Graph Inference with Applications to Low-Resource Audio Search and Indexing

by

Keith Levin

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

May, 2017

Abstract

The task of query-by-example search is to retrieve, from among a collection of data, the observations most similar to a given query. A common approach to this problem is based on viewing the data as vertices in a graph in which edge weights reflect similarities between observations. Errors arise in this graph-based framework both from errors in measuring these similarities and from approximations required for fast retrieval. In this thesis, we use tools from graph inference to analyze and control the sources of these errors. We establish novel theoretical results related to representation learning and to vertex nomination, and use these results to control the effects of model misspecification, noisy similarity measurement and approximation error on search accuracy. We present a state-of-the-art system for query-by-example audio search in the context of low-resource speech recognition, which also serves as an illustrative example and testbed for applying our theoretical results.

Primary Reader: Professor Ben Van Durme

Secondary Reader: Professor Vince Lyzinski

Tertiary Reader: Professor Carey E. Priebe

Acknowledgments

This thesis would not have been possible without the support of countless people. My thanks belong first and foremost to my co-advisors Vince Lyzinski, Ben Van Durme and Carey Priebe, whose guidance and insights have been invaluable. To my father Rick, my brother Craig, and the rest of my family, for your love and encouragement I offer my humble thanks. I owe a debt of gratitude to the faculty and staff of the Johns Hopkins University Center for Language and Speech Processing and the Departments of Computer Science and Applied Math and Statistics. In particular, my thanks to Ruth Scally, Debbie Deford, Cathy Thornton, Laura Graham and Zachary Burwell, without whom I would have been entirely lost, and to Glen Coppersmith, Aren Jansen, Vladimir Braverman, Avanti Athreya, Minh Tang, Donniell Fishkind, Sanjeev Khudanpur, Jason Eisner, Raman Arora and Amitabh Basu, who have all influenced this thesis for the better in discussions over the past few years. My deepest thanks to my fellow students and dear friends at JHU for lively discussion on matters both technical and less so, countless coffees and beers, soccer games and occasional live music. To the educators throughout my life, for setting me on this path, I offer

my sincere appreciation. My thanks are due in particular to Mary Cawley, George Farrell, Eric Michelson, Marty Badoian, Marina Todorova, Neal Pearlmutter, Joanne Miller, Karl Lieberherr, Emanuele Viola and Robert Chase. My deepest thanks to my former colleagues at BBN, in particular to Bernhard Suhm, Jeff Schachter, Pat Peterson and John Makhoul for their patience and advice. Finally, my warmest thanks to my friends Matthew Gray, Eric Woods, Bonnie Le, Dana Brocious, Paige Brocious, Lou Vitelli, Brian Correia, Becky Gerstein, Phil Small and Jenny Bagnyuk.

Dedication

This thesis is dedicated to the memory of my mother, Leslie Levin.

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Motivation

Researchers throughout the sciences are now generating data from both observation and simulation at increasingly large scales, aided by the ubiquity of inexpensive sensors and storage. Indeed, some posit that we have entered a new "fourth paradigm" (Hey et al. 2009) of scientific research, in which collection, curation and analysis of massive data sets are central to the advancement of our understanding of the natural world. Under this fourth paradigm, methods for analyzing, exploring and summarizing data sets are paramount. One such operation is query-by-example similarity search, in which a researcher, having found an observation of interest, called the *query*, wishes to find more like it from among a collection of observations called the *search collection*. Problems of this sort arise in machine learning in the form of recommender systems (Resnick and Varian 1997), in genomics in the form of sequence similarity search (Lipman and Pearson 1985; Altschul et al. 1990), and in computer vision in the form of content-based retrieval (Datta et al. 2008), among countless other applications. Owing to computational constraints and model misspecification, similarity search on large data sets can incur errors that negatively impact the quality of search results and downstream performance. This thesis aims to better control and mitigate the sources of these errors using tools from graph inference.

The typical approach to query-by-example similarity search consists of first embedding the search collection into some finite-dimensional normed linear space in such a way that the similarity structure of the data is preserved. Next, one constructs an index for performing (approximate) near neighbor retrieval. Given a query, this index allows the fast retrieval of near neighbors of the embedded query observation from the search collection. We call these near neighbors the *query results*. In applying this approach, it is typical that one must accept approximation error from a number of sources. Under most realistic circumstances, any given embedding technique will preserve the similarity structure of the search collection only approximately. Additionally, the size of the search collection may make it prohibitively expensive to construct this embedding exactly, and a researcher may have to settle instead for an approximate embedding. The approximate, probabilistic nature of large-scale near-neighbor search adds yet another source of error. Finally, the similarity measure itself may be a source of error, since it is often infeasible in practice to precisely compute the researcher's intended or desired notion of similarity. This makes it necessary to use some other, more readily computed similarity function that is only an approximation to the one actually intended.

In this thesis, we explore how these various approximation errors influence the performance of the search system and attempt to minimize their effects using techniques from graph inference. In particular, we focus on the effects of replacing the intended, ideal notion of similarity, which we call the *oracle* similarity function, with a more feasible approximation, which we call the *ersatz* similarity function. We motivate the choice to focus on this source of error by observing, firstly, that it tends to dominate the error introduced by embedding and near-neighbor retrieval, and secondly that selection of the oracle similarity (and its approximation) occurs prior to the construction of embeddings and the near-neighbor index. A better understanding of the effects of these choices is central to improving systems for large-scale search and for analyzing and summarizing large data sets generally.

1.1 Similarity Search and Indexing of Large Data Sets

The problem of query-by-example search is to find, from among a large number of observations, those that are most similar to a given query observation. This notion of similarity depends, of course, on the domain and application at hand. For example, in the case of astronomy data, a researcher looking to find stars with similar spectra will have in mind a different notion of similarity than does a researcher looking to find stars with similar redshifts (Morison 2008).

Having chosen a similarity function, how should we perform retrieval from the search collection when presented with a query? Naïvely, one could compute the similarity of the query to all observations in the search collection and return those that score highest. This brute force approach is, of course, infeasible for search collections of even moderate size. Near-neighbor retrieval, discussed in detail in Appendix C, suggests a way forward: if one can represent observations as points in such a way that similar observations have corresponding points that are near one another, then we can recast similarity search as near-neighbor retrieval and apply the existing tools of near-neighbor search to the similarity search problem. Of course, this only raises the new issue of representing the search collection by geometric points. Fortunately, this problem is itself well-studied in the areas of dimensionality reduction, manifold learning and metric embedding (see Appendix B for an overview of these areas).

The tools of embeddings and near-neighbor retrieval suggest the following commonly-used pipeline. Having chosen a notion of similarity, one first embeds the search collection into a metric space in which it is easy to perform near-neighbor retrieval. By construction, this embedding is such that observations are similar if and only if their embedded points are near one another. Having embedded the search collection, one then builds an index to facilitate near-neighbor retrieval on the embedded points. Upon arrival of a query observation, one simply embeds the query, finds the points nearest to it, and returns their corresponding observations.

1.1.1 Sources of Error

Unfortunately, approximation error is introduced at every step of this proposed pipeline. All scalable algorithms for near neighbor retrieval are only approximate, and typically involve an accuracy guarantee that holds only probabilistically. Further, most embedding techniques preserve the similarity structure of the data only approximately, and thus in the above pipeline, near-neighbor retrieval is actually retrieving only the observations that are *approximately* the most similar to the query. But both of these sources of error are, in some sense, secondary to the approximation error introduced, before either of these steps take place, by the similarity measure itself.

This approximation error arises from the fact that in most cases of interest, even writing down a sensible notion of similarity is a challenge. Consider an image retrieval task, in which we have a database of images of common objects, and the goal is to retrieve from the database all images that contain the same object as is pictured in a given query image. Here, the ideal notion of similarity is easy to state: two images are similar if and only if they contain the same object. Unfortunately, while most humans can readily identify whether or not, say, a cat is present in a given photo, the same task is a notoriously hard problem in computer vision (Krizhevsky et al. 2012; Szegedy et al. 2013). Thus, even though this ideal notion of similarity is an easy one to state, and even easy for most humans, it is an infeasible one for use in retrieval. In such a situation, a researcher must settle for a simpler notion of similarity that is more easily computed. For example, in the image retrieval task just described, a researcher may instead use a similarity scoring function based on low-level image features. Using an ersatz similarity function yields computational tractability at the cost of accuracy, in that the ersatz similarity does not fully capture the similarity that was originally intended. We illustrate the typical search pipeline as well as many of these computational and approximation concerns in Chapter 2, where we present a system for performing large-scale query-by-example search on speech audio.

A second source of difficulty arises from related but distinct computational concerns. In general, even having chosen an ersatz similarity, there remains the matter of actually computing the embedding of the search collection. In some cases, it may still be prohibitively expensive to compute even the ersatz function for all pairs of observations in the search collection. In such a case, a researcher may back off to computing an approximation of the ersatz function or computing the ersatz function for only a fraction of the pairs of observations in the search collection. We explore how such tradeoffs influence the quality of the embedded points in Chapter 3. Our main result of the chapter shows that a certain class of embeddings are largely unaffected by these various sources of approximation error, provided certain mild assumptions hold concerning the search collection and the nature of the approximation.

1.1.2 Reranking Candidate Matches

A technique commonly applied in large-scale search, called *reranking* (Mei et al. 2014), is to retrieve results in two passes. A first pass performs an inexpensive coarse-grained search on the entire collection, which returns a set of candidate matches far larger than the intended set of results to be returned to the user. A second-pass search, the reranking step, provides a more expensive, more accurate assessment of similarity. This second pass is applied only to the candidate matches, with the goal of refining the ranking of the matches returned by the first pass.

The retrieval system described in the previous sections makes fast large-scale similarity search possible at the cost of introducing approximation error at several steps in the pipeline. Our theoretical results in Chapter 3 suggest that the errors introduced by the ersatz function and the embedding step are not overly large. Is it possible to devise a reranking procedure so that we do not merely control these errors, but reduce their effect on the quality of search results?

It is natural to take a reranking approach in which we use this standard pipeline to perform a fast first-pass search. A naïve approach to this reranking problem would be to simply reorder the search results in decreasing order of similarity to the query, but this approach yields a ranking that reflects the oracle similarity only as well as the ersatz similarity does. If we think of the ersatz similarity function as an estimate of the oracle similarity, then it makes sense to make use of the pairwise similarities for all of the candidate matches. These pairwise similarity scores, taken jointly, define a weighted graph whose vertices correspond to the candidate matches, in which we expect similarity among the candidate matches to yield graph structure that better reflects the oracle similarity. This intuition motivates the results presented in Chapter 4, in which we consider the *vertex nomination* problem. The vertex nomination problem, which we discuss in detail in Appendix E, generalizes this reranking idea by considering a semi-supervised problem in which a few vertices are marked as "interesting" in a given graph $G = (V, E)$, and one wishes to rank the remaining vertices from V so that other vertices also believed to be interesting concentrate at the top of the list. In Chapter 5, we adapt the vertex nomination scheme presented in 4 to the reranking problem discussed above and show that it improves the performance of the audio search system presented in Chapter 2.

1.2 Roadmap

We begin in Chapter 2 by presenting a basic system for performing query-by-example search on large collections of speech audio data. This system illustrates the design issues typical of similarity search, and serves as a testbed for the ideas introduced in later chapters. In Chapter 3, we give more detailed attention to the Laplacian eigenmaps embedding (Belkin and Niyogi 2003) used in the system introduced in Chapter 2. We prove that the Laplacian eigenmaps embedding is robust to misspecification and occlusion of the sort discussed in Section 1.1.1. In Chapter

4, we consider the vertex nomination problem, motivated by the reranking problem discussed in Section 1.1.2. We introduce a maximum-likelihood-based technique for solving the vertex nomination problem and prove its consistency under the stochastic block model. In Chapter 5, we apply our vertex nomination scheme to the reranking problem and show that it improves the performance of the system presented in Chapter 2. We close in Chapter 6 with a discussion of our overall results and directions for future research.

Chapter 2

Low-resource Audio Search Using Fixed-Dimensional Embeddings of Audio Segments

In this chapter, we present a system for performing large-scale search of speech audio in the low-resource setting, where little or no training data is available for building a search system. The low-resource setting is in contrast to the situation usually considered in speech recognition, in which it is assumed that large collections of annotated speech data are available for training statistical models. While such quantities of data are available for well-studied languages such as English and Mandarin, this is not the case for the vast majority of the world's languages. As such, there is a need for approaches to large-scale audio search and related tasks that can operate even with little or no training data. More broadly, this low-resource setting is the norm in many applications beyond speech processing. That is, in many domains and applications, little or no labeled data are available for training supervised statistical models. As such, unsupervised and semi-supervised systems, such as the one presented in this chapter and discussed more broadly throughout this thesis, are crucial.

In the first half of this chapter, we explore a number of methods for representing segmental audio data as fixed-dimensional vectors in such a way that nearness in Euclidean space approximately preserves some notion of linguistic similarity. Such a representation is necessary before we can apply the pipeline discussed in Chapter 1. We consider several methods, varying in the required amount of supervisory information, and compare them on a word discrimination task. We will see that, in particular, an embedding based on Laplacian eigenmaps (Belkin and Niyogi 2003) achieves promising performance on this task. In the second half of this chapter, we will apply these Laplacian eigenmaps embeddings in a large-scale audio search task using a framework akin to that described in Chapter 1. We will see that this search system improves over an earlier system that operated at the frame level rather than performing search at the segmental level.

The material in this chapter appeared originally in slightly altered form in Levin et al. (2013) and Levin et al. (2015). A more detailed introduction to the problem of audio search and indexing, as well as a brief overview of relevant background in speech processing and keyword search can be found in Appendix A. Overviews of fixed-dimensional embeddings and locality-sensitive hashing (LSH), which are central to the search system presented in this chapter, are given in Appendices B and C, respectively.

2.1 Fixed-Dimensional Embeddings of Variable-Length Audio Segments

Historically, the workhorse of speech recognition has been the hidden Markov model (Gales and Young 2008). The speech signal is represented as a sequence of vectors called *frames*. The basic speech recognition architecture consists of an acoustic model, which models the distribution of frames conditioned on a given hidden state, and a language model, which models sequences of states (Jelinek 1997). The hidden states, which can broadly be interpreted as corresponding to phones or other basic units of speech, constitute a sequence of latent variables, assumed to obey the Markov property. That is, the transition the sequence of state transitions is memoryless. Such frame-level independence assumptions make estimation of model parameters and hidden state trajectories feasible (Rabiner 1989), but these assumptions come with well-documented drawbacks (see, for example, Gillick et al. 2011). As a simple example, note that the Markov assumption implies that the number of frames spent in a given state should follow a geometric distribution, while actual durations of speech segments, such as syllables or vowels, do not appear to follow such a distribution (Rosen 2005).

One way forward in light of the shortcomings of frame-based models is to model acoustic features over longer durations, in the hope of capturing segment-level and contextual information. Approaches of this sort, such as sparse exemplar models (Sainath et al. 2012), construct super-vectors of concatenated frames, often followed by dimensionality reduction. While larger windows allow for the modeling of some segment-level information, these windows are still of fixed length. Owing to variation in segment duration (due to, for example, inter- and intra-speaker variability), these fixed-context windows do not always align with meaningful linguistic segments. In contrast, template-based and segmental approaches use variable-length acoustic windows, which enables modeling of whole linguistic units. Template-based acoustic models typically rely on dynamic time warping (DTW; Sakoe and Chiba 1978) to quantify the similarity of phone or word segments (Wachter et al. 2007; Heigold et al. 2012) (refer to Appendix A for an overview of the DTW algorithm and related work). Unfortunately, DTW often misestimates word segment similarity due to, among other factors, oversensitivity to longer phonetic segments (e.g., vowels). Furthermore, DTW alignment requires time polynomial in the duration of the segments being compared. This runtime requirement can prove especially burdensome when comparing test audio to a large repository of exemplars. This drawback could be avoided by embedding arbitrary-length segments into fixed-dimensional spaces in which common distances provide estimates of linguistic dissimilarity. Such embeddings would

(i) enable the application of standard distance learning techniques (Labiak and Livescu 2011; Kulis 2012) to template-based acoustic modeling and

(ii) support a new generation of efficient segment-based audio indexing algorithms,

enabling highly scalable spoken term discovery (Park and Glass 2008; Jansen et al. 2010; Jansen and Van Durme 2011) and query-by-example search (Jansen and Van Durme 2012; Zhang and Glass 2009; Metze et al. 2013) (for an overview of the tasks of spoken term discovery and query-by-example search, refer to Appendix A).

Existing segmental acoustic models use fixed-dimensional representations of hypothesized variable-length segments. The various types of segmental models provide several ways of constructing these representations. These include downsampling (Zue et al. 1989; Glass 2003; Ostendorf 1996; Abdel-Hamid et al. 2013), phonetic acoustic model-derived features (Zweig et al. 2011; Layton and Gales 2005), and convolutional deep neural networks (Maas et al. 2012). These techniques do not necessarily produce linguistically meaningful embeddings, but rather rely on supervision in the segmental feature space for linguistic discrimination. Furthermore, with the exception of basic downsampling, these approaches do not extend well to low- or zero-resource settings, in which supervised training data is limited or non-existent.

With these motivations, we explore multiple unsupervised and supervised approaches to extracting fixed-dimensional embeddings of variable-length audio signals. Our goal is to identify embeddings that preserve word discrimination under simple cosine or Euclidean distances. To apply our techniques to large amounts of speech, we require built-in out-of-sample extension capabilities. We consider three operational settings in which we have access to varying levels of information. At one extreme, we assume that we see each unlabeled speech segment in isolation with no additional training data. Here we are limited, essentially, to downsampling methods. At the opposite extreme, with a training set of word exemplars of known type, we can learn feature maps that maintain word type discrimination. Finally, in the intermediate case, we have a training set of segments of unknown types, but we can still exploit the class-independent distribution of the exemplars. In each of these three cases, we explore both linear and non-linear embeddings and evaluate their effectiveness on a word type discrimination task in a multi-speaker corpus of conversational telephone speech. In all cases, we consider only low-resource settings, i.e., no more than a few hours of labeled speech.

2.2 Embedding Methods

Our goal is to define a function that maps audio signals of arbitrary length to a continuous vector space that parsimoniously encodes the underlying linguistic content. Formally, let $\mathcal{X}$ denote the set of all arbitrary-length acoustic vector time series, $\mathcal{X} = \{X = x_1 x_2 \ldots x_T \mid T \in \mathbb{Z}^+\}$, with each $x_t \in \mathbb{R}^p$, where p is the dimensionality of some frame-level acoustic feature representation (e.g. MFCC, PLP). We would like to learn functions $f : \mathcal{X} \to \mathbb{R}^d$ that map acoustic feature vector time series in $\mathcal{X}$ to points in $\mathbb{R}^d$ such that $f(X)$ and $f(Y)$ are similar if and only if X and Y are acoustic observations generated by similar linguistic units (e.g., phones, morphemes, syllables, words). For now we restrict the discussion and experiments to word segments, but the methods apply similarly to any meaningful unit. We consider three settings for learning these functions, relying on varying amounts of available information:

1. (*NoTrain*) We may access each test word segment $X \in \mathcal{X}$ in isolation with no additional information.

2. (*UnsupTrain*) We have a collection of N_{train} word exemplars $\mathcal{X}_{\text{train}} = \{X_i\}_{i=1}^{N_{\text{train}}}$, with each $X_i \in \mathcal{X}$.

3. (*SupTrain*) In addition to a collection of N_{train} word segments $\mathcal{X}_{\text{train}} \subset \mathcal{X}$, we have the corresponding word labels $\mathcal{W}_{\text{train}} = \{w_i\}_{i=1}^{N_{\text{train}}}$ for those word segments.

In what follows, we define approaches for these three settings. More detailed explanation and discussion of many of these methods can be found in Appendix B.

2.2.1 Time series downsampling

If no information is available to us aside from a given feature vector time series, we must adopt strategies to select a fixed-sized set of observations. The simplest approach is to uniformly downsample so that any segment is represented by a constant number k of vectors: given a feature vector time series $X = x_1x_2\dots x_T \in \mathcal{X}$, with each $x_t \in \mathbb{R}^p$, we sample vectors from X at intervals of T/k with suitable interpolation as needed.

The downsampled time series is concatenated into a single vector of dimensionality $d = kp$. A more sophisticated approach is to perform non-uniform downsampling of the time series using HMMs. For a segment $X = x_1 x_2 \dots x_T \in \mathcal{X}$, we train a k-state left-to-right HMM, modeling the acoustics with a single spherical Gaussian in each state. This approach segments X non-uniformly into k regions. Concatenating the means of these regions into a single vector yields an embedding into $\mathbb{R}^{kp}$ regardless of the length of X. While we restrict our experiments to this HMM-based approach, other HMM-based techniques may be applicable to our setting (e.g., see Tang et al. 2010), as may other non-uniform downsampling approaches (Zue et al. 1989; Glass 2003).

2.2.2 Vector of distances to reference set

When we have access to a collection of training word exemplars $\mathcal{X}_{\text{train}}$, we can consider more sophisticated embedding techniques. Here, we designate a reference set of r exemplars, $\mathcal{X}_{\text{ref}} = \{X_{m_i} | 1 \leq m_i \leq N_{\text{train}}, i = 1, \dots, r\} \subseteq \mathcal{X}_{\text{train}}$, that covers a broad selection of word types and speakers. Given a feature vector time series $X \in \mathcal{X}$, we form a vector $u \in \mathbb{R}^r$ with the i-th component of u given by $\text{DTW}(X, X_{m_i})$, where $\text{DTW}(\cdot, \cdot)$ is the DTW alignment cost between pairs of segments. We refer to u as a *reference vector* for segment X. Note that this is a special case of a Lipschitz embedding in which each reference set has cardinality one (Hjaltason and Samet 2003) and that we use the term *reference set* in a slightly different sense. We can think of this reference vector as representing a word in terms of its similarity to a set of exemplars that forms a “basis” for the space of all words. Thus, this and similar such representations can be applied even to word types not seen in the training set.

One of our motivations for deriving fixed-dimensional word embeddings is to avoid costly DTW alignments over large collections of speech, such as in Jansen and Van Durme (2011, 2012). Here, we are explicitly constructing a representation that requires computing DTW alignment cost against a set of reference examples. While this is an expensive operation, it is still linear in the size of the speech collection if the reference set is fixed. In the context of indexing for search applications, these DTW calculations need only be performed once offline for the entire search collection, allowing sublinear-time search using approximate nearest neighbor techniques (Indyk and Motwani 1998). As commonly employed for costly Lipschitz embeddings, inducing sparsity would also mitigate the computational burden (e.g., see Hristescu and Farach-Colton 1999). In general, the approaches presented here replace DTW alignments with simple Euclidean or cosine distance computations. Thus, letting m and n be the lengths of the vector time series being aligned and letting p be the dimensionality of the vectors in the time series, we replace an operation requiring time $O(mnp)$ with an operation requiring time $O(d)$, where d is the dimensionality of our embedding. Thus, when using the techniques in Indyk and Motwani (1998) to search for a query term of length m in a vector time series of length N, we require only $O(\log N)$ time using approximate nearest neighbor search, rather than $O(Nmp)$

operations required by DTW-based search.

2.2.3 Linear embedding techniques

Linear dimensionality reduction techniques use a collection of labeled or unlabeled data to derive a linear map from the original feature space to a space of lower dimensionality. Applying such techniques to the reference vectors defined in Section 2.2.2, we obtain a projection matrix $P \in \mathbb{R}^{d\times r}$, where $d < r$. Given a new segment $X \in \mathcal{X}$, we project its reference vector $u \in \mathbb{R}^r$ to $u' = Pu \in \mathbb{R}^d$. In the absence of word type information, we may derive P using principal components analysis (PCA). If word labels are available, supervised techniques such as linear discriminant analysis (LDA) can be used. Note that if we use Euclidean distance to compare embedded segment pairs, then operating in the linear embedding space defined by projection matrix P is equivalent to using a Mahalanobis distance parameterized by matrix $M = P^T P$ in the original r-dimensional space.

PCA and LDA. PCA is a well-established unsupervised dimensionality reduction technique. Given $\mathcal{X}_{\text{train}} \subset \mathcal{X}$, we construct the reference vector of each $X_i \in \mathcal{X}_{\text{train}}$. The $d < r$ top (largest-magnitude eigenvalue) eigenvectors of the resulting covariance matrix form a basis of lower dimensionality that best preserves the variance of the data.

When we have word type labels $\mathcal{W}_{\text{train}} = \{w_1, \ldots, w_{N_{\text{train}}}\}$ for the training exemplars, multi-class LDA can be used. Multi-class LDA finds a set of vectors pointing along the directions in which between-class variability is maximized while within-class variability is minimized. Specifically, we form a basis of the first d largest-eigenvalue non-trivial solutions v to the generalized eigenproblem $\Sigma_{\mathrm{B}}v = \lambda\Sigma_{\mathrm{W}}v$, where Σ_{B} and Σ_{W} are the between- and within-class covariance matrices of the training data, respectively. In our implementation, we regularize the within-class covariance matrix with shrinkage by adding a scaled identity matrix.

Metric learning to rank (MLR) Another supervised option is to use one of many existing techniques for discriminatively learning a Mahalanobis distance, given by a positive semidefinite matrix M, with distance between vectors u_1, u_2 defined as $\sqrt{(u_1 - u_2)^T M (u_1 - u_2)}$. Here we use MLR (McFee and Lanckriet 2010), as it optimizes a criterion closely related to our task. MLR is a large-margin approach that aims to separate vectors that are similar to a given query vector from those that are dissimilar by a margin given by a ranking loss, which in our case is mean average precision Given the learned matrix M, we find a matrix U whose i-th row is $\sqrt{|\lambda_i|}v_i$, where v_i is the i-th eigenvector of M with corresponding eigenvalue λ_i. We obtain projection matrix P by retaining only the first d rows of U.

2.2.4 Nonlinear graph embedding techniques

Numerous non-linear dimensionality reduction techniques are available for consideration (e.g., Roweis and Saul 2000; Hinton and Roweis 2002) We use Laplacian eigenmaps (Belkin and Niyogi 2003), including a variant proposed in Belkin et al.

(2006) that defines an out-of-sample extension. In the supervised setting, we can encode word type information by adding graph edges that reflect word identity.

Laplacian eigenmaps. We begin by constructing a graph G with one vertex per training example and edges reflecting the nearest neighbor structure under DTW alignment cost. The binary-valued adjacency matrix $A^{\text{nn}} \in \mathbb{R}^{N_{\text{train}} \times N_{\text{train}}}$ has $A^{\text{nn}}_{ij} = 1$ if and only if example i is one of the k nearest neighbors of example j or vice versa. Given matrix A^{nn}, the *normalized* graph Laplacian operator is defined as $L^{\text{nn}} = I - S^{\frac{1}{2}} A^{\text{nn}} S^{\frac{1}{2}}$, where S is diagonal with $S_{ii} = \sum_j A^{\text{nn}}_{ij}$. Following Belkin and Niyogi (2003), we wish to find a set of d projection maps $\{h_1, \ldots, h_d\}$, where $h_i : V(G) \to \mathbb{R}$, such that vertices near one another under the topology of G are mapped to similar locations in $\mathbb{R}^d$. Since the graph Laplacian operator acts as a measure of smoothness of functions defined on the graph, the desired set $\{h_i\}$ is defined implicitly by the eigenvectors of L^{nn} with the d smallest eigenvalues (after discarding the first trivial eigenvector, which has eigenvalue 0). Each eigenvector encodes the image of the vertex set under a map in $\{h_i\}$.

A problem arises when we wish to project a segment with no corresponding vertex in G into this d-dimensional space. Without some procedure for out-of-sample extension, this technique has little practical utility. An out-of-sample solution for Laplacian eigenmaps is given in Belkin et al. (2006) and is summarized below. We construct matrices A^{nn} and L^{nn} as described above. Our new optimization problem takes the form

$$h^* = \arg\min_{\mathbf{h}\in\mathcal{H}_\kappa} \mathbf{h}^T L^{\mathrm{nn}}\mathbf{h} + \xi\|h\|_\kappa^2, \tag{2.1}$$

where $\mathcal{H}_\kappa$ is the reproducing kernel Hilbert space for some positive semi-definite kernel function $\kappa\!:\!\mathcal{X}\times\mathcal{X}\rightarrow\mathbb{R}$, $\mathbf{h} = \langle h(X_1),\ldots,h(X_{N_{\mathrm{train}}})\rangle^T$ is the vector of values of h computed on the vertices of the graph, and ξ is a non-negative regularization term. We use a kernel function of the form

$$\kappa(X_i, X_j) = \exp\left\{-\frac{[\max(0, \mathrm{DTW}(X_i, X_j) - \eta)]^2}{2\sigma^2}\right\},$$

where $\mathrm{DTW}(\cdot,\cdot)$ is DTW alignment cost and $\eta, \sigma \in \mathbb{R}$. By the RKHS representer theorem (Belkin et al. 2006; Berlinet and Thomas-Agnan 2004), the j-th component of our projection map is

$$h_j^*(X) = \sum_{i=1}^{N_{\mathrm{train}}} \alpha_i^{(j)}\kappa(X_i, X), \tag{2.2}$$

where the $\{\alpha_i^{(j)}\}$ are given by solutions to the generalized eigenvector problem $(L^{\mathrm{nn}}K+\xi I)\alpha = \lambda K\alpha$, with K being the Gram matrix with entries $K_{ij} = \kappa(X_i, X_j)$ for $X_i, X_j \in \mathcal{X}_{\mathrm{train}}$. Intuitively, this eigenproblem attempts to find mappings from $\mathcal{X}_{\mathrm{train}}$ to $\mathbb{R}$ such that word exemplars that are connected in graph G take similar values. In the out-of-sample extension, the kernelization performs an interpolation (similar to the Nyström method; see Appendix B) such that a test exemplar "similar" to a vertex in G takes a similar value. Given the d eigenvectors with the smallest eigenvalues (ignoring the trivial one, as above), we can map an arbitrary segment $X \in \mathcal{X}$ to a point $v \in \mathbb{R}^d$ given by $v = (h_1(X), \ldots, h_d(X))^T$ according to Equation 2.2.

Supervised graph embedding. When available, it is desirable to incorporate class label information into the Laplacian eigenmaps approach. Notable recent algorithms for this problem include locality preserving discriminant analysis (Tomar and Rose 2012), locality sensitive discriminant analysis (LSDA Cai et al. 2007), and marginal Fisher analysis (Yan et al. 2007). In our approach, we construct kernel matrix K and matrix A^{nn} as described above. Additionally, we construct a matrix A^{sup} such that $A^{\text{sup}}_{ij} = 1$ if $i \neq j$ and $w_i = w_j$, and $A^{\text{sup}}_{ij} = 0$ if $w_i \neq w_j$ or if $i = j$. Thus, A^{sup} captures our knowledge of which pairs of words ought to be adjacent to one another in an "ideal" graph reflecting the true class labels. We can combine our supervised and unsupervised information into a single graph Laplacian $L = L^{\text{nn}} + \beta L^{\text{sup}}$, $\beta \in \mathbb{R}$ is non-negative and L^{nn} and L^{sup} are the normalized graph Laplacians of A^{nn} and A^{sup}, respectively. L captures both acoustic similarity and true word label information in a single operator. This is analogous to LSDA, but where we linearly combine the normalized Laplacians of within- and between-class graphs rather than the adjacency matrices. Replacing L^{nn} with L, we proceed as in the previous algorithm, constructing a subspace from the first d non-trivial solutions to Equation 2.1.

LDA applied to graph embeddings. We again assume that we have a labeled set of vector time series, which we use to learn an embedding into $\mathbb{R}^{d'}$ using Laplacian eigenmaps as described above. This map is applied to the training set exemplars and an LDA projection is learned from the resulting vectors and their labels to produce a final embedding into $\mathbb{R}^d$. This two-step process provides an alternate means of introducing supervision into the graph embedding framework. We note that other supervised projections could also be used here, e.g. via Mahalanobis distance learning as in Section 2.2.3, but here we limit ourselves to LDA.

2.3 Comparing Embeddings: Evaluation

To evaluate the techniques described above, we use the task in Carlin et al. (2011), designed to evaluate the word discrimination performance of acoustic front ends and acoustic models that do not explicitly model phones. An evaluation set of presegmented words $\mathcal{X}_{\text{test}}$ is presented. For each pair $(X_i, X_j) \in \mathcal{X}_{\text{test}} \times \mathcal{X}_{\text{test}}$ for $i \neq j$, we compute $D(X_i, X_j)$ under the representation and distance D being evaluated. We set a threshold τ such that we declare words X_i and X_j to be the same if $D(X_i, X_j) \leq \tau$ and declare them to be different otherwise. Discriminative power is then quantified by the average precision (AP), the area under the precision-recall curve, which characterizes discrimination performance at all possible settings of τ. Let $N_{\text{SW}}(\tau)$ denote the number of same-label word pairs with distance less than or equal to τ under the model. We define the model's precision $P_{\text{SW}}(\tau)$ and recall $R_{\text{SW}}(\tau)$ at operating threshold τ as

$$P_{\mathrm{SW}}(\tau) = \frac{N_{\mathrm{SW}}(\tau)}{N(\tau)} R_{\mathrm{SW}}(\tau) = \frac{N_{\mathrm{SW}}(\tau)}{N_{\mathrm{SW}}}, \tag{2.3}$$

where $N(\tau)$ denotes the total number of word pairs in the corpus whose distance under the model is less than or equal to τ (i.e., the number of hypothesized same-word pairs) and N_{SW} is the number of true same-word pairs in the corpus. Thus, to evaluate one of our candidate algorithms, we embed the test set according to that algorithm, compute all pairwise distances between the embedded points and compute the area under the precision-recall curve.

We assembled two collections of words from the Switchboard English corpus, $\mathcal{X}_{\mathrm{train}}$ and $\mathcal{X}_{\mathrm{test}}$, containing $N_{\mathrm{train}} = 10383$ and $N_{\mathrm{test}} = 11024$ words, respectively. Both sets were constrained to include only words of 6 or more orthographic characters and to be at least 50 frames in length (0.5 s). The train and test sets contained 5539 and 3392 word types, respectively, with 6971 unique word types in all. The train set was constructed to have a broad sampling of word types, with at most 5 tokens of any given word type and with each token of a given type taken from a different speaker. The resulting word set covered 360 conversation sides and 156 unique speakers. The test set was identical to that in Carlin et al. (2011). It was constructed to reflect a content word distribution encountered in a typical conversational speech setting. It consisted of all words meeting the above length criteria from 360 conversation sides covering 236 unique speakers, none of whom appeared in the train set. To investigate the effect of acoustic front end on this task, we performed this evaluation using vector time series of

39-dimensional perceptual linear prediction (PLP) feature vectors and 15-dimensional truncated frequency-domain linear prediction (FDLP) feature vectors (Thomas et al. 2009). Previous work has indicated that truncating the spectrum in this way from 13 to 5 dimensions yields a gain in this task relative to front ends with more detailed spectral content (Jansen et al. 2013). Cosine distance, defined for vectors a, b as $1 - a^T b/\|a\|\|b\|$, generally outperformed Euclidean distance for the embedding techniques described here. The basic reference vector and PCA experiments used Euclidean distance between embedded points. All other experiments used cosine distance.

2.3.1 Baselines (the *NoTrain* condition)

Using DTW alignment cost as an inter-word distance measure establishes a baseline for our task. A successful algorithm will be one that can improve upon this result or maintain comparable performance without supervision while being computationally less expensive. Table 2.1 shows the performance of this baseline approach on both PLP and FDLP acoustic features. Also listed in Table 2.1 are the results using uniform and nonuniform downsampling approaches outlined in 2.2.1, where we consider target sample sizes of $n \in \{5, 10, 25, 50\}$ and use cosine distance to compare the resulting supervectors. As is the case for the DTW baseline, the downsampling results using FDLP are consistently comparable to or better than PLP. The gains of nonuniform sampling over uniform are marginal, with the best downsampling APs roughly 1/3 that of the baseline DTW performance for $n \geq 10$.

Table 2.1: Average precision scores achieved by our baseline algorithms in the *NoTrain* condition, by feature type (all scores are given as proportions).

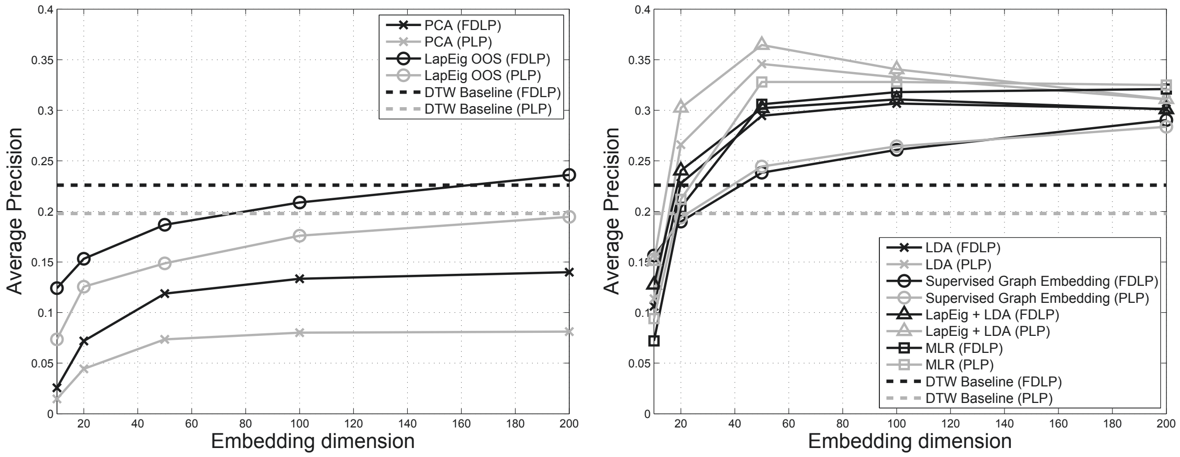
		Ave. Prec.	
Algorithm		**PLP**	**FDLP**
Baseline DTW		0.198	0.226
Uniform Downsampling	$n = 5$	0.036	0.040
	$n = 10$	0.062	0.069
	$n = 25$	0.072	0.081
	$n = 50$	0.074	0.082
Non-uniform Downsampling	$n = 5$	0.050	0.033
	$n = 10$	0.086	0.080
	$n = 25$	0.081	0.088
	$n = 50$	0.076	0.086

2.3.2 Unsupervised embeddings (the *UnsupTrain* condition)

Next we evaluated the reference vectors described in Section 2.2.2. A drawback of this approach (and the approaches that depend on it) is that constructing an acoustic segment's reference vector requires computing $|\mathcal{X}_{\text{ref}}| = r$ DTW alignment costs. Lower-dimensional reference vectors, if still effective in distinguishing words, would allow us to maintain similar performance with fewer DTW calculations required to embed a given word. To examine this possibility, we selected reference sets $\mathcal{X}_{\text{ref}} \subseteq \mathcal{X}_{\text{train}}$ of various sizes r. Reference sets were selected randomly, but biased to favor selecting clusters of same-word tokens. As reflected in Table 2.2, these results fall short of the baseline DTW scores, but they do demonstrate that we can safely shrink the size of our reference set by as much as a factor of 20 without paying too large

Table 2.2: Average precision scores achieved by our basic reference vectors in the *UnsupTrain* condition, by feature type (all scores are proportions).

	Ave. Prec.	
r	**PLP**	**FDLP**
100	0.041	0.078
500	0.089	0.137
1,000	0.089	0.142
5,000	0.094	0.149
10,000	0.096	0.150



(a) **(b)**

Figure 2.1: Average precision as a function of target space dimension for (a) unsupervised embeddings (*UnsupTrain*) and (b) supervised embeddings (*SupTrain*).

a penalty in performance. We leave the problem of optimal reference set design for future work.

We constructed train set reference vectors using a reference set of size r =10,000. We applied PCA to these reference vectors, and applied the learned projection to the test set reference vectors for evaluation. To apply Laplacian eigenmaps to our data, we first calculated all pairwise DTW alignment costs for words in $\mathcal{X}_{\text{train}}$ and, based on those costs, assembled the adjacency matrix A^{nn} and Gram matrix K as described in Section 2.2. Laplacian eigenmaps require setting certain parameters in addition to the target space dimensionality. Performance was reasonably stable for the number of nearest neighbors (k), the regularizer weight (ξ), and the kernel function parameters (η,σ) in the ranges $k \in [7, 30]$, $\xi \in [0.001, 0.1]$, $\eta \in [0.01, 0.05]$, and $\sigma \in [0.15, 0.04]$. We report results for the best-performing parameter settings, leaving the challenge of automatic selection for future work.

Figure 2.1(a) shows the performance of the unsupervised techniques outlined in Section 2.2 for varying target space dimensionalities. We find that using PCA, we can reduce dimension from 10,000 to 100 without substantial loss in performance, but overall performance falls short of the DTW baseline. Laplacian eigenmaps matches the DTW baseline for target dimensionalities $d > 100$ and greatly surpasses PCA at all target dimensionalities, indicating a more efficient use of dimensions than is possible with unsupervised linear methods.

2.3.3 Supervised embeddings (the *SupTrain* condition)

Analogously to PCA, multi-class LDA and MLR were performed on the train set reference vectors with word types as class labels. [1] The resulting linear projections were applied to the test set reference vectors for evaluation. We used a reference

[1] We used Brian McFee's implementation of MLR, available at `https://github.com/bmcfee/mlr/`

set of size $r = 10,000$, except for MLR applied to FDLP features, where we used $r = 5,000$. LDA performance depended moderately on the shrinkage scale factor, observing a change of up to 0.1 AP as we varied the scale factor from 0 to 5. All reported results used a scale factor of 1. MLR results depended moderately on the slack parameter, with typical good values in the range $[10^3, 10^5]$. Supervised graph-based embeddings were obtained using the procedure described in Section 2.2.4. Using the optimal parameter settings for Laplacian eigenmaps and varying β, we found that performance was stable for $\beta \geq 1$, indicating that the utility of supervision dominates that of the nearest neighbor graph structure. Finally, LDA was also applied to the Laplacian eigenmaps embeddings, with the projection again learned on the training set and evaluated on the test set.

Figure 2.1(b) shows the performance of the supervised techniques from Section 2.2 for varying target space dimensionalities. We find that LDA and MLR greatly improve upon the DTW baselines, with AP stable down to 50 dimensions. Interestingly, with supervision the 39-dimensional PLP features usually outperform the cepstral-truncated 15-dimensional FDLP, indicating that increased spectral detail is useful even when supervision is provided indirectly at the word level. Our supervised variant of Laplacian eigenmaps posts significant gains over its unsupervised counterpart, but falls short of direct application of LDA and MLR to the reference vectors. This indicates that supervised discriminative training of a linear embedding is better than nonlinear embedding learned with implicit supervision. This suggests that discriminative nonlinear graph embedding techniques such as marginal Fisher analysis (Yan et al. 2007) may succeed in our setting. LDA applied to the output of unsupervised Laplacian eigenmaps outperforms LDA on its own, indicating that nonlinear graph embedding improves the linear separability of word types.

2.3.4 Discussion

Representative average precision scores for all of our methods are summarized in Table 2.3, organized according to the settings described in Section 2.2, along with the target dimensionalities that yielded the listed scores. For comparison, we include the setting in which an unsupervised Laplacian eigenmap embedding is learned from the test set (*UnsupTest*). This yields the best FDLP performance (0.416 AP) reported here while using only $d = 20$ dimensions. Unfortunately, since it lacks an out-of-sample extension, this embedding is of limited practical utility.

Unsurprisingly, downsampling techniques, even nonuniform ones, fall short of the exhaustive alignment search performed under DTW. Embedding each speech segment with respect to a reference set encodes substantially more duration variability than downsampling, but still does not match the DTW baseline. PCA applied to reference vectors yields good word discriminability with fewer dimensions, but only with supervised embedding (LDA or MLR) do linear methods exceed the DTW baseline. Nonlinear embedding using Laplacian eigenmaps matches DTW using no supervision whatsoever, a significant result for zero-resource applications. Introducing super-

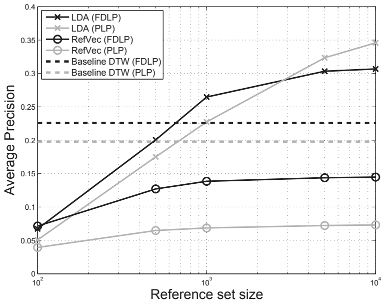


Figure 2.2: Average precision as a function of reference set size.

vision into this algorithm produces substantial gains, but falls short of the linear supervised embeddings produced by LDA and MLR. This indicates that nonlinearity is most important in the unsupervised setting. Combining Laplacian eigenmaps with LDA improves upon LDA alone, suggesting that Laplacian eigenmaps preserves or perhaps magnifies the information that makes LDA effective on its own. While different supervised methods produce the best performance at different operating points – the best performance on PLPs results from LDA applied to Laplacian eigenmaps while MLR posts the best FDLP results – the supervised methods all outperform the baselines and unsupervised methods.

Finally, the reference vectors required by some of our methods are expensive to construct. Table 2.2 shows that reference set size can be reduced with negligible loss in word discriminability. Figure 2.2 shows how reference set size affects task performance, with LDA target dimensionality chosen optimally for each condition. LDA beats the DTW baseline with as few as 1000 reference examples, a promising result, though the large gains in Table 2.3 require several thousand.

Table 2.3: Representative average precision scores attained for each of the embedding schemes using $r = 10,000$ reference examples (when applicable).

			Ave. Prec.	
Setting	**Algorithm**	d	**PLP**	**FDLP**
1. *NoTrain*	Baseline DTW	–	0.198	0.226
	Unif. Downsamp.	$25 \cdot p$	0.072	0.081
	Nonunif. ”	$25 \cdot p$	0.081	0.088
2. *UnsupTrain*	Ref. Vector	10,000	0.096	0.150
	PCA	200	0.081	0.139
	LapEig w/ OOS	200	0.195	0.236
3. *SupTrain*	Sup. LapEig	200	0.284	0.290
	LDA	50	0.346	0.293
	MLR	100	0.328	0.318
	LapEig + LDA	50	0.365	0.302
UnsupTest	Unsup. LapEig	20	0.253	0.416

2.4 Large-Scale Audio Keyword Search

Having explored a number of potential embeddings of acoustic segments, we turn to applying them to the task of keyword search. In keyword search, we are given an example utterance, and wish to locate occurrences of that utterance in a collection of speech audio. Keyword search has received increasing attention in recent years as speech data has become more ubiquitous and ever more integral to mobile phone technology. Consider, for example, that in 2012, YouTube users uploaded one hour of video every second. [2] To search audio collections of this magnitude, we must able to build speech processing systems of unprecedented scale. Most previous approaches have employed lattice indexing techniques (Miller et al. 2007), enabling search of thousands of hours of speech in interactive time. Typical systems build a model to map sequences of frames to segmental units (e.g., phones or words) that are more

[2] `http://www.onehourpersecond.com` Accessed October 30, 2016.

amenable to standard lattice-based approaches. Unfortunately, these techniques require large collections of annotated speech audio to be used as training data, and such training data sets are unavailable in most languages. As a result, the zero-resource setting, in which detailed annotations are unavailable and linguistic structure must be discovered without the aid of training data, has attracted attention both in the speech processing community (Glass 2012) and among scientists interested in human language acquisition (Jansen et al. 2013).

Query-by-example search, where search terms are presented as audio segments rather than in graphemic or phonetic form, has applications in probing large collections of unstructured audio data (Anguera et al. 2013) and in voice interfaces (Chen et al. 2014, 2015). The standard approach involves training a model to map query audio to a sequence of symbols (e.g., a phonetic representation) and searching for this sequence in a lattice built on the search collection (Parada et al. 2009). Finite state automata techniques have made lattice search of this kind both fast and accurate (Allauzen et al. 2004), but the nature of the required training data makes these approaches infeasible in zero- and low-resource settings.

Dynamic time warping (DTW), explained in Appendix A, has been effective in zero-resource query-by-example search (Park and Glass 2008; Jansen et al. 2010; Anguera and Ferrarons 2013). Unfortunately, as mentioned in Section 2.1 and Appendix A, DTW sequence alignment requires time linear in the size of the search collection, which limits its scalability. Techniques such as those in Mantena and Anguera

(2013); Zhang and Glass (2011) have improved runtime by, in essence, reducing the constants in this linear dependence. In contrast, the Randomized Acoustic Indexing and Logarithmic-Time Search (RAILS) system introduced in Jansen and Van Durme (2012) avoids this linear dependence altogether. Given an audio query, RAILS operates in two steps. First, for each frame of the query, similar frames (with similarity defined by cosine distance between frames) are retrieved from the search collection using logarithmic-time approximate nearest-neighbor retrieval (see Appendix C for an overview of near-neighbor retrieval and related problems). As a second step, these frame-level matches are extended to segment-level matches using image processing techniques.

The RAILS system has two main limitations. First, its accuracy depends ultimately on DTW as a measure of segment-level similarity, an issue mentioned above and discussed at more length in Appendix A. Second, the process by which frame-level matches are extended requires a computationally expensive digital image processing step, which introduces a major runtime bottleneck. This motivates the Segmental RAILS (S-RAILS) system, an extension of the RAILS methodology that avoids both of these shortcoming by performing search directly at the segment level using the fixed-dimensional segmental embeddings explored above. As we have seen, such embedding techniques show a marked improvement over a purely DTW-based approach as measured by performance on the evaluation task introduced in Carlin et al. (2011). Further, by performing search directly at the segment level, we avoid the need to extend frame-level matches as in the original RAILS system. In what follows, we introduce the S-RAILS system and evaluate its performance on a query-by-example keyword search task on a corpus of telephone speech, in which our system improves dramatically over the original RAILS system in both accuracy and runtime.

2.5 The S-RAILS System

The S-RAILS system is an adaptation of the RAILS query-by-example search system presented in Jansen and Van Durme (2012). In RAILS, indexing consists of building a structure to support fast approximate nearest-neighbor retrieval at the frame level using an adaptation of the point location in equal balls (PLEB) algorithm (Indyk and Motwani 1998). Given a query, the near neighbors of each frame in the query are retrieved from the index along with scores reflecting their similarity. These frame-level candidate matches are then extended to segment-level matches using digital image processing. We refer the reader to Appendix A for a more detailed description.

These near neighbor frames along with their scores yield a sparse approximation to the frame-level similarity matrix, the entries of which correspond to similarities between frames in the query and frames in the search collection. Segments of the search audio that are similar to the query give rise to approximately diagonal lines in the similarity matrix. These diagonal lines in turn appear as peaks in the Hough transform of the matrix, and thus can be quickly located.

S-RAILS differs from the original RAILS system by indexing the acoustic features of whole word-sized segments directly, altogether avoiding both the intermediate step of frame-level indexing and the need to construct a similarity matrix. It operates as follows:

1. Voice activity detection (VAD) locates regions likely to contain speech.

2. Each VAD region is split into overlapping segments from some minimum duration to some maximum duration. Each segment is mapped to a fixed-dimensional vector using techniques discussed previously in this chapter.

3. An index is constructed for randomized approximate nearest-neighbor retrieval (Indyk and Motwani 1998) on the collection of fixed-dimensional embeddings. Each segment created in the previous step appears as an entry in the index.

4. At query time, a query segment is mapped to its fixed-dimensional representation and the near-neighbors of that representation are retrieved from the index.

5. Candidate matches to a query can be rescored after retrieval, e.g., by computing exact DTW scores as in Jansen and Van Durme (2012).

Figure 2.3 provides a system diagram of the S-RAILS system.

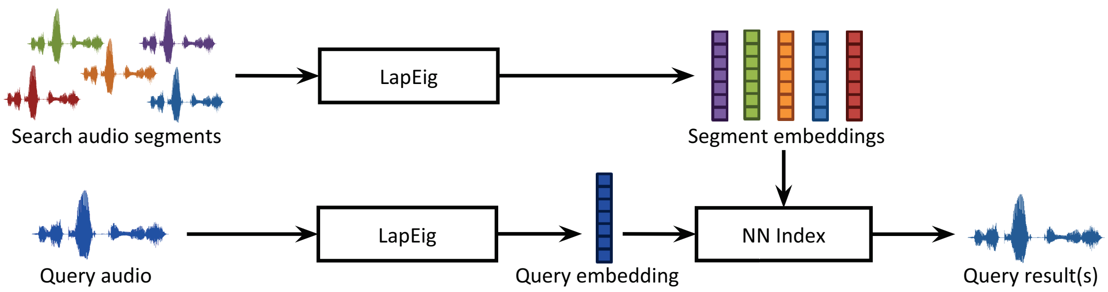


Figure 2.3: Diagram of the S-RAILS audio search system.

2.5.1 Fixed-dimensional Segment Embeddings

To obtain fixed-dimensional representations of speech segments, we use the unsupervised Laplacian eigenmaps embedding described in Section 2.2.4. Letting $\mathcal{X}$ denote the set of all arbitrary-length feature vector time series, $\mathcal{X} = \{X = x_1, x_2, \ldots, x_T : T \in \mathbb{Z}^+\}$, where each $x_i \in \mathbb{R}^p$ and p is the dimensionality of a speech frame, we learn this embedding using a reference set $\mathcal{R} = \{X_1, X_2, \ldots, X_n\} \subset \mathcal{X}$ and a kernel function

$$\kappa(X_i, X_j) = \exp\left\{-\frac{[\max(0, \mathrm{DTW}(X_i, X_j) - \eta)]^2}{2\sigma^2}\right\},$$

where $\mathrm{DTW}(\cdot,\cdot)$ denotes DTW alignment cost and $\eta, \sigma \in \mathbb{R}$ are parameters to be specified.

2.5.2 Near-neighbor retrieval

A crucial step in both RAILS and S-RAILS consists of retrieving a set of embeddings that are similar to a query embedding. Our goal is to build an index which, given a query vector, returns vectors from the index that are near to the query vector under cosine distance. To solve this problem, RAILS used an implementation of point location in equal balls (PLEB) as presented in Indyk and Motwani (1998). PLEB makes use of locality sensitive hash (LSH) functions, which capture the geometric proximity of pairs of items in the sense that nearby items are likely to be hashed to the same value and distant items are unlikely to be hashed to the same value. An overview of the state of the art in LSH and near neighbor search is given in Appendix C.

The LSH variant used here is similar to that used in the original RAILS system Jansen and Van Durme (2012). We map vectors to binary strings of length S, which we call *signatures*. This mapping is chosen such that cosine distance between two vectors can be approximated by some function of the Hamming distance between their respective signatures. These signatures are generated by randomly choosing a set of S hyperplanes through the origin in the vector space. Each bit of a vector's signature is determined by which side of a corresponding hyperplane it falls on. Pairs of vectors with small cosine distance are unlikely to be separated by a randomly-chosen hyperplane, and thus their signatures are likely to be similar. This permits fast retrieval of the approximate near neighbors of a given query vector by computing its signature and returning all vectors from the search collection whose signatures are at a small Hamming distance from it.

The near-neighbor retrieval algorithm used in S-RAILS is discussed in detail in Jansen and Van Durme (2011) and we summarize it here. We let B denote the *beamwidth*, a parameter that controls the number of near neighbors that we retrieve. Retrieval is performed by sorting the signatures in the search collection and returning signatures that share a prefix with the query signature. Given a collection of signatures $\mathcal{Z} = \{z_1, z_2, \ldots, z_N\}$ with each $z_i \in \{0, 1\}^S$, we sort the elements of $\mathcal{Z}$ in lexicographic order. Let π be a permutation of the integers $1, 2, \ldots, N$ such that $z_{\pi(1)}, z_{\pi(2)}, \ldots, z_{\pi(N)}$ is the lexicographic sort of the elements of $\mathcal{Z}$. Given a query signature $q \in \{0, 1\}^S$, we find via binary search the location where q belongs in the sorted list and return the B signatures before that position and the B signatures after that position. That is, if q belongs between $z_{\pi(i)}$ and $z_{\pi(i+1)}$ in the sorted list, we return the set $\{z_{\pi(a)}, z_{\pi(a+1)}, \ldots, z_{\pi(b)}\}$, where $a = \max\{1, i - B + 1\}$ and $b = \min\{N, i + B\}$. Of course, in this lexicographic sorting scheme, a given ordering of the signature bits means that bits appearing early in the signature have a greater influence over which pairs of signatures are considered similar. This problem is mitigated by performing several of these searches under different permutations of the signature bit ordering. We denote by P the number of such permutations that we use. In practice, rather than repeatedly permuting and sorting the signature list, we keep P separate lists of the search collection signatures, each sorted according to a different one of the P permutations. Retrieval of near-neighbors under this scheme requires time logarithmic in N and linear in both P and S. We have observed in our experiments that runtime depends only weakly on S compared to dependence on P and N.

Table 2.4: S-RAILS performance on the *development* search collection, averaged over all query types as a function of signature length S for fixed number of permutations $P = 8$ and beamwidth $B = 10,000$. All scores are percentages.

	Median Example			**Best Example**		
S	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
64	22.3	9.8	9.1	48.7	26.2	45.4
128	27.5	11.4	11.4	56.0	30.4	55.1
512	30.4	14.0	14.4	57.7	33.8	59.1
1024	30.2	13.9	14.8	58.3	35.0	60.7

Table 2.5: S-RAILS performance on the *development* search collection, averaged over all query types as a function of number permutations P for fixed beamwidth $B = 100,000$ and signature length $S = 512$. All scores are percentages.

	Median Example			**Best Example**		
P	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
4	31.3	13.6	15.2	60.7	34.1	58.7
8	33.1	14.5	15.4	63.0	35.2	59.6

2.6 Experiments

Our experiments follow those presented in Jansen and Van Durme (2012). We evaluated our system in a query-by-example keyword search task on the Switchboard corpus, a collection of conversational telephone speech. A 37-hour collection was set

Table 2.6: S-RAILS performance on the evaluation search set, averaged over all query types as a function of beam width B for fixed number of permutations $P = 8$ and signature length $S = 512$. All scores are percentages except Real Time Speedup, which is the ratio of search collection duration to the average time required to perform a single query.

	Median Example			**Best Example**			
B	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**	**Real Time Speedup**
100	7.6	6.0	39.3	19.8	15.5	85.3	307,000,000
1,000	15.0	9.7	38.3	34.1	21.8	87.4	40,800,000
10,000	26.0	12.7	38.6	47.7	26.3	91.6	5,770,000
100,000	37.3	15.1	38.6	56.9	29.6	89.3	510,000

Table 2.7: Baseline RAILS performance, reproduced from Jansen and Van Durme (2012), on the *evaluation* search set averaged over all query types as a function of beamwidth B. All scores are percentages except Real Time Speedup, which is the ratio of search collection duration to the average time required to perform a single query.

	Median Example			**Best Example**			
B	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**	**Real Time Speedup**
500	0.8	0.9	21.0	3.6	2.8	58.4	620,000
5,000	6.7	2.7	44.0	20.7	10.4	84.4	63,000
50,000	19.0	4.7	49.2	39.9	16.5	88.4	7,000
100,000	20.2	4.8	49.8	41.1	16.6	88.1	3,600

aside from which to draw query terms, a 48-hour development search collection was used to explore the effect of different parameters on the system's performance, and a 433-hour evaluation set was used to obtain final performance metrics. Query word types were chosen to have corpus-wide median duration of at least 0.5 seconds and orthographic representation at least six characters long. This resulted in a collection of 43 query word types:

> absolutely basically benefit bottles business California college community companies control crimes definitely deterrent employees expenses expensive important individual insurance interesting mandatory Massachusetts newspaper organization performance plastic policy positive process program punishment recently recycle recycling retirement salary savings situation society understand unfortunately university vacation

Each query type appeared between 20 and 162 times in the query set, between 2 and 188 times in the development search collection, and between 39 and 1386 times in the evaluation collection. More than half of the selected query types had median duration less than 0.55 s and all query types had median duration less than 0.75 s. We considered three common keyword search metrics:

(i) *Figure-of-merit* (FOM), the average recall over the 10 operating points at which the false alarm rate is $1, 2, \ldots, 10$ false alarms per hour of search audio.

(ii) *Oracular term weighted value* (OTWV), a weighted difference between the system's recall and false alarm rate. The oracular variant of this metric assumes an optimal query-specific threshold. See Miller et al. (2007) for a detailed account of this metric.

(iii) *Precision at 10* (P@10), the fraction of the top ten ranked candidate matches that are correct.

Metrics were computed separately for each query type, and are reported as unweighted averages over all 43 query types. Performance is sensitive to the specific query example. Thus, for each metric, we report both (i) the median query example performance, and (ii) the best query example performance.

2.6.1 Selecting Index Parameters

Table 2.4 shows the effect of signature length on system performance for fixed beamwidth $B = 10,000$ and number of permutations $P = 8$. Performance saturates at a signature length of 512 bits. These signatures are larger than the 64-bit signatures used in RAILS owing to the fact that RAILS indexes 39-dimensional feature vectors while S-RAILS indexes 1000-dimensional fixed-dimensional embeddings. As a result, a larger number of bits are required to achieve suitably high fidelity in approximating cosine distance between vectors. Table 2.5 shows system performance as a function of the number of permutations for fixed beamwidth $B = 100,000$ and signature length $S = 512$. We see that $P = 8$ yields a non-negligible performance gain over $P = 4$ in the best-example case, though median performance is largely insensitive to P. These two tables jointly suggest that performance saturates at a signature length of 512 bits and $P = 8$. We use these parameters in the remainder of our evaluation.

2.6.2 Constructing the Index

To segment the search collection, candidate segment boundaries were placed at 3-frame intervals in all VAD regions. Resulting segments with duration at least 40 frames (400 ms) and at most 100 frames (1 s) were included in the index. To construct Laplacian eigenmaps embeddings, we used a set of 10,383 unlabeled word examples from the Switchboard corpus to define our similarity graph. As discussed previously, the process of constructing Laplacian eigenmaps embeddings is slow, since a single embedding requires computing a DTW alignment of a segment with every segment in the similarity graph. Indeed, this process is currently the major bottleneck in constructing an index. In order to speed up the embedding process, rather than explicitly computing $\text{DTW}(X, X_i)$ for all i as in (2.2), we performed a spectral clustering of the 10,383-segment similarity graph and selected a representative segment (the medoid) from each cluster. Given a segment $X \in \mathcal{X}$ to embed, its DTW alignment was computed with each cluster representative. For representatives whose alignment cost was above some threshold, we set $K(X, X_i) = 0$ for all X_i in the corresponding cluster rather than computing exact alignment costs. Experiments showed that 550 clusters with a threshold of 0.17 yielded a very good approximation to the true values of the kernel function. This approximation yielded a factor of 6 speedup with respect to the exact computation, but even with this speedup, computing fixed-dimensional embeddings of speech audio is approximately 130 times slower than real time on current hardware. This process produced approximately 30 million 1,000-dimensional embeddings in the case of the development search collection and approximately 280 million in the case of the evaluation search collection, which became the input to the index.

2.6.3 Controlling False Positives

By the nature of the Laplacian eigenmaps embedding, word examples that are not similar to any words in the reference set are mapped to locations near the origin. At query time, when similarity search is performed under cosine distance, many of these small-norm embeddings are retrieved as candidate matches. This results in many false positives, reflected in the low median precision at 10 scores in Tables 2.4 and 2.5. To reduce this effect, we removed from the index all embeddings with norm less than a set threshold τ_{thresh}. Table 2.8 summarizes the effect of this thresholding. We found a threshold of 0.06 to be best, though performance was comparatively flat for thresholds between 0.01 and 0.1. In experiments on the development search set, this resulted in 50% to 70% relative improvements in median precision at 10, as well approximately 8% relative improvement in maximum precision at 10 and, somewhat surprisingly, small improvements on all other metrics.

Table 2.8: Effect of signature threshold on S-RAILS performance on the *development* search collection, averaged over all query types. All experiments use signature length $S = 512$, number of permutations $P = 8$ and beamwidth $B = 10,000$. All scores are percentages.

	Median Example			**Best Example**		
τ_{thresh}	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
0.2	23.3	16.8	28.6	29.7	24.3	50.4
0.1	36.3	22.9	27.7	54.2	39.1	64.6
0.09	37.4	23.1	26.1	55.6	39.6	64.1
0.08	37.8	23.2	24.7	57.5	40.0	64.2
0.07	37.8	22.6	23.5	59.2	40.1	64.3
0.06	38.6	22.1	23.7	60.3	40.7	65.8
0.05	37.9	21.3	23.1	60.5	40.3	64.6
0.02	36.4	17.6	19.6	61.6	38.5	60.7
0.01	33.8	16.0	16.9	60.6	36.7	61.0
0.005	32.9	15.2	16.2	59.6	35.4	58.7
0.001	31.0	14.2	15.3	58.2	34.3	58.9
0.0001	30.5	14.0	14.4	57.7	33.8	59.1
0.0	30.4	14.0	14.4	57.7	33.8	59.1

2.6.4 Post-processing of query results

Owing to the segmentation scheme used in S-RAILS, the index contains many entries corresponding to overlapping segments, and our embedding technique causes these segments to be mapped to similar fixed-dimensional vectors. The result is that at query time, if one of these segments is retrieved, many other overlapping segments are likely to be retrieved, as well. To eliminate this redundancy, we performed a post-processing step in which retrieved segments whose midpoints were within a given number of frames of one another were greedily merged by discarding the segment with lower score. This operation was repeated until no further merge operations could be performed. We found that merging pairs of segments whose midpoints were within 10 frames of one another proved effective.

2.6.5 Results

Table 2.6 shows system performance on the evaluation search set as a function of beamwidth for fixed number of permutations $P = 8$ and signature length $S = 512$. Table 2.7 shows performance of the original RAILS system for comparison. We note that values of B in RAILS and S-RAILS are not directly comparable, since the two systems operate on different objects, though both systems' runtimes depend linearly on the parameter. Comparing the best performance of the two systems, we see that S-RAILS achieves more than 80% relative improvement over RAILS in median example FOM and upwards of 200% relative improvement in median example OTWV. In the case of best example performance, S-RAILS exhibits approximately 78% relative improvement in OTWV and 38% relative improvement in FOM performance. P@10 scores are less decisive. S-RAILS improves marginally on RAILS in best example P@10, but lags by a non-negligible margin in median example P@10. As alluded to previously, this is due to a small number of particularly high-scoring false alarms introduced by the embedding process. This issue might be ameliorated by a suitable rescoring procedure.

Comparing system runtimes paints a more impressive picture. S-RAILS tends to achieve a speedup of between two and five orders of magnitude with respect to RAILS at any given performance level. To take a particularly striking example, S-RAILS with $B = 100$ achieves better median OTWV performance than RAILS with $B = 200,000$ while running more than 85,000 times faster.

Chapter 3

Laplacian Eigenmaps in the Presence of Noise and Occlusion

In Chapter 2, we compared the performance of several embedding techniques on a word similarity task, and found that Laplacian eigenmaps embeddings yielded strong performance in all training conditions. We saw subsequently that these embeddings yield strong results on a large-scale audio search task. Under the pipeline described in Chapter 1, we would have liked to compute an embedding using the entire search collection as the reference set, but we saw that this was infeasible, since this would require computing DTW alignments for all pairs of segments in the search collection. This motivated our use of the Laplacian eigenmaps out-of-sample extension, which allowed us to compute an embedding of a small number of examples (the reference set), and extend that embedding to apply to the entire search collection, as well as the queries.

We observed in Chapter 2 that the quality of our embeddings thus depended on the reference set. Hence, one way to improve the embedding of the search collection would be to carefully choose a reference set from among the segments in the search collection. Indeed, this was the intuition behind our centroid-based method to speed up the out-of-sample embedding. It is natural to ask, however, whether we might devise an embedding scheme that embeds the entire search collection more accurately. In Chapter 1, we discussed the question of how well a given embedding preserves the similarity structure of the search collection. The acoustic embeddings considered in Chapter 2 rely on the computation of a matrix of pairwise DTW alignments. When the reference set is large (as is the case when we wish to embed the entire search collection), this matrix is expensive to compute. Further, we know DTW is at best an approximation to some ideal notion of word similarity.

These concerns motivate the problem considered in this chapter, in which we investigate the behavior of Laplacian eigenmaps when we replace the kernel matrix $\mathscr{K}$ with a sparse, noisy approximation, in which we have noisy estimates of $\mathscr{K}_{ij}$ for only a handful of the entries of $\mathscr{K}$. Our results show that from our sparse, noisy version of $\mathscr{K}$, we can obtain embeddings that are of quality comparable to those obtained from using the full, clean version of $\mathscr{K}$. These results have applications beyond the search problems considered in this thesis, owing to the ubiquity of embeddings in machine learning. Problems of the sort considered here limit the viability of many dimensionality reduction techniques, which tend to require the computation of all pairwise distance or similarity functions on a set of objects.

The results presented in this chapter appeared first in Levin and Lyzinski (to Appear).

3.1 Introduction and Motivation

Manifold-based dimensionality reduction techniques operate under the assumption that data observed in a high-dimensional space lie on a low-dimensional manifold (Tenenbaum et al. 2000; Roweis and Saul 2000; Belkin and Niyogi 2003; Belkin et al. 2006). Owing to the ubiquity of large high-dimensional data sets, these techniques have been well studied, with applications across many disparate fields (see Appendix B for a more thorough discussion of manifold learning and related material). In addition to the classical linear techniques such as PCA (Jolliffe 2002), MDS (Cox and Cox 2001) and CCA (Hotelling 1936; Hardoon et al. 2004), numerous manifold embedding procedures have been proposed to discover intrinsic low-dimensional structure in nonlinear data. These nonlinear techniques, such as ISOMAP (Tenenbaum et al. 2000) and Laplacian eigenmaps (Belkin and Niyogi 2003), typically attempt to preserve some notion of local geometry in the embedding. As such, they tend to be empirically robust to modest noise and outliers, but general theoretical results in this direction are comparatively few.

In this chapter, we theoretically and practically explore the robustness of Laplacian eigenmaps to very general noise conditions. This work differs from most manifold embedding robustness results in two key ways: first, we assume that the uncertainty lies not in the observations themselves, but rather in our measurement of the pairwise similarities used to construct the kernel matrix. Second, the noise model is entirely nonparametric: we make no distributional assumptions on the noise other than unbiasedness (see Equation (3.2) below).

3.1.1 Problem Description

Suppose that $\mathcal{X}$ is a set of objects, endowed with a notion of similarity captured by a kernel function $\sigma : \mathcal{X} \times \mathcal{X} \to [0, 1]$; i.e., $x, y \in \mathcal{X}$ are similar if $\sigma(x, y) \approx 1$, and $x, y \in \mathcal{X}$ are not similar if $\sigma(x, y) \approx 0$. Given n observations $x_1, x_2, \ldots, x_n \in \mathcal{X}$, we can represent their similarities via a hollow, undirected weighted graph with adjacency matrix $\mathscr{K}$ given by

$$\mathscr{K}_{ij} = \begin{cases} \sigma(x_i, x_j) & \text{if } i \neq j \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Manifold-based dimensionality reduction techniques seek to recover the low-dimensional structure intrinsic in the similarities captured by $\mathscr{K}$. We note that some manifold embedding algorithms rely on distance or dissimilarity measures rather than similarities, but the distinction is immaterial here.

As discussed in Chapter 1, it is often the case that while a researcher may have some oracle similarity σ in mind, one must typically fall back on an ersatz similarity κ that only approximates σ. If κ only approximately captures the oracle notion of similarity between observations, it is natural to ask how this influences the quality of the embedding. Similarly, when $\kappa(x, y)$ is expensive to compute, we might ask whether an embedding of similar quality is possible based on an inexpensive approximation or by computing $\kappa(x, y)$ for only a fraction of all pairs of observations, and inferring the rest of $\mathscr{K}$, for example, by applying Chatterjee's universal singular value thresholding (USVT; Chatterjee 2015).

The Laplacian eigenmaps embeddings constructed in Chapter 2 serve as an illustrative example. Recall that for word examples x_i and x_j, the corresponding entry in the kernel matrix is given by

$$\kappa(x_i, x_j) = \exp\{-d^2(x_i, x_j)/2\sigma^2\},$$

where $d(x_i, x_j)$ is a function of the dynamic time warping (DTW) alignment cost (Sakoe and Chiba 1978) between x_i and x_j. This choice of kernel function is only an approximation to an idealized notion of word similarity, that we cannot hope to compute– as mentioned in Chapter 2, the inadequacies of DTW as a word similarity measure are well known. Additionally, DTW alignment is computationally expensive, requiring time that scales as the product of the lengths of the two aligned sequences. As such, a fast estimate of $d(x_i, x_j)$ or $\kappa(x_i, x_j)$ is acceptable, and we would prefer to avoid computing all $O(n^2)$ alignments required to populate the kernel matrix.

3.1.2 Our Model

In light of the above, we consider the following model. We assume a fixed set of observations $x_1, x_2, \ldots, x_n \in \mathcal{X}$, and an oracle similarity function σ defined on $\mathcal{X} \times \mathcal{X}$, giving rise to a true but unknown symmetric kernel matrix

$$\mathscr{K} = [\mathscr{K}_{ij}] = [\sigma(x_i, x_j)] \in [0,1]^{n \times n}.$$

The embedding learned from $\mathscr{K}$ is the best embedding we could hope to learn, in that it accurately and completely captures all the information available to us about $x_1, x_2, \ldots, x_n$. The data processing inequality (Cover and Thomas 2006) implies that given the data, kernel function and embedding procedure, adding noise and occlusion to $\mathscr{K}$ cannot improve the embeddings from the standpoint of subsequent inference or classification. Suppose, however, that rather than observing $\mathscr{K}$, we observe a random symmetric matrix $Y \in \mathbb{R}^{n \times n}$, whose entries are generated independently as

$$Y_{ij} = Y_{ji} = \begin{cases} K_{ij} & \text{with probability } p \\ 0 & \text{with probability } (1-p), \end{cases} \tag{3.2}$$

where the $K_{ij} \in [0,1]$ are independent random variables with $\mathbb{E}K_{ij} = \mathscr{K}_{ij}$ and $p \in [0,1]$ is the (expected) fraction of entries of $\mathscr{K}$ that are observed. We note that our results hold for similarity functions bounded by any constant, and our use of the range $[0,1]$ is without loss of generality. We can think of K as a corrupted version of $\mathscr{K}$, with errors reflecting, for example, the failure of the ersatz similarity κ to fully reflect the oracle similarity σ, or approximation error arising from estimating a computationally expensive $\kappa(x,y)$. Similarly, we can view the sparsity of Y as reflecting the fact that when n is large or κ is expensive to compute, we would like to avoid computing all $O(n^2)$ pairwise similarities. Our model is meant to account for general uncertainty in the kernel matrix, which may come from many sources (e.g., computational restrictions, estimation, etc.). Ultimately, we require only that errors be entry-wise independent and unbiased.

When $\mathscr{K}_{ij} \approx 0$ or $\mathscr{K}_{ij} \approx 1$, our model allows K_{ij} very little variance. In many applications, the cases when $\kappa(x,y) \approx 0$ or $\kappa(x,y) \approx 1$ are less prone to error, which is reflected in our model. Indeed, it is often easy to detect when two observations are very similar or very dissimilar, whereas one expects higher variance in estimation of similarity when, say, $\kappa(x,y) = 1/2$.

Remark 1 (Error Generalization)**.** Our model is a good approximation to more complicated error models. As an example, consider the Gaussian kernel $\kappa(x,y) = \exp\{-d^2(x,y)/\beta^2\}$, where $\beta > 0$ is the kernel bandwidth. A more natural but less tractable error model is one in which D_{ij} is an estimate (possibly biased) of $d(x_i,x_j)$

and our kernel matrix is $K_{ij} = \exp\{-D_{ij}^2/\beta^2\}$, say, $D_{ij} = d_0 + E_{ij}$ where E_{ij} is a random error term. A Taylor expansion of $\exp\{-t^2/\beta^2\}$ about $d_0 = d(x_i, x_j)$ shows that (taking $\beta = 1$ without loss of generality and using the fact that $\mathscr{K}_{ij} = e^{d_0^2}$)

$$K_{ij} = \mathscr{K}_{ij} - 2d_0 e^{-d_0^2} E_{ij} + (4d_0^2 - 2)e^{-d_0^2} E_{ij}^2 + O(E_{ij}^3).$$

We see that so long as the error term E_{ij} is reasonably well-behaved, we still have $\mathbb{E}K_{ij} \approx \mathscr{K}_{ij}$, and an approximate version of the results presented here will hold. More broadly, we note that so long as $|\mathbb{E}K_{ij} - \mathscr{K}_{ij}|$ is suitably small for most entries, our results can be extended to the case of biased errors. These observations are borne out by experiment (See Figures 3.3 and 3.4).

In this paper, we theoretically and practically explore under what conditions it is suitable to use the embedding learned from Y in place of $\mathscr{K}$. Under such conditions, we can obtain embeddings with quality comparable to those produced from $\mathscr{K}$, at a greatly reduced computational cost. In the present work, we consider the performance of Laplacian eigenmaps (Belkin and Niyogi 2003; Belkin et al. 2006) under this model, though we believe that the results extend to other embedding techniques, as well.

3.1.3 Laplacian Eigenmaps

As presented in Chapter 2, Laplacian eigenmaps (Belkin and Niyogi 2003; Belkin et al. 2006) embeds the observed data $\mathcal{X}$ into $\mathbb{R}^d$ by first constructing the k-nearest-neighbor (k-NN) or ϵ-graph $G = (V, E)$ from $\mathcal{X}$. In the k-NN graph, an edge is present between i and j if x_i is among the k nearest neighbors (according to some distance defined on $\mathcal{X}$) of x_j or vice versa. In the ϵ-graph, i and j are adjacent if $\|x_i - x_j\|^2 < \epsilon$ for a given threshold parameter ϵ. We define W, the weighted adjacency matrix of G, by

$$W_{ij} = \begin{cases} \mathscr{K}_{ij} & \text{if } \{i, j\} \in E \\ \\ 0 & \text{else,} \end{cases}$$

and let $\mathscr{D} \in \mathbb{R}^{n \times n}$ be the diagonal matrix defined by $\mathscr{D}_{ii} = \sum_j W_{ij}$ for $i \in [n]$. Then the normalized weighted graph Laplacian of G (Chung 1997) is given by $\mathscr{L}(W) = \mathscr{D}^{-1/2} W \mathscr{D}^{-1/2}$. If the eigendecomposition of $\mathscr{L}(W)$ is given by $\mathscr{L}(W) = U \Lambda U^\top$ with the diagonal entries of Λ nonincreasing, then Laplacian eigenmaps embeds $\mathcal{X}$ via $U[:, 2 : d+1]$—the first d nontrivial eigenvectors of $\mathscr{L}(W)$. (note that $U[:, 1] = \vec{\mathbf{1}}$, the trivial all-ones vector). This embedding optimally preserves the local geometry of $\mathcal{X}$ in a least squares sense.

In the event that $\mathscr{K}$ is noisily and incompletely observed as Y, how does the d-dimensional Laplacian eigenmaps embedding of Y compare with that of $\mathscr{K}$? Our main result, Theorem 1, deals with the regularized matrix $[Y_{ij} + r]$ rather than Y itself, owing to the fact that when p is small, the matrix $p\mathscr{K} = \mathbb{E}Y$ may be quite sparse, in the sense that some or all of the row sums $\sum_{j=1}^{n} p\mathscr{K}_{ij}$ are too small to guarantee necessary concentration inequalities (Oliviera 2010; Tropp 2012; Le et al. 2016). Regularization prevents this pitfall, at the cost of changing the matrix to which we converge. We discuss regularization at more length in Section 3.2.3. Intuitively, our main theorem states that the embedding produced from a regularized version of Y is similar to that produced by $\mathscr{K}$. This implies that we can avoid the $O(n^2)$ exact computations for $\mathscr{K}$, using instead the potentially less computationally expensive Y, with little loss in downstream performance.

Remark 2. We depart from Laplacian eigenmaps as originally described (Belkin and Niyogi 2003) and as used in Chapter 2 in that we do not build a k-NN graph or ϵ-graph from $\mathcal{X}$. However, a suitably-chosen kernel function (e.g., the Gaussian kernel) ensures that $\mathscr{K}$ approximates a k-NN or ϵ-graph, with Y a noisily-observed subgraph of $\mathscr{K}$.

3.1.4 Notation and conventions

For a set S, we denote the complement of S by S^c. For a matrix $B \in \mathbb{R}^{n \times n}$, we let $\lambda(B)$ denote the multi-set of eigenvalues of B, and for $S \subset \mathbb{R}$, we define $\lambda_S(B) = \lambda(B) \cap S$. We let $J \in \mathbb{R}^n$ denote the matrix of all ones.

We make use of standard big-O notation, writing $f(n) = O(g(n))$ to mean that there exists a constant $C > 0$ such that $f(n) \leq Cg(n)$ for suitably large n. Similarly, we write $f(n) = o(g(n))$ to mean that $f(n)/g(n) \to 0$ as $n \to \infty$. We use $f(n) = \Omega(g(n))$ to denote that f grows at least as quickly as g does, i.e., to denote that $g(n) = O(f(n))$, and we write $f(n) = \omega(g(n))$ when $g(n) = o(f(n))$.

Throughout this chapter, all quantities are assumed to depend on n, a fact that we highlight by subscripting or superscripting with n (e.g., $\mathscr{K} = \mathscr{K}^{(n)}$), but which we will suppress in many places for ease of notation. Our main theorem, Theorem 1, is a finite-sample result, with $\mathscr{K}^{(n)}$ viewed as fixed for each n, and $K^{(n)}$ and $Y^{(n)}$ randomly generated from $\mathscr{K}^{(n)}$. We note that all of our results in this chapter can be restated as holding almost surely as $n \to \infty$ by assuming suitable lower bounds on the constants in the supporting Lemmas so as to ensure that the probabilities of the various "bad events" are summably small. An application of the Borel-Cantelli lemma then implies that our desired events hold almost surely. This modification can be made to work either in the case (a) where we view Y, K and $\mathscr{K}$ as (growing, "nested") principle submatrices of infinite matrices, or (b) in the case where we consider a sequence of fixed matrices $(\mathscr{K}^{(n)})_{n=1}^{\infty}$.

In this chapter, we assume $\mathscr{K}$ to be fixed for each n (i.e., not random– the randomness lies entirely in Y and K). This assumption is made primarily for the sake of brevity and simplicity, since randomness in $\mathscr{K}$ would have to come from random selection of the sample $x_1, x_2, \ldots, x_n \in \mathcal{X}$ according to some distribution F on $\mathcal{X}$. Clearly, the properties of $\mathscr{K}$ depend on the properties of F and $\mathcal{X}$, but a thorough exploration of precisely how F and $\mathcal{X}$ influence $\mathscr{K}$ is beyond the scope of this thesis.

3.2 Related Work

We briefly survey some existing work from the fields of manifold learning, matrix completion and matrix concentration as it relates to the work presented in this chapter. These works are discussed in more detail in Appendices B and D.

3.2.1 Manifold Learning

Manifold learning is a general class of techniques for nonlinear dimensionality reduction that seek to embed a collection of observations into Euclidean space in a way that preserves some aspect of the structure of those observations. For example, given a collection of objects and some notion of distance on those objects, we may wish to embed the objects into Euclidean space in such a way that all pairwise distances are (approximately) preserved (Indyk 2001; Linial 2002). A host of different embedding techniques have been proposed in the literature (see, for example, Roweis and Saul 2000; Tenenbaum et al. 2000; Cox and Cox 2001; Hinton and Roweis 2002; Donoho and Grimes 2003; Coifman and Lafon 2006) to preserve the numerous different notions of structure in the data. As outlined in Yan et al. (2007), it is possible to view many of these approaches as special cases of a more general framework

There is a large amount of literature dedicated to improving the performance of manifold learning and dimensionality reduction algorithms in the presence of noise and missing data; see, for example, Chang and Yeung (2006); Hein and Maier (2007);

Candès et al. (2011); Shahid et al. (2015). The present work differs from most such results in the following key ways: We assume that the uncertainty lies not in the observations themselves, but rather in the computation of the pairwise similarities or distances used to construct the kernel matrix, and our model of this uncertainty is nonparametric. Additionally, we make no assumption that the observations lie in Euclidean space. Rather, the objects under study are arbitrary (e.g., they may be time series, graphs, etc.), and information about the geometry of $\mathcal{X}$ comes through the ersatz kernel function κ.

With the rise of big data and the continued popularity of kernel methods, much research has gone toward faster construction and embedding of the kernel matrix by speeding up the evaluation of the kernel function itself (Williams and Seeger 2001; Le et al. 2013), the embedding procedure (Baglama and Reichel 2005; Brand 2006), and construction of the kernel matrix as a whole (Fine and Scheinberg 2001). Construction of the kernel matrix is often the major bottleneck in machine learning systems (Hofmann et al. 2008; Levin et al. 2013, 2015). In our model, embedding the partially observed noisy kernel matrix Y allows for potentially dramatic speedups compared to the computation of the full, clean kernel $\mathscr{K}$. A similarly-motivated idea was explored in Chen et al. (2009), where the authors presented a pair of divide-and-conquer algorithms for approximately constructing k-NN graphs on observations in Euclidean space. However, unlike our approach, they do not consider noise in the observations themselves or in the assessment of distances between observations.

Another close analogue to our present work is Rohe et al. (2011a), in which the authors theoretically and empirically explored the robustness properties of spectral clustering: i.e., Laplacian eigenmaps applied to a binary adjacency matrix followed by k-means clustering. In the language of this thesis, they considered the inner product kernel matrix $\mathscr{K} \in \mathbb{R}^{n\times n}$ on a fixed (but unknown) subset $\mathcal{X} \subset \mathbb{R}^d$. From this kernel, they observed the matrix $Y \in \{0,1\}^{n\times n}$ with independent entries

$$Y_{ij} = Y_{ji} = \begin{cases} 1 & \text{with probability } \mathscr{K}_{ij} \\ 0 & \text{with probability } (1-\mathscr{K}_{ij}). \end{cases} \tag{3.3}$$

They compared the Laplacian spectral embedding based on $\mathscr{K}$ with that based on Y. Their key result showed that, under some mild assumptions on the spectrum of $\mathscr{L}(\mathscr{K})$ (the normalized Laplacian of $\mathscr{K}$), the eigenspace of $\mathscr{L}(Y)$ does not significantly differ from the corresponding eigenspace of $\mathscr{L}(\mathscr{K})$ (after suitable rotation). As a result, they prove that spectral clustering of $\mathscr{L}(Y)$ consistently estimates the clusters obtained by spectrally clustering $\mathscr{L}(\mathscr{K})$. While our main theorem uses results (Rohe et al. 2011a, Prop. 2.1 and Thm. 2.2) developed in that paper, the generality of our occlusion model (3.2) compared to (3.3) requires new proof techniques. Additionally, our manifolds do not necessarily have a well-defined cluster structure (as the stochastic blockmodel graphs of Rohe et al. (2011a) do), and so we do not consider consistency of clustering of our embedding. Rather, in Theorem 1, we prove that the relevant eigenvectors of $\mathscr{L}(Y)$ do not significantly differ from the corresponding eigenvectors of

$\mathscr{L}(\mathscr{K})$. As in Rohe et al. (2011a), we expect the consistency of subsequent inference to similarly follow.

3.2.2 Matrix Completion and Data Imputation

A natural approach to applying Laplacian eigenmaps to Y is to first impute the missing entries of Y using matrix completion techniques. For example, with the additional assumption that $\mathscr{K}$ is approximately low-rank, it would be possible to impute the missing data via the techniques developed in compressed sensing (Candès and Recht 2009, see Appendix D for a survey of the relevant literature). While some compressed sensing papers have considered matrix completion in the presence of both noise and occlusion (Candès and Plan 2009; Chen et al. 2013), most also require bounds on the incoherence of matrix $\mathscr{K}$, a requirement that need not hold in general for the kernel matrices we consider here.

Some matrix completion work has considered imputing missing entries in a distance matrix (Alfakih et al. 1999; Trosset 2000; Javanmard and Montanari 2013). Among these, the work by Javanmard and Montanari (2013) is closest in spirit to the problem considered here. Javanmard and Montanari (2013) considered the problem of placing n objects into d-dimensional Euclidean space based on noisy, occluded measurements of the $O(n^2)$ pairwise distances. Their semidefinite programming-based approach solves this problem under a very general error model, where nothing is known about the errors other than a bound on their magnitude. However, their model differs from ours in two key ways. First, the observations in question are assumed to lie in d-dimensional Euclidean space, while ours need only be endowed with a kernel function. Second, they assume that distance measurements are taken on all pairs of points within a fixed radius of one another. However, under our model, all entries of $\mathscr{K}$ are equally likely to be (noisily) observed.

Chatterjee (2015) considered the problem of completing an arbitrary matrix based on partial, noisy observations, with no specific assumptions on the matrix structure. His universal singular value thresholding (USVT) procedure constructs a minimax optimal estimate for $\mathscr{K}$ based on its occluded, noisy measurement Y (as defined in (3.2)). Though we believe that the results obtained in this paper would hold in a qualitatively similar way if we used USVT applied to matrix Y prior to embedding, analyzing the behavior of the USVT estimate of $\mathscr{K}$ under the graph Laplacian is theoretically challenging, and we do not pursue it further here. In empirical comparisons, we found our method and Chatterjee's USVT performed nearly identically across our data sets. We do note that USVT requires an expensive SVD computation, and yields a dense matrix as an estimate of $\mathscr{K}$, instead of the sparse Y, which may be computationally intractable for large n.

3.2.3 Matrix Concentration

Recent years have seen a flurry of results proving concentration results for sums of random matrices (Oliviera 2010; Tropp 2012; Chaudhuri et al. 2012; Amini et al.

2013; Joseph and Yu 2014; Qin and Rohe 2013; Le et al. 2016; Tropp 2015), in the spirit of their well-established scalar analogues (Chung and Lu 2006). Many existing concentration results require assumptions about the density of the underlying graphs (Rohe et al. 2011a; Oliviera 2010). For example, many such results hold only in the dense regime and require a lower bound on the average degree (i.e., a lower bound on the row sums of the expected value of the random matrix). It is well known that the high variance associated with small average degree precludes concentration of the Laplacian for general weighted graphs (Chung et al. 2003; Le et al. 2016; Klopp et al. 2015). This is an issue for the problem considered in the present work, especially when we observe only a small fraction of the matrix entries.

Existing empirical and theoretical results show that regularization yields the desired concentration of the graph Laplacian for sparse graphs (see Chaudhuri et al. 2012; Amini et al. 2013; Joseph and Yu 2014; Qin and Rohe 2013; Le et al. 2016, and references therein). This regularization typically takes the form of either adding a small number to each entry of the adjacency matrix, as in Le et al. (2016), or by adding to the degree matrix directly, as in Qin and Rohe (2013). Our result draws on this line of work by investigating the behavior of the Laplacian eigenmaps embeddings when regularization is applied. In this sense, the current work is a natural outgrowth of Rohe et al. (2011a) and Le et al. (2016) in that the former considers concentration of the Laplacian eigenmaps embeddings under the Frobenius norm, and the latter considers concentration of the regularized graph Laplacian under the spectral norm. We follow the former of these two works and consider concentration under the Frobenius norm, rather than spectral norm. This differs from the bounds established in Oliviera (2010); Tropp (2012); Le et al. (2016), which show concentration of the adjacency matrix and graph Laplacian under the spectral norm. We prefer the Frobenius norm formulation of Theorem 1, as the Frobenius norm between the (suitably rotated) eigenspaces has a natural interpretation as the Procrustes alignment error of the two different embeddings.

3.3 Main Results

Our goal is to theoretically and empirically understand the impact of observation error on the embedding obtained via Laplacian eigenmaps. That is, how much does the embedding obtained using matrix Y degrade with respect to that obtained using matrix $\mathscr{K}$? We prove that Laplacian eigenmaps is indeed robust to certain amounts of both occlusion and noise by first proving that (a suitably regularized version of) $\mathscr{L}^2(Y)$ concentrates about (a regularized version of) $\mathscr{L}^2(p\mathscr{K})$, where Y and p are defined as in Equation (3.2). Combining this result with the Davis-Kahan theorem (Davis and Kahan 1970), we obtain in Theorem 1 a guarantee that the embedding learned from the occluded noisy kernel matrix is similar (up to rotation) to that learned from the regularized clean kernel matrix. We provide relevant details below and in Section 3.6.

Let $G = (V, E)$ be an undirected, loop-free, weighted graph on n vertices with edge weights $w_{ij} \geq 0$. We represent G by its adjacency matrix $A \in \mathbb{R}^{n\times n}$, with entries

$$A_{ij} = A_{ji} = \begin{cases} w_{ij} & \text{if } \{i,j\} \in E \\ 0 & \text{if } \{i,j\} \notin E. \end{cases}$$

Given A, we define its normalized graph Laplacian by

$$\mathscr{L}(A) = \mathscr{D}(A)^{-1/2} A \mathscr{D}(A)^{-1/2},$$

where $\mathscr{D}(A) \in \mathbb{R}^{n\times n}$, the degree matrix, is diagonal with $\mathscr{D}(A)_{ii} = \sum_{j=1}^{n} A_{ij}$ and inverse square root defined as

$$\left(\mathscr{D}(A)^{-1/2}\right)_{ii} = \begin{cases} 1/\sqrt{\mathscr{D}(A)_{ii}} & \text{if } \mathscr{D}(A)_{ii} \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

We note that the graph Laplacian as we have defined it differs from the more commonly used $I - \mathscr{D}(A)^{-1/2} A \mathscr{D}(A)^{-1/2}$ (e.g., in Chung 1997). We will be interested in the eigenspace of $\mathscr{L}(A)$, and one can easily check that both our $\mathscr{L}(A)$ and the more commonly used definition have the same eigenspaces.

In general, neither the adjacency matrix nor the graph Laplacian of sparse random graphs concentrate about their means owing to high variance in degree distributions (Chung et al. 2003; Feige and Ofek 2005; Le et al. 2016). This suggests that we should not expect that $\mathscr{L}(Y)$ will concentrate for arbitrary kernel matrices, and hence we turn to regularization. Let $J \in \mathbb{R}^{n\times n}$ denote the matrix of all ones. Our main result will require us to bound $\|\mathscr{L}^2(Y + rJ) - \mathscr{L}^2(p\mathscr{K} + rJ)\|_F$, where Y is the sparse, noisy version of $\mathscr{K}$ as specified in (3.2), and $r \geq 0$ is a regularization parameter. We deal with the squared Laplacians for reasons discussed in Rohe et al. (2011a, Section 2). Namely, we require that $\mathscr{L}(Y + rJ)$ converge to $\mathscr{L}(p\mathscr{K} + rJ)$ in Frobenius norm. To ensure convergence for a suitably broad class of matrices, we must instead consider the squared Laplacians in combination with the following Lemma, proved in Rohe et al. (2011a), which ensures that if certain eigenvectors of $\mathscr{L}^2(Y + rJ)$ converge, then so do the relevant eigenvectors of $\mathscr{L}(Y + rJ)$.

Lemma 1 (Rohe et al. 2011a, Lemma 2.1)**.** *Let $B \in \mathbb{R}^{n\times n}$ be symmetric.*

1. *λ^2 is an eigenvalue of B^2 if and only if either λ or $-\lambda$ is an eigenvalue of B.*

2. *If $Bx = \lambda x$, then $B^2 x = \lambda^2 x$.*

3. *If $B^2 x = \lambda^2 x$, then x can be written as a linear combination of eigenvectors of B with corresponding eigenvalues λ or $-\lambda$.*

Our main theorem, Theorem 1, shows that the span of the eigenvectors corresponding to the largest eigenvalues of the Laplacian of $\mathscr{K}$ and the Laplacian of the sparse noisy kernel matrix Y are close. As a consequence, subsequent inference performed on the Laplacian eigenmaps embeddings will be robust to the errors introduced in Y, since the embeddings will be (nearly) isometric to one another. In the statement of the theorem, we include subscript or superscript n on all quantities that depend on n, though we will drop these subscripts in the sequel for notational convenience. Recall that for $B \in \mathbb{R}^{n \times n}$, $\lambda(B)$ denotes the multi-set of eigenvalues of B and for $S \subset \mathbb{R}$, we define $\lambda_S(B) = \lambda(B) \cap S$.

Theorem 1. *Under the model described in* (3.2)*, for an open interval $S_n \subset \mathbb{R}$, define*

$$\delta_n = \inf\{|\ell - s| : \ell \in \lambda_{S_n^c}(\mathscr{L}^2(p\mathscr{K} + r_n J)), s \in S_n\}, \tag{3.4}$$

$$\delta_n' = \inf\{|\ell - s| : \ell \in \lambda_{S_n}(\mathscr{L}^2(p\mathscr{K} + r_n J)), s \in S_n^c\}, \text{ and}$$

$$S_n' = \{\ell : \ell^2 \in S_n\}.$$

Let $k_n = |\lambda_{S_n'}(\mathscr{L}(Y^{(n)} + r_n J))|$ be the cardinality of $\lambda_{S_n'}(\mathscr{L}(Y^{(n)} + r_n J))$ (counting multiplicities), and let $X_n \in \mathbb{R}^{n \times k_n}$ be the matrix whose columns form an orthonormal basis for the subspace spanned by the eigenvectors of $\mathscr{L}(Y^{(n)} + r_n J)$ with corresponding eigenvalues in $\lambda_{S_n'}(\mathscr{L}(Y^{(n)} + r_n J))$. Let $\bar{k}_n = |\lambda_{S_n'}(\mathscr{L}(p\mathscr{K}^{(n)} + r_n J))|$ and let $\mathcal{X}_n$ be the analogue of X_n for $\mathscr{L}(p\mathscr{K}^{(n)} + r_n J)$.

Let r_n depend on n in such a way that $r_n \min\{\delta_n, \delta_n'\} \geq n^{-1/2} \log n$ for suitably large n. There exist constants $C, c > 0$ and a positive integer N such that $n \geq N$ implies that $k_n = \bar{k}_n$, and there exists orthonormal rotation matrix $\mathcal{O}_n$ such that with *probability at least* $1 - n^{-c}$,

$$\|X_n - \mathcal{X}_n \mathcal{O}_n\|_F \leq C \left(\frac{\log^{1/2} n}{\delta_n r_n n^{1/2}} \right).$$

Proof. By reasoning analogous to that in Rohe et al. (2011a) Theorem 2.3, the assumption on the growth rates of r_n, δ_n, and δ'_n, in combination with Theorem 3, is sufficient to ensure that $k_n = \bar{k}_n$ for suitably large n. For all such n, combining Theorems 2 and 3 yields the result. □

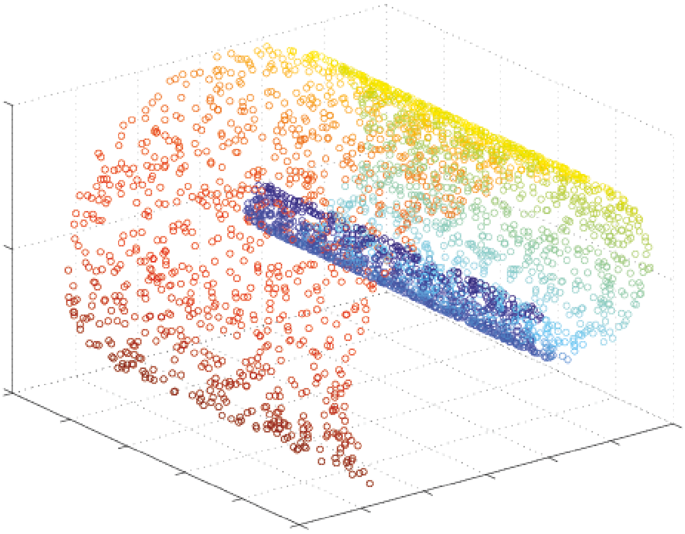


Figure 3.1: Points sampled from a 3-dimensional swiss roll.

Remark 3. A key difference between the main theorem in Rohe et al. (2011a) and our result is that we do not require a restriction on the degrees of $p\mathscr{K}$ directly. Rather, we use regularization to ensure that no row sum is too small. We note that letting $p = 1$ and making minor adjustments to the arguments in our concentration inequalities (namely, lower bounds on the entries of the degree matrix $\mathscr{D}$), we recover the main result of Rohe et al. (2011a), with a slightly better convergence rate. Namely, if we define $\tau = n^{-1} \min_{i \in [n]} \mathscr{D}_{ii}$, our result has τ^{-1} controlling to rate of convergence of the eigenspaces rather than τ^{-2} as in Rohe et al. (2011a) (with dependence on n and δ unchanged)

Remark 4. We note the somewhat surprising fact that the bound in 1 does not depend explicitly on p. This is a result of the presence of regularization parameter r, which prevents $p\mathscr{K} + r$ from becoming too sparse. We note that if one imposes stronger assumptions on the growth of p (namely, restricting the speed with which p can approach 0), our proofs can be adapted to dispense with r altogether, in which case p appears in the bounds instead.

Our main tool for proving Theorem 1 is the Davis-Kahan theorem (Davis and Kahan 1970), which we use in the form presented in Rohe et al. (2011a). We here index all quantities by n to reiterate that all quantities are allowed to depend on n, but remind the reader that we will drop this indexing in much of the sequel for ease of notation.

Theorem 2. *Let $S_n \subset \mathbb{R}$ be an interval and let $\mathcal{X}_n$ be a matrix with orthonormal columns that span the same subspace as that spanned by the eigenvectors of $\mathscr{L}^2(p_n \mathscr{K}^{(n)})$ with corresponding eigenvalues in*

$$\lambda_{S_n}(\mathscr{L}^2(p_n \mathscr{K}^{(n)} + r_n J)) = S_n \cap \lambda(\mathscr{L}^2(p_n \mathscr{K}^{(n)} + r_n J)).$$

Define X_n analogously for $\mathscr{L}^2(Y^{(n)} + r_n J)$. Let δ_n be defined for $\mathscr{L}^2(p_n \mathscr{K}^{(n)} + r_n J)$ as in (3.4)*.*

If $\mathcal{X}_n$ and X_n are of the same dimension, then there exists orthonormal matrix O_n, which depends on $\mathcal{X}_n$ and X_n, such that

$$\begin{aligned}\frac{1}{2}&\|X_n - \mathcal{X}_n O_n\|_F^2 \\ &\leq \frac{\|\mathscr{L}^2(Y^{(n)} + r_n J) - \mathscr{L}^2(p_n \mathscr{K}^{(n)} + r_n J)\|_F^2}{\delta_n^2}.\end{aligned}$$

To apply Theorem 2 toward Theorem 1, we need a concentration bound for $\mathscr{L}^2(Y + rJ)$ about $\mathscr{L}^2(p\mathscr{K} + rJ)$. We note that Y, $\mathscr{K}$, J and r all implicitly depend on n, a fact that we do not generally make explicit in the sequel for ease of notation, but which we highlight here for clarity. For each $n = 1, 2, \ldots$, let $\mathscr{K}^{(n)}$ be a weighted adjacency matrix for a graph on n points in $\mathcal{X}$ as defined in (3.1). Similarly, let $Y^{(n)}$ be the corresponding sparse noisy kernel matrix as defined in (3.2).

Theorem 3. *Assume that regularization parameter r grows with n in such a way that $r = \omega(n^{-1} \log n)$. There exist constants $C, c > 0$ such that for suitably large n,*

$$\|\mathscr{L}^2(Y + rJ) - \mathscr{L}^2(p\mathscr{K} + rJ)\|_F \leq C \frac{\log^{1/2} n}{r n^{1/2}}$$

with probability at least $1 - n^{-c}$.

Proof. This theorem is proven in Section 3.6. □

Remark 5. A number of results exist concerning concentration of the adjacency

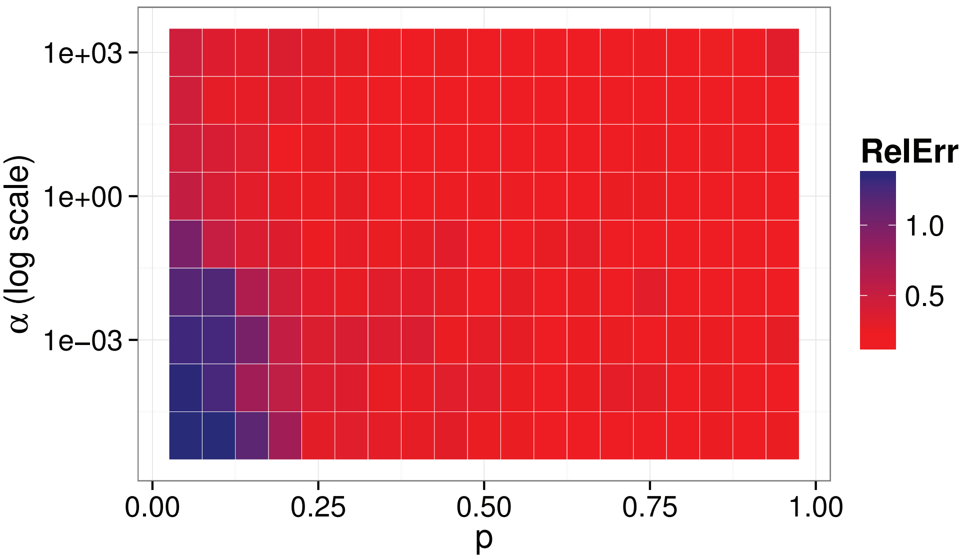


Figure 3.2: Relative error (RelErr) in recovering the clean embedding of the high-dimensional swiss roll as a function of noise and occlusion. Each tile reflects the mean of 50 independent trials. We see that recovery is possible with low relative error except in the extreme case of simultaneous high-noise and heavy occlusion, suggesting that the embeddings are robust to both noise and occlusion of the kernel matrix.

matrix and the graph Laplacian of random graphs (see, for example, Feige and Ofek 2005; Oliviera 2010; Rohe et al. 2011a; Tropp 2012; Le et al. 2016). In general, these results show that the graph Laplacian concentrates in spectral norm about its mean when the quantity $d = n \max_{1 \leq i < j \leq n} p_{ij}$ is of size $\Omega(\log n)$ (here p_{ij} is the probability of an edge appearing between nodes i and j in the random graph). Our result differs from most of these, in that we are concerned with concentration under the Frobenius norm, rather than the spectral norm. We obtain results in a similar regime, as captured by our lower bound requirements on the regularization term r.

A key quantity in Theorem 2 is the spectral gap δ_n as defined in (3.4). δ_n measures how well the eigenvalues in $\lambda_S(\mathscr{L}^2(p\mathscr{K}^{(n)}))$ are isolated from the rest of the spectrum.

δ_n must grow in such a way that for suitably large n, the eigenvalues falling in S_n correspond to the eigenvectors of interest, and the rate of this growth is one of the factors controlling the convergence in Theorem 1. The existence of this eigengap is crucial for the application of the Davis-Kahan Theorem (Davis and Kahan 1970; Rohe et al. 2011a). The eigengap depends on the matrix $p\mathscr{K}^{(n)}$ (i.e., on the topology of the graph this matrix encodes). As discussed in von Luxburg (2007), the existence of such a gap is a reasonable assumption when, for example, the data set (viewed through similarity function κ) has a cluster structure.

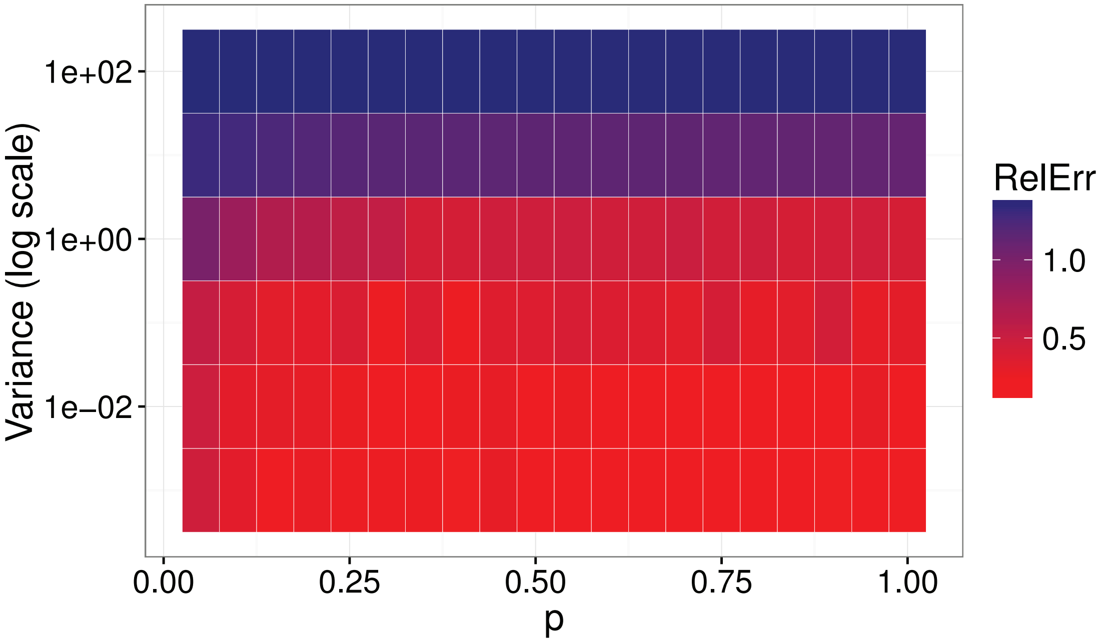
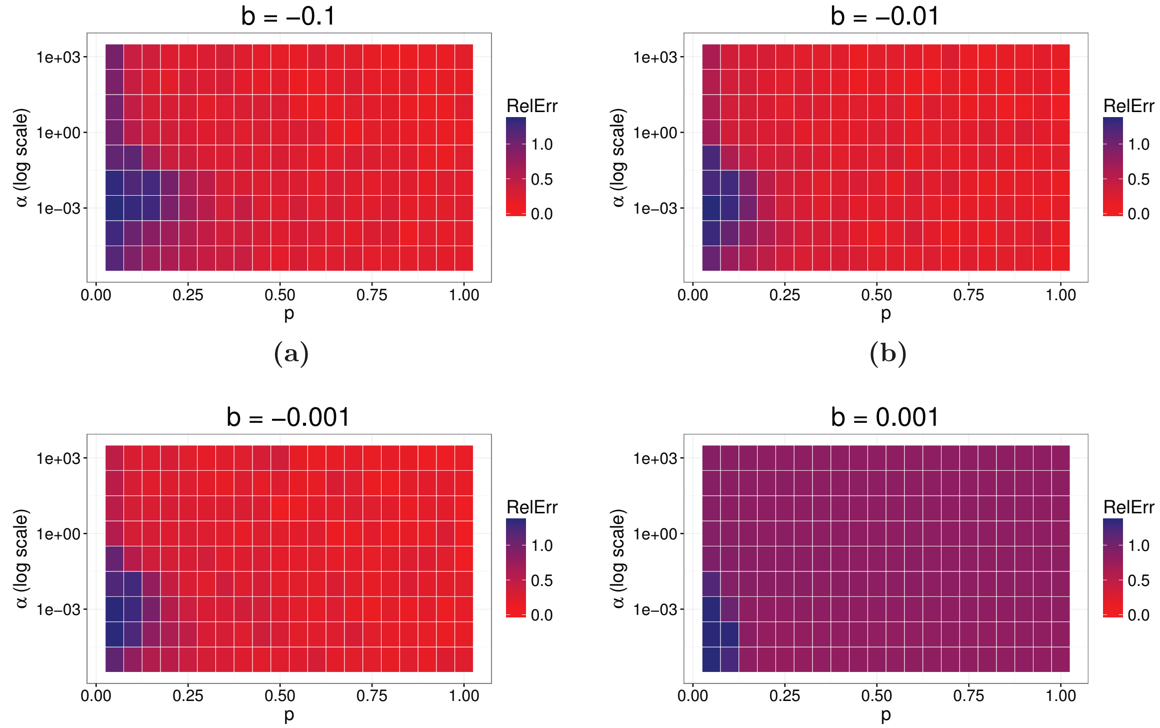


Figure 3.3: Relative error in recovering the Laplacian eigenmaps embedding of the high-dimensional swiss roll as a function of occlusion and variance ν^2 in the multiplicative error model described in Equation (3.5). Each tile is the mean of 50 independent trials. We see that Laplacian eigenmaps is robust to moderate amounts of multiplicative noise, with reasonably good recovery at all values of p provided $\nu^2 \leq 1$ (which we recall is five times the kernel bandwidth $\sigma = 0.2$), but performance degrades sharply when uncertainty on the distance measure becomes too large ($\nu^2 \geq 10$).



(a) (b) (c) (d)

Figure 3.4: Relative error (RelErr) in recovering the clean embedding of the high-dimensional swiss roll as a function of occlusion and noise level for different levels of bias b. Each tile is the mean of 50 independent trials. We see that Laplacian eigenmaps embedding is quite robust to negative bias, but that even a small amount of positive bias in the errors causes a marked decrease in performance at all noise and occlusion levels.

Typically, computing the Laplacian eigenmaps embedding of a data set is not an end in itself, but rather a processing step performed prior to subsequent inference, classification, or data exploration. Such tasks depend entirely upon the geometry of the embedded data points produced by Laplacian eigenmaps. If the geometry of the points produced from the inexpensive embedding based on Y is approximately equal (up to rotation) to that of the embedding based on $\mathscr{K}$, then we can expect comparable performance on downstream tasks that are invariant under rotations of the data (e.g., clustering). Thus, our results show that we can obtain performance comparable to that obtained when using the dense, computationally intensive $\mathscr{K}$ while avoiding the expense of working with $\mathscr{K}$ directly.

3.4 Experiments

In this section, we present simulation and real-world data to complement our theoretical results in Section 3.3.

3.4.1 Data Sets

We consider three data sets, one synthetic, one from connectomics, and one from the speech processing literature.

Synthetic Data (Fig. 3.1, 3.2, 3.6, 3.3, 3.4).

We consider a high-dimensional analogue of the 3-dimensional swiss roll manifold (see Fig. 3.1). We sample n points uniformly at random from the d^*-dimensional unit cube and embed those points into $(d^* + 1)$-dimensional space by applying the swiss roll transform

$$(x, y) \mapsto (cx\cos(cx), y, cx\sin(cx)), \quad x \in \mathbb{R}, y \in \mathbb{R}^{d^*-1}$$

where c controls the curvature of the manifold. In all experiments we use $n = 5000$,

$d^* = 6$ and $c = 5$. We chose this higher-dimensional version of the well-understood, simple swiss roll manifold to examine the effect of both under- and over-estimating the dimension d^*. We obtain a kernel matrix $\mathscr{K}$ from these points by applying a Gaussian kernel with bandwidth σ. Results are fairly stable for a wide range of values of σ. We use $\sigma = 0.2$ in all experiments, while stressing that the task of selecting parameters in dimensionality reduction techniques warrants much additional study.

C. elegans **Connectome (Fig. 3.8).**

We consider the task of clustering the 253 non-isolated neurons in the *C. elegans*, a nematode commonly used as a simple biological model (see Chen et al. 2016, and citations therein). These neurons are categorized according to their function: sensory neurons, interneurons and motor neurons, which make up 27.96%, 29.75% and 42.29% of the connectome, respectively. Our data consists of the symmetric binary adjacency matrix corresponding to the *C. elegans* brain graph, in which each node corresponds to an individual neuron, with an edge between two neurons if they share a synapse. As discussed in Chen et al. (2016), this brain graph can be constructed in multiple ways. Here we consider the subgraph of the chemical connectome induced by the non-isolated vertices of the electrical gap junction connectome. Our goal is to embed the nodes of this graph via Laplacian eigenmaps so that clustering (e.g., by k-means) recovers the three neuron categories enumerated above. We assess the quality of these embeddings using adjusted Rand index (ARI; Hubert and Arabie 1985), which measures how well two partitions agree, adjusted for chance.

Speech Data (Fig. 3.5, 3.7 and 3.9).

We consider the same word discrimination task as in Chapter 2, using a set of $10,383$ spoken word examples, representing $5,539$ distinct word types. Using DTW alignment cost, we define a radial basis kernel on the word examples to obtain a $10,383 \times 10,383$ kernel matrix that serves as our starting point for constructing embeddings. Recall that this evaluation, developed in Carlin et al. (2011), assesses how well a representation distinguishes word types as measured by average precision (AP), which runs between 0 and 1, with 1 representing perfect performance. Performance on this task for this data set varies depends on many factors, e.g., choice of acoustic features, and better performance than reported here has been obtained, for example by changing the features in Chapter 2. The aim of this experiment is not to best that performance, but rather to examine how noise and occlusion influence performance for a given set of observations.

3.4.2 Noise Conditions

We consider the effects of additive noise and occlusion both in isolation and in tandem on the quality of Laplacian eigenmaps embeddings.

Additive Noise. Given a kernel matrix $\mathscr{K} \in [0,1]^{n\times n}$, we produce a random symmetric matrix $K \in [0,1]^{n\times n}$ where $K_{ii} = 0$ for all $i \in [n]$, and $\{K_{ij}\}_{1\leq i<j\leq n}$ are independent with K_{ij} beta-distributed with $\mathbb{E}K_{ij} = \mathscr{K}_{ij}$. We constrain the expected value of beta-distributed K_{ij} in this way by fixing one of the two shape parameters

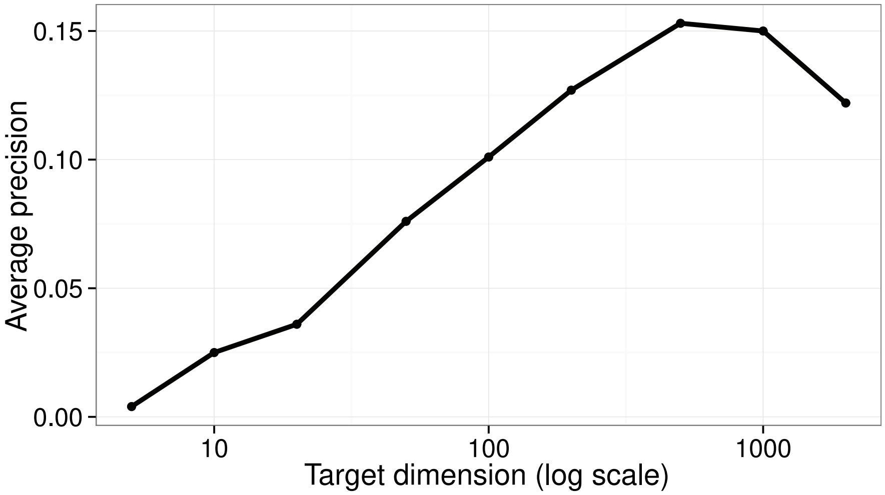
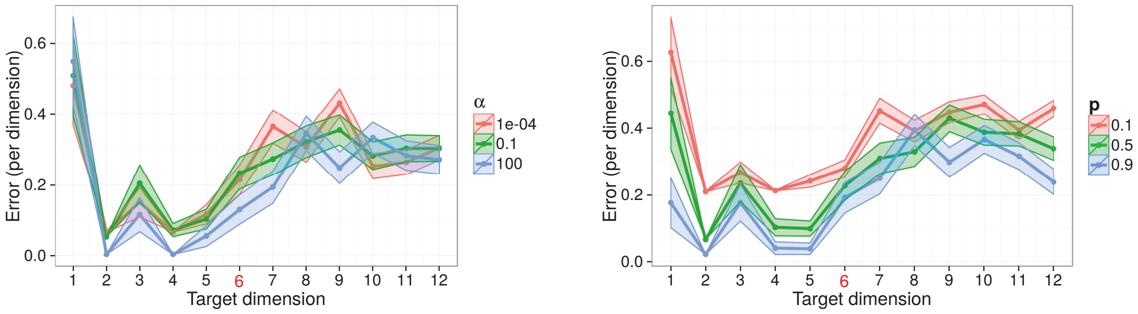


Figure 3.5: Performance on the speech task, measured by average precision, as a function of embedding dimension. We see that performance peaks at an embedding dimension of $d = 500$, with a severe degradation in the case where embedding dimension is chosen too small.

of the beta distribution, and varying the other to change the variance of the K_{ij}. In particular, $K_{ij} \sim \text{Beta}(\alpha_{ij}, \eta_{ij})$ with $\alpha_{ij} > 0$ and $\eta_{ij} > 0$. fixing $\eta_{ij} = \alpha_{ij}(1-\mathscr{K}_{ij})/\mathscr{K}_{ij}$ ensures that $\mathbb{E}K_{ij} = \mathscr{K}_{ij}$ with

$$\operatorname{Var} K_{ij} = \frac{\mathscr{K}_{ij}^2(1-\mathscr{K}_{ij})}{\alpha_{ij} + \mathscr{K}_{ij}},$$

so that we can vary our level of uncertainty on the K_{ij} variables by varying α_{ij}. We select a single global value $\alpha > 0$, and take $K_{ij} \sim \text{Beta}(\alpha, \alpha(1 - \mathscr{K}_{ij})/\mathscr{K}_{ij})$. In the limit $\alpha \to 0$, the K_{ij} are simply Bernoulli random variables with probability of success $p_{ij} = \mathscr{K}_{ij}$. In the limit $\alpha \to \infty$, we have $K_{ij} = \mathscr{K}_{ij}$ almost surely. Thus, we can think of our parameter α as a measure of the accuracy of our measurements of $\mathscr{K}$. We note also that our parameterization implies that the K_{ij} variables do not all have the same variance. Rather, variances are smaller for $\mathscr{K}_{ij}$ nearer to 0 and 1. As discussed in Section 3.1, this is a good model for applications in which the cases $\mathscr{K}_{ij} \approx 0$ and $\mathscr{K}_{ij} \approx 1$ are comparatively easy to handle from an estimation or computation standpoint, and the trouble arises from the cases where $\mathscr{K}_{ij} \approx 1/2$.



(a) **(b)**

Figure 3.6: Relative error in recovering the Laplacian eigenmaps embedding of the high-dimensional swiss roll as a function of dimension at (a) different values of fidelity parameter α and (b) different expected fractions of observed entries p. The true underlying dimension of the data is highlighted in red. Each data point is the mean of 50 independent trials, with error bars indicating one standard error. We see a pattern typical of model selection problems, in which the expressiveness of the model (i.e., higher embedding dimension) comes at the cost of increased variance (i.e., higher relative error in recovering the clean embedding).

Occlusion. We observe an occluded version of $\mathscr{K}$, where entries above the diagonal are observed independently with probability p. We proceed with our embedding using this sparse kernel matrix, with zeros in the unobserved entries.

Additive Noise with Occlusion. This condition combines the preceding two. We observe an occluded, noisy version of matrix $\mathscr{K}$. That is, we generate noisy matrix K from $\mathscr{K}$ with entries drawn independently from suitably chosen beta-distributions, then occlude K by independently observing entries with probability p.

Multiplicative and Biased Errors with Occlusion. Rather than the unbiased additive noise considered above, we consider how more complicated multiplicative and biased errors influence the quality of Laplacian eigenmaps embeddings. As discussed in Section 3.1, provided these errors are sufficiently well-behaved, we can adapt the results presented in this paper to make similar statements about this more general error model.

3.4.3 Effect of Noise and Occlusion on Embeddings

Our main theoretical result suggests that Laplacian eigenmaps embeddings should be robust to noise and occlusion. Fig. 3.2 shows how noise and occlusion influence the error in recovering the clean Laplacian eigenmaps embedding. Here, the target dimension is fixed at $d = d^* = 6$, while the noise and occlusion vary on the two axes. Each tile is the relative error averaged over 50 independent trials. We see that the clean Laplacian eigenmaps embedding is recovered with low error over a wide range of noise levels and occlusion rates, with performance degrading only when the fraction of observed entries goes below 0.25 in high-noise conditions.

Fig. 3.7 further illuminates the results seen in the synthetic data. Rather than looking at the relative error in recovering the clean embedding, we examine how noise and occlusion in the kernel matrix influence the down-stream speech task of distinguishing word types. The plot shows average precision as a function of both noise level and occlusion for three different embedding dimensions. We see that performance decays similarly in all three embedding dimensions, but that choice of embedding dimension has a large effect on overall performance. For example, comparing the $d = 100$ case with the $d = 500$ case, we see that both exhibit similar deterioration patterns with respect to noise level and expected fraction of observed entries, but the 500-dimensional embeddings out-perform the 100-dimensional ones when noise and occlusion are not so severe as to drown out the signal in the kernel matrix.

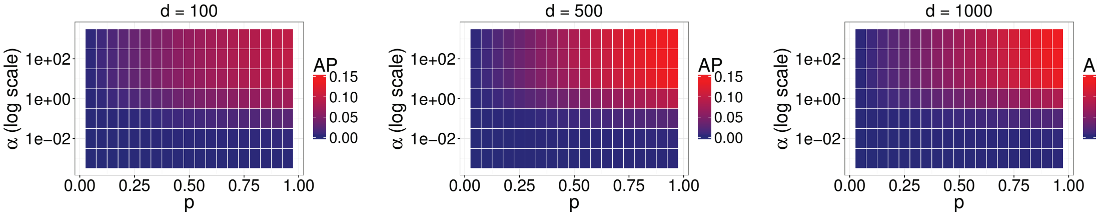


Figure 3.7: Average precision (AP) on the speech data set as a function of occlusion and noise level for different embedding dimensions d. Each tile is the mean of ten independent trials. We see that performance degrades similarly for all three target dimensions in the presence of noise and occlusion.

3.4.4 Effect of Multiplicative Error and Bias

Our theoretical results are for the case of unbiased noise, $\mathbb{E}K_{ij} = \mathscr{K}_{ij}$, and it is natural to ask whether similar results hold for a broader class of error models. As mentioned in Section 3.1, our results can be extended to biased errors ($\mathbb{E}K_{ij} \neq \mathscr{K}_{ij}$), provided those errors are suitably well-behaved. Fig. 3.3 and 3.4 lend experimental support to this point.

Using the same synthetic high-dimensional swiss roll setup as in Fig. 3.2, we consider biased noise, with K_{ij} beta distributed, but with $\mathbb{E}K_{ij} = \mathscr{K}_{ij} + b$, where $b \in \mathbb{R}$ is a bias, clipping $\mathscr{K}_{ij} + b$ to lie in $[0, 1]$ in the event that the bias b pushes $\mathscr{K}_{ij}$ out of its allowed range. Note that this corresponds to making K_{ij} either identically 0 or identically 1, according to whether $\mathscr{K}_{ij} + b$ is less than 0 or greater than 1, respectively. We again vary the parameter α as described above, but now the errors are biased away from $\mathscr{K}_{ij}$. Fig. 3.4 shows relative error in recovering the clean embeddings, again as a function of the parameters p and α, for four different levels of bias $b = -0.1, -0.01, -0.001, 0.001$. The first thing we notice is that performance is far more sensitive positive bias than it is to negative bias, with negative bias as large as -0.1 (a full one tenth of the dynamic range of the similarity measure) having comparatively little effect while a positive bias of just 0.001 results in notably worse relative error at all levels of noise and occlusion when compared to the unbiased errors in Fig. 3.2. This performance makes sense. Positive bias in our estimation of $\mathscr{K}$ results in us embedding a graph that looks highly connected, and the signal present in the comparatively sparse $\mathscr{K}$ is swamped. On the other hand, negative bias in our estimates only serves to further accentuate the few high-weighted observed entries, since only those entries for which $\mathscr{K}_{ij}$ is suitably far from 0 survive the bias. We have observed empirically that a similarly-motivated technique, in which small entries of the kernel matrix are clipped to 0, yields slight performance improvements in speech applications.

We further explore how general errors influence the quality of Laplacian eigenmaps embeddings by considering an error model in which

$$K_{ij} = \exp\{-D_{ij}^2/\sigma^2\}, \tag{3.5}$$

where $D_{ij} = d(x_i, x_j) + Z_{ij}$, and Z_{ij} is a one-dimensional normal random variable with mean 0 and variance ν^2. Thus, we have a distance measure corrupted by unbiased noise, corresponding to the common scenario in which the kernel function $\kappa(x, y)$ is a function of the distance between objects x and y and uncertainty lies in the measurement of that distance. The result, in the case of a nonlinear kernel function, is (typically) non-additive, biased, error, so that $\mathbb{E}K_{ij} \neq \mathscr{K}_{ij} = \kappa(x_i, x_j)$. We again use the same high-dimensional swiss roll as described above. We generate noisy versions of the kernel matrix $\mathscr{K}$, using the same Gaussian kernel with bandwidth $\sigma = 0.2$, but now noise takes the form described in Equation 3.5. Fig. 3.3 shows relative error in our recovery of the clean embeddings, as a function of the fraction of observed entries p and the variance ν^2 of the noise term Z_{ij}. We see that Laplacian eigenmaps embeddings are robust to fairly large amounts of uncertainty in the distance measurement. Indeed, we see that relative error is near zero for variance $\nu^2 \leq 1$, with the exception of particularly small values of p, when nearly all of the kernel matrix is occluded. This performance is impressive in light of the fact that $\nu^2 = 1$ corresponds to a standard deviation a full five times larger than the kernel bandwidth in these experiments.

3.4.5 Model Misspecification

Selecting the target dimension is of the utmost importance for good embeddings. Fig. 3.6 shows how embedding dimension interacts with noise and occlusion on the synthetic data. The two plots show that relative error in recovering the clean embedding is smaller at lower target dimensionalities, and this pattern holds over a wide range of noise levels and occlusion rates. In particular, we note that relative error in the presence of high noise and high occlusion remains comparable to the relative error in low noise and low occlusion conditions. Of course, this only tells part of the story. Fig. 3.5 shows average precision on the speech data set under clean conditions, as a function of embedding dimension. While a low-dimensional embedding performed under noise or occlusion might very closely resemble the corresponding clean embedding as in Fig. 3.6, Fig. 3.5 suggests that such an embedding would not yield satisfactory performance on downstream tasks such as classification. Indeed, we see here a pattern typical of model selection tasks: one must balance estimation error of model parameters against error in fitting the observed data (Shibata 1986; Fraley and Raftery 2002; Raftery and Dean 2006). The noisy embedding can only be as good as the clean embedding we are attempting to recover.

3.4.6 Effect of regularization

In the setting of the current work, when p is too small, we are in the sparse graph setting (Chaudhuri et al. 2012; Amini et al. 2013; Joseph and Yu 2014; Qin and Rohe 2013; Le et al. 2016), and it is natural to consider whether applying regularization might ease the deterioration of embedding quality in this regime. We follow the regularization procedure described in Le et al. (2016), in which a regularization parameter r is added to each entry of the observed matrix. That is, letting Y denote the occluded version of the noisy matrix K, we apply Laplacian eigenmaps to the matrix $[Y_{ij} + r]$ rather than Y itself. Our main theoretical results suggest that under suitable conditions, such an approach will be beneficial. The *C. elegans* brain graph is extremely sparse, and occlusion makes this sparsity still more dramatic. Fig. 3.8 shows how regularization influences downstream performance on the *C. elegans* data under different levels of occlusion. We see that when r is chosen too small, regularization is not enough to significantly change the learned embedding. Similarly, when r is chosen too large, regularization overpowers the signal present in the occluded matrix. However, with the *C. elegans* data, we see that there exists a level ($r \approx 0.01$) at which regularization greatly improves ARI, even when only half of the edges of the graph are known. We note that embeddings produced by the regularization procedure described in Qin and Rohe (2013) resulted in nearly identical performance.

The performance seen here is especially exciting from the neuroscience standpoint– these results suggest that we can recover structural and functional information in

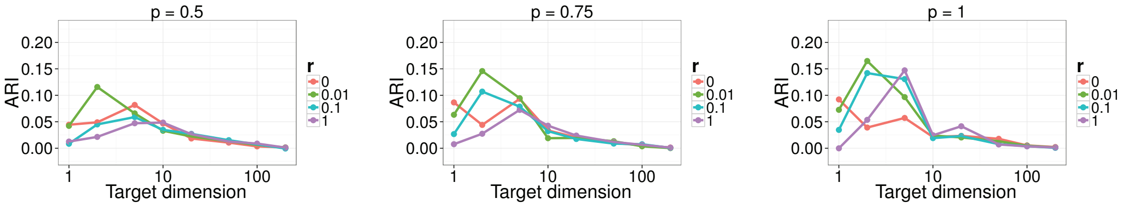
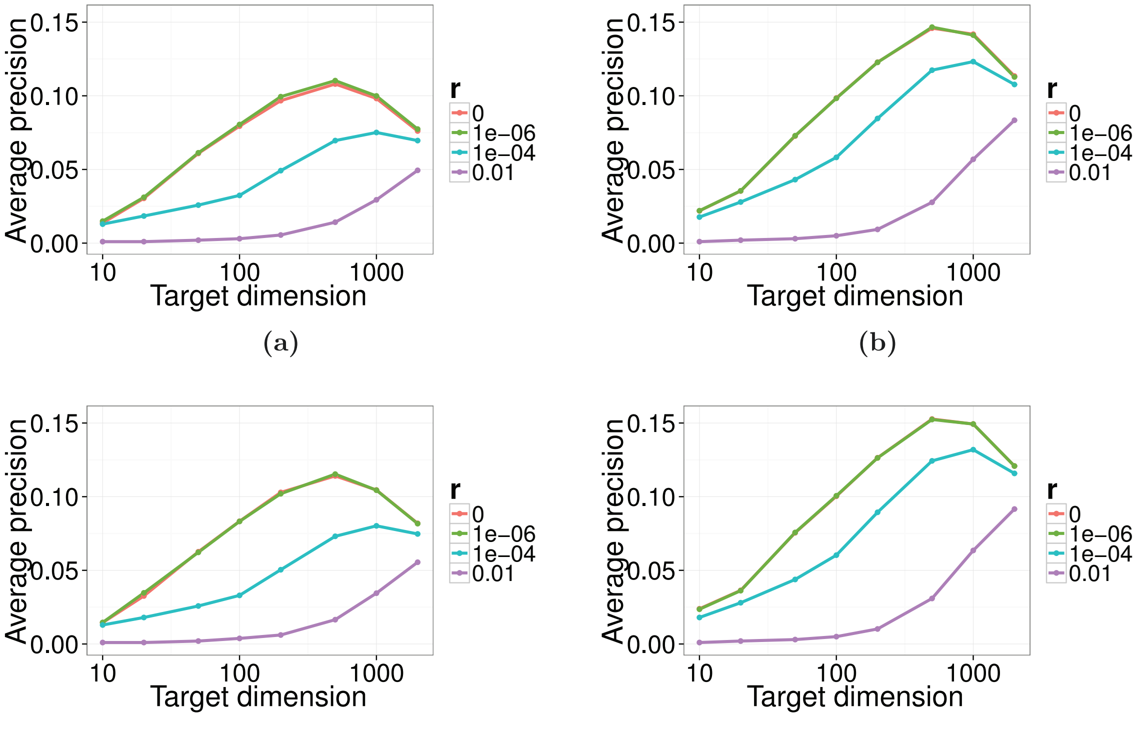


Figure 3.8: Adjusted Rand index (ARI) on the *C. elegans* data set for different levels of regularization as a function of dimension at different values of p, the expected fraction of observed entries. Each data point is the mean of 50 independent trials. We see that regularization enables us to accurately cluster the neurons even when much of the structure of the brain graph is occluded, with performance consistently superior to that obtained without regularization.

connectome data even when accurate assessment of all possible neural connections is impossible. We note the similarity of this phenomenon to that explored in Priebe et al. (2014), where the authors considered graph inference in the setting where one can trade the accuracy of edge assessment against the number of edges assessed. Of course, the usefulness of this result requires that can determine an appropriate value for r for a given data set, a problem that we leave for future work.

We close by illustrating conditions under which regularization does not appear to be a benefit. One would think, initially, and especially given the improvement seen in the *C. elegans* data, that regularization would yield similar gains in our speech task. Fig. 3.9 shows how regularization influences downstream performance on the speech task. We see that regularization does not appear to confer the benefit seen in the *C. elegans* data. Crucially, however, moderate amounts of regularization do not appear have any adverse effects on average precision. One possible explanation for this phenomenon comes from the fact that the kernel bandwidth used in Levin et al. (2013) was chosen so as to give the best possible average precision on precisely the task we are using for evaluation. That is, since the kernel bandwidth has already been tuned so as to yield high-quality embeddings, regularization can do little to improve the embeddings. But this explanation does not account for the fact that regularization does not appear to confer any protection against occlusion and noise in the kernel matrix. It is possible that the speech data set is such that the kernel matrix is sparse enough that regularization does nothing to pull us toward a better embedding. We leave further exploration of this phenomenon to future work.



(a)

(b)

(c)

(d)

Figure 3.9: Average precision on the speech data set as a function of embedding dimension for different levels of regularization under varying amounts of noise and occlusion: (a) $\alpha = 10, p = 0.7$, (b) $\alpha = 10, p = 1.0$, (c) $\alpha = 100, p = 0.7$, (d) $\alpha = 100, p = 1.0$. Each data point is the mean of 10 independent trials. We see that while regularization does not provide the stunning improvement that it does on the *C. elegans* graph, moderate regularization at least does not noticeably harm average precision.

3.5 Discussion

We have presented an analysis of the concentration of the graph Laplacian of certain kernel matrices under occlusion and noise. Crucial to our bound was the presence of a certain structure in the kernel matrix that ensures concentration of the row-sums. Experiments on both synthetic and real data show that a concentration phenomenon similar to that predicted by the theory is present, and has effects both on performance in downstream tasks and on the model selection problem. We close by briefly mentioning some directions for future work.

3.5.1 Adaptive Techniques

The regularization used here was applied uniformly to every vertex of the graph, but regularization is only required to control the high variance associated with small-degree nodes. In light of this, one might consider regularization techniques that apply only to nodes that require it. It is unclear a priori whether such an approach would be advantageous, since regularization does little to change the behavior of high-degree nodes. However, it stands to reason that a well-designed adaptive technique might enable convergence of the regularized estimate to the true expected graph, rather than to its regularized counterpart as in the current work. For example, if only a small fraction of the nodes in a given graph require regularization, then the Frobenius error between the regularized and non-regularized Laplacians can still go to zero even if r

goes to zero slowly.

In a similar vein, it stands to reason that a technique that evaluates entries of the kernel matrix adaptively rather than the edge-independent occlusion model considered here might achieve more accurate recovery of the clean embeddings.

3.5.2 Other Error Models

The noise model we have considered is additive, unbiased and entry-wise independent. As discussed in Section 3.1, our results can be (approximately) extended to multiplicative, biased noise models, at least for certain kernels. However, the concentration bounds we have used require a certain independence structure. As such, it seems likely that novel techniques will be required to handle entry-wise dependent noise and occlusion in the kernel matrix. For example, the techniques in O'Rourke et al. (2016b) might be brought to bear, except that they require structural assumptions on $\mathscr{K}$ that seem unlikely to hold for a non-linear kernel function.

3.5.3 Graph Construction

We have largely ignored the problem of constructing the k-NN or ϵ-graph, the first step in Laplacian eigenmaps and spectral clustering. Rather than using either of these constructions, we have relied on the fact that the kernel matrix can be made to resemble these graphs by using, for example, a Gaussian kernel. We believe that the our analysis can be extended to many of these constructions simply by taking advantage of this resemblance. We leave this extension for future work.

3.5.4 Other Dimensionality Reduction Techniques

To what extent are different embedding techniques robust to uncertainty in similarity measures (as opposed to errors on the observations themselves)? To the best of our knowledge, MDS and Laplacian eigenmaps remain the only techniques for which such questions have been explored. We believe that analyses similar to that pursued in the current work should apply to other dimensionality reduction techniques. Indeed, given the results in Yan et al. (2007), it would be a surprise to learn that no such general result is possible.

As alluded to in Section 3.2, a natural approach to the problem considered in this paper would be to apply Chatterjee's universal singular value thresholding (USVT; Chatterjee 2015) to the occluded, noisy kernel matrix Y (or, in the case where $\kappa(x, y)$ is a function of $d(x, y)$, to transform Y into an occluded matrix of distances D, impute the missing entries of D using USVT, and reapply the kernel function to obtain an estimate of $\mathscr{K}$). Applying USVT in this manner to the speech task considered in Section 3.4 yields results essentially identical to those reported using Y alone at all noise and occlusion rates. Indeed, USVT performed remarkably similarly to our method on all three data sets, a fact that warrants further exploration.

Some well-known dimensionality reduction techniques can be adapted fairly easily to the model in Equation (3.2) by using Chatterjee's USVT to impute the missing entries of Y and proceeding apace. In an experimental setup identical to the synthetic high-dimensional swiss roll experiments presented in Section 3.4, we explored the effect of noise and occlusion on both MDS and kernel PCA (KPCA). We found that neither of these methods compared favorably to the results seen for Laplacian eigenmaps. While direct comparison of the relative errors for these three different methods is not possible (e.g., embeddings produced by MDS are not constrained in the same way that Laplacian eigenmaps embeddings are), from a qualitative standpoint, MDS and KPCA both degraded much more severely in the presence of noise and occlusion when compared with Fig. 3.2. While a direct comparison (experimental or otherwise) of Laplacian eigenmaps with other dimensionality reduction techniques is not the focus of this paper, a more thorough exploration of how different methods fare in the presence of noise and occlusion (and how those methods might be adapted to lessen the impact of uncertainty) warrants additional work in the future.

3.6 Proof details

In what follows, we suppress dependence on n for ease of notation. We remind the reader that all quantities involved, including the parameters r and p all implicitly depend on n. We let $\widehat{Y} = Y + rJ$ denote the regularized version of matrix Y, and define $\widehat{D}$ to be the corresponding degree matrix, so that $\widehat{D}_{ii} = nr + \sum_{j=1}^{n} Y_{ij}$. Denote the regularized version of $p\mathscr{K}$ by $\widehat{\mathscr{K}} = p\mathscr{K} + rJ$, with $\widehat{\mathscr{D}}$ the corresponding degree matrix, $\widehat{\mathscr{D}}_{ii} = nr + \sum_{j=1}^{n} p\mathscr{K}_{ij}$.

Throughout, $C > 0$ denotes a constant (independent of n), which may change from line to line or from one lemma to another. β and γ denote quantities (both depending on n) that will control convergence of the node degrees and the Frobenius norm in Theorem 3, respectively. We will see that the constraints on β and γ required for our concentration bounds are such that when we plug in $\gamma = C'n^{-1/2}r^{-1}\log^{1/2} n$ and $\beta = C''n^{-1/2}r^{-1/2}\log^{1/2} n$ for suitably chosen constants $C', C'' > 0$, we obtain the bound claimed in Theorem 3. We will require that $\beta \to 0$ as $n \to \infty$, i.e., that $r = \omega(n^{-1}\log n)$.

We first establish that with high probability, the row sums of $\widehat{Y}$ concentrate about their expected value.

Lemma 2. *Suppose that there exists constant $c_1 > 0$ such that for all suitably large n we have*

$$\frac{\beta^2 r}{1+\beta} \geq c_1 \frac{\log n}{n}. \tag{3.6}$$

Then for all suitably large n, with probability at least n^{1-c_1}, it holds for all $i \in [n]$ that $|\widehat{D}_{ii} - \widehat{\mathscr{D}}_{ii}| \leq \beta\widehat{\mathscr{D}}_{ii}$.

Proof. Fix $i \in [n]$. By definition,

$$\widehat{D}_{ii} - \widehat{\mathscr{D}}_{ii} = \sum_{j=1}^{n}(Y_{ij} + r) - (p\mathscr{K}_{ij} + r) = \sum_{j=1}^{n} Y_{ij} - p\mathscr{K}_{ij},$$

and $\mathbb{E}Y_{ij} = p\mathscr{K}_{ij}$. By a standard Chernoff-style bound (Chung and Lu 2006),

$$\Pr\left[|\widehat{D}_{ii} - \widehat{\mathscr{D}}_{ii}| \geq \beta\widehat{\mathscr{D}}_{ii}\right] \leq 2\exp\left\{\frac{-3\beta^2\widehat{\mathscr{D}}_{ii}^2}{6V + 2\beta\widehat{\mathscr{D}}_{ii}}\right\},$$

where $V = \sum_{j=1}^n \mathbb{E}Y_{ij}^2$. Since

$$V = \sum_{j=1}^n p\mathbb{E}K_{ij}^2 \leq p\sum_{j=1}^n \mathscr{K}_{ij} \leq \widehat{\mathscr{D}}_{ii},$$

we have

$$\Pr\left[|\widehat{D}_{ii} - \widehat{\mathscr{D}}_{ii}| \geq \beta\widehat{\mathscr{D}}_{ii}\right] \leq 2\exp\left\{\frac{-C\beta^2}{1+\beta}\widehat{\mathscr{D}}_{ii}\right\},$$

where $C > 0$ is a constant. Since $\widehat{\mathscr{D}}_{ii} \geq nr$ by virtue of regularization, our assumption in (3.6) ensures that

$$\Pr\left[|\widehat{D}_{ii} - \widehat{\mathscr{D}}_{ii}| \geq \beta\widehat{\mathscr{D}}_{ii}\right] \leq n^{-c_1}.$$

Applying the union bound over all $i \in [n]$ yields the result. □

Lemma 3. *Suppose that γ depends on n in such a way that there exist constants $C', C'' > 0$ so that for suitably large n,*

$$C'\gamma^2 \geq \frac{16}{n^2r^3} + \frac{16}{n^2} \tag{3.7}$$

and

$$\gamma \geq C''\frac{\log^{1/2} n}{n^{3/2}r^2}. \tag{3.8}$$

Then there exists a constant $c_2 > 0$ such that with probability at least $1 - n^{-c_2}$, we have

$$\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2})^2}{\widehat{\mathscr{D}}_{ii}^2 \widehat{\mathscr{D}}_{kk}^2} \leq C\gamma^2,$$

where $C > 0$ is a constant.

Proof. For ease of notation, define

$$X_{ik} = \frac{\left(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2}\right)^2}{\widehat{\mathscr{D}}_{ii}^2 \widehat{\mathscr{D}}_{kk}^2}.$$

We will bound $\Pr\left[\sum_{i,k} X_{ik} - \mathbb{E}\sum_{i,k} X_{ik} \geq \gamma^2\right]$ and show $\mathbb{E}\sum_{i,k} X_{ik} \leq C'\gamma^2$, implying that $\Pr\left[\sum_{i,k} X_{ik} \geq C\gamma^2\right]$.

A standard Chernoff-style bound lets us write

$$\Pr\left[\sum_{i,k} X_{ik} \geq \gamma^2 + \mathbb{E}\sum_{i,k} X_{ik}\right] \leq \exp\left\{\frac{-3\gamma^4}{6V + 2\gamma^2 M}\right\},$$

where

$$V = \sum_{i,k} \mathbb{E} X_{ik}^2 = \sum_{i,k} \frac{\mathbb{E}\left(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2}\right)^4}{\widehat{\mathscr{D}}_{ii}^4 \widehat{\mathscr{D}}_{kk}^4},$$

$$\text{and } M = \max\left\{1/(\widehat{\mathscr{D}}_{ii}^2 \widehat{\mathscr{D}}_{kk}^2) : i,k \in [n]\right\}.$$

Bounding $V \leq n^{-6} r^{-8}$ and $M \leq (nr)^{-4}$,

$$\Pr\left[\sum_{i,k} X_{ik} \geq \gamma^2 + \mathbb{E}\sum_{i,k} X_{ik}\right] \leq \exp\left\{\frac{-3(\gamma n r)^4}{6n^{-2} r^{-4} + 2\gamma^2}\right\},$$

and using our assumption in (3.8) to lower bound the denominator inside the exponent by $\Omega(n\gamma^2)$, we can guarantee the existence of a constant $c_2 > 0$ such that

$$\Pr\left[\sum_{i,k} X_{ik} \geq \gamma^2 + \mathbb{E}\sum_{i,k} X_{ik}\right] \leq n^{-c_2}.$$

It remains for us to show that $\mathbb{E}\sum_{i,k} X_{ik} \leq C'\gamma^2$. We have

$$\begin{aligned} \mathbb{E}\sum_{i=1}^{n}\sum_{k=1}^{n} X_{ik} &\leq \sum_{i=1}^{n}\sum_{k=1}^{n} \frac{\mathbb{E}\left(\widehat{Y}_{ik}^4 + \widehat{\mathscr{K}_{ik}^4}\right)}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2} \\ &\leq \sum_{i=1}^{n}\sum_{k=1}^{n} \frac{8\left(p\mathbb{E}K_{ik}^4 + r^4\right) + \widehat{\mathscr{K}_{ik}^4}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2}, \end{aligned} \tag{3.9}$$

where we have used the fact that $(a+b)^2 \leq 2a^2 + 2b^2$ for all $a, b \in \mathbb{R}$. Since $\widehat{\mathscr{D}}_{ii} \geq nr$ for all $i \in [n]$, we have

$$\sum_{i=1}^{n} \frac{1}{\widehat{\mathscr{D}}_{ii}} \leq \frac{1}{r} \text{ and } \sum_{i=1}^{n}\sum_{k=1}^{n} \frac{r^4}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2} \leq \frac{1}{n^2}. \tag{3.10}$$

Noting that $\mathbb{E}K_{ik}^4 \leq \mathbb{E}K_{ik} = \mathscr{K}_{ik}$ and applying (3.10), we have

$$\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{p\mathbb{E}K_{ik}^4}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2} \leq \sum_{i=1}^{n} \frac{1}{\widehat{\mathscr{D}}_{ii} n^2 r^2} \leq \frac{1}{n^2 r^3}. \tag{3.11}$$

Recalling that $\widehat{\mathscr{K}_{ik}} = p\mathscr{K}_{ik}+r$ by definition and applying the definition of $\widehat{\mathscr{D}}_{ii}$, (3.10) implies

$$\begin{aligned}\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{\widehat{\mathscr{K}_{ik}^{4}}}{\widehat{\mathscr{D}}_{ii}^{2}\widehat{\mathscr{D}}_{kk}^{2}} &\le 8\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{p^{4}\mathscr{K}_{ik}^{4}+r^{4}}{\widehat{\mathscr{D}}_{ii}^{2}\widehat{\mathscr{D}}_{kk}^{2}}\\ &\le \frac{8p^{3}}{n^{2}r^{2}}\sum_{i=1}^{n}\frac{1}{\widehat{\mathscr{D}}_{ii}}+8\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{r^{4}}{\widehat{\mathscr{D}}_{ii}^{2}\widehat{\mathscr{D}}_{kk}^{2}}\\ &\le \frac{8p^{3}}{n^{2}r^{3}}+\frac{8}{n^{2}}.\end{aligned}$$

Combining this with (3.9) and (3.11) and applying (3.7) completes the proof. □

Lemma 4. *Under the same conditions as Lemma 2, and assuming there exists a constant $C > 0$ such that*

$$C\gamma^{2} \ge \frac{\beta^{2}}{nr^{2}}, \tag{3.12}$$

with probability at least n^{1-c_1}, we have

$$\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{\ell=1}^{n}\frac{(\widehat{Y}_{ik}^{2}-\widehat{\mathscr{K}_{ik}^{2}})(\widehat{Y}_{i\ell}^{2}-\widehat{\mathscr{K}_{i\ell}^{2}})}{\widehat{\mathscr{D}}_{ii}^{2}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \le C\gamma^{2}.$$

Proof. Observing that $\widehat{Y}_{ik}+\widehat{\mathscr{K}_{ik}} \le 1+p+2r$,

$$\begin{aligned}&\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{\ell=1}^{n}\frac{(\widehat{Y}_{ik}^{2}-\widehat{\mathscr{K}_{ik}^{2}})(\widehat{Y}_{i\ell}^{2}-\widehat{\mathscr{K}_{i\ell}^{2}})}{\widehat{\mathscr{D}}_{ii}^{2}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}}\\ &\qquad\le \frac{(1+p+2r)^{2}}{n^{2}r^{2}}\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{\ell=1}^{n}\frac{(\widehat{Y}_{ik}-\widehat{\mathscr{K}_{ik}})(\widehat{Y}_{i\ell}-\widehat{\mathscr{K}_{i\ell}})}{\widehat{\mathscr{D}}_{ii}^{2}}.\end{aligned}$$

By Lemma 2, with probability at least $1-n^{1-c_1}$, it holds for all $i \in [n]$ that

$$\left|\sum_{k=1}^{n}\widehat{Y}_{ik}-\widehat{\mathscr{K}_{ik}}\right| \le \beta\widehat{\mathscr{D}}_{ii},$$

and hence, since $p, r \in [0,1]$ and $\widehat{\mathscr{D}}_{ii} \geq nr$,

$$\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{\ell=1}^{n} \frac{(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2})(\widehat{Y}_{i\ell}^2 - \widehat{\mathscr{K}_{i\ell}^2})}{\widehat{\mathscr{D}}_{ii}^2 \widehat{\mathscr{D}}_{kk} \widehat{\mathscr{D}}_{\ell\ell}} \leq \frac{16\beta^2}{nr^2}.$$

Our assumption in (3.12) yields the desired result. □

Lemma 5.

$$\sum_{i,j,k,\ell} \frac{p^4 \mathscr{K}_{ik}\mathscr{K}_{jk}\mathscr{K}_{i\ell}\mathscr{K}_{j\ell}}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \leq \frac{p}{r}.$$

Proof. Using the following facts:

(i) $\widehat{\mathscr{D}}_{ii} \geq rn$ for all $i \in [n]$,

(ii) $\mathscr{K}_{ik} \in [0,1]$ for all $i, j \in [n]$,

(iii) $\sum_{k=1}^{n} p\mathscr{K}_{ik} \leq \widehat{\mathscr{D}}_{ii}$ for all $i \in [n]$,

we have

$$\begin{aligned}\sum_{i,j,k,\ell} \frac{p^4 \mathscr{K}_{ik}\mathscr{K}_{jk}\mathscr{K}_{i\ell}\mathscr{K}_{j\ell}}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} &\leq \frac{p}{nr} \sum_{i,j,k} \frac{p^2 \mathscr{K}_{ik}\mathscr{K}_{jk}}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}} \sum_{\ell=1}^{n} p\mathscr{K}_{j\ell} \\ &\leq \frac{p}{nr} \sum_{i,j,k} \frac{p^2 \mathscr{K}_{ik}\mathscr{K}_{jk}}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{kk}} \leq \frac{p}{r}.\end{aligned}$$

□

Lemma 6. *For ease of notation, let*

$$X_{ijk\ell} = \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \tag{3.13}$$

and define $T = \{(i,j,k,\ell) : i,j,k,\ell \in [n]$ *distinct.*$\}$. *There exists a constant* $C > 0$ *such that*

$$\sum_{(i,j,k,\ell)\in T} \operatorname{Var} X_{ijk\ell} \leq \frac{C}{n^4 r^5}.$$

Proof. Since i,j,k,ℓ are distinct for each $(i,j,k,\ell) \in T$,

$$\begin{aligned}\operatorname{Var} X_{ijk\ell} &= \mathbb{E} X_{ijk\ell}^2 \\ &= d_{ijk\ell}^{-2} \mathbb{E}\left[\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}}\right]^2 \mathbb{E}\left[\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}}\right]^2,\end{aligned}$$

where $d_{ijk\ell} = \widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}$. Expanding $\widehat{Y}_{ik} = Y_{ik} + r$ and $\widehat{\mathscr{K}_{ik}} = p\mathscr{K}_{ik} + r$ and using linearity of expectation, we have

$$\begin{aligned}&\mathbb{E}\left[\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}}\right]^2 \\ &\quad= \mathbb{E}\big[Y_{ik}Y_{jk} - p^2\mathscr{K}_{ik}\mathscr{K}_{jk} \\ &\qquad\qquad + r(Y_{ik} - p\mathscr{K}_{ik}) + r(Y_{jk} - p\mathscr{K}_{jk})\big]^2 \\ &\quad= \operatorname{Var} Y_{ik}Y_{jk} \\ &\qquad + r(r + 2p\mathscr{K}_{jk}) \operatorname{Var} Y_{ik} + r(r + 2p\mathscr{K}_{ik}) \operatorname{Var} Y_{jk}.\end{aligned}$$

For ease of notation, define

$$Q_{ijk} = p^2\mathscr{K}_{ik}\mathscr{K}_{jk} + r(r+2p)p\mathscr{K}_{ik} + r(r+2p)p\mathscr{K}_{jk}.$$

The Bhatia-Davis inequality (Bhatia and Davis 2000) states that if a random variable

Z satisfies $\Pr[m \leq Z \leq M] = 1$, then $\operatorname{Var} Z \leq (\mathbb{E}Z - m)(M - \mathbb{E}Z)$. Since $\mathscr{K}_{ik} \in [0,1]$ for all $i, k \in [n]$, we have $\operatorname{Var} Y_{ik}Y_{jk} \leq p^2 \mathscr{K}_{ik}\mathscr{K}_{jk}$ and $\operatorname{Var} Y_{ik} \leq p\mathscr{K}_{ik}$, and hence

$$\mathbb{E}\left[\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}}\right]^2 \leq Q_{ijk}.$$

Combining this with (3.13), we have

$$\operatorname{Var} X_{ijk\ell} \leq d_{ijk\ell}^{-2} Q_{ijk} Q_{ij\ell}.$$

Summing, we have

$$\begin{aligned}
\sum_{(i,j,k,\ell) \in T} \operatorname{Var} X_{ijk\ell} &\leq \sum_{(i,j,k,\ell) \in T} d_{ijk\ell}^{-2} Q_{ijk} Q_{ij\ell} \\
&= \sum_{(i,j,k,\ell) \in T} d_{ijk\ell}^{-2} p^4 \mathscr{K}_{ik}\mathscr{K}_{jk}\mathscr{K}_{i\ell}\mathscr{K}_{j\ell} \\
&\quad + 4 \sum_{(i,j,k,\ell) \in T} d_{ijk\ell}^{-2} r(r+2p)p^3 \mathscr{K}_{ik}\mathscr{K}_{jk}\mathscr{K}_{j\ell} \\
&\quad + 2 \sum_{(i,j,k,\ell) \in T} d_{ijk\ell}^{-2} r^2(r+2p)^2 p^2 \mathscr{K}_{ik}\mathscr{K}_{jk} \\
&\quad + 2 \sum_{(i,j,k,\ell) \in T} d_{ijk\ell}^{-2} r^2(r+2p)^2 p^2 \mathscr{K}_{ik}\mathscr{K}_{j\ell} \\
&\leq \frac{p}{n^4 r^5} + 4\frac{(r+2p)}{n^4 r^4} + 4\frac{(r+2p)^2}{n^4 r^4},
\end{aligned}$$

where we have used $\widehat{\mathscr{D}}_{ii} \geq nr$ along with Lemma 5 to bound the first sum after the equality, and the other sums are bounded using reasoning nearly identical to that in the proof of Lemma 5. The result then follows from $r, p \in [0,1]$. □

Lemma 7. *There exists a constant $C > 0$ such that*

$$\sum_{\{(i,j,k,\ell),(i',j',k',\ell')\} \in \binom{T}{2}} \operatorname{Cov}\left(X_{ijk\ell}, X_{i'k'j'\ell'}\right) \leq \frac{C}{n^3 r^4}.$$

Proof. Recall that

$$X_{ijk\ell} = \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}}.$$

Consider first the situation where (a,b,c,d) is a permutation of (i,j,k,ℓ). Call this permutation $\sigma \in S_4$. σ is not the identity permutation, but σ may be such that $X_{ijk\ell} = X_{abcd}$ as happens when, for example, $i = a, j = b, k = d, \ell = c$. By symmetry, it suffices to consider three cases. **Case 1: $\{i,j\} = \{a,b\}$.** In this case, we can assume without loss of generality (by symmetry) that $i = b$, $j = a$, $k = d$ and $\ell = c$, so that

$$\begin{aligned}\mathbb{E}X_{ijk\ell}X_{abcd} &= \frac{\mathbb{E}\left[(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})^2(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})^2\right]}{\widehat{\mathscr{D}}^2_{ii}\widehat{\mathscr{D}}^2_{jj}\widehat{\mathscr{D}}^2_{kk}\widehat{\mathscr{D}}^2_{\ell\ell}} \\ &= \frac{\operatorname{Var}\widehat{Y}_{ik}\widehat{Y}_{jk}\operatorname{Var}\widehat{Y}_{i\ell}\widehat{Y}_{j\ell}}{\widehat{\mathscr{D}}^2_{ii}\widehat{\mathscr{D}}^2_{jj}\widehat{\mathscr{D}}^2_{kk}\widehat{\mathscr{D}}^2_{\ell\ell}} \leq \frac{(1+r)^4\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}}}{\widehat{\mathscr{D}}^2_{ii}\widehat{\mathscr{D}}^2_{jj}\widehat{\mathscr{D}}^2_{kk}\widehat{\mathscr{D}}^2_{\ell\ell}},\end{aligned}$$

where the last inequality follows from the Bhatia-Davis inequality and the fact that $0 \leq \widehat{Y}_{ik} \leq 1 + r$.

Case 2: $\{i,j\} = \{a,c\}$. Without loss of generality, assume that $i = a$, $j = c$,

$k = b$ and $\ell = d$. We have

$$\begin{aligned}\mathbb{E}X_{ijk\ell}X_{abcd} &= \frac{\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{ij}}\widehat{\mathscr{K}_{j\ell}}\widehat{\mathscr{K}_{k\ell}} \operatorname{Var}\widehat{Y}_{jk} \operatorname{Var}\widehat{Y}_{i\ell}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2} \\ &\leq \frac{(1+r)^2\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{j\ell}}\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{i\ell}}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2},\end{aligned}$$

where the inequality follows from the Bhatia-Davis inequality and the fact that $\widehat{\mathscr{K}_{ik}} \leq 1 + r$.

Case 3: $\{i, j\} = \{c, d\}$**.** Without loss of generality, assume that $i = c$, $j = d$, $k = a$ and $\ell = b$. Then

$$\begin{aligned}&\mathbb{E}X_{ijk\ell}X_{abcd} \\ &\quad= \frac{\mathbb{E}\widehat{Y}_{ik}\widehat{Y}_{j\ell}(\widehat{Y}_{jk}+\widehat{Y}_{i\ell})^2 - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{j\ell}}(\widehat{\mathscr{K}_{jk}}+\widehat{\mathscr{K}_{i\ell}})^2}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2} \\ &\quad= \frac{\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{j\ell}}\left(\operatorname{Var}\widehat{Y}_{jk} + \operatorname{Var}\widehat{Y}_{i\ell}\right)}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2}.\end{aligned}$$

Letting $(i, j, k, \ell) \sim (a, b, c, d)$ denote the fact that (a, b, c, d) is a permutation of

(i, j, k, ℓ), we can bound the sum of the covariances under consideration by

$$\begin{aligned}\sum_{(i,j,k,\ell)\in T} \sum_{(a,b,c,d)\sim(i,j,k,\ell)} &\operatorname{Cov}\left(X_{ijk\ell}, X_{abcd}\right) \\ &\leq 2C(1+r)^4 \sum_{i,j,k,\ell} \frac{\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2} \\ &\quad + 2C(1+r) \sum_{(i,j,k,\ell)\in T} \frac{\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{j\ell}}\widehat{\mathscr{K}_{jk}}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{jj}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2} \\ &\leq \frac{C(1+r)^5 + 2C(1+r)}{n^4 r^5},\end{aligned} \tag{3.14}$$

Now, consider the situation where (i, j, k, ℓ) is not a permutation of (a, b, c, d). Clearly, if $\{i, j, k, \ell\} \cap \{a, b, c, d\} = \emptyset$, then $\operatorname{Cov}(X_{ijk\ell}, X_{abcd}) = 0$. Indeed, $\operatorname{Cov}(X_{ijk\ell}, X_{abcd}) \neq 0$ requires that each term of the form $(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})$ be dependent on one of the other three such terms in $X_{ijkl}X_{abcd}$, since otherwise a term of the form $\mathbb{E}(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})$ factors out and the covariance is zero. Indeed, only one other choice (up to permutations of the indices) of (i, j, k, ℓ) and (a, b, c, d) gives rise to a non-zero covariance, namely $\mathbb{E}X_{ijk\ell}X_{ibk\ell}$. By symmetry, to handle the terms of this form, it will suffice for us to bound

$$\sum_{(i,j,k,\ell)\in T} \sum_{b\notin\{i,j,k,\ell\}} \operatorname{Cov}(X_{ijk\ell}, X_{ibk\ell}).$$

Using the fact that $\operatorname{Var}\widehat{Y}_{ik} \leq \widehat{\mathscr{K}_{ik}}$ by the Bhatia-Davis inequality, and applying reasoning similar to that in Lemma 5,

$$\begin{aligned}
&\sum_{(i,j,k,\ell)\in T}\sum_{b\notin\{i,j,k,\ell\}} \operatorname{Cov}(X_{ijk\ell}, X_{ibk\ell}) \\
&= \sum_{(i,j,k,\ell)\in T}\sum_{b\notin\{i,j,k,\ell\}} \frac{\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{bk}}\widehat{\mathscr{K}_{j\ell}}\widehat{\mathscr{K}_{b\ell}} \operatorname{Var}\widehat{Y}_{ik} \operatorname{Var}\widehat{Y}_{i\ell}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{bb}} \\
&\leq \sum_{(i,j,k,\ell)\in T}\sum_{b\notin\{i,j,k,\ell\}} \frac{\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{bk}}\widehat{\mathscr{K}_{j\ell}}\widehat{\mathscr{K}_{b\ell}}\widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{i\ell}}}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2\widehat{\mathscr{D}}_{\ell\ell}^2\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{bb}} \\
&\leq \frac{(1+r)^2}{(nr)^4} \sum_{i,j,k\in[n] \text{ distinct}} \frac{\widehat{\mathscr{K}_{jk}}\widehat{\mathscr{K}_{ik}}}{\widehat{\mathscr{D}}_{kk}^2} \leq \frac{(1+r)^2}{n^3r^4}.
\end{aligned}$$

Combining this with (3.14) and noting that $r > n^{-1}$ implies $(n^3r^4)^{-1} \geq (n^4r^5)^{-1}$, we have our result. □

Lemma 8. *Let $T = \{(i,j,k,\ell) : i,j,k,\ell \in [n]$ distinct.$\}$. For each $(i,j,k,\ell) \in T$, define variable*

$$X_{ijk\ell} = \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}}.$$

There exist constants $C, C_\gamma > 0$ such that with probability at least $1 - C_\gamma(\gamma^4n^3r^4)^{-1}$,

$$\sum_{(i,j,k,\ell)\in T} \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \leq C\gamma^2. \tag{3.15}$$

Proof. By Chebyshev's inequality,

$$\Pr\left[\sum_{(i,j,k,\ell)\in T} X_{ijk\ell} \geq C\gamma^2\right] \leq \frac{\operatorname{Var}\sum_{(i,j,k,\ell)\in T} X_{ijk\ell}}{C^2\gamma^4}.$$

We have

$$\begin{aligned}\operatorname{Var} \sum_{(i,j,k,\ell)\in T} X_{ijk\ell} \\ &= \sum_{(i,j,k,\ell)\in T} \operatorname{Var} X_{ijk\ell} \\ &\quad + \sum_{\{(i,j,k,\ell),(i',j',k',\ell')\}\in\binom{T}{2}} \operatorname{Cov}\left(X_{ijk\ell}, X_{i'k'j'\ell'}\right).\end{aligned}$$

Lemma 6 bounds the first of these two sums by

$$\sum_{(i,j,k,\ell)\in T} \operatorname{Var} X_{ijk\ell} \leq \frac{C'}{n^4r^5},$$

where $C' > 0$ is a constant, and Lemma 7 ensures that

$$\sum_{\{(i,j,k,\ell),(i',j',k',\ell')\}\in\binom{T}{2}} \operatorname{Cov}\left(X_{ijk\ell}, X_{i'k'j'\ell'}\right) \leq \frac{C''}{n^3r^4}$$

for some constant $C'' > 0$. Since $(n^4r^5)^{-1} \leq (n^3r^4)^{-1}$ for $r > 1/n$, we have

$$\Pr\left[\sum_{(i,j,k,\ell)\in T} X_{ijk\ell} \geq C\gamma^2\right] \leq \frac{C'+C''}{C\gamma^4n^3r^4}.$$

Choosing $C_\gamma = (C'+C'')/C$ yields the result. □

Lemma 9. *Under the conditions of the above lemmata, there exist constants $c, C > 0$ such that for all suitably large n, with probability at least $1-3n^{-c}$, we have*

$$\|\widehat{\mathscr{L}}\widehat{\mathscr{L}} - (\widehat{\mathscr{D}}^{-1/2}\widehat{Y}\widehat{\mathscr{D}}^{-1/2})^2\|_F \leq C\gamma.$$

Proof. Expanding the sum and recalling our earlier definition of $T = \{(i,j,k,\ell) : i,j,k,\ell \in [n] \text{ distinct.}\}$, we have

$$
\begin{aligned}
&\|\widehat{\mathscr{L}}\widehat{\mathscr{L}} - (\widehat{\mathscr{D}}^{-1/2}\widehat{Y}\widehat{\mathscr{D}}^{-1/2})^2\|_F^2 \\
&\quad= \sum_{i,j,k,\ell} \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \\
&\quad= \sum_{i=1}^{n}\sum_{k\neq i} \frac{(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2})^2}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}^2} \\
&\qquad+ \sum_{i=1}^{n}\sum_{k\neq i}\sum_{\ell\neq i} \frac{(\widehat{Y}_{ik}^2 - \widehat{\mathscr{K}_{ik}^2})(\widehat{Y}_{i\ell}^2 - \widehat{\mathscr{K}_{i\ell}^2})}{\widehat{\mathscr{D}}_{ii}^2\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \\
&\qquad+ \sum_{(i,j,k,\ell)\in T} \frac{(\widehat{Y}_{ik}\widehat{Y}_{jk} - \widehat{\mathscr{K}_{ik}}\widehat{\mathscr{K}_{jk}})(\widehat{Y}_{i\ell}\widehat{Y}_{j\ell} - \widehat{\mathscr{K}_{i\ell}}\widehat{\mathscr{K}_{j\ell}})}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}}.
\end{aligned}
$$

Each of these three summations is bounded (with high probability) by $C\gamma^2$ by Lemmata 3, 4 and 8, respectively. Let constants $c_1, c_2 > 0$ be as defined in Lemma 2 and Lemma 3 respectively, and choose $c_3 > 0$ so that $C_\gamma(\gamma^4 n^3 r^4)^{-1} \le n^{-c_3}$ for suitably large n, where C_γ is as defined in Lemma 8. By the union bound, with probability at least $1 - (n^{1-c_1} + n^{-c_2} + n^{-c_3})$, all three sums are bounded at once, and the result follows by taking $c = \min\{c_1 - 1, c_2, c_3\}$, □

Lemma 10. *Suppose that $\beta \to 0$ as $n \to \infty$. Under the conditions of Lemma 2, there exists a constant $C > 0$ such that with probability at least $1 - n^{1-c_1}$,*

$$
\|\widehat{L}\widehat{L} - (\widehat{\mathscr{D}}^{-1/2}\widehat{Y}\widehat{\mathscr{D}}^{-1/2})^2\|_F \le C\frac{\beta}{r^{1/2}}.
$$

Proof. Under the conditions of Lemma 2, with probability at least $1 - n^{1-c_1}$ it holds for all $i \in [n]$ that $|\widehat{\mathscr{D}}_{ii} - \sum_{k=1}^{n} \widehat{Y}_{ik}| \leq \beta \widehat{\mathscr{D}}_{ii}$. It follows that for a suitably chosen constant $C' > 0$, for all $i, j, k \in [n]$ we have

$$\left| \frac{1}{\widehat{D}_{ii}^{1/2} \widehat{D}_{jj}^{1/2} \widehat{D}_{kk}} - \frac{1}{\widehat{\mathscr{D}}_{ii}^{1/2} \widehat{\mathscr{D}}_{jj}^{1/2} \widehat{\mathscr{D}}_{kk}} \right| \leq \frac{C'\beta}{\widehat{\mathscr{D}}_{ii}^{1/2} \widehat{\mathscr{D}}_{jj}^{1/2} \widehat{\mathscr{D}}_{kk}}. \tag{3.16}$$

To see why this is the case (here we are following the argument motivating Equation A.6 in Rohe et al. (2011a)), note that when $|\widehat{\mathscr{D}}_{ii} - \sum_{k=1}^{n} \widehat{Y}_{ik}| \leq \beta \widehat{\mathscr{D}}_{ii}$ for all $i \in [n]$, we have

$$\frac{(1+\beta)^{-2}}{\widehat{\mathscr{D}}_{ii}^{1/2} \widehat{\mathscr{D}}_{jj}^{1/2} \widehat{\mathscr{D}}_{kk}} \leq \frac{1}{\widehat{D}_{ii}^{1/2} \widehat{D}_{jj}^{1/2} \widehat{D}_{kk}} \leq \frac{(1-\beta)^{-2}}{\widehat{\mathscr{D}}_{ii}^{1/2} \widehat{\mathscr{D}}_{jj}^{1/2} \widehat{\mathscr{D}}_{kk}},$$

and Equation 3.16 follows, since $\beta \to 0$ as $n \to \infty$, and thus

$$\begin{aligned} (1+\beta)^{-2} &\geq \frac{\beta^{-2} - 1}{(\beta^{-1} + 1)^2} = \frac{\beta^{-1} - 1}{\beta^{-1} + 1} \geq 1 - C''\beta, \\ (1-\beta)^{-2} &= 1 + \frac{2}{\beta^{-1} - 1} + \frac{1}{(\beta^{-1} - 1)^2} \leq 1 + C''\beta. \end{aligned}$$

Using (3.16), we have

$$\|\widehat{L}\widehat{L} - (\widehat{\mathscr{D}}^{-1/2} \widehat{Y} \widehat{\mathscr{D}}^{-1/2})^2\|_F^2 \leq C'\beta^2 \sum_{i,j,k,\ell} \frac{\widehat{Y}_{ik} \widehat{Y}_{jk} \widehat{Y}_{i\ell} \widehat{Y}_{j\ell}}{\widehat{\mathscr{D}}_{ii} \widehat{\mathscr{D}}_{jj} \widehat{\mathscr{D}}_{kk} \widehat{\mathscr{D}}_{\ell\ell}}.$$

Under the same event, we have $\sum_{k=1}^{n} \widehat{Y}_{ik} \leq (1+\beta)\widehat{\mathscr{D}}_{ii}$ for all $i \in [n]$, and making repeated use of this and the facts that $\widehat{Y}_{jk} \leq (1+r)$, and $\widehat{\mathscr{D}}_{ii} \geq nr$, it follows that

$$\begin{aligned}\|\widehat{L}\widehat{L} - (\widehat{\mathscr{D}}^{-1/2}\widehat{Y}\widehat{\mathscr{D}}^{-1/2})^2\|_F^2 &\leq C'\beta^2 \sum_{i,j,k,\ell} \frac{\widehat{Y}_{ik}\widehat{Y}_{jk}\widehat{Y}_{i\ell}\widehat{Y}_{j\ell}}{\widehat{\mathscr{D}}_{ii}\widehat{\mathscr{D}}_{jj}\widehat{\mathscr{D}}_{kk}\widehat{\mathscr{D}}_{\ell\ell}} \\ &\leq \frac{\beta^2(1+r)(1+\beta)^3}{r}.\end{aligned}$$

The result follows since r and β are bounded above by 1. □

To obtain our result in Theorem 3, take $\gamma = C'n^{-1/2}r^{-1}\log^{1/2} n$ and $\beta = C''n^{-1/2}r^{-1/2}\log^{1/2} n$ for suitably large constants $C', C'' > 0$. Note first that these choices of γ and β satisfy all of the constraints of the lemmata required for Lemma 9, so long as $r = \omega(n^{-1}\log n)$. Further, note that $\beta/r^{1/2} = C\gamma$ for some constant $C > 0$, and hence Lemma 10 implies that $\|\widehat{L}\widehat{L} - (\widehat{\mathscr{D}}^{-1/2}\widehat{Y}\widehat{\mathscr{D}}^{-1/2})^2\|_F \leq C\gamma$ with high probability. Combining Lemma 9 and Lemma 10 and applying the triangle inequality then yields Theorem 3.

Chapter 4

Vertex Nomination

In this chapter, we consider a different approach to search problems of the type discussed in Chapter 2. We have seen that the search collection can be represented as a complete weighted graph $G = (V, E)$ with weights given by similarity scores $\kappa(x, y)$ for all $x, y \in V$. In standard approaches to search, given a query q, we simply wish to find the elements $x \in V$ for which the similarity $\kappa(q, x)$ is largest. However, this approach fails to take into account all of the information present in graph G. Rather than looking merely at $\kappa(q, x)$ for all $x \in V$, we wish to perform a search that takes into account all similarities $\{\kappa(x, y) : x, y \in V \cup \{q\}\}$. This task of finding the vertices that are *topologically* most similar to q is the *vertex nomination* problem.

Given a graph in which a few vertices are deemed interesting a priori, the vertex nomination task (Coppersmith 2014) is to order the remaining vertices into a nomination list such that there is a concentration of interesting vertices at the top of the list (see Appendix E for a review of the literature on the vertex nomination problem and related work). Below, we prove that maximum-likelihood (ML)-based vertex nomination is consistent in the sense that the performance of the ML-based scheme asymptotically matches that of the Bayes optimal scheme. We prove theorems of this form both when model parameters are known and when they are unknown. Additionally, we introduce and prove consistency of a related, more scalable restricted-focus ML vertex nomination scheme. Finally, we incorporate vertex and edge features into ML-based vertex nomination and briefly explore the empirical effectiveness of this approach.

This chapter considers the vertex nomination problem as applied to random simple graphs. In Chapter 5, we will see how to generalize the results to a problem that arises in the context of similarity search and reranking problems. The material presented in this chapter appeared originally in slightly altered form in Lyzinski et al. (2016b).

4.1 Introduction and Background

Graphs are a common data modality, useful for modeling complex relationships between objects, with applications spanning fields as varied as biology (Jeong et al. 2001; Bullmore and Sporns 2009), sociology (Wasserman and Faust 1994), and computer vision (Foggia et al. 2014; Kandel et al. 2007), to name a few. For example, in neuroscience, vertices may be neurons and edges adjoin pairs of neurons that share a synapse (Bullmore and Sporns 2009); in social networks, vertices may correspond to people and edges to friendships between them (Carrington et al. 2005; Yang and Leskovec 2015); in computer vision, vertices may represent pixels in an image and edges may represent spatial proximity or multi-resolution mappings (Kandel et al. 2007). In many useful networks, vertices with similar attributes form densely-connected communities compared to vertices with highly disparate attributes, and uncovering these communities is an important step in understanding the structure of the network. There is an extensive literature devoted to uncovering this community structure in network data, including methods based on maximum modularity (Newman and Girvan 2004; Newman 2006b), spectral partitioning algorithms (von Luxburg 2007; Rohe et al. 2011b; Sussman et al. 2012; Lyzinski et al. 2014b), and likelihood-based methods (Bickel and Chen 2009), among others.

In the setting of *vertex nomination*, one community in the network is of particular interest, and the inference task is to order the vertices into a nomination list with those vertices from the community of interest concentrating at the top of the list. Refer to Appendix E for a more thorough discussion of the vertex nomination problem. Vertex nomination is a semi-supervised inference task, with example vertices from the community of interest—and, ideally, also examples not from the community of interest—being leveraged in order to create a nomination list. In this way, the vertex nomination problem is similar to the problem faced by personalized recommender systems (see, for example, Resnick and Varian 1997; Ricci et al. 2011), where, given a training list of objects of interest, the goal is to arrange the remaining objects into a recommendation list with "interesting" objects concentrated at the top of the list. The main difference between the two inference tasks is that in vertex nomination the features of the data are encoded into the topology of a network, rather than being observed directly as features (though see Section 4.5 for the case where vertices are annotated with additional information in the form of features). A more thorough review of the vertex nomination literature and related work can be found in Appendix E.

In this chapter, we prove that the maximum-likelihood vertex nomination scheme of Fishkind et al. (2015) is consistent (see Definition 2) under mild model assumptions on the underlying stochastic block model (Theorem 4). In the process, we propose a new, efficiently exactly solvable likelihood-based nomination scheme, the restricted-focus maximum-likelihood vertex-nomination scheme, $\mathcal{L}_R^{\text{ML}}$, and prove the analogous consistency result (Theorem 5). In addition, under mild model assumptions, we prove that both schemes maintain their consistency when the stochastic block model parameters are unknown and are estimated using the seed vertices (Theorems 6 and 7). In both cases, we show that consistency is possible even when the seeds are an asymptotically vanishing portion of the graph. Lastly, we show how both schemes can be easily modified to incorporate edge weights and vertex features (Section 4.5), before demonstrating the practical effect of our theoretical results on real and synthetic data (Section 4.6) and closing with a brief discussion (Section 4.7).

Before proceeding, we establish notation for this chapter and its sequel, in which we will use many of the ideas developed here. We say that a sequence of random variables $(X_n)_{n=1}^{\infty}$ converges almost surely to random variable X, written $X_n \to X$ a.s., if $\mathbb{P}[\lim_{n\to\infty} X_n = X] = 1$. We say a sequence of events $(A_n)_{n=1}^{\infty}$ occurs almost always almost surely (abbreviated a.a.a.s.) if with probability 1, A_n^c occurs for at most finitely many n. By the Borel-Cantelli lemma, $\sum_{n=1}^{\infty} \mathbb{P}[A_n^c] < \infty$ implies $(A_n)_{n=1}^{\infty}$ a.a.a.s. We write $\mathcal{G}_n$ to denote the set of all (possibly weighted) graphs on n vertices. Throughout, without loss of generality, we will assume that the vertex set is given by $V = \{1, 2, \ldots, n\}$. For a positive integer K, we will often use $[K]$ to denote the set $\{1, 2, \ldots, K\}$. For a set V, we will use $\binom{V}{2}$ to denote the set of all pairs of distinct elements of V. That is, $\binom{V}{2} = \{\{u, v\} : u, v \in V, u \neq v\}$. For a function f with domain V, we write $f_{|U}$ to denote the restriction of f to the set $U \subset V$.

4.1.1 Background

Stochastic block model (SBM; Holland et al. 1983) random graphs offer a theoretically tractable model for graphs with latent community structure (Rohe et al. 2011b; Sussman et al. 2012; Bickel and Chen 2009), and have been widely used in the literature to model community structure in real networks (Airoldi et al. 2008; Karrer and Newman 2011). While stochastic block models can be too simplistic to capture the eccentricities of many real graphs, they have proven to be a useful, tractable surrogate for more complicated networks (Airoldi et al. 2013; Olhede and Wolfe 2014).

Definition 1. *Let K and n be positive integers and let $\vec{n} = (n_1, n_2, \ldots, n_K)^\top \in$*

$\mathbb{R}^K$ *be a vector of positive integers with* $\sum_k n_k = n$*. Let* $b : [n] \to [K]$ *and let* $\Lambda \in [0,1]^{K\times K}$ *be symmetric. A* $\mathcal{G}_n$*-valued random graph* G *is an instantiation of a* $(K, \vec{n}, b, \Lambda)$ *conditional Stochastic Block Model, written* $G \sim \mathrm{SBM}(K, \vec{n}, b, \Lambda)$*, if*

i. The vertex set V *is partitioned into* K *blocks,* $V_1, V_2, \ldots, V_K$ *of cardinalities* $|V_k| = n_k$ *for* $k = 1, 2, \ldots, K$*;*

ii. The block membership function $b : V \to [K]$ *is such that for each* $v \in V$*,* $v \in V_{b(v)}$*;*

iii. The symmetric block communication matrix $\Lambda \in [0,1]^{K\times K}$ *is such that for each* $\{v, u\} \in \binom{V}{2}$*, there is an edge between vertices* u *and* v *with probability* $\Lambda_{b(u),b(v)}$*, independently of all other edges.*

Without loss of generality, let V_1 be the block of interest for vertex nomination. For each $k \in [K]$, we further decompose V_k into $V_k = S_k \cup U_k$ (with $|S_k| = m_k$), where the vertices in $S := \cup_k S_k$ have their block membership observed *a priori*. We call the vertices in S *seed vertices*, and let $m = |S|$. We will denote the set of nonseed vertices by $U = \cup_k U_k$, and for all $k \in [K]$, let $\mathfrak{u}_k := n_k - m_k = |U_k|$ and $n - m = \mathfrak{u} = |U|$. Throughout this chapter, we assume that the seed vertices S are chosen uniformly at random from all possible subsets of V of size m. The task in vertex nomination is to leverage the information contained in the seed vertices to produce a *nomination list* $\mathcal{L} : U \to [\mathfrak{u}]$ (i.e., an ordering of the vertices in U) such that the vertices in U_1 concentrate at the top of the list. We note that, strictly speaking, a nomination list

$\mathscr{L}$ is also a function of the observed graph G, a fact that we suppress for ease of notation. We measure the efficacy of a nomination scheme via *average precision*

$$\mathrm{AP}(\mathcal{L}) = \frac{1}{\mathfrak{u}_1} \sum_{i=1}^{\mathfrak{u}_1} \frac{\sum_{j=1}^{i} \mathbb{I}\{\mathcal{L}^{-1}(j) \in U_1\}}{i}. \tag{4.1}$$

AP ranges from 0 to 1, with a higher value indicating a more effective nomination scheme: indeed, $\mathrm{AP}(\mathcal{L}) = 1$ indicates that the first $\mathfrak{u}_1$ vertices in the nomination list are all from the block of interest, and $\mathrm{AP}(\mathscr{L}) = 0$ indicates that none of the $\mathfrak{u}_1$ top-ranked vertices are from the block of interest. Letting $H_k = \sum_{j=1}^{k} 1/j$ denote the k-th harmonic number, with the convention that $H_0 = 0$, we can rearrange (4.1) as

$$\mathrm{AP}(\mathcal{L}) = \sum_{i=1}^{\mathfrak{u}_1} \frac{H_{\mathfrak{u}_1} - H_{i-1}}{\mathfrak{u}_1} \mathbb{I}\{\mathcal{L}^{-1}(i) \in U_1\},$$

from which we see that the average precision is simply a convex combination of the indicators of correctness in the rank list, in which correctly placing an interesting vertex higher in the nomination list (i.e., with rank close to 1) is rewarded more than correctly placing an interesting vertex lower in the nomination list.

In Fishkind et al. (2015), three vertex nomination schemes were presented in the context of stochastic block model random graphs: the canonical vertex nomination scheme, $\mathcal{L}^{\mathrm{C}}$, which is suitable for small graphs (tens of vertices); the maximum-likelihood vertex-nomination scheme, $\mathcal{L}^{\mathrm{ML}}$, which is suitable for small to medium graphs (up to thousands of vertices); and the spectral partitioning vertex nomination scheme, $\mathcal{L}^{\mathrm{SP}}$, which is suitable for medium to very large graphs (up to tens of millions of vertices). In the stochastic block model setting, the canonical vertex nomination scheme is provably optimal: under mild model assumptions, $\mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{C}}) \geq \mathbb{E}\,\mathrm{AP}(\mathcal{L})$ for any vertex nomination scheme $\mathcal{L}$ (Fishkind et al. 2015), where the expectation is with respect to a $\mathcal{G}_{m+n}$-valued random graph G and the selection of the seed vertices. Thus, the canonical method is the vertex nomination analogue of the Bayes classifier, and this motivates the following definition:

Definition 2. *Let* $G \sim \mathrm{SBM}(K, \vec{n}, b, \Lambda)$. *With notation as above, a vertex nomination scheme* $\mathcal{L}$ *is consistent if*

$$\lim_{n\to\infty} |\mathbb{E}\,\mathrm{AP}(\mathcal{L}^{C}) - \mathbb{E}\,\mathrm{AP}(\mathcal{L})| = 0.$$

In our proofs below, where we establish the consistency of two nomination schemes, we prove a stronger fact, namely that $\mathrm{AP}(\mathscr{L}) = 1$ a.a.a.s. We prefer the definition of consistency given in Definition 2 since it allows us to speak about the best possible nomination scheme even when the model is such that $\lim_{n\to\infty} \mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{C}}) < 1$.

In Fishkind et al. (2015), it was proven that under mild assumptions on the stochastic block model underlying G, we have

$$\lim_{n\to\infty} \mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{SP}}) = 1,$$

from which the consistency of $\mathcal{L}^{\mathrm{SP}}$ follows immediately. The spectral nomination scheme $\mathcal{L}^{\mathrm{SP}}$ proceeds by first K-means clustering the adjacency spectral embedding (Sussman et al. 2012) of G, and then nominating vertices based on their distance to the cluster of interest. Consistency of $\mathcal{L}^{\mathrm{SP}}$ is an immediate consequence of the fact that, under mild model assumptions on the underlying stochastic block model, K-means clustering of the adjacency spectral embedding of G perfectly clusters the vertices of G a.a.a.s. (Lyzinski et al. 2014b).

Bickel and Chen (2009) proved that maximum-likelihood estimation provides consistent estimates of the model parameters in a more common variant of the conditional stochastic block model of Definition 1, namely, in the stochastic block model with random block assignments:

Definition 3. *Let K, n and Λ be as above. Let $\vec{\pi} = \pi_1, \pi_2, \ldots, \pi_K)^\top \in \Delta^{K-1}$ be a probability vector over K outcomes and let $\tau : V \to [K]$ be a random function. A $\mathcal{G}_n$-valued random graph G is an instantiation of a $(K, \vec{\pi}, \tau, \Lambda)$ Stochastic Block Model with random block assignments, written $G \sim \mathrm{SBM}(K, \vec{\pi}, \tau, \Lambda)$, if*

i. For each vertex $v \in V$ and block $k \in [K]$, independently of all other vertices, the block assignment function $\tau : V \to [K]$ assigns v to block k with probability π_k (i.e., $\mathbb{P}[\tau(v) = k] = \pi_k$);

ii. The symmetric block communication matrix $\Lambda \in [0,1]^{K \times K}$ is such that, conditioned on τ, for each $\{v, u\} \in \binom{V}{2}$ there is an edge between vertices u and v with probability $\Lambda_{\tau(u),\tau(v)}$, independently of all other edges.

A consequence of the result of Bickel and Chen (2009) is that the ML estimate of the block assignment function perfectly clusters the vertices a.a.a.s. in the setting where $G \sim \mathrm{SBM}(K, \vec{\pi}, \tau, \Lambda)$. This bears noting, as our maximum-likelihood vertex-nomination schemes $\mathcal{L}^{\mathrm{ML}}$ and $\mathcal{L}_R^{\mathrm{ML}}$ (defined below in Section 4.2) proceed by first constructing a maximum-likelihood estimate of the block membership function b, then ranking vertices based on a measure of model misspecification. Extending the results from Bickel and Chen (2009) to our present framework—where we consider Λ and $\vec{n}$ to be known (or errorfully estimated via seeded vertices) rather than parameters to be optimized over in the likelihood function as done in Bickel and Chen (2009)—is not immediate.

We note the recent result by Newman (2016), which shows the equivalence of maximum-likelihood and maximum-modularity methods in a special case of the stochastic block model when Λ is known. Our results, along with this recent result, immediately imply a consistent maximum-modularity-based vertex-nomination scheme under that special-case model.

4.2 Graph Matching and Maximum Likelihood Estimation

Consider $G \sim \mathrm{SBM}(K, \vec{n}, b, \Lambda)$ with associated adjacency matrix A, and, as above, denote the set of seed vertices by $S = \cup_k S_k$. Define the set of feasible block assignment functions

$$\begin{aligned}
&\mathcal{B} = \mathcal{B}(\vec{n}, b, S) \\
&\quad := \{\phi : V \to [K] \text{ s.t. for all } k \in [K],\, |\phi^{-1}(k)| = n_k, \text{ and } \phi(i) = b(i) \text{ for all } i \in S\}.
\end{aligned}$$

The ML estimator of $b \in \mathcal{B}$ is any member of the set of functions

$$\begin{aligned}
\hat{b} &= \arg\max_{\phi\in\mathcal{B}} \prod_{\{i,j\}\in\binom{V}{2}} \Lambda_{\phi(i),\phi(j)}^{A_{i,j}} (1 - \Lambda_{\phi(i),\phi(j)})^{1-A_{i,j}} \\
&= \arg\max_{\phi\in\mathcal{B}} \sum_{\{i,j\}\in\binom{V}{2}} A_{i,j} \log\left(\frac{\Lambda_{\phi(i),\phi(j)}}{1-\Lambda_{\phi(i),\phi(j)}}\right) \\
&= \arg\max_{\phi\in\mathcal{B}} \sum_{\{i,j\}\in\binom{U}{2}} A_{i,j} \log\left(\frac{\Lambda_{\phi(i),\phi(j)}}{1-\Lambda_{\phi(i),\phi(j)}}\right) + \sum_{(i,j)\in S\times U} A_{i,j} \log\left(\frac{\Lambda_{b(i),\phi(j)}}{1-\Lambda_{b(i),\phi(j)}}\right),
\end{aligned} \tag{4.2}$$

where the second equality follows from independence of the edges and splitting the edges in the sum according to whether or not they are incident to a seed vertex. We can reformulate (4.2) as a graph matching problem by identifying ϕ with a permutation matrix P:

Definition 4. *Let G_1 and G_2 be two n-vertex graphs with respective adjacency matrices A and B. The Graph Matching Problem for aligning G_1 and G_2 is*

$$\min_{P\in\Pi_n} \|AP - PB\|_F,$$

where Π_n *is defined to be the set of all* $n \times n$ *permutation matrices.*

The graph matching problem and its relation to vertex nomination is discussed in Appendix E, and we refer the reader there for further details. Incorporating seed vertices (i.e., vertices whose correspondence across G_1 and G_2 is known *a priori*) into the graph matching problem is immediate (Fishkind et al. 2012). Letting the seed vertices be (without loss of generality) $S = \{1, 2, \dots, m\}$ in both graphs, the seeded graph matching (SGM) problem is

$$\min_{P \in \Pi_{\mathfrak{u}}} \|A(I_m \oplus P) - (I_m \oplus P)B\|_F\,, \tag{4.3}$$

where

$$I_m \oplus P = \begin{bmatrix} I_m & 0 \\ 0 & P \end{bmatrix}.$$

Setting $B \in \mathbb{R}^{n \times n}$ to be the log-odds matrix

$$B_{i,j} := \log\left(\frac{\Lambda_{b(i),b(j)}}{1 - \Lambda_{b(i),b(j)}}\right), \tag{4.4}$$

observe that the optimization problem in Equation (4.2) is equivalent to that in (4.3) if we view B as encoding a weighted graph. Hence, we can apply known graph matching algorithms to approximately find $\hat{b}$.

Decomposing A and B as

$$A = \begin{array}{cc} & \begin{array}{cc} m & \mathfrak{u} \end{array} \\ \begin{array}{c} m \\ \mathfrak{u} \end{array} & \left[\begin{array}{cc} A^{(1,1)} & A^{(1,2)} \\ A^{(2,1)} & A^{(2,2)}) \end{array}\right] \end{array} \qquad B = \begin{array}{cc} & \begin{array}{cc} m & \mathfrak{u} \end{array} \\ \begin{array}{c} m \\ \mathfrak{u} \end{array} & \left[\begin{array}{cc} B^{(1,1)} & B^{(1,2)} \\ B^{(2,1)} & B^{(2,2)} \end{array}\right] \end{array}$$

and using the fact that $P \in \Pi_n$ is unitary, the seeded graph matching problem is equivalent (i.e., has the same minimizer) to

$$\min_{P \in \Pi_{\mathfrak{u}}} - \operatorname{tr}\left(A^{(2,2)} P (B^{(2,2)})^\top P^\top\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right) - \operatorname{tr}\left(A^{(2,1)} (B^{(2,1)})^\top P^\top\right).$$

Thus, we can recast (4.2) as a seeded graph matching problem so that finding

$$\hat{b} = \arg\max_{\phi \in \mathcal{B}} \sum_{\{i,j\} \in \binom{U}{2}} A_{i,j} \log\left(\frac{\Lambda_{\phi(i),\phi(j)}}{1 - \Lambda_{\phi(i),\phi(j)}}\right) + \sum_{(i,j) \in S \times U} A_{i,j} \log\left(\frac{\Lambda_{b(i),\phi(j)}}{1 - \Lambda_{b(i),\phi(j)}}\right)$$

is equivalent to finding

$$\hat{P} = \arg\min_{P \in \Pi_{\mathfrak{u}}} -\frac{1}{2} \operatorname{tr}\left(A^{(2,2)} P (B^{(2,2)})^\top P^\top\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right), \tag{4.5}$$

as we shall explain below.

With B defined as in (4.4), we define

$$\mathcal{Q} = \left\{Q \in \Pi_{\mathfrak{u}} \text{ s.t. } (I_m \oplus Q) B (I_m \oplus Q)^\top = B\right\}.$$

Define an equivalence relation $\sim$ on $\Pi_{\mathfrak{u}}$ via $P_1 \sim P_2$ iff there exists a $Q \in \mathcal{Q}$ such that $P_1 = P_2 Q$; i.e.,

$$(I_m \oplus P_1)B(I_m \oplus P_1)^\top = (I_m \oplus P_2Q)B(I_m \oplus P_2Q)^\top = (I_m \oplus P_2)B(I_m \oplus P_2)^\top.$$

Let $\hat{P}/\sim$ denote the set of equivalence classes of $\hat{P}$ under equivalence relation $\sim$. Solving (4.2) is equivalent to solving (4.5) in that there is a one-to-one correspondence between $\hat{b}$ and $\hat{P}/\sim$: for each $\phi \in \hat{b}$ there is a unique $P \in \hat{P}/\sim$ (with associated permutation σ) such that $\phi_{|_U} = b_{|_U} \circ \sigma$; and for each $P \in \hat{P}/\sim$ (with the permutation associated with $I_m \oplus P$ given by σ), it holds that $b \circ \sigma \in \hat{b}$.

4.2.1 The $\mathcal{L}^{\text{ML}}$ Vertex Nomination Scheme

The maximum-likelihood vertex-nomination scheme proceeds as follows. First, the SGM algorithm (Fishkind et al. 2012; Lyzinski et al. 2014a) is used to approximately find an element of $\hat{P}$, which we shall denote by P. Let the corresponding element of $\hat{b}$ be denoted by ϕ. For any $i, j \in V$ such that $\phi(i) \neq \phi(j)$, define $\phi_{i\leftrightarrow j} \in \mathcal{B}$ as

$$\phi_{i\leftrightarrow j}(v) = \begin{cases} \phi(i) & \text{if } v = j, \\ \phi(j) & \text{if } v = i, \\ \phi(v) & \text{if } v \neq i, j; \end{cases}$$

i.e., $\phi_{i\leftrightarrow j}$ agrees with ϕ except that i and j have their block memberships from ϕ switched in $\phi_{i\leftrightarrow j}$. For $i \in U$ such that $\phi(i) = 1$, define

$$\eta(i) := \left(\prod_{\substack{j\in U \text{ s.t.} \\ \phi(j)\neq 1}} \frac{\ell(\phi_{i\leftrightarrow j}, G)}{\ell(\phi, G)} \right)^{\frac{1}{\mathfrak{u}-\mathfrak{u}_1}},$$

where, for each $\psi \in \mathcal{B}$, the likelihood ℓ is given by

$$\ell(\psi, G) = \prod_{\{i,j\}\in\binom{U}{2}} \Lambda_{\psi(i),\psi(j)}^{A_{i,j}}(1-\Lambda_{\psi(i),\psi(j)})^{1-A_{i,j}} \prod_{(i,j)\in S\times U} \Lambda_{b(i),\psi(j)}^{A_{i,j}}(1-\Lambda_{b(i),\psi(j)})^{1-A_{i,j}}.$$

A low/high value of $\eta(i)$ is a measure of our confidence that i is/is not in the block of interest. For $i \in U$ such that $\phi(i) \neq 1$, define

$$\xi(i) := \left(\prod_{\substack{j\in U \text{ s.t.} \\ \phi(j)=1}} \frac{\ell(\phi_{i\leftrightarrow j}, G)}{\ell(\phi, G)} \right)^{\frac{1}{\mathfrak{u}_1}}.$$

A low/high value of $\xi(i)$ is a measure of our confidence that i is/is not in the block of interest. We are now ready to define the maximum-likelihood nomination scheme

$\mathcal{L}^{\text{ML}}$:

$$\begin{aligned}
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(1) &\in \arg\min\{\eta(v) : \phi(v) = 1\}\\
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(2) &\in \arg\min\left\{\eta(v) : v \in U \setminus \left\{(\mathcal{L}^{\text{ML}})^{-1}(1)\right\}, \phi(v) = 1\right\}\\
&\vdots\\
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(\mathfrak{u}_1) &\in \arg\min\Big\{\eta(v) : v \in U \setminus \left\{(\mathcal{L}^{\text{ML}})^{-1}(i)\right\}_{i=1}^{\mathfrak{u}_1-1}, \phi(v) = 1\Big\}\\
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(\mathfrak{u}_1+1) &\in \arg\max\{\xi(v) : \phi(v) \neq 1\}\\
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(\mathfrak{u}_1+2) &\in \arg\max\left\{\xi(v) : v \in U \setminus \left\{(\mathcal{L}^{\text{ML}})^{-1}(\mathfrak{u}_1+1)\right\}, \phi(v) \neq 1\right\}\\
&\vdots\\
\left(\mathcal{L}^{\text{ML}}\right)^{-1}(\mathfrak{u}) &\in \arg\max\Big\{\xi(v) : v \in U \setminus \left\{(\mathcal{L}^{\text{ML}})^{-1}(i)\right\}_{i=\mathfrak{u}_1+1}^{\mathfrak{u}-1}, \phi(v) \neq 1\Big\}
\end{aligned}$$

Note that in the event that an argmin (or argmax) above contains more than one element, the order in which these elements is nominated should be taken to be uniformly random.

Remark 6. In the event that Λ is unknown *a priori*, we can use the block memberships of the seeds S (assumed to be chosen uniformly at random from V) to estimate the edge probability matrix Λ as

$$\widehat{\Lambda}_{k,\ell} = \frac{|\{\{i,j\} \in E \text{ s.t. } i \in S_k,\, j \in S_\ell\}|}{m_k m_\ell} \text{ for } k \neq \ell,$$

and

$$\widehat{\Lambda}_{k,k} = \frac{|\{\{i,j\} \in E \text{ s.t. } i \in S_k,\, j \in S_k\}|}{\binom{m_k}{2}}.$$

The plug-in estimate $\widehat{B}$ of B, given by

$$\widehat{B}_{i,j} := \log\left(\frac{\widehat{\Lambda}_{b(i),b(j)}}{1 - \widehat{\Lambda}_{b(i),b(j)}}\right),$$

can then be used in place of B in Eq. (4.5). If, in addition, $\vec{n}$ is unknown, we can estimate the block sizes n_k as

$$\hat{n}_k = \frac{m_k n}{m},$$

for each $k \in [K]$, and these estimates can be used to determine the block sizes in $\widehat{B}$.

4.2.2 The $\mathcal{L}_R^{\mathrm{ML}}$ Vertex Nomination Scheme

Graph matching is a computationally difficult problem, and there are no known polynomial time algorithms for solving the general graph matching problem for simple graphs. Furthermore, if the graphs are allowed to be weighted, directed, and loopy, then graph matching is equivalent to the NP-hard quadratic assignment problem. While there are numerous efficient, approximate graph matching algorithms (see, for example, Vogelstein et al. 2015; Fishkind et al. 2012; Zaslavskiy et al. 2009b; Fiori et al. 2013, and the references therein), these algorithms often lack performance guarantees.

Inspired by the restricted-focus seeded graph matching problem considered in Lyzinski et al. (2014a), we now define the computationally tractable restricted-focus maximum-likelihood nomination scheme $\mathcal{L}_R^{\mathrm{ML}}$. Rather than attempting to quickly approximate a solution to the full graph matching problem as in Vogelstein et al. (2015); Fishkind et al. (2012); Zaslavskiy et al. (2009b); Fiori et al. (2013), this approach simplifies the problem by ignoring the edges between unseeded vertices. An analogous restriction for matching simple graphs was introduced in Lyzinski et al. (2014a). We begin by considering the graph matching problem in Eq. (4.5). The objective function

$$-\frac{1}{2}\operatorname{tr}\left(A^{(2,2)}P(B^{(2,2)})^\top P^\top\right)-\operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)}P^\top\right)$$

consists of two terms: $-\frac{1}{2}\operatorname{tr}\left(A^{(2,2)}P(B^{(2,2)})^\top P^\top\right)$, which seeks to align the induced subgraphs of the nonseed vertices; and $-\operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)}P^\top\right)$, which seeks to align the induced bipartite subgraphs between the seed and nonseed vertices. While the graph matching objective function, Eq. (4.5), is quadratic in P, restricting our focus to the second term in Eq. (4.5) yields the following *linear assignment problem*

$$\tilde{P}=\arg\min_{P\in\Pi_{\mathfrak{u}}}-\operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)}P^\top\right), \tag{4.6}$$

which can be efficiently and exactly solved in $O(\mathfrak{u}^3)$ time with the Hungarian algorithm (Kuhn 1955; Jonker and Volgenant 1987). We note that, exactly as was the case of $\hat{P}$ and $\hat{b}$, finding $\tilde{P}$ is equivalent to finding

$$\tilde{b} = \arg\max_{\phi \in \mathcal{B}} \sum_{(i,j) \in S \times U} A_{i,j} \log\left(\frac{\Lambda_{b(i),\phi(j)}}{1 - \Lambda_{b(i),\phi(j)}}\right),$$

in that there is a one-to-one correspondence between $\tilde{b}$ and $\tilde{P}/\sim$.

The $\mathcal{L}_R^{\text{ML}}$ scheme proceeds as follows. First, the linear assignment problem, Eq. (4.6), is exactly solved using, for example, the Hungarian algorithm (Kuhn 1955) or the path augmenting algorithm of Jonker and Volgenant (1987), yielding $P \in \tilde{P}$. Let the corresponding element of $\tilde{b}$ be denoted by ϕ. For $i \in U$ such that $\phi(i) = 1$, define

$$\tilde{\eta}(i) := \left(\prod_{\substack{j \in U \text{ s.t.} \\ \phi(j) \neq 1}} \frac{\ell_R(\phi_{i \leftrightarrow j}, G)}{\ell_R(\phi, G)} \right)^{\frac{1}{u - u_1}},$$

where, for each $\psi \in \mathcal{B}$, the *restricted* likelihood ℓ_R is defined via

$$\ell_R(\psi, G) = \prod_{(i,j) \in S \times U} \Lambda_{b(i),\psi(j)}^{A_{i,j}} (1 - \Lambda_{b(i),\psi(j)})^{1 - A_{i,j}}.$$

As with $\mathcal{L}^{\text{ML}}$, a low/high value of $\tilde{\eta}(i)$ is a measure of our confidence that i is/is not in the block of interest. For $i \in U$ such that $\phi(i) \neq 1$, define

$$\tilde{\xi}(i) := \left(\prod_{\substack{j \in U \text{ s.t.} \\ \phi(j) = 1}} \frac{\ell_R(\phi_{i \leftrightarrow j}, G)}{\ell_R(\phi, G)} \right)^{\frac{1}{u_1}}.$$

As before, a low/high value of $\tilde{\xi}(i)$ is a measure of our confidence that i is/is not in the block of interest. We are now ready to define $\mathcal{L}_R^{\text{ML}}$:

$$\begin{aligned}
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(1) &\in \arg\min\{\tilde{\eta}(v) : \phi(v)=1\}\\
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(2) &\in \arg\min\left\{\tilde{\eta}(v) : v \in U \setminus \left\{(\mathcal{L}_R^{\text{ML}})^{-1}(1)\right\}, \phi(v)=1\right\}\\
&\vdots\\
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(\mathfrak{u}_1) &\in \arg\min\left\{\tilde{\eta}(v) : v \in U \setminus \left\{(\mathcal{L}_R^{\text{ML}})^{-1}(i)\right\}_{i=1}^{\mathfrak{u}_1-1}, \phi(v)=1\right\}\\
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(\mathfrak{u}_1+1) &\in \arg\max\left\{\tilde{\xi}(v) : \phi(v)\neq 1\right\}\\
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(\mathfrak{u}_1+2) &\in \arg\max\left\{\tilde{\xi}(v) : v \in U \setminus \left\{(\mathcal{L}_R^{\text{ML}})^{-1}(\mathfrak{u}_1+1)\right\}, \phi(v)\neq 1\right\}\\
&\vdots\\
\left(\mathcal{L}_R^{\text{ML}}\right)^{-1}(\mathfrak{u}) &\in \arg\max\left\{\tilde{\xi}(v) : v \in U \setminus \left\{(\mathcal{L}_R^{\text{ML}})^{-1}(i)\right\}_{i=\mathfrak{u}_1+1}^{\mathfrak{u}-1}, \phi(v)\neq 1\right\}
\end{aligned}$$

Note that, as before, in the event that the argmin (or argmax) in the definition of $\mathcal{L}_R^{\text{ML}}$ contains more than one element above, the order in which these elements are nominated should be taken to be uniformly random.

Unlike $\mathcal{L}^{\text{ML}}$, the restricted focus scheme $\mathcal{L}_R^{\text{ML}}$ is feasible even for comparatively large graphs (up to thousands of nodes, in our experience). However, we will see in Section 4.6 that the extra information available to $\mathcal{L}^{\text{ML}}$—the adjacency structure among the nonseed vertices—leads to superior precision in the $\mathcal{L}^{\text{ML}}$ nomination lists as compared to $\mathcal{L}_R^{\text{ML}}$. We next turn our attention to proving the consistency of the $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}_R^{\text{ML}}$ schemes.

4.3 Consistency of $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}_R^{\text{ML}}$

In this section, we state theorems ensuring the consistency of the vertex nomination schemes $\mathcal{L}^{\text{ML}}$ (Theorem 4) and $\mathcal{L}_R^{\text{ML}}$ (Theorem 5). For the sake of expository continuity, proofs are given in Section 4.8. We note here that in these Theorems, the parameters of the underlying block model are assumed to be known *a priori.* In Section 4.4, we prove the consistency of $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}_R^{\text{ML}}$ in the setting where the model parameters are unknown and must be estimated, as in Remark 6.

Let $G \sim \text{SBM}(K, \vec{n}, b, \Lambda)$ with associated adjacency matrix A, and let B be defined as in (4.4). For each $P \in \Pi_{\mathfrak{u}}$ (with associated permutation σ) and $k, \ell \in [K]$, define

$$\epsilon_{k,\ell} = \epsilon_{k,\ell}(P) = |\{v \in U_k \text{ s.t. } \sigma(v) \in U_\ell\}|$$

to be the number of vertices in U_k mapped to U_ℓ by $I_m \oplus P$, and for each $k \in [K]$ define

$$\epsilon_{k,\bullet}(P) := \epsilon_{k,\bullet} = \sum_{\ell \neq k} \epsilon_{k,\ell}.$$

Before stating and proving the consistency of $\mathcal{L}^{\text{ML}}$, we first establish some necessary notation. Note that in the definitions and theorems presented next, all values implicitly depend on n, as $\Lambda = \Lambda_n$ is allowed to vary in n. Let L be the set of distinct entries of Λ, and define

$$\alpha = \min_{\{k,\ell\}\text{ s.t. } k\neq\ell} |\Lambda_{k,k} - \Lambda_{k,\ell}| \quad \beta = \min_{\{k,\ell\}\text{ s.t. } k\neq\ell} |B_{k,k} - B_{k,\ell}| \quad c = \max_{i,j,k,\ell} |B_{i,j} - B_{k,\ell}|, \tag{4.7}$$

$$\gamma = \min_{x,y\in L} |x-y|, \quad \kappa = \min_{x,y\in L} \left| \log\left(\frac{x}{1-x}\right) - \log\left(\frac{y}{1-y}\right) \right|. \tag{4.8}$$

Theorem 4. *Let $G \sim \mathrm{SBM}(K, \vec{n}, b, \Lambda)$ and assume that*

i. $K = o(\sqrt{n})$*;*

ii. $\Lambda \in [0,1]^{K\times K}$ *is such that for all* $k, \ell \in [K]$ *with* $k \neq \ell$, $\Lambda_{k,k} \neq \Lambda_{k,\ell}$*;*

iii. For each $k \in [K]$, $\mathfrak{u}_k = \omega(\sqrt{n})$*, and* $m_k = \omega(\log \mathfrak{u}_k)$*;*

iv. $\frac{c^2}{\alpha\beta\kappa\gamma} = \Theta(1)$.

Then it holds that $\lim_{n\to\infty} \mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{ML}}) = 1$*, and* $\mathcal{L}^{\mathrm{ML}}$ *is a consistent nomination scheme.*

A proof of Theorem 4 is given in Section 4.8.

Remark 7. There are numerous assumptions akin to those in Theorem 4 under which we can show that $\mathcal{L}^{\mathrm{ML}}$ is consistent. Essentially, we need to ensure that if we define $\mathcal{P}' = \{P \in \Pi_{\mathfrak{u}} : \epsilon_{1,\bullet}(P) = \Theta(\mathfrak{u}_1)\}$, then $\mathbb{P}\left(\exists\ P \in \mathcal{P}' \text{ s.t. } X_P \leq 0\right)$ is summably small, from which it follows that $\epsilon_{1,\bullet} = o(\mathfrak{u}_1)$ with high probability, which is enough to ensure the desired consistency of $\mathcal{L}^{\mathrm{ML}}$.

Consistency of $\mathcal{L}_R^{\mathrm{ML}}$ holds under similar assumptions.

Theorem 5. *Let* $G \sim \mathrm{SBM}(K, \vec{n}, b, \Lambda)$. *Under the following assumptions*

i. $K = \Theta(1)$*;*

ii. $\Lambda \in [0,1]^{K \times K}$ *is such that for all* $k, \ell \in [K]$ *with* $k \neq \ell$, $\Lambda_{k,k} \neq \Lambda_{k,\ell}$;

iii. For each $k \in [K]$, $\mathfrak{u}_k = \omega(\sqrt{n})$, *and* $m_k = \omega(\log \mathfrak{u}_k)$*;*

iv. $\frac{c^2}{\alpha\beta\kappa\gamma} = \Theta(1)$*;*

it holds that $\lim_{n\to\infty} \mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{ML}}) = 1$, *and* $\mathcal{L}^{\mathrm{ML}}$ *is a consistent nomination scheme.*

A proof of this Theorem can be found in Section 4.8.

4.4 Consistency of $\mathcal{L}^{\mathrm{ML}}$ and $\mathcal{L}_R^{\mathrm{ML}}$ When the Model Parameters are Unknown

If Λ is unknown *a priori*, then the seeds can be used to estimate Λ as $\widehat{\Lambda}$, and n_i as $\hat{n}$ for each $i \in [K]$. In this section, we will prove analogues of the consistency Theorems 4 and 5 in the case where Λ and $\vec{n}$ are estimated using seeds. In Theorems 6 and 7 below, we prove that under mild model assumptions, both $\mathcal{L}^{\mathrm{ML}}$ and $\mathcal{L}_R^{\mathrm{ML}}$ are consistent vertex nomination schemes, even when the seed vertices form a vanishing fraction of the graph.

We now state the consistency result analogous to Theorem 4, this time for the case where we estimate Λ and $\vec{n}$. The proof can be found in Section 4.8.

Theorem 6. *Let $\Lambda \in \mathbb{R}^{K \times K}$ be a fixed, symmetric, block probability matrix satisfying*

i. K is fixed in n;

ii. $\Lambda \in [0,1]^{K \times K}$ is such that for all $k, \ell \in [K]$ with $k \neq \ell$, $\Lambda_{k,k} \neq \Lambda_{k,\ell}$;

iii. For each $k \in [K]$, $n_k = \Theta(n)$ and $m_k = \omega(n^{2/3}\log(n))$;

iv. α and γ defined as in (4.7) and (4.8) are fixed in n.

Suppose that the model parameters of $G \sim (K, \vec{n}, b, \Lambda)$ are estimated as in Remark 6 yielding log-odds matrix estimate $\widehat{B}$ and estimated block sizes $\hat{n} = (\hat{n}_1, \hat{n}_2, \ldots, \hat{n}_K)^T$. If $\mathcal{L}^{\mathrm{ML}}$ is run on A and $\widehat{B}$ using the block sizes given by $\hat{n}$, then under the above assumptions it holds that $\lim_{n\to\infty} \mathbb{E}\,\mathrm{AP}(\mathcal{L}^{\mathrm{ML}}) = 1$, and $\mathcal{L}^{\mathrm{ML}}$ is a consistent nomination scheme.

We now state the analogous consistency result to Theorem 5 when we estimate Λ and $\vec{n}$. The proof is given in Section 4.8.

Theorem 7. *Let $\Lambda \in \mathbb{R}^{K \times K}$ be a fixed, symmetric, block probability matrix satisfying*

i. K is fixed in n;

ii. $\Lambda \in [0,1]^{K \times K}$ is such that for all $k, \ell \in [K]$ with $k \neq \ell$, $\Lambda_{k,k} \neq \Lambda_{k,\ell}$;

iii. For each $k \in [K]$ s.t. $k \neq 1$, $n_k = \Theta(n)$ and $m_k = \omega(n^{2/3}\log(n))$;

iv. $n_1 = \Theta(n)$ *and* $m_1 = \omega(n^{4/5})$*;*

v. α *and* γ *defined at (4.7) and (4.8) are fixed in* n*.*

Suppose that the model parameters of $G \sim (K, \vec{n}, b, \Lambda)$ *are estimated as in Remark 6 yielding* $\widehat{B}$ *and estimated block sizes* $\hat{n} = (\hat{n}_1, \hat{n}_2, \dots, \hat{n}_K)^T$*. If* $\mathcal{L}^{\text{ML}}$ *is run on* A *and* $\widehat{B}$ *using block sizes given by* $\hat{n}$*, then under the above assumptions it holds that* $\lim_{n\to\infty} \mathbb{E}\,\text{AP}(\mathcal{L}^{\text{ML}}) = 1$ *and* $\mathcal{L}^{\text{ML}}$ *is a consistent nomination scheme.*

The two preceding theorems imply that vertex nomination is possible even when the number of seeds is a vanishing fraction of the vertices in the graph. Indeed, we find that in practice, accurate nomination is possible even with just a handful of seed vertices. See the experiments presented in Section 4.6.

4.5 Model Generalizations

Network data rarely appears in isolation. In the vast majority of use cases, the observed graph is richly annotated with information about the vertices and edges of the network. For example, in a social network, in addition to information about which users are friends, we may have vertex-level information in the form of age, education level, hobbies, etc. Similarly, in many networks, not all edges are created equal. Edge weights may encode the strength of a relation, such as the volume of trade between two countries. In this section, we sketch how the $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}^{\text{ML}}_R$ vertex nomination schemes can be extended to such annotated networks by incorporating edge weights and vertex features. To wit, all of the theorems proven above translate *mutatis mutandis* to the setting in which G is a drawn from a bounded canonical exponential family stochastic block model. Consider a single parameter exponential family of distributions whose density can be expressed in canonical form as

$$f(x|\theta) = h(x)e^{T(x)\theta - \mathcal{A}(\theta)}.$$

We will further assume that $h(x)$ has bounded support. We define

Definition 5. *A $\mathcal{G}_n$-valued random graph G is an instantiation of a $(K, \vec{n}, b, \Theta)$ bounded, canonical exponential family stochastic block model, written $G \sim \text{ExpSBM}(K, \vec{n}, b, \Theta)$, if*

i. The vertex set V is partitioned into K blocks, $V_1, V_2, \ldots, V_K$ with sizes $|V_k| = n_k$ for $k = 1, 2, \ldots, K$;

ii. The block membership function $b : V \to [K]$ is such that for each $v \in V$, $v \in V_{b(v)}$;

iii. The symmetric block parameter matrix $\Theta = [\theta_{k,\ell}] \in \mathbb{R}^{K \times K}$ is such that the $\{i, j\} \in \binom{V}{2}$, $A_{i,j}$ $(= A_{j,i})$ are independent, distributed according to the density

$$f_{A_{i,j}}(x|\theta_{b(i),b(j)}) = h(x)e^{T(x)\theta_{b(i),b(j)} - \mathcal{A}(\theta_{b(i),b(j)})}.$$

Note that the exponential family density is usually written as $h(x)e^{-x\theta - A(\theta)}$, where

$A(\cdot)$ is the log-normalization function. We have made the notational substitution to avoid confusion with the adjacency matrix A. If $G \sim \text{ExpSBM}(K, \vec{n}, b, \Theta)$, analogues to Theorems 4, 5, 6 and 7 follow *mutatis mutandis* if we use seeded graph matching to match $\widetilde{A} = [\widetilde{A}_{i,j}] := [T(A_{i,j})]$ to $B = [B_{i,j}] := [\theta_{b(i),b(j)}]$; i.e., under analogous model assumptions, $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}^{\text{ML}}_R$ are both consistent vertex nomination schemes when the model parameters are known or estimated via seeds. The key property being exploited here is that $\mathbb{E}(T(X))$ is a nondecreasing function of θ. We expect that results analogous to Theorems 4, 5, 6 and 7 can be shown to hold for more general weight distributions as well, but we do not pursue this further here.

Incorporating vertex features into $\mathcal{L}^{\text{ML}}$ and $\mathcal{L}^{\text{ML}}_R$ is immediate. Suppose that each vertex $v \in V$ is accompanied by a d-dimensional feature vector $X_v \in \mathcal{R}^d$. The features could encode additional information about the community structure of the underlying network; for example, if $b(v) = k$ then perhaps $X_v \sim \text{Norm}(\mu_k, \Sigma_k)$ where the parameters of the normal distribution vary across blocks and are constant within blocks. This setup, in which vertices are "annotated" or "attributed" with additional information, is quite common. Indeed, in almost all use cases, some auxiliary information about the graph is available, and methods that can leverage this auxiliary information are crucial. See, for example, Yang et al. (2013); Zhang et al. (2015); Newman and Clauset (2016); Franke and Wolfe (2016) and citations therein. We model vertex features as follows. Conditioning on $b(v) = k$, the feature associated to v is drawn, independently of A and of all other features X_u, from a distribution with density $f_{b(v)}$. Define the feature matrix X via

$$X = \begin{array}{c} \\ m \\ \mathfrak{u} \end{array} \begin{array}{c} d \\ \left[\begin{array}{c} X^{(m)} \\ X^{(\mathfrak{u})} \end{array}\right] \end{array},$$

where $X^{(m)}$ represents the features of the seed vertices in S, and $X^{(\mathfrak{u})}$ the features of the nonseed vertices in U. For each block $k \in [K]$, let $\hat{f}_k$ be an estimate of the density f_i, and create matrix $F \in \mathbb{R}^{m+\mathfrak{u}}$ given by

$$F = \begin{array}{c} \mathfrak{u}_1 \\ \mathfrak{u}_2 \\ \vdots \\ \mathfrak{u}_K \end{array} \left[\begin{array}{cccc} \hat{f}_1(X_1) & \hat{f}_1(X_2) & \cdots & \hat{f}_1(X_{\mathfrak{u}}) \\ \hat{f}_2(X_1) & \hat{f}_2(X_2) & \cdots & \hat{f}_2(X_{\mathfrak{u}}) \\ \vdots & & & \vdots \\ \hat{f}_K(X_1) & \hat{f}_K(X_2) & \cdots & \hat{f}_K(X_{\mathfrak{u}}) \end{array}\right].$$

Then we can incorporate the feature density into the seeded graph matching problem in (4.5) by adding a linear factor to the quadratic assignment problem:

$$\hat{P} = \arg\min_{P \in \Pi_{\mathfrak{u}}} -\frac{1}{2} \operatorname{tr}\left(A^{(2,2)} P (B^{(2,2)})^\top P^\top\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right) - \lambda \operatorname{tr} F P^\top. \quad (4.9)$$

The factor $\lambda \in \mathbb{R}^+$ allows us to weight the features encapsulated in X versus the information encoded into the network topology of G.

Vertex nomination proceeds as follows. First, the SGM algorithm of Fishkind et al. (2012); Lyzinski et al. (2014a) is used to approximately find an element of $\hat{P}$ in Eq. (4.9), which we shall denote by P. Let the block membership function corresponding to P be denoted ϕ. For $i \in U$ such that $\phi(i) = 1$, define

$$\eta_F(i) := \left(\prod_{\substack{j \in U \text{ s.t.} \\ \phi(j) \neq 1}} \frac{\ell_F(\phi_{i \leftrightarrow j}, G)}{\ell_F(\phi, G)} \right)^{\frac{1}{\mathfrak{u} - \mathfrak{u}_1}},$$

where, for each $\psi \in \mathcal{B}$, the likelihood ℓ_F is given by

$$\ell_F(\psi, G) = \prod_{\{i,j\} \in \binom{U}{2}} \Lambda_{\psi(i),\psi(j)}^{A_{i,j}} (1 - \Lambda_{\psi(i),\psi(j)})^{1 - A_{i,j}} \\ \cdot \prod_{(i,j) \in S \times U} \Lambda_{b(i),\psi(j)}^{A_{i,j}} (1 - \Lambda_{b(i),\psi(j)})^{1 - A_{i,j}} \prod_{i \in U} \hat{f}_{b(i)}(X_i),$$

where, for $k \in [K]$, $\hat{f}_k(\cdot)$ is the estimated density of the k-th block features. Note that here we assume that the feature densities must be estimated, even when the matrix Λ is known. A low/high value of $\eta_F(i)$ is a measure of our confidence that i is/is not in the block of interest. For $i \in U$ such that $\phi(i) \neq 1$, define

$$\xi_F(i) := \left(\prod_{\substack{j \in U \text{ s.t.} \\ \phi(j) = 1}} \frac{\ell_F(\phi_{i \leftrightarrow j}, G)}{\ell_F(\phi, G)} \right)^{\frac{1}{\mathfrak{u}_1}}.$$

A low/high value of $\xi_F(i)$ is a measure of our confidence that i is/is not in the block of interest. The nomination list produced by $\mathcal{L}_F^{\mathrm{ML}}$ is then realized via:

$$\begin{aligned}
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(1) &\in \arg\min\{\eta_F(v) : \phi(v) = 1\} \\
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(2) &\in \arg\min\left\{\eta_F(v) : v \in U \setminus \left\{(\mathcal{L}_F^{\mathrm{ML}})^{-1}(1)\right\}, \phi(v) = 1\right\} \\
&\vdots \\
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(\mathfrak{u}_1) &\in \arg\min\Big\{\eta_F(v) : v \in U \setminus \left\{(\mathcal{L}_F^{\mathrm{ML}})^{-1}(i)\right\}_{i=1}^{\mathfrak{u}_1 - 1}, \phi(v) = 1\Big\} \\
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(\mathfrak{u}_1 + 1) &\in \arg\max\left\{\xi_F(v) : \phi(v) \neq 1\right\} \\
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(\mathfrak{u}_1 + 2) &\in \arg\max\left\{\xi_F(v) : v \in U \setminus \left\{(\mathcal{L}_F^{\mathrm{ML}})^{-1}(\mathfrak{u}_1 + 1)\right\}, \phi(v) \neq 1\right\} \\
&\vdots \\
\left(\mathcal{L}_F^{\mathrm{ML}}\right)^{-1}(\mathfrak{u}) &\in \arg\max\Big\{\xi_F(v) : v \in U \setminus \left\{(\mathcal{L}_F^{\mathrm{ML}})^{-1}(i)\right\}_{i=\mathfrak{u}_1 + 1}^{\mathfrak{u} - 1}, \phi(v) \neq 1\Big\}
\end{aligned}$$

Note that, once again, in the event that the argmin (or argmax) contains more than one element above, the order in which these elements is nominated should be taken to be uniformly random.

We leave for future work a more thorough investigation of how best to choose the parameter λ. We found that choosing λ approximately equal to the number of nonseed vertices yielded reliably good results, but in general the best choice of λ is likely to be dependent on both the structure of the graph and the available features (e.g., how well the features actually predict block membership). We note also that in the case where the feature densities are not easily estimated or where we would like to relax our distributional assumptions, we might consider other terms to use in lieu of $\operatorname{tr} FP^\top$. For example, let $\hat{\mu}_k = \frac{1}{m_k}\sum_{v \in S_k} X_v$ be the empirical estimate of μ_k, the average feature vector for the seeds in block k, and create let Y be defined via

$$Y = \begin{array}{cc} & d \\ \begin{array}{c} \mathfrak{u}_1 \\ \mathfrak{u}_2 \\ \vdots \\ \mathfrak{u}_K \end{array} & \begin{bmatrix} \hat{\mu}_1 \otimes \vec{1} \\ \hat{\mu}_2 \otimes \vec{1} \\ \vdots \\ \hat{\mu}_k \otimes \vec{1} \end{bmatrix} \end{array}.$$

Incorporating these features into the seeded graph matching problem similarly to (4.9), we have

$$\hat{P} = \arg\min_{P \in \Pi_{\mathfrak{u}}} -\frac{1}{2} \operatorname{tr}\left(A^{(2,2)} P (B^{(2,2)})^\top P^\top\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right) - \lambda \operatorname{tr}(X^{(\mathfrak{u})} Y^\top P^\top). \tag{4.10}$$

We leave further exploration of this and related approaches, as well as how to deal with categorical data (e.g., as in Newman and Clauset (2016)), for future work.

4.6 Experiments

To compare the performance of maximum-likelihood vertex nomination against other methods, we performed experiments on five data sets, one synthetic, the others from linguistics, sociology, political science and ecology.

In all our data sets, we consider vertex nomination both when the edge probability matrix Λ is known and when it must be estimated. When model parameters are unknown, $m < n$ seed vertices are selected at random and the edge probability matrix is estimated based on the subgraph induced by the seeds, with entries of the edge probability matrix estimated via add-one smoothing. In the case of synthetic data, the known-parameter case simply corresponds to the algorithm having access to the parameters used to generate the data. We consider a 3-block stochastic block model (see below), so the known-parameter case corresponds to the true edge probability matrix being given. In the case of our real-world data sets, the notion of a "true" Λ is more hazy. Here, knowing the model parameters corresponds to using the entire graph, along with the true block memberships, to estimate Λ, again using add-one smoothing. This is, in some sense, the best access we can hope to have to the model parameters, to the extent that such parameters even exist in the first place.

4.6.1 Simulations

We consider graphs generated from stochastic block models at two different scales. Following the experiments in Fishkind et al. (2015), we consider 3-block models, where block sizes are given by $\vec{n} = q \cdot (4, 3, 3)^\top$ for $q = 1, 50$, which we term the small and medium cases, respectively. In Fishkind et al. (2015), a third case, with $q = 1000$, was also considered, but since ML vertex nomination is not practical at this scale, we do not include such experiments here, though we note that $\mathcal{L}_R^{\mathrm{ML}}$ can be run successfully on such a graph. We use an edge probability matrix given by

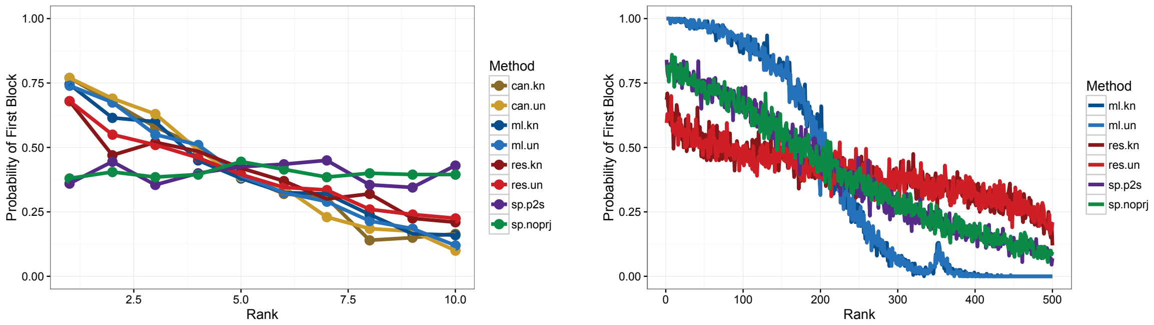
$$\Lambda(t) = t \begin{bmatrix} 0.5 & 0.3 & 0.4 \\ 0.3 & 0.8 & 0.6 \\ 0.4 & 0.6 & 0.3 \end{bmatrix} + (1-t) \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 \end{bmatrix} \tag{4.11}$$

for $t = 1, 0.3$ respectively in the small and medium cases, so that the amount of signal present in the graph is smaller as the number of vertices increases. We consider $m = 4, 20$ seeds in the small and medium scales, respectively. For a given choice of $\vec{n}, m, t$, we generate a single draw of an SBM with edge probability matrix $\Lambda(t)$ and block sizes given by $\vec{n}$. A set of m vertices is chosen uniformly at random from the first block to be seeds. Note that this means that the only model parameter that can be estimated is the intra-block probability for the first block. For all model parameter estimation in the ML methods (i.e., for the unknown case of $\mathcal{L}^{\mathrm{ML}}$ and $\mathcal{L}_R^{\mathrm{ML}}$), we use add-1 smoothing to prevent inaccurate estimates. We note that in all conditions, the block of interest (the first block) is not the densest block of the graph.

Recall that all of the methods under consideration return a list of the nonseed vertices, which we call a *nomination list*, with the vertices sorted according to how likely they are to be in the block of interest. Thus, vertices appearing early in the nomination list are the best candidates to be vertices of interest. Figure 4.1 compares the performance of canonical, spectral, maximum-likelihood and restricted-focus ML vertex nomination by looking at (estimates of) their average nomination lists. The plot shows, for each of the methods under consideration, an estimate (each based on 200 Monte Carlo replicates) of the average nomination list. Each curve describes the empirical probability that the kth-ranked vertex was indeed a vertex of interest. A perfect method, which on every input correctly places the n_1 vertices of interest in the first n_1 entries of the nomination list, would produce a curve in Figure 4.1 resembling a step function, with a step from 1 to 0 at the $(n_1 + 1)$th rank. Conversely, a method operating purely at random would yield an average nomination list that is constant n_1/n. Canonical vertex nomination is shown in gold, ML in blue, restricted-focus ML in red, and spectral vertex nomination is shown in purple and green. These two colors correspond, respectively, to spectral VN in which vertex embeddings are projected to the unit sphere prior to nomination and in which the embeddings are used as-is. In sparse networks, the adjacency spectral embedding places all vertices near to the origin. In such settings, projection to the sphere often makes cluster structure in the embeddings more easily recoverable. Dark colors correspond to the known-parameter case, and light colors correspond to unknown parameters. Note that spectral VN does not make such a distinction.

Examining the plots, we see that in the small case, maximum-likelihood nomination is quite competitive with the canonical method, and restricted-focus ML nomination is not much worse. Somewhat surprising is that these methods perform well seemingly irrespective of whether or not the model parameters are known, though this phenomenon is accounted for by the fact that the smoothed estimates are automatically close to the truth, since Λ is approximately equal to the matrix with all entries 1/2. Meanwhile, the small number of nodes is such that there is little signal available to spectral vertex nomination. We see that spectral vertex nomination performs approximately at-chance regardless of whether or not we project the spectral embeddings to the sphere. 10 nodes are not enough to reveal eigenvalue structure that spectral methods attempt to recover. In the medium case, where there are 500 vertices, enough signal is present that reasonable performance is obtained by spectral vertex nomination, with performance with (purple) and without (green) projection to the sphere again indistinguishable. The comparative density of the SBM in question ensures that projection to the sphere is not necessary, and that doing so does no appreciable harm to nomination. However, in the medium case, ML-based vertex nomination still appears to best spectral methods, with the known and unknown cases being nearly indistinguishable. We note that in both the small and medium cases all of the methods appear to intersect at an empirical probability of 0.4. These intersection points correspond to the transition from the block of interest to the non-interesting vertices: these vertices, about which we are least confident, tend to be nominated correctly at or near chance, which is 40% in both the small and large cases.

A more quantitative assessment of the vertex nomination methods is contained in Tables 4.1 and 4.2, which compare the performance of the methods as assessed by, respectively, average precision (AP) and adjusted Rand index (ARI). As defined in



(a) Small scale simulation results

(b) Medium scale simulation results

Figure 4.1: The mean nomination lists for the (a) small and (b) medium stochastic block model experiments for the different vertex nomination techniques in both the known (dark colors) and unknown (light colors). Plot (a) shows performance for the canonical (gold), maximum likelihood (blue), restricted-focus maximum likelihood (red) and spectral (green and purple) methods. Spectral VN both with and without projection to the sphere is shown in purple and green, respectively. Plot (b) does not include canonical vertex nomination due to runtime constraints.

Equation (4.1), AP is a value between 0 and 1, where a value of 1 indicates perfect performance. ARI Hubert and Arabie (1985) measures how well a given partition of a set recovers some ground truth partition. Here a value of 1 indicates perfect recovery, while randomly partitioning a data set yields ARI approximately 0 (note that negative ARI is possible). We include ARI as an evaluation to highlight the fact that spectral and maximum-likelihood nomination do not merely classify vertices as interesting or not. Rather, they return a partition of the vertices into clusters. Canonical vertex nomination, on the other hand, makes no attempt to recover the full cluster structure of the graph, instead only attempting to classify vertices according to whether or not they are of interest. As such, we do not include ARI numbers for canonical vertex nomination. Turning first to performance in the small graph condition in Table 4.1, we see that $\mathcal{L}^{\mathrm{C}}$ is the best method, so long as the graph in question is small enough that the canonical method is tractable, but $\mathcal{L}^{\mathrm{ML}}$, regardless of whether or not model parameters are known, nearly matches canonical VN, and, unlike its canonical counterpart, scales to graphs with more than a few nodes. The numbers for $\mathcal{L}^{\mathrm{SP}}$ bear out our observation above, that the small graphs contain too little information for spectral VN to act upon, and $\mathcal{L}^{\mathrm{SP}}$ performs approximately at chance, as a result. It is worth noting that while $\mathcal{L}_R^{\mathrm{ML}}$ does not match the performance of $\mathcal{L}^{\mathrm{ML}}$, presumably owing to the fact that the restricted-focus algorithm does not use all of the information present in the graph, it still outperforms spectral nomination, and lags $\mathcal{L}^{\mathrm{ML}}$ by less than 0.1 AP.

Turning our attention to the medium case, we see again that $\mathcal{L}^{\mathrm{ML}}$ and $\mathcal{L}_R^{\mathrm{ML}}$ remain largely impervious to whether model parameters are known or not, presumably a consequence of the use of smoothing—we'll see in the sequel that estimation can be the difference between near-perfect performance and near-chance. With more vertices, we see that spectral improves above chance, leaving restricted ML slightly worse, but spectral still fails to match the performance of ML VN, even when model parameters are unknown.

In sum, these results suggest that different size graphs (and different modeling assumptions) call for different vertex nomination methods. In small graphs, regardless of whether or not model parameters are known, canonical vertex nomination is both tractable and quite effective. In medium graphs, maximum-likelihood vertex nomination remains tractable and achieves impressively good nomination. Of course,

	Known				Unknown			
	ML	RES	SP	CAN	ML	RES	SP	CAN
small	0.670	0.588	0.388	0.700	0.680	0.606	0.415	0.710
medium	0.954	0.545	0.738	–	0.954	0.537	0.735	–

Table 4.1: Empirical estimates of mean average precision on the two stochastic block model data sets for the four methods under consideration. Each data point is the mean of 200 independent trials.

	Known				Unknown			
	ML	RES	SP	CAN	ML	RES	SP	CAN
small	0.338	0.259	0.011	–	0.338	0.259	0.011	–
medium	0.572	0.039	0.268	–	0.572	0.037	0.271	–

Table 4.2: ARI on the different sized data sets for the ML, restricted ML, and spectral methods. Each data point is the mean of 200 independent trials. Performance of canonical vertex nomination is knot included, since canonical vertex nomination makes no attempt to recover all three blocks, and thus ARI is not a sensible measure.

for graphs with thousands of vertices, $\mathcal{L}^{\text{ML}}$ becomes computationally expensive, leaving only $\mathcal{L}^{\text{SP}}$ and $\mathcal{L}_R^{\text{ML}}$ as options. We have observed that $\mathcal{L}_R^{\text{ML}}$ tends to lag $\mathcal{L}^{\text{SP}}$ in such large graphs, though increasing the number of seeds (and hence the amount of information available to $\mathcal{L}_R^{\text{ML}}$) closes this gap considerably. We leave for future work a more thorough exploration of under what circumstances we might expect $\mathcal{L}_R^{\text{ML}}$ to be competitive with $\mathcal{L}^{\text{SP}}$ in graphs on thousands of vertices.

4.6.2 Word Co-occurrences

We consider a linguistic data set consisting of co-occurrences of 54 nouns and 58 adjectives in Charles Dickens' novel *David Copperfield* Newman (2006a). We construct a graph in which each node corresponds to a word, and an edge connects

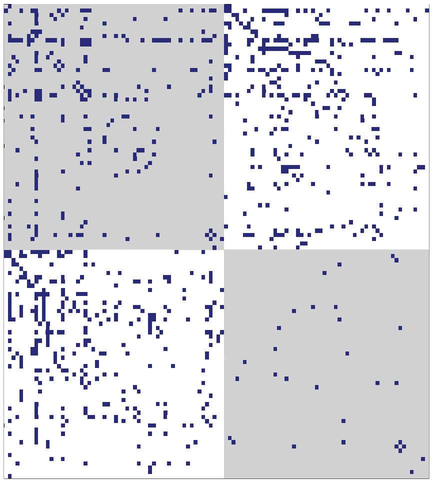
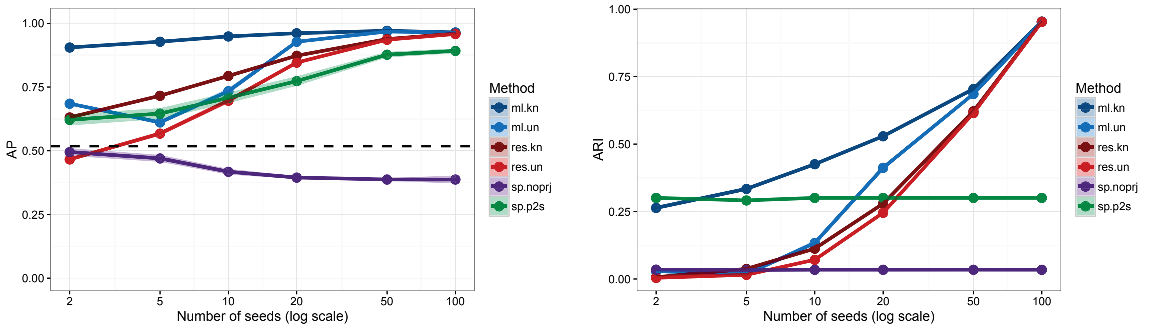


Figure 4.2: Adjacency matrix of the linguistic data set, arranged to highlight the graph's structure. The grey shading indicates the two blocks, with adjectives in the upper left and nouns in the lower right. Note the disassortative block structure.

two nodes if the two corresponding words occurred adjacent to one another in the text. The adjacency matrix of this graph is shown in Figure 4.2. Visual inspection reveals a clear block structure, and that this block structure is clearly not assortative (i.e., inter-block edges are more frequent than intra-block edges). This runs contrary to many commonly-studied data sets and model assumptions. Figure 4.3 shows the performance of spectral and maximum-likelihood vertex nomination, measured by (a) average precision and adjusted Rand index (ARI) at various numbers of seeds. Each data point is the average over 1000 trials. In each trial, a set of m seeds was chosen uniformly at random from the 112 nodes, with the restriction that at least one noun and one adjective be included in the seed set. Performance was then measured as the mean average precision in identifying the adjective block.

Figure 4.3 shows the performance of the VN schemes under consideration, as a function of the number of seed vertices, using both known (dark colors) and estimated (light colors) model parameters. Looking first at AP in Figure 4.3 (a), we see that ML in the known-parameter case (dark blue) does consistently well, even with only a handful of seeds, and attains near-perfect performance for $m \geq 20$. When model parameters must be estimated (light blue), ML is less dominant, thought it still performs nearly perfectly for $m \geq 20$. We note the dip in unknown-parameters ML as m increases from 2 to 5 to 10, a phenomenon we attribute to the bias-variance tradeoff. Namely, with more seeds available, variance in the estimated model parameters increases, but for $m < 20$, this increase in variance is not offset by an appreciable improvement in estimation, possibly attributable to our use of add-one smoothing. Somewhat surprisingly, restricted-focus ML performs quite well, consistently improving on spectral VN in the known parameter case for $m > 2$, and in the unknown parameter case once $m > 10$. Finally, we turn our attention to spectral VN, shown in green for the variant in which we project embeddings to the sphere and in purple for the variant in which we do not. In contrast to our simulations, the sparsity of this network makes projection to the sphere a critical requirement for successful retrieval of the first block. Without projection to the sphere, spectral VN fails to rise appreciably above chance performance.



(a) **(b)**

Figure 4.3: Performance on the linguistic data set as measured by (a) AP and (b) ARI as a function of the number of seeds for the ML vertex nomination (blue), restricted-focus ML (red), and spectral vertex nomination with (green) and without projection to the sphere (violet), when model parameters are known (light colors) and unknown (dark colors). Each data point is the mean of 1000 Monte Carlo trials, and shaded regions indicate two standard deviations of the mean.

4.6.3 Zachary's Karate Club

We consider the classic sociological data set, Zachary's karate club network Zachary (1977). The graph, visualized in Figure 4.4, consists of 34 nodes, each corresponding to a member of a college karate club, with edges joining pairs of club members according to whether or not those members were observed to interact consistently outside of the club. Over the course of Zachary's observation of the group, a conflict emerged that led to the formation of two factions, led by the individuals numbered 1 and 34 in Figure 4.4, and these two factions constitute the two blocks in this experiment. Zachary's karate data set is particularly well-suited for spectral methods. Indeed, the flow-based model originally proposed by Zachary recovers factions nearly perfectly, and visual inspection of the graph (Figure 4.4) suggests a natural cut separating the two factions. As such, we expect ML-based vertex nomination to lose out against

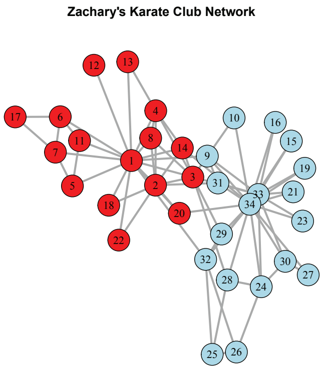
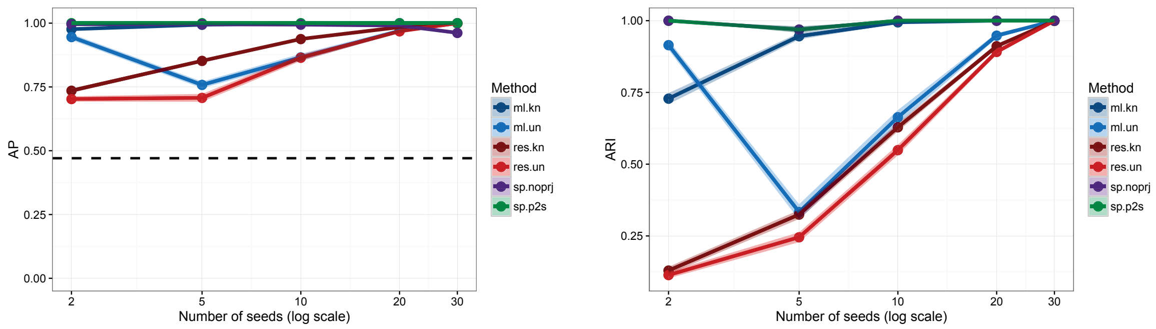


Figure 4.4: Visualization of the graph corresponding to Zachary's karate club data set. The vertices are colored according to which of the two clubs each member chose to join after the schism. Our block of interest is in red.

the spectral-based method. Figure 4.5 shows performance of the two algorithms as measured by ARI and average precision. We see, as expected, that spectral performance performs nearly perfectly, irrespective of the number of seeds. Surprisingly, maximum-likelihood nomination is largely competitive with spectral VN, but only provided that the model parameters are already known. Interesting to note that here again we see the phenomenon discussed previously in which ML performance with an unknown edge probability matrix degrades when going from $s = 2$ seeds to $s = 5$ before improving again, with AP comparable to the known case for $s \geq 20$.

4.6.4 Political Blogs

We consider a network of American political blogs in the lead-up to the 2004 election Adamic and Glance (2005), where an edge joins two blogs if one links to the other, with blogs classified according to political leaning (liberal vs conservative).

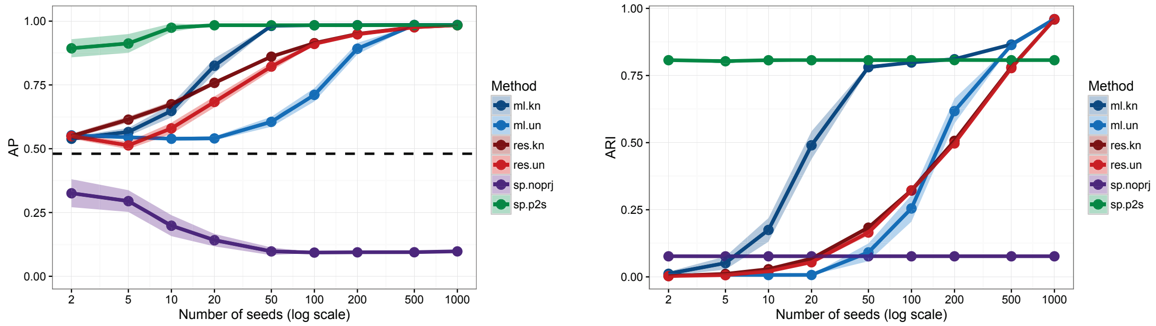


(a)

(b)

Figure 4.5: Performance on the karate data set as a function of the number of seeds for the ML vertex nomination (blue), restricted-focus ML nomination (red), and spectral vertex nomination with (green) and without projection to the sphere (violet), when model parameters are known (light colors) and unknown (dark colors), as measured by (a) AP and (b) ARI. The black dashed line indicates chance performance. Each observation is the mean of 1000 independent trials, with the shaded bars indicating two standard errors of the mean in either direction.

From an initial 1490 vertices, we removed all isolated vertices to obtain a network of 1224 vertices and 16718 edges. Figure 4.6 shows the performance of the spectral- and ML-based methods in recovering the liberal block. We observe first and foremost that the sparsity of this network results in exceptionally poor performance in both AP and ARI for spectral VN unless the embeddings are projected to the sphere, but that spectral vertex nomination is otherwise quite effective at recovering the liberal block, with performance nearly perfect for $m > 10$. Unsurprisingly, ML and its restricted counterpart both perform approximately at-chance when $m < 10$. We see that in both the known and unknown cases, ML VN is competitive with spectral VN for suitably large m ($m \geq 50$ for known, $m \geq 500$ for unknown). As expected in such a sparse network, restricted-focus ML lags ML VN in the known-parameter case, but surprisingly, in the unknown-parameter case, restricted ML achieves remarkably



(a) **(b)**

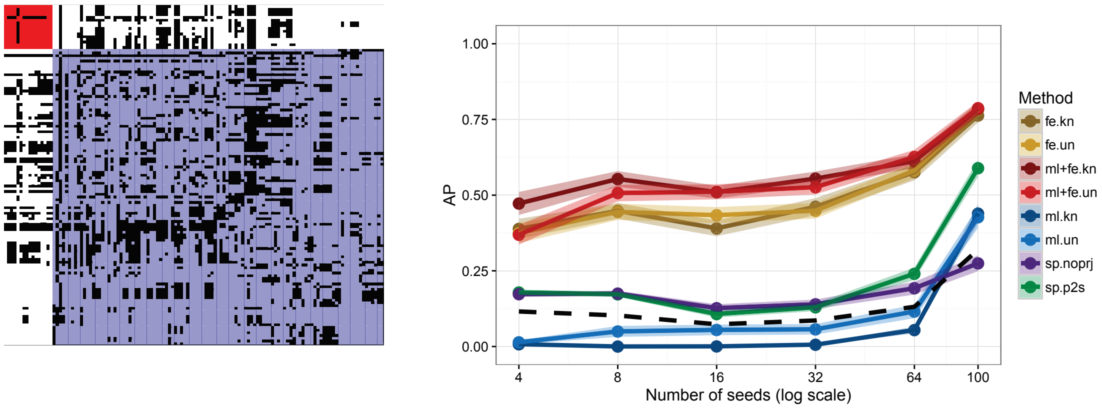
Figure 4.6: Performance on the political blogs data set as a function of the number of seeds for the ML vertex nomination (blue), restricted-focus ML (red), and spectral vertex nomination with (green) and without projection to the sphere (violet), when model parameters are known (light colors) and unknown (dark colors), as measured by (a) AP and (b) ARI.

better AP than does ML, a fact we are unable to account for, though it is worth noting that looking at ARI in Figure 4.6 (b), no such gap appears between ML and its restricted-focus counterpart in the unknown-parameter case.

4.6.5 Ecological Network

We consider a trophic network, consisting of 125 nodes and 1907 edges, in which nodes correspond to (groups of) organisms in the Florida Bay ecosystem Ulanowicz et al. (1997); Nooy et al. (2011), and an edge joins a pair of organisms if one feeds on the other. Our features are the (log) mass of organisms. We take our community of interest to be the 16 different types of birds in the ecosystem. This choice makes for an interesting task for several reasons. Firstly, unlike the other data sets we consider, our community of interest is a comparatively small fraction of the network—it consists of a mere 16 nodes of 125 in total. Further, our block of interest is comparatively heterogeneous in the sense that the roles of the different types of birds in the Florida Bay ecosystem is quite diverse. For example, the block of interest includes both raptors and shorebirds, which feed on quite different collections of organisms. Finally, it stands to reason that the mass of the organisms in question might be a crucial piece of information for disambiguating, say, a raptor from a shark. Thus, we expect that using node features will be crucial for retrieving the block of interest.

The topology of the Florida Bay network is shown in Figure 4.7 (a). Note that the block of interest, indicated in red, has a strongly disassortative structure. Indeed, all intra-block edges in the red block are incident to the node corresponding to raptors. Figure 4.7 (b) summarizes vertex nomination performance for several methods. The plot shows performance, as measured by mean average precision (AP), as a function of the number of seeds for several different nomination schemes. As in earlier plots, dark colors correspond to model parameters being known, while light colors correspond to model parameters being estimated using the seed vertices. We see immediately that spectral nomination (green and purple) and ML VN (blue) fail to improve appreciably upon chance performance except when the vast majority of the vertices' labels are observed. Like in the linguistic data set presented above, the disassortative structure of the data appears to cause problems for spectral nomination. The failure of ML suggests that no useful information is encoded in the graph itself, but turning our attention to the curves corresponding to $\mathcal{L}_F^{\mathrm{ML}}$ (red) and using only features (gold), we



(a) **(b)**

Figure 4.7: (a) The adjacency matrix of the Florida Bay trophic network. Nodes correspond to classes of plants and animals (e.g., sharks, rays, shorebirds, zooplankton, phytoplankton). An edge joins two nodes if the corresponding organisms are in a predator-prey relation. The sixteen types of birds in the network are highlighted in the red block. Note the disassortative structure of the bird block (the edges within the red block are all incident to the node that corresponds to raptors). (b) Average precision in identifying the bird nodes as a function of the number of seed vertices for ML vertex nomination (blue), restricted-focus ML (red), and spectral vertex nomination with (green) and without projection to the sphere (violet), when model parameters are known (light colors) and unknown (dark colors). The black dashed line indicates chance performance.

see that this is not the case. Indeed, we see that while using features alone achieves a marked improvement over both spectral and ML-based nomination, using both features and graph matching in the form of $\mathcal{L}_F^{\mathrm{ML}}$ yields an additional improvement of some 0.1 AP in the range of $m = 8, 16, 32$. This result suggests that there may be cases where the only reliable way to retrieve vertices of interest is to leverage both features and graph topology jointly.

4.7 Discussion and Future Work

Network data has become ubiquitous in the sciences, giving rise to a vast array of computational and statistical problems that are only beginning to be explored. In this chapter, we have explored one such problem that arises when working with network data, namely the task of performing vertex nomination. This task, in some sense the graph analogue of the classic information retrieval problem, is fundamental to exploratory data analysis on graphs as well as to machine learning applications. Above, we established the consistency of two methods of vertex nomination: a maximum-likelihood scheme $\mathcal{L}^{\mathrm{ML}}$ and its restricted-focus variant $\mathcal{L}_R^{\mathrm{ML}}$, in which we obtain a feasibly exactly-solvable optimization problem at the expense of using less than the full information available in the graph. Additionally, we have introduced a maximum-likelihood nomination scheme for the case where vertices are endowed with features and when (possibly weighted) edges are drawn from a canonical exponential family. The key to all of these methods is the ability to quickly approximate a solution to the seeded graph matching problem.

We have presented experimental comparisons of these methods against each other and against several other benchmark methods, where we see that the best choice of method depends highly on graph size and structure. The major tradeoff appears to be that large graphs (tens of thousands of vertices) are not tractable for $\mathcal{L}^{\mathrm{ML}}$, but in smaller and medium-sized graphs, $\mathcal{L}^{\mathrm{ML}}$ can detect signal where spectral methods fail to do so. It is worth noting that $\mathcal{L}^{\mathrm{ML}}$, and, to a lesser extent, $\mathcal{L}_R^{\mathrm{ML}}$, is quite competitive with $\mathcal{L}^{\mathrm{SP}}$, and even manages to best $\mathcal{L}^{\mathrm{SP}}$ when the structure of the graph is ill-suited to the typical assumptions of spectral methods, as in the case of our linguistic data set. All told, our experimental results mirror those in Fishkind et al. (2015) and point toward a theory of which methods are best-suited to which graphs, a direction that warrants further exploration.

In the next chapter, we will see that the vertex nomination technique developed here can be brought to bear on the reranking problem in similarity search.

4.8 Proof details

Before proving Theorem 4, we first state a useful lemma.

Lemma 11. *Let $\vec{x} = (x_1, x_2, \dots, x_k)$ be a vector with distinct entries in $\mathbb{R}^k$. Let $f(\cdot)$ be a strictly increasing real valued function (with the abuse of notation, $f(\vec{x})$, denoting $f(\cdot)$ applied entry-wise to $\vec{x}$). Let the order statistics of $\vec{x}$ be denoted*

$$x_{(1)} < x_{(2)} < \cdots < x_{(k)},$$

and define $\alpha = \min_{i\in\{2,3,\dots,k\}} |x_{(i)} - x_{(i-1)}|$, and $\beta = \min_{i\in\{2,3,\dots,k\}} |f(x_{(i)}) - f(x_{(i-1)})|$. If σ is the cyclic permutation

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & \cdots & k \\ 2 & 3 & 4 & \cdots & 1 \end{pmatrix},$$

then

$$\langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\sigma(\vec{x}))\rangle \geq (k-1)\alpha\beta.$$

Proof. We will induct on k. To establish the base case, $k=2$, let $x_1 = x_{(1)}$ without loss of generality and observe that

$$\begin{aligned}\langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\sigma(\vec{x}))\rangle &= (x_2 - x_1)(f(x_2) - f(x_1))\\ &= (x_{(2)} - x_{(1)})(f(x_{(2)}) - f(x_{(1)})) \geq \alpha\beta.\end{aligned}$$

For general k, again, without loss of generality let $x_1 = x_{(1)}$, and define the permutation

$$\tau = \begin{pmatrix} 2 & 3 & \cdots & k \\ 3 & 4 & \cdots & 2 \end{pmatrix}.$$

Then

$$\begin{aligned}\langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\sigma(\vec{x}))\rangle &= \langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\tau(\vec{x}))\rangle + \langle \vec{x}, f(\tau(\vec{x}))\rangle - \langle \vec{x}, f(\sigma(\vec{x}))\rangle\\ &= \langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\tau(\vec{x}))\rangle + (x_k - x_1)(f(x_2) - f(x_1))\\ &\geq \langle \vec{x}, f(\vec{x})\rangle - \langle \vec{x}, f(\tau(\vec{x}))\rangle + \alpha\beta,\end{aligned}$$

and the result follows from the inductive hypothesis. □

Remark 8. It follows immediately that in Lemma 11, if there exists an index $i \in [k]$ such that $\alpha_i = \min_{j\neq i} |x_{(i)} - x_{(j)}| > 0$, and $\beta_i = \min_{j\neq i} |f(x_{(i)}) - f(x_{(j)})| > 0$, then

$\langle \vec{x}, f(\vec{x}) \rangle - \langle \vec{x}, f(\sigma(\vec{x})) \rangle \geq \alpha_i \beta_i$.

We are now ready to prove Theorem 4.

Proof of Theorem 4. Define

$$X_P := \operatorname{tr}(AB^\top) - \operatorname{tr}(A(I_m \oplus P)B(I_m \oplus P)^\top)$$

and define $\mathcal{P} = \{P \in \Pi_{\mathfrak{u}} : \epsilon_{1,\bullet}(P) > 0\}$. We will show that

$$\mathbb{P}\left(\exists\ P \in \mathcal{P} \text{ s.t. } X_P \leq 0\right) = O(1/n^2),$$

from which the desired consistency of $\mathcal{L}^{\mathrm{ML}}$ follows by the Borel-Cantelli Lemma, since this probability is summable in n. Fix $P \in \mathcal{P}$, and let $\sigma_P \in S_n$ be the permutation associated with $I_m \oplus P$. The action of shuffling B via $I_m \oplus P$ is equivalent to permuting the $[n^2]$ elements of $\operatorname{vec}(B)$ via a permutation τ_P, in that

$$\operatorname{tr}(A(I_m \oplus P)B(I_m \oplus P)^\top) = \langle \operatorname{vec}(A), \tau_P(\operatorname{vec}(B)) \rangle.$$

Moreover, τ_P can be chosen so that, in the cyclic decomposition of $\tau_P = \tau_P^{(1)} \tau_P^{(2)} \cdots \tau_P^{(\ell)}$, each (disjoint) cycle is acting on a set of distinct real numbers. Note that Lemma 11 implies that the contribution of each cycle $\tau_P^{(i)}$ to $\mathbb{E}(X_P)$ is nonnegative, and the assumptions of Theorem 4 imply that for each $i, j \in [K]$ such that $i \neq j$, the contribution of each (nontrivial) cycle permuting a $\Lambda_{i,i}$ entry to a $\Lambda_{i,j}$ entry contributes at least $\alpha\beta$ to $\mathbb{E}(X_P)$. It follows immediately that

$$\begin{aligned}
\mathbb{E}(X_P) &= \mathbb{E}\left(\mathrm{tr}(AB) - \mathrm{tr}(APBP^\top)\right) \\
&= \mathbb{E}\left(\langle \mathrm{vec}(A), \mathrm{vec}(B)\rangle - \langle \mathrm{vec}(A), \tau_P(\mathrm{vec}(B))\rangle\right) \\
&\geq 2\alpha\beta \sum_i \left(\frac{1}{2}\sum_j \sum_{k\neq j} \epsilon_{i,j}\epsilon_{i,k} + m_i\epsilon_{i,\bullet}\right) \\
&\geq 2\alpha\beta \sum_i \left(\frac{(\mathfrak{u}_i - \epsilon_{i,\bullet})\epsilon_{i,\bullet}}{2} + m_i\epsilon_{i,\bullet}\right).
\end{aligned}$$

Let $\mathfrak{n}(P)$ be the total number of distinct entries of $\mathrm{vec}(B)$ permuted by τ_P, and note that an application of Lemma 11 yields

$$\begin{aligned}
\mathbb{E}(X_P) &= \mathbb{E}\left(\mathrm{tr}(AB) - \mathrm{tr}(APBP^\top)\right) \\
&= \mathbb{E}\left(\langle \mathrm{vec}(A), \mathrm{vec}(B)\rangle - \langle \mathrm{vec}(A), \tau_P(\mathrm{vec}(B))\rangle\right) \\
&\geq \frac{1}{2}\mathfrak{n}(P)\gamma\kappa.
\end{aligned}$$

The assumptions in the Theorem also immediately yield that

$$\mathfrak{n}(P) \geq \sum_k \left(\frac{(\mathfrak{u}_k - \epsilon_{k,\bullet})\epsilon_{k,\bullet}}{2} + m_k\epsilon_{k,\bullet}\right).$$

We next note that X_P is a sum of $\mathfrak{n}(P)$ independent random variables, each bounded in $[-c, c]$. An application of Hoeffding's inequality then yields

$$\begin{aligned}\mathbb{P}(X_P \leq 0) &\leq \mathbb{P}\left(|X_P - \mathbb{E}X_P| \geq \mathbb{E}X_P\right) \leq 2\exp\left\{-\frac{2\mathbb{E}^2 X_P}{4c^2\mathfrak{n}(P)}\right\}\\ &\leq 2\exp\left\{-\frac{|\mathbb{E}X_P|\kappa\gamma}{2c^2}\right\} \leq 2\exp\left\{-\frac{\alpha\beta\kappa\gamma}{4c^2}\sum_k\left(\frac{(\mathfrak{u}_k - \epsilon_{k,\bullet})\epsilon_{k,\bullet}}{2} + m_k\epsilon_{k,\bullet}\right)\right\}.\end{aligned}$$

Next, note that

$$|\{P \in \mathcal{P} \text{ s.t. } X_P \leq 0\}| = 0 \text{ iff } |\{P \in \mathcal{P}/\sim \text{ s.t. } X_P \leq 0\}| = 0.$$

Given $\{\epsilon_{k,\ell}\}_{k,\ell=1}^K$ satisfying $\mathfrak{u}_k = \sum_\ell \epsilon_{k,\ell} = \sum_\ell \epsilon_{\ell,k}$ for all $k \in [K]$, the number of elements $P \in \mathcal{P}/\sim$ with $\epsilon_{k,\ell}(P) = \epsilon_{k,\ell}$ for all $k, \ell \in [K]$ is at most

$$\begin{aligned}\mathfrak{u}_1^{\sum_{\ell\neq 1}\epsilon_{1,\ell}}\mathfrak{u}_2^{\sum_{\ell\neq 2}\epsilon_{2,\ell}}\cdots\mathfrak{u}_K^{\sum_{\ell\neq K}\epsilon_{K,\ell}} &= \mathfrak{u}_1^{\mathfrak{u}_1-\epsilon_{1,1}}\mathfrak{u}_2^{\mathfrak{u}_2-\epsilon_{2,2}}\cdots\mathfrak{u}_K^{\mathfrak{u}_K-\epsilon_{K,K}}\\ &= e^{\sum_k(\mathfrak{u}_k-\epsilon_{k,k})\log(\mathfrak{u}_k)}. \qquad (4.12)\end{aligned}$$

The number of ways to choose such a set (i.e. the $\{\epsilon_{k,\ell}\}_{k,\ell}^K$) is bounded above by

$$\prod_{k \text{ s.t. } \epsilon_{k,\bullet}\neq 0}(\mathfrak{u}_k + K)^K = e^{\sum_{k \text{ s.t. } \epsilon_{k,\bullet}\neq 0} K\log(\mathfrak{u}_k+K)}. \qquad (4.13)$$

Applying the union bound over all $P \in \mathcal{P}/\sim$, we then have

$$\begin{align}
&\mathbb{P}\big(\exists\, P \in \mathcal{P} \text{ s.t. } X_P \le 0\big) = \mathbb{P}\big(\exists\, P \in \mathcal{P}/\sim \text{ s.t. } X_P \le 0\big) \notag\\
&\le \exp\Bigg\{ -\frac{\alpha\beta\kappa\gamma}{2c^2}\sum_k \bigg(\frac{(\mathfrak{u}_k - \epsilon_{k,\bullet})\epsilon_{k,\bullet}}{2} + m_k\epsilon_{k,\bullet}\bigg) \tag{4.14}\\
&\qquad\qquad + \sum_k (\mathfrak{u}_k - \epsilon_{k,k})\log \mathfrak{u}_k + \sum_{k \text{ s.t. } \epsilon_{k,\bullet}\neq 0} K\log(\mathfrak{u}_k + K)\Bigg\}. \tag{4.15}
\end{align}$$

It remains for us to establish that the expression inside the exponent goes to $-\infty$ fast enough to ensure our desired bound. For each k, the contribution to the exponent in (4.14) is

$$\begin{align}
&-\frac{\alpha\beta\kappa\gamma}{2c^2}\bigg(\frac{(\mathfrak{u}_k - \epsilon_{k,\bullet})\epsilon_{k,\bullet}}{2} + m_k\epsilon_{k,\bullet}\bigg) + (\mathfrak{u}_k - \epsilon_{k,k})\log\mathfrak{u}_k + \mathbb{I}\{\epsilon_{k,\bullet}\neq 0\}K\log(\mathfrak{u}_k + K) \notag\\
&= -\frac{\alpha\beta\kappa\gamma}{2c^2}\bigg(\frac{\epsilon_{k,k}\epsilon_{k,\bullet}}{2} + m_k\epsilon_{k,\bullet}\bigg) + \epsilon_{k,\bullet}\log\mathfrak{u}_k + \mathbb{I}\{\epsilon_{k,\bullet}\neq 0\}K\log(\mathfrak{u}_k + K) \tag{4.16}
\end{align}$$

If $\mathfrak{u}_k/2 \le \epsilon_{k,k} < \mathfrak{u}_k$, then

$$\epsilon_{k,k}\epsilon_{k,\bullet} \ge \frac{\mathfrak{u}_k\epsilon_{k,\bullet}}{2} = \omega(\epsilon_{k,\bullet}\log\mathfrak{u}_k), \text{ and } \epsilon_{k,k}\epsilon_{k,\bullet} \ge \frac{\mathfrak{u}_k\epsilon_{k,\bullet}}{2} = \omega(K\log(\mathfrak{u}_k + K)),$$

and the contribution to the exponent in (4.14) from k, Eq. (4.16), is clearly bounded above by $-2\log(n)$ for sufficiently large n. If $\epsilon_{k,k} \le \mathfrak{u}_k/2$ then $\epsilon_{k,\bullet} > \mathfrak{u}_k/2$, and

$$m_k\epsilon_{k,\bullet} = \omega(\epsilon_{k,\bullet}\log\mathfrak{u}_k), \text{ and } m_k\epsilon_{k,\bullet} \ge \frac{m_k\mathfrak{u}_k}{2} = \omega(K\log(\mathfrak{u}_k + K)),$$

and the contribution to the exponent in (4.14) from k, Eq. (4.16), is clearly bounded above by $-2\log(n)$ for sufficiently large n. If $\epsilon_{k,k} = \mathfrak{u}_k$, then all terms in the exponent (4.16) are equal to 0. For sufficiently large n, Eq. (4.14) is then bounded above by

$$\exp\left\{ - \sum_{k \text{ s.t. } \epsilon_{k,\bullet} \neq 0} 2\log(n) \right\} \leq \exp\left\{-2\log(n)\right\},$$

and the result follows. □

Consistency of $\mathcal{L}_R^{\mathrm{ML}}$ as claimed in Theorem 5 follows similarly to that of $\mathcal{L}^{\mathrm{ML}}$, and we next briefly sketch the details of the proof.

Proof of Theorem 5 (Sketch). Analogously to the proof of Theorem 4, define

$$X_P := \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)}\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right).$$

The proof follows *mutatis mutandis* to the proof of Theorem 4, with the key difference being that in this case,

$$\begin{aligned}\mathbb{E}(X_P) &= \mathbb{E}\left(\operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)}\right) - \operatorname{tr}\left((A^{(1,2)})^\top B^{(1,2)} P^\top\right)\right) \\ &\geq 2\alpha\beta \sum_k m_k \epsilon_{k,\bullet}.\end{aligned}$$

Details are omitted for brevity. □

Before proving Theorem 6 we establish some preliminary concentration results for our estimates $\widehat{\Lambda}$, and $\hat{n}_k$, $k \in [K]$. An application of Hoeffding's inequality yields that for $k, \ell \in [K]$ such that $k \neq \ell$,

$$\mathbb{P}\left(\left|\widehat{\Lambda}_{k,\ell} - \Lambda_{k,\ell}\right| \geq \frac{\sqrt{n \log n}}{m_k m_\ell}\right) \leq 2\text{exp}\left\{-2n \log n\right\}, \tag{4.17}$$

and for $k \in [K]$,

$$\mathbb{P}\left(\left|\widehat{\Lambda}_{k,k} - \Lambda_{k,k}\right| \geq \frac{\sqrt{n \log n}}{\binom{m_k}{2}}\right) \leq 2\text{exp}\left\{-2n \log n\right\}, \tag{4.18}$$

and

$$\mathbb{P}\left(\left|\hat{n}_k - n_k\right| \geq t\right) \leq 2\text{exp}\left\{\frac{-2mt^2}{n^2}\right\}, \tag{4.19}$$

With γ defined as in (4.8), define the events $\mathcal{E}_n^{(1)}$ and $\mathcal{E}_n^{(2)}$ via

$$\mathcal{E}_n^{(1)} = \left\{\forall\ \{k, \ell\} \in \binom{[K]}{2},\ \text{s.t}\ |\Lambda_{k,k} - \Lambda_{k,\ell}| > \gamma,\ \text{it holds that}\ \left|\widehat{\Lambda}_{k,k} - \widehat{\Lambda}_{k,\ell}\right| > \frac{\gamma}{2}\right\};$$

$$\mathcal{E}_n^{(2)} = \left\{\forall\ k \in [K],\ |\hat{n}_k - n_k| \leq n_k^{2/3}\right\}.$$

Combining (4.17)—(4.19), we see that if for each $k \in [K]$, $n_k = \Theta(n)$, $\min_k m_k = \omega(\sqrt{n_k} \log(n_k))$, then for sufficiently large n,

$$\mathbb{P}\left((\mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)})^c\right) \leq e^{-2 \log n}. \tag{4.20}$$

We are now ready to prove Theorem 6, proving the consistency of $\mathcal{L}^{\text{ML}}$ when the model parameters are unknown.

Proof of Theorem 6. Let $\widehat{B}$ be our estimate of B using the seed vertices; i.e., there are $\hat{n}_k$ vertices from block k for each $k \in [K]$, and for each $k, \ell \in [K]$, the entry of $\widehat{B}$ between a block k vertex and a block ℓ vertex is

$$\log\left(\frac{\widehat{\Lambda}_{k,\ell}}{1-\widehat{\Lambda}_{k,\ell}}\right).$$

Let $\widehat{L}$ be the set of distinct entries of $\widehat{\Lambda}$, and define

$$\hat{\alpha} = \min_{\{k,\ell\} \text{ s.t. } k\neq\ell} |\widehat{\Lambda}_{k,k} - \widehat{\Lambda}_{k,\ell}| \quad \hat{\beta} = \min_{\{k,\ell\} \text{ s.t. } k\neq\ell} |\widehat{B}_{k,k} - B_{k,\ell}| \quad \hat{c} = \max_{i,j,k,\ell} |\widehat{B}_{i,j} - \widehat{B}_{k,\ell}|, \tag{4.21}$$

$$\hat{\gamma} = \min_{x,y\in\widehat{L}} |x-y|, \quad \hat{\kappa} = \min_{x,y\in\widehat{L}} \left|\log\left(\frac{x}{1-x}\right) - \log\left(\frac{y}{1-y}\right)\right|. \tag{4.22}$$

Note that conditioning on $\mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)}$ and assumption *iv.* ensures that each of $\hat{\alpha}$, $\hat{\beta}$, $\hat{c}$, $\hat{\gamma}$, and $\hat{\kappa}$ is bounded away from 0 by an absolute constant for sufficiently large n. For each $k \in [K]$, define

$$\mathfrak{e}_k := |\hat{n}_k - n_k| = |\hat{\mathfrak{u}}_k - \mathfrak{u}_k|, \quad \mathfrak{e} = \sum_k \mathfrak{e}_k, \quad \eta_k := \min(n_k, \hat{n}_k), \quad \eta = \sum_k \eta_k, \tag{4.23}$$

and note that conditioning on $\mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)}$ ensures that $\mathfrak{e}_k = O(n_k^{2/3})$ for all $k \in [K]$. An immediate result of this is that, conditioning on $\mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)}$, we have that $\eta_k =$

$\Theta(n_k) = \Theta(n)$ for all $k \in [K]$.

Define $\mathcal{P} := \{P \in \Pi_{\mathfrak{u}} : \epsilon_{1,\bullet}(P) > n^{2/3}\log n\}$, and for $P \in \Pi_{\mathfrak{u}}$, define

$$X_P := \operatorname{tr}(A\tilde{B}^\top) - \operatorname{tr}(A(I_m \oplus P)\tilde{B}(I_m \oplus P)^\top).$$

We will show that

$$\mathbb{P}\left(\exists\ P \in \mathcal{P} \text{ s.t. } X_P \leq 0\right) = O(1/n^2),$$

and the desired consistency of $\mathcal{L}^{\mathrm{ML}}$ follows immediately. To this end, decompose A and B as

$$A = \begin{array}{cc} & \begin{array}{cc} \eta & \mathfrak{e} \end{array} \\ \begin{array}{c} \eta \\ \mathfrak{e} \end{array} & \begin{bmatrix} A^{(c,c)} & A^{(c,e)} \\ A^{(e,c)} & A^{(e,e)} \end{bmatrix} \end{array} \qquad B = \begin{array}{cc} & \begin{array}{cc} \eta & \mathfrak{e} \end{array} \\ \begin{array}{c} \eta \\ \mathfrak{e} \end{array} & \begin{bmatrix} B^{(c,c)} & B^{(c,e)} \\ B^{(e,c)} & B^{(e,e)} \end{bmatrix} \end{array},$$

where $A^{(c,c)}$ (resp., $B^{(c,c)}$) is an $\eta \times \eta$ submatrix of A (resp., B)—which contains the seed vertices in A—with exactly η_k vertices (resp., labels) from block k for each $k \in [K]$. We view $A^{(c,c)}$ as the "core" matrix of A (with $A^{(e,e)}$ and $A^{(c,e)}$ being the "errorful" part of A), as $A^{(c,c)}$ is a submatrix of A that we could potentially cluster perfectly along block assignments. Note that similarly decomposing P as

$$P = \begin{array}{cc} & \begin{array}{cc} \eta & \mathfrak{e} \end{array} \\ \begin{array}{c} \eta \\ \mathfrak{e} \end{array} & \begin{bmatrix} P^{(c,c)} & P^{(c,e)} \\ P^{(e,c)} & P^{(e,e)} \end{bmatrix} \end{array},$$

we see that there exists a principal permutation submatrix of $P^{(c,c)}$ of size $(\eta-2\mathfrak{e})\times(\eta-2\mathfrak{e})$, which we denote $\tilde{P}$ (with associated permutation $\tilde{\sigma}$). This matrix represents a subgraph of the core vertices of A mapped to a subgraph of the core vertices in B. We can then write $P=\tilde{P}\oplus Q$, where $Q\in\Pi_{3\mathfrak{e}}$. For each $k,\ell\in[K]$, let

$$\tilde{\epsilon}_{k,\ell}=\tilde{\epsilon}_{k,\ell}(\tilde{P})=|\{v\in U_k \text{ s.t. } \tilde{\sigma}(v)\in U_k\}|$$

Consider now

$$X_P=\operatorname{tr}(A(I_{\eta-3\mathfrak{e}}\oplus Q)B(I_{\eta-3\mathfrak{e}}\oplus Q)^\top)-\operatorname{tr}(A(\tilde{P}\oplus Q)B(\tilde{P}\oplus Q)^\top). \tag{4.24}$$

Letting $\tilde{\mathfrak{u}}_k$ denote the number of vertices from the k-th block acted on by $\tilde{P}$, our assumptions yield

$$\mathbb{E}(X_P)\geq 2\hat{\alpha}\hat{\beta}\sum_k\left(\frac{(\tilde{\mathfrak{u}}_k-\tilde{\epsilon}_{k,\bullet})\tilde{\epsilon}_{k,\bullet}}{2}+m_k\tilde{\epsilon}_{k,\bullet}\right)-\Theta(\eta\mathfrak{e})-\Theta(\mathfrak{e}^2).$$

Let $\tilde{\mathfrak{n}}(P)$ be the total number of distinct entries of $\operatorname{vec}(B^{(c,c)})$ permuted by $\tilde{P}$, and note that another application of Lemma 11 yields

$$\mathbb{E}(X_P)\geq\frac{1}{2}\tilde{\mathfrak{n}}(P)\hat{\gamma}\hat{\kappa}-\Theta(\eta\mathfrak{e})-\Theta(\mathfrak{e}^2).$$

The assumptions in the Theorem also immediately yield that

$$\tilde{\mathfrak{n}}(P) \geq \sum_k \left(\frac{(\tilde{\mathfrak{u}}_k - \tilde{\epsilon}_{k,\bullet})\tilde{\epsilon}_{k,\bullet}}{2} + m_k \tilde{\epsilon}_{k,\bullet} \right).$$

We then have that there exists a constants $c_1 > 0$ and $c_2 > 0$ such that

$$\begin{aligned}
&\mathbb{P}\big(\,\exists\, P \in \mathcal{P} \text{ s.t. } X_P \leq 0 \,\big|\, \mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)}\big) = \mathbb{P}\big(\,\exists\, P \in \mathcal{P}/\sim \text{ s.t. } X_P \leq 0 \,\big|\, \mathcal{E}_n^{(1)} \cup \mathcal{E}_n^{(2)}\big) \\
&\leq \exp\Bigg\{ -\frac{\hat{\alpha}\hat{\beta}\hat{\kappa}\hat{\gamma}}{2\hat{c}^2} \sum_k \left(\frac{(\tilde{\mathfrak{u}}_k - \tilde{\epsilon}_{k,\bullet})\tilde{\epsilon}_{k,\bullet}}{2} + m_k \tilde{\epsilon}_{k,\bullet} \right) + \Theta(\eta\mathfrak{e}) + \Theta(\mathfrak{e}^2) \\
&\qquad + \sum_k (\tilde{\mathfrak{u}}_k - \tilde{\epsilon}_{k,k}) \log \tilde{\mathfrak{u}}_k + \sum_{k \text{ s.t. } \tilde{\epsilon}_{k,\bullet} \neq 0} K \log(\tilde{\mathfrak{u}}_k + K) + O(\mathfrak{e} \log \mathfrak{e}) \Bigg\} \\
&= \exp\Bigg\{ -c_1 \sum_k \left(\frac{(\tilde{\mathfrak{u}}_k - \tilde{\epsilon}_{k,\bullet})\tilde{\epsilon}_{k,\bullet}}{2} + m_k \tilde{\epsilon}_{k,\bullet} \right) \qquad (4.25) \\
&\qquad + \sum_k \tilde{\epsilon}_{k,\bullet} \log \tilde{\mathfrak{u}}_k + \sum_{k \text{ s.t. } \tilde{\epsilon}_{k,\bullet} \neq 0} K \log(\tilde{\mathfrak{u}}_k + K) + \Theta(n\mathfrak{e}) \Bigg\} \\
&\leq \exp\{-c_2 n^{7/4} \log n\}. \qquad (4.26)
\end{aligned}$$

Unconditioning Equation (4.25) combined with Equation (4.20) yields the desired result. □

Proof of Theorem 7 (Sketch). The proof of Theorem 7 is a straightforward combination of the proofs of Theorems 5 and 6 once we have defined

$$\mathcal{P} := \{P \in \Pi_{\mathfrak{u}} : \epsilon_{1,\bullet}(P) > n^{8/9} \log n\}.$$

Details are omitted for the sake of brevity. □

Chapter 5

Query Reranking Using Vertex Nomination

In the previous chapter, we established the statistical soundness of a maximum-likelihood approach to vertex nomination. We turn now to applying this technique to the problem of rescoring query results under the search framework discussed in earlier chapters. We recall our basic framework: we have a search collection $\mathcal{S}$, a multiset of objects from some set of possible observations $\mathcal{X}$. $\mathcal{X}$ is endowed with a similarity measure $\sigma : \mathcal{X} \times \mathcal{X} \to [0, 1]$ that captures our ideal notion of similarity for the task at hand, but this oracle similarity is intractable due to computational constraints. We use in place of σ, then, a more tractable ersatz similarity function $\kappa : \mathcal{X} \times \mathcal{X} \to [0, 1]$, and embed $\mathcal{S}$ in $\mathbb{R}^d$ according to some mapping $f : \mathcal{S} \to \mathbb{R}^d$ given by, for example, Laplacian eigenmaps (see Appendix B). In Chapter 3, we showed that in the case of

Laplacian eigenmaps embeddings, only small error is incurred in the embeddings by replacing σ by a noisy, possibly biased estimate κ.

A second source of error in our search pipeline, not addressed by the results in Chapter 3, arises at query time as a result of using an out-of-sample extension and near-neighbor retrieval. Recall that having constructed embeddings $E_{\mathcal{S}} = \{f(x) : x \in \mathcal{S}\} \subseteq \mathbb{R}^d$, we build a near-neighbor index $\mathcal{I}$ (see Appendix C for an overview of near-neighbor retrieval) that allows us to quickly find points in $E_{\mathcal{S}}$ near a given query vector $z \in \mathbb{R}^d$. Having built index $\mathcal{I}$, let $q \in \mathcal{X}$ be a new query observation. In order to retrieve candidate matches to q, we must first embed it in $\mathbb{R}^d$ according to the same embedding that was applied to $\mathcal{S}$. The query is embedded as $\tilde{f}(q) \in \mathbb{R}^d$, where $\tilde{f}$ is an out-of-sample extension of embedding f. In most cases, this out-of-sample extension is based on the Nyström method (see Appendix B for further discussion). As such, $\tilde{f}$ is only an approximation to an ideal embedding $f^* : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^d$, which we would work with if it were feasible to apply our embedding technique to all of $\mathcal{X}$. Indeed, in the audio search system presented in Chapter 2, it was infeasible to even embed the search collection directly, and we settled for an out-of-sample extension based on a *reference set* of 10,383 observations. Thus, we incur error when we retrieve near-neighbors of $\tilde{f}(q)$ rather than near-neighbors of $f^*(q)$, arising from the discrepancy between the two embedding functions.

In addition to this out-of-sample extension error, the near neighbor retrieval index is a source of error, in that retrieval is approximate. When we query $\mathcal{I}$ for the near neighbors of $\tilde{f}(q)$, we obtain a set of candidate matches $R_q \subset \mathcal{S}$ whose embeddings corresponding to the near-neighbor points to $\tilde{f}(q)$. The points $\{f(x) : x \in R_q\}$ are the near neighbors of $\tilde{f}(q)$ *retrieved from* $\mathcal{I}$. As such, they are not necessarily the true nearest points to $\tilde{f}(q)$ in $E_{\mathcal{S}}$. That is to say, error is introduced in the fact that we must perform *approximate*, rather than exact, near neighbor retrieval.

To recapitulate, we have discussed at various points in this thesis four related sources of error:

1. Error in approximating the oracle similarity σ with an ersatz similarity κ, arising from model misspecification or computational constraints.

2. Error arising from further approximation of κ and occlusion of the pairwise similarities $\{\kappa(x, y) : x, y \in \mathcal{S}\}$.

3. Approximation error due to using the out-of-sample extension $\tilde{f}$ of embedding f rather than an embedding computed for all of $\mathcal{X}$.

4. Error in retrieval due to the inherently approximate nature of fast near neighbor algorithms.

In Chapter 3, we proved that the first two of these sources of error could be controlled, in the sense that under suitable conditions on the set $\mathcal{S}$ and the similarity functions σ and κ the error introduced by the embedding process becomes arbitrarily small as the number of observations in $\mathcal{S}$ increases. In this chapter, we will apply the vertex nomination tools presented in the previous chapter to mitigate the latter two sources of error.

5.1 Reranking: The basic problem

Given a set of query results for a query $q \in \mathcal{X}$, it is typical that we wish to order them by relevance. That is, order them so that the results appearing early in the list are those that we believe to be the best matches to the query q. The sources of error discussed above suggest that this ordering may not be the best one if our ultimate goal is to rank the query results according to their similarity to q as measured by κ (let alone σ), since (a) the embedding $\tilde{f}(q)$ is an out-of-sample approximation to some more faithful embedding $f^*(q)$, (b) by the nature of near neighbor retrieval, the ranking of the results in R_q is with respect to an *approximation* of the appropriate notion of nearness to $\tilde{f}(q)$ in $\mathbb{R}^d$, and (c) nearness in $\mathbb{R}^d$ only approximately reflects the similarity measure κ. In light of these sources of error, how can we reorder the query results R_q to better reflect κ, the similarity with respect to which they were initially embedded? This is an example of a rescoring or reranking problem.

Rescoring is a core component of the typical pipeline in speech processing (Rastrow et al. 2011; Peng et al. 2013; Soto et al. 2014; Pham et al. 2016), machine translation (Paul et al. 2004; Duh 2009; Blackwood 2010), and image processing (Russakovsky et al. 2015; Malik et al. 2016), to name just a few domains. In typical applications, we have a slow, expensive, but comparatively trustworthy measure of similarity or quality, but evaluating this measure on many pairs of objects is expensive. Instead, an inexpensive but less accurate method is used to quickly search over a large collection of objects. This yields a more manageable set of candidate matches, which is then reassessed using the expensive, more accurate measure. In some applications, most prominently in document retrieval, rescoring may take into account an assessment of result quality, provided either by a user or by a classifier, typically referred to as *relevance feedback* (Ruthaven and Lalmas 2003; Manning et al. 2008; Carpineto and Romano 2012).

In the case of our search framework, the fast, inaccurate method corresponds to retrieving near neighbors $N_q \subseteq f(\mathcal{S})$ of the query embedding $\tilde{f}(q)$ from index $\mathcal{I}$. This first, inexpensive step, yields query results

$$R_q = \{x : f(x) \in N_q\} \subseteq \mathcal{S}.$$

The rescoring problem becomes a question of how best to order the elements of R_q. In particular, the goal is to order the elements of R_q so that “correct” query results tend to appear higher in the list.

As an illustrative example, consider the S-RAILS system presented in Chapter 2. There, the index $\mathcal{I}$ operated by assigning each segment embedding to a bit signature according to its position relative to a set of hyperplanes. Retrieval then consisted of comparing a query signature against those in the index and retrieving those that shared prefixes (and repeating this comparison under several permutations of the bit signatures). These bit signatures allowed an approximation of the cosine similarity between pairs of vectors, and the results were ordered according to this approximate similarity to the query.

Much of the work related to reranking in machine learning has focused on *learning to rank*, in which the goal is to learn a (partial) ordering on observations that reflects some notion of goodness (e.g., quality of transcriptions in the case of speech processing or of parses in the case of natural language processing). Often, this ranking must be learned from a collection of labeled examples, and it is typical that we find ourselves in the semi-supervised setting, in which there are many available observations, but supervisory information is available for only a few pairs of these observations or for only a few lists of query results (Duh 2009). Many approaches to the semi-supervised learning to rank problem apply ideas from representation learning (see Appendix B) to learn an embedding or kernel function that reflects a suitable notion of goodness (Duh and Kirchhoff 2008; McFee and Lanckriet 2010). Others have cast the problem as one of ordinal regression (Herbrich et al. 2000; Shashua and Levin 2003) or one of data imputation to recover unobserved pairwise similarities (Zhou et al. 2004; Wang et al. 2005).

The low-resource search task addressed by the S-RAILS system in Chapter 2 belongs to this semi-supervised setting. We have a collection of millions of audio segments, but pairwise information in the form of word type labels is available for only a few thousand segments. Our word similarity experiments in Chapter 2 suggest that a learning to rank algorithm such as MLR (McFee and Lanckriet 2010) might be effective in reranking our query results, provided that MLR can correct the errors introduced by the embedding process. Rather than applying a reranking procedure, we consider here an altogether different approach based on vertex nomination, as presented in Chapter 4.

A natural approach to rescoring in the context of Chapter 2 would have been to rerank the results according to their similarity to the query, i.e., ascending in DTW distance to the query. In the notation of our framework, this corresponds to reranking R_q according to $\kappa(q, x)$ for all $x \in R_q$. This rescores R_q according to the intended measure of similarity, and thus we expect that it should result in a better ranking of the results. However, this approach does not make use all of the available information. Given our query q and results R_q, we in fact have a graph of similarities $\{\kappa(x, y) : x, y \in \{q\} \cup R_q\}$. Ideally, we should use all of the information available in these pairwise similarity measurements to perform our reranking. Indeed, if we believe that each $\kappa(x, y)$ is an estimate of the oracle similarity $\sigma(x, y)$, then we might hope that jointly using all the available pairwise similarities would improve our ranking. Vertex nomination, as discussed in the previous chapter, provides one possible way to exploit this structure.

5.2 Reranking using Vertex Nomination

We let $G_q = (V_q, E_q)$ be a weighted undirected graph on $r_q = |R_q| + 1$ vertices. These vertices correspond to the results in R_q and the query q itself, with weights given by $w_{x,y} = \kappa(x, y) \in [0, 1]$. We would like to devise a reranking scheme that uses all of the available information in this weighted graph. If we think of the result set R_q as containing "correct" and "incorrect" results, then it is natural to expect that a corresponding block structure will manifest in matrix G_q, with one block corresponding to the correct results and others corresponding to the incorrect results. We will suppose that in R_q there are $n_1 < r_q$ correct matches, and these n_1 segments will comprise the block of interest that we wish to recover. Of course, it is possible that the remaining elements of R_q have block structure of their own. Thus, we will model G_q as an exponential family SBM, which we introduced in Definition 5 in Chapter 4. In particular, we will assume that $G_q \sim \text{ExpSBM}(K, \vec{n}, b, \Theta)$, in which the first block corresponds to the correct matches to the query, and we will perform vertex nomination with precisely one seed vertex, namely the query itself. Three concerns arise immediately from this formulation, and we will briefly address them in turn.

Vertex nomination as considered in most of Chapter 4 matches an unweighted graph to a weighted graph in such a way that the resulting optimization problem is equivalent to a maximum likelihood alignment of the vertices of G_q with the vertices of the matrix encoded by B. In the present setting, G_q is weighted, and it isn't necessarily the case that matching G_q with B corresponds to a maximum likelihood solution. As sketched in Section 4.5, the exponential family SBM allows us to extend the vertex nomination to the case where $G_q \sim \text{ExpSBM}(K, \vec{n}, b, \Theta)$, in which the edge weights $w_{x,y}$ are distributed independently according to a model parameterized by $\Theta_{b(x),b(y)}$. This leaves open the question of what distribution to choose for the edge weights and how to parameterize it, a choice that depends on the problem domain.

In Chapter 4, we performed vertex nomination by aligning a matrix to a block-structured matrix B, the entries of which were a function of a block communication matrix $\Lambda \in [0,1]^{K \times K}$ (or, in the case of the exponential family SBM, a parameter matrix $\Theta \in \mathbb{R}^{K \times K}$). The entries of Λ were estimated based on the seed vertices. In the present case, we have only one seed vertex, and hence have no way, a priori, to estimate the entries of the parameter matrix $\Theta \in \mathbb{R}^{K \times K}$. As such, we need some other way to estimate the model parameters.

A third concern pertains to estimating the block sizes. In Chapter 4, our model assumed that we knew the correct block sizes n_k for $k = 1, 2, \dots, K$, or that we had seed vertices from each block with which to estimate the block sizes. In the present setting, it is not clear how we should choose $\vec{n}$, since we have only one seed vertex with which to perform estimation. While we might attempt to estimate block structure by examining the weight matrix of G_q or decide on block sizes based on domain knowledge, it is not clear a priori how to proceed.

How we deal with the above concerns will, in general, depend on domain-specific factors. In the remainder of this chapter, we will consider query reranking in the context of the S-RAILS audio search system presented in Chapter 2. Our goal is to explore whether or not VN-based reranking of the query results returned by S-RAILS improves upon the evaluation scores presented in that chapter. In the next section, we will describe a basic system for reranking in the S-RAILS system based on vertex nomination. In the sequel, we will explore the effects of modifying these approaches in various ways.

5.3 VN-based reranking for audio search

Recall that in the setting of Chapter 2, the set $\mathcal{X}$ of possible observations corresponded to the set of all possible utterances of length between 500 and 1,000 ms, represented by their feature vector time series, so that any $x \in \mathcal{X}$ could be written as $x = x_1, x_2, \ldots, x_m$ for some number m, where $x_i \in \mathbb{R}^p$ for all $1 \leq i \leq m$. Our ersatz similarity took the form of a Gaussian kernel

$$\kappa(x, y) = \exp\left\{-\frac{[\max(0, \mathrm{DTW}(x, y) - \eta)]^2}{2\sigma^2}\right\},$$

where $\eta > 0, \sigma > 0$ are parameters and $\mathrm{DTW}(x, y)$ denotes the cost of dynamic time warping (DTW; see Appendix A for a discussion) alignment between x and y. Thus, the vertices of our results graph $G_q = (V_q, E_q)$ correspond to audio segments. Under the assumption that the query results contain audio segments corresponding to only a few different words or phrases, we expect G_q to exhibit an approximate block structure in which the blocks correspond to these word types. Thus, we will model G_q as being distributed as $G_q \sim \mathrm{ExpSBM}(K, \vec{n}, b, \Theta)$.

5.3.1 Modeling edge weights

Following a similar approach to that taken to the error model in the synthetic experiments in Chapter 3, we will model the weights $w_{x,y} \in [0, 1]$ as being distributed according to a one-dimensional subfamily of the beta distribution. In particular, we will take the approach in which the beta distribution $\mathrm{Beta}(\alpha, \beta)$ is reparameterized in terms of its mean $\mu = \alpha/(\alpha + \beta)$ and "sample size" $\nu = \alpha + \beta$. To obtain a one-parameter subfamily, we will assume that the sample size parameter ν is the same across all blocks. That is, we will make the assumption that the distribution of $w_{x,y}$ has the same ν value regardless of the block memberships $b(x), b(y)$, so that only the expected value of $w_{x,y}$ varies by block. This decision is consistent with an assumption that the edge weights differ in expectation based on the block memberships of the audio segments (e.g., based on whether or not the audio segments correspond to the same word), but that the variance of $w_{x,y}$ about its mean does not depend on the block memberships.

Fixing for now some global value for $\nu = \alpha + \beta$ that we will specify later, and letting $C > 0$ denote a normalization constant, $w_{x,y}$ has density

$$f(w \mid \alpha, \beta) = C w^{\alpha-1}(1-w)^{\beta-1},$$

and has log-likelihood

$$\begin{aligned}\log f(w \mid \alpha_{x,y}, \beta_{x,y}) &= (\alpha_{b(x,y} - 1) \log w_{x,y} + (\beta_{x,y} - 1) \log(1 - w_{x,y}) + \log C \\ &= (\alpha_{x,y} - 1) \log \frac{w_{x,y}}{1 - w_{x,y}} + (\beta_{x,y} + \alpha_{x,y} - 2) \log(1 - w_{x,y}) + \log C \\ &= (\mu_{x,y}\nu - 1) \log \frac{w_{x,y}}{1 - w_{x,y}} + (\nu - 2) \log(1 - w_{x,y}) + \log C,\end{aligned}$$

where we have used the fact that under this reparameterization we have $\nu = \alpha + \beta$ and $\mu = \alpha/\nu$. Following the framework outlined in Section 4.5, we find that we will want to align the matrix

$$A = [A_{x,y}] = \left[\log \frac{w_{x,y}}{1 - w_{x,y}}\right]$$

to the matrix

$$B = [B_{x,y}] = \left[\mu_{b(x),b(y)}\nu - 1\right],$$

where the block assignment function $b : V \to [K]$ is chosen to reflect our choice of block sizes (see Section 5.3.3).

5.3.2 Parameter Estimation

In the search task considered in Chapter 2, we had access to a set of approximately ten thousand isolated, labeled word examples. We will use those word examples to estimate the entries of the block parameter matrix Θ. In Chapter 4, we estimated a new block communication matrix Λ for each graph we considered. Such an approach is less feasible here. Instead, we will estimate one matrix of parameters Θ, and use it for all queries. In particular, we are interested in two parameters, μ_1 and μ_2, the mean similarity between same-word and different-word segments, respectively. Our parameter matrix Θ will then have

$$\Theta_{i,j} = \begin{cases} \mu_1 & \text{if } i = j \\ \\ \mu_0 & \text{otherwise,} \end{cases}$$

reflecting the fact that we assume that each block of our SBM corresponds to a different word type.

Denoting our collection of labeled word examples by M, let $c : M \times M \to \{0, 1\}$ be such that $c(x, y) = 1$ if x and y have the same word label and $c(x, y) = 0$ otherwise. Define the sets

$$M_1 = \{\{x, y\} : c(x, y) = 1\}, \;\; M_0 = \{\{x, y\} : c(x, y) = 0\}.$$

Using these labeled examples, we can estimate the block probability parameters by

$$\begin{aligned}\hat{\mu}_1 &= \frac{1}{|M_1|}\sum_{\{x,y\}\in M_1}\kappa(x,y)\\ \hat{\mu}_0 &= \widehat{\Lambda}_{2,1} = \widehat{\Lambda}_{2,2} = \frac{1}{|M_0|}\sum_{\{x,y\}\in M_0}\kappa(x,y).\end{aligned}$$

Of course, more complicated approaches to estimating these parameters are possible, for example by trying to separately estimate $\widehat{\Lambda}_{1,2}$ and $\widehat{\Lambda}_{2,2}$ to better capture block structure that may be present in the non-matching results, but in the present setting, only the sets M_0 and M_1 are sensible ones to ask about.

Applying this estimation to the reference set of 10,383 word examples used in Chapter 2, we obtain estimates $\hat{\mu}_1 = 0.6332, \hat{\mu}_0 = 0.2671$. Alternate estimates can be obtained based on a plug-in estimate, using the fact that $\mu = \alpha/(\alpha+\beta)$, where α and β are the shape parameters of the Beta distribution. Maximum likelihood estimation of the shape parameters applied separately to the sets $\{x,y\} \in M_1$ and $\{x,y\} \in M_0$ yields estimates

$$\begin{aligned}\hat{\alpha}_1 &= 4.5227, \quad \hat{\beta}_1 = 2.6344\\ \hat{\alpha}_0 &= 4.0492, \quad \hat{\beta}_0 = 11.0714,\end{aligned}$$

from which we obtain plug-in estimates for the corresponding μ and ν parameters

$$\begin{aligned}\hat{\mu}_1 &= 0.6319, \quad \hat{\nu}_1 = 7.1570\\ \hat{\mu}_0 &= 0.2678, \quad \hat{\nu}_0 = 15.1206,\end{aligned}$$

which we will use below. We choose as our global, fixed value of ν the inter-match parameter estimate $\hat{\nu} = \hat{\nu}_0 = 15.126$. Empirically, we found little to no difference in performance between using $\hat{\nu} = \hat{\nu}_0$ and $\hat{\nu} = \hat{\nu}_1$.

5.3.3 Choosing block sizes

It remains to address how we will choose the sizes of the blocks when performing VN reranking. There are a number of possible block structures to attempt to capture. For the time being, we will sketch one, which we will call the *flat* structure. We assume only two blocks, one corresponding to the segments that are correct matches and one corresponding to the segments that are not correct matches ("non-matches"). Rather than attempting to estimate the sizes of these two blocks in G_q, we will take advantage of the $\mathcal{L}^{\text{ML}}$ ranking function, defined in 4.2.1. Roughly speaking, $\mathcal{L}^{\text{ML}}$ ranks the non-seed vertices according to how much the likelihood improves when each given vertex is added to the interesting block. Thus, we take the interesting block to have size $\mathfrak{u}_1 = 1$, and take the other block to be of size $\mathfrak{u}_2 = r_q - 1$.

5.4 Experiments

We turn now to assessing whether or not VN reranking can improve the query results in the S-RAILS system presented in Chapter 2. For each of the 2756 unique queries in the development set, we will rerank the results returned by the S-RAILS system presented in Chapter 2. In Chapter 2, we considered the effect of beamwidth B on system performance. In the present setting, the computational costs of the maximum likelihood VN procedure make it infeasible to rerank more than the top 1,000 results. In light of this, we limit ourselves to the case of beamwidth $B = 1,000$. Baseline results are summarized in Table 5.1 for various settings of the signature length and number of of permutations. All experiments use the signature threshold $\tau_{\text{thresh}} = 0.06$, as in Chapter 2. We refer the reader to Chapter 2 for a discussion of the three evaluation metrics FOM, OTWV and P@10.

Table 5.1: Baseline S-RAILS performance on the *development* search collection, averaged over all query types as a function of signature length S and number of permutations P for beamwidth $B = 1,000$ and using signature threshold $\tau_{\text{thresh}} = 0.06$. All scores are percentages.

		Median Example			**Best Example**		
S	P	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
64	4	19.6	13.6	16.0	43.2	31.0	52.3
64	8	24.1	16.4	17.3	48.3	33.2	54.1
128	4	21.6	14.4	19.0	44.5	31.8	57.4
128	8	27.3	16.9	19.9	52.4	37.1	62.1
256	4	25.0	17.4	27.0	46.4	35.1	60.2
256	8	31.7	20.2	28.0	52.7	38.6	63.0
512	4	21.7	15.5	22.8	45.9	34.0	59.3
512	8	28.2	19.0	24.3	52.7	38.4	64.1
1024	4	22.0	16.6	24.4	45.9	34.3	58.7
1024	8	29.5	19.3	25.3	52.7	38.9	62.9

We recall that using beamwidth $B = 1,000$ as in Table 5.1 corresponds to retrieving the nearest 2000 segments. For a given query segment q, merging the overlapping result segments yields $r_q \leq 2,000$ results. Ideally, we would reorder all r_q results, but the computational complexity of ML-based VN makes it infeasible to reorder more than about 1,000 segments. Thus, we will settle for reordering only the top $\min\{n_{\text{top}}, r_q\}$ results most similar to q under $\kappa(q, \cdot)$, where $n_{\text{top}} \leq 1,000$ is a number that we must choose. As described in the previous section, our first attempt at reranking consists of aligning the query results to a two-block model, in which the interesting block has size $n_1 = 1$. The results of this reranking are summarized for $n_{\text{top}} = 500$ in Table 5.2, and for $n_{\text{top}} = 1,000$ in Table 5.3. In both tables, we list the relative improvement with respect to the results listed in Table 5.1, given as a percentage, so that large positive numbers correspond to marked improvements over the baseline performance in Table 5.1, while negative numbers correspond to performance inferior to that seen in the baseline system. We found that performance was fairly stable over choices of the size of the interesting block n_1 up to $n_1 = 25$, at which point results tended to degrade slightly with respect to those in Tables 5.2 and 5.3.

Table 5.2: VN-reranking S-RAILS performance on the *development* search collection, averaged over all query types as a function of signature length S and number of permutations P for interesting block size $n_1 = 1$, reranking set size $n_{\text{top}} = 500$, beamwidth $B = 1,000$ and using signature threshold $\tau_{\text{thresh}} = 0.06$. All scores are relative improvement (i.e., percentage) with respect to the scores in Table 5.1.

		Median Example			**Best Example**		
S	P	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
64	4	6.6	5.5	36.4	3.8	7.1	8.7
64	8	7.9	0.8	34.9	3.5	11.1	11.3
128	4	3.5	1.2	10.5	1.9	4.2	6.0
128	8	4.4	-5.0	17.7	2.2	0.4	2.6
256	4	0.4	-5.7	-0.8	0.3	1.9	-1.2
256	8	-0.3	-12.4	-5.8	0.5	-1.5	0.6
512	4	0.2	-3.3	-1.7	0.3	2.9	1.1
512	8	-0.4	-11.5	-3.1	0.4	-2.5	-1.8
1024	4	0.4	-7.6	0.4	0.5	0.7	0.5
1024	8	0.0	-11.4	-1.6	0.9	-1.8	0.2

At a high level, we see that VN reranking tends to improve performance for the case of signature length $S = 64$, but tends to have a smaller effect and sometimes even hurts performance for longer signatures. It also appears to be the case that VN reranking improves performance more (or hurts it less) in the case of $P = 4$ compared to $P = 8$ permutations. The effects of both signature length S and number of permutations P may be explained by the fact that smaller values of S and P correspond to coarser results sets. When S and P are large, the approximation of the cosine distance by the S-RAILS index is better, and thus the segments it retrieves are more likely to be the correct ones (and are more likely to be correctly ranked), leading to less benefit in reranking of the results. Comparing the $n_{\text{top}} = 500$ reranking in Table 5.2 against the $n_{\text{top}} = 1,000$ reranking in Table 5.3, we see that $n_{\text{top}} = 500$ tends to yield marginally better improvements over the baseline scores. This is surprising at first, since we saw in Chapter 4 that working under the stochastic block model, more vertices tended to yield better nomination. In the present setting, the number of vertices of interest does not grow linearly with the total number of vertices– recall from Chapter 2 that the queries in the development set have between 2 and 188 times in the development search collection. Thus, it is likely that reranking $n_{\text{top}} = 1,000$ results rather than $n_{\text{top}} = 500$ serves primarily to introduce noise in the form of additional edges rather than to add more correct matches to G_q, which would reduce the variance in our block assignments.

Table 5.3: VN-reranking S-RAILS performance on the *development* search collection, averaged over all query types as a function of signature length S and number of permutations P for interesting block size $n_1 = 1$, reranking set size $n_{\text{top}} = 1,000$, beamwidth $B = 1,000$ and using signature threshold $\tau_{\text{thresh}} = 0.06$. All scores are relative improvement with respect to the scores in Table 5.1.

		Median Example			**Best Example**		
S	P	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
64	4	5.5	2.1	33.7	4.1	4.0	10.8
64	8	8.2	-2.7	26.2	5.4	8.2	11.0
128	4	3.3	-0.3	5.0	0.9	2.4	4.4
128	8	2.1	-7.5	11.1	1.6	-1.5	1.9
256	4	-0.1	-7.4	-1.9	0.0	1.6	-2.2
256	8	-1.5	-14.2	-6.1	0.3	-2.0	0.3
512	4	-0.4	-5.1	-3.3	0.2	2.4	-0.6
512	8	-1.4	-12.8	-1.8	0.3	-3.2	-3.5
1024	4	-0.1	-8.4	0.9	0.4	0.4	0.5
1024	8	-1.5	-12.2	-0.9	0.9	-2.3	1.0

It is possible that by the nature of the $\mathcal{L}^{\text{ML}}$ ranking function used in our VN reranking procedure and the fact that we have used an interesting block of size 1, much of the ranking that we obtain is driven simply by the similarity of individual segments to the query segment. To isolate this effect, we introduce the *kernel reranking* procedure. The S-RAILS system performs approximate near neighbor retrieval on the embeddings $f(\mathcal{S}) = \{f(x) : x \in \mathcal{S}\} \subseteq \mathbb{R}^d$ by replacing computation of the cosine distance between $f(q)$ and $f(x)$ with an approximation based on Hamming distance between signatures. Thus, the results in Table 5.1 correspond to a ranking of R_q according to descending value of $s_{\mathcal{I}}(x, q)$. The similarities $\{s_{\mathcal{I}}(x, q) : x \in R_q\}$ are approximations of cosine similarities $s(f(x), \tilde{f}(q))$, which are in turn meant to reflect the structure of κ, and it is natural to rerank R_q by evaluating the ersatz function $\kappa(q, x)$

on each of the query results $x \in R_q$ and rerank the results $R_q = \{x_1, x_2, \ldots, x_{r_q}\}$ according to permutation $\rho \in S_{r_q}$ so that $\kappa(x_{\rho(1)}, q) \geq \kappa(x_{\rho(2)}, q) \geq \cdots \geq \kappa(x_{\rho(r_q)}, q)$. We will refer to this as *kernel reranking*. Note that since the query is the only seed, this approach is equivalent to performing restricted-focus VN on the graph G_q.

Effectiveness of the kernel reranking in improving the baseline S-RAILS result is summarized in Table 5.4. As in Tables 5.2 and 5.3, all scores are reported in relative improvement compared to the baseline. We see that kernel reranking is broadly comparable to VN-based reranking, with kernel reranking sometimes outperforming VN-based (e.g., in P@10 for $S = 128$) and sometimes falling short of it (e.g., in P@10 for $S = 64$). On the whole, VN-based reranking does not appear to improve the S-RAILS baseline performance any more than the simpler, computationally cheaper kernel reranking.

Table 5.4: Kernel-reranking S-RAILS performance on the development search collection, averaged over all query types as a function of signature length S and number of permutations P for beamwidth $B = 1,000$ and using signature threshold $\tau_{\text{thresh}} = 0.06$. All scores are percentage relative improvement over the baseline performance in Table 5.1.

		Median Example			**Best Example**		
S	P	**FOM**	**OTWV**	**P@10**	**FOM**	**OTWV**	**P@10**
64	4	7.4	3.3	38.3	5.0	5.7	12.8
64	8	9.8	-3.5	24.5	8.2	10.7	11.4
128	4	3.5	0.6	5.0	1.9	4.1	4.8
128	8	2.9	-6.0	8.0	2.3	-0.6	3.0
256	4	0.4	-6.1	-4.1	0.2	2.1	-0.1
256	8	-0.7	-12.3	-6.5	0.5	-1.5	0.8
512	4	0.2	-3.6	-1.7	0.3	2.9	0.7
512	8	-0.4	-11.1	-6.0	0.4	-3.0	-2.4
1024	4	0.4	-7.0	-1.0	0.5	0.9	2.5
1024	8	-0.3	-11.2	-2.0	0.9	-2.6	-0.1

How should we account for the failure of the VN-based reranking to improve appreciably on the kernel-based reranking? We have already mentioned that choice of interesting block size parameter n_1 has fairly little effect on performance. One possible explanation is that the VN-based reranking fails to adequately capture the block structure more broadly. Suppose, for example, that among the non-matching segments there are segments corresponding to 3 other distinct word types. This would manifest as a 4-block structure, which the flat structure used above in Tables 5.2 and 5.3 is unable to capture. In experiments, we have tried varying the number of blocks K and the selection of the parameter matrix Θ. Our results largely suggest that modeling every set of query results as comprising exactly K blocks for some fixed choice of K tends only to degrade as K increases past 2.

Ultimately, it seems that either our assumption of block structure is incorrect or the noise associated with the DTW distance is too large for VN to overcome. As discussed in Chapter 2 and Appendix A, DTW is a notoriously poor measure of audio similarity. In particular, the sensitivity of DTW to speaker and channel variability is such that it likely that some segments that should be labeled as query matches will often not have this fact accurately reflected in the similarity graph G_q. An examination of a selection of known correct matching segments suggests that this is indeed the case, though a more thorough examination of this fact is beyond the scope of this chapter. Experiments similar to the synthetic experiments in Chapter 4, this time using the Beta SBM described above with the estimated values for Θ and $\hat{\nu}$, show that accurate nomination is possible when the data are actually generated according to the posited model. These results suggest that the lackluster performance of the reranking techniques explored above is likely due to model misspecification, most likely owing to the noise associated with DTW distance.

One possible explanation for the poor performance of VN-based reranking is that there is only one query example, i.e., only one vertex in the interesting block. The result is that our ranking is based upon similarities to only a single seed vertex, and this signal is liable to be swamped by the hundreds of similarities between the non-seed vertices. One possible solution to this is to artificially increase the number of seed vertices by artificially inflating the set of interesting seeds with multiple copies of the query vertex. Experiments with this approach have shown mixed results, and largely match those seen in Tables 5.2, 5.3 and 5.4.

5.4.1 Augmenting Query Examples

We saw in Chapter 4 that the accuracy of vertex nomination improves with the availability of additional seed vertices. It stands to reason that additional seed vertices should improve the performance of VN reranking of S-RAILS output. Given a query of a certain word type, we can augment the query segment with additional examples of the same word type. It is expected that these *augmenting seeds* will provide additional block structure for the ML-based VN reranking to take advantage of. Adding these seeds also provides an opportunity to investigate whether the comparatively small improvement yielded by VN reranking is due to noise and model misspecification or merely to a dearth of seed vertices. In this section we briefly explore how the presence of additional seeds influences reranking.

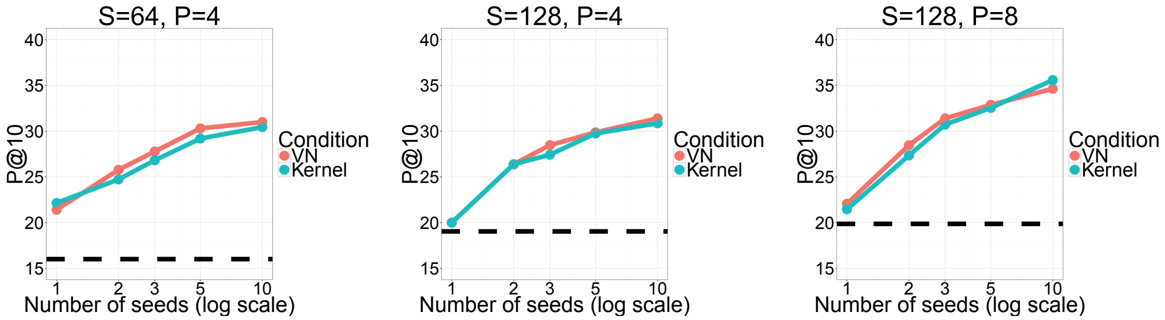
As above, we assume that all seeds come from the interesting block. That is, we have $m_i = 0$ for $i \neq 1$. Thus, specifying the number of seed vertices amounts to choosing the number of augmenting seeds $m_1 - 1$. For each query q in the S-RAILS development set described in Chapter 2 and revisited above, we choose $m_1 - 1$ query examples at random from the other query examples of that type. In the event that there exist fewer than m_1 unique examples of the query word type, we choose seeds with replacement. Having chosen $m_1 - 1$ additional query examples, we can add these to the results graph G_q to obtain a new graph $\tilde{G}_q$. These augmenting seeds provide additional graph structure, and we can proceed with vertex nomination as above and in Chapter 4. This augmenting procedure can also be naturally extended to the kernel-based reranking described above. Instead of ordering the query results $x_i \in R_q$ according to $\kappa(q, x_i)$, we rank the query results according to the augmented kernel function

$$\tilde{\kappa}(x) = \sum_{j=1}^{m_1} \kappa(\tilde{q}_j, x),$$

where $\tilde{q}_j$ denotes the j-th query example (i.e., j indexes the query and its augmenting seeds).

Varying m_1 (note that $m_1 = 1$ recovers the reranking procedures described earlier in this chapter), we can see how the number of seed vertices influences reranking, summarized in Figure 5.1, which shows the effect of the number of seeds m_1 on reranking quality. Owing to the fact that OTWV and FOM are primarily functions of recall and false alarm rate, rescoring does little to improve them, as we see in Tables 5.2, 5.3 and 5.4. As such, we restrict our attention here to precision at 10 (P@10). Turning our attention to Figure 5.1, we see that more augmenting seeds improves the performance of both the VN-based reranking and its kernel-based counterpart, and that both techniques are largely identical in their performance. This is evidence that augmenting the query segment with additional examples does indeed improve system accuracy. We note that VN-based reranking appears to have a slight edge on kernel reranking in the $S = 64, P = 4$ setting, which is particularly promising for the case of rescoring, since this corresponds to the setting in which we expect precision (prior to rescoring) to be especially low. All told, however, the performance seen in Figure 5.1 is largely in agreement with that seen in Tables 5.2, 5.3 and 5.4, in that VN does not appear to improve substantially over kernel-based reranking, even with additional seed vertices. This lends evidence to our speculation above that model misspecification, both in the model of edge weights and the graph structure (i.e., number of blocks) is overwhelming whatever additional information would be otherwise gained by VN's use of the full similarity graph.

Other settings of the S-RAILS parameters S and P showed performance broadly similar to that seen in Figure 5.1, with larger signature lengths S and larger numbers of permutations P yielding generally better performance, but with the gain over the



(a) **(b)** **(c)**

Figure 5.1: Performance of augmented reranking on the development set, measured by precision at ten (P@10), as a function of the number of seeds, for different S-RAILS parameter settings (signature length S and number of permutations P). The plots show the median P@10 score as a function of the number of seeds for the VN-based (orange) and kernel-based (blue) augmented reranking for (a) signature length $S = 64$ bits, and $P = 4$ permutations, (b) signature length $S = 128$ bits and $P = 4$ permutations, and (c) signature length $S = 128$ bits and $P = 8$ permutations. The black dashed line denotes the performance of the baseline S-RAILS system for the given signature length and number of permutations.

baseline decreasing somewhat at those higher levels, similar to the patterns seen in Tables 5.2, 5.3 and 5.4. Performance was also largely independent of graph structure choices such as the size of the first block and the number of blocks, similar to that described in our experiments in the previous section.

5.5 Discussion

We have introduced a method for reranking query results based on the maximum-likelihood vertex nomination scheme described in Chapter 4, and examined its ability to improve the S-RAILS system presented in Chapter 2. Our experiments suggest limited improvement, especially for short signature lengths and smaller number of permutations, but much of this improvement is also exhibited by a less complex reranking scheme based solely on the DTW distance between results and the query itself.

The failures of the VN-based reranking system to more markedly improve S-RAILS performance are plausibly explained either by model misspecification or by a lack of seed vertices. Additional experiments, in which the query segment is supplemented with additional seed vertices in the similarity graph, suggest that this failure is due primarily to the former. These model misspecifications are due primarily to two factors. First, the pairwise similarities are subject to large amounts of noise due to their being based on DTW alignment. Secondly, the block structure assumed by the exponential family SBM does not appear to accurately capture the structure of the data in most cases. In the specific case of the S-RAILS system, there is little we can do about the former point, short of developing a better method for assessing audio similarity, but one might consider applying denoising methods (see Appendix B) to the matrix of similarities in hopes of alleviating the effect of high variance in the DTW similarities.

On the other hand, a wide range of more sophisticated approaches to modeling the block structure in G_q are possible, and are good targets for future work. For example, one might attempt to model the structure in G_q separately for each query q rather than estimating a single parameter matrix Θ to be used across all queries. Similarly, we might altogether abandon any attempt at explicitly modeling block structure and back off to permuting the weight matrix of G_q to maximize a modularity-based objective or use some other VN scheme (Fishkind et al. 2015). A key challenge in future work on VN-based reranking will be to separate the effects of model misspecification in the SBM itself from the failure to accurately model the edge distributions.

Chapter 6

Discussion and Future Work

In this thesis, we have explored sources of error in a commonly-used framework for similarity search over large data sets. Applying tools from graph inference, we have proved theoretical results in Chapters 3 and 4 showing that these errors can be controlled, and we have explored the efficacy of these theoretical approaches in a speech processing search task in Chapters 2 and 5. In closing, we will briefly recapitulate these results and discuss directions for future work.

6.1 The S-RAILS System

In Chapter 2, we introduced the S-RAILS system for large-scale audio search. The S-RAILS system performs audio search by embedding audio segments according to Laplacian eigenmaps and retrieving query results using fast approximate near-neighbor retrieval techniques. We presented this system to serve as a baseline with which investigate the accuracy of our theoretical results and to serve as an illustration of the search pipeline first introduced in Chapter 1.

We saw, firstly, that embedding speech audio via Laplacian eigenmaps (Belkin and Niyogi 2003; Belkin et al. 2006) yielded performance on a word discrimination task superior to several other embedding methods. Improving this embedding procedure is a promising avenue for future work. We have seen in empirical experiments that a simple change of speech features, such as from PLP to FDLP as in Chapter 2. As such, further exploration of how choice of features influences system performance is quite warranted. It is unlikely, however, that any choice of feature will entirely correct for the underlying inadequacy of assessing segment similarity via DTW alignment cost. As such, one might consider learning transformations of speech features or sequences thereof with the explicit objective of making the transformed feature segments more conducive to comparison via DTW. For example, a transformation that identifies likely phones or syllables and renormalizes their lengths would lessen the outsize influence of vowel segments on DTW alignment cost. In a different direction, one might consider how our choice of reference segments influences the quality of the resulting embeddings in the case of the embeddings that made use of the reference set. For example, what is the effect of using a reference set taken from another language, or using a reference set that consists of more haphazardly-chosen segments, rather than whole words?

Laplacian eigenmaps embeddings using the normalized graph Laplacian have the advantage of being comparatively robust to noise in pairwise similarity measurements, as we proved in Chapter 3. Nonetheless, a more thorough exploration of which types of embeddings are most effective is certainly warranted, but beyond the scope of this work. For example, in Chapter 2, we compared Laplacian eigenmaps against many other embeddings and found it to be largely superior, but we did not attempt a comparison against other manifold-based techniques such as those discussed in Appendix B. Similarly, as discussed in Appendix C, there are many existing methods for near neighbor retrieval, and it is entirely possible that the one used in the S-RAILS system as presented in Chapter 2 is not optimal for the task at hand. As such, a comparison of near neighbor retrieval techniques and the associated tradeoffs in terms of speed, signature length and accuracy would likely lead to marked improvements in the S-RAILS system.

In the S-RAILS system, we would ideally like to embed the entire search collection according to a single Laplacian eigenmaps embedding, rather than using the out-of-sample extension based on a reference set. To do this precisely would require computing all the entries of a kernel matrix with one row for each segment, an utterly infeasible number of DTW alignments. Our results in Chapter 3 suggest that we might instead back off to an embedding that only sparsely populates this massive matrix. Initial attempts to apply such an approach met with failure, owing to the fact that most pairs of speech segments have high DTW alignment cost. This made it necessary to evaluate the majority of the pairwise DTW alignments before non-trivial structure in the embeddings would emerge. Nonetheless, it seems quite possible that additional engineering effort might overcome this hurdle, for example, by following the centroid-based approach taken in Chapter 2.

6.2 Convergence of Sparse, Noisy Laplacian Eigenmaps

In Chapter 3, we turned to the question of how the Laplacian eigenmaps embeddings used in the S-RAILS system behave in the presence of noisy in the similarity measurements $\kappa(x, y)$ and in the presence of occlusion of the kernel matrix $K = [\kappa(x, y)]$. We showed that Laplacian eigenmaps embeddings are robust to such noise and occlusion, and that in particular, the Laplacian eigenmaps embeddings used in Chapter 2 maintained their performance on the word discrimination task even in the face of this noise and occlusion.

A natural question in light of our theoretical results is the extent to which these results can be extended to other manifold-based embedding methods. It seems clear that a similar analysis can be applied to, for example, diffusion maps (Coifman and Lafon 2006), owing to the normalization structure of the embedding, but it is less clear whether such results can be obtained for other embeddings discussed in Appendix B, such as MDS.

The motivation for the analysis in Chapter 3 came from the S-RAILS system, in particular from the facts that (a) DTW is at best an approximation to our intended notion of segment similarity, (b) DTW is expensive to compute precisely. However, our error model in Chapter 3 fails to fully capture some of our larger concerns about DTW, namely that an unbiased error model is likely insufficient. In Chapter 3, we briefly discussed the matter of biased or nonlinear error models, and sketched how our analysis could be extended to those cases. A more thorough exploration, perhaps relating the size of the bias or the structure of the errors to downstream task performance, would further our understanding of how these embeddings will tend to behave when applied to real data, which, as we saw in Chapter 5, often has far more complicated noise structure. A still more ambitious tack would be to relax the independence assumptions in Chapter 3.

We note the recent result of Tang and Priebe (2016), which gives a central limit theorem for entries of the top eigenvectors of the graph Laplacian of a random dot product graph (Young and Scheinerman 2007). We suspect that the techniques presented therein might be extended to the model considered in Chapter 3.

6.3 Vertex Nomination

In Chapter 4, we presented the vertex nomination problem, along with a maximum-likelihood based solution. We proved the consistency of this approach, i.e., its ability to correctly recover the vertices of interest, under the stochastic block model, and sketched its extension to attributed graphs and to the broader class of exponential family SBMs, in which edges of the graph are distributed according to a one-parameter exponential family. We demonstrated the utility of ML-based VN both on synthetic data and on real-world data sets.

A natural direction for future work related to vertex nomination is to extend the results to a broader class of random graph models. In the former case, a natural first step would be to work in the random dot product graph (Young and Scheinerman 2007), but here already a few technical challenges arise, primarily due to the loss of block structure– the in absence of block structure, it is not clear how best to incorporate information from seed vertices. Indeed, it is not even clear a priori how to define the notion of interesting vertices, in the absence of an interesting block. More broadly, one might consider extending the results of Chapter 4 to the degree-corrected SBM or other commonly-used, more realistic models of networks. Indeed, such extensions are of the utmost importance for practitioners working with larger networks, which do not always exhibit the block structure assumed by the SBM and exhibited (to a degree) by the comparatively small real data sets considered in Chapter 4.

The main drawback of the ML-based vertex nomination scheme is that it is only feasible for graphs of a few thousand vertices at the very most (we found that graphs of 1000 vertices could typically be processed in MATLAB in between 20 and 40 minutes). One possible solution is to adopt a coarse-to-fine approach, somewhat similar to the rescoring framework adopted in Chapter 5, in which we use the fast spectral nomination scheme to retrieve a large set (say, 1000) of candidate vertices, then use ML-based VN to perform a more nuanced reranking. It is likely that this approach will encounter similar problems to those seen in Chapter 5, where modeling the block structure of the coarse-grained results proved challenging.

6.4 Reranking using VN

In Chapter 5, we used the ML-based VN approach presented in Chapter 4 to devise a rescoring scheme for similarity search in the framework presented in Chapter 1. We outlined the major the design choices required to apply this rescoring to actual data, and illustrated one possible set of design choices in the context of rescoring the results of the S-RAILS system first presented in Chapter 2. We demonstrated that VN-based rescoring of S-RAILS query results tends to improve the evaluation metrics, especially for system settings that correspond to coarser-grained retrieval. However, we also demonstrated that a simpler reranking scheme, based only on computation of similarity scores between the query results and the query, rather than on all pairwise similarities between the query results, as in the case of VN. We concluded that the failure of the VN reranking to improve significantly over the simpler approach was likely due to a combination of noise in the pairwise DTW scores and inability to correctly capture the structure of the similarity graph G_q.

As demonstrated in the synthetic experiments in Chapter 4 and the similar experiments mentioned in Chapter 5, data generated from the correct model (i.e., in the case of Chapter 5, a submodel of the Beta distribution) is handled, by and large, correctly. Thus, first steps toward improving the performance of VN-based reranking is to determine a more suitable model either for the DTW similarities themselves or for the structure of the similarity graph. One possible approach would be to replace the ML-based vertex nomination with an approach that depends less heavily on estimating model parameters. For example, we might devise a VN scheme based on maximizing modularity (Newman 2006b), though the equivalence of maximum modularity clustering and maximum likelihood clustering in the special case considered by Newman (2016) suggests that this may not be the best way forward. A more thorough understanding of how modeling choices (e.g., number of blocks and their sizes) influence rescoring performance would be ideal. While a theoretical approach to this problem seems challenging, further empirical exploration of these questions might lead to VN query reranking that outperforms its simpler counterpart.

Appendix A

Large-Scale Audio Search

In this section, we give an overview of large-scale speech audio search as it pertains to the work presented in this thesis. As we cannot hope to give a complete overview of speech processing in such a small space, and we refer the reader to Quatieri (2002) for more thorough discussion of processing and representing the speech signal. For a discussion of the statistical models typical of speech recognition, we refer the reader to Jelinek (1997), and Rabiner (1989); Gales and Young (2008) for a discussion of hidden Markov models in particular.

A.1 Speech Processing: an Overview

The standard approach to speech recognition begins by representing the speech signal by a series of vectors called *frames*. Each frame reflects the speech signal over a small span of time, typically about 25 ms, typically capturing relevant spectral information in the signal, as well as change in the signal Quatieri (2002); Athineos and Ellis (2003); Thomas et al. (2009). To model this sequence of frames, it is typical to break the generative process into two pieces. The first, called the *acoustic model* seeks to capture the probability of generating a sequence of speech frames based on a sequence of words (e.g., a sentence). A second model, called the *language model*, attempts to model the sequences of words themselves. Thus, the task of speech recognition is, in short, to solve the maximization (see, for example, Jelinek 1997)

$$\hat{W} = \arg\max_{W} P_{\text{LM}}(W) P_{\text{AM}}(A \mid W),$$

where A is a sequence of frames and the maximization is over all possible word sequences W.

We are concerned here, primarily, with the acoustic model P_{AM}. Typically, modeling of the speech signal takes place under an assumption of frame-level independence, such as in the case of hidden Markov modeling (Gales and Young 2008). The typical approach consists of choosing, for a given word sequence W, a corresponding sequence of hidden states, one for each frame in the acoustic signal A. These hidden states are typically assumed to correspond, roughly, to basic units of speech, e.g., phones or sequences thereof. explain sequences of these hidden states as generated based on a given word sequence. The acoustic model must account for both the hidden state trajectories and the state-conditional probabilities of observing the corresponding frames. This is done, classically by modeling the state transitions as a stationary Markov chain and the state-conditional distributions as Gaussian mixture models (GMMs). This HMM-GMM approach is quite common (Jelinek 1997; Gales and Young 2008), though in recent years the Gaussian mixture models have been increasingly replaced with feed-forward deep neural nets (DNNs; Mohamed et al. 2012), with some approaches doing away with the HMM-GMM framework entirely in favor of more complicated neural sequence models (Sak et al. 2014, 2015). Indeed, recent approaches along these lines have, for the first time, obtained error rates comparable to humans on a transcription task (Xiong et al. 2016). We note that these models, while they improve upon the expressive power of the classical models, tend to suffer from the same drawbacks outlined in Chapter 2. Most importantly, these models require massive collections of supervised training data.

A.2 Keyword Search

The goal of keyword search is to locate occurrences of a given utterance, called the *query*, in a large collection of speech audio. The standard approaches to this problem have made use of lattice indexing techniques (Miller et al. 2007). These approaches operate by first building a lattice, a directed graph that captures which words were likely to have been spoken over different intervals of time. A path of directed edges through this lattice corresponds to a sequence of words, and numbers associated with the edges allow us to assign probabilities or scores to these paths (Jelinek 1997; Allauzen et al. 2004; Chelba and Acero 2005) Lattice indexing proceedings by building a structure for finding in the lattice all words from a given vocabulary V. Typically (see, for example, Miller et al. 2007), this structure locates, for each word $w \in V$, a set of arcs in the lattice likely to correspond to instances of word w. Given a query word q, the standard speech recognition pipeline is applied to q to obtain a ranking of V (or sequences of words from V), and occurrences of the words or word sequences appearing high in this ranking are retrieved from the index. These lattice-based techniques can be extended to query-by-example search, in the query q is presented as an audio segment, with some additional work. For example, in Parada et al. (2009), an acoustic model is used to map the query audio to a sequence of states before applying standard lattice indexing.

A.3 Dynamic Time Warping

These lattice-based techniques have made it possible to search thousands of hours of speech audio quickly, but they require large collections of training data, both in the form of annotated audio and large collections of text. As such, the standard approaches to speech audio search are infeasible in the zero- and low-resource settings, where little to no audio is available annotated at the requisite level of detail. These constraints have motivated the use of dynamic time warping (DTW; Sakoe and Chiba 1978) to avoid the need for large collections of annotated training data (Park and Glass 2008; Jansen et al. 2010; Glass 2012; Anguera and Ferrarons 2013).

DTW seeks to align two time series so as to minimize the total cost of that alignment. Formally, let $\mathcal{X}$ denote the set of all vector times series such that for all $X_i \in \mathcal{X}$, we have $X = x_1^{(i)}, x_2^{(i)}, \ldots, x_{m_i}^{(i)}$, where each $x_j \in \mathbb{R}^p$. Suppose that we also have a distance function $\rho : \mathbb{R}^p \times \mathbb{R}^p \mapsto \mathbb{R}_{\geq 0}$. An alignment π of sequence X_i with sequence X_j can be represented by a pair of functions $\pi_1 : [p_{ij}] \to [m_i]$ and $\pi_2 : [p_{ij}] \to [m_j]$, obeying

$$1 = \pi_1(1) \leq \pi_1(2) \leq \cdots \leq \pi_1(p_{ij}) = m_i$$

$$1 = \pi_2(1) \leq \pi_2(2) \leq \cdots \leq \pi_2(p_{ij}) = m_j,$$

for some $p_{ij} \leq m_i + m_j - 1$, and

$$(\pi_1(k+1) - \pi_1(k), \pi_2(k+1) - \pi_1(k)) \in \{(0,1),(1,0),(1,1)\}$$

for all $1 \leq k < p_{ij}$. Given an alignment π specified by functions π_1, π_2, we can define the cost of alignment π as

$$D(X_i, X_j, \pi) = \sum_{k=1}^{p_{ij}} \rho(x_{\pi_1(k)}^{(i)}, x_{\pi_2(k)}^{(j)}).$$

With this definition in hand, the DTW distance between time series X_i and time series X_j is given by

$$\text{DTW}(X_i, X_j) = \min_{\pi \in \mathcal{A}(X_i, X_j)} D(X_i, X_j, \pi),$$

where $\mathcal{A}(X_i, X_j)$ denotes the set of all alignments of X_i with X_j.

The DTW alignment of two sequences can be computed using standard dynamic programming techniques (Bertsekas 2000). The naïve approach to DTW-based search simply aligns the query audio with the search collection, as done in Park and Glass (2008). Unfortunately, these approaches require time that scales as $m_i m_j$, which is infeasible if the sequences to be aligned are long. This has motivated a large body of work dedicated to speeding up or approximating DTW alignment (see, for example, Fu et al. 2005; Rakthanmanon et al. 2012, and citations therein). We discuss here three approaches specific to speech recognition.

Zhang and Glass (2011) attempt to speed DTW computations by finding a lower bound on DTW alignment that allows one to terminate computations early once it is known that the alignment is too large. Their lower bound is designed to work for alignment of phone posteriorgrams, in which each frame is a probability vector. These frames, say $u, v \in \mathbb{R}^p$, have a natural distance given by $d(u, v) = -\log u^\top v$, and since the entries of these vectors are non-negative, a simple lower bound on DTW can be found by replacing one of the sequences to be aligned with an upper-bound envelope, defined by an entrywise sliding-window maximum. This lower bound can be computed more quickly, at the cost of a weaker lower bound, by replacing the two sequences to be aligned with piece-wise constant approximations.

The RAILS system (Jansen and Van Durme 2012), discussed in Chapter 2, avoids computing an exhaustive alignment of the query audio with the search collection by using fast approximate near-neighbor retrieval techniques (see Appendix C for discussion) to retrieve frame-level near-neighbors of the query frames from the search collection. These near neighbor frames, along with their scores, yield a sparse approximation to the frame-level similarity matrix, the entries of which correspond to similarities between frames in the query and frames in the search collection. Segments of the search audio that are similar to the query give rise to approximately diagonal lines in the similarity matrix. These diagonal lines appear as peaks in the Hough transform (Duda and Hart 1972) of the matrix, and thus can be located quickly.

Mantena and Anguera (2013) present a similarly-motivated query-by-example speedup that avoids computing a complete alignment of the query audio with the search collection and instead performs a frame-level search, similar to that in the RAILS system. This frame-level search finds frames from the search audio similar to each of the frames in the query using a tree-like index structure in which nodes correspond to k-means clusters. At query time, frame-level matches of the query frames are retrieved from this index, and these are then expanded to segmental matches, similarly to Jansen and Van Durme (2012).

Appendix B

Representation Learning

In this section, we give an overview of work related to dimension reduction and manifold learning. We cannot hope to comprehensively discuss this field, but we collect here some of the landmark results and commonly-used algorithms as well as some of the standard theoretical tools. Throughout this section, we will use $x_1, x_2, \ldots, x_n$ to denote a set of n observed data points, which are assumed to lie either in Euclidean space $\mathbb{R}^p$ or in a more general space $\mathcal{X}$. Which of these two spaces we are working in at a given time will be clear from context.

B.1 Dimension Reduction

In many applications, we are given data points $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$, where the dimension p is quite large. Indeed, we may have $p \gg n$, often termed *large p small n*

problems (Johnstone and Titterington 2009). For p even moderately large, working with p-dimensional data may be intractable, and this motivates the dimensionality-reduction problem: broadly speaking, given points $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$, and given $d \ll p$, we wish to find a mapping $f : \mathbb{R}^p \to \mathbb{R}^d$ such that the points $f(x_1), f(x_2), \ldots, f(x_n) \in \mathbb{R}^d$ approximate some property of the points $\{x_1, x_2, \ldots, x_n\}$. For example, we may want to (approximately) preserve the distances between the data:

$$\|x_i - x_j\| \approx \|f(x_i) - f(x_j)\| \text{ for all } i, j \in [n].$$

We note that the norms on either side of the preceding (approximate) equation need not, in general, be the same norm.

Similar problems have been studied extensively in the theoretical computer science literature, but these *metric embedding* results are largely outside the scope of this thesis. We refer the reader to the excellent surveys Indyk (2001) and Linial (2002).

B.1.1 Random Projections

A classic approach of dimensionality reduction is given by the celebrated Johnson-Lindenstrauss Lemma (Johnson and Lindenstrauss 1984). We state here the version proved in Dasgupta and Gupta (2003), though we note that many variants of this theorem have been published improving the bounds in various ways and extending the kinds of random functions for which such theorems hold. See, for example, Kane and Nelson (2014); Larsen and Nelson (2016) and citations therein.

Theorem 8. *For any $0 < \epsilon < 1$, if $d \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n$, then for any set of n points $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$, there exists a function $f : \mathbb{R}^p \to \mathbb{R}^d$ such that for all $i, j \in [n]$*

$$(1-\epsilon)\|x_i - x_j\|_2^2 \leq \|f(x_i) - f(x_j)\|_2^2 \leq (1+\epsilon)\|x_i - x_j\|_2^2. \tag{B.1}$$

The function f can be found in randomized polynomial time (i.e., $O(n^c)$ runtime for some constant $c \geq 0$ and succeeds with constant probability).

In Dasgupta and Gupta (2003), the function f is taken to be a projection onto a randomly-chosen d-dimensional subspace of $\mathbb{R}^p$, but as mentioned above, other versions of the lemma are possible. For example, one can prove a similar result for the case where the map f is given by a matrix of 0-mean Gaussians, (Baraniuk et al. 2008). Theorems of this sort are examples of the *concentration of measure* phenomenon (Milman and Schechtman 1986; Boucheron et al. 2013), in which a sum of random variables lies, with high probability, close to its expected value.

B.1.2 PCA and Related Methods

A now-classic dimensionality-reduction technique in machine learning is principle component analysis (PCA; Jolliffe 2002), in which the data are projected onto the top d eigenvectors of the sample covariance matrix. That is, if $C \in \mathbb{R}^{p \times p}$ is the sample covariance matrix of an i.i.d.sample $X_1, X_2, \ldots, X_n \in \mathbb{R}^p$, we take $f(X_i)$ to be the projection of X_i onto the d eigenvectors of C with largest-magnitude eigenvalues (equivalently, onto the top d right singular vectors of the matrix $\mathbf{X} = [X_1' X_2' \cdots X_n']^T \in \mathbb{R}^{n \times p}$, where $X_i' = X_i - \sum_{j=1}^n X_j/n$). Intuitively, PCA attempts to reduce the dimensionality of the data by projecting it onto a d-dimensional subspace that best-preserves the covariance structure of the data.

Candès et al. (2011) considered performing PCA under the condition that a few entries of the vectors $X_1, X_2, \ldots, X_n \in \mathbb{R}^p$ are measured incorrectly. That is, for each $i = 1, 2, \ldots, n$ and each $j = 1, 2, \ldots, p$, there is a small probability that the j-th entry of X_i is corrupted by arbitrary noise. Using techniques from compressed sensing and sparse recovery (see Appendix D for a discussion of these techniques), Candès et al. (2011) showed that low-dimensional structure present in the data can still be (approximately) recovered via linear programming. We note that this work was hardly the first to consider the problem of performing PCA in the presence of non-Gaussian noise. For example, de la Torre and Black (2003) used an approach inspired by M-estimators (Huber 2009; van der Vaart 2000) to develop a version of PCA that is robust to outlier observations. Shahid et al. (2015) adapted the robust PCA devised by Candès et al. (2011) by including additional structure in the form of a similarity graph. We discuss graph-based approaches of this sort at more length below in Section B.2.2.

We note that there exist a number of supervised variants of PCA, in which the learned embedding takes into account known supervisory information in the form of labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$ and $y_i \in [k]$. The classic example is linear discriminant analysis (LDA; Rao 1948), in which the data are assumed to be generated according to a mixture of k Gaussians with means $\{\mu_j\}_{j=1}^k$ with shared covariance matrix Σ. The goal of LDA is to find a d-dimensional projection of the data that captures a maximal amount of the between-class variance and a minimal amount of the within-class variance. LDA projects the data onto the top d eigenvectors of the matrix $S = C^{-1}C_B$, where C is an estimate of Σ and C_B is an estimate of the between-class covariance, given by

$$C_B = \frac{1}{k}\sum_{j=1}^{k}(\hat{\mu}_j - \hat{\mu})(\hat{\mu}_j - \hat{\mu})^T,$$

where $\hat{\mu}$ is an estimate of the global mean and $\{\hat{\mu}_j\}_{j=1}^k$ are estimates of the means of the k Gaussians. Our goal is to find an orthonormal set of d vectors $\{u_j\}_{j=1}^d \subseteq \mathbb{R}^p$ maximizing

$$\sum_{j=1}^{k}\frac{u_j^T C_B u_j}{u_j^T C u_j}.$$

A standard use of Lagrange multipliers (Boyd and Vandenberghe 2004) yields that the solutions are the top d solutions to the eigenproblem $C_B u = \lambda C u$. We note that it is common to apply shrinkage to the covariance matrix estimates as done in the LDA embeddings presented in Chapter 2.

Metric learning to rank (MLR; McFee and Lanckriet 2010) is a supervised learning technique that constructs a metric on $\{x_1, x_2, \ldots, x_n\}$ that is well-suited for query-by-example and near-neighbor problems. Training data comes in the form of a ranking, for each $q \in \{x_i\}_{i=1}^n$, of the elements of $\{x_i\}_{i=1}^n$ according to their quality as query results for query q. McFee and Lanckriet (2010) use techniques from structural SVM training (Tsochantaridis et al. 2005) to learn from these rankings a positive semidefinite matrix $W \in \mathbb{R}^{p \times p}$ that specifies a distance

$$d_W(x, y) = \sqrt{(x - y)^T W (x - y)}.$$

We observe that it is possible to embed the data $\{x_i\}_{i=1}^n$ according to $f(x_i) = W^{1/2} x_i$, and one can perform dimensionality reduction by choosing W to have $\operatorname{rank} W \ll p$ and representing $W^{1/2} x_i$ in a suitable basis.

B.1.3 Kernel Methods

The assumption of linear structure inherent in PCA reduces our problem to one of finding eigenvalues, but if the data have a more complicated nonlinear structure, PCA may fail to capture it. One approach to this issue is to make use of the *kernel trick* (Hofmann et al. 2008), in which we assume that the data have a linear structure in some inner product space $\mathcal{H}$. The kernel trick amounts to choosing a kernel function $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ that is equivalent to the inner product in $\mathcal{H}$. That is, so that

$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$$

for some function $\phi : \mathbb{R}^p \to \mathcal{H}$ called the *feature map.* The kernel trick and related techniques are widely used in machine learning and statistics owing to the simple and elegant properties of positive definite kernels and reproducing kernel Hilbert spaces (RKHS; Hofmann et al. 2008; Berlinet and Thomas-Agnan 2004). Applying the kernel trick to PCA, we obtain kernel PCA (Schölkopf et al. 1998), in which we perform PCA implicitly on the points $\{\phi(x_i)\}_{i=1}^n \subset \mathcal{H}$ rather than on $\{x_i\}_{i=1}^n \subset \mathbb{R}^p$. Ham et al. (2004) observe that several of the commonly-used manifold learning algorithms, which we discuss below, can be viewed as applying kernel PCA to suitably chosen Gram matrices.

There exists a large body of work dedicated to speeding up the construction of the kernel matrix $K = [k(x_i, x_j)]$ and its eigendecomposition for techniques such as kernel PCA. Indeed, similar concerns are among the motivations for the problem considered in Chapter 3. The two most prominent approaches to approximating the eigenvectors and eigenvalues of K have been the Nyström method (Delves and Mohamed 1985; Williams and Seeger 2001; Drineas and Mahoney 2005) and random features (Rahimi and Recht 2008, 2009; Le et al. 2013).

Achlioptas et al. (2002) explored three approaches to quickly approximating the relevant information in kernel K for performing kernel PCA. The first is a sparsification and quantization of K, with a bound that follows similar reasoning to our use of the Davis-Kahan Theorem (Davis and Kahan 1970) in proving the main result in

Chapter 3. The second pertains to the computation of the expansion

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i).$$

By retaining only the large-magnitude components of the vector $(\alpha_i)_{i=1}^n$ and applying randomized rounding to the remaining values, Achlioptas et al. (2002) obtain $\hat{f}(x)$, an unbiased estimate of $f(x)$. Standard concentration arguments show that this estimate is close to $f(x)$, and furthermore that many entries of the rounded vector $(\hat{\alpha}_i)_{i=1}^n$ are zero, i.e., the rounded vector is sparse. A third approach to approximating K applies a random projection to x and y. By the Johnson-Lindenstrauss Lemma discussed above, this approximately preserves the geometry of the input space and hence allows a fast approximation of $k(x, y)$ in the case where $k(\cdot, \cdot)$ is a function only of the distance or inner product between x and y.

The algorithm presented in Smola and Schölkopf (2000) seeks to approximate K by a matrix $\tilde{K}$ expressible as a linear combination of $q \ll n$ columns of n. The authors presented a randomized algorithm for choosing these columns' indices $I \subset [n]$ that operates by repeatedly

1. choosing greedily from among a randomly chosen subset of the columns of K based on an estimate of the improvement in residuals
2. updating the columns of K to reflect the signal not yet accounted for.

Fine and Scheinberg (2001) developed a similarly-motivated interior point method for computing a low-rank approximation of the kernel matrix and presented an accompanying bound on the approximation error. Their method approximates K by approximating its Cholesky factorization (Golub and Loan 2013) via repeated rank-1 updates.

Drineas and Mahoney (2005) applied methods for fast matrix multiplication and decomposition (Drineas et al. 2006a,b) to develop a sampling strategy for choosing $\{x'_j\}_{j=1}^q$, and proved bounds on the resulting approximation error in recovering the full kernel matrix K. These bounds show that one can obtain a rank-q approximation $\tilde{K}^{(q)}$ to K such that with high probability, $\tilde{K}^{(q)}$ approximates the true kernel matrix K nearly as well as the best possible rank-q approximation. This approximation requires that we be able to sample the rows and columns of K according to a probability distribution that depends on the values $\{k(x_i, x_i)\}_{i=1}^n$. A similar result was proved in Frieze et al. (2004), though the sampling algorithm requires a more complicated probability distribution over the rows and columns of K. An empirical comparison of several variations on the Nyström method by Kumar et al. (2009) showed that uniform sampling of the points as done in Williams and Seeger (2001) outperforms more complicated sampling schemes such as the one in Drineas and Mahoney (2005). In light of this observation, they presented theoretical bounds for the performance of the Nyström method, showing that with high probability, a uniform sampling version of the Nyström method approximates the kernel matrix nearly as well as does the best rank-q approximation.

An approach related to but distinct from the Nyström method is that of finding a *reduced set*. First introduced by Burges (1996) in the context of support vector machines, the reduced set problem is to choose a set of $q \ll n$ points $\{x'_1, x'_2, \ldots, x'_q\} \subset \{x_1, x_2, \ldots, x_n\}$ so that $\phi(x) \in \mathcal{H}$ for different values of x can be well approximated by linear combinations of the $\{\phi(x'_i)\}_{i=1}^q$ rather than of the larger set $\{\phi(x_i)\}_{i=1}^n$. Thus, rather than sampling points, one explicitly constructs a set of points that would be a good sample from the standpoint of the Nyström method. By considering the problem of finding preimages of features $\phi(x)$, Schölkopf et al. (1999) recast the reduced set selection problem as an eigenproblem similar to the kernel PCA problem itself, and developed an approximate solution to this problem. They also presented a different approach to the reduced set problem based on an L_1-regularized minimization problem that encourages sparse coefficients in expressing $\phi(x)$.

Random features (Rahimi and Recht 2008, 2009) refers to a class of methods that seek to use sampling to approximate the sum

$$k(x, y) = \sum_{i=1}^{N} \lambda_i u_i(x) u_i(y) = \langle \phi(x), \phi(y) \rangle,$$

where N is the (possibly infinite) number of nonzero eigenvalues of the kernel $k(\cdot, \cdot)$ and u_i are the corresponding eigenfunctions. We note that $\langle \phi(x), \phi(y) \rangle = k(x, y)$ is an inner product in a high-dimensional space $\mathbb{R}^{p'}$, where $p' \gg p$. Rahimi and Recht (2008) considered replacing $k(x, y)$ with a cheap inner product that approximates this high-dimensional inner product by transforming the observations as $x \mapsto g(x) \in \mathbb{R}^r$, where $r \ll p'$, so that $\langle g(x), g(y) \rangle \approx k(x, y)$. This allows us to replace the expansion $f(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i)$ with the approximation $f(x) \approx \langle w, g(x) \rangle$, for a suitably chosen vector $w \in \mathbb{R}^r$. The transformation $g : \mathbb{R}^p \to \mathbb{R}^r$ is given by a random matrix $Z \in \mathbb{R}^{r \times p}$ with entries drawn as i.i.d. normals with 0 mean and suitably chosen variance. Rahimi and Recht (2009) generalized this technique to a broader class of learning problems. Le et al. (2013) reduced the space and time required for storage and application of the matrix Z by replacing the matrix of Gaussians with a product of a Hadamard matrix and random diagonal matrices.

B.2 Manifold Learning

The methods considered above tend to assume that the data have an inherent linear structure, either in the ambient space $\mathbb{R}^p$ or in some high-dimensional feature space. However, it is often the case that data has an inherently low-dimensional but non-linear structure. Consider, for example, image data generated by photographing an object from different angles and under different lighting directions. These observations are of a dimension equal to the number of pixels in the images, but they have an inherent low-dimensional structure in the sense that they can be reparameterized by specifying the angle and light direction of each photo was taken (see Figure 1a in Tenenbaum et al. 2000, for an illustration).

More formally, suppose that the data $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$ lie on or near a d-dimensional ($d \ll p$) surface $\mathcal{S}$ in $\mathbb{R}^p$. It stands to reason that if this is the case, then we should be able to approximately represent the data points in d dimensions rather than p. If $\mathcal{S}$ is simply a hyperplane, then PCA will find this structure, but if $\mathcal{S}$ is, for example, a swiss roll as pictured in Figure 3.1, then d-dimensional PCA will not adequately capture the information present in the data, even though the data are, in some sense, inherently d-dimensional. This intuition is the motivation for *manifold learning*, sometimes called *nonlinear dimensionality reduction*, in which we assume that the data lie on a low-dimensional surface in the ambient space $\mathbb{R}^p$. As high-dimensional data have become central to machine learning and the sciences as a whole, manifold learning has become a standard tool for data exploration and analysis. We refer the reader to the surveys by van der Maaten et al. (2009) and Bengio et al. (2013) as well as the textbook by Lee and Verleysen (2007) for a more thorough discussion of the relevant ideas and a comparison of popular techniques.

A wide variety of manifold learning techniques and algorithms exist, but they can largely be divided into two sets of approaches. The first, exemplified by multidimensional scaling (MDS Torgerson 1952) and ISOMAP (Tenenbaum et al. 2000), seek to preserve the global geometry of the data, so that $\|f(x_i) - f(x_j)\| \approx \|x_i - x_j\|$ for all $i, j \in [n]$, for example as in (B.1). The second, exemplified by Laplacian eigenmaps (Belkin and Niyogi 2003; Belkin et al. 2006) and locally linear embeddings (LLE Roweis and Saul 2000; Saul and Roweis 2003), do not attempt to preserve the global geometry of the full data. Instead, these methods seek only to preserve the local geometry of the data, so that points near each other in the original space $\mathbb{R}^p$ are also near each other in $\mathbb{R}^d$ after application of the function f. This latter approach relaxes the global preservation of distances under the intuition that for most applications, the local information contained in the data is paramount. Most of these local geometry-based techniques have the useful property that, similar to algorithms based on rigidity theory (see, for example Javanmard and Montanari 2013), the embedding depends only on assessing the similarity of nearby points. That is, we can construct an LLE embedding even if we are only able to locate each point in relation to its neighbors.

B.2.1 Global methods

Torgerson (1952) introduced multidimensional scaling (MDS), in which observations from some space $\mathcal{X}$ are embedded based on their squared dissimilarities, represented by a matrix

$$\Delta = [d^2(x_i, x_j)] \in \mathbb{R}^{n \times n}$$

for some dissimilarity function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Note that MDS does not require that our observations $x_1, x_2, \ldots, x_n$ lie in Euclidean space. Indeed, it does not even require that $d(\cdot, \cdot)$ be a metric, only that it be nonnegative, symmetric and satisfy $d(x, x) = 0$. Classical MDS embeds these observations in Euclidean space according to the eigenvalues and eigenvectors of $B = -H\Delta H/2$, where $H = I - ee^T/n$ denotes the "recentering" matrix. Many variants of MDS have been posited since Torgerson (1952). See Borg and Groenen (2005) for detailed discussion.

The ISOMAP algorithm (Tenenbaum et al. 2000) expands upon MDS by seeking to preserve a specific notion of distance, given by the shortest-path distance on a near-neighbor graph. The algorithm first constructs a graph G with vertices corresponding to the observations $x_1, x_2, \ldots, x_n$, and vertices i and j sharing an edge of weight $d(x_i, x_j)$ if and only if one is within the k nearest neighbors of the other for some user-specified k. Letting $d_G(i, j)$ denote the shortest-path distance in G, the observations are embedded according to MDS applied to the matrix $[\delta_{ij}] = [d_G^2(i, j)]$. As pointed out by Tenenbaum et al. (2000), this algorithm escapes the linearity of PCA and MDS by using the shortest path distance as an approximation to the geodesic distance on the manifold.

B.2.2 Local Methods

Most embedding techniques based on local manifold structure operate by encoding the local geometry of the data in a graph. Another common approach is to view the data as encoding a random walk or diffusion process, which captures the structure of the underlying manifold. These diffusion-based approaches have the advantage of being able to capture both the geometry of the data and the density of those points in space. We give an overview of a few of these local methods here. We note that many of these methods can be viewed as special cases of a single graph embedding framework (Yan et al. 2007).

Locally linear embeddings (LLE), introduced by Roweis and Saul (2000), is among the earliest examples of the local geometry-based approaches to manifold learning. LLE seeks to recover low-dimensional structure inherent in a set of high-dimensional observations $x_1, x_2, \ldots, x_n \in \mathbb{R}^p$, Rather than attempting to preserve the global geometry of the data, LLE attempts to embed the observations into $\mathbb{R}^d$ so as to preserve the neighborhood geometry of the data. The algorithm represents each observed data point x_i as an affine combination of its nearest neighbors, seeking to minimize a sum of squared errors of the form

$$\sum_{i=1}^{n} \|x_i - \sum_{j \neq i}^{n} W_{i,j} x_j\|^2, \tag{B.2}$$

where the weight matrix $[W_{i,j}] \in \mathbb{R}^{n \times n}$ is subject to the constraints that $W_{i,j} = 0$ for all j for which x_j is not among the near neighbors of x_i and such that $\sum_{j=1}^{n} W_{i,j} = 1$ for all $i \in [n]$. The data are then embedded as $y_1, y_2, \ldots, y_n \in \mathbb{R}^d$ by fixing the weight matrix W and choosing the $\{y_i\}_{i=1}^{n}$ to minimize

$$\sum_{i=1}^{n} \|y_i - \sum_{j \neq i}^{n} W_{i,j} y_j\|^2, \tag{B.3}$$

subject to constraints on the centering and scaling of the $\{y_i\}$. LLE has the advantage of being comparatively easy to solve, since (B.2) can be solved trivially by considering each row of W independently and (B.3) can be reframed as solving an eigenvalue problem for the sparse matrix $M = (I - W)^T(I - W)$. A robust version of LLE was presented by Chang and Yeung (2006), using techniques from Holland and Welsch (1977) and de la Torre and Black (2003) to prevent outliers from significantly corrupting the estimated weights.

Stochastic neighbor embeddings (SNE Hinton and Roweis 2002) attempts to construct a low-dimensional embedding by replacing the hard near-neighbor assignments that appear in most local embedding procedures with a probabilistic notion of near neighbors. In this sense, SNE is not a strictly local technique, since it produces an embedding that takes all pairwise dissimilarities into account. For each observation x_i, SNE constructs a probability distribution over $\{x_{ij} : j \neq i\}$, with probabilities proportional to $\exp\{-d^2(x_i, x_j)\}$. The points $x_1, x_2, \ldots, x_n$ are them embedded by finding points $y_1, y_2, \ldots, y_n \in \mathbb{R}^d$ such that the analogous probabilities $q_{i,j} \propto \exp\{-\|y_i - y_j\|^2\}$ minimize the sum of KL-divergences

$$\sum_{i=1}^{n} \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}} = \sum_{i=1}^{n} D(P_i \parallel Q_i).$$

This minimization can be done via gradient descent, but later work (Cook et al. 2007; van der Maaten and Hinton 2008) refined this approach by replacing this cost function with one in which the $\{p_{ij}\}$ and $\{q_{ij}\}$ terms are interpreted as specifying distributions over $1 \leq i < j \leq n$, rather than specifying n separate distributions.

Laplacian eigenmaps (Belkin and Niyogi 2003; Belkin et al. 2006) seeks to embed the observations $x_1, x_2, \ldots, x_n \in \mathcal{X}$ into $\mathbb{R}^d$ so as to preserve the local geometry of the data. This local geometry is encoded by an undirected graph G on n nodes corresponding to the observations. This graph can constructed in any of several ways, provided it adequately captures the local geometry of the observations. For example, the graph be either binary or weighted according to a similarity function (e.g., a Gaussian kernel). Similarly, it can be constructed either as the near-neighbor graph, in which nodes i and j share an edge if and only if x_i is among the nearest neighbors of x_j (or vice versa) or as an ϵ-graph, in which i and j share an edge if and only if x_i and x_j are suitably close to one another. The Laplacian eigenmaps embedding arises from attempting to embed the observations $x_1, x_2, \ldots, x_n$ as $y_1^{(1)}, y_2^{(1)}, \ldots, y_n^{(1)} \in \mathbb{R}$ so as to minimize the weighted sum of squares

$$\sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} (y_i^{(1)} - y_j^{(1)})^2 \tag{B.4}$$

subject to certain orthogonality constraints. Letting $D \in \mathbb{R}^{n \times n}$ be the diagonal *degree matrix* of the (possibly weighted) graph G with entries $D_{ii} = \sum_{j=1}^{n} W_{ij}$, we can rewrite (B.4) as

$$\sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} (y_i^{(1)} - y_j^{(1)})^2 = \langle y^{(1)}, L y^{(1)} \rangle,$$

where $L = D - W$ is the combinatorial graph Laplacian. Repeating this procedure subject to orthogonality constraints on the embeddings $\{y^{(t)}\}_{t=1}^{d}$ (see, for example Bhatia 1997, Corollary III.1.2), we obtain an embedding of the observations as $y_1, y_2, \ldots, y_n \in \mathbb{R}^d$ according to the d non-trivial solutions to the eigenproblem $Lf = \lambda Df$ with smallest eigenvalues (note that $f = \vec{\mathbf{1}}, \lambda = 0$ is a trivial solution). We note that the version of Laplacian eigenmaps first developed by Belkin and Niyogi (2003), which we present here, differs from the one considered in Chapter 2 and studied in Chapter 3, where we considered the matrix $\mathscr{L} = D^{-1/2}(D - W)D^{-1/2}$, which we have found to yield better embeddings of the data. This matrix, typically called the *symmetric normalized graph Laplacian*, corresponds to a different weighting the vertices of G (see, for example, Chung 1997). Using techniques based on the Nyström method, Bengio et al. (2004) developed an out-of-sample extension for Laplacian eigenmaps as well as for LLE, ISOMAP and MDS, allowing one to embed previously unseen data according to the embedding of $x_1, x_2, \ldots, x_n$. Belkin et al. (2006) developed a regularized out-of-sample extension of the Laplacian eigenmaps embedding, in which the solution is penalized in both the intrinsic geometry of the manifold and in the ambient space $\mathbb{R}^p$.

As discussed in Belkin and Niyogi (2003), the graph Laplacian can be viewed as a discrete analogue of the Laplace-Beltrami operator, defined on functions $f : \mathcal{M} \to \mathbb{R}$ by $\Delta f = \nabla^2 f$ (see Rosenberg 1997, for detailed exposition). Indeed, this connection becomes clear in light of the results proven by Belkin and Niyogi (2005), showing that under suitable conditions the finite-sample Laplacian operator

$$L_n^t(f)(x) = f(x)\sum_{j=1}^{n} \exp\left\{\frac{-\|x - x_j\|^2}{4t}\right\} - \sum_{j=1}^{n} f(x_j)\exp\left\{\frac{-\|x - x_j\|^2}{4t}\right\}, \qquad \text{(B.5)}$$

based on a uniform sample $\{x_1, x_2, \ldots, x_n\}$ from $\mathcal{M}$, converges pointwise in probability to the the Laplace-Beltrami operator associated with $\mathcal{M}$. The core proof technique hinges on relating the Laplace operator to the heat operators on the manifold $\mathcal{M}$, and observing that for t suitably large, the Gaussians in (B.5) are good approximations to the heat kernel. Similar, more general, results were presented by Hein et al. (2005) using an approach based on smoothing operators in which the kernel function plays a role similar to the heat kernel in Belkin and Niyogi (2005). Based on these convergence results and the connections of the Laplacian to diffusion processes, Hein and Maier (2007) developed an algorithm for denoising points sampled from a manifold $\mathcal{M}$.

Rohe et al. (2011a) theoretically and empirically explored the consistency of Laplacian eigenmaps applied to a binary adjacency matrix followed by k-means clustering. In the language of Chapter 3, they considered the kernel matrix $\mathscr{K} \in \mathbb{R}^{n\times n}$ on a fixed (but unknown) subset $\mathcal{X} \subset \mathbb{R}^p$. From this kernel, they observed the matrix

$Y \in \{0,1\}^{n\times n}$ with independent entries

$$Y_{ij} = Y_{ji} = \begin{cases} 1 & \text{with probability } \mathscr{K}_{ij} \\ 0 & \text{with probability } (1 - \mathscr{K}_{ij}). \end{cases}$$

They compared the Laplacian spectral embedding based on $\mathscr{K}$ with that based on Y. Their key result showed that, similar to our main result in Chapter 3, under some mild assumptions on the spectrum of $\mathscr{L}(\mathscr{K})$ (the normalized Laplacian of $\mathscr{K}$), the eigenspace of $\mathscr{L}(Y)$ does not significantly differ from the corresponding eigenspace of $\mathscr{L}(\mathscr{K})$ (after suitable rotation). As a result, they prove that spectral clustering of $\mathscr{L}(Y)$ consistently estimates the clusters obtained by spectrally clustering $\mathscr{L}(\mathscr{K})$. Similar ideas were explored in an earlier paper by von Luxburg et al. (2008), where the authors established the consistency of both the normalized and unnormalized Laplacians using techniques from empirical process theory (van der Vaart and Wellner 1996) and perturbation theory (Kato 1995).

Trosset and Tang (2010) demonstrated an interesting connection between combinatorial Laplacian eigenmaps, classical MDS and ISOMAP by showing that combinatorial Laplacian eigenmaps generate embeddings equivalent to applying MDS so as to preserve the commute times of the random walk defined by transition matrix $P = D^{-1}W$. embedding the data according to MDS. This facilitates direct comparison of Laplacian eigenmaps and ISOMAP provided one has a suitable mapping between similarities and distances on the data. This random walk interpretation of the Laplacian provides one of the main intuitions for why Laplacian eigenmaps and similar methods should be expected to discover cluster structure in data. One expects clusters to manifest as low-potential wells in the stationary distribution of the random walk defined by P (see von Luxburg 2007, for further discussion).

Donoho and Grimes (2003) combine ideas from LLE (Roweis and Saul 2000) and Laplacian eigenmaps (Belkin and Niyogi 2003) to develop Hessian eigenmaps. Hessian eigenmaps attempts to capture the local structure of the data $\{x_1, x_2, \ldots, x_n\} \subseteq \mathbb{R}^p$ by embedding the points according to the local coordinates of the tangent spaces of the d-dimensional data manifold $\mathcal{M}$. The method adapts the intuition of Laplacian eigenmaps in that is uses, in place of the Laplace-Beltrami operator the Hessian operator, defined by

$$\mathcal{H}(f) = \int_{\mathcal{M}} \|H_f(z)\|_F^2 dz,$$

where $f : \mathcal{M} \to \mathbb{R}$ and $H_f(z)$ denotes the Hessian of f at z.

Diffusion maps (Coifman and Lafon 2006) expands upon the intuition behind Laplacian eigenmaps by constructing embeddings based on a random walk on the data. Given data $\mathcal{S} = \{x_1, x_2, \ldots, x_n\}$, diffusion maps seeks to embed the observations based on the behavior of a Markov chain on state space $\mathcal{S}$ with transition matrix

$$P = [P_{ij}] = [p(x_i, x_j)] = [\frac{k(x_i, x_j)}{d(x_i)}],$$

where

$$d(x) = \sum_{i=1}^{n} k(x, x_i).$$

Letting $(P^t)_{i,j} = p_t(x_i, x_j)$ denote the probability that a t-step random walk started at x_i ends at x_j, diffusion maps aims to preserve the diffusion distances

$$D_t(x, y) = \|p_t(x, \cdot) - p_t(y, \cdot)\|_2.$$

Following this reasoning, the diffusion maps embedding of $\mathcal{S}$ into $\mathbb{R}^d$ is given, for a user-specified value of t, by $f_t(x_i) = [\lambda_1^t u_1(x_i), \lambda_2^t u_2(x_i), \ldots, \lambda_d^t u_d(x_i)]^T$, where $u_j(x_i)$ denotes the component of the j-th eigenvector of P^t corresponding to observation x_i. Coifman and Lafon (2006) approximate the diffusion distance D_t to arbitrary precision by choosing d suitably large. That is, $D_t(x_i, x_j) \approx \|f_t(x_i) - f_t(x_j)\|$, with the approximation becoming good for t large. Nadler et al. (2006) applied this framework to analysis of data generated from dynamical systems, in which either the density or geometry of the data may be of interest, depending on the application.

Locality sensitive discriminant analysis (LSDA; Cai et al. 2007) adapts LDA to the setting where we wish to capture local structure rather than global structure of the data. Rather than considering within- and between-class covariances, one considers a pair of near-neighbor graphs G_w and G_b that encode the within- and between-class local structures, respectively. LSDA embeds the data so as to maximize the distances between embeddings of observations from different classes while simultaneously embedding same-class observations near one another. Tomar and Rose (2012) followed a similar line of reasoning in developing locality preserving discriminant analysis (LPDA), which embeds the data so as to maximize a measure of between-class scatter while minimizing a measure of within-class scatter. Broadly similar ideas were explored earlier by Cai et al. (2007), using a different optimization approach.

Finally, we briefly discuss a related embedding method, suited for graph data, adjacency spectral embedding (ASE; Sussman et al. 2012). The ASE embedding arises naturally from latent position models of graphs (Hoff et al. 2002), in particular from random dot product graphs (RDPGs; Young and Scheinerman 2007). In the RDPG model, a random graph $G = (V, E)$ on $|V| = n$ vertices is generated, conditioned on an assignment of vertices $u \in V$ to latent positions $x_u \in \mathbb{R}^d$, subject to the condition that $x_u^T x_v \in [0, 1]$ for all $u, v \in V$. Typically, these latent positions are drawn identically and independently according to some distribution F on $\mathbb{R}^d$. Conditioned on these latent positions, edges are present or absent in graph G independently according to $\Pr[\{u, v\} \in E] = x_u^T x_v$. Working in this model, a natural inference task is to recover the latent positions based on the observed graph, and this motivates adjacency spectral embedding. Letting $A = U\Sigma U^*$ be the singular value decomposition of the adjacency matrix, let $U_A \in \mathbb{R}^{n\times d}$ be the matrix whose columns correspond to the top d singular vectors, and let $\Sigma_A \in \mathbb{R}^{d\times d}$ denote the matrix of the top d singular values. ASE estimates the latent positions of graph G as the rows of the matrix $\hat{X} = U_A \Sigma_A \in \mathbb{R}^{n\times d}$. It has been shown that $\hat{X}$ consistently estimates the latent positions up to an orthonormal transformation of the latent positions (Sussman et al. 2012). A number of additional asymptotic properties of ASE are known, though a detailed discussion of these results is beyond the scope of this survey. We refer the interested reader to Sussman et al. (2012); Fishkind et al. (2013); Athreya et al. (2016); Tang et al. (2013); Lyzinski et al. (2014b); Sussman et al. (2014), and highlight here only an intermediate result from Lyzinski et al. (2014b), used to prove asymptotic consistency of clustering of the estimated latent positions. Lemma 5 in Lyzinski et al. (2014b) gives a concentration result somewhat comparable to that presented in Chapter 3. In particular, it shows that under suitable model conditions the matrix $\hat{X}$ concentrates about the matrix of true latent positions X (after suitable rotation) under the $(2, \infty)$ norm.

Appendix C

Locality-Sensitive Hashing and Near-Neighbor Retrieval

In this section, we give an overview of work related to locality-sensitive hashing and near-neighbor retrieval. These techniques have become fundamental to machine learning and related fields. See, for example, Sundaram et al. (2014), in which the authors use these techniques to build a state-of-the-art system for performing search over billions of text documents.

C.1 Problem definition

Consider a metric space $\mathcal{M} = (X, d)$ and a set of points $P = \{p_1, p_2, \ldots, p_n\} \subseteq \mathcal{M}$, called the *search collection* from that metric space.

Definition 6. *k-nearest neighbor (k-NN) retrieval Given a query point $q \in \mathcal{M}$ and a non-negative integer k, find the k points from P nearest to q under distance d. That is, to retrieve a set $N_k(q) = \{x_1, x_2, \ldots, x_k\} \subseteq P$ such that for all $p \in P$ and all $i = 1, 2, \ldots, k$, we have $d(p, q) \geq d(x_i, q)$.*

One observes that in general, solving the k-NN problem exactly can be quite a challenge. Barring the existence of some additional structure on the search collection, computation of the distances $\{d(q, p) : p \in P\}$ can require time linear in the size of the search collection. Similarly, if distances are hard to compute precisely, then solving the k-NN problem exactly may be expensive. For example, consider a case where all points in P are either at distance $d(p, q) = 1$ or distance $d(p, q) = 1 + \epsilon$ for some small $\epsilon > 0$, and suppose that computing $d(p, q)$ to within an additive error of ϵ is expensive. Under such conditions, finding a suitable set $N_k(q)$ satisfying our problem statement in Definition 6 above may be quite challenging. We note, however, that in many cases, it is not necessary to solve k-NN so precisely. In the example just given, it may be the case that differences in distance of size ϵ is immaterial in terms of downstream performance. This is often in the case in recommender and information retrieval systems, and motivates the definition, initially given by Indyk and Motwani (1998), of a relaxation of the exact near neighbor problem given in Definition 6. See He et al. (2012) for an interesting approach to assessing the difficulty of near neighbor search on a given data set and query distribution.

Definition 7. **ϵ-approximate nearest neighbor (ANN) search** *Given a set* *of points $P = \{p_1, p_2, \ldots, p_n\}$ from metric space $\mathcal{M} = (X, d)$, build a data structure supporting the following operation: given a query point $q \in X$, find $p \in P$ so that for all $p' \in P$, $d(q, p') \leq (1 + \epsilon)d(p, q)$.*

Note that this problem is a relaxation of exact 1-NN retrieval problem, where we would ask to retrieve $p \in P$ so that for all $p' \in P$, $d(q, p') \leq d(p, q)$. The approximate versions of the k-NN problem for $k > 1$ can be constructed similarly.

C.2 Locality Sensitive Hashing: Initial Work

Locality sensitive hashing (LSH) was first introduced in 1998 by Indyk and Motwani (1998) for the purpose of ANN search. The goal of LSH is to define hash functions on geometric objects (i.e., points) in such a way that objects that are near one another are likely to be hashed to the same value, while objects that are distant are unlikely to be hashed to the same value. Given metric space $\mathcal{M} = (X, d)$, we have the following definition.

Definition 8. **(r_1, r_2, p_1, p_2)-sensitive hash function** *(Indyk and Motwani 1998) A family of functions $\mathcal{F} = \{f : \mathcal{X} \mapsto U\}$ is (r_1, r_2, p_1, p_2)-sensitive if the following two conditions hold for all $u, v \in \mathcal{X}$:*

(a) If $d(u, v) \leq r_1$ then $\Pr_{\mathcal{F}}[f(u) = f(v)] \geq p_1$.

(b) If $d(u, v) > r_2$ then $\Pr_{\mathcal{F}}[f(u) = f(v)] \leq p_2$.

$\Pr_{\mathcal{F}}$ *is some distribution over family* $\mathcal{F}$*.*

Indyk and Motwani (1998) point out that for family $\mathcal{F}$ to be useful for approximate nearest neighbor retrieval, we must have $r_2 > r_1$ and $p_1 > p_2$.

In the decade and a half since Indyk and Motwani's paper, LSH functions have been developed for a variety of distance functions and metric spaces. The primary focus of this line of research lies in the tradeoff between query time and index size. That is, broadly speaking, one can retrieve approximate near-neighbors quickly, at the expense of more memory usage to build a more complicated index, or one can use less memory at the cost of slower retrieval.

An improvement of the algorithm introduced in Indyk and Motwani (1998) for use in the case of high-dimensional data was presented in Gionis et al. (1999), in which the authors took advantage of the fact that under some conditions, Hamming distance can be used as a stand-in for ℓ_1 distance. Gionis et al. (1999) improved the $O(dn^{1/\epsilon})$ query time of the original paper by Indyk and Motwani (1998) to the more manageable (and sublinear, for $\epsilon > 0$) $O(dn^{1/(1+\epsilon)})$ query time.

A widely-used algorithm for LSH on points in Euclidean space was presented by Andoni and Indyk (2006). Their technique improves over the LSH functions presented in Indyk and Motwani (1998) by further improving both the query time and runtime (though we note that the two papers are not directly comparable, since they solve slightly different versions of the near-neighbor problem). The algorithm operates by first covering $\mathbb{R}^d$ with a set of randomly-chosen grids of hyperspheres, with all hyperspheres in a given grid having the same small radius. They bound the number of such grids and the radii required for use with each grid in such a way as to guarantee that the set of grids will cover the space with high probability (specifically, this is actually achieved by projecting to a lower-dimensional space that is easier to cover). These grids can be used to form a family of locality-sensitive hash functions, since if two points are near one another, it is far more likely that they will be covered by the same balls.

Shortly thereafter, Andoni et al. (2006) improved upon Andoni and Indyk (2006) by introducing a hashing scheme that performs retrieval on $\mathbb{R}^d$ under the ℓ_s norm for any $s \in [0, 2]$. They describe a scheme for LSH that makes use of p-stable distributions. Distribution D is p-stable if given $v_1, \ldots, v_n \in \mathbb{R}$, and random variables $X_1, \ldots, X_n$ independently identically distributed according to D, then $\sum_{i=1}^{n} v_i X_i$ has the same distribution as $(X \sum_i |v_i|^p)^{1/p}$, where X is drawn from distribution D independently of $\{X_i : i = 1, 2, \ldots, n\}$. p-stable distributions have been used elsewhere in the algorithms literature, for example in Indyk (2000), where they are used them for sketching and metric embeddings. In Andoni et al. (2006), the idea is to generate a random vector $a \in \mathbb{R}^d$, with each component of a drawn from a p-stable distribution D. Then, given a vector $v \in \mathbb{R}^d$, the inner product $\langle a, v \rangle >$ is distributed as $X\|v\|_p$, where X is drawn from the same p-stable distribution. This allows us to use such an inner product to project a vector onto the real line, and then partition the real line into intervals such that vectors projected into the same interval are assigned the same hash value. Performing several such projections with different random vectors, we can project onto a subspace that has dimension of our choosing. Given vectors $u, v \in \mathbb{R}^d$, the distance between their projections is distributed as $\|u - v\|_p X'$, where X' is again drawn from D independently of the variables $\{X, X_1, \ldots, X_n\}$ mentioned above. Thus, vectors near each other are likely to be hashed to the same location.

C.3 Locality-Sensitive Hashing: More Recent Progress

In the ensuing years, a number of more specialized approaches to LSH have emerged. These approaches can be roughly divided into two categories. The first tries to find LSH families that work well for any data set. The second of these categories seeks to learn a set of functions that works well for a specific data set. That is, these approaches attempt to find a set of functions that give rise to a good hashing scheme for a specific search collection P, rather than finding a family of functions that have the locality-sensitivity that works with high probability for any possible collection of points. The early work on LSH, including the original LSH papers by Indyk and his colleagues focused on the former of these categories, but the vast majority of techniques published more recently have fallen in the latter of these categories, most of them applying unsupervised learning techniques to discover good hash functions for a given data distribution.

Work by Salakhutdinov and Hinton (2007) falls into the latter of these categories. They developed a technique that applies an autoencoder to estimate a set of hyperplanes that jointly give rise to an effective LSH code. In a similar spirit, Weiss et al. (2009) introduced a hashing scheme for arbitrary sets of objects under some distance function. They motivate their approach as opposed to that given in Salakhutdinov and Hinton (2007) by claiming that geometric hash codes learned by autoencoders tend not to make efficient use of code length. Their goal, then, was to find semantic hashes whose binary codes are as close to optimal as possible in an information theoretic sense. Thus, they required that the hash signature of a randomly chosen item in the ambient space has its bits independently distributed. Weiss, et al., posit this requirement as an optimization problem, which they show to be NP-hard by a reduction to a graph cut problem. Relaxing this problem, they require only that hash signature bits be uncorrelated. The result is a problem whose solutions are precisely the eigenfunctions of the graph Laplacian (refer to Appendix B for an overview of Laplacian eigenmaps). To hash an element, one simply applies the out-of-sample extension to the Laplacian eigenmaps embedding (Belkin et al. 2006), and round its components to $\{0, 1\}$ according to their signs. Weiss et al. (2009) presented experiments showing that their approach out-performed several other state-of-the-art hashing techniques on an image search task. A similarly-motivated problem was considered by Masci et al. (2014), where the goal was to construct hash functions that preserved pairwise similarity and dissimilarity labels of a set of points, rather than directly preserving distances. The authors proposed a feed-forward neural net to solve an objective function subject to ℓ_1-regularization to enforce sparsity.

Wang et al. (2010) took a different approach, attempting to learn a sequence of hash functions in such a way that hash functions learned later can "correct" those learned earlier in the sequence. Their approach, which requires supervision in the form of observation pairs labeled as to whether or not they are correct matches, is based on minimizing an objective function that penalizes hash codes that assign different labels to matching pairs or assign the same label to non-matching pairs. Rather than solve this objective directly, which would lead to overfitting, the authors include regularization that encourages high-entropy hash functions. This is similar to the approach taken by Weiss et al. (2009), where the obstacle was that maximizing the entropy resulted in an NP-hard optimization problem. To circumvent this issue, Wang et al. (2010) replace their maximum-entropy regularization term with one that encourages high variance in the bits of the hash functions.

Extending the ideas of Weiss et al. (2009), Liu et al. (2011) developed the idea of an *anchor graph*. Rather than attempting to compute all $O(n^2)$ pair-wise distances to construct the graph Laplacian, they select using k-means a set of $m \ll n$ *anchor points* from the collection of observations and use these to build an approximation to the true nearest neighbor graph. Their technique achieves strong performance against a number of other recent LSH techniques, though they gave no theoretical analysis of their approach.

Working along similar lines, Lin et al. (2013) presented *compressed hashing*, which uses techniques from compressed sensing (in particular, the restricted isometry property Candès (2008)), to learn an efficient coding of the sparse representations of a given data set. The authors also used anchor points as in Liu et al. (2011), but rather than using them to estimate the near-neighbor graph, they were used to estimate linear operators on a reproducing kernel Hilbert space. They use this approximation along with the observation from Donoho (2006b) that pairwise distances between a set of vectors in the ℓ_1 ball can be approximately preserved by storing only a small number of the largest-magnitude entries from each vector. Another approach, also motivated by ideas from compressed sensing, was explored by Cherian et al. (2014), who used sparse dictionary learning to build hash functions, with the goal of making the hash signatures robust to noise in the data.

Kulis and Grauman (2009) developed a technique for applying LSH to kernelized data when the feature map is not easily computed (see Appendix B for a discussion of the relevant ideas from kernel methods). Their core observation is that a random sample of the data will have feature representations that are approximately normally distributed about a population mean in the feature space. This observation leads to a formulation similar to the Nyström method (see Appendix B for discussion), and the hash functions become weighted sums of the form $h(x) = \sum_{i=1}^{t} w_i k(x, x_i)$.

Appendix D

Matrix Completion and Random Matrices

In many applications, the memory required to store large matrices can be a major obstacle to data processing and analysis. One solution to this problem is to replace a matrix $X \in \mathbb{R}^{n\times n}$ with some suitable approximation $\hat{X} \in \mathbb{R}^{n\times n}$, chosen so that $\hat{X}$ requires less storage space than X. For example, suppose $\hat{X}$ is positive semidefinite, say $\hat{X} = BB^T$ for some $B \in \mathbb{R}^{n\times k}$. If $k \ll n$, then $\hat{X}$ requires only $O(nk)$ memory rather than $O(n^2)$. Similarly, if $\hat{X}$ is sparse (i.e., most entries of $\hat{X}$ are zero), then $\hat{X}$ can be stored economically by only recording the entries for which $\hat{X}$ is nonzero. Depending on the specific problem at hand, analysis of $\hat{X}$ will yield approximately the same results as if the analysis had been applied to X, while requiring a fraction of the storage. The technique presented in Chapter 3 is precisely such an approximation result. In the last two decades, much research has been devoted to ideas of this sort, and we give here a brief summary of some of those lines of work.

D.1 Matrix Completion and Compressed Sensing

Computer science, statistics and engineering have been revolutionized by the ideas of compressed sensing and sparsity Donoho (2006b). The key to these advances has been the observation that many common signals, such as waveforms and images, can be succinctly (approximately) represented in a suitable basis. Crucially, a handful of pioneering papers (Donoho 2006a; Candès et al. 2006a,b) showed that the problem of expressing a signal in a given basis can, under suitable conditions, be solved efficiently using standard linear programming.

These ideas have given rise to a class of techniques for solving the problem of *low-rank matrix completion*, where we observe a small subset of entries of a matrix $M \in \mathbb{R}^{n_1 \times n_2}$ and wish to determine the values of the unobserved entries. Of course, for general matrices, this is an impossibility, but when M is low-rank, solutions exist and can be recovered using well-known optimization techniques (Candès and Recht 2009). More formally, in matrix completion we are presented with the entries M_{ij} of a matrix $M \in \mathbb{R}^{n_1 \times n_2}$ for some (possibly random) set of entries indexed by $\Omega \subseteq [n_1] \times [n_2]$. We say that the entries $M_{i,j}$ for $(i,j) \in \Omega$ are *observed*, and call all other entries *unobserved*. The goal is to solve the optimization problem

$$\begin{aligned} &\text{minimize} && \operatorname{rank} X \\ &\text{subject to} && X_{ij} = M_{ij} \quad (i,j) \in \Omega. \end{aligned} \tag{D.1}$$

That is, we wish to find a low-rank matrix X that agrees with the observed entries of M. In the remainder of this chapter we will assume for ease of exhibition and notation that $n_1 = n_2 = n$, i.e., that M is square, but we note that all results presented here can be extended to rectangular matrices (though a few of the stated orders of growth may depend on the ratio n_1/n_2 in the case of rectangular M). Candès and Recht (2009) showed that under certain conditions the problem in (D.1) can be solved exactly, despite the fact that it is NP-hard in general (Chistov and Grigoriev 1984), by solving the surrogate problem

$$\begin{aligned} &\text{minimize} && \|X\|_* \\ &\text{subject to} && X_{ij} = M_{ij} \quad (i,j) \in \Omega, \end{aligned} \tag{D.2}$$

where $\|\cdot\|_*$ denotes the nuclear norm

$$\|X\|_* = \sum_{k=1}^{r} \sigma_k(X),$$

where $r = \operatorname{rank} X$ and $\sigma_k(X)$ is the k-th largest singular value of matrix X. We can think of (D.2) as a relaxation of (D.1), in which we have replaced the rank-based objective with an objective that encourages low-rank solutions.

Candès and Recht (2009) considered a matrix $M \in \mathbb{R}^{n \times n}$ with $\operatorname{rank} M = r$ in which we observe m entries of M whose locations are chosen uniformly at random. The question of interest concerns how large m must be in order to ensure that with high probability, solving (D.2) yields a solution to (D.1). A trivial lower bound on m can be established by noting that completion of the matrix requires that we observe at least one entry from each row and each column, and thus by the coupon collector's problem (Mitzenmacher and Upfal 2005), $m \geq n \log n$ entries are required for this event to hold with high probability. Writing the singular value decomposition $M = U \Sigma V^T$, we see that M is in fact fully specified by $(2n - r)r$ numbers. When $r = \operatorname{rank} M$ is small compared to n, then, it is intuitively reasonable that we should be able to recover M based on far fewer than n^2 entries. As observed in Candès and Recht (2009), there exist low-rank matrices for which m must be close to n^2. For example, if M is a matrix with all entries equal to 0 save for a single entry $M_{1,1} = 1$, then M is certainly low-rank, but in order to recover M we must observe entry $(1, 1)$, which requires that m be large enough that $(1, 1) \in \Omega$ with high probability. Despite this, it was shown in Candès and Recht (2009) that a broad class of low-rank matrices are amenable to the above approach. In short, matrix completion requires both low-rankedness and a property called *incoherence*. Given a k-dimensional subspace $U \subseteq \mathbb{R}^n$ with orthonormal basis $\{u_1, u_2, \ldots, u_k\}$, the *coherence* of U with respect to the canonical basis $\{e_1, e_2, \ldots, e_n\}$ is defined as

$$\mu(U) = \frac{n}{k} \max_{i \in [n]} \|P_U e_i\|^2, \tag{D.3}$$

where P_U denotes the orthogonal projection onto U. Note that $1 \leq \mu(U) \leq n/k$. The main result of Candès and Recht (2009) shows that if μ_0 is an upper bound on $\mu(U)$ and $\mu(V)$, and $\mu_1\sqrt{r}/n$ is an upper bound on the maximum entry of the matrix $\sum_{k=1}^{r} u_k v_k^*$, then

$$m = \Omega(\max\{\mu_1^2, \mu_0^{1/2}\mu_1, \mu_0 n^{1/4}\} nr \log n)$$

entries suffice to recover M exactly with high probability. The proof due to Candès and Recht (2009) was the first of several to use the same general outline. In short, the goal is to show that D.2 has a unique solution with high probability. At the heart of this technique is the observation that the spectral norm is dual to the nuclear norm. Rather than working directly with the problem in Equation D.2, Candès and Recht (2009) proved that with high probability, a dual certificate for Equation D.2 exists. Candès and Recht (2009) constructed their proof under the Bernoulli model, in which $(i, j) \in \Omega$ with probability $p = m/n^2$ independently over all pairs $(i, j) \in [n] \times [n]$. We note that this is the model that we use in Chapter 3. An argument from Candès et al. (2006b) shows that the probability of failure under the Bernoulli model is at most twice that of the probability of failure under the $|\Omega| = m$ "uniform" model, so that bounds for the Bernoulli model are sufficient for most purposes.

Subsequent work has improved the results due to Candès and Recht (2009), typically by relaxing both the requisite lower bounds on the number of samples m and the coherence constraints on matrix M required to ensure high probability of success.

Keshavan et al. (2010a) developed a spectral algorithm, based on a singular value decomposition of the partially observed matrix, which recovers M to within a root mean square error (RMSE)

$$\frac{\|\hat{M} - M\|_F}{n\sqrt{r}} = O(\sqrt{nr/m}). \tag{D.4}$$

Additionally, they showed that under incoherence assumptions similar to those made in Candès and Recht (2009) and assumptions bounding the singular values of M away from 0 and ∞, recovery of M is exact. The algorithm presented by Keshavan et al. (2010a) is distinct from the SDP-based algorithms considered by, for example, Candès and Recht (2009) and Candès and Tao (2010), and is broadly similar to a singular value thresholding approach later pursued by Chatterjee (2015), which we discuss below. We note that a subsequent paper by the same authors extended their analysis to the case where the matrix M is also corrupted by noise (Keshavan et al. 2010b).

Candès and Tao (2010) showed that m can be brought closer to the coupon collector $\Omega(n \log n)$ lower bound provided we accept stronger incoherence constraints on M, than the incoherence property introduced by Candès and Recht (2009). Under these stronger incoherence properties, Candès and Tao (2010) showed that (Theorem 1.2) $m = \Omega(\mu^2 nr \log^6 n)$ samples suffice for the nuclear norm minimization in Equation (D.2) to recover M exactly with high probability. Their proof uses combinatorial techniques that are broadly similar to those that appear in the moment method proof of Wigner's semi-circle law, a classic result in random matrix theory (Wigner 1958; Füredi and Komlós 1981; Bai and Silverstein 2010), though the argument given by Candès and Tao (2010) requires a far more complicated combinatorial argument, owing to the occlusion of M.

Gross (2011) further improved these results, showing that recovery of M succeeds with high probability provided $m = \Omega(nr\nu \log^2 n)$, where ν is a coherence parameter roughly comparable to μ_0 in Recht (2011), discussed below. The sampling model considered by Gross (2011) differs from many related papers in that he considered a model in which the entries of Ω are chosen independently with replacement. Among the proof techniques used by Gross (2011) is the celebrated "golfing scheme", in which a dual certificate Y is expressed as a sum of random matrices, generated according to a process so that matrices later in the sequence can (partially) correct errors caused by earlier elements in the sequence. This technique has emerged as a standard tool for constructing dual certificates in matrix completion problems (see, for example Candès et al. 2011; Chen et al. 2013).

Recht (2011) established a similar result to that in Gross (2011), showing that $m = \Omega(\max\{\mu_0, \mu_1^2\} rn \log^2 2n)$ entries suffice to recover M with high probability.

Here μ_0 is an upper bound on the coherences of the row and column spaces of M and μ_1 is an upper bound on the entries of the matrix UV^*. Recht's analysis follows the general form of that in Gross (2011), using slightly different operator norm bounds. Recht speculated that it might be possible to remove dependence on μ_1 from the lower bound on m. More recent results have shown, in keeping with experimental observations, that indeed this incoherence bound on the maximum entry of UV^*, required in one form or another by all of the initial results on low-rank matrix completion, is superfluous. Chen (2015) showed that only incoherence of the row and column spaces with respect to the standard basis is necessary for exact low-rank completion and that only $m = \Omega(\mu n r \log^2 n)$ entries are necessary. Chen's analysis largely follows the reasoning in Candès and Recht (2009) and Candès and Tao (2010) but departs from these prior results by bounding the $\infty, 2$-norm $\|UV^T\|_{\infty,2}$ rather than $\|UV^T\|_\infty$, thus escaping the need for a bound on the coherence between the row and column spaces of $M = U\Sigma V^T$.

D.1.1 Matrix Completion with Noise

In many applications, including the one considered in Chapter 3, the matrix M is not only occluded, but is also noisy. That is, we assume that rather than observing entries $M_{i,j}$ for $(i,j) \in \Omega$, we observe $Y_{i,j} = M_{i,j} + Z_{i,j}$ where $\{Z_{i,j} : (i,j) \in \Omega\}$ are a collection of error terms, typically assumed to be independent.

Candès and Plan (2009) considered this problem, starting from the ideas first presented by Candès and Tao (2010). Letting $Z \in \mathbb{R}^{n \times n}$ denote a matrix of noise terms, and letting $Y = M + Z$ denote the noisy version of the matrix M, Candès and Plan (2009) adapt the problem in (D.2) to write

$$\begin{aligned} &\text{minimize} && \|X\|_* \\ &\text{subject to} && \|\mathcal{P}_\Omega(X - Y)\|_F \leq \delta \end{aligned} \tag{D.5}$$

where $\|\mathcal{P}_\Omega(Z)\| \leq \delta$. Candès and Plan (2009) viewed Equation (D.5) as a semidefinite program (SDP) and showed that when a dual certificate for (D.5) exists and obeys a certain semidefinite inequality, then, letting $m = |\Omega|$, the solution $\hat{M}$ obeys

$$\|\hat{M} - M\|_F = O\left(\delta\sqrt{n^3/m}\right). \tag{D.6}$$

Crucially, the results proved by Candès and Tao (2010) imply that such a dual certificate exists with high probability, and hence (D.6) holds with high probability. In essence, the result shows that recovery of matrix M in the presence of noise is possible under precisely the same conditions as in the clean condition presented in Candès and Tao (2010).

Chen et al. (2013) considered the case where the matrix M is occluded as in the cases above, but also has at most a constant (in n) fraction of its entries corrupted by arbitrary noise. That is, we observe $\mathcal{P}_\Omega(Y)$ where $Y = M + Z$ for Z a matrix of errors, with at least a constant fraction of its entries equal to 0. Their results, which match those of Chen (2015) up to a polylogarithmic factor, were proved via an approach similar to the golfing scheme described above.

Chatterjee (2015) considered the matrix completion problem in the presence of noise, under the Bernoulli model with observation probability p, taking an approach that is broadly similar to, but simpler than, the algorithm presented by Keshavan et al. (2010a). Chatterjee (2015) considered a matrix $M \in \mathbb{R}^{n \times n}$ corrupted by unbiased noise with entries bounded in absolute value almost surely. By suitable recentering and rescaling of entries, the entries of $Y = M + Z$ can be assumed to lie in $[-1, 1]$. From the sampled matrix $\mathcal{P}_\Omega(Y)$, Chatterjee's universal singular value thresholding (USVT) estimator constructs an estimate $\hat{M}$ by first estimating the Bernoulli parameter p as $\hat{p} = |\Omega|/n^2$, then discarding all singular values of $\mathcal{P}_\Omega(Y)$ that are below the threshold $(2 + \eta)\sqrt{n\hat{p}}$, where $\eta \in (0, 1)$ is a parameter required by the concentration inequalities applied in the proofs. This step of throwing out the small singular values is on the one hand quite similar to the algorithm due to Keshavan et al. (2010a) in its aim, while also being in some sense at odds with it. Keshavan et al. (2010a) discard certain rows of their sampled matrix to decrease the large singular values (and then later rescale and clean the matrix), while Chatterjee (2015) achieves a similar "smoothing" effect by discarding the small singular values. The main result is that the estimate $\hat{M}$ obtained via this method has, provided $p > n^{-1+\epsilon}$ for some $\epsilon > 0$,

$$\frac{\mathbb{E}\|\hat{M} - M\|_F^2}{n^2} = O\left(\min\left\{1, \frac{\|M\|_*}{pn^{3/2}}, \frac{\|M\|_*^2}{n^2}\right\}\right) + C_\epsilon \exp\{-O(np)\},$$

where C_ϵ depends on ϵ and η. The proof of this result made heavy use of several spectral norm bounds, along with classic concentration inequalities (Bhatia 1997; Talagrand 1996; Boucheron et al. 2013). The core of the proof consists of (i) a spectral norm bound for random matrices based on the moment method (Bai and Silverstein 2010) and an application of a concentration inequality due to Talagrand (1996) and (ii) a bound on the Frobenius-norm error in estimating a matrix B using singular value thresholding of $A - B$ for some other matrix A.

D.2 Matrix Perturbation

Many of the results summarized in the previous section are, at their core, results about the behavior of matrices under perturbation. We are concerned with the behavior of the matrix $M + E$ where $M \in \mathbb{R}^{n \times n}$ is a matrix, and $E \in \mathbb{R}^{n \times n}$ is a (possibly random) *perturbation* of M. This area of mathematics is well-studied, and a thorough literature review is beyond the scope of this thesis. We instead collect here only a few key results, mostly focused on applications to sums of random matrices. For a more thorough treatment, we refer the reader to the classic textbooks Stewart and Sun (1990); Kato (1995); Bhatia (1997) and to the recent manuscript Tropp (2015).

As an example, consider how the spectrum of M relates to that of $M + E$. A classic theorem due to Weyl (Bhatia 1997, Theorem III.2.1) relates the spectrum of a Hermitian matrix H to the spectrum of a perturbed version of H given by $H + E$.

Theorem 9. *Let $A, B \in \mathbb{R}^{n \times n}$ be Hermitian matrices with eigenvalues $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_n(A)$ and $\lambda_1(B) \geq \lambda_2(B) \geq \cdots \geq \lambda_n(B)$. Then*

$$\lambda_j(A + B) \leq \lambda_i(A) + \lambda_{j-i+1}(B) \text{ for } i \leq j$$

and

$$\lambda_j(A + B) \geq \lambda_i(A) + \lambda_{j-i+1}(B) \text{ for } i \geq j.$$

Theorems of the sort just quoted allow us to examine how the spectrum of $M + E$ relates to those of M and E. For example, they let us bound the amount by which perturbation E can change the eigenvalues of M as $|\lambda_i(M + E) - \lambda_i(M)| \leq \|E\|$.

While matrix perturbation bounds similar to those above are plentiful in the case of eigenvalues and singular values (Horn and Johnson 1994; Bhatia 1997, see, e.g.,), analogous bounds for eigenvectors and singular vectors are harder to come by. The most prominent such result, the *Davis-Kahan* $\sin \Theta$ *theorem* (Davis and Kahan 1970), remains as the standard approach to such bounds. We paraphrase here the version stated in Bhatia (1997), which we contrast with the version of the theorem used to prove Theorem 1 in Chapter 3.

Theorem 10. *Let $A, B \in \mathbb{R}^{n \times n}$ be Hermitian and let $S_1, S_2 \subset \mathbb{R}$ such that $d(S_1, S_2) =$*

$\delta > 0$. *Let* $Q_1 = P_A(S_1)$ *be the orthogonal projection onto the span of the eigenvectors of* A *with corresponding eigenvalues in* S_1*, and define* $Q_2 = P_B(S_2)$ *analogously. Then for any unitarily invariant norm* $\|\cdot\|$*, we have*

$$\|Q_1 Q_2\| \leq \frac{\pi}{2\delta} \|Q_1(A - B)Q_2\| \leq \frac{\pi}{2\delta} \|Q_1 Q_2\|.$$

The reference to $\sin \Theta$ in the name of the theorem derives from the observation that the quantity $\|Q_1 Q_2\|$ is precisely the sine of the (largest) canonical angle between the range of Q_1 and the orthogonal complement of the range of Q_2 (Bhatia 1997). While general results improving upon the Davis-Kahan theorem have been few and far between, we note that there do exist a number of results characterizing the eigenvectors of random matrices. See, in particular, the recent survey by O'Rourke et al. (2016a).

Bounds of this sort have been extended to the case of sums of (typically independent) random matrices in the form of *matrix concentration inequalities.* Just as sums of real-valued random variables exhibit the concentration of measure phenomenon (Boucheron et al. 2013), sums of random matrices concentrate about their mean under certain conditions. Based on results first developed in statistical mechanics (Golden 1965; Thompson 1965) and quantum information theory (Ahlswede and Winter 2002), these results have fast become standard tools in mathematics, physics, engineering and statistics (see the survey Tropp 2015, and citations therein).

Of particular interest for the purposes of this thesis are applications to spectral graph theory (Chung et al. 2003; Feige and Ofek 2005; Oliviera 2010; Lu and Peng 2013), where these tools have enabled the development and investigation of *spectral methods* in statistics and machine learning, in which the eigenvalues and eigenvectors of graphs or matrices are of prime interest. Such methods include Laplacian eigenmaps (Belkin and Niyogi 2003), adjacency spectral embedding (Sussman et al. 2012) and spectral clustering (von Luxburg 2007), and of course the sparse, noisy Laplacian eigenmaps embeddings discussed in Chapter 3 fall under this heading. Refer to Appendix B for a discussion of these methods.

Just as the Laplace transform yields a plethora of concentration inequalities for sums of random variables (Boucheron et al. 2013), a similar idea has been central to developing analogous bounds for sums of random matrices. The following theorem, originally due to Lieb (1973), and which we quote from Tropp (2015), lies at the heart of these matrix concentration inequalities:

Theorem 11. *Let $H \in \mathbb{R}^{n\times n}$ be a Hermitian matrix. Then the function $f(A) = \operatorname{tr}\exp\{H + \log A\}$ is concave on the cone of positive-definite matrices.*

As an example of a typical matrix concentration result, also from Tropp (2015), consider the following "matrix Bernstein" theorem, so-called owing to its similarity to an inequality originally due to S. Bernstein for sums of independent real-valued random variables (Chung and Lu 2006; Boucheron et al. 2013).

Theorem 12. *Let $X_1, X_2, \ldots, X_n$ be a sequence of n independent random d-by-d matrices with $\mathbb{E}X_k = 0$ and $\|X_k\| \leq L$ for all $k \in [n]$. Define the matrix $Z = \sum_{k=1}^n X_k$, and define*

$$v(Z) = \max\left\{\|\mathbb{E}ZZ^*\|, \|\mathbb{E}Z^*Z\right\}.$$

For all $t \geq 0$, we have

$$\Pr\left[\|Z\| \geq t\right] \leq 2d \exp\left\{\frac{-t^2/2}{v(Z) + Lt/3}\right\}.$$

As an illustrative example of these concentration results in action, consider a random undirected simple graph $G = (V, E)$ on n nodes with adjacency matrix $A \in \mathbb{R}^{n\times n}$, so that $A_{ij} = 1$ if and only if an edge joins vertices i and j in G, and $A_{ii} = 0$ for all $i \in [n]$. Suppose that the indicator random variables $\{A_{ij} : 1 \leq i < j \leq n\}$ are independent Bernoulli random variables with probability of success P_{ij}. That is, the edges $\{i, j\}$ are present or absent in G independently, with $\Pr[\{i, j\} \in E] = P_{ij}$, and we have $\mathbb{E}A = P$. An application of the matrix Bernstein theorem given above shows that A concentrates about its expected value in spectral norm, a result that appeared first in Oliviera (2010) and also appears in slightly more general form in Tropp (2012).

Theorem 13. *Let A be the adjacency matrix of the random graph G just described. Define the maximum expected degree $\Delta = \max_i \sum_{j=1}^n P_{ij}$, and suppose that there exists* *a constant $c > 0$ such that*

$$\frac{3\Delta}{6\Delta + 2\sqrt{\Delta \log n}} \geq c.$$

Then with probability at least $1 - 2n^{1-c}$,

$$\|A - P\| \leq \sqrt{\Delta \log n}. \tag{D.7}$$

That is, the spectrum of the adjacency matrix concentrates within $\sqrt{\Delta \log n}$ of the spectrum of the matrix P.

Proof. Observe that $\mathbb{E}(A - P) = 0$, and letting $F_{ij} = e_i e_j^T + e_j e_i^T$, we can express $A - P$ as a sum

$$A - P = \sum_{1 \leq i < j \leq n} (A - P)_{ij} F_{ij},$$

in which every summand is bounded in spectral norm by 1. Noting that

$$\|\mathbb{E}(A - P)^2\| \leq \max_{i \in [n]} \sum_{j=1}^{n} \mathbb{E}(A - P)_{ij}^2 \leq \Delta,$$

an application of Theorem 12 with $t = \sqrt{\Delta \log n}$ and the assumption in (D.7) yields the result. □

Many results require a bound on the average degree like the one just seen in order to ensure concentration Rohe et al. (2011a); Oliviera (2010). In the case of the graph Laplacian, the high variance associated with small average degree precludes concentration for general weighted graphs Chung et al. (2003); Le et al. (2016); Klopp et al. (2015). As part of their examination of spectral clustering in sparse graphs, Amini et al. (2013) noted that adding a small number of low-weight edges, which reconnects the disconnected components that tend to occur in sparse random graphs, results in better performance of spectral clustering. Joseph and Yu (2014) examined the approach of Amini et al. (2013) theoretically in the case of the stochastic block model, and showed that the eigenvalues and eigenvectors of the normalized graph Laplacian concentrate once we regularize the adjacency matrix A as $A_r = A + rJ$ for small regularization parameter $r > 0$. Chaudhuri et al. (2012) considered similar ideas, working in a related random graph model, where they showed that the normalized graph Laplacian concentrates after *degree-correction*, in which the adjacency matrix A is replaced by $(D + \tau I)^{-1}$, where D is the diagonal degree matrix. Qin and Rohe (2013) considered a spectral clustering algorithm inspired by the approaches of Chaudhuri et al. (2012) and Amini et al. (2013). They considered regularization in which the degree matrix D is replaced by $D_\tau = D + \tau I$, and showed that under the degree-corrected SBM, the regularized normalized graph Laplacian $D_\tau^{-1/2} A D_\tau^{-1/2}$ concentrates in spectral norm, strengthening the analogous result presented in Chaudhuri et al. (2012).

Le et al. (2016) considered concentration of both the adjacency matrix and the graph Laplacian. In the case of the graph Laplacian, where small-degree vertices prevent concentration, the authors showed that regularizing the adjacency matrix as $A_\tau = A + (\tau/n)J$, for $\tau \in \mathbb{R}$ chosen approximately equal to the average degree, ensures that with high probability

$$\|\mathscr{L}(A_\tau) - \mathscr{L}(\mathbb{E}A_\tau)\| = O(d^{-1/2}),$$

where $d = \max_{1 \le i < j \le n} n\mathbb{E}A_{ij}$. The proof of this fact follows from an analogous concentration result for the regularized adjacency matrix A_τ about its mean. This concentration of the adjacency matrix is proved by decomposing the graph into a set of high-degree vertices, for which the degrees concentrate about their means, and a set of small-degree vertices, which are shown to contribute little to the spectral norm.

Appendix E

Vertex Nomination and Graph Matching

In Chapter 4, the graph matching problem is important primarily because it makes possible a maximum-likelihood-based solution to the vertex nomination problem. As network data has become ever more prominent in the sciences, problems of this sort have become increasingly central to data analysis. In this appendix, we review the recent literature on the problems of graph matching and vertex nomination. We refer the reader to the excellent survey by Conte et al. (2004) for an overview of earlier approaches, and focus here primarily on the results of the past fifteen years.

E.1 Graph Matching

The graph matching (GM) problem is to find a correspondence between the vertices of two graphs that best preserves graph topology.

Definition 9. *Given graphs G_1 and G_2 on common vertex set $V = [n]$ with respective adjacency matrices $A, B \in \mathbb{R}^{n \times n}$, the* graph matching *problem is to find a permutation matrix $P^* \in S_n$ solving*

$$\min_{P \in S_n} \|A - PBP^T\|_F, \tag{E.1}$$

where S_n is the set of n-by-n permutation matrices.

In what follows, we assume that all graphs involved are undirected, so that A and B are symmetric. This assumption is made only for ease of presentation, as most of the techniques discussed are easily extended to the directed case. We note the similarity of the minimization in (E.1) to the well-known graph isomorphism problem (Garey and Johnson 1979), in which the goal is to determine whether graphs G_1 and G_2 are isomorphic. Indeed, the graph matching problem and many related problems are known to be NP-hard (Conte et al. 2004). Many early approaches attempted to treat GM as a search problem, with various search and pruning heuristics suggested in the literature (see citations in van Wyk et al. 2002). Most approaches in the last 25 years have instead considered approximate solutions to (E.1) rather than attempting to find an exact optimal solution. These approximate approaches can be broadly divided into two categorizes. The first attempts to map the vertices of graphs G_1 and

G_2 into some common (typically Euclidean) space and then finds a correspondence between the points using, for example, Procrustes alignment (Luo and Hancock 2000; Gower and Dijksterhuis 2004). The second category of approach applies optimization techniques, typically by relaxing the minimization problem in (E.1).

The first category of approach had its start in the work of Umeyama (1988). At the heart of Umeyama's approach is the observation that the minimization

$$\min_{Q^T Q = I} \|A - QBQ^T\|_F \tag{E.2}$$

is minimized by choosing any $Q = U_B S U_A^T$, where A and B have singular value decompositions $A = U_A \Lambda_A U_A^T$ and $B = U_B \Lambda_B U_B^T$ and S is a diagonal matrix with entries $S_{ii} \in \{-1, 1\}$. Umeyama's method is, in effect, applying a Procrustes alignment of the latent vertex positions of graphs G_1 and G_2 in the case where vertex positions are given by rows of the matrices U_A and U_B. Umeyama motivated this approach in the case where G_1 and G_2 are *nearly* isomorphic by conjecturing that in such cases this procedure will find a good initial solution, which can be refined by hill climbing or other local search methods.

Embedding-based approaches in the spirit of Umeyama (1988) have been especially prominent in image processing, where the geometrical interpretation of image features or mesh points is particularly natural (see, for example, Scott and Longuet-Higgins 1991; Shapiro and Brady 1992; Sclaroff and Pentland 1995; Cross and Hancock

1998; Wang and Hancock 2006; Knossow et al. 2009, and citations therein). Scott and Longuet-Higgins (1991) considered the problem of discovering a correspondence between the features in two images by vectorizing the images and maximizing a trace $\operatorname{tr} Q^T K$, where K is a (possibly rectangular) matrix whose entries capture affinities between nodes in G_1 and G_2, and Q is an orthogonal matrix (with orthogonal rows, in the event that K is rectangular). A singular value decomposition similar to that in Umeyama (1988) yields a solution to the problem.

Observing that the approach in Scott and Longuet-Higgins (1991) did little to respect structure within the images being aligned, Shapiro and Brady (1992) proposed an alignment method that first applies a spectral embedding of the image features according to the images' geometry and then aligns those embeddings. Cross and Hancock (1998) presented a method for aligning pairs of point-sets based on expectation-maximization (EM; Dempster et al. 1977), in which alignment of points and (possibly nonlinear) transformation of those points occur probabilistically. This approach allows uncertainty in the image features to be incorporated into the probabilistic model, making the algorithm comparatively robust to noise and outliers. Carcassoni and Hancock (2003) combined this EM-based framework with spectral methods.

Keselman et al. (2003) extended the graph matching problem to the case where vertex correspondence need not be one-to-one, instead allowing a many-to-many correspondence between the vertices of graphs G_1 and G_2. Their method used techniques from metric embeddings (Indyk 2001; Linial 2002) to map the vertices of G_1 and G_2 into normed spaces B_1 and B_2, respectively, then aligned the spaces B_1 and B_2 with the goal of minimizing the earth mover's distance between the graphs' corresponding point clouds (the earth mover's distance is a metric on probability distributions; see, for example, Rubner et al. 1998). Leordeanu and Hebert (2005) presented a method for one-to-many graph matching that applies spectral clustering to the matrix of feature affinities K to find ensembles of feature correspondences that are jointly preferable. Cour et al. (2007) modified the spectral method of Leordeanu and Hebert (2005) to include an affine constraint that prevents a solution from assigning any vertex in G_1 to too many vertices in G_2.

Knossow et al. (2009) considered aligning graphs with possibly different numbers of vertices. Their approach computes a Laplacian eigenmaps embedding (Belkin and Niyogi 2003) of each of the graphs, then seeks to align those embeddings. Viewing the eigenvectors of the two graphs as mappings of the vertices onto the real line, histograms are built of these two projections, and the histograms of the two graphs are aligned using the Hungarian method (Kuhn 1955). Viewed differently, we can think of the method of Knossow et al. (2009) as aligning spectral clusterings of G_1 and G_2. Broadly similar approaches were considered earlier by Caelli and Kosinov (2004) and Robles-Kelly and Hancock (2007). Laplacian eigenmaps and diffusion-based embeddings (Belkin and Niyogi 2003; Coifman and Lafon 2006) were also used by Xiao et al. (2009) to determine vertex correspondences across multiple graphs at once. Escolano et al. (2011) proposed mapping vertices to Euclidean space via a commute-time embedding (Qiu and Hancock 2007; Trosset and Tang 2010) and aligning the resulting point clouds according to an information theoretic objective (see Escolano et al. 2013, for further discussion of this and related information-theoretic graph matching methods).

A different approach to graph matching for image processing was considered by Zhou and la Torre (2016), working in the context of directed graphs in which both edges and nodes are endowed with features. Their approach works by factorizing two matrices K_1 and K_2 that capture affinities among the nodes and edges, respectively. The authors solved an optimization problem similar to Equation (E.4) below, using a path-following algorithm similar to that in Zaslavskiy et al. (2009a).

Lyzinski et al. (2014a) considered an extension of the graph matching problem in which one knows, a priori, a few vertex correspondences between graphs G_1 and G_2, and the goal is to find a matching that preserves the correspondence between these *seed* vertices while also leveraging the information that they provide about the topology of the two graphs. This *seeded graph matching* (SGM) problem is central to the vertex nomination techniques considered in Chapters 4 and 5. Lyzinski et al. (2014a) considered the case of matching graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ when the pair (G_1, G_2) is drawn from the ρ-correlated Erdős-Rényimodel, which we define here.

Definition 10. *Erdős-Rényirandom graph (Erdős and Rényi 1959) A random graph*

$G = (V, E)$ is said to be an Erdős-Rényirandom graph if there is some $p \in [0, 1]$ so that the possible edges of G are present or absent, independently, with probability p.

Definition 11. *ρ-correlated Erdős-Rényirandom graphs We say that the random graphs G_1 and G_2 are ρ-correlated Erdős-Rényirandom graphs if G_1 and G_2 are each marginally distributed as Erdős-Rényirandom graphs and if, letting A and B denote the adjacency matrices of G_1 and G_2 respectively, the Bernoulli random variables*

$$\{A_{ij} : 1 \leq i < j \leq n = |V|\} \cup \{B_{ij} : 1 \leq i < j \leq n\}$$

are independent except that the pairs (A_{ij}, B_{ij}) have correlation ρ for all $1 \leq i < j \leq |V| = n$.

Lyzinski et al. (2014a) considered how the values of ρ and p influence the feasibility of graph matching and how the number of seeds s influences the quality of seeded graph matching. They showed that under suitable growth conditions on p and ρ, graph matching is still feasible, even for comparatively small values of ρ, in the sense that the minimizer of (E.1) is unaffected by the uncertainty introduced by the imperfect correlation. They proved a similar result for the case of seeded graph matching, which shows that the presence of seed vertices makes over a wider range of values for p and ρ. Their methods are largely similar to those used in Chapter 4, in that they use standard concentration inequalities to bound the probability that too many edges are flipped in a way that drastically alters the minimizer of (E.1).

Lyzinski et al. (2014a) explored the in-practice efficacy of seeded graph matching by applying Frank-Wolfe to a modification of the relaxation in (E.4) that takes seed information into account.

Lyzinski et al. (2015) considered a spectral embedding approach to the seeded graph matching problem. Their approach makes use of the adjacency spectral embedding (ASE; see Sussman et al. 2012, or refer to Appendix B for a brief overview), which (asymptotically) perfectly recovers latent cluster structure in the graphs G_1 and G_2. Their method first embeds the vertices of graphs G_1 and G_2 into a common space $\mathbb{R}^d$ using ASE followed by Procrustes analysis. k-means clustering is then applied to the $2n$ embeddings of the two graphs to obtain k clusters. Matching is then performed *within* clusters, across the two graphs. That is, for each of the k clusters, the subgraphs of G_1 and G_2 induced by the vertices in that cluster are matched. This method is motivated by the fact that, owing to the consistency properties of ASE (Sussman et al. 2012; Lyzinski et al. 2014b), the joint clustering of the embeddings will, in the large-n limit, perfectly recover the k clusters of G_1 and G_2 when such structure exists (i.e., when G_1 and G_2 are both k-block SBMs). Working under the stochastic block model, the main theoretical result showed, using techniques broadly similar to those used to prove the results in Chapter 4, that this spectral seeded graph matching approach correctly recovers the vertex correspondence up to a permutation within the blocks.

An early example (though hardly the first; see, e.g., Davis 1979; Li 1992; Almohamad and Duffuaa 1993) of the optimization-based approach to graph-matching can be found in Gold and Rangarajan (1996). The authors considered a relaxation of the objective function in (E.1) given by

$$\min_{M \in N_n} \|A - MBM^T\|_F, \tag{E.3}$$

where $N_n = \{M : \forall i, j : M_{ij} \geq 0\}$. Letting Q_{ij} denote the gradient of the cost function in (E.3) with respect to M_{ij}, optimization proceeds by repeatedly updating $\hat{M}_{ij}^{(t+1)} = \exp\{-\beta Q_{ij}\}$, and applying Sinkhorn's method (Sinkhorn 1964) to transform $\hat{M}$ into a doubly stochastic matrix. Here $\beta > 0$ is a parameter that increases with the number of steps. Gold and Rangarajan (1996) also convexify the objective in (E.3) by adding an entropy term of the form $\frac{1}{\beta}\sum_{i,j} M_{ij} \log M_{ij}$. Thus, as β increases, the problem becomes less convex while the $\hat{M}_{ij}$ terms are closer to binary.

The PATH algorithm presented in Zaslavskiy et al. (2009a) follows a classic approach in optimization– one finds a solution to a convex relaxation of a given problem, and then projects that solution back to an integral solution (Raghavan and Thompson 1987). Letting $\mathscr{D}_n$ denote the set of all n-by-n doubly stochastic matrices (i.e., the Birkhoff polytope Bhatia 1997), the algorithm proceeds by alternatingly solving a convex relaxation of (E.1), given by

$$\min_{D \in \mathscr{D}_n} \|A - DBD^T\|_F, \tag{E.4}$$

and the concave problem

$$\min_{P \in \mathscr{D}_n} - \operatorname{tr} \Delta P - 2\vec{(P)}^T (L_1^T \otimes L_2^T) \vec{(P)}, \tag{E.5}$$

where $\Delta \in \mathbb{R}^{n \times n}$ is the matrix with entries $\Delta_{ij} = (\sum_{k=1}^n A_{ik} - B_{jk})^2$, and L_1, L_2 are the combinatorial Laplacians of graphs G_1 and G_2, respectively. That is, $L_1 = D_1 - A$, where D_1 is the degree matrix of G_1. Here $\otimes$ denotes the Kronecker product and $\vec{(P)} \in \mathbb{R}^{n^2}$ denotes the vectorization of matrix $P \in \mathbb{R}^{n \times n}$ (Horn and Johnson 1994, Chapter 4). The problem in (E.4) can be solved efficiently using the celebrated Frank-Wolfe algorithm Frank and Wolfe (1956), but (E.5) has no such efficient algorithm. Zaslavskiy et al. (2009a) eased this drawback by considering linear interpolations between the objectives in (E.4) and (E.5). By slowly varying the coefficient of this interpolation and repeatedly finding local solutions, a path of solutions are found, leading to a locally optimal solution to (E.5). The PATH algorithm achieved state-of-the-art performance on the QAPLIB benchmarks (Burkard et al. 1997), a collection of quadratic assignment problem (QAP; Burkard et al. 2009) instances. Later work by Liu et al. (2012) explored several variations on the concave relaxation (E.5) and demonstrated similarly strong performance on the same benchmarks.

Zaslavskiy et al. (2010) considered the case of many-to-many graph matching. In the case of geometric approaches to graph matching, this extension is natural via, for example, the binning approach taken in Knossow et al. (2009). In the case of the more general optimization approaches, such as the PATH algorithm (Zaslavskiy et al. 2009a), it is less clear how to extend to the many-to-many case. Letting n_1 be the number of vertices in graph G_1, n_2 be the number of vertices in graph G_2, and $n = \min\{n_1, n_2\}$, Zaslavskiy et al. (2010) suggest the optimization

$$\min_{P_1 \in \Pi_{n_1,n}, P_2 \in \Pi_{n_2,n}} \|P_1 A P_1^T - P_2 B P_2^T\|_F^2, \tag{E.6}$$

where

$$\Pi_{n,k} = \{P \in \{0,1\}^{n \times k}, P\vec{\mathbf{1}} = \vec{\mathbf{1}}, \vec{\mathbf{1}}^T P \leq k_{\max}, \vec{\mathbf{1}}\},$$

and $k_{\max}$ is a chosen upper bound on how many vertices in one graph may be assigned to the same vertex in the other. Zaslavskiy et al. (2010) presented two approaches to solving the optimization problem in (E.6). The first, based on a convex relaxation of the sets $\Pi_{n_1,n}$ and $\Pi_{n_2,n}$, follows an approach largely similar to the approach of Zaslavskiy et al. (2009a) to the convex relaxation in (E.4). The second approach was based on a semidefinite relaxation of (E.6). Earlier examples of SDP-based approaches to one-to-many and many-to-many graph matching can be found in Torr (2003), in which the author adapted the randomized rounding approach of Goemans and Williamson (1995), and Schellewald and Schnörr (2005), in which the authors applied semidefinite programming techniques to the the natural semidefinite relaxation of an objective similar to that in (E.1).

Cho and Lee (2012), working in the setting of one-to-one graph matching, considered a different approach, based on building up a vertex correspondence incrementally. Their approach is based on estimating a distribution over candidate vertex correspondences, conditioned on a current matching between two *active* graphs, which are subgraphs of graphs G_1 and G_2. At each time step of the algorithm, this distribution functions similarly to a sum of experts (Kittler and Alkoot 2003) to decide how to update the current matching.

Singh et al. (2008), working in the context of biological graph matching applications such as protein-protein interaction networks, considered the task of attributed graph alignment, in which the nodes and edges are endowed with features which we wish to preserve in our alignment. Their method is based on an optimization that can be recast as an eigenproblem, solved via the power method, which is feasible since the networks involved are typically large but sparse. Rather than a more typical rounding solution, the authors use the result of this optimization to obtain a set of scores $\{R_{ij}\}$, where i and j range over the vertices of G_1 and G_2, respectively. Singh et al. (2008) used these ranking scores to align vertices according to a heuristic. Working on a similar biologically-motivated attributed alignment problem, Zhang and Tong (2016) sought an alignment preserving graph topology, node-level attribute consistency, and edge-level attribute consistency all at once. These concerns led an objection function in which the node- and edge-level feature similarities are taken into account, similarly to the vertex nomination generalizations discussed in Section 4.5. Properties of the Kronecker product (Horn and Johnson 1994) allow a simple gradient descent approach, which Zhang and Tong (2016) prove converges to the global optimizer. We note that a number of papers have considered attributed graph matching problems of this sort, primarily with applications in biology. Many of these techniques are based on heuristics or are tailored to the domain of application, and are thus outside the scope of this survey. We refer the interested reader to Singh et al. (2008); Klau (2009); Zhang and Tong (2016) and citations therein.

An approach to many-to-one based on message-passing was considered by Bayati et al. (2013), in which the authors presented two approximation schemes for solving an indefinite QAP similar to (E.7). The local information in the nodes and edges of the graphs G_1 and G_2 is encoded in a factor graph (Koller and Friedman 2009), with variable nodes that capture whether or not edge-matching constraints are met and function nodes that fire if and only if the integer constraints of the original (integral) QAP are satisfied. Standard belief propagation techniques (Koller and Friedman 2009) are used to maximize the score in the factor graph, with a final bipartite matching problem used to round the solution to a proper assignment function.

Rather than the relaxation (E.4), Vogelstein et al. (2015) considered the relaxation

$$\min_{D \in \mathscr{D}_n} - \operatorname{tr} D^T A D B. \tag{E.7}$$

This relaxation has the disadvantage that its objective is indefinite (if G_1 and G_2 are hollow graphs, the Hessian of $\operatorname{tr} D^T A B D$ with respect to D has trace 0), and hence unlike in the case of (E.4), we cannot expect to find a global minimizer in $\mathscr{D}_n$ using the standard convex optimization tools. Nonetheless, Vogelstein et al. (2015) proposed the fast approximate QAP (FAQ) algorithm to approximately solve (E.1) by finding a local minimizer $D \in \mathscr{D}_n$ of (E.7) via Frank-Wolfe (Frank and Wolfe 1956) and projecting back to a permutation matrix $P^* = \arg\max_{P \in \Pi_n} \operatorname{tr} DP^T$, found via the Hungarian algorithm. The natural convex relaxation in (E.4), used in, for example, Zaslavskiy et al. (2009a), has the advantage of being solvable in polynomial time via the Frank-Wolfe algorithm (Frank and Wolfe 1956), but, Lyzinski et al. (2016a) showed that (E.7) is, in a certain sense, the correct optimization to consider. The authors considered the case of ρ-correlated Bernoulli random graphs A and B, and showed that under suitable conditions on the model parameters, the solution to (E.7) is, with high probability, the solution to (E.1). In a partial converse, the authors showed that with high probability the solution to (E.4) is not a solution to (E.1). The proof follows an argument broadly similar to that used in Chapter 4, in which standard concentration inequalities are used to show that certain conditions do or don't hold with high probability. Aflalo et al. (2015) considered similar questions related to the convex relaxation in (E.4), for a broad class of graphs, which they term *friendly graphs*. A friendly graph is defined to have a simple spectrum (i.e., the eigenvalues of its adjacency matrix are distinct) and none of its eigenvectors orthogonal to the vector of all ones. It is immediate that friendly graphs are a subset of the asymmetric graphs. The authors showed that in the case of friendly graphs, solutions to (E.4) are also solutions to (E.1), and extended their analysis to symmetric graphs (which are necessarily unfriendly) for a certain class of regularized versions of (E.4).

Based on ideas from sparse recovery (see Appendix D), Fiori et al. (2013) presented the group lasso graph matching (GLAG) algorithm. They modeled the adjacency matrices A and B as noisy, possibly permuted copies of an underlying adjacency matrix T, so that $A = T + O_A$ and $B = QTQ^T + O_B$, where $Q \in \Pi_n$ and O_A, O_B are sparse (but their nonzero entries may be arbitrary). Applying the group Lasso (Yuan and Lin 2006) with one entry for each graph edge, they obtained the penalty function

$$F(P) = \sum_{i,j} \sqrt{(AP)_{ij}^2 + (PB)_{ij}^2}$$

and the corresponding relaxed optimization

$$\hat{P} = \arg\min_{D \in \mathscr{D}_n} F(D).$$

This nonconvex optimization problem is solved (approximately) using the alternating direction method of multipliers (ADMM; Boyd et al. 2010).

E.2 Vertex Nomination

We turn now to a brief overview of the vertex nomination (VN) problem. This problem is the focus of Chapter 4. As discussed in that chapter, the VN problem is to find, given a collection of vertices labeled as being "interesting" or "uninteresting", other vertices in the graph that are likely to be interesting. In particular, we wish to rank the unlabeled vertices by how likely we believe them to be interesting. We call a ranking procedure of this sort a *nomination scheme*. Recalling the notation from Chapter 4, we are given a graph $G = (V, E)$ with vertex set V of size $|V| = n$, partitioned into K disjoint "blocks" $V_1, V_2, \dots, V_K$ so that $V = V_1 \cup V_2 \cup \cdots \cup V_K$, with $|V_k| = n_k$ for all $k \in [K]$. We let V_1 be the block of interest in the graph, with all other blocks being "uninteresting". We let $S \subseteq V$ denote the $|S| = m$ seed vertices, chosen uniformly at random from among all subsets of V of size m, whose block membership labels are known. These labels are unobserved for the non-seed vertices $U = V \setminus S$. We let $\mathfrak{u} = |U| = n - m$ denote the number of non-seed vertices, and we partition the blocks of V, so that $V_k = S_k \cup U_k$ for all $k \in [K]$. The goal of VN is to produce a nomination list $\mathscr{L} : U \to [\mathfrak{u}]$, i.e., a ranking of the non-seed vertices, so that the vertices in U_1 concentrate near the top of the ranking. Recall that we assess the quality of a ranking scheme by *average precision* (AP),

$$\mathrm{AP}(\mathcal{L}) = \frac{1}{\mathfrak{u}_1} \sum_{i=1}^{\mathfrak{u}_1} \frac{\sum_{j=1}^{i} \mathbb{I}\{\mathcal{L}^{-1}(j) \in U_1\}}{i}, \tag{E.8}$$

which ranges from 0 to 1, with $\mathrm{AP}(\mathscr{L}) = 1$ corresponding to perfect performance and $\mathrm{AP}(\mathscr{L}) = 0$ corresponding to the fact that none of the interesting vertices were ranked among the $\mathfrak{u}_1$ top vertices.

The VN problem overlaps quite heavily with the problem faced by recommender systems, a parallel discussed briefly in the survey by Coppersmith (2014). The typical recommender system task is to retrieve, based on a small number of documents deemed interesting, more documents from among a large collection that are also likely to be interesting. The key difference between this task and that of VN is that graph topology is explicit in the VN problem, whereas such network structure is not necessarily present in the case of, say, document retrieval. We refer the reader to the survey by Resnick and Varian (1997) and the handbook by Ricci et al. (2011) for a more thorough overview and discussion of recommender systems.

The vertex nomination problem was introduced in Coppersmith and Priebe (2012), in which the authors were motivated by the task of performing vertex nomination in the context of the Enron email data set (Priebe et al. 2005, see, e.g.,). Coppersmith and Priebe (2012) considered ranking schemes based on assumptions about the stochastic behavior of interesting vertices as compared to uninteresting vertices. They contrasted *context*-based statistics, which leverage the attributes (e.g., block labels, or other additional information) of neighboring vertices to classify a given vertex $u \in U$, with *content*-based statistics, which make use of attributes of the the edges incident on a vertex $u \in U$. The authors considered a range of convex combinations of such approaches, and explored their empirical performance. Suwan et al. (2015) extended this approach, recasting the problem within in a Bayesian framework.

Motivated by a similar task, Marchette et al. (2011) considered the vertex nomination problem in the context of random dot product graphs (RDPGs; Young and Scheinerman 2007), a latent position model (Hoff et al. 2002) that generalizes the notion of the stochastic block model. The authors extended this model to include edge-level attribute distributions, so that the attribute value of edge $(u, v) \in E$ depends on the latent positions of vertices $u, v \in V$. This attributed model lends itself to a VN scheme based on a linear discriminant, in contrast to an approach that simply uses a latent position estimation procedure akin to adjacency spectral embedding (ASE; see Appendix B for a brief overview). In follow-up work, Sun et al. (2012) compared the performance of two embedding-based nomination schemes and explored the task of estimating the power of such schemes using the Wilcoxon rank sum test (Wilcoxon 1945) and investigated the performance of embeddings related to ASE and multidimensional scaling (MDS; see Appendix B for a brief overview of ASE and MDS).

Finally, Fishkind et al. (2015), discussed at some length in Chapter 4, explored three approaches to vertex nomination. The first, called the *canonical* nomination, was based on an explicit computation of the distribution over all possible membership labelings, conditioned on the seeds S and the model parameters. This canonical approach is feasible only for small numbers of vertices, but was proved to be Bayes optimal for the vertex nomination scheme. A second approach, based on ASE, and a third approach, based on a maximum-likelihood formulation, were also presented. We refer the reader to Chapter 4 for a more thorough discussion of these three approaches. In particular, the maximum-likelihood approach is of particular interest in Chapters 4 and 5, and makes heavy use of the graph matching techniques discussed in the previous section.

Bibliography

O. Abdel-Hamid, L. Deng, D. Yu, and H. Jiang. Deep segmental neural networks for speech recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2013.

D. Achlioptas, F. McSherry, and B. Schölkopf. Sampling techniques for kernel methods. In *Advances in Neural Information Processing Systems (NIPS) 14*, pages 335–342, 2002.

L. A. Adamic and N. Glance. The political blogosphere and the 2004 US election. In *Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem*, 2005.

Y. Aflalo, A. Bronstein, and R. Kimmel. On convex relaxation of graph isomorphism. *Proceedings of the National Academy of Sciences of the United States of America*, 112(10):2942–2947, 2015.

R. Ahlswede and A. Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.

BIBLIOGRAPHY

E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

E. M. Airoldi, T. B. Costa, and S. H. Chan. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 692–700, 2013.

A. Y. Alfakih, A. Khandani, and H. Wolkowicz. Solving Euclidean distance matrix completion problems via semidefinite programming. *Computational Optimization and Applications*, 12:13–30, 1999.

C. Allauzen, M. Mohri, and M. Saraclar. General indexation of weighted automata–application to spoken utterance retrieval. In *Proceedings of the Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL*, 2004.

H. A. Almohamad and S. O. Duffuaa. A linear programming approach for the weighted graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):522–525, 1993.

S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

A. A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41(4): 2097–2122, 2013.

BIBLIOGRAPHY

A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

A. Andoni, M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In T. Darrell, P. Indyk, and G. Shakhnarovich, editors, *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press, 2006.

X. Anguera and M. Ferrarons. Memory efficient subsequence DTW for query-by-example spoken term detection. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2013.

X. Anguera, F. Metze, A. Buzo, I. Szöke, and L. J. Rodriguez-Fuentes. The spoken web search task. In *Proceedings of the MediaEval 2013 Multimedia Benchmark Workshop*, 2013.

M. Athineos and D. P. W. Ellis. Frequency-domain linear prediction for temporal features. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 261–266, 2003.

A. Athreya, C. E. Priebe, M. Tang, V. Lyzinski, D. J. Marchette, and D. L. Sussman. A limit theorem for scaled eigenvectors of random dot product graphs. *Sankhya A*, 78(1):1–18, 2016.

BIBLIOGRAPHY

J. Baglama and L. Reichel. Augmented implicitly restarted Lanczos bidiagonalization methods. *SIAM Journal of Scientific Computing*, 27(1):19–42, 2005.

Z. Bai and J. W. Silverstein. *Spectral Analysis of Large Dimensional Random Matrices*. Springer, 2010.

R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28 (3):253–263, 2008.

M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang. Message-passing algorithms for sparse network alignment. *ACM Transactions on Knowledge Discovery from Data*, 7(1):3:1–3:31, 2013.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. In *Proceedings of the 18th Annual Conference on Learning Theory*, pages 486–500, 2005.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold Regularization: A Geometric Framework for Learning from Examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Y. Bengio, J. F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems (NIPS) 16*, pages 177–184, 2004.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: a review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (8):1798–1828, 2013.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Springer, 2004.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

R. Bhatia. *Matrix Analysis*. Springer, 1997.

R. Bhatia and C. Davis. A better bound on the variance. *The American Mathematical Monthly*, 107(4):353–357, 2000.

P. J. Bickel and A. Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proceedings of the National Academy of Sciences of the United States of America*, 106:21068–21073, 2009.

G. Blackwood. *Lattice rescoring methods for statistical machine translation*. PhD thesis, Cambridge University Engineering Department and Clare College, 2010.

I. Borg and P. J. F. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: a Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

M. Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear algebra and its applications*, 415(1):20–30, 2006.

E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10(3):186–198, 2009.

C. J. C. Burges. Simplified support vector decision rules. In *Proceedings of the 13th International Conference on Machine Learning*, pages 71–77. Morgan Kaufmann, 1996.

R. Burkard, M. Dell'Amica, and S. Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, 2009.

BIBLIOGRAPHY

R. E. Burkard, S. E. Karisch, and F. Rendl. QAPLIB – a quadratic assignment problem library. *Journal of Global Optimization*, 10(4):391–403, 1997.

T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(4):515–519, 2004.

D. Cai, J. Han, X. He, K. Zhou, and H. Bao. Locality sensitive discriminant analysis. *International Joint Conference on Artificial Intelligence*, pages 708–713, 2007.

E. Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346:589–592, 2008.

E. Candès and T. Tao. The power of convex relaxation: near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5), 2010.

E. J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2009.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

E. J. Candès , J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006a.

E. J. Candès , J. K. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete fourier information. *IEEE Transactions on Information Theory*, 52(8):489–509, 2006b.

E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principle component analysis? *Journal of the ACM*, 58(3), 2011.

M. Carcassoni and E. R. Hancock. Spectral correspondence for point pattern matching. *Pattern Recognition*, 36(1):193–204, 2003.

M. Carlin, S. Thomas, A. Jansen, and H. Hermansky. Rapid evaluation of speech representations for spoken term discovery. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2011.

C. Carpineto and G. Romano. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys*, 44(1):1–50, 2012.

P. J. Carrington, J. Scott, and S. Wasserman. *Models and Methods in Social Network Analysis*. Cambridge University Press, 2005.

H. Chang and D.-Y. Yeung. Robust locally linear embedding. *Pattern Recognition*, 39(6):1053–1065, 2006.

S. Chatterjee. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2015.

K. Chaudhuri, F. Chung, and A. Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *Proceedings of the 25th Annual Conference on Learning Theory*, 2012.

C. Chelba and A. Acero. Position specific posterior lattices for indexing speech. In *Proceedings of the Association for Computational Linguistics*, 2005.

G. Chen, C. Parada, and G. Heigold. Small-footprint keyword spotting using deep neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

G. Chen, C. Parada, and T. Sainath. Query-by-example keyword spotting using long short-term memory networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

J. Chen, H.-R. Fang, and Y. Saad. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research*, 10:1989–2012, 2009.

L. Chen, J.T. Vogelstein, V. Lyzinski, and C. E. Priebe. A joint graph inference case study: the c.elegans chemical and electrical connectomes. *Worm*, 5(2):e1142041, 2016.

Y. Chen. Incoherence-optimal matrix completion. *IEEE Transactions on Information Theory*, 61(5):2909–2923, 2015.

Y. Chen, A. Jalali, S. Sanghavi, and C. Caramanis. Low-rank matrix recovery from errors and erasures. *IEEE Transactions on Information Theory*, 59(7):4324–4337, 2013.

A. Cherian, S. Sra, V. Morellas, and N. Papanikolopoulos. Efficient nearest neighbors via robust sparse hashing. *IEEE Transactions on Image Processing*, 23(8):3646–3655, 2014.

A. L. Chistov and D. Yu. Grigoriev. Complexity of quantifier elimination in the theory of algebraically closed fields. *Lecture Notes in Computer Science: Proceedings of the 11th Symposium on Mathematical Foundations of Computer Science*, 176:17–31, 1984.

M. Cho and K. M. Lee. Progressive graph matching: Making a move of graphs via probabilistic voting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 398–405, 2012.

F. Chung. *Spectral Graph Theory.* Number 92 in Conference Board of the Mathematical Sciences Regional Conference Series in Mathematics. American Mathematical Society, 1997.

F. Chung and L. Lu. Concentration inequalities and martingale inequalities: a survey. *Internet Math*, 3(1):79–127, 2006.

F. Chung, L. Lu, and V. Vu. Spectra of random graphs with given expected degrees.

Proceedings of the National Academy of Sciences of the United States of America, 100(11):6313–6318, 2003.

R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006.

D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265–298, 2004.

J. A. Cook, I. Sutskever, A. Mnih, and G. E. Hinton. Visualizing similarity data with a mixture of maps. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, volume 2, pages 67–74, 2007.

G. A. Coppersmith. Vertex nomination. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(2):144–153, 2014.

G. A. Coppersmith and C. E. Priebe. Vertex nomination via content and context. *arXiv preprint arXiv:1201.4118*, 2012.

T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *Advances in Neural Information Processing Systems (NIPS) 19*, pages 313–320. MIT Press, 2007.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2006.

T. Cox and M. Cox. Multidimensional scaling. *Monographs on Statistics and Applied Probability*, 88, 2001.

A. D. J. Cross and E. R. Hancock. Graph matching with a dual-step EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1236–1253, 1998.

S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstraus. *Random Structures and Algorithms*, 22(1):60–65, 2003.

R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40, 2008.

C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. *SIAM Journal of Numerical Analysis*, 7(1), March 1970.

L. S. Davis. Shape matching using relaxation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1):60–72, 1979.

F. de la Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1/2/3):117–142, 2003.

L. M. Delves and J. L. Mohamed. *Computational methods for integral equations.* Cambridge University Press, Cambridge, 1985.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39 (1):1–38, 1977.

D. L. Donoho. For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6):797–829, 2006a.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52 (4):1289–1306, 2006b.

D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, 100(10):5591–5596, 2003.

P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning*, 6:2153–2175, 2005.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006a.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast monte carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006b.

R. O. Duda and P. E. Hart. Use of the Hough transform to detect lines and curves in pictures. *Communications of the ACM*, 15(1), 1972.

K. Duh and K. Kirchhoff. Learning to rank with partially-labeled data. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 2008.

K. K. Duh. *Learning to Rank with Partially-Labeled Data*. PhD thesis, University of Washington, 2009.

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

F. Escolano, E. R. Hancock, and M. A. Lozano. Graph matching through entropic manifold alginment. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, Colorado, 2011.

F. Escolano, E. R. Hancock, M. Liu, and M. A. Lozano. Information-theoretic dissimilarities for graphs. In *Proceedings of the 2nd Similarity-Based Pattern Recognition International Workshop*, number 7953 in Lecture Notes in Computer Science, pages 90–105. Springer-Verlag, 2013.

U. Feige and E. Ofek. Spectral techniques applied to sparse random graphs. *Random Structures and Algorithms*, 27(2), 2005.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

M. Fiori, P. Sprechmann, J. T. Vogelstein, P. Musé, and G. Sapiro. Robust multimodal graph matching: Sparse coding meets graph matching. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 127–135, 2013.

D. E. Fishkind, S. Adali, and C. E. Priebe. Seeded graph matching. *arXiv preprint arXiv:1209.0367*, 2012.

D. E. Fishkind, D. L. Sussman, M. Tang, J. T. Vogelstein, and Carey E Priebe. Consistent adjacency-spectral partitioning for the stochastic block model when the model parameters are unknown. *SIAM Journal on Matrix Analysis and Applications*, 34:23–39, 2013.

D. E. Fishkind, V. Lyzinski, H. Pao, L. Chen, and C. E. Priebe. Vertex nomination schemes for membership prediction. *The Annals of Applied Statistics*, 9(3):1510–1532, 2015.

P. Foggia, G. Percannella, and M. Vento. Graph matching and learning in pattern recognition in the last 10 years. *International Journal of Pattern Recognition and Artificial Intelligence*, 28(01):1450001, 2014.

C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

B. Franke and P. J. Wolfe. Network modularity in the presence of covariates. *arXiv preprint arXiv:1603.01214*, 2016.

A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.

A. W. Fu, E. Keogh, L. Y. H. Lau, and C. A. Ratanamahatana. Scaling and time warping in time series querying. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2005.

Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.

M. Gales and S. Young. The application of hidden markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2008.

M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, 1979.

D. Gillick, L. Gillick, and S. Wegmann. Don't multiply lightly: Quantifying problems with the acoustic model assumptions in speech recognition. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 71–76, 2011.

A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 1999.

J. Glass. Towards unsupervised speech processing. In *Proceedings of the 11th International Conference on Information Sciences, Signal Processing and their Applications (ISSPA)*, 2012.

J. R. Glass. A probabilistic framework for segment-based speech recognition. *Speech Communication*, 17:137–152, 2003.

M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.

S. Golden. Lower bounds for the Helmholtz function. *Physical Review B*, 137:1127–1128, 1965.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, 4th edition, 2013.

J. C. Gower and G. B. Dijksterhuis. *Procrustes Problems*. Number 30 in Oxford Statistical Science Series. Oxford University Press, 2004.

D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57:1548–1566, 2011.

J. Ham, D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the 21st International Conference on Machine Learning*. ACM, 2004.

D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12): 2639–2664, 2004.

J. He, S. Kumar, and S.-F. Chang. On the difficulty of nearest neighbor search. In *29th International Conference on Machine Learning*, Edinburgh, Scotland, 2012.

G. Heigold, P. Nguyen, M. Weintraub, and V. Vanhoucke. Investigations on exemplar-based features for speech recognition towards thousands of hours of unsupervised, noisy data. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4437–4440, 2012.

M. Hein and M. Maier. Manifold denoising. In *Advances in Neural Information Processing Systems (NIPS) 19*, pages 561–568, 2007.

M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds – weak and strong pointwise consistency of graph Laplacians. In *Proceedings of the 18th Annual Conference on Learning Theory*, pages 470–485, 2005.

R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 29–45. MIT Press, 2000.

T. Hey, S. Tansley, and K. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

G. E. Hinton and S. T. Roweis. Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems (NIPS) 15*, pages 857–864, 2002.

G. R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):530–549, 2003.

P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36:1171–1220, 2008.

P. W. Holland and R. E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics: Theory and Methods*, 6(9):813–827, 1977.

P. W. Holland, K. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1994.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 1936.

G. Hristescu and M. Farach-Colton. Cluster-preserving embedding of proteins. Technical Report 99-50, Rutgers University, 1999.

P. J. Huber. *Robust Statistics*. John Wiley & Sons, 2nd edition, 2009.

L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

P. Indyk. Stable distributions, pseudorandom number generators, embeddings and data stream computation. In *Symposium on Foundations of Computer Science*, 2000.

P. Indyk. Algorithmic applications of low-distortion geometric embeddings. In *42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001.

P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.

A. Jansen and B. Van Durme. Efficient spoken term discovery using randomized algorithms. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2011.

A. Jansen and B. Van Durme. Indexing raw acoustic features for scalable zero resource search. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2012.

A. Jansen, K. Church, and H. Hermansky. Towards spoken term discovery at scale with zero resources. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2010.

A. Jansen, E. Dupoux, S. Goldwater, M. Johnson, S. Khudanpur, K. Church, N. Feldman, H. Hermansky, F. Metze, R. Rose, M. Seltzer, P. Clark, I. McGraw, B. Varadarajan, E. Bennett, B. Borschinger, J. Chiu, E. Dunbar, A. Fourtassi, D. Harwath, Chia ying Lee, K. Levin, A. Norouzian, V. Peddinti, R. Richardson, T. Schatz, and S. Thomas. A summary of the 2012 CLSP workshop on zero resource speech technologies and models of early language acquisition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.

A. Javanmard and A. Montanari. Localization from incomplete noisy distance measurements. *Foundations of Computational Mathematics*, 13(3):297–345, 2013.

F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.

W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

I. M. Johnstone and D. M. Titterington. Statistical challenges of high-dimensional data. *Philosophical Transactions of the Royal Society of London A*, 367(1906): 4237–4253, 2009.

I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2nd edition, 2002.

R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987.

A. Joseph and B. Yu. Impact of regularization on spectral clustering. In *Proceedings of the IEEE Information Theory and Applications Workshop*, pages 1–2, 2014.

A. Kandel, H. Bunke, and M. Last. *Applied Graph Theory in Computer Vision and Pattern Recognition*, volume 1. Springer, 2007.

D. M. Kane and J. Nelson. Sparser Johnson-Lindenstrauss transforms. *Journal of the ACM*, 61(1), 2014.

B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83, 2011.

T. Kato. *Perturbation Theory for Linear Operators.* Springer-Verlag, 1995.

Y. Keselman, A. Shokoufandeh, M. F. Demirci, and S. Dickinson. Many-to-many graph matching via metric embedding. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 850–857, 2003.

H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2010a.

H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11:2057–2078, 2010b.

J. Kittler and F. M. Alkoot. Sum verus vote fusion in multiple classifier systems. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 25(1):110–115, 2003.

G. W. Klau. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics*, 10(1):S59, 2009.

O. Klopp, A. B. Tsybakov, and N. Verzelen. Oracle inequalities for network models and sparse graphon estimation. *Retrieved from arXiv*, 2015. URL `http://arxiv.org/abs/1507.04118`.

D. Knossow, A. Sharma, D. Mateus, and R. Horaud. Inexact matching of large and sparse graphs using Laplacian eigenvectors. In *Proceedings of the 7th International Workshop on Graph-Based Representations in Pattern Recognition*, 2009.

D. Koller and N. Friedman. *Probabilistic Graphical Models*. MIT Press, Cambridge, MA, 2009.

A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS) 25*, pages 1097–1105, 2012.

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

B. Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2012.

B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *Proceedings of the 12th International Conference on Computer Vision*, 2009.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling techniques for the Nyström method.

In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, 2009.

J. Labiak and K. Livescu. Nearest neighbors with learned distances for phonetic frame classification. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2337–2340, 2011.

K. G. Larsen and J. Nelson. The Johnson-Lindenstrauss lemma is optimal for linear dimensionality reduction. In *Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP)*, 2016.

M. I. Layton and M. J. F. Gales. Acoustic modelling using continuous rational kernels. In *Proceedings of the IEEE Workshop on Machine Learning for Signal Processing*, pages 165–170, 2005.

C. M. Le, E. Levina, and R. Vershynin. Concentration and regularization of random graphs. *Retrieved from arXiv*, 2016. URL http://arxiv.org/abs/1506.00669.

Q. Le, T. Sarlós, and A. Smola. Fastfood – approximating kernel expansions in loglinear time. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Information Science and Statistics. Springer, 2007.

M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Proceedings of the Tenth IEEE Conference on Computer Vision*, pages 1482–1489, 2005.

K. Levin and V. Lyzinski. Laplacian eigenmaps from sparse, noisy similarity measurements. *IEEE Transactions on Signal Processing*, to Appear.

K. Levin, K. Henry, A. Jansen, and K. Livescu. Fixed-dimension acoustic embeddings of variable-length segments in low-resource settings. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013.

K. Levin, A. Jansen, and B. Van Durme. Segmental acoustic indexing for zero resource keyword search. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

S. Z. Li. Matching: Invariant to translations, rotations and scale changes. *Pattern Recognition*, 25(6):583–594, 1992.

E. H. Lieb. Convex trace functions and the Wigner-Yanase-Dyson conjecture. *Advances in Mathematics*, 11(3):267–288, 1973.

Y. Lin, R. Jin, D. Cai, S. Yan, and X. Li. Compressed hashing. In *Proceedings of the IEEE Computer Society Converence on Computer Vision and Pattern Recognition*, pages 446–451, 2013.

N. Linial. Finite metric spaces– combinatorics, geometry and algorithms. In *Proceedings of the International Congress of Mathematicians III*, pages 573–586, 2002.

D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985.

W. Liu, J. Wang, S. Kumar, and S. Chang. Hashing with graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*, Bellevue, WA, USA, 2011.

Z. Y. Liu, H. Qiao, and L. Xu. An extended path following algorithm for graph-matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1451–1456, 2012.

L. Lu and X. Peng. Spectra of edge-independent random graphs. *Electronic Journal of Combinatorics*, 20, 2013.

B. Luo and E. R. Hancock. Alignment and correspondence using singular value decomposition. In *Proceedings of the Joint International Association of Pattern Recognition International Workshops SSPR 2000 and SPR 2000*. Springer, 2000.

V. Lyzinski, D. E. Fishkind, and C. E. Priebe. Seeded graph matching for correlated Erdős-Rényi graphs. *Journal of Machine Learning Research*, 15:3513–3540, 2014a.

V. Lyzinski, D. L. Sussman, M. Tang, A. Athreya, and C. E. Priebe. Perfect clustering for stochastic blockmodel graphs via adjacency spectral embedding. *Electronic Journal of Statistics*, 8(2):2905–2922, 2014b.

V. Lyzinski, D. L. Sussman, D. E. Fishkind, H. Pao, L. Chen, J. T. Vogelstein,

Y. Park, and C. E. Priebe. Spectral clustering for divide-and-conquer graph matching. *Parallel Computing*, 47:70–87, 2015.

V. Lyzinski, D. E. Fishkind, M. Fiori, J. T. Vogelstein, C. E. Priebe, and G. Sapiro. Graph matching: Relax at your own risk. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):60–73, 2016a.

V. Lyzinski, K. Levin, D. Fishkind, and C. E. Priebe. On the consistency of the likelihood maximization vertex nomination scheme: Bridging the gap between maximum likelihood estimation and graph matching. *Journal of Machine Learning Research*, 17, 2016b.

A. L. Maas, S. D. Miller, T. M. O'Neil, A. Y. Ng, and P. Nguyen. Word-level acoustic modeling with convolutional vector regression. In *Proceedings of the ICML 2012 Workshop on Representation Learning*, 2012.

J. Malik, P. Arbeláez, J. Carreira, K. Fragkiadaki, R. Girshick, G. Gkioxari, S. Gupta, B. Hariharan, A. Kar, and S. Tulsiani. The three R's of computer vision: Recognition, reconstruction and reorganization. *Pattern Recognition Letters*, 72:4–14, 2016.

C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

G. Mantena and X. Anguera. Speed improvements to information retrieval-based dynamic time warping using hierarchical k-means clustering. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.

D. Marchette, C. E. Priebe, and G. A. Coppersmith. Vertex nomination via attributed random dot product graphs. In *Proceedings of the 57th ISI World Statistics Congress*, volume 6, page 16, 2011.

J. Masci, A. M. Bronstein, M. M. Bronstein, P. Sprechmann, and G. Sapiro. Sparse similarity-preserving hashing. In *Proceedings of the International Conference on Learning Representations*, 2014.

B. McFee and G. R. Lanckriet. Metric learning to rank. In *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

T. Mei, Y. Rui, S. Li, and Q. Tian. Multimedia search reranking: A literature survey. *ACM Computing Surveys*, 46(3):38:1–38:38, 2014.

F. Metze, X. Anguera, E. Barnard, M. Davel, and G. Gravier. The spoken web search task at MediaEval 2012. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.

D. R. H. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish. Rapid and accurate spoken term detection. In *Proceedings* *of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2007.

V. D. Milman and G. Schechtman. *Asymptotic theory of finite dimensional normed spaces*, volume 1200 of *Lecture Notes in Mathematics*. Springer, 1986.

M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

A. R. Mohamed, G. E. Dahl, and G. E. Hinton. Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech and Language Processing*, 20(1): 14–22, 2012.

I. Morison. *Introduction to Astronomy and Cosmology*. WIley-Blackwell, 2008.

B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006.

M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104, 2006a.

M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23): 8577–8582, 2006b.

M. E. J. Newman. Community detection in networks: Modularity optimization and maximum likelihood are equivalent. *arXiv preprint arXiv:1606.02319*, 2016.

M. E. J. Newman and A. Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(11863), 2016.

M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):1–15, February 2004. ISSN 1539-3755.

W. De Nooy, A. Mrvar, and V. Batagelj. *Exploratory social network analysis with Pajek*. Cambridge University Press, 2011.

S. C. Olhede and P. J. Wolfe. Network histograms and universality of block model approximation. *Proceedings of the National Academy of Sciences of the United States of America*, 111:14722–14727, 2014.

R. I. Oliviera. Concentration of the adjacency matrix and of the Laplacian in random graphs with independent edges. *arXiv preprint arXiv:0911.0600*, 2010.

S. O'Rourke, V. Vu, and K. Wang. Eigenvectors of random matrices: a survey. *Journal of Combinatorial Theory, Series A*, 144:361–442, 2016a.

S. O'Rourke, V. Vu, and K. Wang. Random perturbation of low rank matrices: Improving classical bounds. *arXiv preprint arXiv:1311.2657*, 2016b.

M. Ostendorf. From HMMs to segment models: Stochastic modelling for CSR. In

C.-H. Lee, F. K. Soong, and K. K. Paliwal, editors, *Automatic Speech and Speaker Recognition: Advanced Topics*, chapter 8, pages 185–209. Springer, 1996.

C. Parada, A. Sethy, and B. Ramabhadran. Query-by-example spoken term detection for OOV terms. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009.

A. Park and J. R. Glass. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

M. Paul, E. Sumita, and S. Yamamoto. Example-based rescoring of statistical machine translation output. In *Proceedings of HLT-NAACL: Short Papers*, 2004.

F. Peng, S. Roy, B. Shahshahani, and F. Beaufays. Search results based N-best hypothesis rescoring with maximum entropy classification. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013.

V. T. Pham, H. Xu, X. Xiao, N. F. Chen, E. S. Chng, and H. Li. Keyword search using query expansion for graph-based rescoring of hypothesized detections. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park. Scan stiatistics on Enron graphs. *Computational and Mathematical Organization Theory*, 11:229–247, 2005.

C. E. Priebe, D. L. Sussman, M. Tang, and J. T. Vogelstein. Statistical inference on errorfully observed graphs. *Journal of Computational and Graphical Statistics*, 24 (4):930–953, 2014.

T. Qin and K. Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 3120–3128, 2013.

H. J. Qiu and E. R. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1873–1890, 2007.

T. F. Quatieri. *Discrete-Time Speech Signal Processing: Principles and Practice.* Prentice Hall, 2002.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

A. E. Raftery and N. Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101(473):168–178, 2006.

P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Ad-*

vances in Neural Information Processing Systems (NIPS) 20, pages 1177–1184, 2008.

A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems (NIPS) 21*, pages 1313–1320, 2009.

T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2012.

C. R. Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Society, Series B*, 10(2):159–203, 1948.

A. Rastrow, M. Dreyer, A. Sethy, S. Khudanpur, B. Ramabhadran, and M. Dredze. Hill climbing on speech lattices: a new rescoring framework. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

B. Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12:3413–3430, 2011.

P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

F. Ricci, L. Rokach, and B. Shapira. *Introduction to recommender systems handbook*. Springer, 2011.

A. Robles-Kelly and E. Hancock. A Riemannian approach to graph embedding. *Pattern Recognition*, 40(3):1042–1056, 2007.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011a.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39:1878–1915, 2011b.

K. M. Rosen. Analysis of speech segment duration with the lognormal distribution: A basis for unification and comparison. *Journal of Phonetics*, 33(4), 2005.

S. Rosenberg. *The Laplacian on a Riemannian manifold*. Cambridge University Press, 1997.

S. T. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision*, pages 59–66, 1998.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3): 211–252, 2015.

I. Ruthaven and M. Lalmas. A survey on the use of relevance feedback for information access systems. *The Knowledge Engineering Review*, 18(2):95–145, 2003.

T.N. Sainath, B. Ramabhadran, D. Nahamoo, D. Kanevsky, D. Van Compernolle, K. Demuynck, J. F. Gemmeke, J. R. Bellegarda, and S. Sundaram. Exemplar-based processing for speech recognition: An overview. *IEEE Signal Processing Magazine*, 29(6):98–113, 2012.

H. Sak, A. W. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2014.

H. Sak, A. W. Senior, K. Rao, and F. Beaufays. Fast and accurate recurrent neural network acoustic models for speech recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2015.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

R. R. Salakhutdinov and G. E. Hinton. Semantic hashing. In *Proceedings of the ACM SIGIR Workshop on Information Retrieval and Applications of Graphical Models*, 2007.

L. Saul and S. T. Roweis. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

C. Schellewald and C. Schnörr. Probabilistic subgraph matching based on convex relaxation. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 171–186. Springer, 2005.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):545–561, 1995.

G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two images. *Proceedings of the Royal Society B*, 224(1309):21–26, 1991.

N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Robust principle component analysis on graphs. *Retrieved from arXiv*, 2015. URL `http://arxiv.org/abs/1504.06151`.

L. S. Shapiro and J. M. Brady. Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, 10(5):283–288, 1992.

A. Shashua and A. Levin. Taxonomy of large margin principle algorithms for ordinal regression. In *Advances in Neural Information Processing Systems (NIPS) 15*, pages 961–968, 2003.

R. Shibata. Consistency of model selection and parameter estimation. *Journal of Applied Probability*, 23:127–141, 1986.

R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences of the United States of America*, 105(35):12763–12768, 2008.

R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Annals of Mathematical Statistics*, 35(2):876–879, 1964.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the 17th International Conference on Machine Learning*, 2000.

V. Soto, L. Mangu, A. Rosenberg, and J. Hirschberg. A comparison of multiple methods for rescoring keyword search lists for low resource languages. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2014.

G. W. Stewart and J. Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

M. Sun, M. Tang, and C. E. Priebe. A comparison of graph embedding methods for vertex nomination. In *11th International Conference on Machine Learning and Applications*, volume 1, pages 398–403, 2012.

N. Sundaram, A. Turmukhametova, N. Satish, T. Mostak, P. Indyk, S. Madden, and P. Dubey. Streaming similarity search over one billion tweets using parallel locality-sensitive hashing. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2014.

D. L. Sussman, M. Tang, D. E. Fishkind, and C. E. Priebe. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107(499):1119–1128, 2012.

D. L. Sussman, M. Tang, and C. E. Priebe. Consistent latent position estimation and vertex classification for random dot product graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:48–57, 2014.

S. Suwan, D. S. Lee, and C. E. Priebe. Bayesian vertex nomination using content and context. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(6):400–416, 2015.

C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *Advances in Neural Information Processing Systems (NIPS) 26*, pages 2553–2561, 2013.

M. Talagrand. A new look at independence. *Annals of Probability*, 24:1–34, 1996.

H. Tang, M. Hasegawa-Johnson, and T. Huang. A novel vector representation of stochastic signals based on adapted ergodic HMMs. *IEEE Signal Processing Letters*, 17(8):715–718, 2010.

M. Tang and C. E. Priebe. Limit theorems for eigenvectors of the normalized Laplacian for random graphs. *arXiv preprint arXiv:1607.08601*, 2016.

M. Tang, D. L. Sussman, and C. E. Priebe. Universally consistent vertex classification for latent position graphs. *The Annals of Statistics*, 31:1406–1430, 2013.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

S. Thomas, S. Ganapathy, and H. Hermansky. Phoneme recognition using spectral envelope and modulation frequency features. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

C. J. Thompson. Inequality with applications in statistical mechanics. *Journal of Mathematical Physics*, 6:1812–1813, 1965.

V. S. Tomar and R. C. Rose. Application of a locality preserving discriminant analysis approach to ASR. In *Proceedings of the 11th International Conference on Information Sciences, Signal Processing and their Applications (ISSPA)*, pages 103–107, 2012.

W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

P. H. S. Torr. Solving Markov random fields using semi definite programming. In C. M. Bishop and B. J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, January 2003.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Found. Computational Mathematics*, 12(4):389–434, 2012.

J. A. Tropp. An introduction to matrix concentration inequalities. *Found. and Trends in Machine Learning*, 8(1-2):1–230, 2015.

M. W. Trosset. Distance matrix completion by numerical optimization. *Computational Optimization and Applications*, 17(1):11–22, 2000.

M. W. Trosset and M. Tang. On combinatorial laplacian eigenmaps. Technical Report 10-02, Department of Statistics, Indiana University, October 2010.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

R. E. Ulanowicz, C. Bondavalli, and M. S. Egnotovich. Network analysis of trophic dynamics in South Florida ecosystems, FY 97: The Florida Bay ecosystem. Annual Report to the U.S. Geological Survey, Biological Resources Division. Ref. No. [UMCES]CBL 98-123, 1997.

S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

L. van der Maaten and G. E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

L. J. P. van der Maaten, E. O. Postma, and H. J. van den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10(1-41): 66–71, 2009.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 2000.

A. W. van der Vaart and J. A. Wellner. *Weak convergence and empirical processes*. Springer, New York, 1996.

M. A. van Wyk, T. S. Durrani, and B. J. van Wyk. RKHS interpolator-based graph matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):988–995, 2002.

J. T. Vogelstein, J. M. Conroy, V. Lyzinski, L. J. Podrazik, S. G. Kratzer, E. T. Harley, D. E. Fishkind, R. T. Vogelstein, and C. E. Priebe. Fast approximate quadratic programming for graph matching. *PLoS ONE*, 10(04), 2015.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4): 395–416, 2007.

U. von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, 36(2):555–586, 2008.

M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle. Template-based continuous speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1377–1390, 2007.

H. F. Wang and E. R. Hancock. Correspondence matching using kernel principal components analysis and label consistency constraints. *Pattern Recognition*, 39(6): 1012–1025, 2006.

J. Wang, M. Li, Z. Li, and W.-Y. Ma. Learning ranking function via relevance propagation. Technical report, Microsoft Research Asia, 2005.

J. Wang, S. Kumar, and S. Chang. Sequential projection learning for hashing with compact codes. In *Proceedings of the 27th International Conference on Machine learning*, Haifa, Israel, 2010.

S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems (NIPS) 21*, pages 1753–1760, 2009.

E. P. Wigner. On the distributions of the roots of certain symmetric matrices. *Annals of Mathematics*, 67:325–327, 1958.

F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6): 80–83, 1945.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS) 13*, pages 682–688, 2001.

B. Xiao, E. R. Hancock, and R. C. Wilson. A generative model for graph matching and embedding. *Computer Vision and Image Understanding*, 113(7):777–789, 2009.

W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig. Achieving human parity in conversational speech recognition. Technical Report MSR-TR-2016-71, Microsoft Research, 2016.

S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007.

J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, 2015.

J. Yang, J. McAuley, and J. Leskovec. Community detection in networks with node attributes. In *Proceedings of the IEEE 13th International Conference on Data Mining*, pages 1151–1156, 2013.

S. Young and E. Scheinerman. Random dot product graph models for social networks. In *Proceedings of the 5th International Conference on Algorithms and Models for the Web-graph*, pages 138–149, 2007.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 68(1):49–67, 2006.

W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.

M. Zaslavskiy, F. Bach, and J.P. Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227–2242, 2009a.

M. Zaslavskiy, F. Bach, and J.P. Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227–2242, 2009b.

M. Zaslavskiy, F. Bach, and J.P. Vert. Many-to-many graph matching: A continuous relaxation approach. In J. L. Balcázar, F. Bonchi, A. Bionis, and M. Sebag, editors, *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases: ECML PKDD 2010*, pages 515–530. Springer Berlin Heidelberg, 2010.

S. Zhang and H. Tong. FINAL: fast attributed network alignment. In *Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2016.

Y. Zhang and J. Glass. Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009.

Y. Zhang and J. Glass. A piecewise aggregate approximation lower-bound estimate for posteriorgram-based dynamic time warping. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2011.

Y. Zhang, E. Levina, and J. Zhu. Community detection in networks with node features. *arXiv preprint arXiv:1509.01173*, 2015.

D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *Advances in Neural Information Processing Systems (NIPS) 16*, pages 169–176, 2004.

F. Zhou and F. De la Torre. Factorized graph matching. *IEEE Pattern Analysis and Machine Intelligence*, 38(9):1774–1789, 2016.

V. Zue, J. Glass, M. Phillips, and S. Seneff. The MIT SUMMIT speech recognition system: A progress report. In *Proceedings of the Workshop on Speech and Natural Language*, pages 179–189, 1989.

G. Zweig, P. Nguyen, D. Van Compernolle, K. Demuynck, L. Atlas, P. Clark, G. Sell, M. Wang, F. Sha, H. Hermanskyi, D. Karakos, A. Jansen, S. Thomas, S. Bowman, J. Kao, and G. S. V. S. Sivaram. Speech recognition with segmental conditional random fields: A summary of the JHU CLSP 2010 summer workshop. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

Vita

Keith Levin was born in 1988 in Boston, Massachusetts. He received B.S. degrees in Psychology and Linguistics from Northeastern University in 2011. During his time at Northeastern University, he worked as a research assistant for Dr. Neal Pearlmutter and Dr. Joanne Miller, and later worked at BBN Technologies as a data analyst. He enrolled in the Computer Science Ph.D. program at Johns Hopkins University in 2012. His research focuses on large-scale search and indexing problems and network analysis, with primary applications in neuroscience and speech processing.