odds ratio is important to logistic regression, where the coefficients are usually

the log-odds ratios of occurrence vs. not (log OR).

Yule's Y, also called the “coefficient of colligation”, tends to scale with association strength in an intuitive way, so that proximity to 1 or -1 paints a more useful picture than the odds-ratio alone.

Another association measure, based in information theory, asks “how much can I learn about one variable by observing another?”

Note 3: Mutual information

An estimate for the mutual information (i.e. between the sample distributions) can be derived from the marginals, as above, though it is more compactly represented as a pairwise sum over the domains of each distribution being compared[CITE?]:

$$MI(A, B) \approx \sum_{i,j \in [0,1]} p_{ij} \log \left(\frac{p_{ij}}{p_{i\bullet} p_{\bullet j}} \right)$$

It is non-negative, with 0 occurring when A and B are independent. There are many other information-theoretic measures related to MI, but we specifically bring this up as it will be the basis for the Chow Liu method, later on.

Sometimes, especially in social networks, we might want to avoid overcounting relationships with very well-connected nodes. This was brought up with respect to the normalized Laplacian before, but we could also perform a normalization on the underlying bipartite adjacencies.

Note 4: Hyperbolic Projection

Attempts to account for the overcounting of co-occurrences on frequently occurring nodes, vs. rarer ones.[[75](#)]

$$X^T \text{diag}(1 + \mathbf{s}_j)^{-1} X \quad \mathbf{s}_j = \sum_j \mathbf{x}'_j$$

This reweights each observation by its degree in the original bipartite graph.

So far this is the first measure that re-weights observations *before* contraction, so that it depends on having the individual observations available (rather than only the gram matrix). In this case, each observation's entries are all equally re-weighted by the number of activations in it (each node's "activation fraction" in that observation).

4.1.2 Resource and Information Flow

These methods are somewhere between local and global constraint scales. This is accomplished by imagining nodes as having some amount of a resource (like information or energy) and correcting for observed noise in edge activation by reinforcing the *geodesics* that most likely transmitted that resource.

First, closely related to hyperbolic re-weighting, we can imagine the bipartite connections as evenly dividing each nodes' resources, before reallocating them to the nodes they touch, in turn. For instance, we might say each author splits their time among all of the papers they are on, and in turn every co-author "receives" an evenly divided proportion of everyone's time they are co-authoring with.

Note 5: Resource Allocation

Goes one step further than hyperbolic projection, by viewing each node as having some “amount” of a resource to spend, which gets re-allocated by observational unit. [64]

$$\text{diag}(\mathbf{s}_i)^{-1} X^T \text{diag}(\mathbf{s}_j)^{-1} X$$

Interestingly, we could see this as a two-step random-walk normalization of the bipartite adjacency matrix First X is row-normalized, then column-normalized. The final matrix is asymmetric, so a symmetric edge strength estimate is often retrieved by mean, max, or min reduction operations.

Rather than stop after two iterations, continuing to enforce unit marginals to convergence is known as the Sinkhorn-Knopp algorithm, which converges to a doubly-stochastic matrix (both marginal directions sum to 1).

Note 6: Doubly Stochastic

If $A \in \mathbb{S}^{n \times n}$ is positive, then there exists d_1, d_2 such that

$$W = \text{diag}(d_1) A \text{diag}(d_2)$$

is doubly-stochastic, and $W(u, v)$ is the *optimal transport plan* between u and v with regularized Euclidean distance between them on a graph.[4, 45]

The doubly-stochastic filter [59] removes edges from W until just before the graph would become disconnected.

As the name implies, the optimal transport plan reflects the minimum cost to move some amount of resource from one node to another. By focusing on best-case cost, we enforce a kind of “principle of least action” to bias recovery toward edges along these geodesics.

A more direct way to do this, perhaps, is to find the shortest paths from every node to each other node, and aggregate them.

Note 7: High-Salience Skeleton

Count the number of shortest-path trees an edge participates in, out of all the shortest-path-trees (one for every node).

$$\frac{1}{n} \sum_{i=1}^n \mathcal{T}_{\text{sp}}(i)$$

where $\mathcal{T}_{\text{sp}}(n)$ is the shortest-path tree rooted at n [51]

Unfortunately, HSS is forced to scale with the number of nodes, and must calculate the entire spanning tree for each one.

4.1.3 Global Structural Assumptions

Often times these constraints are as simple as “the underlying dependency graph must belong to a family \mathcal{C} ” of graphs. Observations are thought of as emissions from a set of node distributions, where edges are representations of dependency relationships between them. To provide a foundation to formalize this notion, one framework is that of Markov Random Fields, which are undirected generalizations of bayes nets

[CITE?] that use edges to encode *conditional dependence* between node distributions.

One of the original structures for MRFs that we could recover from observed data was a *tree*.

Note 8: Chow-Liu Spanning Tree

Enforces tree structure globally. Approximates a joint probability

$$P\left(\bigcap_{i=1}^n v_i\right) \approx P' = \prod_{e \in T} P(u_n(e) | v_n(e)) \quad T \in \mathcal{T}$$

where \mathcal{T} is the set of spanning trees for the nodes. The Chow-Liu tree minimizes the Kullback-Leibler (KL) Divergence $\text{KL}(P \| P')$ by finding the minimum spanning tree over pairwise mutual information weights. [89]

Recent work has made it possible to enforce spanning tree structure while efficiently performing monte-carlo-style bayesian inference, which estimates a distribution over spanning trees that explain observed behavior, and by extension the likelihood each edge is in one of these trees. [9]

If instead we imagine our MRF as being made up of individual Gaussian emissions, then the overall network will be a multivariate gaussian with pairwise dependencies along the edges. In fact, as a consequence of the Hammersley–Clifford theorem, the conditionally independent variables are *precisely* the set of zero entries in the precision (inverse-covariance) matrix Θ of the multivariate model. Exploiting this fact leads to a semidefinite program to minimize the frobenius-norm of Θ with the

sample covariance $\|\hat{\Sigma}\Theta\|_F$ ¹

Note 9: GLASSO

Semidefinite program to find (regularized) maximum likelihood precision of graph-structured multivariate Normal distribution using ℓ_1 (“LASSO”) penalty [60]

$$\operatorname{argmax}_{\Theta \prec 0} \mathcal{L}(\Theta | \hat{\Sigma}) = \operatorname{argmin}_{\Theta \prec 0} \operatorname{tr}(\hat{\Sigma}\Theta) - \log |\Theta| + \rho \|\Theta\|_1 \quad (4.1)$$

In the binary case Equation 4.1 is still guaranteed to find the structure of the generating MRF, but *only for graphs with singleton separators*, as shown in Loh and Wainwright [53].

The “singleton separator” condition means that only MRFs structured like trees or block graphs will have graph-structured precision matrices returned by the GLASSO program. In effect (for the purpose of recovery from binary node activations) using GLASSO assumes either multivariate gaussian structure, or at the very least that all clique-separator sets are single-node. In practice, GLASSO is used for more than just this, but with theoretical miss-specification we must rely more on empirical validation²

There are many other models in this class, which provide strong global assumptions to make inference tractable. Not all look like structural assumptions on the graph structure itself, like *Degree Sequence Models*. [11, 42] They assume that the

¹Since the sample covariance will not give an unbiased estimate for precision, these problems often require significant regularization. This class of problems is called “covariance shrinkage”, though we more specifically care about *precision shrinkage* as illustrated in Note 9.

²something that this work attempts to begin addressing through standard reference datasets like MENDR, more on which is discussed in Section 6.4

fundamental property of these datasets is their bipartite node degree distributions, leading to a generative model that can sample bipartite adjacency matrices with similar observation/node degree distributions³

Still others assume that the graphical structure can be described as generated by *Stochastic Block Models*, a meta-network of communities and their inter-community connection probabilities. They don't constrain the graph class itself, but instead prescribe a generative process for graph creation (through community blocks). These have very nice properties for bayesian inference of structure, and can be modularly combined or nested for varying levels of specificity and hierarchical structure, sometimes with incredible computational efficiency. [23, 28]

4.2 A Path Forward

In addition to the categories above, there is a second “axis” that practitioners should keep in mind when selecting their recovery algorithm of choice. Each of the above listed techniques can be mostly separated into two categories, based on whether hypergraphic/bipartite observations are assumed to be in *data space*, or in *model space*.

4.2.1 Observation space assumptions

Recall from Section 2.2: an operator takes our model parameters and maps them to data space. The implication for inverse problems is a need to *remove the effect* of the operator, because we cannot directly observe phenomena in a way compatible with

³In that literature the “observation” partition’s nodes are typically called “artifacts” instead.

our model (e.g. which might model underlying causal effects).

This is a core point of view in Peel et al. [6], where those authors describe nearly all of network analysis as *inferring hidden structure*:

“Here we argue that the most appropriate stance to take is to frame network analysis as a problem of inference, where the actual network abstraction is hidden from view, and needs to be reconstructed given indirect data.”

Peel et al., [6]

We note that this isn’t strictly true, *assuming* that the “network” is intended to represent something measured by the direct observation. For instance, if a network is intended to represent a discretization of distances (such as a k-nearest neighbors approximation) for computational efficiency. The co-occurrence measures can be thought of as estimators of node-node distances, especially with appropriate smoothing to remove zero-valued distances from undersampling.⁴ In otherwords, if an analyst wishes to discretize distances as incidences in a complex network, they are effectively using “high-pass” filter to remove low-similarity entries, which is an effective way to assess community structure—exactly like clustering for continuous data.⁵

In fact, for an example of this exact network-as-discretization idea being used for state-of-the-art clustering performance, see HDBSCAN in Malzer and Baum [17].

The most important is for algorithm creators to transparently describe their

⁴See Section 6.2.2 for an elaboration of this connection via the “forest kernel”.

⁵Or, at a slight risk of reductionism, drawing a world atlas with two colors for “above and below sea-level”: useful simplification for rapid assessment of shapes.

technique's data-space assumption: are observations already in *model space*, perhaps with with aleatoric noise to be removed? Or are they in *data space* and require solving some form of inverse problem to recover a model specification? As stated in the same paper:

Surprisingly, the development of theory and domain-specific applications often occur in isolation, risking an effective disconnect between theoretical and methodological advances and the way network science is employed in practice.

Peel et al. [6]

This is a community and technology transfer problem, which standardization and community toolkit support can hopefully work toward fixing.

4.2.2 Filling the local+data “gap”

With this in mind, we show in Table 4.1 an overview of the covered approaches, and whether the method presumes operation on observations in the same space as the *model*, or if some inverse problem is needed.

To add to the point, the last column in Table 4.1 shows whether the full bipartite representation is even needed to perform the technique, or if it is possible with the gram matrix or marginal values alone.⁶

In the next sections, we focus on filling a gap for models that only make local assumptions and preserve additivity, but *assumes that data is not represented directly*

⁶note that any of these could make use of the bipartite “design matrix”, e.g. to estimate the edge support with *stability selection*[58] by subsampling it multiple times and repeating the algorithm accordingly.

Table 4.1: Organizing recovery methods by representation space and level

(a)

<i>observations in...</i>	Data space?	Model space?	Bipartite?
<i>assumptions for...</i>			
Local Structure			
Ochiai Coeff.		•	
Hyperbolic Proj.		•	•
Mutual Info.		•	
Information Flow			
Resource Allocation		•	•
Doubly Stochastic	?	•	
High-Salience Skeleton	•		
Global Structure			
Chow Liu	•		
MRF/GLASSO	•		
Deg. Sequence		•	•
S.B.M.	•		•

in the network model-space. Much like the role that nonparametric estimators like KDE/Nadarya-Watson play in regression, or Kaplan-Meier estimators in survival analysis,[CITE?] additive models that only make assumptions about local structure can provide critical insight into data and its structure, and push analysts to make regular “sanity checks” when results or assumptions conflict.

Part II

Nonparametric Network Recovery With Random Spanning Forests

Chapter 5: Latent Graphs with Desire Paths

Addressing gaps discussed in the previous section to reach a generative model for network recovery requires careful attention to the generation mechanism for node activations. While there are many ways we might imagine bipartite data to be generated, presuming the existence of a dependency graph that *causes* activation patterns will give us useful ways to narrow down the generative specification.

First, we will investigate the common assumption that pairwise co-occurrences can serve as proxies for measuring relatedness, and how this “gambit of the group” is, in fact, a strong bias toward dense, clique-filled recovered networks. Using the relationship between matrix products and sums of vector outer-products, we then motivate a generalization of co-occurrence estimation that can be flexibly adapted to domain knowledge, as appropriate, and avoid undue clique bias altogether. Finally, we propose a simple mechanism to frame network reconstruction as combinations of multiple overlaid subgraphs, by treating edges as i.i.d. Beta-Bernoulli random variables.

Inspired by the behavior of so-called “desire paths”, we constrain the beta prior to ensure desired behavior, which we call the *Desire Path Density* estimate of the global graph structure.

5.1 The Gambit of the Inner Product

As we saw repeatedly in Chapter 4, networks are regularly assumed to arise from co-occurrences, whether directly as counts or weighted in some way. This assumption can be a significant a source of bias in the measurement of edges. *Why* a flat count of co-occurrence leads to “hairballs” and bias for dense clusters can be related to the use of inner products on node activation vectors.

5.1.1 Gambit of the Group

It seems reasonable, when interactions are unobserved, to assume some evidence for all possible interactions is implied by co-occurrence. However, similar to the use of uniform priors in other types of inference, if we don’t have a good reason to employ a fully-connected co-occurrence prior on interaction dependencies, we are adding systematic bias to our inference. Whether co-occurrence observations can be used to infer interaction networks directly was discussed at length in Whitehead and Dufault [78], where Whitehead and Dufault call this the *gambit of the group*.

So, consciously or unconsciously, many ethnologists studying social organization make what might be called the “gambit of the group”: they assume that animals which are clustered [...] are interacting with one another and then use membership of the same cluster [...] to define association.

This work was rediscovered in the context of measuring assortativity for social networks,¹ where the author of Fisher et al. [33] advises that “group-based methods”

¹Assortativity is, roughly, the correlation between node degree and the degrees of its neighbors.

can introduce sampling bias to the calculation of assortativity, namely, systematic overestimation when the sample count is low.

The general problems with failing to specify a model of what “edges” actually *are* get analyzed in more depth in Peel et al. [6]. They include a warning not to naively use correlational measures with a threshold, since even simple 3-node systems will easily yield false positives edges. Still, it would be helpful for practitioners to have a more explicit mental model of *why* co-occurrence-based models yield systematic bias,

5.1.2 Inner-Product Projections and “Clique Bias”

Underlying correlation and co-occurrence models for edge strength is a reliance on inner products between node occurrence vectors. They all use gram matrices (or centered/scaled versions of them), which were brought up in Section 2.4. The matrix multiplication performed represents inner products between all pairs of feature vectors. For $X(i,j) \in \mathbb{B}$, these inner products sum together the times in each observation that two nodes were activated together.

However, another (equivalent) way to view matrix multiplication is as a sum of outer products

$$G(j,j') = X^T X = \sum_{i=1}^m X(i,j)X(i,j') = \sum_{i=1}^m \mathbf{x}_i \mathbf{x}_i^T$$

Those outer products of binary vectors create $m \times m$ matrices that have a 1 in every i,j entry where nodes i,j both occurred, shown in Figure 5.1. Each outer product is effectively operating as a $D_i + A_i$ with degrees normalized to 1. If the off-diagonals can be seen as adjacency matrices, they would strictly represent a clique on nodes

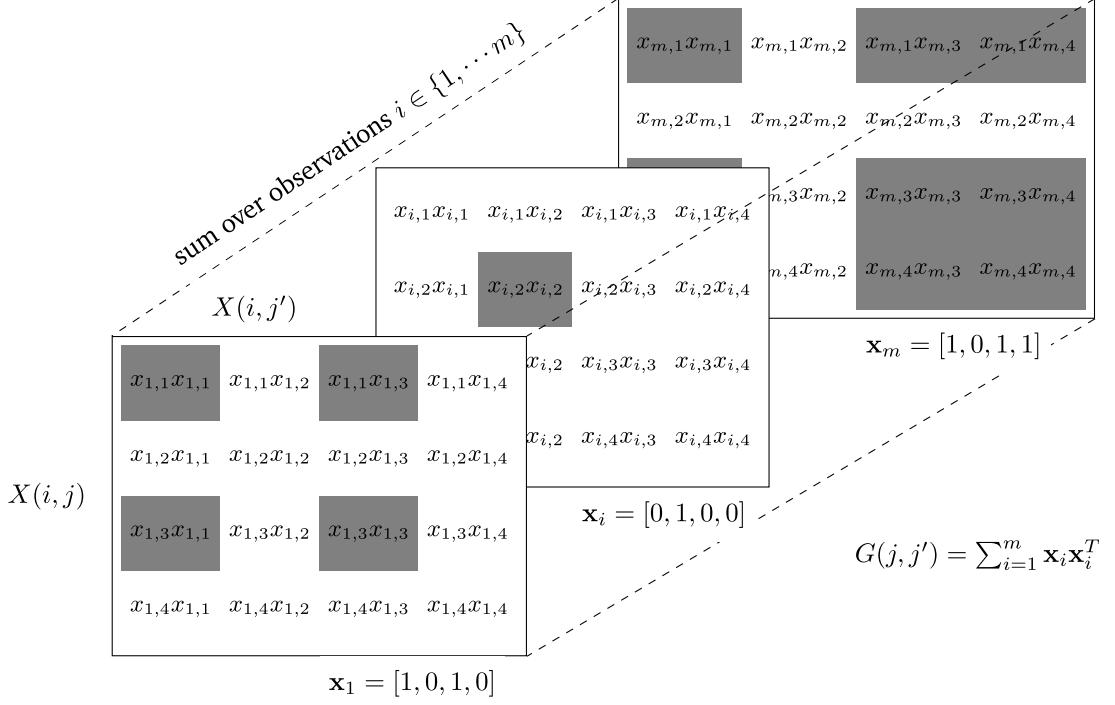


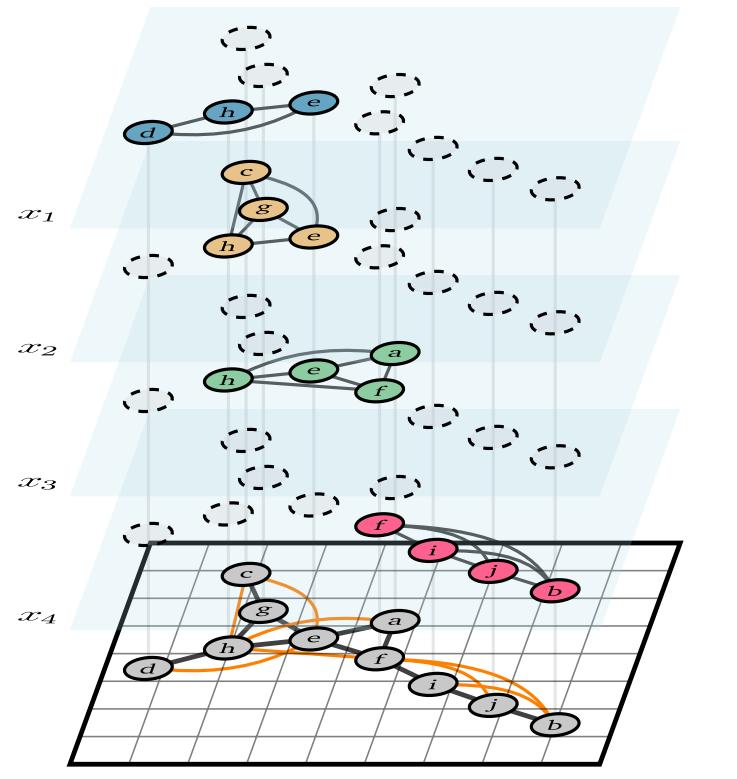
Figure 5.1: Gram matrix as sum of observation outer products

activated in the i th observation. In this sense, any method that involves transforming or re-weighting a gram matrix, is implicitly believing that the k th observation was a *complete graph*. This is illustrated in Figure 5.2.

For many modeling scenarios, this paradigm allows practitioners to make a more straight-forward intuition-check: do clique observations *make sense* here? When a list of authors for a paper is finished, does that imply that all authors mutually interacted with all others directly to equally arrive at the decision to publish? This would be similar to assuming the authors might simultaneously enter the same room, look at a number of others (who all look exclusively at each other, as well), and *all at once* decide to publish together. In our introduction, we described a more likely scenario we could expect from an observer on the ground: a researcher asks a colleague or two to collaborate, who might know a couple more with relevant expertise, and so

$$\begin{aligned}
 \{d, h, e\} &= x_1 \\
 \{g, c, e, h\} &= x_2 \\
 \{f, e, a, h\} &= x_3 \\
 \{i, j, f, b\} &= x_4
 \end{aligned}$$

(a) Observations as activation sets



(b) Edge Measurements with Group Gambit (BoW) assumption

Figure 5.2: Inner-product projections as sums of cliques illustrating “clique bias”.

on.

5.2 Networks as Desire Path Density Estimates

Unfortunately, methods based on inner-product thresholding are still incredibly common, in no small part due to how *easy* it is to create them from occurrence data, regardless of this “clique-bias”. The ability to map an operation onto every observation, e.g. in parallel, and then reduce all the observations into an aggregate edge estimate is a highly desirable algorithmic trait. This may be a reason so many projection and backboning techniques attempt two re-weight (but retain) the same basic structure, time and again.

What we need is a way to maintain the ease-of-use of inner-product network creation:

- map an operation onto each observation
- reduce to an aggregate edge guess over all observations

but with a more domain-appropriate operator at the observation level.

Let’s start with replacements for the clique assumption. There are many non-clique classes of graphs we might believe local interactions occur on: path-graphs, trees, or any number of graphs that reflect the topology or mechanism of local interactions in our domain of interest. Authors have proposed classes of graphs that mirror human perception of set shapes [RELATIVE NEIGHBORHOOD]², or graphs whose modeled

²e.g. for dependencies based on perception, such as human decision making tendencies, or causes based on color names.

dependencies are strictly planar [planar maximally filtered graphs]³. Alternatively, the interactions might be scale free, small-world, trees, or samples from stochastic block models.[CITE] In any case, these assumptions provide an explicit way to describe the set of *possible* interaction graphs we believe our individual observations are sampled from.

5.2.1 Subgraph Distributions

Let's use the notation from Equation 3.9, such that each observation of nodes \mathbf{x}_i is implicitly derived from a set of activated edges \mathbf{r}_i . To start, our current belief about what overall structure the whole network can take is $G^* = (V, E, B^*)$, where B^* might always return 1 to start out (the complete graph). To further constrain the problem, let us assume that node activation is noiseless: any activated nodes were truly activated, and unactivated nodes were truly inactive (no false negative or false positive activations).⁴ So, each \mathbf{x}_i will induce a subgraph $g_i = G^*[V_i]$, where $V_i = \{v \in \mathcal{V} | X(i, v) = 1\}$. Then, our domain knowledge takes the form of a constraint on edges within that subgraph, which will induce a family of subgraphs on g_i . This family \mathcal{C}_i belongs to the relevant class of graphs \mathcal{C} , limited to nodes V_i , i.e.

$$\begin{aligned}\mathcal{C}_i &= \{(V, E, B_i) \in \mathcal{C} | B_i(e, v) = B^*(e, v) \mathbf{1}_{V_i}(v) \mathbf{1}_{E_i}(e)\} \\ E_i &\in \{\mathcal{E} \in \mathcal{P}(E) | g_i[\mathcal{E}] \in \mathcal{C}\} V_i = \{v \in \mathcal{V} | X(i, v) = 1\}\end{aligned}\tag{5.1}$$

³e.g. when interactions are limited to planar dependencies, like inferring ancient geographic borders.

⁴Hidden nodes (unobserved nodes beyond the n) are outside the scope of this work, though could potentially be implied for certain structures e.g. when the metric is known to be tree-like. Sonthalia and Gilbert [20] use a greedy embedding that minimizes distortion to estimate the need for added *Steiner* nodes.

In other words, the edges that “caused” to the node activations in a given observation must *together* belong to a graph that, in turn, belongs to our desired class, which must occur on the nodes that were activated.

Assuming we can define an associated measure $\mu_i(c)$ on for each $c \in \mathcal{C}_i$, we are able to define a probability distribution over subgraphs in the class.⁵ Using notation similar to distributions over spanning trees in Zmigrod et al. [15]:

$$p_i(c) = \frac{\mu_i(c)}{Z_i} \quad (5.2)$$

$$Z_i = \sum_{c \in \mathcal{C}_i} \mu_i(c)$$

This can be represented using the vectorization in Equation 3.9, due to the one-to-one correspondence established, so that, with a slight abuse of notation treating \mathcal{C}_i as the parameter of distribution p_i :

$$\mathbf{r}_i(e) | \mathbf{x}_i \sim p_i(\mathcal{C}, E, V) \quad (5.3)$$

So we may not have an exact vector, but we have established a way to specify a family of edge vectors that could be responsible. With $p_i(c)$, we also have a mechanism to sample them when a partition function is available (or able to be

⁵ $\mathcal{P}(A)$ is the powerset of A , i.e. the set of all subsets of A .

⁶This is certainly not a trivial assumption, and might either be ill-defined or require techniques like the Gumbel trick[18] to approximate, unless the partition function Z has a closed form, or μ is already a probability measure on some σ -algebra over \mathcal{C} . Closed-form Z values on \mathcal{C} are known for a handful of graph classes, such as the space of spanning trees, $\mathcal{C} = \mathcal{T}$. However, another way this might be accomplished is through random geometric graphs[CITE], or geometric spanners like Urqhart[CITE] and Relative Neighborhood [CITE] graphs on a “sprinkling” of points that preserves their observed pairwise distances.

This is further discussed in Section 10.2.

approximated).

With these mechanics in place, we see the choice of a clique (implied by the inner product) is a trivial application of Equation 5.3, since that is equivalent to selecting the class of cliques on V_i nodes. This has only one element ($\|\mathcal{C}_{\text{clique}}\| = 1$), there is only 1 possible selection, with probability $p_i(K^{V_i}) = 1$.

5.2.2 Graph Unions as Desire Paths

With a distribution over subgraphs each observation, we are potentially able to sample from them for bootstrap or Monte Carlo estimation purposes, or simply find a maximum likelihood estimate for each distribution. Assuming this is true, we may now sample or approximate a matrix $R(i, e) : I \times E \rightarrow \mathbb{B}$.

Methods for doing this efficiently in certain cases are the focus of Section 6.3 and Section 7.2.2. However, once $R(i, e)$ is estimated, a reasonable mechanism for estimating the support of the set of edges is to use $\frac{\text{count}}{\text{exposure}}$, but with a few needed modifications.

First, while the nodes counts in $\sum_i B(i, \cdot)$ are by assumption *not* independent, or even pairwise independent, the *edge traversal* counts $\sum_i R(i, \cdot)$ could more reasonably be considered so. A model certainly could be constructed where edge existence depends on other edges existing (or not). But nothing is inherently self-inconsistent with a model that assumes the traversability of individual edges will be independent of one another.

let $P(e)$ be the probability that an edge is traversed (in any observation), and $P(u, v)$ the probability that nodes u, v co-occur. To approximate the overall traversability of an edge, we can calculate an empirical estimate for the conditional probabil-

ity $P(e | (u, v)) = \frac{P(e) \cap P(u, v)}{P(u, v)}$ that an edge is traversed, given that the two incident nodes are activated. This estimate can use the same beta-bernoulli distribution from Equation 2.2.

Still, how do we ensure our estimate is approximating traversability, so that the probability that an edge probability converges toward 1 as long as it *has to be possible* for e to be traversed? Recall from the introduction that, from a measurement perspective, the underlying networks rarely “exist” in the sense that we never truly find the underlying network, but only samples from (or caused by) it. Imagine that the “real” network is a set of paved sidewalks: our procedure is similar to watching people walk along paths between locations, and wanting to estimate which of the paths would be tread “enough” to be paved. This is where we build on an intuition based on the popular idea of *desire paths* which is a colloquial name for paths that form when individuals walk over grass or dirt enough to carve a trail. In network analysis and recovery, we usually are only allowed to see the desire paths that might have formed from our data, never the actual “pavement”.

If presented with two equivalent paths, an individual is likely to choose the one that has been tread more often before i.e. the “more beaten” path. So, we don’t want a probability that an edge has been traversed, but a probability over fractions of the time we expect an edge to have been traversed *more often than not*: how likely it is to “be beaten”. This is accomplished by forcing $\alpha, \beta < 1$. For the case where $\alpha = \beta = 0.5$, we call this special case an *arcsine* distribution.

In a coin tossing game where each “heads” gives a point to player A and “tails” to player B, then the point differential is modeled as a random walk. The arcsine

distribution $\text{Beta}(\frac{1}{2}, \frac{1}{2})$ is exactly the probability distribution for the fraction of time we expect one player to be “ahead” of the other. [27]

“Contrary to popular opinion, it is quite likely that in a long coin-tossing game one of the players remains practically the whole time on the winning side, the other on the losing side.”

William Feller[90, Chapter III]

This is desirable behavior for a distribution over edge traversability! We expect the vast majority of edges to have a 0 or 1 as the most likely values, with the expected fraction of observations that an edge being traversed was “ahead” of it being *not* traversed matching our empirical count. We generalize this with $\alpha = 1 - \beta$, with $\alpha + \beta = 1$, such that the new beta posterior from Equation 2.2 with s successes over k trials is:

$$\begin{aligned}\pi &\sim \text{Beta}(\alpha + c, 1 - \alpha - c) \\ c &= \frac{s - ak}{k + 1}\end{aligned}\tag{5.4}$$

Important to note is that k is measured over the *co-occurrences* (u, v) , and successes are the traversals e derived from our samples in R . This lets us formulate a likelihood model for each edge’s traversability in the global network G (i.e. whether $B(e, v) > 0$ for any v), which is i.i.d. Bernoulli.

$$\begin{aligned}\pi_e &\sim \text{Beta}(\alpha, 1 - \alpha), e \in E \\ E &\sim \text{Bernoulli}(\pi_e), e \in E\end{aligned}\tag{5.5}$$

This does not yet specify a likelihood for \mathcal{C}_i , because we have not included a mechanism for the down-selection to each \mathbf{x}_i . This will be addressed more completely for the special case of $\mathcal{C} = \mathcal{F}$, the set of spanning forests on a graph, in Section 7.2. In general, however, failing to specify the prior distribution on \mathcal{C}_i does not necessarily make Equation 5.3 unusable, but necessitates an “empirical bayes” approach. With the marginals and co-occurrence counts for nonzero values in X , we can make a point estimate for each edge *given* a node subset, without needing to consider a prior distribution for each \mathbf{x}_i .

The nonparametric approach, in cases that merit the use of spanning trees, will be central to accurate, scalable estimation of B through our proposed method, Forest Pursuit.

Chapter 6: Approximate Recovery in Near-linear Time by *Forest Pursuit*

In this chapter, we build on the notion of “Desire Path” estimation of a dependency network from node activations — sampling from a class of subgraphs constrained to active nodes, then merging them. We present *Forest Pursuit* (FP), a method that is scalable, trivially parallelizes, and offline-capable, while also outperforming commonly used algorithms across a battery of standardized tests and metrics. The key application for using FP is in domains where node activation can be reasonably modeled as arising due to *random walks*—or similar spreading process—on an underlying dependency graph.

Because we wish to model our node activations as being *caused* by other nodes (that they depend on), we draw a connection to a class of models for *spreading*, or, *diffusive processes*. We outline how random-walks are related to these diffusive models of graph traversal, enabled by an investigation of the graph’s “regularized laplacian” from Avrachenkov et al. [32].

First, we build an intuition for the use of trees as an unbiased estimator for desire path estimation when spreading processes are at work on the latent network.

Then, the groundwork for FP is laid by combining sparse approximation through *matching pursuit* with a loss function modeled after the Chow Liu representation for joint probability distributions. The approximate complexity for FP is linear in the spreading rate of the modeled random walks, and linear in dataset size, while running in $O(1)$ time with respect to the network size. This departs dramatically from other methods in the space, all of which assume to scale in the number of nodes. We then test FP against an array of alternative methods (including GLASSO) with MENDR, a newly-developed standard reference dataset and testbench for network recovery. FP outperforms other tested algorithms in nearly every case, and empirically confirms its complexity scaling for sub-thousand network sizes.

6.1 Random Walks as Spanning Forests

The class of diffusive processes we focus on “spread” from one node to another. If a node is activated, it is able to activate other nodes it is connected to, directly encoding our need for the graph edges to represent that nodes “depend” on others to be activated. In this case, a node activates when another node it depends on spreads their state to it. These single-cause activations are often modeled as a random-walk on the dependency graph: visiting a node leads to its activation.

6.1.1 Random Walk Activations

Random walks are regularly employed to model spreading and diffusive processes on networks. If a network consists of locations, states, agents, etc. as “nodes”, and

relationships between nodes as “edges”, then random walks consist of a stochastic process that “visits” nodes by randomly “walking” between them along connecting edges. Epidemiological models, cognitive search in semantic networks, stock price influences, website traffic routing, social and information cascades, and many other domains also involving complex systems, have used the statistical framework of random walks to describe, alter, and predict their behaviors. [CITE...lots?]

When network structure is known, the dynamics of random-walks are used to capture the network structure via sampling [LITTLEBALLOFFUR, etc], estimate node importance’s[PAGERANK], or predict phase-changes in node states (e.g. infected vs. uninfected)[SIR I think]. In our case, Since we have been encoding the activations as binary activation vectors, the “jump” information is lost—activations are “emitted” for observation only upon the random walker’s initial visit.[41] Further, the ordering of emissions has been removed in our binary vector representation, leaving only co-occurrence groups in each \mathbf{x}_i .¹ In many cases, however, the existence of relationships is not known already, and analysts might *assume* their data was generated by random-walk-like processes, and want to use that knowledge to estimate the underlying structure of the relationships between nodes.

6.1.2 Activations in a Forest

As a general setting, the number of node activations (e.g. for datasets like co-authorship) is much smaller than the set of nodes ($\|\mathbf{x}_i \in \mathbb{S}^n\|_0 \ll n$)²

¹For a brief treatment of the case that INVITE emission order is preserved, see Chapter 9

² $\|\cdot\|_0$ is the ℓ_0 “pseudonorm”, counting non-zero elements (the support) of its argument.

Avrachenkov et al. [32] go to some length describing discrete- and continuous-time random walk models that can give rise to binary activation vectors like our $X(i, j) : I \times J \rightarrow \mathbb{B}$. The *regularized laplacian* (or *forest*) kernel of a graph[22] plays a central role in their analysis, as it will in our discussion going forward.

$$Q_\beta = (I + \beta L)^{-1} \quad (6.1)$$

In that work, it is discussed as the optimal solution to the semi-supervised “node labeling” problem, having a regularization parameter β , though its uses go far beyond this.[35, 47, 65] Q generalizes the so-called “heat kernel” $\exp(-t\tilde{L})$, in the sense that it solves a lagrangian relaxation of a loss function based on the heat equation. This can be related to the PageRank ($\exp(-tL^{\text{rw}})$) kernel as well, which is explicitly based on random walk transition probabilities.

In fact, Q can be viewed as a transition matrix for a random walk having a geometrically distributed number of steps, giving us a small expected support for x_i , as needed.³ However we interpret Q , a remarkable fact emerges due to a theorem by Chebotarev: each entry $q = Q(u, v)$ is equal to the probability⁴ that nodes u, v are connected in a randomly sampled *spanning rooted forest*

In other words, co-occurring node activations due to a random walk or heat kernel are deeply tied to the chance that those nodes find themselves *on the same tree in a forest*.

³ Q can also be interpreted as a continuous-time random walk location probability, after exponentially distributed time, if spending exponentially-distributed time in each node.

⁴edge weights scaled by \square

6.1.3 Spreading Dependencies as Trees

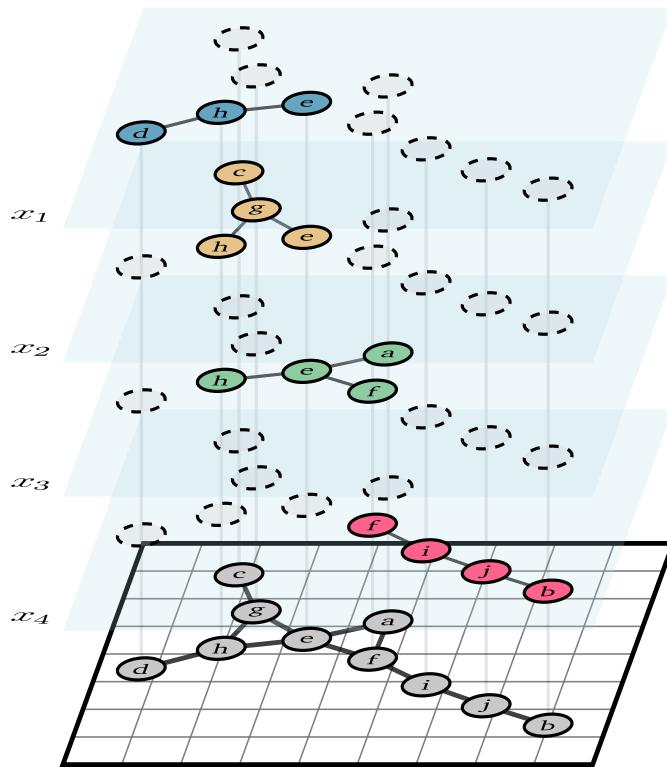


Figure 6.1: Edge Measurements with true (tree) dependencies known

With the overt link from spreading processes to counts of trees made, there's room for a more intuitive bridge.

For single-cause, single source spreading process activations—on a graph—the activation dependency graph for each observation/spread/random walk *must be a tree*. With a single cause (the “root”), which is the starting position of a random walker, a node can only be reached (activated) by another currently activated node. If the random walk jumps from one visited node to another, previously visited one, that transition did not result in an activation , so the *dependency* count for that edge should not increase. This description of a random walk, where subsequent visits do not “store” the incoming transition, is roughly equivalent to one more commonly

described as a *Loop-Erased* random walk. It is precisely used to uniformly sample the space of spanning trees on a graph.[\[80\]](#)

Much like a reluctant co-author “worn down” by multiple requests, we can even include random walks that “receive” activation potential from more than one source. Say a node is activated when some fraction of its neighbors have all originated a random walk transition to it, or a node activates on its second visit, or similar. We simply count (as dependency evidence) the ultimate transition that precipitated activation. This could be justified from an empirical perspective as well: say we observe an author turn down requests for one paper from two individuals, but accept a third. We could actually infer a *lowered* dependency on the first two, *despite* the eventual coauthorship. Only the interaction that was observed as successful necessarily counts toward success-dependency, barring any contradicting information.

It’s important to add here that *mutual convincing* by multiple collaborators simultaneously (or over time) is expressly left out. In other words, only pairwise interactions are permitted. This is not an additional assumption, but a key limitation of our use of graphs in the first place! As Torres et al. go to great lengths elaborating in [\[14\]](#), it is critical to correctly model dependencies when selecting a structural representation of our problem to avoid data loss. The possibility for multi-way interactions would necessitate the use of either a simplicial complex or a hypergraph as the carrier structure, *not a graph*.

Figure [6.1](#) demonstrates the use of trees as the distribution for subgraphs, instead of outer-products/cliques.

6.2 Sparse Approximation

As indicated previously, we desire a representation of each observation that takes the “node space” vectors (\mathbf{x}_i) to “edge space” ones (\mathbf{r}_i). We have separated each observation with the intention of finding a point-estimate for the “best” edges, such that the edge vector induces a subgraph belonging to a desired class. If we assume that each edge vector is in \mathbb{B}^ω , so that the interactions are unweighted, undirected, simple graphs, then for any family of subgraphs we will be selecting from at most $\omega \leq \binom{n}{2}$ edges.

Representing a vector as sparse combination of a known set of vectors (also known as “atoms”) is called *sparse approximation*.

6.2.1 Problem Specification

Sparse approximation of a vector \mathbf{x} as a representation \mathbf{r} using a dictionary of atoms (columns of D) is specified more concretely as [61]:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{\operatorname{argmin}} \|\mathbf{x} - D\mathbf{r}\|_2^2 \quad \text{s.t. } \|\mathbf{r}\|_0 \leq N \quad (6.2)$$

where N serves as a sparsity constraint (at most N non-zero entries). This is known to be NP-hard, though a number of efficient methods to approximate a solution are well-studies and widely used. Solving the lagrangian form of Equation 6.2, with an ℓ_1 -norm in place of ℓ_0 , is known as Basis Pursuit[81], while greedily solving for the non-zeros of \mathbf{r} one-at-a-time is called *matching pursuit*[83]. In that work, each

iteration selects the atom with the largest inner product $\langle \mathbf{d}_{i!}, \mathbf{x} \rangle$.

We take an approach similar to this, but with the insight that the inner product will not result in desired sparsity (namely, a tree). Our dictionary in this case will be the set of edges given by B (see Section 5.2.1), while our sparsity is given by the relationship of the numbers of nodes and edges in a tree:

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{\operatorname{argmin}} \|\mathbf{x} - B^T \mathbf{r}\|_2^2 \quad \text{s.t. } \|\mathbf{r}\|_0 \leq \|\mathbf{x}\|_0 - 1 \quad (6.3)$$

There are some oddities to take into account here. As a linear operator (see Section 2.2), B^T takes a vector of edges to node-space, counting the number of edges each node was incident to. This means that, even with a ground-truth set of interactions, B^T would take them to a new matrix $X_{\deg}(i,j) : I \times J \rightarrow \mathbb{N}$, which has entries of the number of interactions each individual in observation i was involved in. While very useful for downstream analysis (see Section 9.2.4), the MSE loss in Equation 6.4 will never be zero, since X_{\deg} entries are not boolean. Large-degree “hub” nodes in the true graph would give a large residual, and the adjoint would subsequently fail to remove the effect of B^T on the edge vectors.

It might be possible to utilize a specific semiring, such as $(\min, +)$, to enforce inner products (see Section 2.4) that take us back to a binary vector.⁵ Instead, we will take an empirical bayes approach to the estimation of sparse vectors.[63] As a probabilistic

⁵This is more than a simple hack, and belies a great depth of possible connection to the problem at hand. It is known that “lines” (arising from equations of the inner product) in tropical projective space *are trees*.[70] In addition, the tropical equivalent to Kirchoff’s polynomial (which counts over all possible spanning trees), is the direct computation of the minimum spanning tree.[10] For treatment of sparse approximation using tropical matrix factorization, see Omanović et al. [12]

graphical model, we assume each observation is emitted from a (tree-structured) MRF [explained?] on the activated nodes. This is underdetermined (any spanning tree could equally emit the observed activations), so we use an empirical prior as a form of shrinkage: the co-occurrences of nodes across all observed activation patterns. This lets us optimize a likelihood from Equation 5.3, for the distribution of spanning trees on the subgraph of G^* induced by \mathbf{x} .

$$\hat{\mathbf{r}} = \underset{\mathbf{r}}{\operatorname{argmax}} \mathcal{L}(\mathbf{r} | \mathbf{x}) \quad \text{s.t.} \quad \mathbf{r} \sim \mathcal{T}(G^*[\mathbf{x}]) \quad (6.4)$$

6.2.2 Max. Spanning (Steiner) Trees

The point estimate $\hat{\mathbf{r}}$ is therefore the mode of a distribution over trees, which is precisely the maximum spanning tree.[15] If we allow the use of all observations X to find an empirical prior for \mathbf{r} , then we can calculate a value for the mutual information for the activated nodes, and use this to directly calculate the Chow-Liu estimate. One algorithm for finding a maximum spanning tree is Prim's[CITE?], which effectively performs the matching pursuit technique of greedily adding an edge (i.e. non-zero entry in our vector) one-by-one. In this way, we effectively do perform matching pursuit, but minimizing the KL-divergence between observed node activations and a tree-structured MRF limited to those nodes, alone (rather than the mean-square-error).

However, the mode of the tree distribution is not strictly the one that uses mutual information as edge weights. There is reason to believe that edge weights based on

pairwise joint probabilities might also be appropriate. Namely, the Hunter-Worsley bound for unions of (dependent) variables says that the sum of marginal probabilities over-counts the true union of activations (including by dependence relations). This alone would be known as Boole’s inequality, but the amount it overcounts is *at most* the weight of the maximum spanning tree over pairwise joint probabilities.[87] Adding the tree of joint co-occurrence probabilities is the most conservative way to arrive at the observed marginals from the probability of at least one node occurring (which could then be the “root”).

Finally, we realize that the problem statement (“find the maximum weight tree on the subgraph”) is not the same as an MST, per-se, but rather the so-called “Steiner Tree” problem. In other words, we would like our tree of interactions to be of minimum weight on a node-induced subgraph of *the true graph*. The distribution of trees that our interactions are sampled from should be over the available edges in the recovered graph, *which we do not yet have*. Thankfully, a well-known algorithm for approximating the (graph) Steiner tree problem instead finds the minimum spanning tree over the *metric closure* of the graph.[85]

This metric closure is a complete graph having weights given by the shortest-path distances between nodes. While we don’t know those exact values either, we do have the fact that the distance metric implied by the forest kernel (in Equation 6.1) is something of a relaxation of shortest paths. In the limit $\beta \rightarrow 0$, Q is proportional to shortest path distances, while $\beta \rightarrow \infty$ instead gives commute/resistance distances.[32] And that kernel is counting the probability of co-occurrence on trees in any random spanning forest!

All this is to say that node co-occurrence measures are more similar to node-node distances in the underlying graph, *not estimators of edge existence*. But we can use this as an empirical prior to approximate Steiner trees that *are on the true graph*.

6.3 Forest Pursuit

Instantiating the above, we propose *Forest Pursuit*, an relatively simple algorithm for correction of clique-bias under a spreading process assumption

6.3.1 Algorithm Summary

Once again, we assume m observations of activations over n nodes, represented as the design matrix $X : I \times J \rightarrow \mathbb{B}$. Like GLASSO, we assume that a Gram matrix (or re-scaling of it) is precomputed, for the non-streaming case.

Based on the discussion in Note 1 we will use the cosine similarity as a degree-corrected co-occurrence measure, with node-node distances estimated as $d_K = -\log \text{Ochiai}(j, j')$.⁶

For each observation, the provided distances serve to approximate the metric closure of the underlying subgraph induced by \mathbf{x} . This is passed to an algorithm for finding the minimum spanning tree. Given a metric closure, the MST in turn would be an approximation of the desired Steiner tree within $2 - \frac{2}{\|\mathbf{x}\|_0}$ of optimal.[85] For $\|\mathbf{x}\|_0 \ll n$, this error bound will often be close to 1, and approaching 2 as the expected number of activations-per-observation grows.

⁶Note that any kernel could be used, given other justification, though anecdotal evidence has the negative-log-Ochiai distance performing marginally better than MI distance or Yule's Q .

After the point estimates for \mathbf{r} have been calculated as trees, we can use the desire-path beta-binomial model (Equation 5.4) to calculate the overall empirical Bayes estimate for \hat{G} . As a prior for α , instead of a Jeffrey's or Laplace prior, we bias the network toward maximal sparsity, while still retaining connectivity. In other words, we assume that n nodes only need about $n - 1$ edges to be fully connected, which implies a prior expected sparsity of

$$\alpha^* = \frac{n-1}{\frac{1}{2}n(n-1)} = \frac{2}{n} \quad (6.5)$$

which we can use as a sparsity-promoting initial value for $\text{Beta}(\alpha^*, 1 - \alpha^*)$.

Algorithm 6.1 outlines the algorithm in pseudocode for reproducibility.

6.3.2 Approximate Complexity

The Forest Pursuit calculation presented in Algorithm 6.1 assumes an initial value for the distance matrix, which is similar to the covariance estimate that is pre-computed (as an initial guess) for GLASSO. Therefore we do not include the matrix multiplication for the gram matrix in our analysis, at least in the non-streaming case. Because every observation is dealt with completely independently, the FP estimation of R is linear in observation count. It is also trivially parallelizable⁷ and the bayesian update for $\hat{\alpha}_m$ can be performed in a streaming manner, as well.

Each observation requires a call to “PursueTree”, which involves an MST call for

⁷Although, no common python implementation of MST algorithms are as-yet vectorized or parallelized for simultaneous application over many observations. We see development of non-blocking MSTs in common analysis frameworks as important future work.

Algorithm 6.1 Forest Pursuit

Require: $X \in \mathbb{B}^{m \times n}, d_K \in \mathbb{R}_{\geq 0}^{n \times n}, 0 < \alpha < 1$

Ensure: $R \in \mathbb{B}^{m \times \binom{n}{2}}$

```

1: procedure FORESTPURSUITEDGEPROB( $X, d_K, \alpha$ )
2:    $R \leftarrow$  FORESTPURSUIT( $X, d_K$ )
3:    $\hat{\alpha}_m \leftarrow$  DESIREPATHBETA( $X, R, \alpha$ )
4:   return  $\hat{\alpha}_m$ 
5: end procedure
6: procedure FORESTPURSUIT( $X, d_K$ )
7:    $R(\cdot, \cdot) \leftarrow 0$                                       $\triangleright$  each observation
8:   for  $i \leftarrow 1, m$  do
9:      $\mathbf{x}_i \leftarrow X(i, \cdot)$ 
10:     $R(i, \cdot) \leftarrow$  PURSUETREE( $\mathbf{x}_i, d_K$ )
11:   end for
12:   return  $R$ 
13: end procedure
14: procedure PURSUETREE( $\mathbf{x}, d$ )                                 $\triangleright$  Approximate Steiner Tree
15:    $V \leftarrow \{v \in \mathcal{V} | \mathbf{x}(v) = 1\}$                        $\triangleright$  activated nodes
16:    $T \leftarrow$  MST( $d[V, V]$ )                                          $\triangleright$  e.g. Prim's Algorithm
17:    $\mathbf{u}, \mathbf{v} \leftarrow \{(j, j') \in J \times J | T(j, j') \neq 0\}$ 
18:    $\mathbf{r} \leftarrow e_n(\mathbf{u}, \mathbf{v})$                                           $\triangleright$  unroll tree adjacency
19:   return  $\mathbf{r}$ 
20: end procedure
21: procedure DESIREPATHBETA( $X, R, \alpha$ )
22:    $s \leftarrow \sum_{i=1}^m R(i, e)$ 
23:    $\sigma \leftarrow \sum_{i=1}^m X(i, j)$ 
24:    $\mathbf{k} \leftarrow e_n(\sigma \sigma^T)$                                           $\triangleright$  co-occurrence counts
25:    $\hat{\alpha}_m \leftarrow \alpha + \frac{s - \alpha k}{k+1}$ 
26:   return  $\hat{\alpha}_m$ 
27: end procedure

```

the pre-computed subset of distances on nodes activated for that observation. Note the use of MST algorithm could apply Prim's or Kruskal's algorithm, or similar. It is recommended to utilize Prims in this case, however, since Prims on a sufficiently dense graph can be made to run in $O(n)$ time for n activated nodes by using a d -tree for its heap queue.[CITE] Since we are always using the metric closure, Prim's will always run on a complete graph.

Importantly, this means that FP does not scale with the size of the network, but only the worst-case activation count of a given observation, $O(s_{\max})$, where $s_{\max} = \max_i (\|X(i, \cdot)\|_0)$. We say this is *approximately* constant in node size:

- the total number of nodes is typically a *given* for a single problem setting
- in many domains, the basic *spreading* rate of diffusion model (e.g. R_0 , or heat conductivity), does not scale with the total size of an observation

That last point means that constant scaling with network size is generally down to the domain in question. For instance, a heat equation simulated over a small area, having a given conductivity, will not have a different conductivity over a larger area; conductivity is a material property. Similarly, a virus might have a particular basic reproduction rate, or a set of authors might have a static distribution over how many collaborators they wish to work with. The former is down to viral load generation, and the latter a sociological limit: more a bigger department usually doesn't imply more co-authors.

Similar to Equation 6.5, we might reasonably assume that the expected degree of nodes is roughly constant with network size i.e. a property of domain. So, the density of activation vectors (as a fraction of all possible edges) is going to scale with the inverse of n . This makes a process linear in activation count to be constant in network size. Then, if \bar{s} is the expected non-zero count of each row of X , the final approximate complexity of FP is $O(m\bar{s})$.⁸

⁸In our reference implementation, which uses Kruskal's algorithm, the theoretical complexity is likewise $O(m\bar{s}^2 \log \bar{s})$, though in our experience the values of \bar{s} are small enough to not impact the runtime significantly.

6.4 Simulation Study

To test the performance of FP against other backboning and recovery methods, we have developed a public repository [affinis](#) containing reference implementations for FP, along with many co-occurrence and backboning techniques. The library contains source code and examples for many of the presented methods, and more. [UPDATE w/ DOI?]

In addition, to support the community and provide for a standard set of benchmarks for network recovery from activations, the [MENDR](#) reference dataset and test-bench was developed. To make reproducible comparison of recovery algorithms easier, MENDR includes hundreds of randomly generated networks in several classes, along with random walks sampled *on those networks*. It can also be extended through community contribution, using data versioning to allow consistent comparison between different reports and publications over time.

6.4.1 Experimental Method

For each algorithm shown in Table 6.2, every combination of the parameters in Table 6.1 was tested. 30 random graphs for each of nodes were tested, which was repeated again for each of Tree, Block, and scale-free types.

Table 6.1: Experiment Settings (MENDR Dataset)

parameters	values
random graph kind	Tree, Block, BA($m \in \{1, 2\}$)

parameters	values
network n-nodes	10,30,100,300
random walks	1 sample $m \sim \text{NegBinomial}(2, \frac{1}{n}) + 10$
random walk jumps	1 sample $j \sim \text{Geometric}(\frac{1}{n}) + 5$
random walk root	1 sample $n_0 \sim \text{Multinomial}(\mathbf{n}, 1)$
random seed	1, 2, ..., 30

Table 6.2: Summary of algorithms compared

Algorithm	abbrev.	α ?	class	source
Forest Pursuit	FP	Yes	hybrid	-
GLASSO	GL	No	MRF	[53, 60]
Ochiai Coef.	CS	Yes	Counts	[84]
Hyperbolic Projection	HYP	No	Counts	[75]
Doubly-Stochastic	eOT	Yes	Transport	[45, 59]
High-Salience Skeleton	HSS	Yes	Transport	[51]
Resource Allocation	RP	Yes	Transport	[64]

For each algorithm that could be supplied a prior via additive smoothing, seen in Table 6.2 as “ α ? Yes”, a minimum connected (tree) level of sparsity was supplied $\alpha = \frac{2}{n}$. The others, esp. GLASSO, do not have a $\frac{\text{count}}{\text{exposure}}$ form, and could not be easily interpreted in a way that allowed for additive smoothing. However, since the regularization parameter for GLASSO is often critical for finding good solutions, a

5-fold cross validation was performed for each experiment to select a “best” value, with the final result run using that value. While this does have a constant-time penalty for each experiment, the reconstruction accuracy is significantly improved with this technique, and would reflect common practice in using GLASSO for this reason.

The three classes of random graphs represent common use cases in sparse graph recovery. In addition, the block and tree graphs are types we expect GLASSO to correctly recover in this binary setting.[\[53\]](#) The block graphs of size n were formed by taking the line-graph of randomly generated trees of size $n + 1$.

Trees were randomly generated using Prüfer sequences as implemented in NetworkX [CITE]. To simulate possible social networks and other complex systems that show evidence of preferential attachment, scale-free graphs were sampled through the Barabási–Albert (BA) model, which was randomly seeded with a re-attachment parameter m of either 1 or 2.

Every graph has a static ID provided by MENDR, along with generation and retrieval code for public review. New graphs kinds and sizes are simple to add for future benchmarking capability.

6.4.2 Metrics

To compare each algorithm consistently, several performance measures have been included in the MENDR testbench. They are all functions of the True Positive/Negative (TP/TN) and False Positive/Negative (FP/FN) values.

Precision (P) Fraction of positive predictions that are true, also called “positive predictive value” (PPV)

$$P = \frac{TP}{TP + FP}$$

Recall (R) Fraction of true values that were returned as positive. Also called the TP-rate (TPR), and has an inherent trade-off with precision.

$$R = \frac{TP}{TP + FN}$$

Matthews Correlation Coefficient (MCC) Balances all of TP,TN,FP,FN. Preferred for class-imbalanced problems (like sparse recovery) [2]

$$\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Fowlkes-Mallows (F-M) Geometric mean of Precision and Recall, as opposed to the F-Measure that returns the harmonic mean. Also known to be the limit of the MCC as TN approaches infinity[3], which is useful as TN grows with n^2 but TP only with n .

$$\sqrt{P \cdot R}$$

Because this work is focused on unsupervised performance, specifically for the use of these algorithms by analysts investigating dependencies, we opt to calculate TP,TN,FP,FN at every unique edge probability/strength value returned by each algorithm. Then, because we do not know a priori which threshold level will be selected

by an analyst in the unsupervised setting, we take a conservative approach and report the expected values $E[MCC]$ and $E[F-M]$ over all unique threshold values. To consistently compare the expected values, we transform the thresholds for every experiment to the range $[\epsilon, 1 - \epsilon]$, to avoid division-by-0 at the extremes.

Another common approach is to report the Average Precision Score (APS). This is not the average precision over the thresholds however, but instead the expected precision over the possible recall values achievable by the algorithm. It is approximating the integral under the parametric P-R curve, instead of the thresholds themselves.

$$APS = \sum_{e=1}^{\omega} P(e)(R(e) - R(e-1))$$

where $P(e)$ and $R(e)$ are the precision and recall at the threshold set by the edge e , in rank-order. This is more commonly done for supervised settings, however, and will report a high value as long as *any* threshold is able to return both a high precision and a high recall, simultaneously.

6.4.3 Results - Scoring

The results over every experiment are shown in Figure 6.2. Only FP is able to report MCC and F-M values with medians over about 0.5, regularly reaching over 0.8. GLASSO is clearly the second-best at recovery in these experiments, though for scale-free networks the improvement over simply thresholding the Ochiai coefficient is negligible. For APS, both GLASSO and Ochiai are equally able to return high scores, indicating at least one threshold for each that performed well. A simple mechanism

for FP to perform equally well at APS is discussed in Section 7.1.

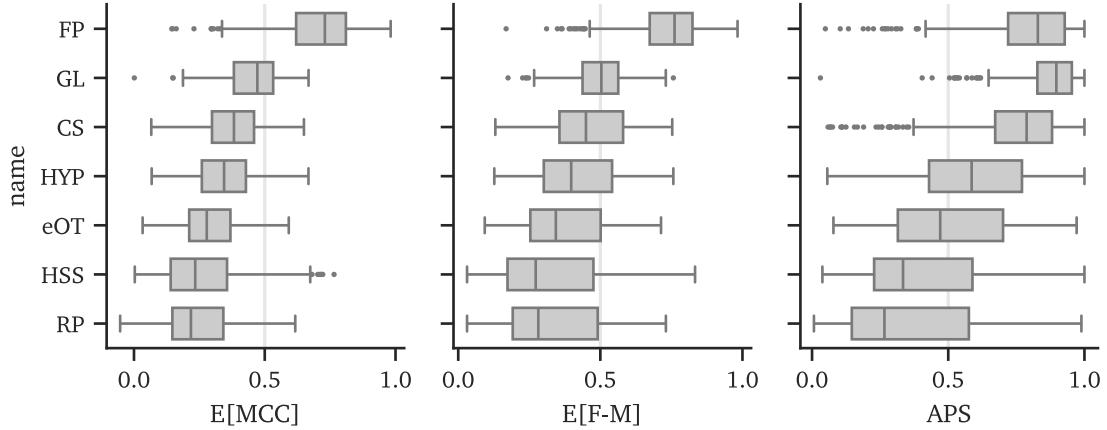


Figure 6.2: Comparison of MENDR recovery scores

Breaking down the results by graph kind in Figure 6.3, we see the remarkable ability of FP to dramatically outperform every other algorithm in MCC and F-M, showing remarkable accuracy *together with stability* over the set of threshold values. This is indicative of FP’s ability to more directly estimate the support of each edge, with lower values occurring only when co-occurrences aren’t being consistently explained with the same set of incidences.

Another important capability of any recovery algorithm is to improve its estimate when provided with more data. Of course, this also will depend on other factors, such as the dimensionality of the problem (network size), and specifically for us, whether *longer* random walks makes network inference better or worse.

As Figure 6.4 shows, FP is positively correlated with *all three*. Most importantly, the trend for FP is strongest as the number of observations increases, which is not a phenomenon seen in the other methods. In fact, it appears that count-based methods’ scores are negatively correlated with added random walk length and added

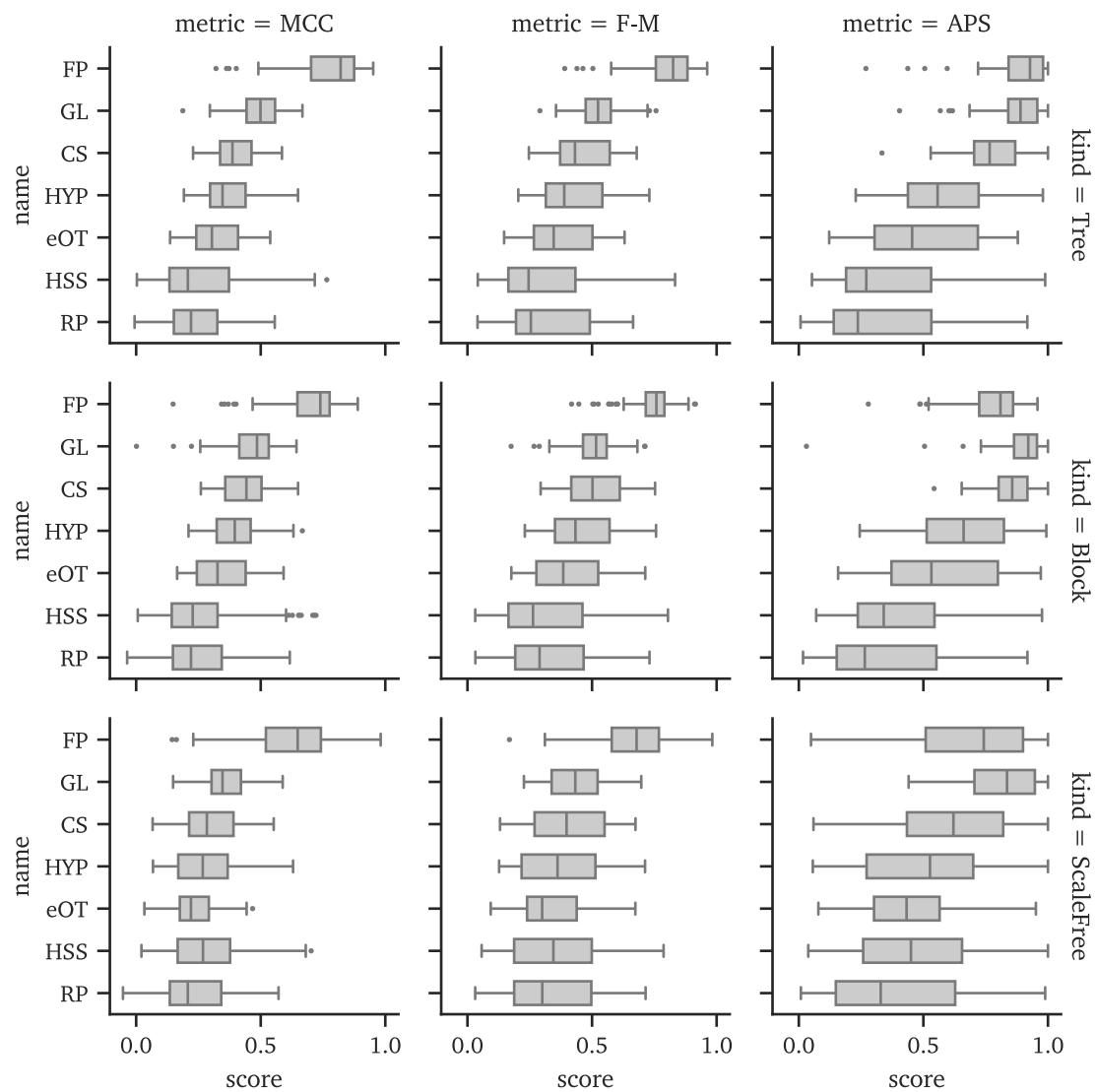
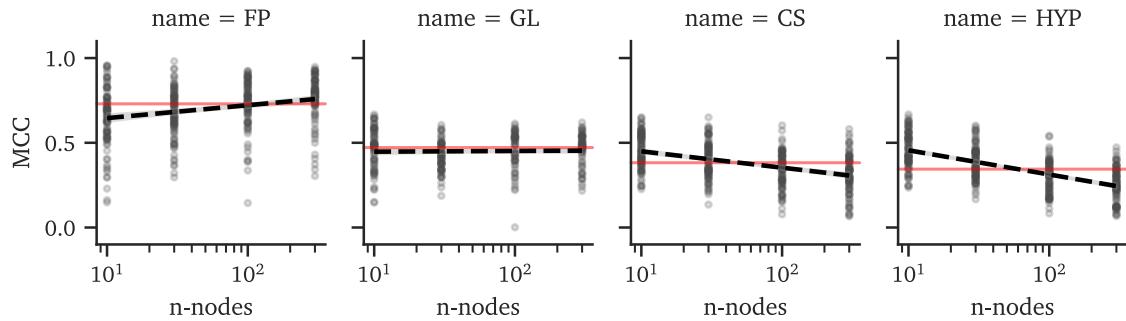
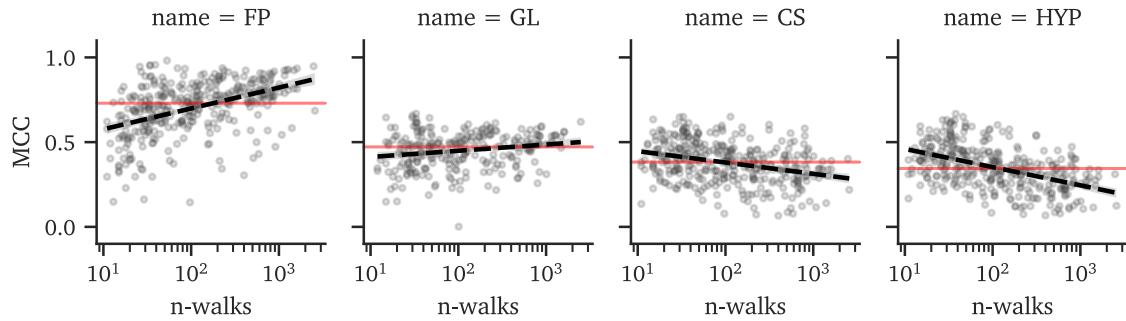


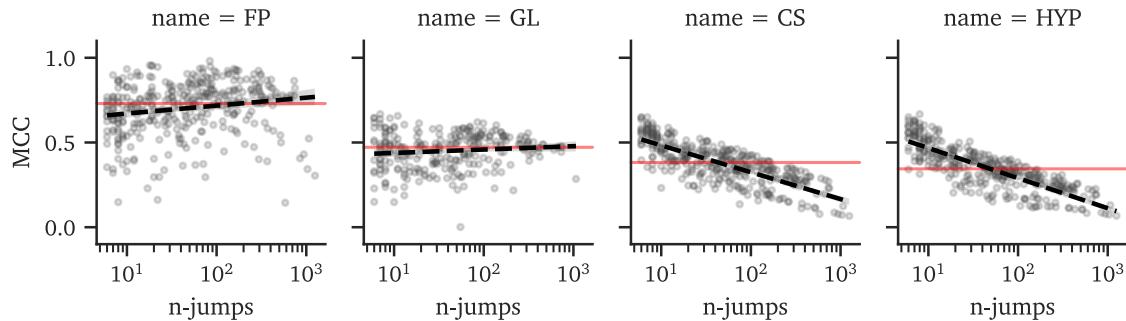
Figure 6.3: Comparison of MENDR Recovery Scores by Graph Type



(a) Trend: MCC vs network size



(b) Trend: MCC vs observation count



(c) Trend: MCC vs random-walk length

Figure 6.4: Score trends vs problem scaling

observations. Only HYP and CS scores are shown in Figure 6.4, but all other tested methods (other than FP and GLASSO) show the same trend.

However, because the graph sampling protocol includes n in the distributions for the observation count and random-walk length, we additionally performed a linear regression on the (log) parameters. The *partial residual* plots are shown in Figure 6.5, which shows the trends of each variable after controlling for the others.

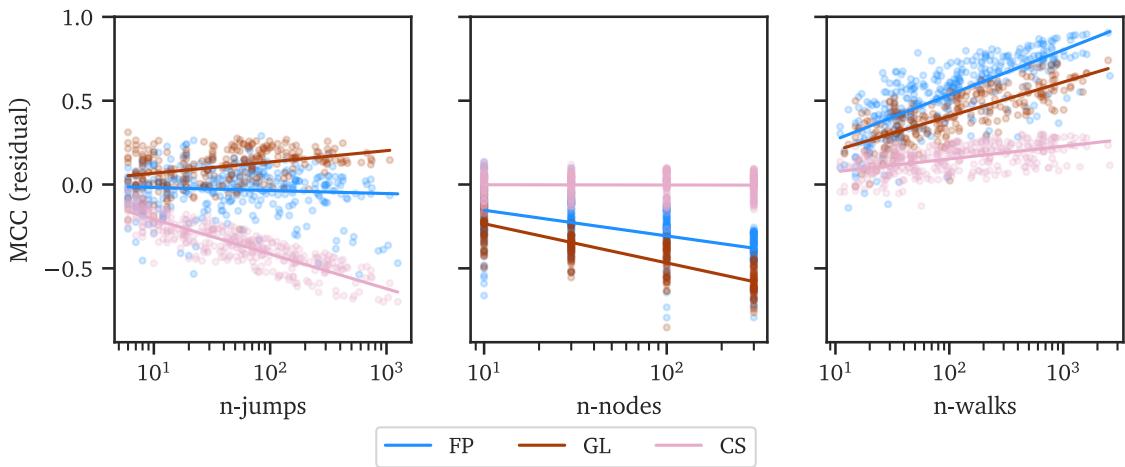


Figure 6.5: Partial Residuals (regression on $E[\text{MCC}]$)

This analysis indicates that *all* methods should likely increase in their performance when extra observations are added, though FP does this more efficiently than either CS or GLASSO. Interestingly, CS is largely unaffected by network size, compared to FP and GL, though GL performs the worst in this regard. However, it is in the random-walk length that we see the benefit of dependency-based algorithms. The Ochiai coefficient suffers dramatically as more nodes are activated by the spreading process, since this means the implied clique size grows by the square of the number of activations. FP remains unaffected by walk-length, while (impressively) GLASSO appears to have a marginal boost in performance when walk lengths are high.

6.4.4 Results - Runtime Performance

For both Forest Pursuit and GLASSO, runtime efficiency is critical if these algorithms are going to be adopted by analysts for backboning and recovery. Figure 6.6 shows the (log-)seconds against the same parameters from before. For similar sized networks, FP is consistently taking 10-100x less time to reach a result than GLASSO does. Additionally, many of the experiments led to ill-conditioned matrices that failed to converge for GLASSO under any of the regularization parameters tests (the “x” markers in Figure 6.6). As expected, the number of observations plot shows a clear limit in terms of controlling the lower-bound of FPs runtime, since in this serial version every observation runs one more call to MST. On the other hand, GLASSO appears to have significant banding for walk length and observation counts, likely indicating dominance of network size for its runtime.

To control for each of the variables, and to empirically validate the theoretical analysis in Section 6.3.2}, a regression of the same three (log-)parameters was performed against (log-)seconds. The slopes in Figure 6.7, which are plotted on a log-log scale, correspond roughly to polynomial powers in linear scale. In regression terms, we are fitting the log of

$$y_{\text{sec}} = ax_{\text{param}}^{\gamma}$$

so that the slope in a log-log plot is γ .

In a very close match to our previous analysis, the scaling of FP is almost entirely

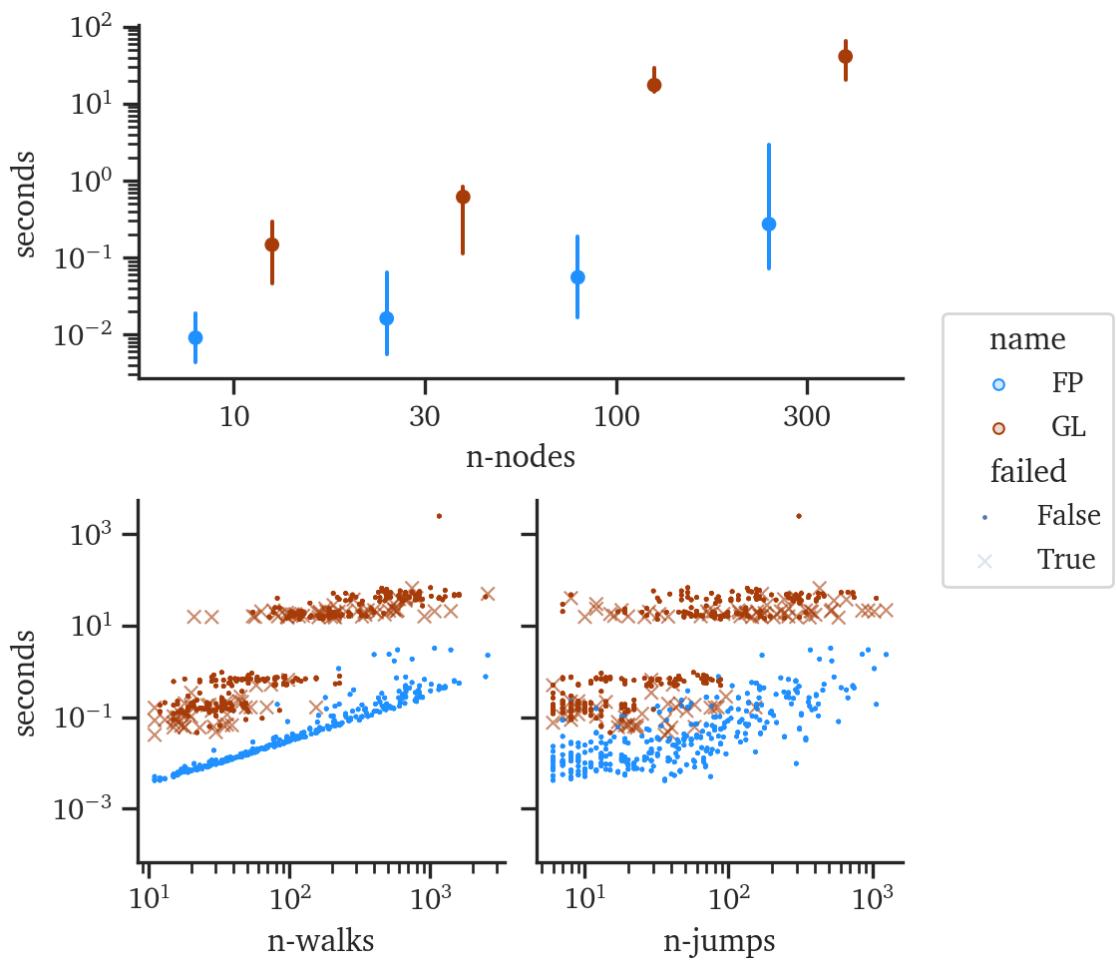


Figure 6.6: Runtime Scaling (Forest-Pursuit vs GLASSO)

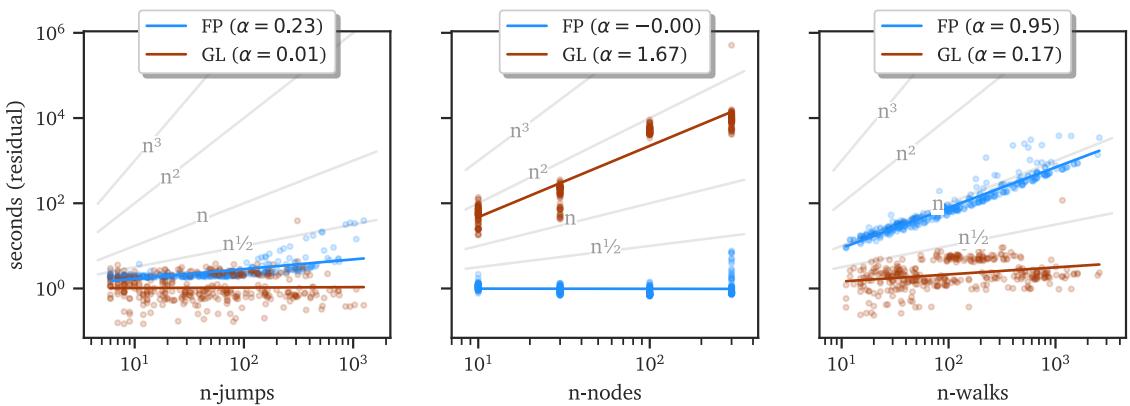


Figure 6.7: Partial Residuals (regression on computation time)

explained by the observation count and random-walk length, alone: the coefficient on network size shows constant-time scaling. Similarly, the scaling with observation count is very nearly linear time, as predicted. The residuals show non-linear behavior for the random-walk length parameter, which would make sense, due to the theoretical $\|E\| \log \|V\|$ scaling of kruskal's algorithm. At this scale, $n \log n$ and $n^2 \log n$ complexity might appear smaller than linear time, due to the log factor. GLASSO hardly scales with random walk length, and only marginally with observation count. In typical GLASSO, the observation count has already been collapsed to calculate the empirical covariance matrix, so its effects here might be due instead to the cross-validation and the need to calculate empirical covariance for observation subsets. The big difference, however, is GLASSO scaling in significantly superlinear time—almost $O(n^2)$. This is usually the limiting factor for analyst use of such an algorithm in network analysis more generally.

Chapter 7: Modifications & Extensions

Forest Pursuit is a flexible base for adding or modifying behavior as needed. In this chapter we demonstrate two ways to extend it, along with a probabilistic model to serve as a foundation for future extensions. First, we address the perceived shortcoming of FP compared to GLASSO in terms of APS score, showing that a simple modification to the way FP weights edge probability recovers the same performance as GLASSO (at the cost of other scores). Then we use the implicit causal dependency tree structure of each observation, together with the Matrix Forest Theorem [47, 65] to more generally define our generative node activation model. This leads to a generative model for binary activation data as rooted random spanning forests on the underlying dependency graph. Finally, we use this model to continue where FP left off by alternating between estimation of R and B , in a method we call _Expected Forest Maximization_(EFM). EFM shows small but consistent improvement in performance over FP, at the cost of computation (and unknown convergence) time.

7.1 Forest Pursuit Interaction Probability

Without using stability selection[58], GLASSO is not directly estimating the “support” of the edge matrix, but the strength of each edge. to do similar with FP, we could

directly estimate the frequency of edge occurrence using $R(i, e)$ marginal averages, rather than conditioning on co-occurrence. Simply multiplying each FP edge probability by the co-occurrence probability of each node-node pair gives this as well, which we call FPi: the direct “interaction probability” for each pair of nodes.

7.1.1 Simulation Study Revisited

By doing this simple re-weighting, FPi actually beats GLASSO’s median APS for the dataset, but at the cost of MCC and F-M scores (which both drop to between FP and GLASSO), as Figure 7.1 demonstrates. Similarly, the individual breakdown by graph kind in Table 7.1 shows a similar pattern, with FPi coming close to GLASSO for scale-free networks, but exceeding it for trees and matching for block graphs. Still, the difference is small enough, and at such a significant penalty to MCC and F-M scores over a variety of thresholds, that it is hard to recommend the FPi re-weighting unless rate-based edge analysis is desired, e.g. if Poisson or Exponential occurrence models are desired.

7.1.2 Simulation Case Study

To illustrate what is going on, we have selected two specific experiments as a case study, in Figure 7.2. In the first, BL-N030S01, a 30-node block graph with 53 random walk samples, has FP performing worse than GLASSO and Ochiai, in terms of APS (which is reported in the legends). We see that FP shows high precision, which drops off significantly to increase recall at all. Only a few edges had high probability (which is usually desirable for sparse approximation), and some of the true edges

Table 7.1: Comparing scores for FP, FPi and GLASSO

	FP	FPi	GL
Block			
APS	0.78	0.9	0.9
F-M	0.74	0.57	0.51
MCC	0.7	0.55	0.46
ScaleFree			
APS	0.69	0.76	0.81
F-M	0.67	0.46	0.44
MCC	0.63	0.43	0.36
Tree			
APS	0.9	0.92	0.88
F-M	0.81	0.66	0.53
MCC	0.78	0.65	0.49

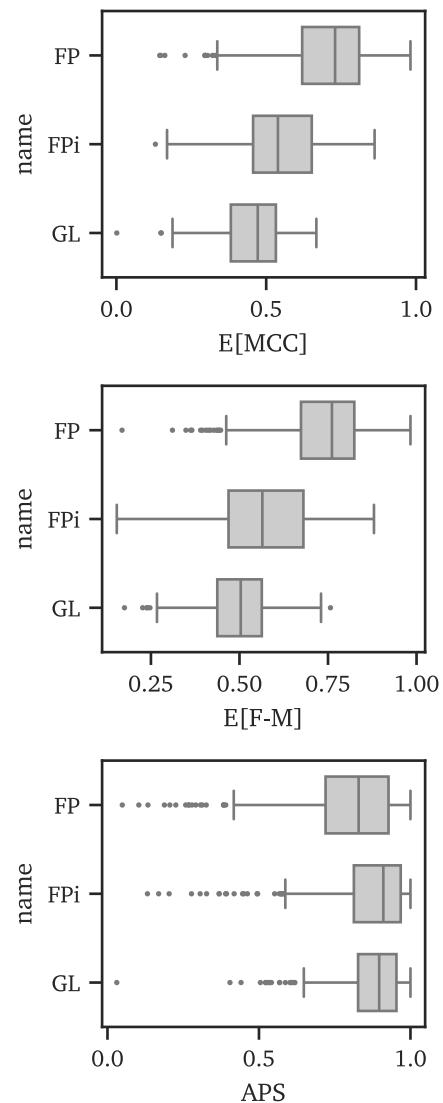


Figure 7.1: FPi shows best APS, lower MCC,F-M

were missed this way. However, FPi rescaling makes rarer edges fall off earlier in the thresholding, letting the recall rise by dropping rare edges, rather than simply the low-confidence ones.

In the second, SC-N300S01 is a 300-node scale-free network with 281 walks. Both FP and FPi show significantly better recovery capability, since enough walks have visited a variety of nodes to give FP better edge coverage. In this graph, no algorithm comes within 0.25 of FP’s impressive 0.88 APS for 300 nodes fewer than that many walks.

7.2 Generative Model

Symmetry of marked directed and undirected trees (symmetry in Q)[\[65\]](#)

7.2.1 Marked Random Spanning Forest (RSFm) distribution

Every spanning forest on a graph described by Q can be thought of as an equivalent spanning tree over a graph augmented with an extra “source” node, which is connected to every other node with a weight $\frac{1}{\beta}$. Sampling random spanning trees on the augmented graph is equivalent to random spanning forests on the original.[\[32\]](#) We can use this fact to create a distribution for node activation sets based on *co-occurrence on a rooted tree*.

A “rooted tree” is a tree with a marked node. In Figure [7.3](#) we see this illustrated, where a randomly sampled tree on the graph augmented with R leads to many subtrees in the original graph. Marking one node (d) at random selects the tree that

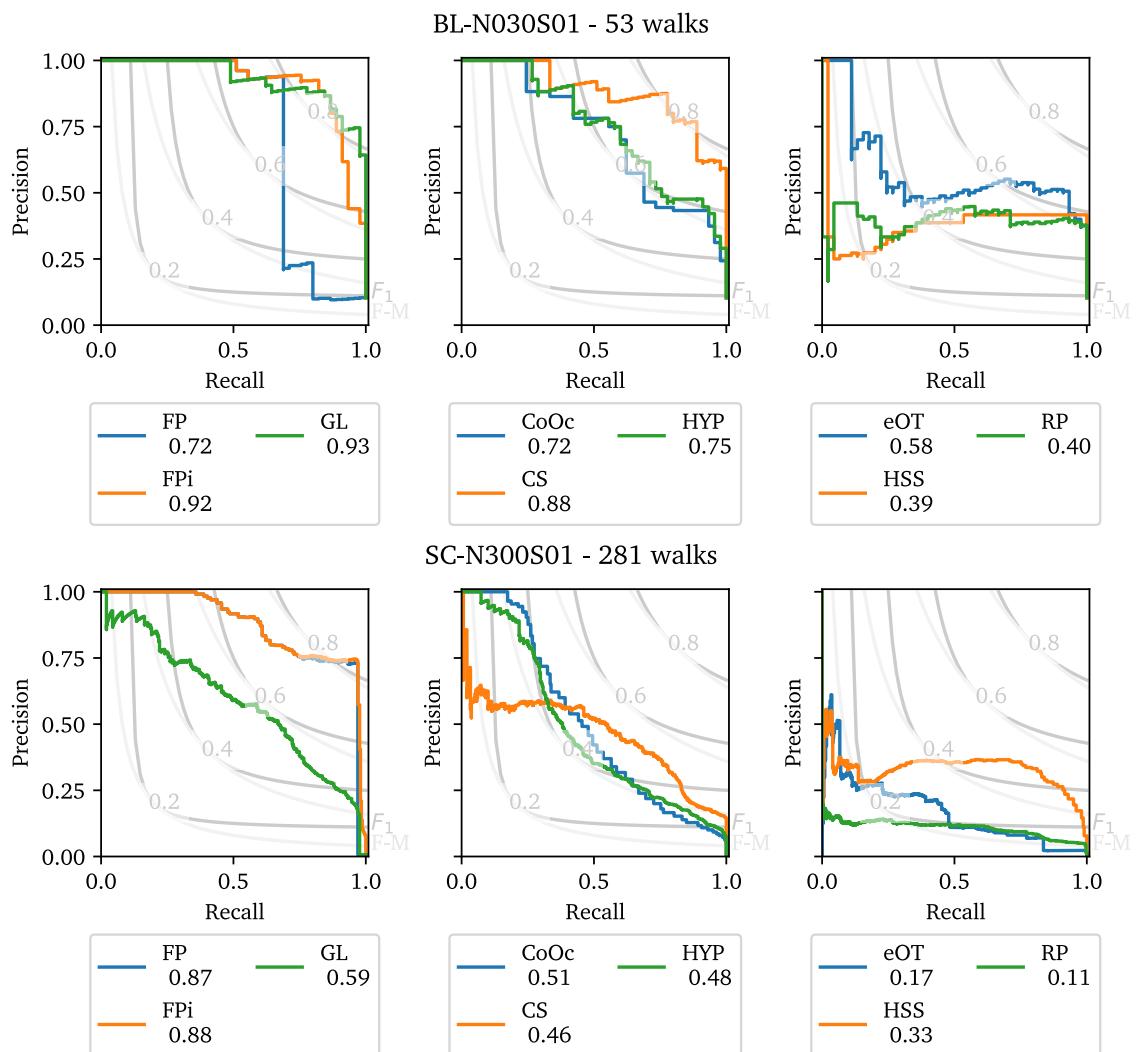


Figure 7.2: P-R curves for two experiments

contains (d,h,e) , which corresponds to record x_1 back in Figure 5.2a.

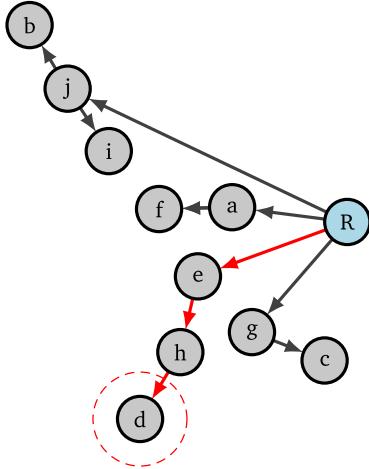


Figure 7.3: Dissemination plan as rooted RST on augmented graph

Note that the marked node does not necessarily need to be the one that the source “injected” to, since the observed activations set is equivalent for any of (d,h,e) being marked. This is an important symmetry when we will not know which node “actually” started each cascade, during inference.

It means that the node activation set and the graph structure are conditionally independent, given a sampled spanning tree on the augmented graph. Sampling efficiently from a spanning tree distribution is a well studied problem, and we can use that efficiency in combination with a node-marking (categorical) distribution to formulate an overall distribution for node activation.

Therefore, RSFm distribution models the probability of emitting node j in the i -th observation as the probability of occurring in the same tree as a marked “root” node ϕ_{ij} , given a graph of z_E edges and a source “distance” parameter β . Since the graph is assumed to be independent of the trees sampled on it, we can use the law of total (conditional) probability:

$$P(x_{ij} | z_E, \phi_{ij}) = \sum_{T \in \mathcal{T}_{+R}} P(x_{ij} | T, \phi_{ij}) P(T | z_E)$$

Incredibly, the probabilities for $P(T | z_E)$ and $P(x | T, \phi)$ all have closed form representations. The spanning tree distribution discussed elsewhere[9, 15] can be used to motivate a spanning forest distribution, which is based on the Laplacian of the augmented graph L_+ .¹ This means that the probability of a spanning tree in the augmented graph is proportional to the probability of a forest in the original, or,

$$P(T \in \mathcal{T}_{+R} | z_E) \propto P(F \in \mathcal{F} | z_E).$$

$$P(x_{ij} | z_E, \phi_{ij}) = \sum_{F \in \mathcal{F}} P(x_{ij} | F, \phi_{ij}) P(F | z_E)$$

Using this equivalence on Kirchoff's matrix tree theorem gives the Chebotarev-Shamis Forest Theorem[65] for the partition function over spanning forests $Z_{\mathcal{F}}$, which gives a closed form for Equation 5.2 over all spanning forests (not just spanning trees on a subgraph). If $\mu(F)$ is a measure on forests, e.g. the product of weights of edges, then:

$$P(F | z_E) = \frac{1}{Z_{\mathcal{F}}} \mu(F)$$

$$Z_{\mathcal{F}} = \det(I + \beta L)$$

There are a variety of techniques already cited for sampling from this distribution, because of the connection with spanning trees on an augmented graph. This means we can sample from RSFm with only one modification from normal spanning tree

¹Here, $L_+(G)$ and $L_+(G/e)$ represent the Laplacian of G_{+R} and G_{+R} without edge e , respectively.

distributions:

- Generate random spanning trees on G_{+R} , according to $P(F | z_E)$.
- Uniformly sample a marked node ϕ .
- Activate all nodes connected to ϕ in the induced spanning forest.

To do inference, however, we may again use the forest kernel Q , which gives us probabilities for node co-occurrence on trees in a spanning forest. We only need to combine entries of Q for the observed nodes to determine the likelihood of an observation. $\sum_{\mathcal{F}} P(x_{ij} | F, \phi_{ij})$ would be the probability of a node occurring (or not) on the same subtree as the marked node, and as previously discussed, this is just $Q(u \in V_i, \phi)$ for each node that is activated, and $1 - Q(v \notin V_i, \phi)$, for each that is not. Note that this assumes we marginalize over a uniformly distributed ϕ .

7.2.2 Model Specification

The overall model that fills out the gaps left in Equation 5.5 has the form:

$$\pi_{e \in E} \sim \text{Beta}_E(\alpha, 1 - \alpha)$$

$$z_{e \in E} \sim \text{Bernoulli}_E(\pi_e)$$

$$\phi_{i \in I, j \in J} \sim \text{Categorical}_I(\theta)$$

$$x_{i \in I, j \in J} \sim \text{RSFm}_K(\beta, z_e, \phi_n)$$

The above still has the problem of needing an estimate for $Q = (I + \beta L_z)^{-1}$ based on the graph described by the active edges in z . It could be done with a monte-carlo sampler, though the nested likelihood may prove difficult.

The conditional independence mentioned previously could lend itself to a collapsed Gibbs-sampling scheme. Each observation is a marked node and a sampled random spanning forest, which can equivalently be described as a sampled spanning tree over the activated nodes, like originally discussed in Equation 5.2. This works because we can marginalize out the equal marked-node probability ϕ if we assume a uniform probability of selection for each node, and because we can jointly use the desire-path model to derive an estimate for z_E from the *internal* edge samples of RSFm.

Beginning with the FP point estimate, each edge in every spanning subtree can be efficiently resampled according to the *Bayesian Spanning Tree* distribution from Duan and Dunson [9]. Once every edge in a tree has been resampled, the overall estimate for the desire path network can be updated, and sampling can continue. This would be very similar to the way collapsed Gibbs sampling works for Latent Dirichlet Allocation[CITE], but with edges selected from a spanning forest distribution instead of “topics” from a multinomial. Derivations and implementation of such a scheme is left for future work.

7.3 Expected Forest Maximization

Another possibility is to approach the problem as a kind of matrix factorization, jointly estimating B and R in an alternating manner. Where *Forest Pursuit* was an empirical Bayes estimate for R , alternating from there between B and R leads to a simple Expectation Maximization scheme:

Alternate embedding the node activations as edge activations and combining these under a desire path model to estimate the graph structure.

7.3.1 Factorization & Dictionary Learning

Jointly finding a sparse representation of data using a linear combination of basis vectors (called a “dictionary”), and finding an optimal dictionary with which to do the embedding, is called “sparse dictionary learning”. One of the original proposed methods to solve this was the Method of Optimal Directions (MOD)[77], which uses a sparse “pursuit”-like method to find the representation vectors, and then uses that representation to find $\hat{B} \leftarrow X R^+$.² However, our desire path estimation for independent (arcsine) Bernoulli edges gives an efficient workaround to needing the pseudo-inverse. Based on our interpretation of the forest kernel, counting co-occurrences on trees is equivalent to estimating the inverse of the regularized graph laplacian, anyway. Each Forest Pursuit estimate can yield a new distance $d_Q(u, v)$, which can be used as a regularized shortest-path distance for small enough β . Then, new steiner tree estimates will approximate the modes of each spanning tree distribution, which can be used for a new desire-path estimate of the edges, and so-on.

Probabilistically, we are finding the expectation over spanning forests (collapsing over uniform ϕ) in the form of calculating Q , and then maximizing the likelihood of each observation through *Forest Pursuit*. The proposed algorithm *Expected Forest Maximization* is outlined in Algorithm 7.1.

² R^+ is the pseudo-inverse of R .

Algorithm 7.1 Expected Forest Maximization (EFM)

Require: $X \in \mathbb{B}^{m \times n}, d_K \in \mathbb{R}_{\geq 0}^{n \times n}, 0 < \alpha < 1$

Ensure: $R \in \mathbb{B}^{m \times \binom{n}{2}}$

```
1: procedure EFM( $X, d_K, \alpha, \beta, \epsilon$ )
2:    $R \leftarrow \text{FORESTPURSUIT}(X, d_K)$ 
3:    $\hat{\alpha}_m \leftarrow \text{DESIREPATHBETA}(X, R, \alpha)$ 
4:   while  $\|\hat{\alpha} - \alpha\|_\infty > \epsilon$  do
5:      $\alpha_m \leftarrow \hat{\alpha}_m$ 
6:      $Q \leftarrow (I + \beta L_{\text{sym}}(\alpha_m))^{-1}$ 
7:      $d_Q \leftarrow d_K$ 
8:      $R \leftarrow \text{FORESTPURSUIT}(X, d_Q)$ 
9:      $\hat{\alpha}_m \leftarrow \text{DESIREPATHBETA}(X, R, \alpha_m)$ 
10:    end while
11:    return  $\hat{\alpha}_m$ 
12: end procedure
```

In practice, we limit iterations to less than 100, though that limit is not reached in our testing. We use a small $\beta = 10^{-3}$ to approximate shortest path distances, and we can do so *within* the forest pursuit loop to avoid inverting large matrices. Instead of d_K , the graph structure itself (once an initial estimate is made) can be passed to *Forest Pursuit*, where each subgraph can estimate d_{Ki} independently. After all, shortest paths in a subgraph when we assume *no hidden nodes* will be the same as in the global graph. Note as well the use of the symmetric normalized Laplacian L_{sym} . We use this to enable the splitting of inversions like this, because the global Laplacian has higher possible degrees than the subgraph, and we do not wish to bias Steiner tree estimation by node degree. Not doing so leads to rapid divergence of the loss function, as all MSTs will tend toward star-graphs around the highest-degree node.

7.3.2 EFM Simulation Study

One-shot Forest Pursuit appears to perform quite well, so it's useful to quantify the expected gain in performance by repeating it an unknown number of times. There are no generic guarantees for EM convergence, though anecdotally the number of iterations was limited in our experiments to under a thousand, and that limit was never hit while using a convergence parameter of $\epsilon = 1e - 5$.

The distribution of $E[MCC]$ score change vs. FP is shown in Figure 7.4.

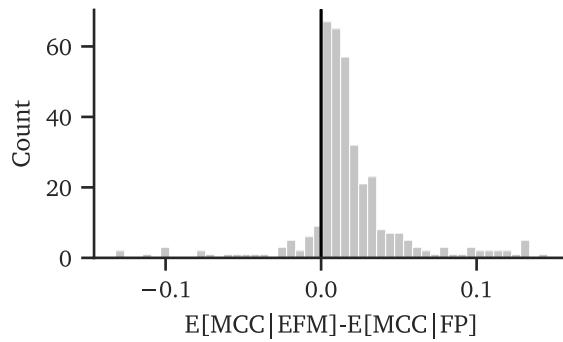


Figure 7.4: Change in Expected MCC (EFM vs FP)

While useful, it's not clear whether individual edges are more likely to be “true” edges, given a bigger change in EFM score. To test this, a logistic regression was performed for every experiment in MENDR against the true edge values, using the change in scores on those edges between FP and EFM as training data. To avoid overfitting, a significant amount of regularization was applied, chosen using 5-fold crossvalidation. The coefficients for all experiments are shown as a histogram in Figure 7.5.

The graph kind did not make a significant difference to EFM improvement, but overall log-odds improvement is very low. Still, the value is positive accross the entire

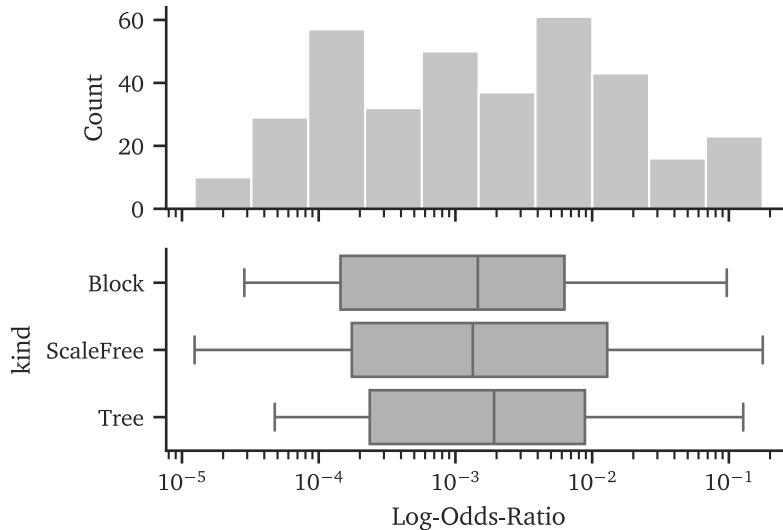


Figure 7.5: Logistic Regression Coef. (EFM - FP) vs. (Ground Truth)

dataset, so EFM does have a very small-but-nonzero impact on improving edge prediction.

The runtime graphs can also be updated, with EFM shown in Figure 7.6 and Figure 7.7 against FP and Glasso. EFM still ran significantly faster than GLASSO in this region. However, the scaling with network size is no longer constant-time, especially since convergence used above is the max-abs error, which requires that *every node* reach a minimum level of convergence and might take much longer, overall.

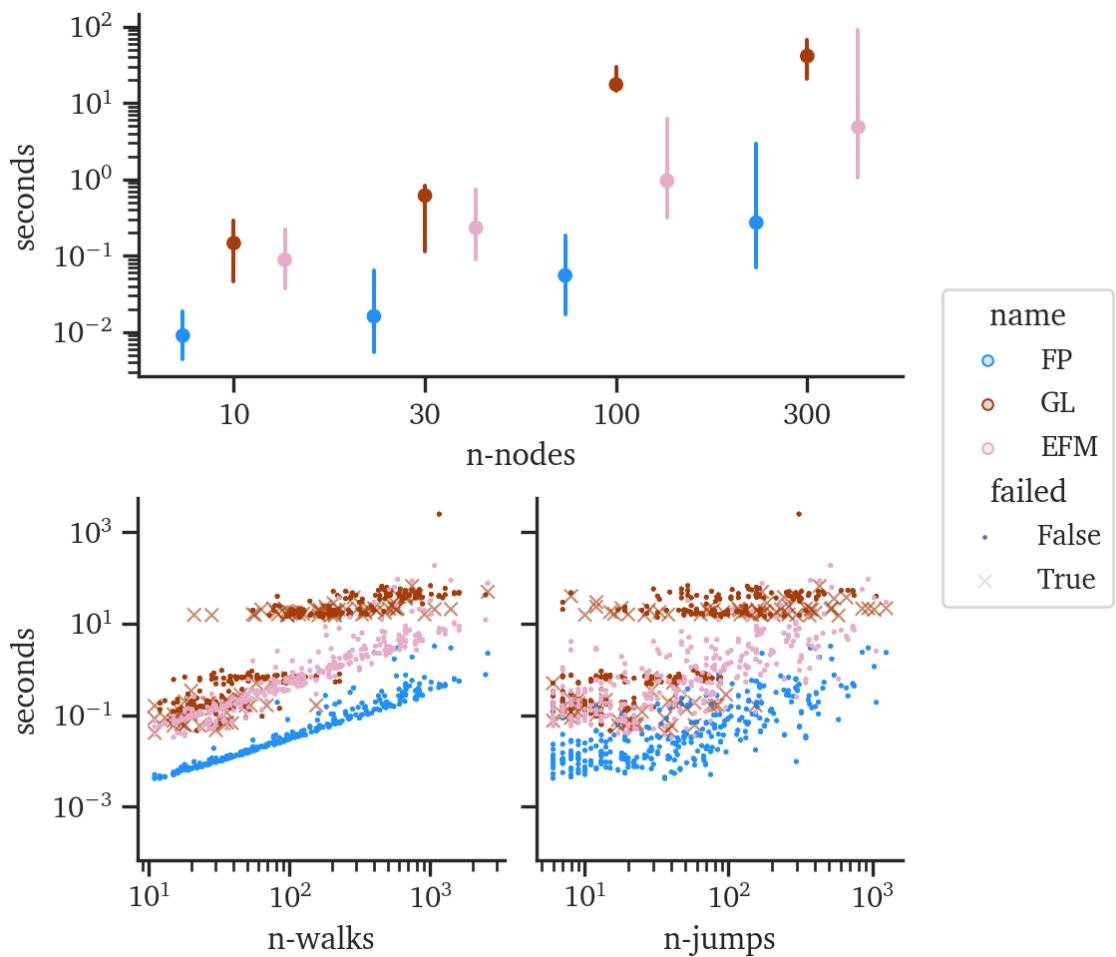


Figure 7.6: Runtime Scaling (Forest-Pursuit vs GLASSO)

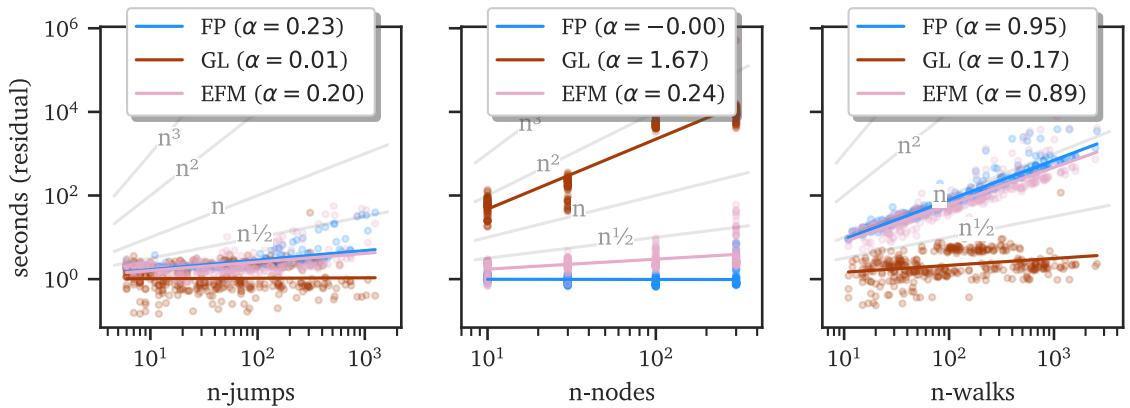


Figure 7.7: Partial Residuals (regression on computation time)

Part III

Applications & Case Studies

Chapter 8: Qualitative Application of Relationship Recovery

A more realistic test of *Forest Pursuit*'s behavior is in unsupervised, qualitative analysis of complex systems and networks. In this chapter we will recreate two well-known bipartite datasets from network science, and compare several network recreation methods on them. As we have discussed previously[33], there is some interest in quantifying network assortativity for social networks, and whether the positive assortativity effect is a methodological artifact. Assuming an analyst is interested in social network *dependencies*—like which nodes are directly *influencing* which others—we provide a glimpse at how correcting for clique bias with *Forest Pursuit* (FP) affects assortativity.

In the second case study, we also investigate how centrality measures can be adversely affected when clique-bias is not accounted for, especially when appearance in cliques is known to be rare for certain node types.

8.1 Network Science Collaboration Network

Following in the steps of Newman and Girvan [69], we recreate a network from co-authorships in works cited by two literature review papers, Newman [72] and [CITE]. The list of all cited papers was reconstructed via web-of-science queries, and these were used to construct author “activation” observations.¹ A baseline co-authorship network was initially constructed using the largest connected component, and a modularity-maximizing community detection algorithm [Clauset et al. [68];Findingevaluatingcommunity_Newman2004] was used to highlight community structure. Nodes and papers were retained if they had two or more papers and authors listed, respectively. This baseline network can be seen in Figure 8.1, where (to keep with the treatment in Newman and Girvan [69]) we have added a quotient network of the communities themselves.

The detected assortativity $r = 0.059$ is small, but positive. As expected, each community tends to have a dominant “clique”, since many members of these groups tend to mutually author papers together (simultaneously).

However, if we are after a “social network” that describes social influence of individuals w.r.t. others, then we are hoping to recover collaborative *dependency relationships*.

- Who is causing who to join papers?
- Who are the central “collaborators” that influence the paper writing of many

¹An additional paper (Sneppen and Newman [79]) was added to ensure the the largest connected component contained necessary communities that were originally missing compared to Newman and Girvan [69].

Network Scientist Co-Occurrence Graph ($r = 0.059$)

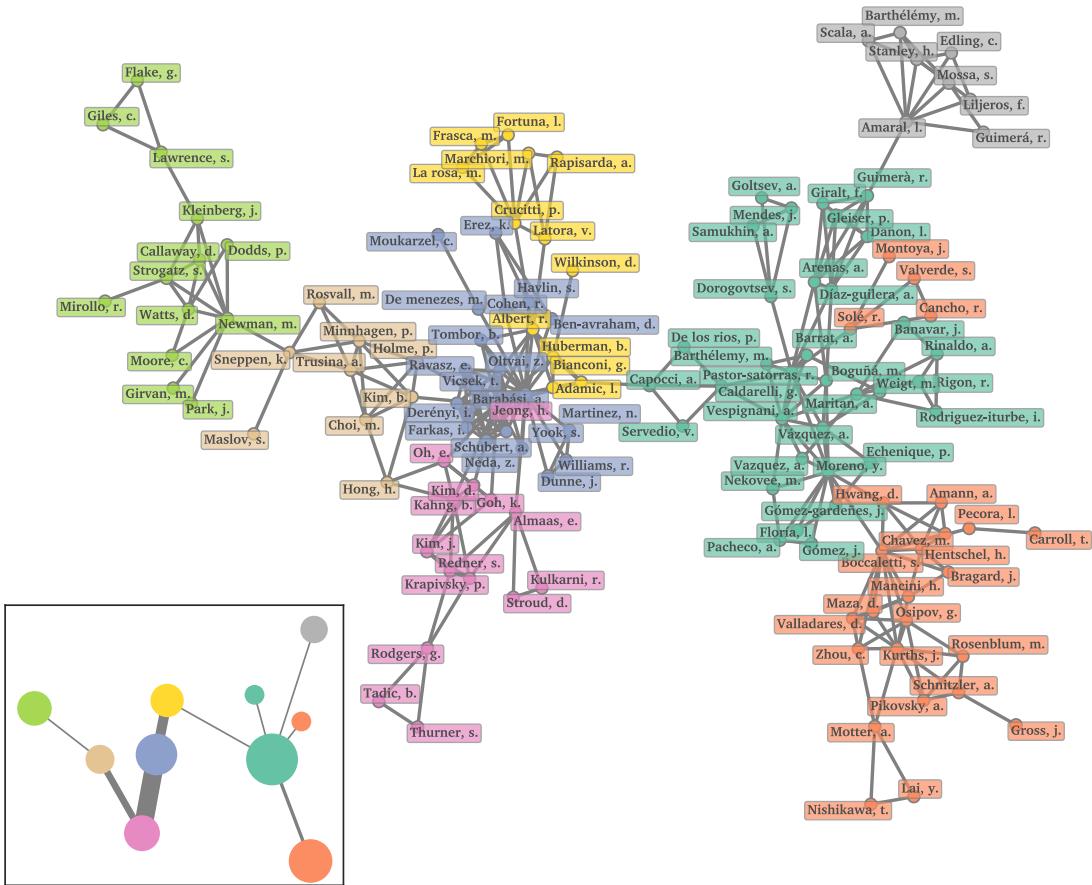


Figure 8.1: 134 Network scientists connected by co-authorship

others?

With some background knowledge of how university labs tend to work (with students, post-docs, advisors, colleagues, etc.), we might not believe that every member of a community has significant influence on the writing of every other member. So, to estimate a dependency network of collaboration relationships, we could turn to a Chow-Liu tree, which is shown in Figure 8.2.

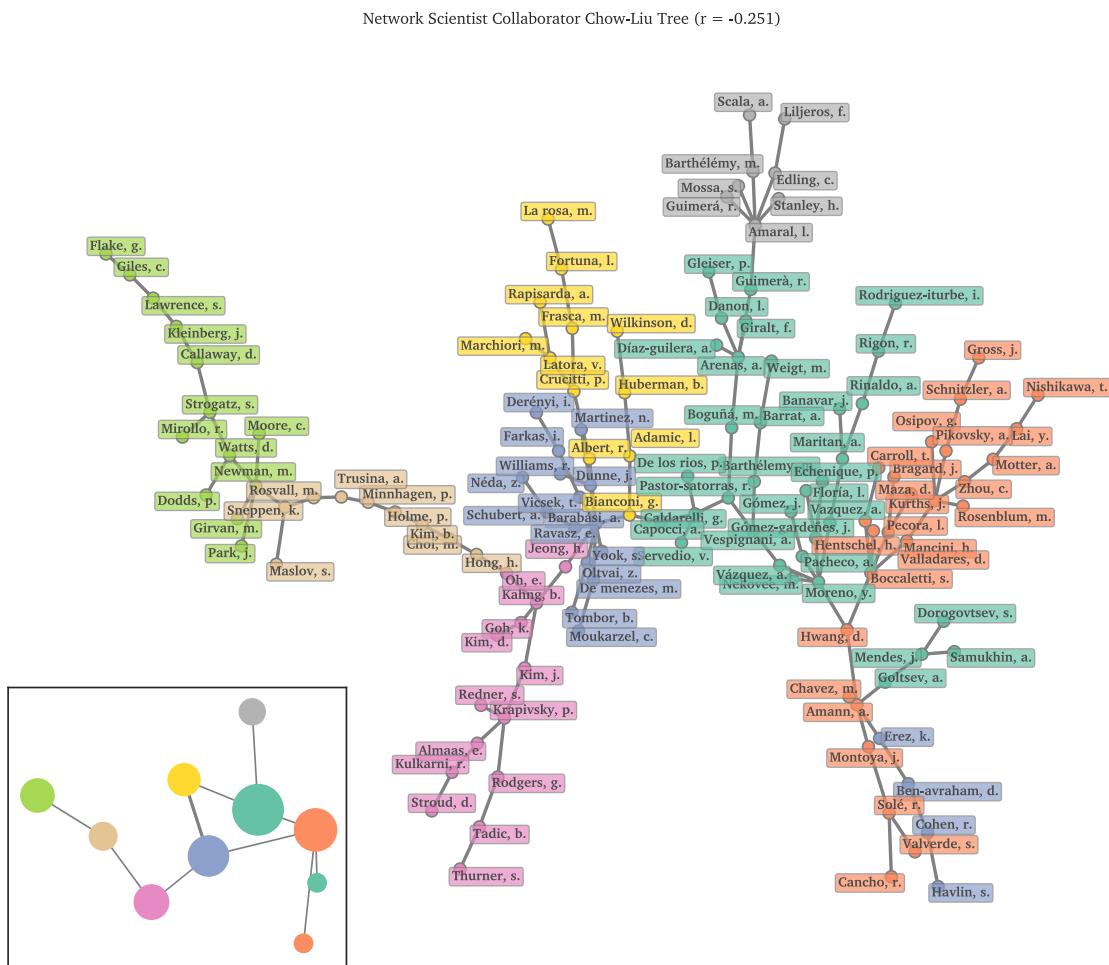


Figure 8.2: Chow-Liu tree of NetSci collaborator dependency relationships

This dependency is much more sparse, and indeed the assortativity coefficient has dropped to $r = -0.251$. In academic writing, we might even expect a lowered

assortativity, since students that may not go on to be prolific in their original field would still seek out prolific advisors, initially. Unfortunately, enforcing a tree structure has some negative side effects. Overlaying the original community structure from Figure 8.1 onto the tree and calculating a quotient graph shows that the community structure is impacted by the change. Many of the communities are “unrolled” into long chains of authors, since trees cannot allow small loops or cycles. This makes for a shortest-path distance between community hubs (in the same field) to be upwards of 9-10 jumps, which goes against the scale-free/small-world nature we expect from social systems.

With *Forest Pursuit*, we can attempt to correct for “clique bias” in the measurement of dependency relationships, while allowing for flexibility in the global network structure. The FP recovered collaborator graph is shown in Figure 8.3.

The communities from the co-occurrence network are entirely preserved, with a nearly identical community structure in the graph quotient compared to the co-occurrence graph. But now we see an assortativity that is close to zero (slightly negative at $r = -0.069$). This brings the assortativity more in line with the World Wide Web networks, or even close to results for random preferential attachment networks that are zero in the limit[74].

Relative to the tree network, there are not so many long chains, as many authors have been allowed to “loop-back” and have relationships with nearby colleagues, reducing the path distance between communities in the process. Still, like the trees, FP tends to reduce the degree of each node, which is summarized in Figure 8.4.

From a domain modeling perspective, it might be reasonable to assume that

Network Scientist Collaboration Network Estimate ($r = -0.069$)

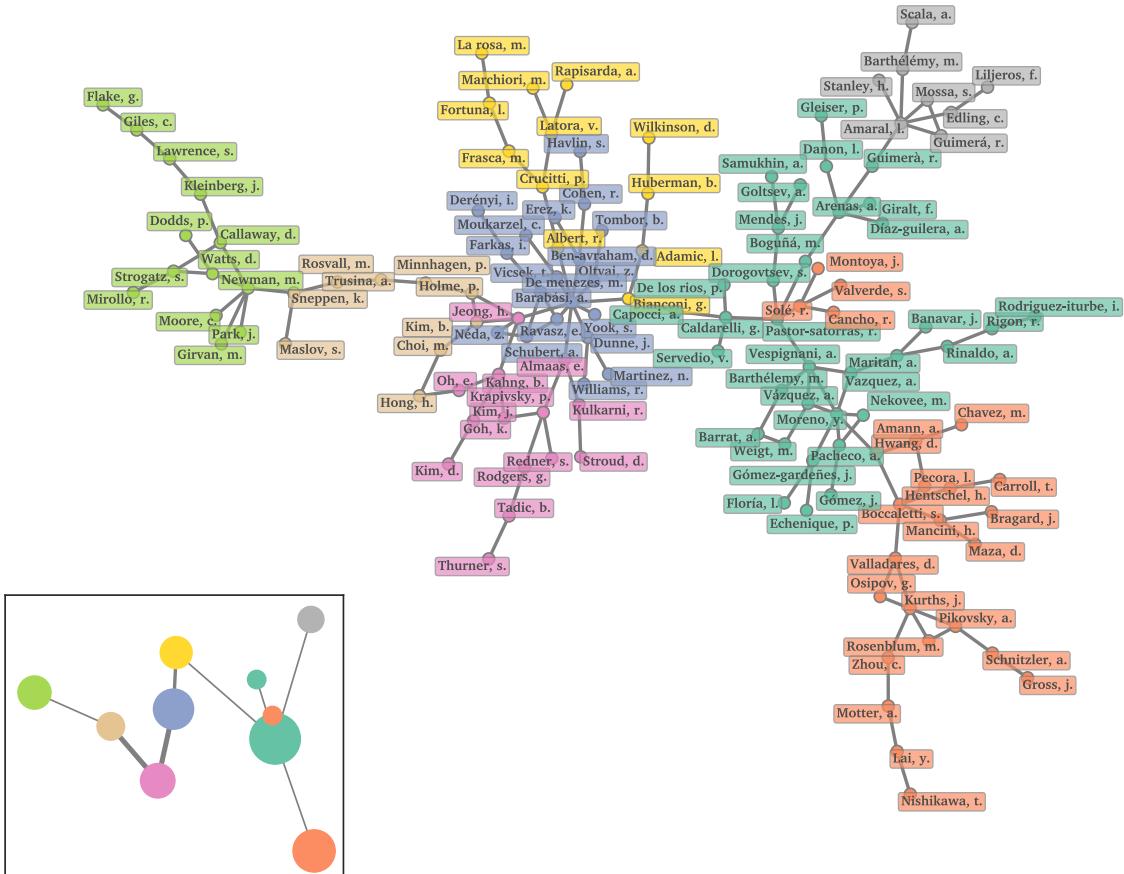


Figure 8.3: Forest Pursuit estimate of NetSci collaborator dependency relationships

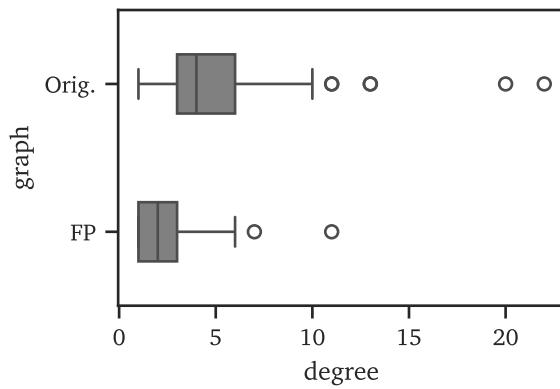


Figure 8.4: Degree distributions of FP vs co-occurrence social networks

any given author has 2-3 influential collaboration relationships shown, especially considering students and post-docs are likely to constitute a good number of nodes. Even from a logistics perspective (during a given time window working with an advisor) the number of times a student asks/is asked to participate in a paper has to be limited, given average publishing rates.

Only rare individuals would have upwards of 10 influential relationships, with a mean closer to the 2-3 of the FP network, rather than a full quarter having 10 and the median being 4-5 (as in the co-occurrence network). In addition, recall that this is a social network derived from literature review paper citations, not exhaustive inventories of each author's work, so every influential relationship should not be shown in this sample.

8.2 Les Miserables Character Network

Another famous network derives from Knuth [82] via mining character co-occurrences in chapters of *Les Misérables* (1862), Victor Hugo's sprawling saga of [in]equality and [in]justice in 19th-century France. This dataset is often reported as being inherently a *graph*, but we have reconstructed the underlying bipartite observations for the purpose of removing clique bias from the network reconstruction. In the original network, an extra count is added to the weight of an edge every time a character occurs in a chapter with another. Once again, we have only retained characters/chapters that appeared with two or more chapters/characters, respectively. In Figure 8.5, we run the same community detection scheme to improve the visual understandability of

the network, along with inset quotient graphs.

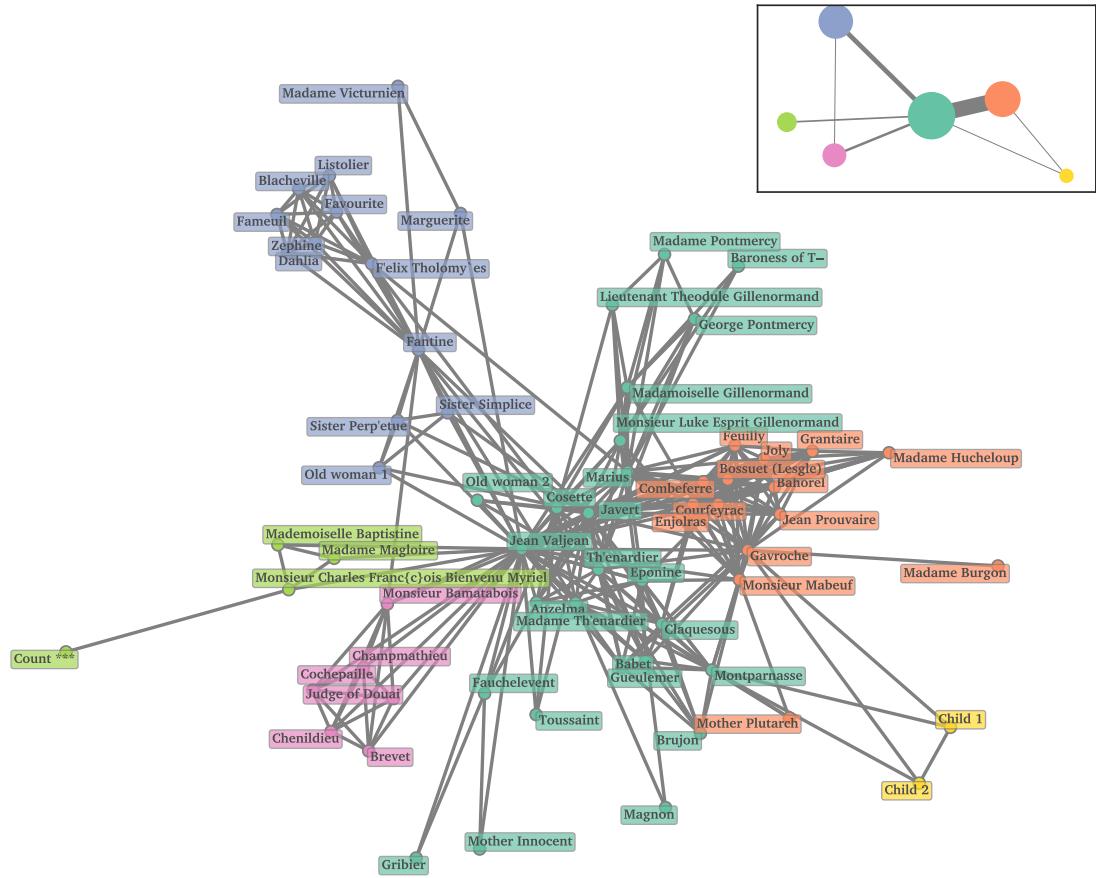


Figure 8.5: *Les Miserables* character co-occurrence network

Note that the communities demonstrate clear clique-like behavior. On one hand, this does make assessment of clusters easier, since characters in roughly the same scenes are densely grouped together. However, this network makes it difficult to parse relationships, such that backboning would become necessary.

We should also ask *what we want* out of this “social” network. Another way we might think about a social network of characters is “which characters *influence* the appearance (or not) of which other characters?” By extension, “which characters are *significant*, in the sense that they dictate the appearance of more characters than

others?” From a domain modeling perspective, this is like asking *how* an author is deciding which characters to include in a chapter. By not correcting for clique-bias, we are implicitly assuming that Victor Hugo would be setting out to write a chapter and immediately writes down a list of every character, independently, should appear. Instead, it’s likely that the desired appearance of certain characters in a chapter *leads to* the inclusion of others, as the plot requires.

Because the inclusion of certain characters *leads to* the inclusion of others (by this model), we can reasonably model the authorial “character inclusion process” as *spreading* from character to character. With this in mind, we apply Forest Pursuit to correct for clique-bias, and show the resulting dependency network (with the original community partition) in Figure 8.6.

The network has been filtered to only include necessary edges to retain overall connectivity. As before, our community structure shown in the quotient graph has been preserved, even with the significant edge density reduction.

Applying these networks, we might wish to distinguish through them which characters are “main” and which are “supporting”, though more on a gradient than in a binary/classification sense. One way to do this is through centrality measures [CITE], which estimate the “importance” of nodes in various ways. Eigenvector centrality, specifically, finds the importance of nodes relative to the importance of each node’s neighbors²

The topology of a network greatly impacts the centrality measure recovered. In

²Regularized eigenvector centrality is equivalent to PageRank centrality, *i.e.* the seminal work from [CITE], of Google fame.

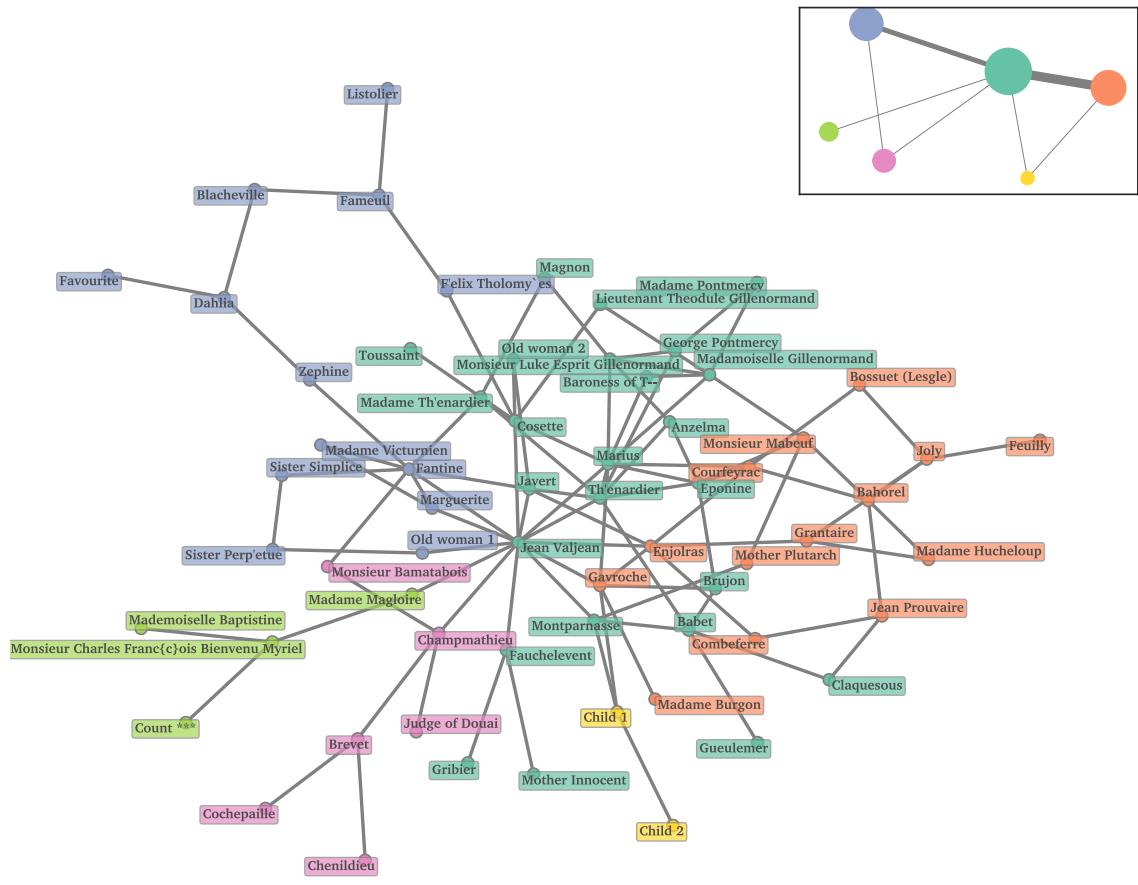


Figure 8.6: *Les Misérables* character dependency network (Forest Pursuit)

Figure 8.7 we show a bump-plot of the ranks of the top 15 most central nodes for both the original co-occurrence network and the FP network estimate. In the original network, centrality rewards characters appearing in mutual cliques as being central, especially the “revolutionary” group (shown in orange in Figure 8.6 and Figure 8.5). Are these really the most important characters? Even Javert, though being an important recurring character, largely plays the role of generic law enforcement throughout, while characters like Marius play central motivating roles across several distinct social groups throughout the novel.

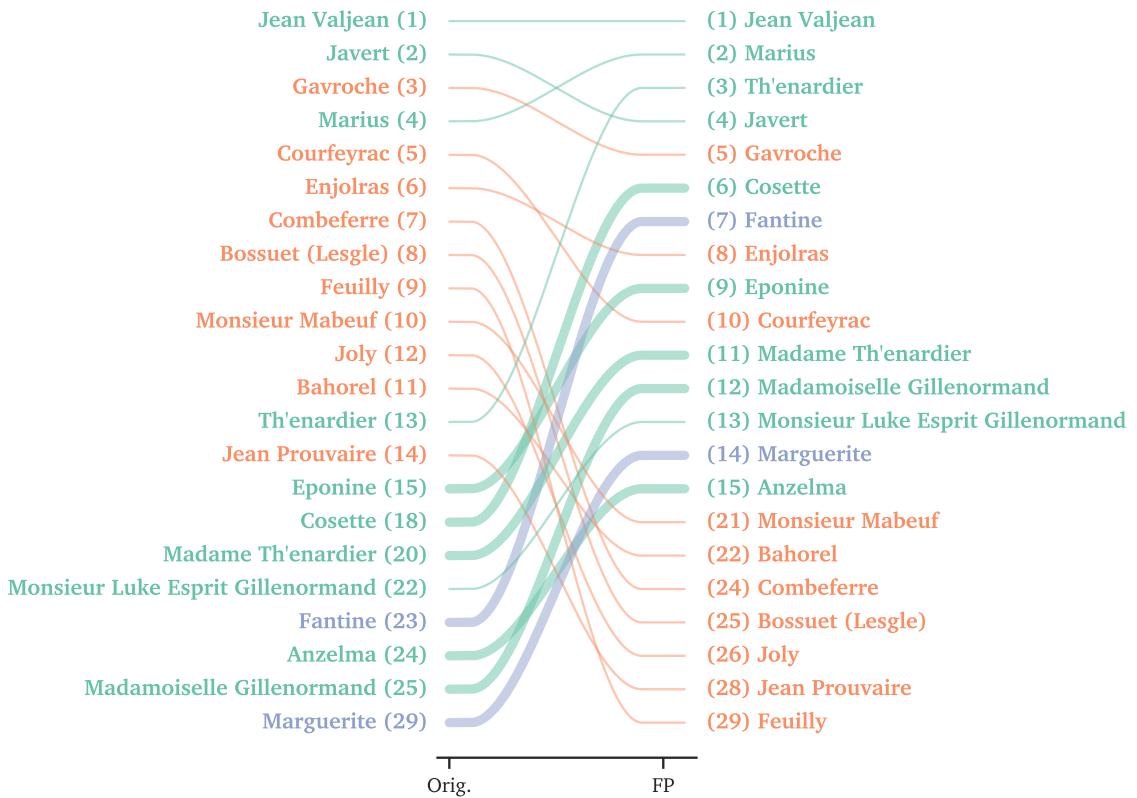


Figure 8.7: Changes in character centrality ranking for FP vs co-occurrence

This brings up another benefit of correcting clique-bias: estimating character importance *despite* systematic interaction-reduction bias from authors. As made famous

by the so-called “Bechdel test” [CITE], authors historically tend to systematically reduce the agency and interaction rates of women in literature[1]. We highlight every female character in Figure 8.7, and which shows that the co-occurrence network had a single female character in the top 15 most central (Eponine, ranked 14th). By correcting for clique-bias, *every* female character significantly increases in centrality ranking. In context, titles of entire books and volumes of *Les Misérables* are named after women we would consider crucial to the plot (e.g. Fantine, Cosette, and Eponine).

As Stuhler [1] indicates, however, authors are likely to under-represent their agency and interactions with other characters throughout the story, so networks based primarily on co-occurrence will likewise systematically underestimate their importance.

Chapter 9: Recovery from Working Memory & Partial Orders

This whole time we have assumed that the ordering of nodes was unknown, or at the very least unreliable. However, there are frequently cases, especially in text processing applications, where we have some sense of an ordering on activations. By *partial order* on a set (a “poset”), we mean that all elements are either comparable as greater (before) or less (after), or incomparable. The set of posets is therefore precisely isomorphic to the set of *directed acyclic graphs*, based on reachability.

Our original example with authors might be thought of as a poset: (i) precedes/asks (f) and (j), (j) precedes/asks (b), but (b) and (f) are incomparable. We don’t know if (i) asked (f) before or after (j) asked (b).¹

Sadly our co-authorship example does not often include the (partial) order of author additions,² but other common network recovery problems do have an inherent order to them. A very common need is to recover concept association networks, whether from lists of tags or directly from a corpus of written language. What’s

¹Interestingly, in the case where the node activations *are* given a total order (in the form of *timestamps*), Gomez-Rodriguez et al. [50] derive an algorithm called *NetInf*. It utilizes sums over minimum spanning trees that satisfy time constraints, similar to a special case of Forest Pursuit where all activations have a total ordering.

²though some domains do imbue special meaning to the listed author order, a fact that might be interesting to investigate in future network recovery efforts!

needed is an assumption on how the observed partial order of concepts is generated. Hills et al. [39] proposes a “foraging” mechanism, so that concepts get sequentially recalled from “semantic patches” of nearby concepts in memory. The partial order comes from our ability to maintain more than one concept in working memory, so that the next concept can be “foraged” from any of the other recently recalled ones[52, 91].

In this section, we briefly cover a method for network inference by Jun et al. [41] that utilizes partial order information from ordered lists of concepts, called INVITE. We use it to demonstrate improvement over bag-of-words and markov assumptions for downstream technical language processing [7] tasks, as originally demonstrated in [24, and 19]

Finally, we show that using *Forest Pursuit* for partially ordered data can still be quite useful for network backboning, and for a fraction of the computational cost. We investigate a network recovery task from verbal/semantic fluency data [31], which involves recovery of a network of animal relationships from memory and recall experiments. Even without directly using partial order information, proper data preparation along with previously-discussed (un-ordered) recovery methods can lead to significantly improved network backboning and analysis capability

9.1 Technical Language Processing with INVITE

Maintenance work orders are often represented as categorized data, though increasingly quantitative use of a technician’s descriptions is being pursued by maintenance

Table 9.1: Maintenance Work Order as categorized vs. tagged data

(a)	
<i>“Hydraulic Leak at cutoff unit; Missing fitting replaced”</i>	
Categorization:	
Subsystem	142_HYD_SYSTEM
Error Code	ERR_142A
Action Taken	PART_ORDERED
Tags:	
objects	cutoff_unit, hydraulic, fitting
problems/actions	leak, replace

management operations [25, 30]. Tags are another way to flexibly structure this otherwise “unstructured” data, which Table 9.1 shows in comparison to more traditional categorization.

Whether entered directly or generated from text by keyword extraction, the tags will tend to have ordering information readily available. A traditional way to model this kind of text is through either bag-of-words (the co-occurrence node activation data already discussed) or as a sequence of order-n markov model emissions. Unlike the clique bias from before, assuming markov jumps for each observation leads to a different kind of bias, with higher precision but reduced recall as shown in Figure 9.1.

Without knowing the underlying dependency relationships, it’s difficult to estimate which edges were used by a random-walker, since subsequent visits in memory to a “tag” are not being reported once a technician first adds it. [41] call this an “Initial-visit-emitting” random walk, or INVITE for short. To more accurately recover the network, they suggest maximizing the absorption probability for each step of a partial order, individually, knowing which nodes have already been activated.

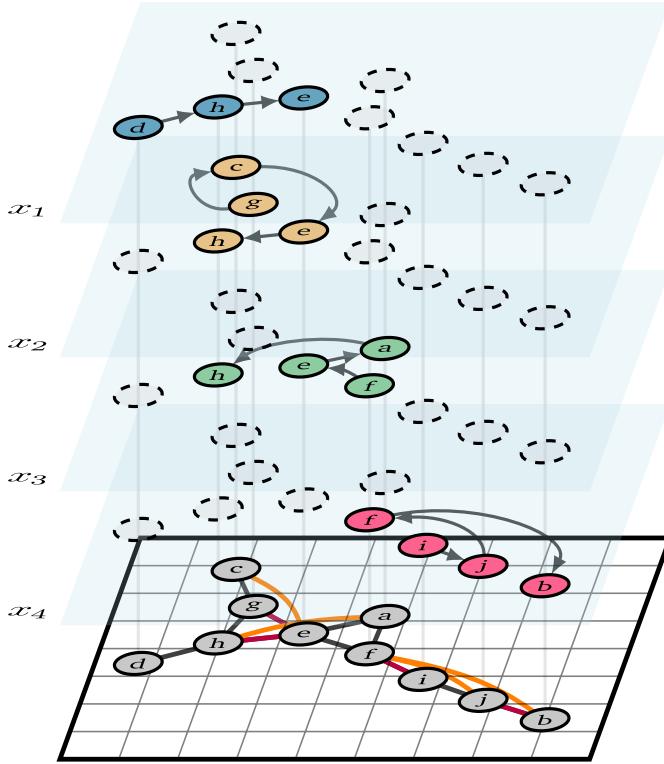


Figure 9.1: Partial-order edge measurements with Markov assumption

9.1.1 Optimizing absorbing-state probabilities

Say the set of components or concepts that have a corresponding tag in our system is denoted by the node-set N . A user-given set of T ³ for a specific record can be denoted as a Random Walk (RW) trajectory $\mathbf{t} = \{t_1, t_2, t_3, \dots, t_T\}$, where $T \leq N$. This limit on the size of T assumes tags are a set of unique entries. Any transitions between previously visited tags in \mathbf{t} will not be directly observed, making the transitions observed in \mathbf{t} strictly non-Markov, and allowing for a *potentially infinite* number of possible paths to arrive at the next tag *through previously visited ones*. Instead of directly computing over this intractable model for generating \mathbf{t} , the key insight from the original INVITE

³While some sources use “tagging” as a proxy for a set of strictly un-ordered labels (as in multi-label classification), we preserve the mechanism by which the tags were generated in the first place, i.e., in a *specific order*.

paper [41] comes from partitioning \mathbf{t} into $T - 1$ Markov chains with absorbing states.

Previously visited tags are “transient” states, and unseen tags are “absorbing”. It is then possible to calculate the absorption probability into the k transition ($t_k \rightarrow t_{k+1}$) using the *fundamental matrix* of each partition. If the partitions at this jump consist of q transient states with transition matrix among themselves $\mathbf{Q}_{q \times q}^{(k)}$, and r absorbing states with transitions into them from q as $\mathbf{R}_{q \times r}^{(k)}$, the Markov transition matrix $\mathbf{M}_{n \times n}^{(k)}$ has the form

$$\mathbf{M}^{(k)} = \begin{pmatrix} \mathbf{Q}^{(k)} & \mathbf{R}^{(k)} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \quad (9.1)$$

Here $\mathbf{0}$, \mathbf{I} represent lack of transition between/from absorbing states. It follows from [76] that the probability P of a chain starting at t_k being absorbed into state $k + 1$, is given as

$$P(t_{k+1} | t_{1:k}, \mathbf{M}) = \mathbf{N}^{(k)} R^{(k)} \Big|_{q,1} \quad (9.2)$$

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1}$$

The probability of being absorbed at $k + 1$ conditioned on jumps $1 : k$ is thus equivalent to the probability of observing the $k + 1$ INVITE tag. If we approximate an a priori distribution of tag probabilities to initialize our chain as $t_1 \sim \text{Cat}(n, \theta)$ (which could be empirically derived or simulated), then the likelihood of our observed tag chain \mathbf{t} , given a transition matrix, is:

$$\mathcal{L}(\mathbf{t} | \theta, \mathbf{M}) = \theta(t_1) \prod_{k=1}^{T-1} P(t_{k+1} | t_{1:k}, \mathbf{M})$$

Finally, if we observe a set of tag lists $\mathbf{C} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_c\}$, and assume θ can be

estimated independently of \mathbf{M} , then we can frame the problem of structure mining on observed INVITE data as a minimization of negative log-likelihood. A point estimate for our association network given \mathbf{M} can be found as:

$$\mathbf{M}^* \leftarrow \operatorname{argmin}_{\mathbf{M}} \sum_{i=1}^C \sum_{k=1}^{T_i-1} -\log \mathcal{L}\left(t_{k+1}^{(i)} \mid t_{1:k}^{(i)}, \mathbf{M}\right)$$

This (deeply nested) likelihood can now be optimized using standard solvers, and our reference implementation uses stochastic gradient descent via PyTorch [36].⁴

9.1.2 Application: Mining Excavator MWOs

To assess the applicability of the INVITE-based similarity measure to real-world scenarios, we apply our model to tags annotated for a mining dataset pertaining to 8 similarly-sized excavators at various sites across Australia [34, 37].

The tags were created by a subject-matter expert spending 1 hour of time in the annotation assistance tool nestor [26], using a methodology outlined in a previous benchmarking study for that annotation method [30].

That work compared the ability of tags to estimate survival curves and mean time-to-failure, when compared with a custom-designed keyword extraction tool based on classifying the maintenance issues by subsystem. While certain sets of tags were able to predict time-to-failure with high accuracy for certain subsystems, a key problem identified in that work is knowing a priori which tags best indicate when a subsystem is failing?

⁴The n-gram markov transition models (MC1, MC2) trained for comparison vs INVITE were trained using pomegranate[29].

Which tags best represent a given subsystem?

Some tags are sufficient (albeit unnecessary) conditions to indicate a subsystem. That the “hydraulic” tag indicates a *Hydraulic System* MWO is obvious, but so might a “valve” tag—“hydraulic” is implied but not present. Consequently, we can treat the problem of assigning tags to represent a subsystem as a semi-supervised multi-class classification problem in the style of Avrachenkov et al. [32]. Like in that work, we need to know a selection tag→subsystem assignments, as well as network of weighted tag-tag edges.

Then, if we can compare the semi-supervised tag classifications to a ground-truth classification by a human annotator (which are available for the excavator dataset thanks to Ho [40]), we can assess the ability of each network to capture the human annotator’s internal/cognitive tag relationship structure.

Which network assigns tags to subsystems most like a domain expert?

9.1.3 Results

To test the ability of the similarity measures to accomplish this, the top three most common subsystems in the data were used as classes, namely, Hydraulic System, Engine, and Bucket. The tags “hydraulic”, “engine”, and “bucket” were assigned to those subsystems as known labels, respectively. Tags were filtered to only include ones of high-importance and sufficient information: only work orders containing at least 3 unique tags, and only tags that occurred at least 10 unique times within the those work orders, were included for this analysis ($C = 263$ MWOs, $N = 40$ tags).

Then the number of occurrences for every tag can be compared across subsystems, giving each tag a ground-truth multinomial (categorical) probability distribution for occurring within each subsystem. We determine ground-truth classification labels as subsystems that account for $\geq 60\%$ of each tag’s occurrences. Tags more balanced than that are considered “unknown subsystem”.

To perform semi-supervised classification on the recovered relationship graphs, we use a label-spreading algorithm described in [71], which itself was inspired by spreading activation networks in experimental psychology [44, @Observationphase-transitions_Shrager1987]. The result of this algorithm is tags having a score for each class, with the classification being the maximally scored class for that tag. These class assignments can then be compared to the ground-truth labels, which we have done by weighted macro-averaging of the F_1 -score (see Figure 9.2a).

The classification of the INVITE-based similarity measure far outperforms the other measures as a preprocessor for label-spreading, when measured by average F_1 -score. However, since these “classifications” are actually thresholded multinomial distributions (with some tags regularly occurring across multiple subsystems), how do we know if an underlying structure has actually been recovered, rather than simply a black-box classifier that happens to perform well at this setting?

To begin answering this question, we might ask whether the relative scores returned by label-spreading are similar to the original multinomial distributions themselves, rather than the overall classification. To find out, we use softmax normalization⁵ to transform each tag’s scores into a “predicted multinomial”, before finally

⁵For visualization, a temperature parameter was added to softmax, and this was optimized for

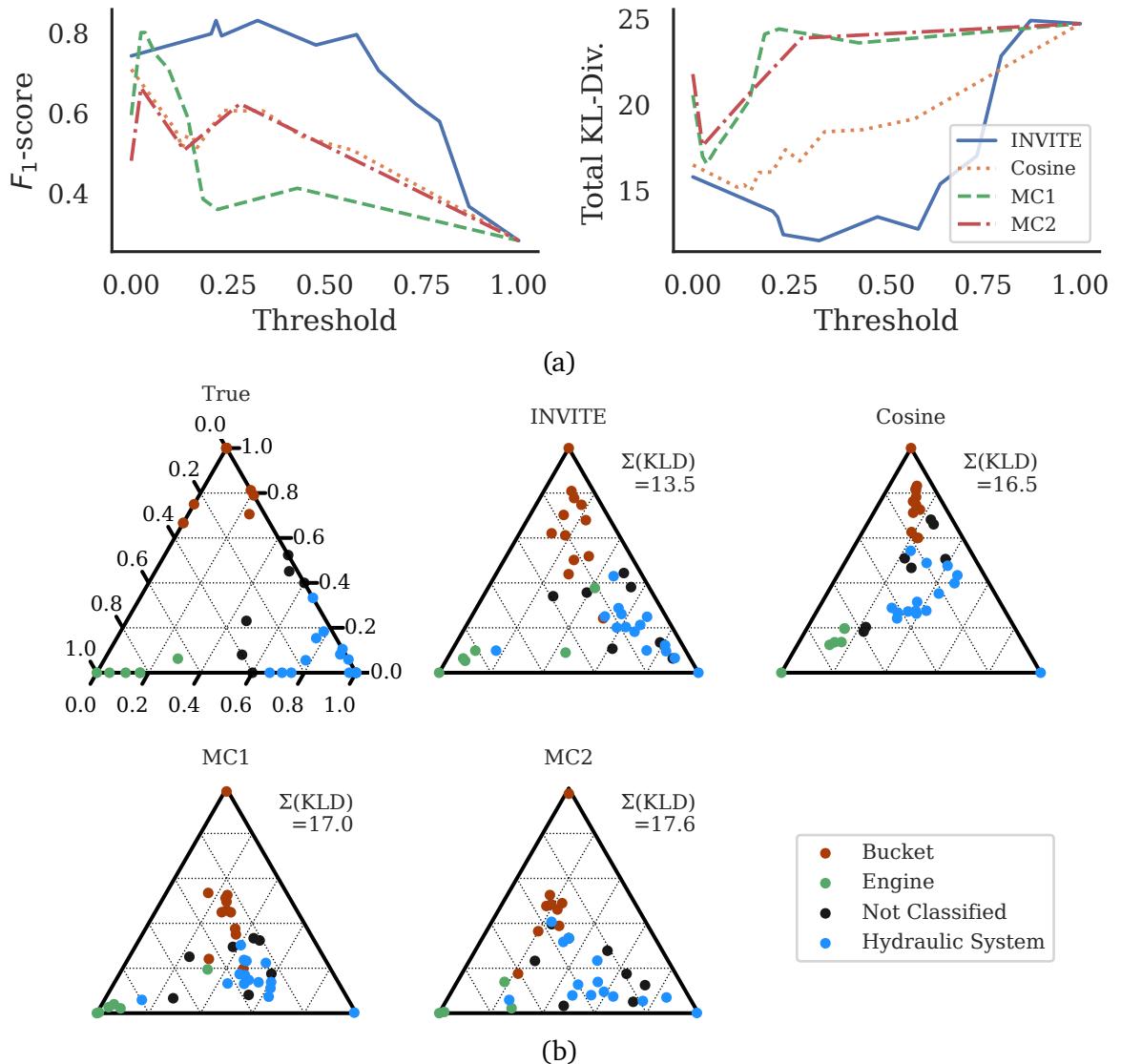


Figure 9.2: Semisupervised MWO Tag Classification with Network Label Propagation

calculating the Kullback-Leibler divergence (KLD) between the true and predicted multinomials for every tag. The total KLD, summed over all tags, is also shown in Figure 9.2b, along with positions of each tag’s multinomial as projected onto the 2-simplex for the true and F_1 -optimal predicted distributions. Once again, the INVITE performs much better at this task, over a wide range of σ (lower is better).

A reason for the performance disparity can be seen in the simplex projections: recovered topology via INVITE-similarity does a much better job of separating the three classes, while not letting any single tag overcompensate by dominating a subsystem’s area. Even the “unknown” tags are correctly placed roughly between Bucket and Hydraulic System regions, reflecting the true topology of the system.

9.2 Forest Pursuit Animal Network

For our last case study, we return to Forest Pursuit as a useful tool for analysis even when that might mean ignoring partial ordering information. Note that Equation 9.2 is effectively the same as a ($\beta = 1$) forest matrix if the transition (normalized adjacency) matrix was replaced by a (sub-)graph laplacian. Intuitively, the INVITE loss function is summing up over absorption (log-)probabilities at each new node activation: i.e. each step of a tree’s creation. Because the probability of a sampled tree is precisely proportional to the product of its edge weights, then weighing a tree by its absorption probabilities and running INVITE should have a mathematically similar effect as the FP estimate. The similarity would be exact whenever the tree that set of minimum KLD via Brent’s method [88] for each similarity measure independently to provide an equal footing for comparison.

nodes traveled along *was* the mode of the tree distribution: the MST.

In other words, whenever the random walks *did* use the minimum distance to reach each node, the two methods should be equivalent. While this isn't happening much individually, the effect of *many* random walks will average out to this MST, precisely because *it is the tree distribution mode* from which we assume node activations sample under RSFm.

To illustrate FP's efficacy in network estimation despite ignoring partial order information, we turn to a classic network recovery problem in this space: *semantic networks from fluency data*[[31](#), [67](#)].

9.2.1 Domain-aware preprocessing

Verbal (semantic) fluency tests involve asking participants to list as many items belonging to a prompted category in their available time. For instance, an “animals” prompt could lead to an answer like “dog, cat, lion, tiger, bear, wolf...”, etc. Like before, the general idea is that recall of each item derives from a random walk in a cognitive “memory space”, with emissions (usually) only when a new animal is encountered. A participant might have backtracked internally to *dog* from *bear* and jumped to *wolf*, for example.

Using a high-dimensional multilabel embedding space is possible, with one-column-per-animal, but the lists tend to be quite long and give networks with “hairball” tendencies. However, if our intention is to find *dependencies* between animals for a large number of participants, we might reasonably limit the co-occurrence to only a nearby subset of each list. Effectively, we can limit the set of possible co-occurring

animals based on our domain knowledge, namely, the limits of working memory. Conservatively, with a well-cited work claiming 7 ± 2 semantic units at a time in working memory[91], we can limit the set of possible dependencies for a given item to the 5 items on either side. The 5 previous might have originated a jump to the current term, and the 5 after might be subsequent targets, for a *rolling window* of 10 terms at-a-time.⁶

For our case study, we use lists of animals submitted by participants as described in [31, 56]. We limit the animals under consideration to those that occurred in more than 30 lists for a good sample size, as well as lists with at least two animals. This resulted in 100 animals over 293 fluency lists. However, we ignore this filtering when creating the 10-animal rolling windows, to avoid inclusion of unrelated animals into prematurely filtered windows. After re-applying the filter, 100 animals appear across 8020 working-memory windows. Figure 9.3 shows the effects of this preprocessing on marginal distributions.

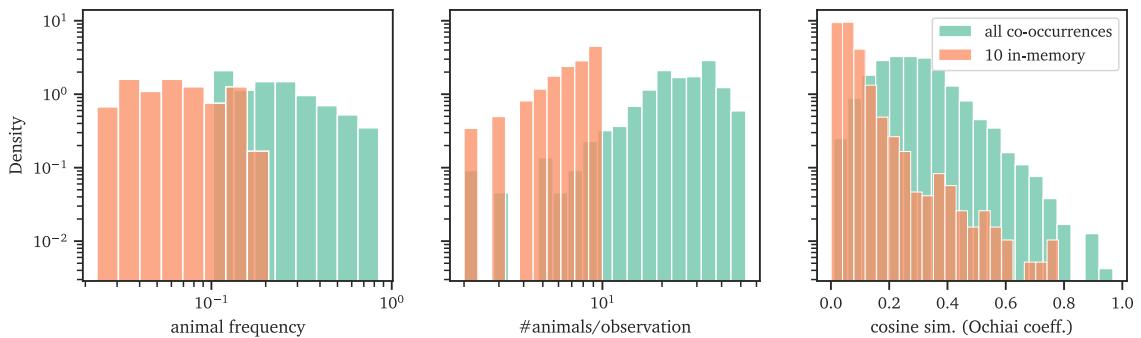


Figure 9.3: Effects of rolling-window activations on observation data

Doing this preprocessing (for rolling-windows of 10) shifts relative animal fre-

⁶This is precisely the logic that leads to the use of Skip-Gram and Continuous-Bag-of-Words transformations of text to weakly supervise word2vec and GloVe models[43, 48].

quencies downward (since there are many more “observations” from the rolling window), while also shifting the number of animals-per-observation to be strictly less than 10. As desired, the pairwise cosine similarity of all vectors $\mathbf{x}'_j, \mathbf{x}'_j$ is significantly reduced. While many participants might cover similar animals *overall*, we want to investigate animal dependencies locally, and we don’t expect individuals to always recall animals in the same memory “area” the whole time.

9.2.2 Edge Connective Efficiency and Diversity

To compare the results of different backboning techniques, we introduce a new simple measure to quantify a network’s sparsity, in terms of how many edges more than $n - 1$ (the minimum number needed to connect all n nodes) are being used.

$$\begin{aligned}\psi_n(e) &= \frac{e_{\min}}{e} \frac{e_{\max} - e}{e_{\max} - e_{\min}} \\ e_{\max} &= \frac{n(n-1)}{2} \quad e_{\min} = n-1\end{aligned}\tag{9.3}$$

We call ψ the graph’s “connective efficiency”, and it will range from 0 when the graph is fully connected, to 1 when it is a tree, to >1 when it has insufficient edges to be connected. This measure is intended to compare graphs that have had edges removed until it is about to be disconnected, such as with the *Doubly Stochastic Filter* (DS) [59]. However, values greater than 1 also give insight to how much sparser than a tree some graph is.

The DS animal network is shown in Figure 9.4. With large, deeply connected clusters centered around contexts animals are found in Figure 9.4 looks very similar

to the network recovered to make Fig. 4 in [56]. Clusters approximate animal types by their location relative to humans: farm/livestock, ocean/water creatures, “African” and jungle animals, small indoor mammals, small outdoor mammals, etc.

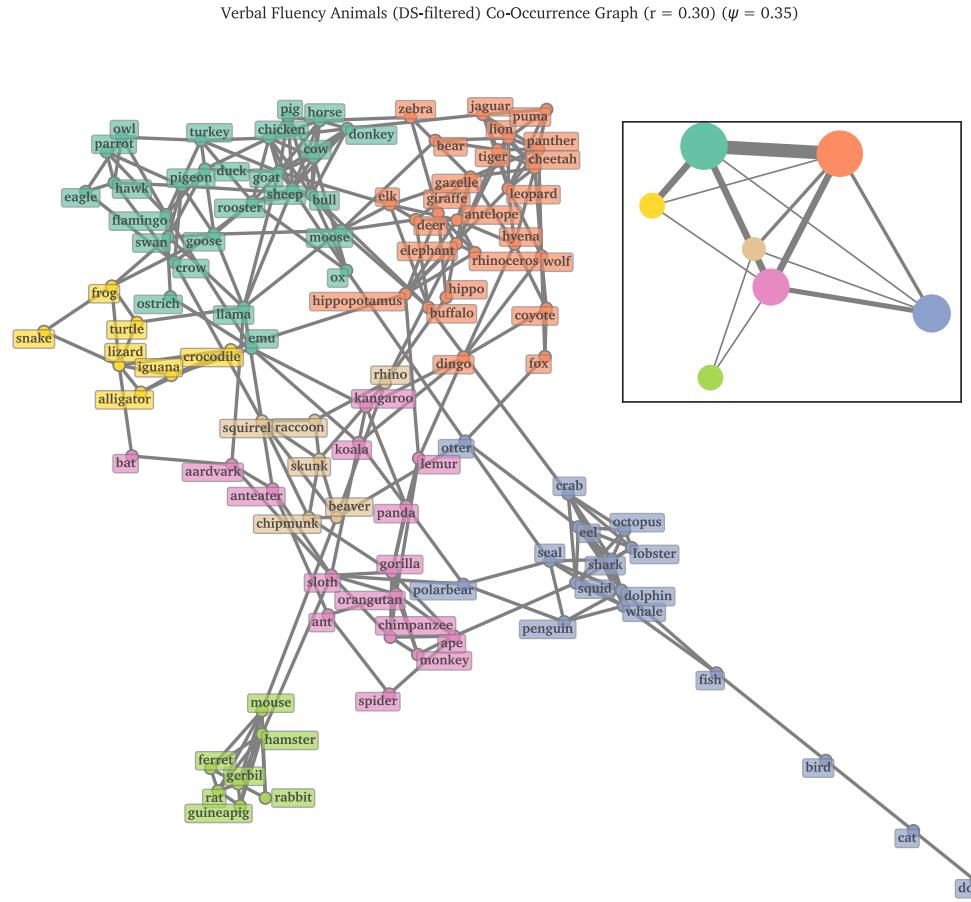


Figure 9.4: Verbal Fluency (animals) Network Backbone (Doubly-Stochastic)

This is largely how the literature on semantic fluency leaves their network recovery solution, with clusters based on human proximity or physical location. However, there are other ways people might relate animals than simply by location. Additionally, clique bias is quite strong in this network: why must every farm animal be mutually connected if it's possible to recall any of them through one or two “hub” animals? This

is related to the incredible inefficiency of this backbone, with a $\psi = 0.35$ being rather closer to fully connected than sparse. Additionally, the two animals that seem to appear *regardless* of others are *cat* and *dog*, which ironically makes DS penalize their proximity to any of the clusters. Both are ironically clustered with ocean animals due to their tendency to be listed near *fish*.

The Chow-Liu tree network is shown in Figure 9.5, and goes some way to alleviating these issues. Clusters are largely intact, instead represented by large branches/subtrees off the main group. However, some community relationships have been sacrificed to maintain strong individual edges, such as *monkey+giraffe* for location similarity at the expense of separating two halves of the pink cluster across a wide distance. More alternate paths between creatures (i.e. loops) are needed to better represent our perception of animal relationships.

The other dependency network recovery method is GLASSO, which we have similarly thresholded at the minimum-connected point. It only slightly improves on connective efficiency ($\psi = 0.44$), though the cliques are replaced with much more dispersed connections throughout the graph. We also see that reasonable inter-group connections are better represented, such as *rabbit+squirrel*, though *cat* and *dog* are still isolated due to overrepresentation throughout the dataset.

Subjectively, the GLASSO network is still difficult for an analyst to synthesize into useful knowledge, with so many edges, while the DS network only really managed to communicate one “kind” of knowledge (the context clusters). We would ideally prefer a backbone that provides a wider diversity of important edge “types”, for an analyst to better understand the kinds of animal relationships humans perceive.

Verbal Fluency Animal Dependencies (Chow-Liu) Network ($r = -0.14$) ($\psi = 1.00$)

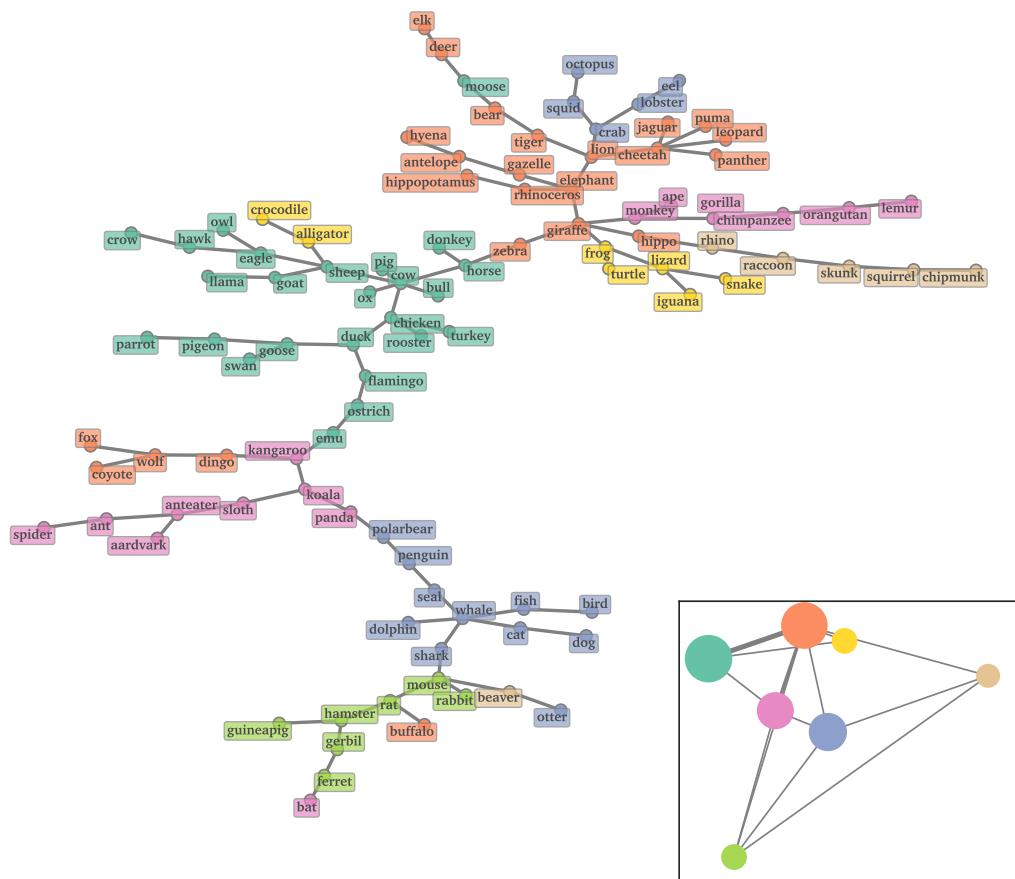


Figure 9.5: Verbal Fluency (animals) Dependency Network (Chow-Liu Tree)

Verbal Fluency Animal Dependencies (GLASSO) Network ($r = -0.01$) ($\psi = 0.44$)

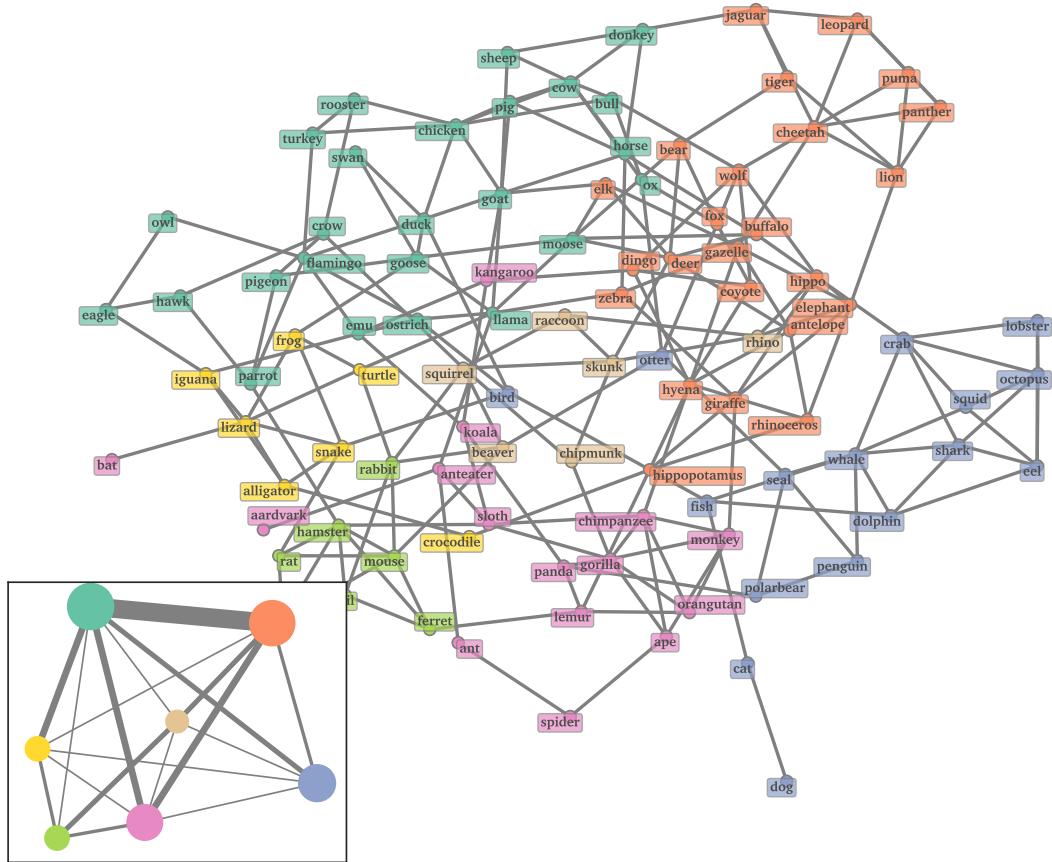


Figure 9.6: Verbal Fluency (animals) Dependency Network (GLASSO)

To illustrate this, we show the *Forest Pursuit*(FP) network in Figure 9.7. It has also been filtered to minimum-connected, like DS and GLASSO, though in this case the connective efficiency to reach that threshold is a staggering $\psi = 0.88$.

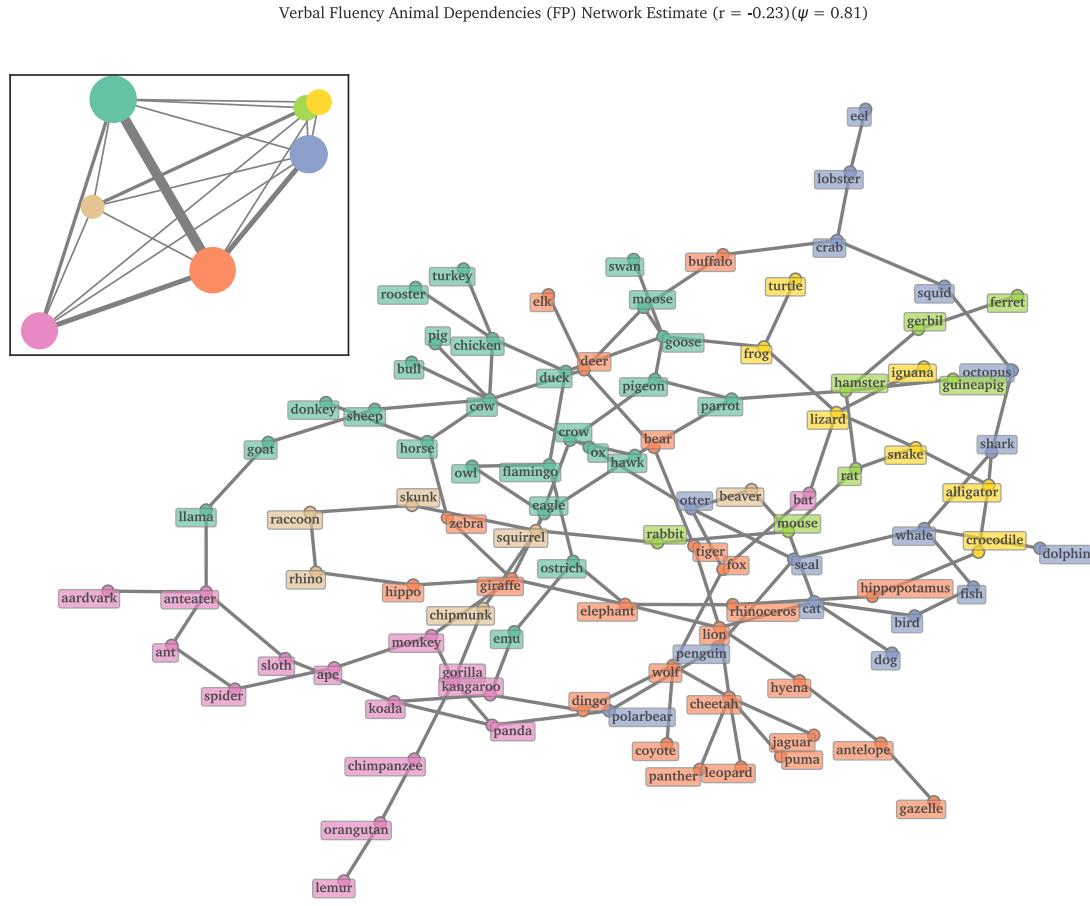


Figure 9.7: Verbal Fluency (animals) Dependency Network (Forest Pursuit)

Unlike the other networks that push “generalist” nodes like *cat* and *dog* onto long, distant chains, those chains are used in the FP network to hold rare subgroups of clusters, treating them as “gated” by the prominent “hubs” of those groups. For example, *cat* is correctly linked to *mouse* and *lion* (in addition to *bird*), while *lemur* is pushed down a longer chain of primates, “gated” by *gorilla*. Similarly with *eel*

through *lobster* and *crab*.

A much broader edge-type diversity is also made apparent with many non-context-based relationships made obvious with the improved sparsity. An analyst has an easier job of creating “edge-type inventories”, making the FP backbone an excellent exploratory assistant: animals can be related because they are:

- Co-located
- Taxonomically similar (*cheetah+leopard*)
- Famous predator/prey pairs (*cat+mouse*)
- Pop-culture references (*lion→tiger→bear*)⁷
- Similar in ecological niche/role (*koala+sloth*)
- Related through conservation or public awareness (*panda+gorilla*)
- etc.

This is further reflected in how FP alters the way *centrality* measures behave. Replicating Figure 8.7 for these graphs, Figure 9.8 shows the change in rank across the top 15 animals for the DS, GLASSO, and FP networks.

The DS centrality finds the most densely connected clique and gives all of its members incredibly high values. Meanwhile, the none of the top 5 most common animals (*dog,cat,lion,tiger,bear*)⁸ have high centrality *at all*. Both GLASSO and DS have farm animals (*chicken,goat,cow*) as the most important, despite the idea that

⁷Note that the dependency-based methods correctly interpreted these three as *not* being mutually connected in a triad, but specifically with this ordering (*tiger* in the middle).

⁸Interestingly, *dog* never appears in centrality measures, and *none* of the networks connect *dog* to any other animal than *cat*. Meanwhile, *wolf* is associated more with *fox, coyote, dingo*, etc., which are notably all predators of farm animals.

With the primary community structures being what they are (context/location-based), it seems that humans tend to put dogs in a category all their own.

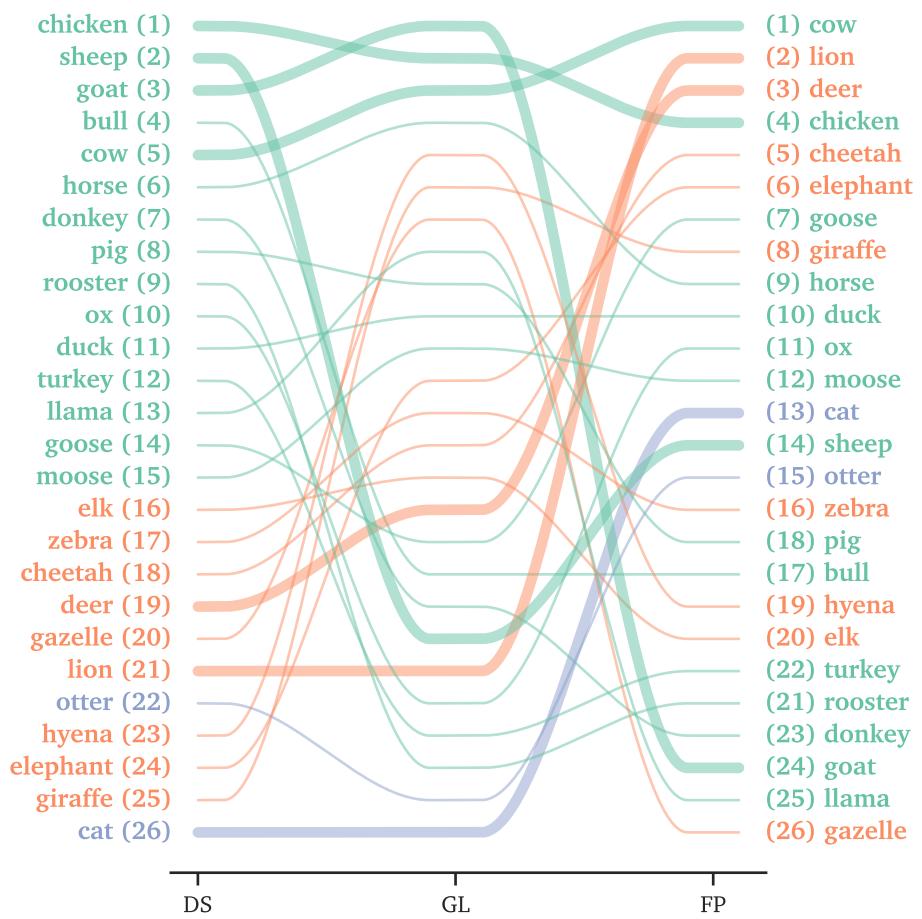


Figure 9.8: Changes in animal centrality ranking for FP vs co-occurrence, GLASSO

goat likely could be reached from e.g. *cow* quite often. FP adds more variety, giving hub-animals from different communities high centrality scores, each of which could lead to a variety of different paths. While subjective, the ranks from FP appear to be a more holistic inventory of “lynchpin” animals, which provide nearby coverage for a large amount of others.

9.2.3 Thresholded Structure Preservation

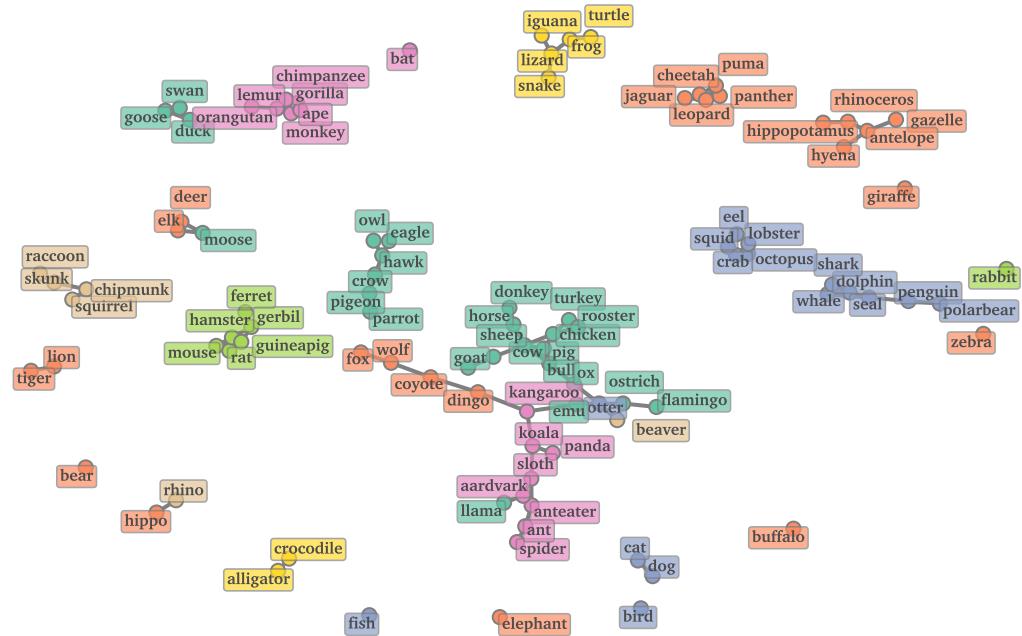
Another beneficial feature of *Forest Pursuit* is how it fares under increased thresholding. Because co-occurrence methods prioritize cliques, those cliques will remain connected as other edges are removed, effectively destroying the global structure of the animal network past a certain threshold. As seen in Figure 9.9, in keeping only the top 2% of edges, they are used to connect separated islands of animal communities.

Meanwhile, the FP network is quite robust to this excessive thresholding, with the global structure preserved at 2%. The removed edges have simply detached some of the rarer animals from the network entirely. These isolates not only reflect a more metrologically-sound idea (that rarer nodes would be disconnected at higher certainty thresholds), but are also beneficial to analysts, since manually re-connecting isolates of rare animals is simpler than manually determining a global reconnection strategy for island groups.

9.2.4 Forest Pursuit as Preprocessing

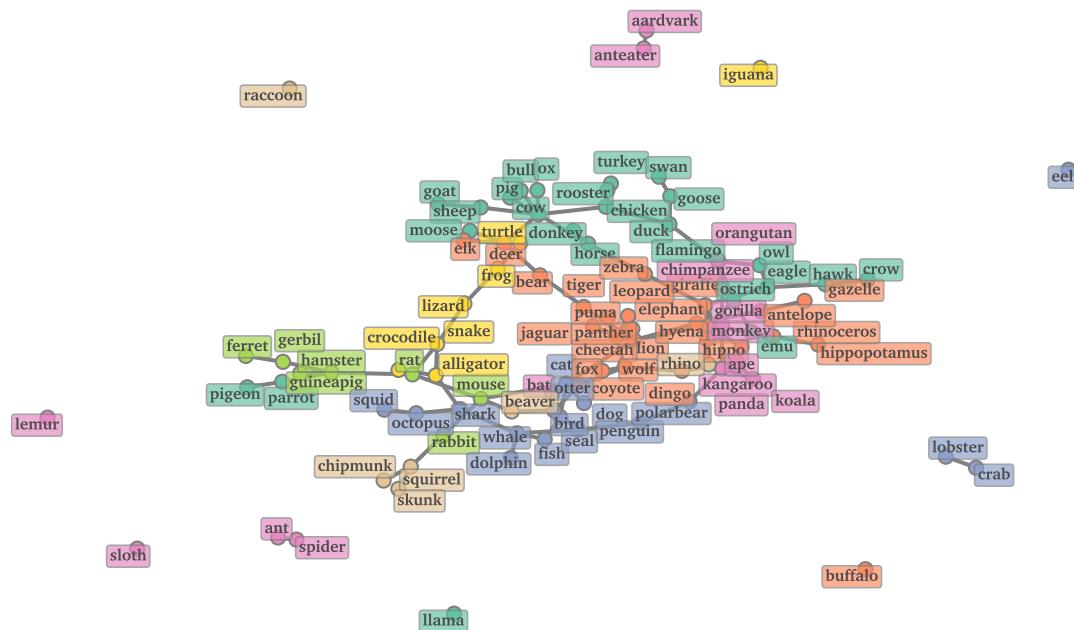
Because *Forest Pursuit* creates a representation of the observed data in *edge space*, we can use Equation 6.4 in the forward direction, creating a “new”design matrix

Verbal Fluency Animals Co-Occurrence (DS 98%) Network ($r = 0.41$) ($\psi = 1.00$)



(a) co-occurrence retains local communities at the cost of global structure

Verbal Fluency Animal Dependencies (FP 98%) Network ($r = -0.11$) ($\psi = 1.01$)



(b) clique-bias correction preserves central structure by disconnecting rare nodes

Figure 9.9: Differences in structural preservation with over-thresholding.

$X \leftarrow BR$. As discussed in Section 6.2.1, Equation 6.4 will create a design matrix of interaction counts for each node (its degree in the steiner tree approximation), rather than a binary “on/off” indicator.

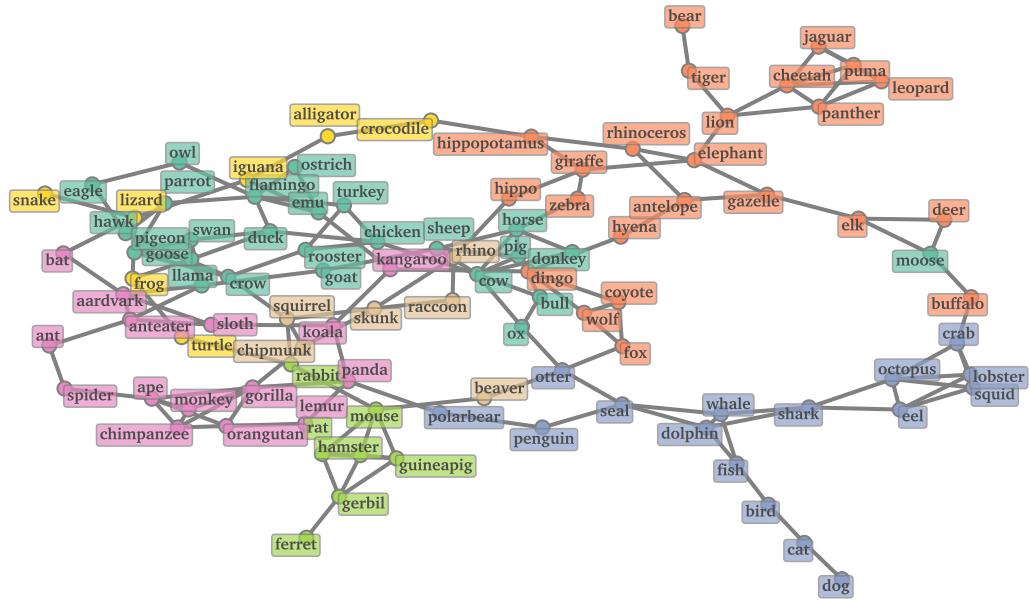
By supplying *other algorithms* with this new estimate for X , this makes FP a kind of preprocessing on the data itself. We can do this to bias the other methods toward greater sparsity in the backbone, without explicitly relying on point estimates for each observation (any tree with those node degrees would do the same). As shown in Figure 9.10, GLASSO and DS both increase their connective efficiency under FP preprocessing.

This is likely due to the increase in “signal-to-noise” ratio for each datapoint, since observations are only similar when they have a node interacting the same “amount”, not merely when similar nodes are activated.

Because the entries are now integer counts, FP preprocessing might also give a better path to using distribution-based embedding and clustering techniques, such as Hellinger distances between multinomial sample counts. This goes for many other techniques from text processing that rely on multinomial assumptions (i.e. techniques otherwise inapplicable to binary data).

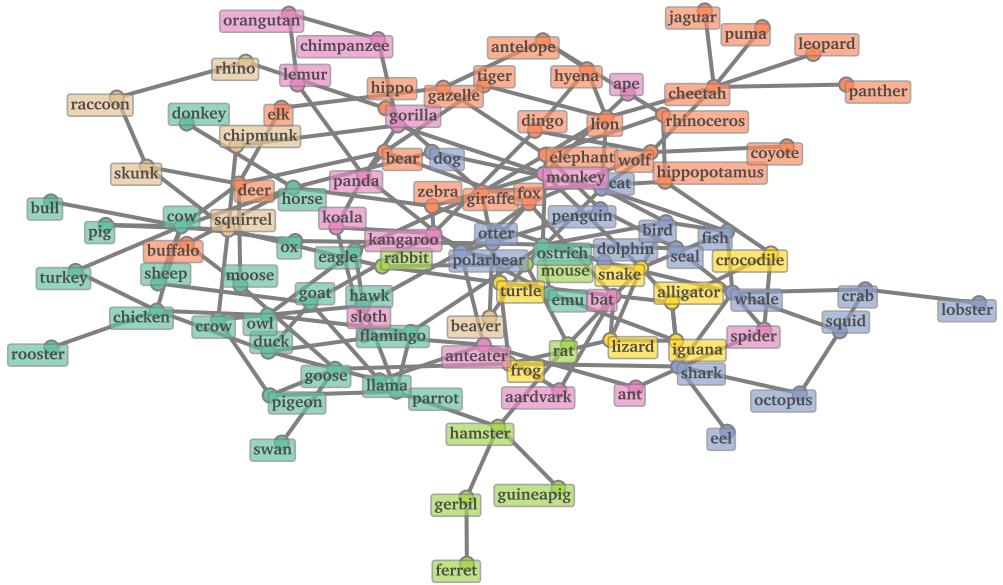
FP Preprocessing, with further empirical and theoretical validation, might prove to be a powerful tool for practitioners to flexibly backbone and analyse their networks with a variety of new techniques.

Animal Dependencies (FP → DS) Network ($r = 0.06$) ($\psi = 0.60$)



(a) Doubly Stochastic filter with FP (node degree) preprocessing

Animal Dependencies (FP → GLASSO) Network ($r = -0.19$) ($\psi = 0.60$)



(b) GLASSO Precision estimate with FP (node-degree) preprocessing

Figure 9.10: Forest Pursuit preprocessing for Doubly-Stochastic and GLASSO recovered networks

Chapter 10: Conclusion & Future Work

We showed...

10.1 Modifications and Extensions to Forest Pursuit

10.1.1 Multi-source Spreading

include the root, independent bernoulli injections.

10.1.2 Hidden Node Discovery

Adding steiner nodes to improve hyperbolicity of metric [20]

10.1.3 MCMC and Gibbs Sampling

10.2 Generalizing Inner Products on Incidences

10.2.1 special orthogonal group

the “root” node affects a time-like dimension

10.2.2 \mathcal{H}_2 embedding and beltrami voronoi

hyperbolic version of [5]

10.3 Potential Applications and Case Studies

10.3.1 Topology-Dependent Graph Application

- reinventory of assortativity estimates for spreading process graphs
- re-examine semantic network diagnoses for Alzheimers/Schizophrenia

10.3.2 Human-in-the-loop for Exploratory Relational Analysis

Update [66], especially for doing grounded coding [13].

Bibliography

- [1] O. Stuhler, “The gender agency gap in fiction writing (1850 to 2010),” *Proceedings of the National Academy of Sciences*, vol. 121, no. 29, Jul. 2024, ISSN: 1091-6490. doi: [10.1073/pnas.2319514121](https://doi.org/10.1073/pnas.2319514121).
- [2] D. Chicco and G. Jurman, “A statistical comparison between matthews correlation coefficient (MCC), prevalence threshold, and fowlkes–mallows index,” *Journal of Biomedical Informatics*, vol. 144, p. 104426, Aug. 1, 2023, ZSCC: 0000018, ISSN: 1532-0464. doi: [10.1016/j.jbi.2023.104426](https://doi.org/10.1016/j.jbi.2023.104426). Accessed: Sep. 13, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1532046423001478>.
- [3] J. Crall, “The mcc approaches the geometric mean of precision and recall as true negatives approach infinity,” Apr. 30, 2023. doi: [10.48550/ARXIV.2305.00594](https://doi.org/10.48550/ARXIV.2305.00594). arXiv: [2305.00594 \[cs.CV\]](https://arxiv.org/abs/2305.00594).
- [4] B. Landa and X. Cheng, “Robust inference of manifold density and geometry by doubly stochastic scaling,” *SIAM Journal on Mathematics of Data Science*, vol. 5, no. 3, pp. 589–614, Sep. 30, 2023, ZSCC: 0000002. doi: [10.1137/22M1516968](https://doi.org/10.1137/22M1516968). Accessed: Oct. 12, 2023.
- [5] R. T. Q. Chen, B. Amos, and M. Nickel, “Semi-discrete normalizing flows through differentiable tessellation,” in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 14 878–14 889. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/5f61939af1699c82dab00ed36c887968-Paper-Conference.pdf.
- [6] L. Peel, T. P. Peixoto, and M. De Domenico, “Statistical inference links data and theory in network science,” *Nature Communications*, vol. 13, no. 1, Nov. 2022, ISSN: 2041-1723. doi: [10.1038/s41467-022-34267-9](https://doi.org/10.1038/s41467-022-34267-9). [Online]. Available: <https://www.nature.com/articles/s41467-022-34267-9>.
- [7] M. P. Brundage, R. Sexton, M. Hodkiewicz, A. Dima, and S. Lukens, “Technical language processing: Unlocking maintenance knowledge,” *Manufacturing Letters*, vol. 27, pp. 42–46, Jan. 2021, ISSN: 2213-8463. doi: [10.1016/j.mfglet.2020.11.001](https://doi.org/10.1016/j.mfglet.2020.11.001).
- [8] M. Coscia, *The atlas for the aspiring network scientist*. Michele Coscia, Jan. 2021. arXiv: [2101.00863](https://arxiv.org/abs/2101.00863).

- [9] L. L. Duan and D. B. Dunson, “Bayesian spanning tree: Estimating the backbone of the dependence graph,” arXiv, arXiv:2106.16120, Jun. 30, 2021, ZSCC: 0000001 type: article. doi: [10.48550/arXiv.2106.16120](https://doi.org/10.48550/arXiv.2106.16120). arXiv: [2106.16120](https://arxiv.org/abs/2106.16120).
- [10] S. Jukna and H. Seiwert, “Tropical kirchhoff’s formula and postoptimality in matroid optimization,” *Discrete Applied Mathematics*, vol. 289, pp. 12–21, Jan. 31, 2021, ZSCC: NoCitationData[s0], issn: 0166-218X. doi: [10.1016/j.dam.2020.09.018](https://doi.org/10.1016/j.dam.2020.09.018). Accessed: Feb. 15, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0166218X2030439X>.
- [11] Z. P. Neal, R. Domagalski, and B. Sagan, “Comparing alternatives to the fixed degree sequence model for extracting the backbone of bipartite projections,” *Scientific Reports*, vol. 11, no. 1, p. 23 929, Dec. 14, 2021, ZSCC: 0000023, issn: 2045-2322. doi: [10.1038/s41598-021-03238-3](https://doi.org/10.1038/s41598-021-03238-3). Accessed: May 7, 2024. [Online]. Available: <https://www.nature.com/articles/s41598-021-03238-3>.
- [12] A. Omanović, H. Kazan, P. Oblak, and T. Curk, “Sparse data embedding and prediction by tropical matrix factorization,” *BMC Bioinformatics*, vol. 22, no. 1, Feb. 2021, issn: 1471-2105. doi: [10.1186/s12859-021-04023-9](https://doi.org/10.1186/s12859-021-04023-9).
- [13] J. Saldaña, “The coding manual for qualitative researchers,” 2021.
- [14] L. Torres, A. S. Blevins, D. Bassett, and T. Eliassi-Rad, “The why, how, and when of representations for complex systems,” *SIAM Review*, vol. 63, no. 3, pp. 435–485, Jan. 2021, issn: 0036-1445. doi: [10.1137/20M1355896](https://doi.org/10.1137/20M1355896). Accessed: Feb. 1, 2023.
- [15] R. Zmigrod, T. Vieira, and R. Cotterell, “Efficient computation of expectations under spanning tree distributions,” *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 675–690, Jul. 8, 2021, ZSCC: 0000004, issn: 2307-387X. doi: [10.1162/tacl_a_00391](https://doi.org/10.1162/tacl_a_00391). Accessed: May 8, 2023.
- [16] M. Coscia, “Generalized euclidean measure to estimate network distances,” in *Proceedings of the international AAAI conference on web and social media*, vol. 14, Association for the Advancement of Artificial Intelligence (AAAI), 2020, pp. 119–129. doi: [10.1609/icwsm.v14i1.7284](https://doi.org/10.1609/icwsm.v14i1.7284).
- [17] C. Malzer and M. Baum, “A hybrid approach to hierarchical density-based cluster selection,” in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, IEEE, Sep. 2020, pp. 223–228. doi: [10.1109/mfi49285.2020.9235263](https://doi.org/10.1109/mfi49285.2020.9235263).
- [18] M. B. Paulus, D. Choi, D. Tarlow, A. Krause, and C. J. Maddison, *Gradient estimation with stochastic softmax tricks*, 2020. doi: [10.48550/ARXIV.2006.08063](https://doi.org/10.48550/ARXIV.2006.08063).
- [19] R. Sexton and M. Fuge, “Organizing tagged knowledge: Similarity measures and semantic fluency in structure mining,” *Journal of Mechanical Design*, vol. 142, no. 3, Jan. 2020, issn: 1050-0472. doi: [10.1115/1.4045686](https://doi.org/10.1115/1.4045686).

- [20] R. Sonthalia and A. C. Gilbert, “Tree! i am no tree! i am a low dimensional hyperbolic embedding,” arXiv, arXiv:2005.03847, Oct. 22, 2020, type: article. doi: [10.48550/arXiv.2005.03847](https://doi.org/10.48550/arXiv.2005.03847). arXiv: [2005.03847](https://arxiv.org/abs/2005.03847).
- [21] M. Angeletti, J.-M. Bonny, and J. Koko, “Parallel Euclidean distance matrix computation on big datasets,” working paper or preprint, 2019. [Online]. Available: <https://hal.science/hal-02047514>.
- [22] K. Avrachenkov, P. Chebotarev, and D. Rubanov, “Similarities on graphs: Kernels versus proximity measures,” *European Journal of Combinatorics*, Special Issue in Memory of Michel Marie Deza, vol. 80, pp. 47–56, Aug. 1, 2019, ZSCC: 0000018, issn: 0195-6698. doi: [10.1016/j.ejc.2018.02.002](https://doi.org/10.1016/j.ejc.2018.02.002). Accessed: May 8, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0195669818300155>.
- [23] T. P. Peixoto, “Network reconstruction and community detection from dynamics,” *Physical Review Letters*, vol. 123, no. 12, p. 128 301, Sep. 18, 2019, ZSCC: 0000115. doi: [10.1103/PhysRevLett.123.128301](https://doi.org/10.1103/PhysRevLett.123.128301). Accessed: Nov. 2, 2023.
- [24] R. Sexton and M. Fuge, “Using semantic fluency models improves network reconstruction accuracy of tacit engineering knowledge,” in *Volume 2A: 45th Design Automation Conference*, American Society of Mechanical Engineers, Aug. 2019. doi: [10.1115/detc2019-98429](https://doi.org/10.1115/detc2019-98429).
- [25] R. Sexton, M. Hodkiewicz, and M. P. Brundage, “Categorization errors for data entry in maintenance work-orders,” *Annual Conference of the PHM Society*, vol. 11, no. 1, Sep. 2019, issn: 2325-0178. doi: [10.36001/phmconf.2019.v11i1.790](https://doi.org/10.36001/phmconf.2019.v11i1.790).
- [26] R. B. Sexton and M. B. Brundage, “Nestor: A tool for natural language annotation of short texts,” *Journal of Research of the National Institute of Standards and Technology*, vol. 124, Nov. 2019, issn: 2165-7254. doi: [10.6028/jres.124.029](https://doi.org/10.6028/jres.124.029).
- [27] E. Ackelsberg, “What is the arcsine law?,” 2018. [Online]. Available: https://math.osu.edu/sites/math.osu.edu/files/What_is_2018_Arcsine_Law.pdf.
- [28] T. P. Peixoto, “Reconstructing networks with unknown and heterogeneous errors,” *Physical Review X*, vol. 8, no. 4, p. 041 011, Oct. 16, 2018, ZSCC: 0000099. doi: [10.1103/PhysRevX.8.041011](https://doi.org/10.1103/PhysRevX.8.041011). Accessed: Nov. 2, 2023.
- [29] J. Schreiber, “Pomegranate: Fast and flexible probabilistic modeling in python,” *Journal of Machine Learning Research*, vol. 18, no. 164, pp. 1–6, 2018.
- [30] R. Sexton, M. Hodkiewicz, M. P. Brundage, and T. Smoker, “Benchmarking for keyword extraction methodologies in maintenance work orders,” *Annual Conference of the PHM Society*, vol. 10, no. 1, Sep. 2018, issn: 2325-0178. doi: [10.36001/phmconf.2018.v10i1.541](https://doi.org/10.36001/phmconf.2018.v10i1.541).
- [31] J. C. Zemla and J. L. Austerweil, “Estimating semantic networks of groups and individuals from fluency data,” *Computational brain & behavior*, vol. 1, pp. 36–58, 2018. doi: [10.31234/osf.io/kg45r](https://doi.org/10.31234/osf.io/kg45r).

- [32] K. Avrachenkov, P. Chebotarev, and A. Mishenin, “Semi-supervised learning with regularized laplacian,” *Optimization Methods and Software*, vol. 32, no. 2, pp. 222–236, Mar. 4, 2017, ZSCC: 0000013, ISSN: 1055-6788. doi: [10.1080/10556788.2016.1193176](https://doi.org/10.1080/10556788.2016.1193176). Accessed: May 8, 2023.
- [33] D. N. Fisher, M. J. Silk, and D. W. Franks, “The perceived assortativity of social networks: Methodological problems and solutions,” in *Trends in Social Network Analysis*. Springer International Publishing, 2017, pp. 1–19, ISBN: 9783319534206. doi: [10.1007/978-3-319-53420-6_1](https://doi.org/10.1007/978-3-319-53420-6_1).
- [34] M. R. Hodkiewicz, Z. Batsioudis, T. Radomiljac, and M. T. Ho, “Why autonomous assets are good for reliability—the impact of ‘operator-related component’ failures on heavy mobile equipment reliability,” in *Annual Conference of the Prognostics and Health Management Society 2017*, 2017.
- [35] J. Pang and G. Cheung, “Graph laplacian regularization for image denoising: Analysis in the continuous domain,” *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770–1785, Apr. 2017, ZSCC: 0000231, ISSN: 1941-0042. doi: [10.1109/TIP.2017.2651400](https://doi.org/10.1109/TIP.2017.2651400).
- [36] A. Paszke et al., “Automatic differentiation in pytorch,” in *NIPS-W*, 2017.
- [37] M. Hodkiewicz and M. T.-W. Ho, “Cleaning historical maintenance work order data for reliability analysis,” *Journal of Quality in Maintenance Engineering*, vol. 22, no. 2, pp. 146–163, 2016.
- [38] J. Kepner et al., “Mathematical foundations of the GraphBLAS,” in *2016 IEEE High Performance Extreme Computing Conference (HPEC)*, Sep. 2016, pp. 1–9. doi: [10.1109/HPEC.2016.7761646](https://doi.org/10.1109/HPEC.2016.7761646).
- [39] T. T. Hills, P. M. Todd, and M. N. Jones, “Foraging in semantic fields: How we search through memory,” *Topics in Cognitive Science*, vol. 7, no. 3, pp. 513–534, Jun. 2015, ISSN: 1756-8765. doi: [10.1111/tops.12151](https://doi.org/10.1111/tops.12151).
- [40] M. Ho, “A shared reliability database for mobile mining equipment,” Ph.D. dissertation, University of Western Australia, 2015.
- [41] K.-S. Jun, J. Zhu, T. T. Rogers, Z. Yang, et al., “Human memory search as initial-visit emitting random walk,” *Advances in neural information processing systems*, vol. 28, no. 20, pp. 2389–2393, 2015, ISSN: 0375-9601. doi: [10.1016/j.physleta.2019.04.060](https://doi.org/10.1016/j.physleta.2019.04.060).
- [42] Z. Neal, “The backbone of bipartite projections: Inferring relationships from co-authorship, co-sponsorship, co-attendance and other co-behaviors,” *Social Networks*, vol. 39, pp. 84–97, Oct. 1, 2014, ZSCC: 0000200, ISSN: 0378-8733. doi: [10.1016/j.socnet.2014.06.001](https://doi.org/10.1016/j.socnet.2014.06.001). Accessed: May 7, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378873314000343>.
- [43] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [44] J. R. Anderson, *The architecture of cognition*. Psychology Press, 2013.

- [45] M. Cuturi, “Sinkhorn distances: Lightspeed computation of optimal transport,” in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS’13, Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 2292–2300.
- [46] B. Dai, S. Ding, and G. Wahba, “Multivariate bernoulli distribution,” *Bernoulli*, vol. 19, no. 4, Sep. 2013, ISSN: 1350-7265. doi: [10.3150/12-bejsp10](https://doi.org/10.3150/12-bejsp10).
- [47] O. Knill, “Counting rooted forests in a network,” arXiv, arXiv:1307.3810, Jul. 18, 2013, ZSCC: 0000014 type: article. doi: [10.48550/arXiv.1307.3810](https://doi.org/10.48550/arXiv.1307.3810). arXiv: [1307.3810](https://arxiv.org/abs/1307.3810).
- [48] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.
- [49] D. Allard, A. Comunian, and P. Renard, “Probability aggregation methods in geoscience,” *Mathematical Geosciences*, vol. 44, no. 5, pp. 545–581, Jul. 1, 2012, ZSCC: 0000100, ISSN: 1874-8953. doi: [10.1007/s11004-012-9396-3](https://doi.org/10.1007/s11004-012-9396-3). Accessed: Aug. 8, 2023.
- [50] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, “Inferring networks of diffusion and influence,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 4, pp. 1–37, 2012, ISSN: 1556-472X. doi: [10.1145/2086737.2086741](https://doi.org/10.1145/2086737.2086741).
- [51] D. Grady, C. Thiemann, and D. Brockmann, “Robust classification of salient links in complex networks,” *Nature Communications*, vol. 3, no. 1, May 2012, ISSN: 2041-1723. doi: [10.1038/ncomms1847](https://doi.org/10.1038/ncomms1847).
- [52] T. T. Hills and T. Pachur, “Dynamic search and working memory in social recall.,” *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 38, no. 1, pp. 218–228, 2012, ISSN: 0278-7393. doi: [10.1037/a0025161](https://doi.org/10.1037/a0025161).
- [53] P.-l. Loh and M. J. Wainwright, “Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses,” in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012. doi: [10.1214/13-aos1162](https://doi.org/10.1214/13-aos1162). Accessed: Feb. 1, 2023. [Online]. Available: <https://proceedings.neurips.cc/paper/2012/hash/6ba1085b788407963fe0e89c699a7396-Abstract.html>.
- [54] N. Reff and L. J. Rusnak, “An oriented hypergraphic approach to algebraic graph theory,” *Linear Algebra and its Applications*, vol. 437, no. 9, pp. 2262–2270, Nov. 1, 2012, ISSN: 0024-3795. doi: [10.1016/j.laa.2012.06.011](https://doi.org/10.1016/j.laa.2012.06.011). Accessed: Feb. 1, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0024379512004466>.
- [55] D. J. Wang, X. Shi, D. A. McFarland, and J. Leskovec, “Measurement error in network data: A re-classification,” *Social Networks*, vol. 34, no. 4, pp. 396–409, Oct. 2012, ISSN: 0378-8733. doi: [10.1016/j.socnet.2012.01.003](https://doi.org/10.1016/j.socnet.2012.01.003).

- [56] J. Goñi et al., “The semantic organization of the animal category: Evidence from semantic verbal fluency and network theory,” *Cognitive processing*, vol. 12, no. 2, pp. 183–196, 2011.
- [57] J. Kepner and J. Gilbert, *Graph Algorithms in the Language of Linear Algebra*. Philadelphia: Society for Industrial and Applied Mathematics, Aug. 4, 2011, 375 pp., ISBN: 9780898719901.
- [58] N. Meinshausen and P. Bühlmann, “Stability selection,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 72, no. 4, pp. 417–473, Aug. 2010, ISSN: 1467-9868. DOI: [10.1111/j.1467-9868.2010.00740.x](https://doi.org/10.1111/j.1467-9868.2010.00740.x).
- [59] P. B. Slater, “A two-stage algorithm for extracting the multiscale backbone of complex weighted networks,” *Proceedings of the National Academy of Sciences*, vol. 106, no. 26, E66–E66, Jun. 30, 2009, ZSCC: 0000040. DOI: [10.1073/pnas.0904725106](https://doi.org/10.1073/pnas.0904725106). Accessed: May 7, 2024. [Online]. Available: <https://www.pnas.org/doi/abs/10.1073/pnas.0904725106>.
- [60] J. Friedman, T. Hastie, and R. Tibshirani, “Sparse inverse covariance estimation with the graphical lasso,” *Biostatistics (Oxford, England)*, vol. 9, no. 3, pp. 432–441, Jul. 2008, ZSCC: 0005870, ISSN: 1465-4644. DOI: [10.1093/biostatistics/kxm045](https://doi.org/10.1093/biostatistics/kxm045). Accessed: Feb. 15, 2023. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3019769/>.
- [61] R. Rubinstein, M. Zibulevsky, and M. Elad, “Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit,” *Cs Technion*, vol. 40, no. 8, pp. 1–15, 2008.
- [62] G. E. Moorhouse, “Incidence geometry,” *University of Wyoming*, pp. 3–5, 2007.
- [63] D. P. Wipf and B. D. Rao, “An empirical bayesian strategy for solving the simultaneous sparse approximation problem,” *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3704–3716, Jul. 2007, ISSN: 1053-587X. DOI: [10.1109/tsp.2007.894265](https://doi.org/10.1109/tsp.2007.894265).
- [64] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, “Bipartite network projection and personal recommendation,” *Phys. Rev. E*, vol. 76, no. 4, p. 046115, 4 Oct. 2007, ISSN: 1550-2376. DOI: [10.1103/PhysRevE.76.046115](https://doi.org/10.1103/PhysRevE.76.046115). [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevE.76.046115>.
- [65] P. Chebotarev and E. Shamis, “The matrix-forest theorem and measuring relations in small social groups,” arXiv, arXiv:math/0602070, Feb. 4, 2006, ZSCC: 0000285 type: article. DOI: [10.48550/arXiv.math/0602070](https://doi.org/10.48550/arXiv.math/0602070). arXiv: [math/0602070](https://arxiv.org/abs/math/0602070) [math].
- [66] P. Heymann and H. Garcia-Molina, “Collaborative creation of communal hierarchical taxonomies in social tagging systems,” Stanford InfoLab, Technical Report 2006-10, Apr. 2006. [Online]. Available: <http://ilpubs.stanford.edu:8090/775/>.

- [67] T. J. Prescott, L. D. Newton, N. U. Mir, P. W. R. Woodruff, and R. W. Parks, “A new dissimilarity measure for finding semantic structure in category fluency data with implications for understanding memory organization in schizophrenia.,” *Neuropsychology*, vol. 20, no. 6, pp. 685–699, Nov. 2006, ISSN: 0894-4105. doi: [10.1037/0894-4105.20.6.685](https://doi.org/10.1037/0894-4105.20.6.685).
- [68] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, p. 066 111, Dec. 2004, ISSN: 1550-2376. doi: [10.1103/physreve.70.066111](https://doi.org/10.1103/physreve.70.066111).
- [69] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review E*, vol. 69, no. 2, p. 026 113, Feb. 2004, ISSN: 1550-2376. doi: [10.1103/physreve.69.026113](https://doi.org/10.1103/physreve.69.026113).
- [70] “The tropical grassmannian,” *advg*, vol. 4, no. 3, pp. 389–411, Jul. 2004, ISSN: 1615-715X. doi: [10.1515/advg.2004.023](https://doi.org/10.1515/advg.2004.023).
- [71] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, “Learning with local and global consistency,” in *Advances in neural information processing systems*, 2004, pp. 321–328.
- [72] M. E. J. Newman, “The structure and function of complex networks,” *SIAM Review*, vol. 45, no. 2, pp. 167–256, Jan. 2003, ISSN: 1095-7200. doi: [10.1137/s003614450342480](https://doi.org/10.1137/s003614450342480).
- [73] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002, ISSN: 1091-6490. doi: [10.1073/pnas.122653799](https://doi.org/10.1073/pnas.122653799).
- [74] M. E. J. Newman, “Assortative mixing in networks,” *Physical Review Letters*, vol. 89, no. 20, p. 208 701, Oct. 2002, ISSN: 1079-7114. doi: [10.1103/physrevlett.89.208701](https://doi.org/10.1103/physrevlett.89.208701).
- [75] M. E. J. Newman, “Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality,” *Physical Review E*, vol. 64, no. 1, p. 016 132, Jun. 2001, ISSN: 1095-3787. doi: [10.1103/physreve.64.016132](https://doi.org/10.1103/physreve.64.016132).
- [76] P. G. Doyle and J. L. Snell, *Random walks and electric networks*, 2000. doi: [10.48550/ARXIV.MATH/0001057](https://arxiv.org/abs/math/0001057).
- [77] K. Engan, S. Aase, and J. Hakon Husoy, “Method of optimal directions for frame design,” in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, IEEE, 1999. doi: [10.1109/icassp.1999.760624](https://doi.org/10.1109/icassp.1999.760624).
- [78] H. Whitehead and S. Dufault, “Techniques for analyzing vertebrate social structure using identified individuals: Review and recommendations,” *Advances in the Study of Behavior*, vol. 28, no. 28, pp. 33–74, 1999.
- [79] K. Sneppen and M. Newman, “Coherent noise, scale invariance and intermittency in large systems,” *Physica D: Nonlinear Phenomena*, vol. 110, no. 3–4, pp. 209–222, Dec. 1997, ISSN: 0167-2789. doi: [10.1016/s0167-2789\(97\)00128-0](https://doi.org/10.1016/s0167-2789(97)00128-0).

- [80] D. B. Wilson, “Generating random spanning trees more quickly than the cover time,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC ’96*, ser. STOC ’96, ACM Press, 1996, pp. 296–303. doi: [10.1145/237814.237880](https://doi.org/10.1145/237814.237880).
- [81] B. K. Natarajan, “Sparse approximate solutions to linear systems,” *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, Apr. 1995, issn: 1095-7111. doi: [10.1137/s0097539792240406](https://doi.org/10.1137/s0097539792240406).
- [82] D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing*. AcM Press New York, 1993, vol. 1.
- [83] S. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993, issn: 1053-587X. doi: [10.1109/78.258082](https://doi.org/10.1109/78.258082).
- [84] S. Janson and J. Vegelius, “Measures of ecological association,” *Oecologia*, vol. 49, no. 3, pp. 371–376, Jul. 1981, issn: 1432-1939. doi: [10.1007/bf00347601](https://doi.org/10.1007/bf00347601).
- [85] L. Kou, G. Markowsky, and L. Berman, “A fast algorithm for steiner trees,” *Acta Informatica*, vol. 15, no. 2, pp. 141–145, 1981, issn: 1432-0525. doi: [10.1007/bf00288961](https://doi.org/10.1007/bf00288961).
- [86] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, Dec. 1977, issn: 2153-3806. doi: [10.1086/jar.33.4.3629752](https://doi.org/10.1086/jar.33.4.3629752).
- [87] D. Hunter, “An upper bound for the probability of a union,” *Journal of Applied Probability*, vol. 13, no. 3, pp. 597–603, Sep. 1976, issn: 1475-6072. doi: [10.2307/3212481](https://doi.org/10.2307/3212481).
- [88] R. P. Brent, “An algorithm with guaranteed convergence for finding a zero of a function,” *The Computer Journal*, vol. 14, no. 4, pp. 422–425, 1971.
- [89] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, May 1968, issn: 0018-9448. doi: [10.1109/tit.1968.1054142](https://doi.org/10.1109/tit.1968.1054142).
- [90] W. Feller, *An Introduction to Probability Theory and its Applications, Volume 1*. J. Wiley & Sons: New York, 1968.
- [91] G. A. Miller, “The magical number seven, plus or minus two: Some limits on our capacity for processing information.,” *Psychological Review*, vol. 63, no. 2, pp. 81–97, Mar. 1956, issn: 0033-295X. doi: [10.1037/h0043158](https://doi.org/10.1037/h0043158).
- [92] G. U. Yule, “On the methods of measuring association between two attributes,” *Journal of the Royal Statistical Society*, vol. 75, no. 6, p. 579, May 1912, issn: 0952-8385. doi: [10.2307/2340126](https://doi.org/10.2307/2340126).