NONCONVEX NONSMOOTH OPTIMIZATION FOR BIOINFORMATICS

By

Amina Shabbeer

A Dissertation Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: COMPUTER SCIENCE

Examining Committee:

Kristin P. Bennett, Dissertation Adviser

Bülent Yener, Member

John Mitchell, Member

Malik Magdon-Ismail, Member

Rensselaer Polytechnic Institute
Troy, New York

August 2013
(For Graduation December 2013)

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

CONTENTS

APPENDICES

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I would like to thank my advisor Prof. Kristin P. Bennett for her guidance and constant encouragement and support. I have been fortunate to have had the chance to work under her; her unique vision has shaped the way I think about research problems. I will be forever grateful for her knowledge and the many opportunities she has given me throughout. This thesis would not be possible without her.

I would like to thank my committee members Prof. Bülent Yener, Prof. John Mitchell and Prof. Malik Magdon-Ismail for their keen insight and valuable suggestions. I am especially grateful to Prof. Yener for his benevolence and wisdom. His invaluable support and encouragement gave me the motivation to persist in working towards my goals. I am very thankful to Prof. Chris Bystroff, Prof. John Mitchell, Prof. Mark Goldberg and Prof. Mohammed Zaki for their enthusiastic and comprehensive teaching. I have learned a lot from their courses; the knowledge they imparted has helped me broaden the scope of my work. I am thankful to Prof. George Plopper for his great advice during my interactions with him as a teaching assistant at RPI.

I would like to thank my co-authors Dr. Lauren Cowan of the Centers of Disease Control, and Dr. Nalin Rastogi, Institut Pasteur de Guadeloupe for sharing their knowledge and their thoughtful and detailed analysis. Doing research with members of the TB-Insight team: Dr. Minoo Aminian, James Blondin, Veronica Ahiati, has been a great learning experience. I am thankful for the opportunity to work with them. I would like to acknowledge Dr. Inna Vitol, Dr. Gregory Moore and Dr. Charles Bergeron; their prior work has had a significant impact on my research. I am grateful to my collaborators Jacob Katz, Dan Croft, Michael Gonzales, Janani Ranganathan and Srivatsan Raghavan for their contributions that have facilitated the development and progress of my research.

I would like to express my gratitude to the truly helpful staff of the Computer Science and Mathematical Sciences Departments, graduate co-ordinators Terry Hayden and Dawnmarie Robens, and Chris Coonrad and Joanne Kessler.

My time at RPI was made pleasant by the company of excellent friends. Thank you to Prof. Curtis Bahn for introducing me to sitar, that made my time at RPI so much more enjoyable. I would like to express my heartfelt gratitude to John Felix for his continuous encouragement and help.

I remain indebted to my brother, Khurush and sister, Shabana who have been my source of strength and inspiration throughout. The immense love, kindness and understanding of my parents have made so many things possible. I am extremely fortunate to have the support of my loving family.

ABSTRACT

In this thesis, we investigate nonconvex nonsmooth optimization problems that arise in bioinformatics and general visual data analytics tasks. This work is motivated by a need for tools that provide a view of the genetic diversity in the *Mycobacterium tuberculosis complex* (MTBC) population by extracting information from molecular epidemiological data. Such tools are crucial for the effective tracking and control of tuberculosis (TB). We survey classification tools that group MTBC strains into genetic families and visualization tools for molecular epidemiology of TB. We develop TB-Lineage a classification tool that employs Bayesian Networks and domain knowledge of signature patterns in DNA fingerprint data for MTBC major lineage classification. However, there is no consensus on MTBC lineage and sublineage definitions amongst experts from the perspective of both phylogenetic analysis and epidemiology. Understanding the evolutionary history and genetic relatedness of strains is essential to developing more accurate lineage definitions. We create *spoligoforests* as a tool for visualization of biogeographic diversity of MTBC that accurately represents both the underlying evolutionary relationships and the genetic distances between strains thus creating new insights on lineage definitions. Generating such a graph embedding involves addressing two challenges (i) preserve proximity relations as measured by some embedding objective, and (ii) simultaneous optimization of an aesthetic criterion, no edge-crossings in the embedding, to create a clear representation of the underlying graph structure. We propose a new approach to generating such an embedding that optimizes for multiple criteria. The method uses the theorems of the alternative to express the condition for no edge-crossings as a system of nonlinear inequality constraints. This approach has an intuitive geometric interpretation closely related to support vector machine classification. While edge crossing minimization can be utilized in conjunction with any optimization-based embedding objective, here we demonstrate the approach on multidimensional scaling by modifying the stress majorization algorithm to include penalties for edge crossings. We use an alternating approach to solve this nonconvex problem, iteratively performing two steps: computing the layout for a given set of constraints, and altering the constraints based on the new embedding. We provide a detailed analysis of the convergence of this algorithm. Alternating Directions of Multiplier Methods (ADMM) are proposed as a method for efficiently handling a large number of non-smooth constraints. *MAA+*, an iterative solution using ADMM is described for generating graph embeddings that adhere to non-intersection constraints between edges. We create *spoligoforests* generated for all strains of TB observed in patients diagnosed with TB in the U.S. from 2006 to 2010. We also developed a standalone tool for drawing spoligoforests. The method is also demonstrated on a suite of randomly generated graphs with corresponding Euclidean distances that have planar embeddings with high stress. The proposed edge-crossing constraints and iterative penalty algorithm can be readily adapted to other supervised and unsupervised optimization-based embedding or dimensionality reduction methods. The constraints can be generalized to remove intersections of general convex polygons including node-edge and node-node intersections. Host-pathogen maps that visualize relationships between MTBC strains and host groups are proposed as a testbed for minimizing intersections between nodes of arbitrary shape and size. *MAA+* is applied to variations of the proximity preservation and edge-crossing minimization problem, to create low stress embeddings with no overlaps between nodes, edges and subgraphs.

CHAPTER 1
Computational Tools for TB Epidemiology

This work was motivated by the need for a better understanding of the genetic diversity in the *Mycobacterium tuberculosis* complex (MTBC), the causative agent of tuberculosis (TB). The observed biogeographic diversity in MTBC has important phenotypic consequences. Strains are known to vary in their virulence, immunogenicity, transmissibility, associations with host groups and drug-resistance [48, 47]. A better understanding of the genetic diversity and structure of the MTBC population will help in the development of better TB tracking and control measures.

We begin this thesis with a survey of classification tools for determining MTBC lineages and visualization tools for the molecular epidemiology of TB that help create insight into MTBC genetic diversity. . We also present TB-Lineage, our own tool for major lineage that incorporates domain and expert knowledge, later in Chapter 1. Note, however, there is no clear consensus on the definition of each lineage or the desired level of granularity in lineage classification. Phylogenetic analyses using different biomarkers have resulted in varying definitions of clades of the MTBC [56, 5, 48]. From an epidemiological perspective as well, varying levels of granularity for subdivisions of the MTBC population have been proposed [93, 40, 41, 19]. Therefore, in addition to automated classification tools, there is a need to provide researchers with tools that provide a view of the genetic diversity of MTBC strains and generate insights about the variations in the MTBC population.

Spoligoforests, described in this chapter, are *phylogenetic forests* that represent the evolutionary relationships and relatedness of strains. They are an effort towards addressing the need for visualizations of genetic diversity of MTBC. These visualizations must satisfy

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Web tools for molecular epidemiology of tuberculosis*, Infect. Genet. Evol., 12 (2012), pp. 767-781.
(2) A. SHABBEER, L. COWAN, C. OZCAGLAR, ET AL., *TB-Lineage: An online tool for classification and analysis of strains of Mycobacterium tuberculosis complex*, Infect. Genet. Evol., 12 (2012), pp. 789-797.
(3) G. KUNAPULI, K. P. BENNETT, A. SHABBEER, ET AL., *Online knowledge-based support vector machines, in Machine Learning and Knowledge Discovery in Databases*, Barcelona, Spain, 2010, pp. 145-161.

the following two requirements: depict proximities between strains as well as clearly represent the evolutionary history of strains as described by the network structure i.e. have no intersections between nodes and edges obscuring the graph structure. There exist several good graph drawing techniques that achieve these embedding goals individually. However, developing methods that jointly accomplish both goals, proximity preservation as well as minimizing intersection of nodes and edges, presents interesting challenges. We propose to address these challenges using mathematical programming techniques. We now define the three main contributions of this work:

- Formulation of a novel embedding objective that seeks to satisfy multiple functional and aesthetic requirements for graph layouts – preservation of proximity relations between nodes as defined in a high-dimensional space in the two-dimensional embedding, and non-intersection of nodes and edges so as to create a clear and effective representation. We show that the condition for non-intersection of edges (nodes) is equivalent to the feasibility of a system of nonlinear inequalities. These inequality constraints can be incorporated as penalties into a continuous embedding objective. This concept is demonstrated by defining an algorithm, **MAA** that minimizes stress (discrepancy between distances in high-dimensional space and distances in the reduced Euclidean space) while also reducing the number of edge-crossings. The algorithm is a modification of the stress majorization technique that includes penalty functions for edge-crossings into the stress function. The formulation of the edge-crossing constraints, the algorithm and initial results are provided in Chapter 4. A background of existing proximity preserving graph drawing methods and edge-cross minimization techniques is provided in Chapter 3.

- We propose the use of Alternating Directions of Multiplier Methods (ADMM) for the development of scalable algorithms that can efficiently minimize the nonconvex, nonsmooth objectives associated with the problems arising from the constrained embedding task. In Chapter 5, we describe the motivation and algorithmic framework of ADMM. We then describe **MAA+** that applies ADMM-based solutions to the two problems of finding the separating planes for all pairs of edges (nodes) and minimizing the penalized embedding objective. We present results of spoligoforest generation for the entire set of MTBC strains identified in the U.S. since 2006 obtained from the CDC.

- We present practical applications that serve as testbeds for the concepts developed in this work. In Chapter 5, we present a standalone spoligoforest generating tool that implements implements MAA+ to create spoligoforests described in Chapter 1. The tool also provides various measures of genetic similarities for spoligotypes and MIRU types to represent proximity between strains. In Chapter 5 we describe host-pathogen maps for depicting host-pathogen associations. We describe the challenges posed by node overlaps and node-edge intersections that occur when minimizing an objective that preserves proximity relations between nodes and propose solutions to address these. We demonstrate the application of overlap removal on objects of arbitrary shape and size in several variations of the problem such as minimizing overlaps between pairs of nodes, edges and subgraphs, while simultaneously minimizing the proximity-preserving function.

1.1 Molecular Epidemiology of TB

In this chapter, we explore some classification tools that utilize molecular epidemiological data to address current challenges in the understanding of the genetic diversity of MTBC. While it does not constitute the main contribution of the thesis, these tools laid the foundation for and provided the original motivation for the work addressed in subsequent chapters.

Molecular epidemiology integrates molecular biology with traditional epidemiological approaches to study the influence of factors identified at the molecular level on the characteristics of MTBC, and the distribution and control of TB. TB surveillance and control programs now routinely perform DNA fingerprinting for almost all culture positive cases identified in the US. Although DNA fingerprints of strains are used primarily to identify clustered cases and help detect recent transmissions and outbreaks, they capture information that could potentially be used to develop more advanced tracking and control measures. Classification of strains into lineages provides perspective on the phenotypic consequences of the genetic variations of MTBC strains. Phylogenetic analyses conducted using large sequence polymorphisms (LSPs) and single nucleotide polymorphisms (SNPs) of MTBC strains have helped investigate the history of evolution of MTBC and established the existence of distinct clades. Further investigations into the diversity of MTBC strains reveal differences in phenotype of strains belonging to the various clades [26, 48]. Large databases of DNA fingerprint information representing the bio-geographic diversity of strains in the MTBC population have been amassed by TB researchers worldwide. Computational methods that classify strains into these clades using DNA fingerprint data are needed to label data in these databases and that collected for routine surveillance measures.

In Section 1.2, we provide some background of the molecular methods utilized in the epidemiology of TB.

1.2 DNA Fingerprinting Methods

In this section, we present a brief description of current methods used for MTBC genotyping that are referenced in this work. Although, earlier studies found negligible genetic diversity between MTBC strains [44, 94, 75], the advent of molecular epidemiology has revealed considerable inter-strain diversity. We discuss potential applications of such methods to answer some of the questions facing TB researchers today. A more detailed discussion of the methods can be found in [106, 6, 74].

1.2.1 IS6110 Restriction Fragment Length Polymorphism (RFLP)

This method is a Southern blot hybridization technique and is the gold standard for molecular epidemiology of MTBC strains [105]. Strains are typed based on the copy number of IS6110 insertion sequences and the variability in the positions of these sequences. Strains isolated from epidemiologically linked patients are believed to have identical RFLP patterns and hence can be used to identify/verify clustered cases. The method has high discriminative power for strains with copy number greater than six. It was shown that this method can be used to distinguish closely related strains, but is not suitable to study the evolutionary history of strains, since the order of events cannot be inferred [37]. The data format makes it difficult to compare results between labs [74]. Moreover, this method involves a complicated and time-consuming process requiring sub-culturing of isolates to obtain sufficient quantities of DNA. Other Polymerase Chain Reaction-based (PCR-based) techniques have been developed to overcome these disadvantages.

1.2.2 Spoligotyping

Spoligotyping is a PCR-based reverse hybridization technique for genotyping MTBC [66]. It is a frequently used genotyping tool, and performed on all newly identified culture positive case of tuberculosis in the United States (US). The method exploits the polymorphisms in the Direct Repeat (DR) locus to distinguish strains. The DR locus contains 36-bp repeats interspersed with non-repetitive short sequences called spacers. Spoligotypes are represented as a 43-bit long binary string, constructed on the basis of presence or absence of these spacers. The portable data format facilitates easy inter-laboratory comparisons. The loss of spacers can occur due to homologous recombination, or due to the transposition of IS6110 insertion sequences [37, 72]. Since such mutations by the loss of spacers in the DR region are irreversible; spoligotypes can be used in the construction of evolutionary history. However, some caution must be used when using spoligotyping to study the evolution of strains, as convergent evolution of phylogenetically unrelated strains to a common spoligotype has been observed [108].

1.2.3 MIRU-VNTR Analysis

Mycobacterial Interspersed Repetitive Units-Variable Number Tandem Repeats (MIRU-VNTR) based analysis exploits the polymorphisms observed in a selected number of 41 identified mini-satellite like regions distributed on the chromosome of MTBC. MIRU typing is based on the number of repeats observed at each of the 12, 15 or 24 selected MIRU loci determined using a PCR-based method. Each locus was found to differ in its discriminative power and in its variability in alleles [27, 3]. Twelve, fifteen on twenty-four selected loci can be used for genotyping for epidemiological discrimination of strains [99]. It was determined that optimally 24 loci should be used to capture the genetic diversity of strains for phylogenetic studies [99]. A comparative analyses of typing methods has shown that the discrimination power of MIRU typing is higher than spoligotyping, and only slightly lower than IS6110-RFLP [100]. The present CDC standard requires spoligotyping and MIRU typing to be performed on isolates from every TB case identified in the US.

1.2.4 Single Nucleotide Polymorphisms

Single Nucleotide Polymorphisms (SNP) that characterize strains of MTBC have been identified. Non-synonymous polymorphisms (nsSNP) are changes that alter the polypeptide sequence and could possibly provide selective advantage to strains. The nsSNP are especially useful in understanding the acquisition and spread of drug-resistance in strains, even in strains that share the same DNA fingerprint [82, 25]. On the other hand, synonymous SNPs (sSNP), silent mutations that are considered to be functionally neutral, are useful in phylogenetic analyses to study the evolutionary relationships between strains

[94, 42, 5, 56, 57].

1.2.5 Long Sequence Polymorphisms

Long Sequence Polymorphisms (LSP) or Regions of Deletions (RD) play an important role in phylogenetic analyses and studies of inter-strain diversity. Specific sequences occurring in progenitor strains are preserved by all strains that have evolved from it [47, 43]. Studies have identified RDs that characterize MTBC lineages. e.g. all ancestral strains have the TbD1 sequence conserved, while it is absent in all modern strains [18]. This highly clonal structure of the MTBC population is a result of the lack of horizontal gene exchange [18, 63]. Studies have also found associations between LSPs and the pathogenicity, drug-resistance and virulence of strains e.g. RD1 [112, 36].

1.3 TB-Insight: TB-Lineage

In this section, we describe a web-based tool that classifies MTBC strains into major lineages using DNA fingerprint information. For this we culled literature to develop a coherent set of ordered rules based on observed spoligotype patterns. This tool is described in greater detail in [89]. Here we provide a summarized description.

The classification of *M. tuberculosis* complex (MTBC) strains into a phylogenetic framework has a long history. The analysis of spoligotype patterns led by Sola and Rastogi revealed the presence of related spoligotypes which have since been grouped into 62 spoligotype families or clades [19]. Early studies of sequence diversity in selected genes described three principal genetic groups which correlated well with spoligotype families [94]. The sequencing of several MTBC complex genomes revealed additional SNPs defining additional genetic groupings [42, 56]. Gagneux and colleagues used comparative genomics to identify large sequence polymorphisms defining major lineages in the MTBC [48] which were later supported by the sequencing of 89 genes from 108 strains [61]. These studies have culminated in the description of a phylogenetic framework for strain classification along with a proposal for standardized nomenclature [26] which is summarized in Table 1.1. Although SNPs and LSPs remain the gold standard for strain classification, the congruence between spoligotypes and LSPs has been reported [26, 67]. We created TB-Lineage, a suite of web tools that provide automated methods to classify MTBC strains using DNA fingerprint data into lineages as defined by SNPs based phylogenetic analyses[89, 3]. The tools also provide an additional feature: generation of spoligoforests to visualize genotype datasets augmented with lineage information determined by the classification tools.

- *Input:* (i) Spoligotype strains in binary or octal format (ii) 1 (locus MIRU24), 12, 15 or 24 loci of MIRU-VNTR (optional)
- *Functionality:*
  - Classify strains into major lineages using (i) A rule-based system that uses spoligotype and the MIRU24 locus (when available) to assign lineage labels. (ii) A Conformal Bayes Network (CBN) that uses a blend of PCR-based methods.
- URL : http://www.tbinsight.cs.rpi.edu/run_tb_lineage.html

1.3.1 TB-Lineage: TB-Rules

TB-Rules is a rule-based classification system available as part of the TB-Insight suite of tools that classifies strains of MTBC into major lineages: the *modern* lineages Euro-American, East-African Indian (CAS), East-Asian (Beijing) and the *ancestral* lineages *M. africanum* (West African 1 and 2), *M. bovis*, Indo-Oceanic. The rules were culled from prior literature on spoligotype signatures and various epidemiological studies conducted worldwide [19, 88, 39, 95, 40] and refined based on expert guidelines from the Centers for Disease Control and Prevention (CDC) [89]. Prediction is done using the spoligotype sequence and, if available, the MIRU24. The locus MIRU24 is known to correlate with the TbD1 deletion, and therefore, can be used to characterize strains as modern or ancestral [98]. However, some strains do not conform to the observed patterns in the number of repeats at MIRU24. Or in some cases, as with historical datasets, MIRU data are not available. For such cases, the tool employs a Naive Bayes classifier that uses spoligotype data alone to predict whether a strain is *modern* or *ancestral.* The prediction is then used as a surrogate for the MIRU24 locus in the rules. This system relies on certain signature deletions which are assumed to correspond to speciation events.

1.3.1.1 Spoligotype Signatures

Prior literature describes spoligotype-based visual rules for classification, as well as bioinformatics approaches that identify specific spacers and loci that can be used to differentiate strains into groups [14, 19, 29, 39, 40, 41, 88, 93]. This literature including

Table 1.1: Mapping of lineage names from conventions in prior literature.

TB-Lineage Family	LSP-based Lineages [48]	SpolDB4 Family [19]	Principal genetic group [94]	TbD1 assignment [18]
M. bovis	*M. bovis*	BOV1, BOV2, BOV3, BOV4, BOV1-variant1, BOV2-variant1, BOV1-variant2, BOV2-variant2	1	Ancient
West African 1	West African 1 (lineage 5)	Afri2, Afri3	1	Ancient
West African 2	West African 2 (lineage 6)	Afri1	1	Ancient
Indo-Oceanic	Indo-Oceanic (lineage 1)	EAI-5, EAI1-SOM, EAI2-Manila, EAI2-Nonthaburi, EAI3-IND, EAI4-VNM, EAI6-BGD1, EAI6-BGD2, EAI8-MDG	1	Ancient
East Asian	East Asian (lineage 2)	Beijing, Beijing-like	1	Modern
East-African Indian	East-African Indian (lineage 3)	CAS1-Delhi, CAS1-Kili, CAS1-variant CAS2	1	Modern
Euro-American	Euro-American (lineage 4)	T1, T1-RUS2, T2,T2-Uganda, T3, T3-ETH, T4, T4-CEU1, T5-Madrid2, T5-RUS1, Tuscany, S,X1, X2, X3, X2-variant1, X3-variant1, X3-variant2, H1, H1-variant1, H2, H3, LAM01, LAM02, LAM03, LAM04, LAM05, LAM06, LAM07, LAM08, LAM09, LAM10, LAM07-TUR, LAM10-CAM, LAM11-ZWE, LAM12-Madrid	2 and 3	Modern

descriptions of visual rules for SpolDB4 sub-families was synthesized to generate a set of observations for major lineage classification:

- Spacers 29-32 and 33-36 in the direct repeat locus may be used to determine if a strain belongs to principal genetic group 1 (PGG1) or groups 2 and 3 (PGG2/3). If spacers 33 36 are absent and at least one of spacers 29-32 is present, the strain belongs to PGG2/3. If at least one of spacers 33 36 is present, then the strain belong to PGG1. If spacers 29 36 are absent, then the principal genetic group cannot be determined on the basis of spoligotype alone.
- PGG 1 strains with spacer 3, 9, 16 and 39-43 absent are *M. bovis.*
- PGG1 strains with spacers 8, 9, and 39 absent are *M. africanum. M. africanum* strains that have spacers 8-12 and 37-39 absent belong to West African 1 lineage and strains that have spacers 7-9 and 39 absent belong to West African 2.
- PGG 1 strains with spacers 29-32 absent and spacer 34 absent belong to the Indo-Oceanic lineage.
- PGG1 strains with spacers 1-34 absent belong to the East Asian lineage.
- PGG 1 strains with spacers 4-7 and 23-24 absent belong to the East-African Indian lineage.
- PGG 2 and 3 strains belong to the Euro-American lineage.
- The number of repeated units at locus MIRU24 correlates with the TbD1 deletion. Strains with the sequence intact (ancient) will have more than one repeated unit at MIRU24 while strains with the sequence deleted will have a single repeated unit in this locus.

A coherent set of rules for lineage classification was generated on the basis of the observations listed above. The order of the rules is equivalent to checking membership in PGG 1 first. When there is a deletion from 29-36, since assignment to principal genetic groups cannot be made on the basis of spoligotype alone, the MIRU24 is used to determine if a strain is ancestral and belongs to PGG 1. If the MIRU24 data is not available the Naive Bayes classifier is used to predict if a strain is ancestral and belongs to PGG1 based on the spoligotype. Assignment to specific lineages within PGG 1 is based on MIRU24

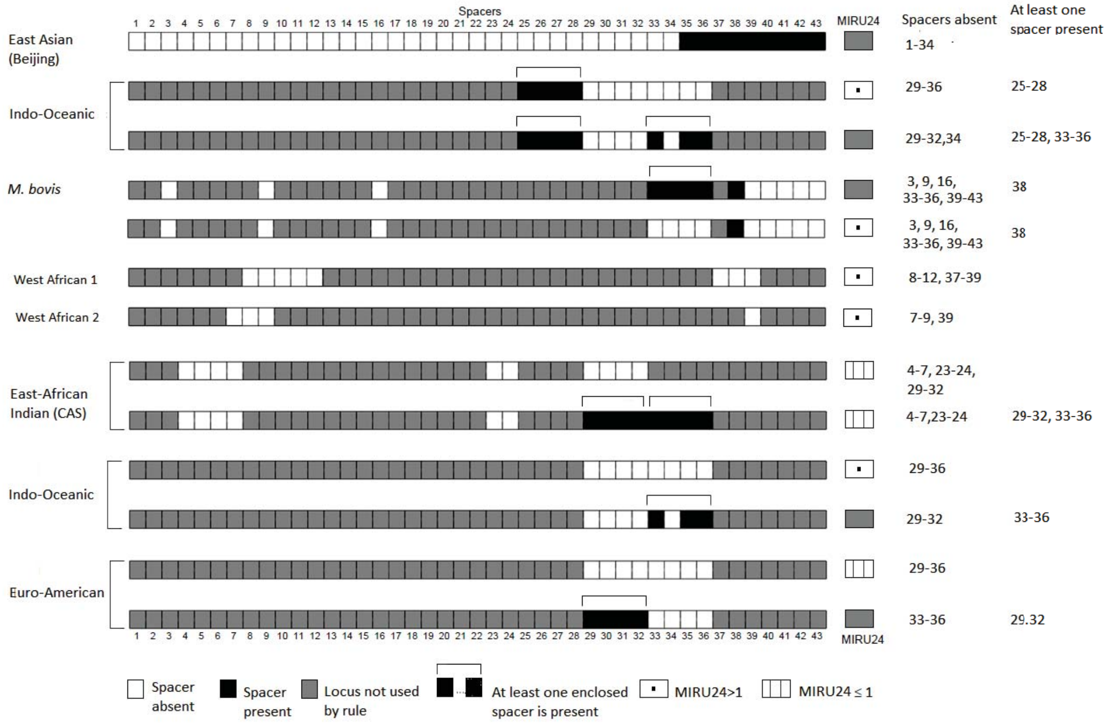


Figure 1.1: Rules to classify *M. tuberculosis* strains into major lineages based on the presence or absence of spacer sequences and the number of repeats observed at the MIRU24 locus. The rules are applied in the order specified. The label assigned to a strain corresponds to the first rule that meeting criteriais satisfied.

(or modern/ancestral prediction by the Naive Bayes classifier) and pertinent clauses for the lineages as represented in Fig 1.1.

Some rules rely on the number of repeats observed at MIRU24 locus to predict whether a strain is ancestral or modern. There is an observed correlation between the number of repeats at locus MIRU 24 and the TbD1 deletion that is used to discriminate between modern and ancestral strains [18]. Modern strains typically contain one repeat at MIRU24 while ancestral strains contain greater than one repeat [39]. However, there are a non-trivial percentage of exceptions to this rule in the CDC dataset. The number of repeats at the MIRU24 locus deviates from the expected value with a non-negligible probability for *M. bovis* and *M. africanum*. Moreover, since MIRU-VNTR typing was adopted as a national standard in 2004, MIRU-VNTR type data may not be available for earlier records. In order to overcome these limitations, a Naive Bayes classifier was developed to classify spoligotypes as ancestral or modern. The model was trained using labeled spoligotype data with spoligotypes belonging to the East Asian, East-African Indian, and Euro-American lineages assigned label modern and spoligotypes belonging to the *M. africanum, M. bovis*, and Indo-Oceanic lineages assigned label ancestral. Previously, MTBC strains have been successfully classified at varying levels of granularity on the basis of spoligotype data using Bayesian Networks [107]. Instead of the MIRU24 value that originally served as a surrogate for modern or ancestral, the modern or ancestral prediction made by the Nave Bayes classifier is substituted into the rules.

1.3.1.2 Naive Bayes Classifer to predict Modern or Ancestral

Each spoligotype is represented as a binary 12-dimensional feature vector. Each dimension represents the presence/absence of a contiguous deletion. Presence of a deletion means no spacers are present in the subsequence, while absence means at least one spacer is present in the subsequence. The evolution of the DR locus occurs via deletion of one or more contiguous Direct Variable Repeats (DVRs) with some non-negligible probability, whereas insertion of DVRs is highly unlikely [108]. The features selected were single deletions of spacers 3, 16, 8, 9, 39 and contiguous deletions of spacers 1-34, 25-28, 29-32, 33-36, 39-43, 4-7, and 23-24. These deletions are believed to be speciation events as described earlier. If at least one spacer is present in the deletion sequence it is represented as a 1, presence of the deletion is represented by a 0. These bits are concatenated to form a binary vector. Eg. Considering the deletions in the order specified, if spacers 3,16,8,9,39 and 23-24 are absent, while there is at least one spacer present in sequences 1-34, 25-28, 29-32, 33-36, 39-43, and 4-7 the spoligotype is represented as 000000000000770. The selection of deletion sequences was made on the basis of the observations listed above about the sequences that are relevant for the lineage classification task and further validated based on the information gain as described in the supplementary information.

The Naive Bayes model assumes that each feature S_d is independent given the class C_i (modern or ancestral). S_d takes value 1 if at least one spacer is present in deletion d, and 0 otherwise. The probability of the deletion d being absent (at least one spacer being present) for a strain of class i is given by p_{id}, and the probability of a deletion d being present (no spacer(s) being present) is given by $(1-p_{id})$. We assume the conditional independence of the features of variable S. The probability of occurrence of a spoligotype

S given the class C_i is therefore

$$P(S|C_i) = \prod_{d=1}^{12} (p_{id})^{S_d}(1 - p_{id})^{(1-S_d)} \tag{1.1}$$

In the test phase, strains were classified into the 2 classes modern and ancestral. The probability of a strain S belonging to class C_i was computed as

$$\begin{aligned} P(C_i|S) &\propto P(C_i)P(S|C_i)/P(S) \\ &\propto P(C_i)P(S|C_i) \end{aligned}$$

The values of p_{id} and $P(C_i)$ were calculated by counting the appropriate proportion of values in the data with Laplace smoothing to deal with deletions observed 0 times. The strain S is assigned to the class i for which its conditional probability $P(S|C_i)$ is the highest. The value of the MIRU24 locus is predicted on the basis of the modern/ancestral classification and used in place of the MIRU24 in the rules.

The accuracy of the rules was evaluated on two datasets from the CDC, with CDC2011 containing genotypes not analyzed during development, along with datasets from MIRU-VNTRplus [1], Brussels[1] and SpolDB4 [19] described in prior publications. A summarized description of these datasets is provided in Table 1.2. A high level of accuracy was reported on all datasets, with the labels assigned by the online tool matching CDC labels, LSP-based analysis or SNP-based analysis in greater than 99% of cases. A comparison of the f-measure of the rules used with spoligotype and MIRU-VNTR for each lineage applied to all the datasets is reported in Table 1.3. Near perfect f-measure, i.e. close to 1, were observed across all lineages. The rules work equally well when only spoligotypes are available.

1.3.1.3 TB-Insight: Conformal Bayes Net

This tool employs a hierarchical Bayesian network to classify MTBC strains into the major genetic lineages using different blends of PCR-based biomarkers [3]. The design of the probability-based model exploits known properties about the structure, position and mutation mechanisms of spoligotypes and MIRUs. The MIRU loci are distributed across the genome of the MTBC but mostly away from the DR locus. The assumption of independence between the MIRU loci and between the MIRU loci and the spacers is

Table 1.2: **Summary description of the datasets used in the development of the online tool TB-Lineage.**

Dataset	No. distinct spoligotypes	No. distinct MIRU	No. distinct genotypes	*M. bovis*	West African 1	West African 2	Indo-Oceanic	East Asian	East-African Indian	Euro-American
CDC	3198	5430	10828	685	65	78	5177	4829	1446	24781
CDC2011	1548	2363	3144	56	19	22	421	269	403	1948
MIRU-VNTRplus	86	120	145	11	18	11	16	10	10	87
Brussels	197	422	378	17	4	9	27	16	30	339
SpolDB4	1604	-	1589	4731	228	89	2716	3973	425	19050

Table 1.3: **Comparison of f-measure of classification in the datasets used in this study.**

Dataset	East Asian	East-African Indian	Euro-American	Indo-Oceanic	West African 1	West African 2	M. bovis
CDC	1.0	0.9935	0.9995	0.9990	0.9559	0.9873	0.9985
CDC2011	0.9926	0.9926	0.9969	0.9916	0.95	1.0	0.9907
MIRUVNTRplus	1.0	0.9524	0.9942	1.0	1.0	1.0	1.0
Brussels	0.9697	1.0	0.9985	0.9643	0.8000	1.0	1.0
SpolDB4	0.9926	0.9926	0.9969	0.9905	0.9268	1.0	0.990

made on the basis of this fact. The number of repeats at each MIRU locus is frequently between 0 and 9, and in some cases greater than 9. Therefore, each MIRU locus is modeled as a multinomial distribution with possible values 0, 1,. . . ,8, and $\geq$ 9 [2]. Spoligotypes are modeled as a multivariate Bernoulli distribution with "Hidden Parents", as in the SPOTCLUST model [107]. The CBN model exploits the correlation between the number of repeats at locus MIRU24 and the conservation of the TbD1 deletion [98] and represents this as a causal relationship. Therefore, a top-level classification into *modern* and *ancestral* strains, characterized by the absence or presence of the TbD1 deletion respectively, is done on the basis of locus MIRU24. Thus, the CBN models the distributions of the MIRU loci and the spoligotypes of various lineages, and provides a method of assigning labels for strains based on the spoligotype and/or any number of MIRU loci as summarized in the pseudocode provided below.

The independence assumption between all loci allows for predictions to be made using any number of biomarkers depending on availability. The design of the model allows it to be trained using all available data, even if the data is incomplete, i.e. not all biomarkers are available. This model is flexible and extensible to including different

program 1 Assign strain major lineage label using CBN.

```
$x \leftarrow spoligotype\ of\ strain$
$M \leftarrow MIRU\ pattern\ of\ strain$
$c_j \leftarrow Major\ lineage\ j$
$m_{11} \leftarrow P(x_d = 1|H_d = 1) \leftarrow 0.9$
$m_{10} \leftarrow P(x_d = 1|H_d = 0) \leftarrow 10^{-7}$
$m_{00} \leftarrow P(x_d = 0|H_d = 0) \leftarrow 1 - m_{10}$
$m_{01} \leftarrow P(x_d = 0|H_d = 1) \leftarrow 1 - m_{11}$
for MTBC lineage $c_j$ do
    $t1 \leftarrow \prod_{d=1}^{43}(p_{jd}m_{11} + (1 - p_{jd})m_{10})^{x_d}((1 - p_{jd})(1 - m_{10}) + p_{jd}(1 - m_{11}))^{(1-x_d)}$
    $t2 \leftarrow \sum_{l=1}^{k} P(c_l) \prod_{d=1}^{43}(p_{ld}m_{11} + (1 - p_{ld})m_{10})^{x_d}((1 - p_{ld})(1 - m_{10}) + p_{ld}(1 - m_{11}))^{(1-x_d)}$
    $P(c_j|x) \leftarrow \frac{P(c_j)t1 \prod_{i \in MIRU} P(M_{ir}|c_j)P(c_j|M_{24})P(M_{24})}{t2 \prod_{i \in MIRU} P(M_{ir}|c_j)P(c_j|M_{24})P(M_{24})}$
    {$p_{jd}$ is the probability that spacer d is present given class $c_j$}
    {$P(M_{ir}|c_j)$ is the probability of having $r$ repeats at MIRU locus $i$ given class
    $c_j$, where $r$ is the number of repeats present at locus $i$ in user strain}
    {$P(M_{24})$ is the probability of having less than 2 repeats at locus MIRU24}
end for
$lineage \leftarrow argmax_{c_j} P(c_j|x)$
```

types of biomarkers as they become available. Host-related characteristics and risk factors may also be included into such a model, as suggested in [53]. Hybrid approaches that incorporate a rule base into a Bayesian Network model have been developed recently that benefit from the advantages of both rule-based and probabilistic approaches [4].

Thus, classification of MTBC strains into lineages provides insight into the genetic diversity of the strains being investigated and helps identify the predominant genetic groups in a population. Further, strains associated with different lineages have been found to vary in their immunogenicity, pathogenicity, virulence, transmissibility and drug susceptibility [104, 83, 47, 48]. The observed associations between clades previously identified by phylogenetic analysis and geographical regions indicate the influence of social factors such as migration, and other host-related factors on disease dynamics. TB-Lineage provides a suite of methods to classify strains into these groups using only the DNA fingerprint data, collected as part of routine TB surveillance.

1.4 Spoligoforests: Validation of Lineages

Spoligoforests are a visualization of the number of occurrences of strains, their distribution by lineage, and potential evolutionary relationships between strains. The visualization of spoligotypes provided by TB-Vis builds on the design of spoligoforests [84], wherein each node represents a unique spoligotype and each edge between the parent spoligotype and the child represents a putative mutation event. The spoligoforest created based on the genotypes in the CDC dataset is shown in Figure 2. Each lineage corresponds to a unique color. A node represents a cluster of strains of the same spoligotype but different MIRU types, and is assigned a color based on the lineage to which the spoligotype belongs. Node sizes are representative of the number of occurrences of strains of that spoligotype in the dataset on a log scale (base 2). Edges indicate potential evolutionary relationships resulting from a contiguous deletion of spacers and changes in the number of repeats at loci of MIRU-VNTR types associated with the spoligotype, and are thus an indication of the relatedness of strains.

However, the spoligoforest does not visualize all the information we need to understand the genetic diversity in the dataset. There is a need to preserve proximity relations between nodes i.e. the distances between nodes in the visualization must match the genetic distances. This goal introduces new challenges such as edge-crossings and node-overlaps that obscure the underlying relationships in the dataset. We investigate background work pertaining to this problem in Chapter 3 and our approach in Chapter 4.

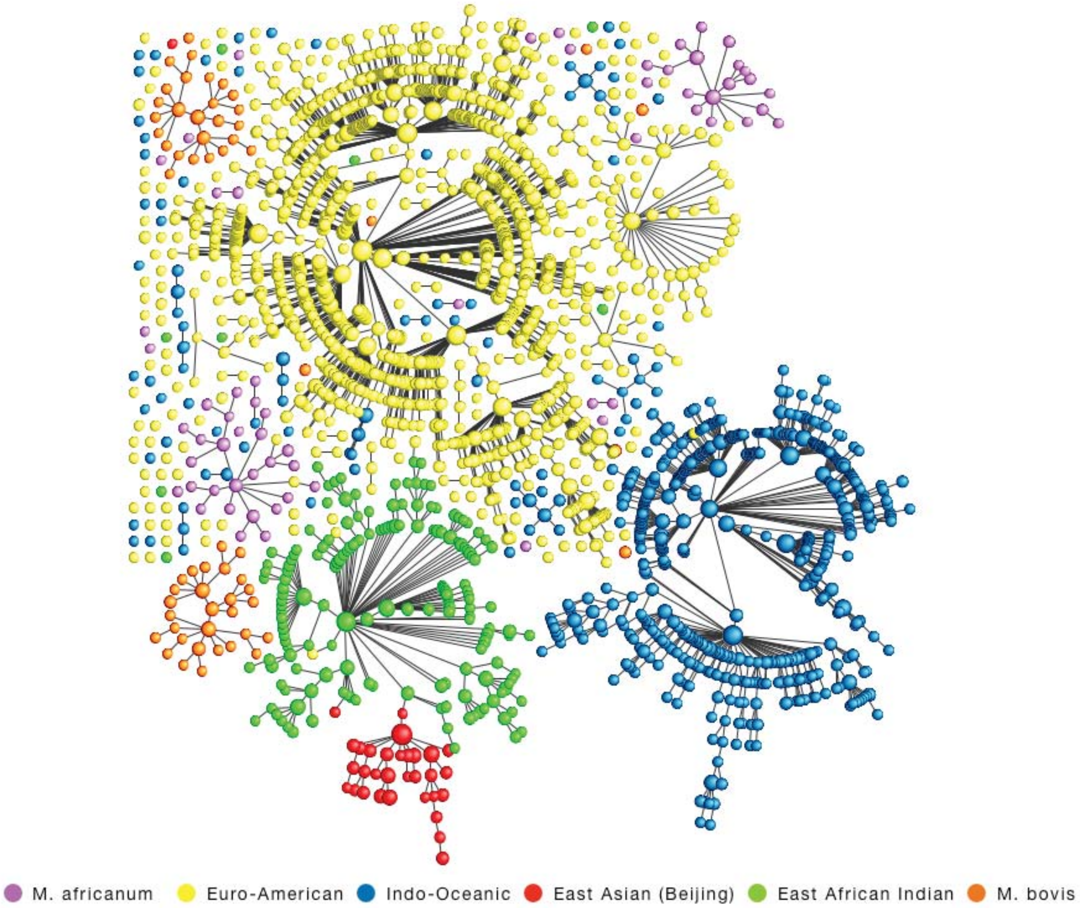


Figure 1.2: Spoligoforest representation of genetic diversity of MTBC strains in 37,061 isolates collected from TB patients in the United States between 2004-2008. Each lineage corresponds to a unique color as shown in the legend. Each node represents a cluster of strains of the same spoligotype but different MIRU types, and the size represents the number of isolates on a log scale. The lineages are highly cohesive with few edges between lineages.

CHAPTER 2
Classification and Visualization Tools for TB

In this chapter, we survey publicly available classification and visualization tools designed to use molecular epidemiological data to extract information that can be employed for the effective tracking and control of tuberculosis (TB) . The application of molecular methods for the epidemiology of TB complement traditional approaches used in public health. DNA fingerprinting methods are now routinely employed in TB surveillance programs and are primarily used to detect recent transmissions and in outbreak investigations. Here we present web tools that facilitate systematic analysis of *Mycobacterium tuberculosis* complex (MTBC) genotype information and provide a view of the genetic diversity in the MTBC population. They provide an integrated platform for researchers to use molecular epidemiological data to address current challenges in the understanding of TB dynamics and the characteristics of MTBC.

This chapter contains relevant extracts from the survey paper [90] on 'Web Tools for Molecular Epidemiology of Tuberculosis'. A summary of all tools surveyed in this paper are in the companion website at http://www.tbinsight.cs.rpi.edu/molepisurvey.html. Throughout this paper, we utilize the surveillance data obtained from the New York State Department of Health (henceforth referenced as NYS), comprised of spoligotype and MIRU type information of MTBC strains from patients diagnosed during the period 2004-07. The NYS dataset is comprised of 674 isolates: 268 distinct spoligotypes, 361 distinct MIRU types and 500 distinct RFLP patterns. This genotype information augmented with expert-assigned major lineage labels is used to explore and test the various tools presented.

In Section 2.1, we analyze various classification models. Phylogenetic analyses have shown that MTBC strains may be classified into related genetic groups using various biomarkers. We look at some tools that can classify strains efficiently using only the DNA fingerprint, and will help in the investigation of phenotypic characteristics shared by strains within each lineage.

In Section 2.5, we cover visualization methods that represent surveillance data in

This chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Web tools for molecular epidemiology of tuberculosis*, Infect. Genet. Evol., 12 (2012), pp. 767-781.

ways that help study the diversity in strain and host populations. These can reveal unobserved epi-links and help identify typical as well as anomalous associations between strain and host groups.

2.1 Classification Tools for TB

Classification of MTBC strains into lineages provides insight into the genetic diversity of the strains being investigated and helps identify the predominant genetic groups in a population. Further, strains associated with different lineages have been found to vary in their immunogenicity, pathogenicity, virulence, transmissibility and drug susceptibility [104, 83, 48, 47]. The observed associations between clades previously identified by phylogenetic analysis and geographical regions indicate the influence of social factors such as migration, and other host-related factors on disease dynamics [48]. We need methods to classify strains into these groups using only the DNA fingerprint data, collected as part of routine TB surveillance.

Although visual rules and nearest neighbor based approaches performed on highly curated databases can be used to classify strains, these do not constitute scalable solutions. Visual methods may involve considerable human effort and nearest neighbor approaches may involve a large number of pairwise comparisons. To address this problem, computational methods for lineage classification have been developed. These tools complement existing visual rules. Classification results will help recognize variations in phenotypic characteristics of lineages. In this section, we present some web-based automated tools that allow classification to be performed easily and efficiently on large datasets. Such tools can provide perspective on differences in phenotypic characteristics, and phylogeographic associations of MTBC strains with host populations.

2.2 MIRU-VNTR*plus*

MIRU-VNTR*plus* is a nearest neighbor based classification tool for MTBC strains. It relies on a highly curated database comprising of the detailed profiles of 186 strains representing all MTBC lineages [1]. The website provides access to the LSP, SNP, IS6110 RFLP fingerprint, spoligotype, 24-locus MIRU profile, and drug resistance profile of these reference strains as well as species, lineage, and epidemiologic information pertaining to these strains. The website provides additional tools that facilitate analysis of user strains with respect to these reference strains.

- *Input:* User strains comprising of one or more of the following fields in specified format:
  (i) MIRU-VNTR type (ii) Spoligotype (iii) RD (iv) SNP (v) Susceptibility
- *Functionality:*
  - Compare user strains with reference strains
  - Identify strain lineages by similarity search
  - Construct phylogenetic trees using neighbour-joining (NJ) or Unweighted Pair Group Method with Arithmetic Mean (UPGMA) algorithms
  - Helps establish universal nomenclature by facilitating assignment and querying of MtbC15-9 codes
- URL: http://www.miru-vntrplus.org

The identification program assigns species labels, lineage labels, SpolDB4 lineage labels [19] and RD Gagneux's lineage labels [47] to user uploaded strains. Identification by similarity search is a best-match approach which employs a nearest-neighbor or parsimony-based analysis to find reference strains whose genetic profiles most closely match the user strains. Genetic relatedness between strains is determined based on one of the following genetic distance measures provided.

- Categorical Distance *(default for all markers)*: This is the normalized sum of the number of markers that have different alleles.
- Chord distance D_C *(MIRU-VNTR only)*: This is a geometric interpretation based on the "angular distance" between the two strain groups determined as the sum of square root of frequencies of each allele observed in both strain groups [21].

  The following two distance measures are especially applicable for tandem repeat loci following a stepwise mutation model (SMM), such as the MIRU-VNTR. The SMM assumes that at each mutation step, microsatellites only gain or lose one repeat.
- $(\delta_\mu)^2$: This is the average of the difference in repeat numbers of alleles at all loci [54].
- D_{SW} (stepwise weighted distance): This measures the probability that two alleles are different for two different strain groups, weighted by the absolute value of the difference in the number of repeats for the two strains [92].

Lineage labels may be assigned to a strain if there exists one or more matching reference strains exist that are within the user-defined distance cut-off. The cut-off specifies the accepted tolerance in finding the closest match. This choice needs to balance a trade-off between choosing a large value to reduce the effect of noise, such as erroneous or irrelevant markers, and a small value, for higher specificity in lineage identification. Additionally, better sensitivity and specificity were reported when multiple markers are used in conjunction as opposed to a single biomarker. These genetic distances (normalized over the number of loci in each biomarker) may be weighted based on user discretion. The tool provides default values based on heuristics to guide the user to getting best results in strain identification.

Users can construct a phylogenetic tree to inspect the genetic relatedness of strains using the neighbor-joining (NJ) algorithm (recommended by [1]) or the Unweighted Pair Group Method with Arithmetic Means (UPGMA). Single or multiple markers may be used to determine the distance between strains, as with the similarity search. Reference database strains can optionally be included in the tree calculation. The tool guidelines recommend that the tree be rerooted using an *M. cannettii* strain. Strains may be colored based on user input or automatically using fields such as species or lineage. The tree-based identification feature can also be used to refine labels assigned by similarity search. The phylogenetic tree can be viewed and saved in various file formats, and may also be embedded alongside the strain information. The *Calculate-Tree* feature helps view the inferred evolutionary relationships between strains in the dataset based on various distance measures as well as additional information incorporated from the reference dataset. In the following subsection, we illustrate the functionality of this tool by analyzing genotype data in the NYS dataset.

Analysis of NYS dataset using MIRU-VNTRplus: 500 isolates genotyped by spoligotyping and MIRU typing from the NYS dataset were uploaded, and 435 of these were assigned labels using the similarity search tool. Figure 2.1 represents a tree in radial format created from this dataset using the NJ algorithm. In most cases, the strains that were not labeled by similarity search, can be easily identified based on the other user and reference strains belonging to the same subtree in the phylogenetic tree. From Figure 1 it can be seen that the Euro-American lineage comprising of sub-lineages such as LAM (77 strains), Haarlem (101 strains) and X (57 strains) are the most prevalent in the NYS dataset. This is in accordance with the previously observed stable associations between

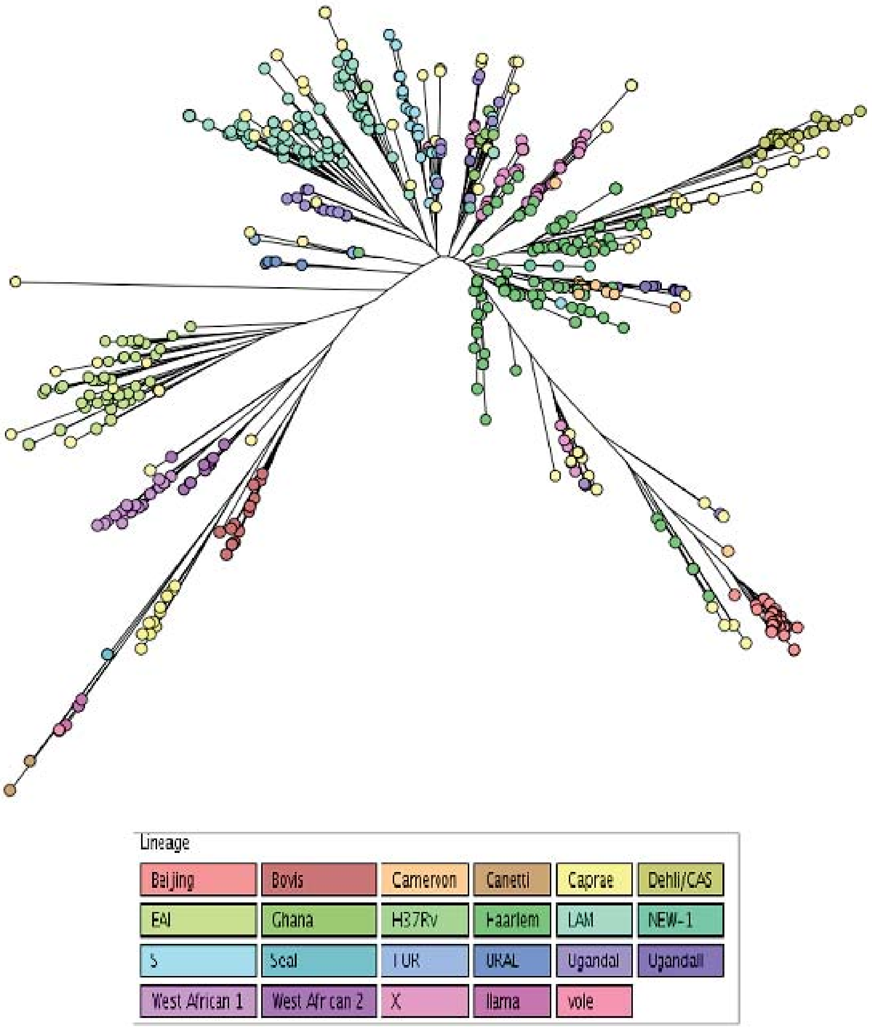


Figure 2.1: NJ tree in radial format created from the NYS dataset using the tree-based identification tool on MIRUVNTRplus.org. Lineage labels were assigned using similarity search, followed by tree-based analysis. The strains were assigned colors based on their lineage using the options available on the Calculate Tree tool. The scale of the genetic distance and the colors associated with each lineage are indicated in the legend.

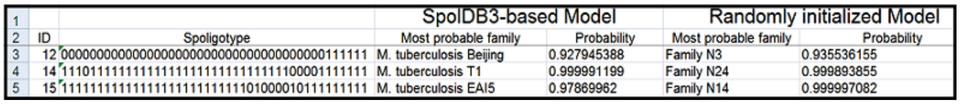
host and MTBC populations [63], and hypotheses of host-pathogen co-evolution [61].

2.3 TB-Insight: SPOTCLUST

SPOTCLUST is a mixture-model-based approach to determining MTBC lineages of spoligotypes [107]. The MTBC lineages predicted by the tool correspond to highly cohesive genetic groups identified by the two different probability models (i) the 36-component SpolDB3 based Model [40, 41], and (ii) Randomly Initialized Model (48 components).

- *Input:* Spoligotype strains in binary or octal format
- *Functionality:*
    - Determine MTBC lineages of the spoligotype strains using (i) SpolDB3 model (ii) Randomly Initialized Model. Results also include the probability that the spoligotype belongs to the lineage which indicates the confidence of the model.
- URL: http://www.tbinsight.cs.rpi.edu/run_spotclust.html

The two models are created differently, using different number of clusters k and initial class probabilities. Randomly Initialized Model (RIM) used Monte Carlo Cross Validation to determine an initial value of the number of clusters, while the SpolDB3 model is based on findings in [40]. The *families* identified by the SpolDB3 model closely match the lineages defined in SpolDB3 [40]. In both models, spoligotypes are represented as multivariate Bernoulli distributions[107]. The probability of the presence of each spacer is a parameter of the model, and each of the 43 parameters are treated as conditionally independent. Expectation Maximization (EM) was used to find maximum likelihood estimates of the model parameters: the class probabilities and the 43-variable Bernoulli distribution (the conditional probabilities of the presence of the 43 spacers given the class). Both SPOTCLUST models take into account the fact that spacers are lost but rarely gained. To do so, the models use hidden variables called ‘Hidden Parents‘, whose ‘children‘, the observed strains, may lose a spacer with a small but non-negligible probability. The probability of a child gaining a spacer however, is extremely small. A Naive Bayes classifier determines the most likely class label c_j for a spoligotype x, as the one with highest $P(c_j|x)$. The $P(c_j|x)$ is determined for each lineage based on the product of probabilities of presence or absence of spacers given the class (as defined by the model) while accounting for the Hidden Parent model.



1			SpolDB3-based Model		Randomly initialized Model	
2	ID	Spoligotype	Most probable family	Probability	Most probable family	Probability
3	12	0000000000000000000000000000000000000111111	M. tuberculosis Beijing	0.927945388	Family N3	0.935536155
4	14	1110111111111111111111111111111100001111111	M. tuberculosis T1	0.999991199	Family N24	0.999893855
5	15	1111111111111111111111111101000010111111111	M. tuberculosis EAI5	0.97869962	Family N14	0.999997082

Figure 2.2: Most probable family using RIM and SpolDB3 model of SPOTCLUST. The probability alongside the lineage prediction is a measure of the confidence of the prediction.

Analysis of NYS dataset using SPOTCLUST: Based on the correspondence between SpolDB3 lineages and the major lineages, 99.25% of the labels assigned to the NYS dataset by SPOTCLUST were correct. A sample output for 3 strains from the NYS dataset obtained by running through SPOTCLUST is provided in Fig. 2.2.

2.4 TB-Insight: CBN

TB-Insight is a set of web tools that provide automated methods to classify MTBC strains and visualize genotype datasets [89, 3].

- *Input:* (i) Spoligotype strains in binary or octal format (ii) 1 (locus MIRU24), 12, 15 or 24 loci of MIRU-VNTR (Optional)
- *Functionality:*
    - Classify strains into 6 major lineages using (i) A rule-based system that uses spoligotype and the MIRU24 locus (when available) to assign lineage labels. (ii) A Conformal Bayes Network (CBN) that uses a blend of PCR-based methods.
- URL: http://www.tbinsight.cs.rpi.edu/run_tb_lineage.html

This tool employs a hierarchical Bayesian network to classify MTBC strains into the major genetic lineages using different blends of PCR-based biomarkers [3]. The design of the probability-based model exploits known properties about the structure, position and mutation mechanisms of spoligotypes and MIRUs. The MIRU loci are distributed across the genome of the MTBC but mostly away from the DR locus. The assumption of independence between the MIRU loci and between the MIRU loci and the spacers is made on the basis of this fact. The number of repeats at each MIRU locus is frequently between 0 and 9, and in some cases greater than 9. Therefore, each MIRU locus is modeled as multinomial distributions with possible values 0, 1,. . . ,8, and $\geq$ 9 [2]. Spoligotypes are modeled as a multivariate Bernoulli distribution with 'Hidden Parents', as in the SPOTCLUST model [107]. The CBN model exploits the correlation between the number of repeats at locus MIRU24 and the conservation of the TbD1 deletion [98] and represents this as a causal relationship. Therefore, a top-level classification into *modern* and *ancestral* strains, characterized by the absence or presence of the TbD1 deletion respectively, is done on the basis of locus MIRU24. Thus, the CBN models the distributions of the MIRU loci and the spoligotypes of various lineages, and provides a method of assigning labels for strains based on the spoligotype and/or any number of MIRU loci.

The independence assumption between all loci allows for predictions to be made using any number of biomarkers depending on availability. The design of the model allows it to be trained using all available data, even if the data is incomplete, i.e. not all biomarkers are available. This model is flexible and extensible to including different types of biomarkers as they become available. Host-related characteristics and risk factors may also be included into such a model, as suggested in [53]. Hybrid approaches that incorporate a rule base into a Bayesian Network model have been developed recently that benefit from the advantages of both rule-based and probabilistic approaches[4].

Analysis of NYS dataset using TB-Lineage: The CBN and TB-Rules were tested on the NYS dataset. All the lineage labels assigned by TB-Rules matched the labels assigned by human experts at the CDC exactly. The CBN classified 673 isolates correctly, and misclassified a single strain, ST 2, as East-Asian instead of Euro-American. This misclassification occurs since this spoligotype sequence, 000000004020771, closely matches the signature of Beijing strains. The error can possibly be explained by the fact that the Bayesian Network uses individual spacers as features, and this does not account for the fact that one or more adjacent spacers may be deleted in a single mutation event. Such errors can potentially be avoided by using contiguous deletions as features as in [89, 14, 79], instead of the individual spacers.

Other computational approaches have also been applied to accomplish classification of strains of MTBC. Data mining approaches have been specified in [88, 39] that apply decision tree based approaches to classify strains. Feature selection was performed on the basis of information gain to identify the smallest subset of loci that need to be investigated to classify strains. In addition to web tools for classification of strains into major lineages, there is a need for tools that will perform automatic classification of strains into sub-lineages, i.e. genetic groups of finer granularity. Existing methods can be extended for this purpose by incorporating expert knowledge based on previously defined visual rules for sub-lineage classification [19, 95].

2.5 Visualization Tools for TB

Visualization of public health data is emerging as a popular aid to traditional methods of epidemiology. Modeling and visualizing genetic relatedness and patterns of mutation over relatively short periods of time are crucial for epidemiological studies as they help analyze recent transmission trends. Identifying previously unrecognized epi-links and associations between patient and strain groups helps focus public health efforts in an effective manner. In this section, we look at web-based tools that address the need for such visualizations and discuss other possibilities that could help further the application of visual analytics for TB epidemiology.

2.6 Spoligoforests

- *Input:* (i) Spoligotype strains in binary or octal format (ii) 12, 15 or 24 loci of MIRU-VNTR
- *Functionality:*
  - Draw spoligoforest colored by lineage (obtained from TB-Lineage) depicting genetic diversity and relatedness of strains in dataset.
- URL: http://www.tbinsight.cs.rpi.edu/run_tb_vis/spoligoforests.html

TB-Vis provides a visualization tool based on spoligoforests designed by [84] that depicts the genetic diversity in the MTBC strain population by lineage and the possible evolutionary relationships between strains. Figure 2.3 represents a spoligoforest constructed from the NYS dataset using TB-Vis. The MTBC strain population is depicted in the form of a forest of radial trees in which each node represents a distinct spoligotype that may be associated with one or more MIRU types. The sizes of the nodes represent the number of distinct MIRU types associated with the spoligotypes and are an indication of the inter-strain genetic diversity. An edge represents a possible mutation. The children of each node i, thus represent the strains that i can mutate into. Spoligotype mutation is modeled by deletion of one or more adjacent spacers, whereas a change in the number of repeats at a MIRU locus is regarded as a single mutation event for the MIRU type. The evolutionary relationships between strains is modeled based on genetic distances between MIRU patterns and spoligotypes of strains. Each strain may have multiple candidate parents. A single parent is chosen for each strain based on a comparison of the genetic distances between each parent-child pair. The following distance measures are used to select the most likely parent from amongst the candidates generated: (i) Hamming distance between MIRUs (the number of loci in which the two MIRU types differ) (ii) Hamming distance between spoligotypes (the number of spacers in which the two spoligotype sequences differ) (iii) Euclidean distance between MIRUs (root sum of squared differences in the numbers of repeats at each MIRU locus of the two strains).The nodes are colored based on the lineage identified by TB-Insight (TB-Rules). Thus, the visualization provides insight into the relatedness of strains based on the spoligotype and MIRU type of strains, as well as a view of the distribution of strains by lineage.

2.7 Host-Pathogen Treemaps

- *Input:*
  - Genotype information which may include (i) Spoligotype strains in binary or octal format (ii) 12, 15 or 24 loci of MIRU-VNTR (iii) RFLP (iv) SNP
  - Patient's continent of birth
- *Functionality:*
  - Draw host-pathogen treemap to visualize trends in associations between strain and patient groups, and identify anomalies.
- URL: http://www.tbinsight.cs.rpi.edu/run_tb_vis/treemaps.html

Host-pathogen maps available at TB-Vis, provide a graphical representation of strain and patient associations. Patients are represented as nodes within the nested boxes depicting strains. The visualization depicts each strain by telescopic boxes depending on the number of biomarkers uploaded. In Figure 2.4, the nested boxes represent the spoligotype, MIRU type and RFLP pattern, respectively. Other biomarkers such as SNPs may also be used. Patient characteristics such as birth-place are represented by color coding the nodes by continent of birth. This visualization provides a means of tracking trends in transmissions between patients infected with the strains of interest. It can help reveal

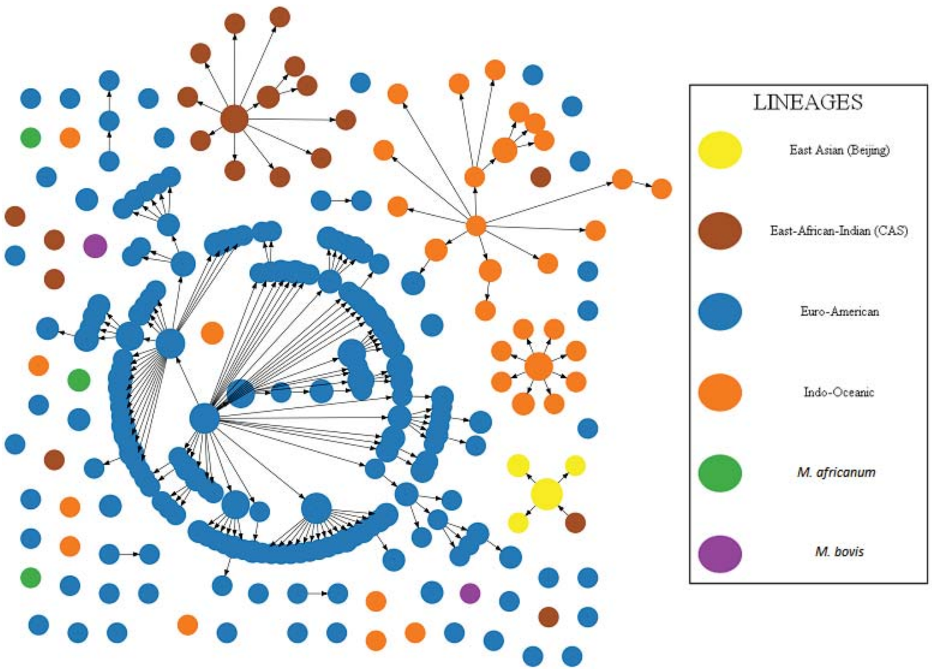
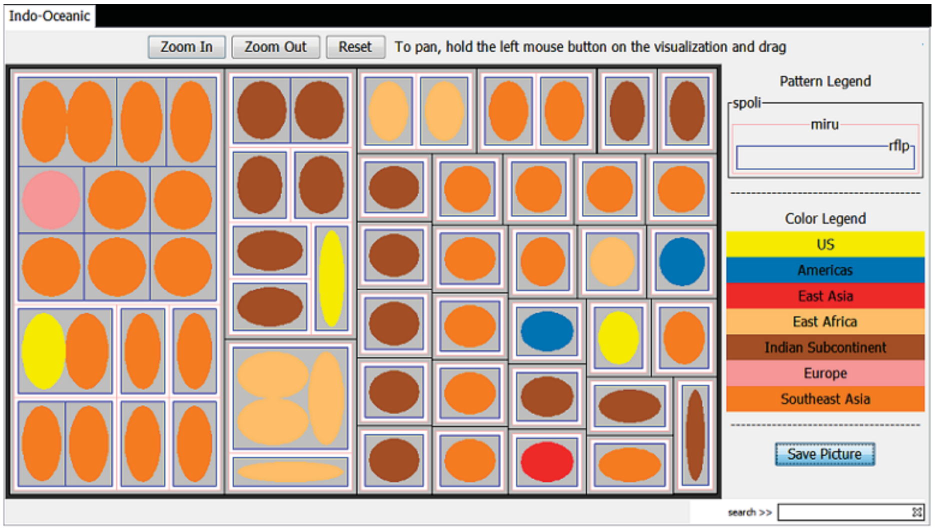


Figure 2.3: Spoligoforests of 268 distinct spoligotype strains from the NYS dataset of 674 isolates generated using the visualization tool of TB-Lineage. Each lineage corresponds to a unique color as shown in the legend. Each node represents a cluster of strains of the same spoligotype and the node size represents the number of isolates on a log scale. Each edge represents a mutation from a parent spoligotype sequence to the child sequence by the loss of one or more adjacent spacers i.e. a contiguous deletion. Note lineages are highly cohesive with few edges between lineages. This indicates the high degree of genetic relatedness between strains within a lineage.

previously unrecognized epidemiological links between patients. Anomalous behavior of strain groups can also be identified. Epidemiological investigations require the investment of significant time and resources. Therefore, identifying suspicious clusters using such visual tools will help towards the efficient allocation of efforts for case investigations.

The host-pathogen maps are based on the design of treemaps [91]. Hence, the use of nested boxes to depict strains is well-suited to capture the inherent hierarchical relationship between biomarkers used for MTBC genotyping arising out of differences in their discriminative abilities [69, 70]. The efficient use of space by treemaps allows a big-picture view of a large number of strains, and thus enables identifying typical and anomalous behavior of a strain with respect to the others in the study. This could lead



To pan, hold the left mouse button on the visualization and drag

Figure 2.4: Host-pathogen maps of patients from the NYS dataset infected with strains of the Indo-Oceanic lineage that visualize associations between the genotype and host characteristics. Strains are represented by triples of spoligotype, MIRU and RFLP patterns and are depicted by nested boxes. Patients are depicted as nodes colored by region of birth. The visualization shows the predominance of strains of the Indo-Oceanic lineage in patients from South-East Asia and the Indian subcontinent. Clusters of cases with identical associated genotype appear in bigger boxes, thus bringing attention to possible outbreaks.

to the identification or prediction of outbreaks. The treemaps are interactive enabling the user to search for and zoom in on strains of interest. Thus, treemap-based host pathogen maps provide a compact overview of the patient-strain associations, represent transmission trends and help in the identification of exceptions in these trends.

While these existing programs can help epidemiologists make more informed decisions, there is much scope for the application of information visualization to develop tools for molecular epidemiology. There is the need to incorporate time in order to visualize TB dynamics. Interactivity in visual representations can offer the ability to filter and zoom in on individual or groups of strains and/or patients and view relevant statistics. Graph visualizations can be used to represent social networks inferred from epidemiological investigations. The application of visual analytics is a promising new direction for TB epidemiology.

CHAPTER 3
Graph Visualization Background: Proximity Preservation and Crossing Minimization

The quality of a graph visualization can be gauged based on how easily it can be understood and interpreted. A drawing for graph G(V,E) is an embedding of the graph in two dimensional space. Each vertex $v \in V$ is represented as a distinct point, and each edge $e \in E$ is depicted by a line segment or an arc between the two nodes. If two arcs share a common point, we say there exists a crossing at this point of intersection. Studies have shown that reducing the number of edge-crossings is the most important aesthetic for graph drawing [81]. Another important requirement of a visualization is preservation of proximity relations. Proximity relations between nodes must be faithfully represented so that the visualization conveys correct information about the relatedness or nearness of nodes and preserves the “mental map” that the user may already have based on relative dissimilarities and local structure. In this chapter, we describe these two (sometimes contradictory) goals of graph embedding. In section 3.1, we describe various strategies employed for crossing minimization in graphs, and in 3.2 we list several families of techniques employed for depicting pairwise proximities in visualizations.

3.1 Minimizing Edge Crossings

Generating an *optimal* drawing with the least number of crossings is called the *crossing minimization* problem. By reduction of the *crossing minimization* problem to the optimal linear arrangement problem in [52], it was shown this problem is NP-hard. The minimal number of crossings for a graph is known as the *crossing-number*, $cr(G)$. The *crossing minimization* problem and many of its variants are an important class of problems in graph theory [35]. One such variation is determining rectilinear crossing-number,

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, AND K. P. BENNETT, *Crossing minimization within graph embeddings*, arXiv preprint arXiv:1210., (2012).
(2) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Preserving proximity relations and minimizing edge-crossings in high dimensional graph visualizations*, IEEE Symposium on Large Data Analysis and Visualization (LDAV), Providence, RI, 2011, pp. 131-132.
(3) A. SHABBEER AND K. P BENNETT, *Proximity preservation and crossing-minimization for graph embedding*, in The Snowbird Learning Workshop, 2012.

$rcr(G)$, the minimum crossing-number for a graph if the edges must be straight lines. This problem is also NP-hard. By Fàry's theorem, [38], if a planar embedding exists for a graph, then the graph can also be drawn using straight-line edges and no crossings. For general graphs though, $cr(G) \leq rcr(G)$. No polynomial time algorithm exists to determine the optimal embedding of a graph with minimal number of crossings. Some restricted versions of the crossing minimization problem such as 2-layer crossing minimization are also NP-hard [65], however the reduced flexibility in layout makes solutions to these problems easier in practice. Often however, there is a need for even greater flexibility in placement of nodes such as when we want to optimize with respect to some other embedding objective e.g. preserving proximity relations.

In addition to being a problem of great theoretic value and relevance, minimizing edge-crossings has much direct practical value in generating graph drawings. Satisfying this objective makes a clear and effective representation of underlying structure thus reducing cognitive load [81]. [1] The focus of this work is on generating graph layouts that satisfy such aesthetic and functional requirements that arise in practice. In this background section, we cover exact-crossing minimization techniques and planarization techniques, the two main families of algorithms for minimizing crossings. We also briefly describe planar embedding techniques for planar graphs in section 3.1.3.

The exact-crossing minimization problem is extremely hard, and therefore has long computation times. Planarization techniques on the other hand have relatively fast running times. They are directly applicable to straight-line drawings. And since graph drawings with straight lines and minimum edge-bends are desirable for understanding and interpretation of graphs, they are very useful for generating graph drawings in practice. However, planarization techniques are complex to implement. However, both classes of algorithms do not allow for much flexibility in placement of nodes implying limited additional constraint satisfaction capability such as preserving proximity relations. Additionally, the topological transformations involved alter the user's mental map of the data that may be based on local structure or relative proximities.

3.1.1 Exact Crossing Minimization

While determining the exact minimum number of crossings is an NP-Hard problem, it is also a very difficult problem in practice. There exist some algorithms that

[1] It is also of interest in the computation of VLSI layouts, as every crossing of wires between transistors on the chip leads to increase in cost.

can determine the minimum number of crossings for graphs that are sparse and have a relatively small number of nodes(most recently upto a 100 nodes). For dense graphs, the problem is increasingly difficult. In fact, the minimum number of crossings for complete graphs with even a small number of nodes is unknown. The Rectilinear Crossing Number Project has reported the rectilinear crossing numbers for K_n for $n \leq 17$ http://www.ist.tugraz.at/staff/aichholzer/crossings.html. Here we describe Ordering-based Optimal Crossing Minimization (OOCM), a recent successful integer linear programming (ILP) approach that can find a provably optimal layout for sparse graphs with $|V| \leq 60$ under an hour. However, as the size of the graph increases the percentage of graphs for which the optimal layout can be found decreases. For example, for a graph of hundred nodes, the percentage of graphs solved to optimality is 50%.

The OOCM method formulates crossing minimization as an ILP. The method uses a Branch-and-Cut based algorithm that exploits known heuristics for efficient computation. The algorithm models edge-crossings using indicator variables. For each pair of edges, the corresponding indicator variable is set to 1 if the edges cross and set to 0 otherwise. The crossing number polytope defined using these constraints in addition to Kuratowski constraints [71] constitutes the feasible region.

Since, there are a large number of constraints involved, for efficiency the authors propose the strategy of using a subset of the constraints to obtain a solution to the LP-relaxation to the problem. The fractional solution obtained is used to identify violated constraints not included in the initial iteration. Linear-time planarity-testing routines that generate Kuratowski subdivisions of a graph as certificates of non-planarity are used to find violated Kuratowski subdivisions. This guides the inclusion of only as many constraints as necessary in the description of the feasible polytope. If the resulting solution after incorporation of all violated constraints is still not integer feasible, a branching strategy is employed. Subproblems are generated based on setting one variable. The reduced subproblem is solved in lieu of the original problem.

Thus, this ILP approach defines a set of constraints that can be used towards describing the crossing number polytope and a strategy to incorporating these constraints to develop an optimal solution to the crossing-number problem. However, exact crossing-minimization is a difficult problem and this process of determining a solution is time-consuming. In practice, heuristics based approaches as described in section 3.1.2 are frequently used.

3.1.2 The Planarization Approach

In practice crossing minimization is often achieved by inexact approaches such as the planarization approach [58]. The approach was first introduced in [7]. It is a two-step, often heuristics-based, strategy. First, a planar subgraph of the original graph is computed, followed by reinsertion of edges that introduce as few crossings as possible. There are a variety of existing methods to solve each stage. Determining the maximum planar subgraph is NP-hard. Instead of solving the optimum subgraph problem e.g. using an exact branch-and-bound method [64], heuristics are often employed. A possible strategy for obtaining a planar subgraph is as follows: beginning from an empty graph, one edge is added at a time, and a (polynomial-time) planarity test is performed. If the addition of the edge causes a crossing, the edge is removed.

3.1.3 Planar Embedding

A *planar drawing* of a graph is an embedding in two-dimensional space without any edge crossings. Testing whether a graph is planar and finding a planar embedding for a graph can be achieved in polynomial time. These polynomial time algorithms can be broadly categorized as cycle-based and vertex-addition algorithms [101]. Cycle-based embedding algorithms are based on the observation that a if a cycle exists, the graph can be embedded such that the cycle causes two connected regions. In order to obtain a planar drawing any other connected parts not including the cycle must be placed inside one of the two regions completely. Note also that acyclic graphs e.g. forests are planar as well. The other class of planar embedding algorithms, vertex-addition based algorithms as in [16], begin with a subgraph of the original graph and instead of reintroducing edges, vertices are re-added one at a time. At each step, planarity testing algorithms are performed.

Another related graph drawing problem is that of generating planar grid embeddings that maps vertices to points on a grid, and edges to grid paths. A popular graph drawing technique devised in [102], involves a linear time planar grid embedding algorithm that also minimizes edge-bends (another important aesthetic criteria). The algorithm involves generating a visibility representation (a mapping from vertices to horizontal segments and edges to vertical segments), followed by transformation from the visibility representation to an orthogonal embedding by substituting vertical segments with predefined structures based on heuristics. Further transformations based on heuristics are performed to obtain a bend-minimized orthogonal representation.

3.2 Proximity Preservation

It is desirable that proximity relations between nodes in the graph are represented faithfully in the visualization. Proximity denotes nearness and may be defined/measured in a variety of ways. In this section, we describe some methods that strive to preserve proximity relations.

3.2.1 Multidimensional Scaling

Multidimensional Scaling (MDS) is a technique used for the analysis of similarities and distances between a set of objects. It is used in a variety of applications, dimensionality reduction, information visualization and analysis, and graph drawing to name a few. Metric MDS can be described as a search for the co-ordinates of points embedded in a reduced (usually Euclidean) space. The objective requires the points in this reduced space to represent the original pairwise dissimilarities between the objects in the original space[28]. A variation to this dimensionality reduction task, non-metric MDS, is the search for an embedding in Euclidean space such that the Euclidean distances have the same ordering as the original dissimilarities, i.e. only ordinal relationships need to be preserved. Non-metric MDS is especially applicable when distances are unreliable or simply unavailable and only the rank-order of the dissimilarity matrix contains significant information. Isotonic regression is commonly used to find a configuration that minimizes the discrepancy in the rank-order of the original dissimilarities and the pairwise distances in the embedding[13]. In this study, we focus on metric MDS, since we need to visualize the pairwise genetic distances in order to represent the genetic diversity in the MTBC population.

3.2.1.1 Stress Majorization

In order to preserve the pairwise distances of all the nodes, we seek to minimize the stress function that measures the deviation of the Euclidean distance between points in the new reduced space and their corresponding dissimilarities. The stress function is defined as follows:

$$stress(X) = \sum_{i<j} w_{ij}(||X_i - X_j|| - d_{ij})^2 \tag{3.1}$$

where X_i is the position of the node i in the embedding and d_{ij} represents the distance (dissimilarity) between nodes i and j. The normalization constant $w_{ij} = d_{ij}^{-\alpha}$, $\alpha = 2$

is commonly used. This constant can be tweaked to alter the emphasis on preserving distances between nearby or faraway nodes as needed.

However this objective is non-convex, therefore most successful algorithms have been methods to find approximate solutions, the most popular being stress majorization. Introduced by deLeew [49], the majorization process iteratively minimizes quadratic approximations to the original stress function. Each approximation, known as the majorization function, touches the original stress function at a single point *(supporting point)*. The majorization function is simpler to minimize than the original stress function and always takes a value greater than or equal to the original function. This concept is illustrated in Fig. 3.1 where the majorization function $g(x, x0)$ touches the stress function $f(x)$ at the point $x0$. The minimum of this function, $x1$ is the next supporting point.

This iterative majorization method can be applied to minimizing the stress function (3.1). The corresponding majorization function for (3.1) can be defined as follows [13]:

$$\sigma(X, \bar{X}) = \sum_{i<j} w_{ij} d_{ij}^2 + Tr(X' L^w X) - 2Tr(X' L^{\bar{X}} \bar{X}) \tag{3.2}$$

where the $n \times n$ weighted Laplacian is defined as follows

$$L^w_{i,j} = \begin{cases} -w_{ij} & i \neq j, \\ \sum_{i \neq k} w_{ik} & i = j \end{cases}$$

and

$$L^{\bar{X}}_{i,j} = \begin{cases} -w_{ij} d_{ij} inv\left(||\bar{X}_i - \bar{X}_j||\right) & i \neq j, \\ -\sum\limits_{i \neq j} L^{\bar{X}}_{i,j} & i = j \end{cases}$$

where $\bar{X}$ is the supporting point, the value of X in the previous iteration.

The advantage of this technique is that it involves minimizing a series of simple quadratic functions resulting in a sequence of non-increasing function values. However, the final solution may be a local minimum given the nonconvex nature of the problem.

3.2.2 Other Graph Embedding Methods

In this section, we list a few other frequently used methods that optimize layouts with respect to some notion of proximity. While the definition of proximity varies, the overall objectives are similar.

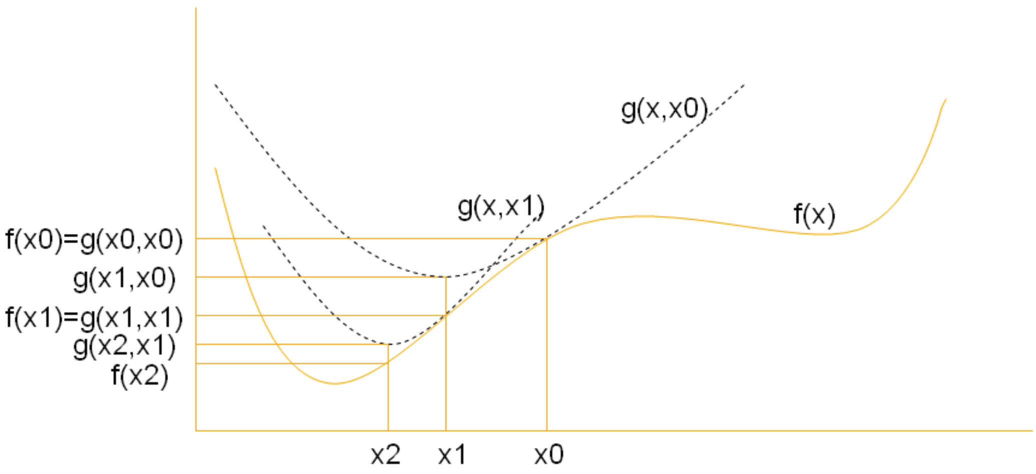


Figure 3.1: In each iteration, we minimize the majorization function $g(x, z)$ that is an upper bound on the original function $f(x)$ and touches it at a single point.

3.2.2.1 Force-directed Methods

These extremely popular and intuitive graph drawing algorithms are based on the principle of treating a graph as a physical system comprising of particles representing nodes that exert attractive and repulsive forces on each other [45, 96, 59, 8]. The goal is to find a state of equilibrium where the forces balance each other out. Such algorithms are also known as *spring embedding* methods, based on the analogy of treating nodes as metal rings connected by springs (edges). Often these methods assume that all nodes exert repulsive forces on each other, while nodes connected by edges also have attractive forces between them. In [45], the authors discuss the effect of various possibilities of functions to represent the forces. They recommend attractive forces be modeled as quadratic functions proportional to the square of the distance between nodes and repulsive forces inversely proportional to node distances in order to help the solution escape local minima. The authors also provide a more efficient grid variant of the algorithm, where repulsive forces are computed only between nodes lying in neighboring grids, or rather in a ball of $radius = 2k$ from the node. Such a strategy of obtaining the equilibrium state in which attractive and repulsive forces are balanced leads to the following two advantages: (1) Nodes that are connected by edges and that are far away from each other experience very strong attractive forces towards each other that act towards reducing the distance between the nodes. Indirectly, this leads to a reduction in the number of edge-crossings. (2) The repulsive forces prevent overcrowding and result in a uniform distribution of nodes. Methods for obtaining fast approximations of spring embeddings have also been proposed [59].

3.2.2.2 Spectral Techniques

Spectral graph drawing techniques generate layouts based on the eigenvectors of a matrix associated with the graph. Eigen decompositions of various associated matrices such as the adjacency, Laplacian, the weighted Laplacian and distance matrix have been used in previous works [68, 60, 17, 24]. In [101], the authors show that the eigen-projection method based on using the low eigenvectors of the Laplacian (neglecting the lowest eigenvector) to represent the node co-ordinates is equivalent to the energy-minimization approach of force-directed methods. Using the low degree-normalized eigenvectors of the Laplacian [68] was also related to desirable aesthetic criteria such as placing nodes with high degree in the center while simultaneously trying to enlarge the scatter, generating layouts with more evenly distributed nodes. Laplacian Eigenmaps [10] is a general dimensionality reduction technique for data that uses concepts from spectral graph embedding to visualize data such that locality information is preserved. The algorithm involves inferring an adjacency matrix based on neighborhood information, generating weights based on a heat kernel representing pairwise distances or the adjacency matrix, and finally computing eigenvalues and eigenvectors of the weighted Laplacian calculated based on the weights. The low eigenvectors are used to represent the node co-ordinates. Local structure is said to be preserved by this spectral embedding technique. Some other locality preserving embedding techniques are listed in the next section. Most importantly, spectral graph drawing methods have the advantage of rapid computation time and are therefore especially useful for drawing very large graphs.

3.2.2.3 Locality Methods

While MDS creates a linear embedding of the data, there has been a vast body of work on intrinsically low dimensional data that has a non-linear structure lying in high dimensional space [86, 97, 62, 103]. While the data may belong to a high dimensional space, the intrinsic dimensionality of the data may be significantly smaller. These are vastly different methods that have the common objective of discovering the manifold embedded in a reduced space.

One such method, Stochastic Neighborhood Embedding (SNE) is based on representing proximities in high-dimensional space as conditional probabilities and preserving these probabilities in the reduced space. The Euclidean distances in high-dimensional space are converted into probabilities as follows:

$$p_{j|i} = \frac{exp(-|x_i - x_j|^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-|x_i - x_k|^2/2\sigma_i^2)}$$

This term $p_{j|i}$ serves as a measure of similarity and represents the conditional probability that node i would pick node j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at i. A similar conditional probability $q_{j|i}$ is determined in the reduced space. The Kullback-Leibler divergence is used as the loss function that seeks to minimize the discrepancy between the conditional probabilities $p_{j|i}$ and $q_{j|i}$. The authors of [62] recommend using a gradient descent strategy to minimize this objective. A variation to this method, the t-SNE prescribes modeling the conditional probability $q_{j|i}$ in the reduced space as a Student-t distribution with one degree of freedom, in order to overcome problems of overcrowding. Locality preserving methods such as SNE that faithfully represent the local structure have become popular for visualization tasks.

In this chapter, we described a wide array of existing proximity preservation techniques, as well as methods for minimizing edge-crossings. However, the crossing minimization approaches leaves little flexibility in the placement of nodes that can be utilized to satisfy other equally or possibly more important requirements of the visualization such as defined by proximity preservation objectives. Similarly, existing proximity preserving embedding objectives do not directly address aesthetic criteria like minimizing edge-crossings and node-overlaps. In the next chapter, we describe how some of the continuous embedding objectives described in section 3.2 can be combined with constraints to minimize edge-crossings. Toward this goal, we formulate new constraints for edge-crossings that can easily be incorporated as penalties and used in conjunction with objective functions for proximity preservation.

CHAPTER 4
Penalty Methods for Proximity Preservation and Crossing Minimization in Graph Visualizations

We propose MAA, a novel approach to embedding heterogeneous data in high-dimensional space characterized by a graph. The nodes of the graph are themselves data points with associated metric distances defining proximity relations. Targeted towards data visualization, the objectives of the embedding are two-fold: (i) preserve proximity relations as measured by some embedding objective, and (ii) simultaneously optimize an aesthetic criterion, no edge-crossings in the embedding, to create a clear representation of the underlying graph structure. In this chapter we propose a new approach for addressing the later criteria that exploit the theorems of the alternative to re-express the condition for no edge-crossings as a system of nonlinear inequality constraints. The approach has an intuitive geometric interpretation closely related to support vector machine classification. While edge crossing minimization can be utilized in conjunction with any optimization-based embedding objective, here we demonstrate the approach on multi-dimensional scaling by modifying the stress majorization algorithm to include penalties for edge crossings. An iterative penalty algorithm is developed and applied to creating *spoligoforests*. As described in Chapter 1, spoligoforests are *phylogenetic forests* used to visualize genetic relatedness betweens trains described by 55 biomarkers (spoligotype and MIRU loci) and associated non-Euclidean distances. The method is also demonstrated on a suite of randomly generated graphs with corresponding Euclidean distances that have planar embeddings with high stress. The proposed edge-crossing constraints and iterative penalty algorithm can be readily adapted to other supervised and unsupervised optimization-based embedding or dimensionality reduction methods. The constraints can be generalized to remove intersections of general convex polygons including node-edge and

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, AND K. P. BENNETT, *Crossing minimization within graph embeddings*, arXiv preprint arXiv:1210., (2012).
(2) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Preserving proximity relations and minimizing edge-crossings in high dimensional graph visualizations*, IEEE Symposium on Large Data Analysis and Visualization (LDAV), Providence, RI, 2011, pp. 131-132.
(3) A. SHABBEER AND K. P BENNETT, *Proximity preservation and crossing-minimization for graph embedding*, in The Snowbird Learning Workshop, 2012.

node-node intersections.

4.1 Introduction

Graphs model relationships between entities, where the entities are represented by nodes and the relations by edges. Frequently, the nodes of a graph represent objects that have their own intrinsic features with associated distances or similarity measures. Often we need a graph embedding that is a mapping of nodes from high-dimensional space to low-dimensional vectors that preserve these pairwise distances. The embedding can serve as a visualization tool to better understand underlying relationships in a dataset. The quality of a visualization can be gauged on the basis of how easily it can be understood and interpreted. Certain criteria have been identified that characterize a good visualization. As described in Chapter 3, for graphs, it is desirable to minimize edge crossings, a challenging problem in itself [65, 76]. While polynomial-time planarity testing algorithms exist, determining the minimum number of crossings for a graph is NP-complete [52]. For general data embedding, the desired quality is frequently expressed as a function of the embedding and then optimized. For example in Multidimensional Scaling (MDS) (or equivalently the Kamada-Kawai model in force-directed graph placement (FDP)), the goal is to produce an embedding that minimizes the difference between the actual distances and Euclidean distances between all nodes in the embedding. Thus, a natural question for such heterogeneous data that comprises of data points characterized by features and by an underlying graph structure [49, 33] is how to optimize the embedding criteria while minimizing the number of edge crossings in the embedded graph. The principle contribution of the work in this chapter is to express edge-crossing minimization as a continuous optimization problem so that both the embedding criteria and aesthetic graph visualization criteria can be simultaneously optimized.

The motivating heterogenous data/graph application for this work is visualization of phylogenetic forests of bacteria (spoligoforests) described in Chapter 1. Each node represents a genetic strain of *Mycobacterium tuberculosis* complex (MTBC) and each edge represents a putative evolutionary change. Each node or strain has a genetic fingerprint and a natural non-Euclidean distance that can be defined to every other strain even if they are not connected in the underlying graph (phylogenetic forest). Similarly, in a Web hyperlinks graph, each node may be a web page and the edge may represent a hyperlink between the pages. Each webpage is a document with intrinsic properties, so there is an associated distance or similarity measure between nodes even if no link exists between them.

Existing embedding and graph drawing methods typically do a good job at accurately capturing distances or drawing planar graphs but not both as described in Chapter 3. Figure 4.1 shows the visualization of planar spoligoforest for the LAM subfamilies of MTBC created by the proposed approach and 3 other embedding methods: (a) the proposed approach that preserves pairwise distances while minimizing edge crossings; (b) MDS using stress majorization as implemented by Graphviz Neato; (c) Graphviz Twopi - a planar radial graph algorithm http://www.graphviz.org/ and (d) Spectral embedding technique using the weighted Laplacian of graph as described in Laplacian Eigenmaps [10] for the original adjacency of graph. In (c), the radial graph is visually appealing but inaccurate, especially when the graphs are disconnected, because genetically similar strains belonging to the same TB sublineage indicated by identically colored nodes but different connected components are placed far apart. In (b), the distances between strains match the sublineage structure but there are many edge crossings. In (d) while genetically similar groups cluster together, there are edge-crossings and the genetic relatedness between all pairs of strains is less evident. The proposed approach in (a) represents distance correctly in a naturally emerging radial structure without any edge crossings in the layout by optimizing the MDS embedding or dimensionality reduction objective with additional edge cross penalties.

The key theoretical insight of the chapter is that the condition that two edges do **not** cross is equivalent to the feasibility of a system of nonlinear inequalities. In Section 4.2, we prove this using a theorem of the alternative. The transformed system ensures that the two edges are separated by a linear hyperplane. Thus the edge-crossing constraint reduces to a classification problem which is very closely related to support vector machines (SVM). The system of inequalities can then be relaxed to create a natural penalty function for each possible edge crossing. This non-negative function goes to zero if no edge crossings occur. This general approach is applicable to the intersection of groups of convex polyhedrons including nodes represented as boxes and edges represented as bars. The approach is a distinct and important departure from prior proximity preserving embedding methods that ignore crossings or that employ heuristics to generate layouts that avoid crossings [45, 33, 110] as well as crossing-minimization approaches as described in Chapter 3 that do not allow for flexibility in the placement of nodes. Spectral graph drawing methods

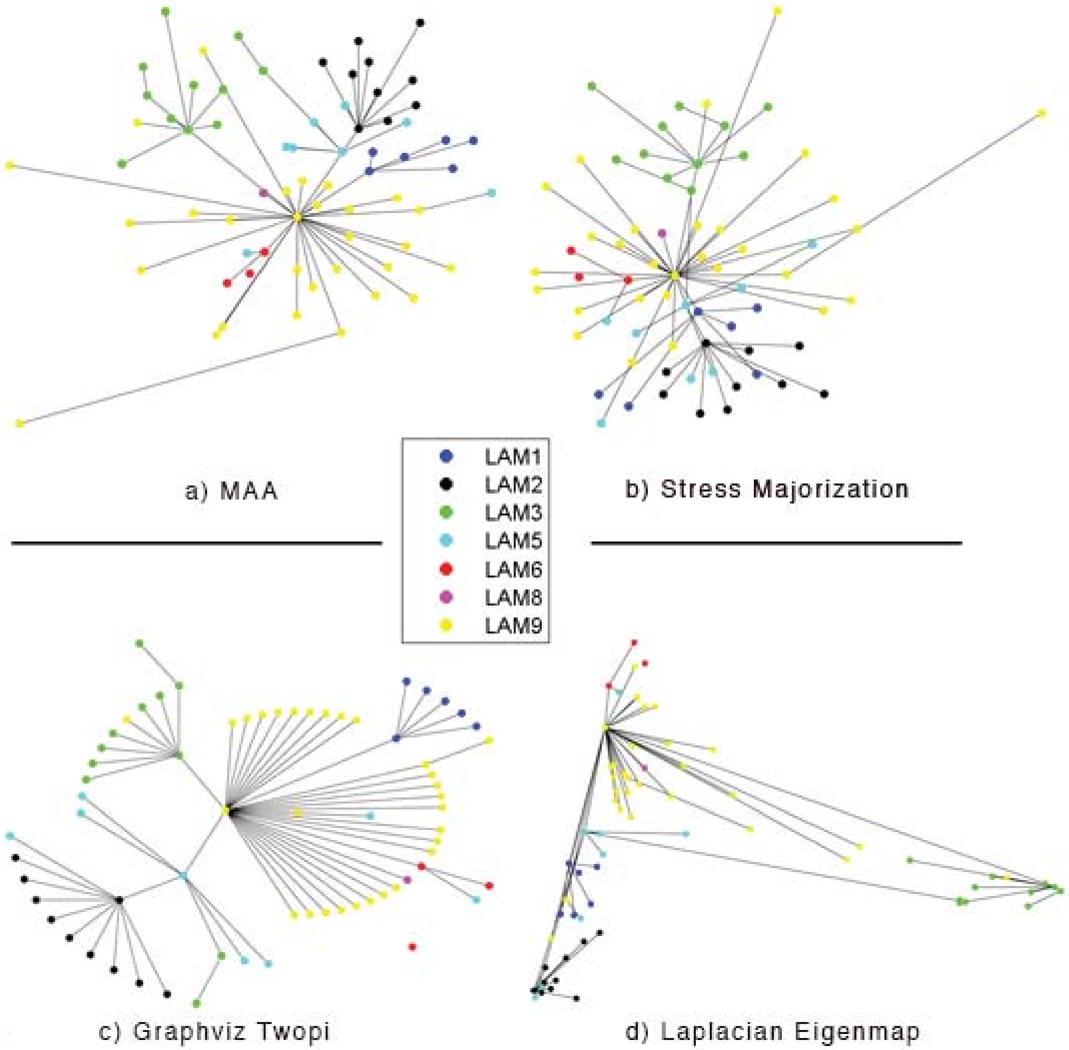


Figure 4.1: Embeddings of spoligoforests of LAM (Latin-American-Mediterranean) sublineages. Graph (c) is a planar embedding generated using Twopi, the radial layout is visually appealing, but genetic distances between strains are not faithfully reflected. Graph (d) generated by spectral decomposition of the weighted Laplacian preserves local structure but has edge-crossings. Graph (b), that optimizes the MDS objective and generated using Neato, preserves proximity relations but has edge-crossings. In graph (a), the proposed approach eliminates all edge crossings with little change in the overall stress. Note how in graph (a), the radial structure emerges naturally when both distances and the graph structure are considered.

like [68] generate embeddings based on the eigenspace of the Laplacian matrix. The Laplacian Eigenmap uses the eigenvectors of the weighted Laplacian for dimensionality reduction such that locality properties are preserved. However, such methods do not optimize aesthetic criteria such as minimizing edge-crossings.

In Section 4.3, we explore how edge-crossing constraints can be added to stress majorization algorithms for MDS or FDP. We develop an algorithm which simultaneously minimizes stress while eliminating or reducing edge crossings using penalized stress majorization. The method solves a series of unconstrained nonlinear programs in MATLAB. We demonstrate the method on graph embedding problems with associated metric distances. We first demonstrate the approach on a compelling problem involving genetic distances in tuberculosis molecular epidemiology. We provide further discussion of measures of genetic distances for MTBC biomarkers in Chapter 5. The graphical results are shown for spoligoforests drawn using a set of fifty-five biomarkers. The method found planar graph embeddings with smaller stress than those generated using the state-of-the-art NEATO algorithm (MDS). We then demonstrate the approach on randomly generated high-dimensional graphs designed to have planar embeddings with high stress. The results show that the proposed approach can produce two-dimensional embeddings with minimal edge-crossings with little increase in stress. Additional illustrations as well as animations of the algorithm illustrating how the edge crossing penalty progressively transforms the graphs as well are provided in the appendix and at http://www.cs.rpi.edu/~shabba/FinalGD/.

The following notation is used: for a vector x in $\mathbb{R}^n$, x_+ denotes the vector in $\mathbb{R}^n$ with components $(x_+)_i = max(x_i, 0), i = 1..n$, the 2-norm and 1-norm of x will be denoted by $||x||_2$ and $||x||_1$ respectively.

4.2 Continuous Edge-Crossing Constraints

We show how edge-crossing constraints can be expressed as a system of nonlinear inequalities through the introduction of additional variables for each edge crossing. Expressing the constraint that two edges must not cross as a system of nonlinear equalities is a key non-obvious first step for developing a continuous objective function to minimize edge crossings. Each point on an edge can be represented as the convex combination of the extreme points of the edge. Consider edge $\mathcal{A}$ with end points $a = [a_x \ \ a_y]$ and $c = [c_x \ \ c_y]$ and edge $\mathcal{B}$ with end points $b = [b_x \ \ b_y]$ and $d = [d_x \ \ d_y]$. The matrices A and B contain the end or extreme points of the edges $\mathcal{A}$ and $\mathcal{B}$ respectively. Any point in the intersection of edge $\mathcal{A}$ and $\mathcal{B}$ can be written as a convex combination of the extreme points of $\mathcal{A}$ and convex combination of the extreme points of $\mathcal{B}$. Therefore, two edges do not intersect if and only if the following system of equations has **no solution**:

$$\nexists \quad \delta_A \text{ and } \delta_B \text{ such that } \quad A'\delta_A = B'\delta_B \quad e'\delta_A = 1 \quad e'\delta_B = 1 \quad \delta_A \geq 0 \quad \delta_B \geq 0 \qquad (4.1)$$

where e is a vector of ones of appropriate dimension and $A = \begin{bmatrix} a_x & a_y \\ c_x & c_y \end{bmatrix}$ and $B = \begin{bmatrix} b_x & b_y \\ d_x & d_y \end{bmatrix}$.

The conditions that two given edges do *not* cross, i.e. that (4.1) has no solution, are precisely characterized by using Theorems of the Alternative that states that the linear system $Du \geq 0$, $d'u > 0$ has no solution u if and only if the system $D'v = d$, $v \geq 0$ has a solution v.

Theorem 1 (Conditions for no edge crossing)**.** *The edges $\mathcal{A}$ and $\mathcal{B}$ do not cross if and only if there exists u, α and β,*

$$\textit{such that } Au \geq \alpha e \qquad Bu \leq \beta e \qquad \alpha - \beta > 0. \qquad (4.2)$$

Proof. By Theorems of the Alternative, (4.2) has a solution if and only if the following system has no solution

$$A'\delta_A - B'\delta_B = 0, \quad e'\delta_A = 1 \quad e'\delta_B = 1 \quad \delta_A \geq 0 \qquad \delta_B \geq 0. \qquad (4.3)$$

System 4.3 has no solution if and only if the convex combination of the extreme points of $\mathcal{A}$ and $\mathcal{B}$ do not intersect. □

Allowing $\mathcal{A}$ and $\mathcal{B}$ to be of an arbitrary number of extreme points, the following corollary can be easily proven to apply to intersections between convex polyhedrons expressed as a convex combination of their extreme points.

Corollary 1 (Conditions for no intersection of two polyhedrons)**.** *Consider the polyhedrons $\mathcal{A} = \{x | x = A'\delta_A, e'\delta_A = 1, \delta_A \geq 0\}$ and $\mathcal{B} = \{x | x = B'\delta_B, e'\delta_B = 1, \delta_B \geq 0\}$. The polyhedrons do not intersect, $\mathcal{A} \bigcap \mathcal{B} = \emptyset$, if and only if $\exists u$ and γ such that*

$$\begin{aligned} Au - \gamma e &\geq e \\ Bu - \gamma e &\leq -e \end{aligned} \qquad (4.4)$$

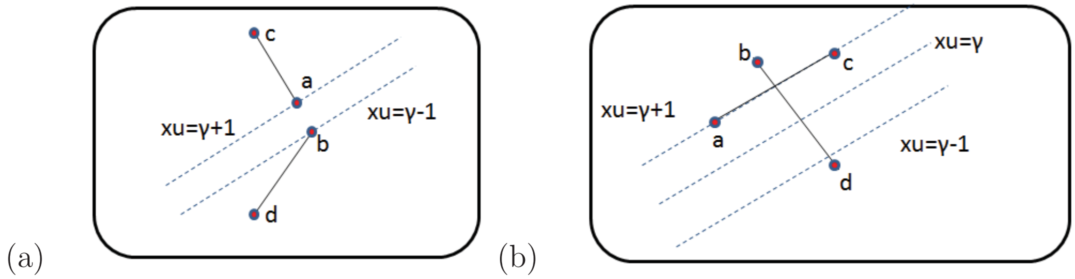


Figure 4.2: In (a) Edge $\mathcal{A}$ from a to c and edge $\mathcal{B}$ from b to d do not cross. Any line between $xu-\gamma = 1$ and $xu-\gamma = -1$ strictly separates the edges. Using a soft margin, the plane in (b) $xu - \gamma = 0$ separates the plane into half spaces that should contain each edge.

Therefore, two edges (or more generally two polyhedrons) **do not** intersect if and only if

$$\begin{array}{l} 0 = \min_{u,\gamma} ||(-Au + (\gamma+1)e)_+||_q^q + ||(Bu - (\gamma-1)e)_+||_q^q \quad l \\ \text{ where } (z)_+ = max(0, z) \text{ for } q = 1 \text{ or } \; q = 2. \end{array} \tag{4.5}$$

The minimization function in (4.5) provides a natural function for penalizing edges that do cross. Much like soft-margin SVM classification, two edges (or more generally two polyhedrons) do not intersect if and only if there exists a hyperplane ($xu' = \gamma$) that strictly separates the extreme points of $\mathcal{A}$ and $\mathcal{B}$. If the edges do not cross, then the optimal objective of (4.5) will be 0; while it will be strictly greater than 0 if the edges do cross. As in SVM, (4.5) can be converted into a linear or quadratic program depending on the choice of $q = 1$, or $q = 2$ respectively Figure 4.2 illustrates that the no-edge-crossing constraint corresponds to introducing a separating hyperplane and requiring each edge to lie in opposite half spaces.

4.3 Stress Majorization with Edge-Crossing Penalization

Edge-crossing constraints and penalties are a practical and flexible paradigm that can be used directly to minimize edge crossings as part of graph embedding optimization algorithms. Many algorithms are possible depending on the variant of the formulation of the penalty terms used. In this work, we used the differentiable least-squares loss for the penalty terms in (4.7). For ℓ edge objects there are $\frac{l(l-1)}{2}$ possible intersections and $O(\ell^2)$ penalty terms. However, an optimized embedding will typically only produce a fraction of the possible edge crossings. Thus this suggest an efficient iterative penalty approach because it need only deal with the small set of edge crossings that actually occur during the course of the algorithm.

Using a multi-objective penalty approach, edge crossing minimization can be incorporated into any optimization-based embedding or graph drawing formulation. In this chapter, we do multi-objective minimization combining edge-cross minimization with the stress function given by the highly-successful and widely-used MDS [28] objective:

$$stress(X) = \sum_{i<j} w_{ij}(||X_i - X_j|| - d_{ij})^2 \tag{4.6}$$

Here X_i is the position of the node i in the embedding and d_{ij} represents the distance between nodes i and j. The normalization constant $w_{ij} = d_{ij}^{-\alpha}$, $\alpha = 2$ is used. The stress function measures the deviance of the Euclidean distance between points in the new reduced space and their corresponding defined m-dimensional proximities. Thus, by minimizing the stress function, we aim to preserve the pairwise distances of all the nodes. Note, however, that the proximities in the high-dimensional space may be non-Euclidean in practice.

The SMACOF [30, 49] algorithm for minimizing stress is based on iterative majorization as described in Chapter 3. The method iteratively minimizes an auxillary function $\sigma(X, \bar{X})$ which is a quadratic approximation of the *stress* function such that $stress(X) \leq \sigma(X, \bar{X})$ always holds, and σ touches the surface of the $stress(X)$ function at the fixed point $\bar{X}$ *(supporting point)*. Recall σ was defined in (3.2).

The addition of 2-norm edge crossing penalties produces

$$\begin{aligned}\min_{X,u,\gamma} \bar{\sigma}(X, u, \gamma, \bar{X}) = \sigma(X, \bar{X}) + \sum_{i=1}^{m} \frac{\rho_i}{2}[||(-A^i(X)u^i + (\gamma^i + 1)e)_+||_2 \\ +||(B^i(X)u^i - (\gamma^i - 1)e)_+||_2]\end{aligned} \tag{4.7}$$

The iterative penalty method given in Algorithm 2 progressively increases the penalty parameters on each potential edge crossing until the algorithm converges. Algorithm 2 was implemented in MATLAB using the Optimization toolkit. The initial solution X^1 is calculated using *stress majorization*. The QP solver *quadprog* is used to solve QP (4.5). The BFGS algorithm as implemented in the MATLAB function *fminunc* is used to solve the edge-crossing penalized stress function. The following parameters are used: $\epsilon_2 = 1e - 3$, $\epsilon = 1e - 6$, $\kappa = 4$, $\rho_{min} = Stress(X^1)/\kappa$, $\rho_{inc} = 1.1$ and $\rho_{max} = 10^6$.

The algorithm begins by finding the stress-majorization solution X^1 (either by using this algorithm with no penalties or by using the NEATO package) and then refining the solution by introducing penalties for crossed edges. At each iteration, the edge crossing detection QP (4.5) is solved to both detect edge crossings and calculate the hyperplanes used in the edge-crossing penalties. For crossed edges, the penalties for those edge pairs are increased at each iteration until a maximum penalty is reached. The penalty is increased slowly to avoid problems with ill-conditioning. For this work, we emphasized edge crossing minimization so ρ_{max} is set high. But ρ_{max} can be reduced to examine the trade-offs between the embedding and edge-crossing minimization objectives. Computational efficiency is gained due to the fact that edges that never cross in the course of the algorithm have a penalty of 0. A formal analysis of the computational complexity of this algorithm is left for future work.

program 2 MAA: Stress majorization with edge-crossing penalties

```
Input: Pairwise distances, Adjacency matrix
while $||X^j - X^{j-1}|| \geq \epsilon$ and $\frac{\partial}{\partial X}(\bar{\sigma}(X^j, u, \gamma, X^{j-1})) \geq \epsilon$ and edge crossings exist do
  while $||X^j - X^{j-1}|| \geq \epsilon_2$ and edge crossings exist do
    for each edge pair $i = 1, \ldots, m$ do
      Find the crossed edges by solving QP (4.5) for edge pair $i$ to get $(u^i, \gamma^i, z^i)$

      if $||z^i|| \geq \epsilon$ then
        $\rho_i \leftarrow \rho_{min}$ for each new crossing $i$
      end if
    end for{X-phase}
    $Z \leftarrow X^j$
    $L^Z \leftarrow L^{X_j}$
    $X^{j+1} \leftarrow argmin(\sigma(X, \bar{X}) + \sum_{i=1}^{m} \frac{\rho_i}{2}[||(-A^i(X)u^i + (\gamma^i + 1)e)_+||_2 + ||(B^i(X)u^i - (\gamma^i - 1)e)_+||_2])$
    $j \leftarrow j + 1$
  end while
  Increase penalties for edges that remain crossed: $\rho_i = min(\rho_{inc} \times \rho_i, \rho_{max})$
end while
```

For each fixed value of ρ, Problem 4.7 is solved using an algorithm that alternates between minimizing with respect to X and u. For the X phase, a modified version of "Stress Majorization" [49] is used to optimize (4.7). The MATLAB BFGS optimization algorithm "fminunc" is used to optimize a quadratic upper bound on the stress plus the edge crossing penalties for a fixed u. In the u phase, the soft margin separating plane (u)

for each edge pair as defined by X is determined by solving (4.5). An inexpensive heuristic is used to reduce the number of edge-pairs checked (no crossing possible if bounding boxes enclosing edges do not intersect). The penalties for crossed edges are driven higher until no edge crossings exist or the problem converges; thus, most edge pairs have penalty parameter $\rho_i = 0$ since they never cross.

4.4 Results and Interpretation

Our approach was evaluated on a real-world applicatiom (visualization of spoligoforests) and challenging random graphs with non-Euclidean and Euclidean distances respectively.

4.4.1 Embedding of Spoligoforests with Non-Euclidean Distances

To demonstrate the performance of the approach, we return to the motivating application: visualization of spoligoforests [84] created from DNA fingerprints of MTBC. We examine the visualization of spoligoforests with non-Euclidean distance matrices defined using spoligotype and MIRU type (MTBC biomarkers) for four problems as summarized in Table 4.1 and shown in Figure 4.1 and the supplementary material. Each node of the spoligoforest corresponds to a distinct genotype of MTBC as determined by two types of DNA fingerprints: 43 spoligotypes and 12 MIRU. The spoligoforest is determined as in [89] and is always planar by definition since it consists of multiple trees. The distance is measured by the number of distinct changes in the spoligotypes and MIRU. Thus it is not a Euclidean distance. MAA was initialized using the MDS produced by the stress majorization algorithm and run for all edge crossing penalties until convergence. All stresses are scaled such that the MDS stress produced by NEATO is 1.

The table shows the efficacy of the method in optimizing the multiple objective For each problem, we present four visualizations of the spoligoforest drawn using: a) MDS+edge-crossing penalties, (b) MDS alone using stress majorization as implemented in Graphviz Neato (c) Planar Embedding as produced by radial layout producing algorithms Graphviz Twopi. (d) Laplacian Eigenmap embedding based on weighted Laplacian. In every case, the proposed method can dramatically reduce the edge crossings, while making only minor changes in the total stress (values = 1 are the same as the stress found by MDS as implemented in NEATO). Note while the original graph is in a 55 dimensional space, the data is inherently lower dimensional, thus many embeddings are possible with

Table 4.1: Comparison of the stress and number of crossings in embeddings generated by the proposed approach MAA that optimizes with respect to proximity preservation as well as edge-crossings with (i) MDS using Stress Majorization (as implemented in Neato) that minimizes proximity stress, but not edge-crossings(ii) Planar Embeddings (drawn using Twopi for spoligoforests, original embeddings for random graphs) that minimize edge crossings but not stress (iii) Laplacian Eigenmaps that minimize an alternative proximity preservation objective only. All stress results are normalized so that the NEATO stress is 1. Results shown for three MTBC spoligoforest datasets.

Spoligoforest	**|V|**	**|E|**		**MAA**	**MDS**	**Twopi**	**Lap. Eig.**
LAMs	68	66	**stress**	0.91	1.0	3.12	9.75
LAMs	68	66	**# cross.**	0	43	0	37
M. africanum	97	89	**stress**	0.99	1.0	6.32	15.75
M. africanum	97	89	**# cross.**	0	11	0	35
H, X, LAM	45	29	**stress**	0.90	1.0	8.64	18.92
H, X, LAM	45	29	**# cross.**	0	9	0	9
SpolDB4	151	138	**stress**	1.06	1.0	3.32	12.5
SpolDB4	151	138	**# cross.**	2	51	0	144

similar stress. In three of the four graphs, MDS+edge-crossing penalties actually produced graphs with less stress than the MDS results returned by NEATO, illustrating that edge-crossing penalties may help guide stress majorization to a more desirable local minima with little or no change in the overall stress. The pictures produced are more informative and accurate than those produced by all existing spoligoforest visualization software that use Graphviz algorithms, including Twopi, that disregards genetic distances available in the heterogeneous data [84, 89] as well as the visualization produced by Laplacian Eigenmaps. The results reported were performed on a Lenovo Thinkpad W500 laptop with 4GB RAM. The proposed approach can be used to dynamically remove edge crossings in an existing graph. An animation of the proposed algorithm altering the initial MDS solutions can be viewed in the supplementary material.

4.4.2 Randomly Generated Planar Graphs

In this section, we demonstrate the performance of the method on more complex graphs for which there exists some planar embedding. We illustrate how graphs in high-

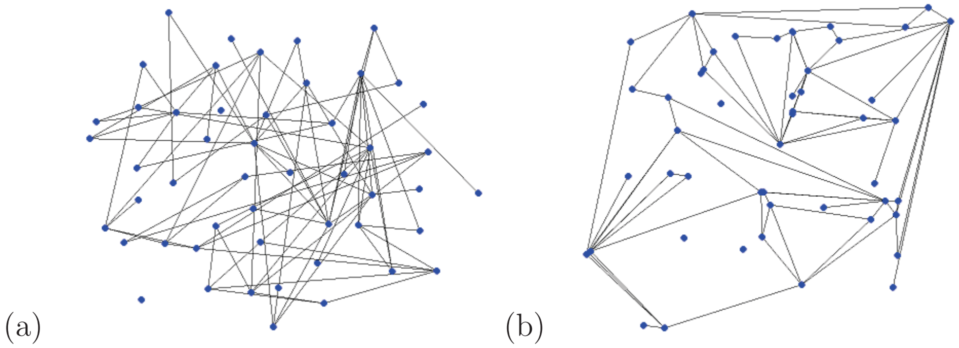


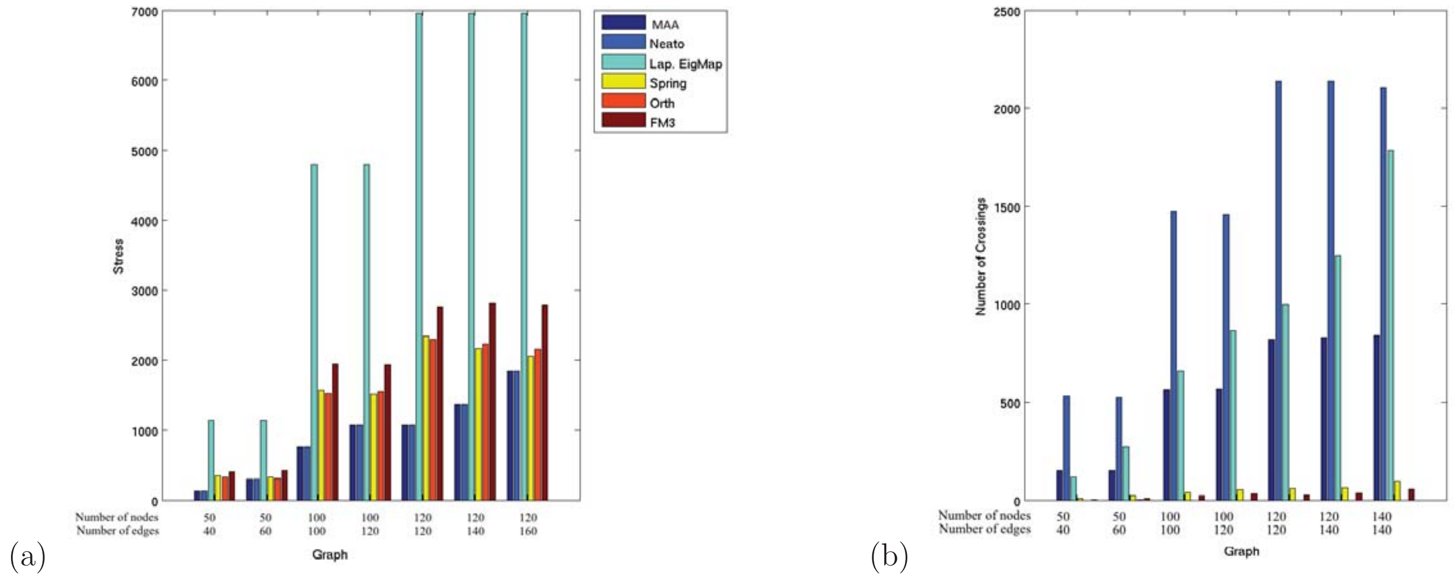
Figure 4.3: Embeddings for randomly generated graph in $\mathbb{R}^7$ with 50 nodes and 80 edges using (a) Stress majorization *(stress=131.8, number of crossings=369)* and (b) MAA *(stress=272.1, number of crossings=0)*. The original planar embedding had *stress= 352.5*.

dimensional space can be clearly and accurately represented in $\mathbb{R}^2$ while preserving pairwise distances and minimizing edge-crossings.

In order to evaluate the performance of the algorithm we generate random graphs that have at least one known planar embedding. However, this known planar embedding violates the proximity preservation requirement. $|V|$ points were generated in $\mathbb{R}^n$ (for n=7, 15 and 20). The Euclidean distances between each pair of points was determined. The points were projected in $\mathbb{R}^2$ and $\|E\|$ edges were introduced between nodes so that planarity is preserved, as per the method in [31] using a Markov Chain Algorithm. Since the planar embedding has high stress and is not truly representative of the proximity relations in the data, it is not the most desirable embedding. By relaxing the requirement for 0 crossings, we can find alternate embeddings that preserve stress with significant reduction in edge-crossings. Our multi-objective approach finds such embeddings that achieve a balance between the two often contrary objectives.

Figure 4.3 shows the graph embedding produced using only the MDS objective as compared with applying edge-crossing penalties.

The results presented in Fig. 4.4 are from of a set of 160 randomly generated graphs with 50 to 120 nodes and 40 to 160 edges averaged for all graphs with the same $|E|$. Optimizing with respect to the stress alone (MDS), results in embeddings that have edge-crossings. The Laplacian Eigenmap embedding that is optimized with respect to a different proximity preservation objective aimed at preserving local structure also has a large number of crossings. Force-directed placement methods denoted by Spring, FM3



(a)

(b)

Figure 4.4: Comparison of (a) stress and (b) number of crossings in embeddings for randomly generated graphs with 50-120 nodes and 40-160 edges generated using 6 different algorithms MAA, Neato (Stress Majorization), Lapacian Eigenmaps, Spring, Orthogonal, Fast Multipole Multilevel Method (FM3).

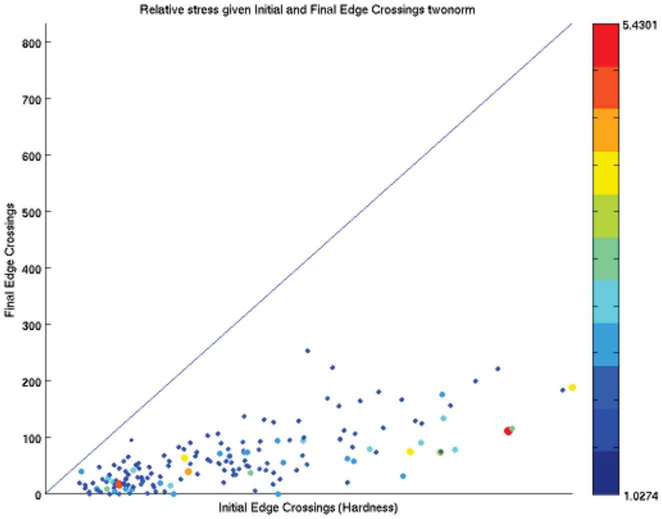


Figure 4.5: Plot of final edge-crossings vs initial edge crossings in MAA embeddings for 160 randomly generated graphs with 50 nodes and 80 edges. The size and color of the nodes represents the ratio of final stress to the stress majorization solution as found by Neato. MAA can produce embeddings with a significant reduction in the number of crossings with small increase in stress.

[45, 59] and planar grid embedding techniques like ORTH [102] have a low number of edge-crossings but can have high stress. Alternate embeddings are generated by MAA that preserve proximity relations while keeping the number of edge-crossings low, thus representing the underlying graph structure (adjacency and connectivity information) clearly. Figure 4.5 illustrates the performance of the algorithm in terms of final stress and number of crossings in MAA embeddings. The embeddings found for a majority of the graphs have marginal increase in stress as compared to the MDS solution, but with a significant reduction in number of crossings as indicated by the blue dots.

4.5 Discussion

We developed a novel multiobjective approach to simultaneously optimizing preservation of proximity relations and aesthetic criteria for heterogeneous graph data by introducing a fundamentally new paradigm for elimination of edge crossings in graph embeddings. This work demonstrates how edge-crossing constraints can be formulated as a system of nonconvex constraints. Edges do not cross if and only if they can be strictly separated by a hyperplane. If the edges cross, then the hyperplane defines the desired half-spaces that the edges should lie within. The edge-crossing constraints can be transformed into a continuous edge-crossing penalty function in either 1-norm or least-squares form. We developed a stress majorization algorithm with edge-crossing penalties. Computational results demonstrate that this approach is quite practical and tractable. Continuous optimization methods can be used to effectively find local solutions, a very desirable outcome since drawing graphs with a minimum number of edge crossing is NP-Hard. Successful results were illustrated on problems of the epidemiology of tuberculosis with genetic distances and phylogenetic forests that were not adequately addressed using existing planar graphing approaches since they give undesirable results on disconnected graphs. Interestingly, MAA, i.e. MDS with edge crossing penalties actually found embeddings with less stress and no edge crossings than the NEATO algorithm. This may be caused by the fact that the MTBC data is from a 55 dimensional space and the MDS stress is highly nonconvex with many possible locally optimal embeddings existing with similar stress, thus the edge crossing constraints may help guide the algorithm to a more desirable local solution both from a stress and aesthetic point of view. Results on high dimensional random graphs with planar embeddings show that the method can find much more desirable solutions from a visualization point of view with only relatively small changes in stress.

This work opens up many avenues for future research at the intersection of machine learning and data visualization. Here we focused on elimination of edge crossings and stress optimization (MDS). The general multiobjective approach is applicable to any optimization-based graph drawing, dimensionality reduction or embedding methods [103, 33] used for data visualization in both supervised and unsupervised learning. Also, the theorems and algorithms are directly applicable to the intersection of convex polygons in general within embeddings of arbitrary dimensions. Thus, the method can also be used to eliminate node-node overlaps and node-edge crossings. Our work was limited to planar graphs, but the penalty approach can be used to reduce crossings in more general graphs. Since the edge-crossing constraints are very closely related to linear SVM, all the different classification and regularization loss functions could be used to produce crossing-penalty functions with different aesthetic effects and algorithmic ramifications. For example, maximum margin separation can enforce minimum spacing between objects. This work used the MATLAB function “fminunc” as its primary workhorse – which inherently limits the problem size. In reality, there is a great potential for making highly scalable special purpose algorithms for edge-crossing-constrained graph embeddings. While the method was motivated by the need to preserve pairwise distances in heterogeneous graph data as defined by the MDS objective, it can be used to eliminate edge-crossing with any embedding objective. In the next chapters, we explore some of these promising research directions. We describe one such such scalable algorithm Alternating Directions of Multiplier Methods (ADMM) that can be potentially adapted to this problem in Chapter 5.

CHAPTER 5
Alternating Directions of Multiplier Methods for Visual Analytics

In this chapter, we discuss the algorithm MAA+ for generating spoligoforest layouts, an improvement on MAA described in Chapter 4. Proximity preserving embedding objectives with penalties for intersection of objects, such as the one presented in Chapter 4, are nonconvex and nonsmooth (due to *L1-penalties*). The subproblem of finding the separating planes is nonsmooth as well. There is a need for a scalable algorithm to solve these problems with nonsmooth objectives. We propose the use of Alternating Directions of Multiplier Methods (ADMM) algorithm as a simple and elegant solution. In section 5.1, we provide a background of the ADMM framework, followed by section 5.2, where we provide a description of the ADMM algorithm applied to the constrained graph embedding problem. We show how the ADMM framework could be used to solve both the *x-subproblem* and the *u-subproblem.* The MAA+ algorithm uses ADMM to solve the *x-subproblem.* For the u-subproblem, the algorithm uses closed form approximations described in Chapter 7. In section 5.3, we describe computational experiments that show the efficacy of the MAA+ method.

5.1 ADMM Framework

In this section, we discuss the Alternating Directions of Multiplier Methods (ADMM) algorithm, a method well suited to dealing with massive optimization problems with nonsmooth objectives or constraints. Nonsmooth problems are difficult because there are multiple nonunique subgradients that do not necessarily provide information about a direction of improvement. Various approaches are used to handle nonsmoothness such as bundle methods, generating smooth approximations [73, 77]. Bundle methods essentially

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, AND K. P. BENNETT, *Crossing minimization within graph embeddings*, arXiv preprint arXiv:1210., (2012).
(2) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Preserving proximity relations and minimizing edge-crossings in high dimensional graph visualizations*, IEEE Symposium on Large Data Analysis and Visualization (LDAV), Providence, RI, 2011, pp. 131-132.
(3) A. SHABBEER AND K. P BENNETT, *Proximity preservation and crossing-minimization for graph embedding*, in The Snowbird Learning Workshop, 2012.

approximate the subdifferential of the objective by collecting subgradient information in successive iterations and using this “bundle” to construct a piecewise linear approximation of the original objective. Such methods represent alternate methods to solving the constrained embedding problem. However, we propose the use of Alternating Directions of Multiplier Methods (ADMM) algorithm that elegantly handles nonsmoothness by providing an iterative solution where the computation of each iterate has a simple closed-form solution.

In the rest of this section we look at some of the foundations of the ADMM algorithm. The ADMM algorithm was originally designed so as to combine the benefits of fast convergence rates offered by methods of multipliers and the potential for distribution of the dual decomposition method [15]. The motivation and algorithmic framework of ADMM are described below.

The concepts underlying *Augmented Lagrangian methods* (also known as *multiplier methods*) form the basis for ADMM [12]. The fundamental strategy of multiplier methods is to generate a sequence of approximate problems which is equivalent to the original constrained optimization problem. Each approximate problem is an unconstrained optimization problem and is much easier to solve than the original. Moreover, each approximate problem need not be solved to perfect accuracy, and solutions to each iteration are used to guide the solution to the subsequent iteration. Multiplier methods are related to the penalty methods used in Chapter 4, that convert a constrained optimization problem into an unconstrained one by applying a large penalty on any infeasible solutions. However, multiplier methods reduce the possibility of ill-conditioning by explicitly including an estimate of the Lagrange multiplier y in the formulation of the function to be minimized- the augmented Lagrangian. The augmented Lagrangian of the equality-constrained optimization problem defined in (5.1) is formulated as in (5.2).

$$\begin{aligned} \min_{x}. \quad & f(x) \\ \text{s.t.} \quad & Ax = b \end{aligned} \tag{5.1}$$

with variable $x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m \times n}$ and $f : \mathbb{R}^n \to \mathbb{R}$ is convex.

$$L_\rho(x, y) = f(x) + y'(Ax - b) + \frac{\rho}{2}||Ax - b||^2 \tag{5.2}$$

Here $\rho > 0$ is the penalty parameter and essentially defines the weight that we assign to constraint satisfaction relative to minimization of the original objective f [78].

The intuition behind the methodology of multiplier methods whereby minimizing the sequence of augmented Lagrangians (5.2) is equivalent to minimizing the original function (5.1) is as follows. For a bounded series y_k and $\rho \to \infty$, the additional terms $y'(Ax-b) + \frac{\rho}{2}||Ax-b||^2$ that are *augmented* with the original function f are zero. While these additional terms tend to ∞ if the constraints are not satisfied. It can be shown that under some assumptions e.g. f is a continuous function defined over a closed set, the limit point of x_k for $k = 1 \ldots$ obtained by minimizing $L_\rho(x, y^k)$ for a bounded series y_k and $\rho \to \infty$ is equivalent to the global minimum of (5.1).

The working of the method of multipliers can also be explained as follows. Consider the *augmented* dual function to be defined as

$$g(y) = \inf_x L_\rho(x, y) \tag{5.3}$$

and therefore the dual problem is

$$\underset{x}{\text{maximize}}\; g(y)$$

The multiplier method involves solving the dual problem using a gradient ascent approach *(dual ascent)*. The dual optimal point y^* can be used to obtain the primal optimal point as follows:

$$x^* = \underset{x}{\text{argmin}}\; L_\rho(x, y^*) \tag{5.4}$$

Fundamentally, the multiplier method is an iterative algorithmic framework in which the following two steps are performed in each iteration. First, we determine

$$x^{k+1} = \underset{x}{\text{argmin}}\; L_\rho(x, y^k) \tag{5.5}$$

Then the gradient of the dual function is determined using the updated x^{k+1} value. The gradient $\nabla g(y) = Ax^{k+1} - b$ is used to update the dual variable via gradient ascent using a step-size of α.

$$y^{k+1} = y^k + \alpha^k (Ax^{k+1} - b) \tag{5.6}$$

Steps (5.5) and (5.6) are performed for $k = 1 \ldots n$ until convergence criteria are met. In successive iterations of this method the primal residual $Ax - b$ converges to zero resulting in the optimal solution.

Note, it can be shown that using a step-size $\alpha = \rho$ makes each iterate dual feasible. Note that the optimality conditions of the original problem (5.1) are, primal feasibility: $Ax^* - b = 0$ and dual feasibility: $\nabla f(x^*) + A'y^* = 0$. By definition, x^{k+1} minimizes $L_\rho(x, y^k)$. Therefore,

$$\nabla_x f(x^{k+1}) + A'(y^k + \rho(Ax^{k+1} - b)) = 0 \tag{5.7}$$

If a step-size of ρ is used in the dual variable update, then by substituting in (5.7),

$$\nabla_x f(x^{k+1}) + A'y^{k+1} = 0$$

Therefore, using a step-size of ρ makes the next iterate (x^{k+1}, y^{k+1}) is dual feasible.

The ADMM algorithm is a modification of the multiplier method to help overcome the following disadvantage: Even when f is separable the augmented Lagrangian is not. The traditional multiplier method approach involves minimizing with respect to all the primal variables jointly. ADMM is designed to take advantage of the separable structure using an alternating strategy as follows. Let the problem be expressed as below, by splitting the original variable x into x and z,

$$\begin{aligned} \min_{x,z}. \quad & f(x) + g(z) \\ \text{s.t.} \quad & Ax + Bz = c \end{aligned} \tag{5.8}$$

Instead of performing the primal variable update with respect to x and z jointly as specified by multiplier methods strategy, x and z are updated in an alternating fashion. The $x - update$ is performed for a fixed z from a previous iteration, and similarly for the $z - update$. Therefore, each update step involves solving a simpler optimization problem rather than jointly minimizing with respect to both x and z. These iterations are repeated until convergence.

```
**for** $k = 1 \ldots$ until convergence **do**
    $x^{k+1} = \underset{x}{\text{argmin}} L_\rho(x, z^k, y^k)$
    $z^{k+1} = \underset{z}{\text{argmin}} L_\rho(x^k + 1, z, y^k)$
    $y^{k+1} = y^k + \rho(Ax^{k+1} + Bz^{k+1} - b)$
**end for**
```

Another potential advantage of ADMM is its foundations in the dual decomposition method. In this work, we decomposed the problem into several small problems: the *x-subproblem* and several small *u-subproblems*. Solving for each of these variables independently makes the computation significantly easier as compared to solving a single optimization problem with respect to all variables jointly. Dual decomposition is a variant of the dual ascent, and used when the objective f is separable i.e. can be expressed as $f(x) = \sum_{i=1}^{n} f_i(x_i)$, where x_i are subvectors of x. The x-minimization step can also be split into n separate problems that are independent and can be solved in parallel making computation more efficient. Since it allows for distributed optimization it is possible to develop solutions capable of handling a large numbers of groups of independent constraints in parallel. This parallelization of these tasks offers an interesting future direction.

In the following section we describe solutions to the constrained embedding problem using the ADMM framework described above. We discuss the suitability of this algorithm to the two subproblems: (i) finding separating hyperplanes and (ii) minimizing a penalized objective to obtain layouts with low stress and edge-crossings.

5.2 ADMM for Constrained Graph Embedding

In this section, we provide a description of the application of ADMM to the constrained graph embedding problem. We use the the problem developed in Chapter 4 as an example. Both subproblems in the alternating scheme described in Chapter 4 can be solved using ADMM. Recall that the first problem involves computing the separating planes as defined by solving (4.5) and is described in section 5.2.1. The second problem, described in section 5.2.2, involves computing the layout with minimal stress subject to constraints defined by the previously determined separating planes.

5.2.1 ADMM for Finding Separating Planes

We can rewrite (4.5) in ADMM form 5.8 and determine the separating plane for two edge objects as follows

$$\begin{aligned} \min_{z,x} \quad & e'z_+ \\ \text{s.t.} \quad & Cx + e = z \end{aligned} \tag{5.9}$$

where $C = \begin{bmatrix} -A & e \\ B & -e \end{bmatrix}$, $x = \begin{bmatrix} u \\ r \end{bmatrix}$, $x \in \mathbb{R}^3$ defines the separating plane and and e is a vector of ones of appropriate dimension. Note that adding a regularization term of $||u||^2$ to the objective is equivalent to finding the maximum margin hyperplane and can help generate more aesthetically pleasing graphs. Similarly, constraining

The augmented Lagrangian for (5.9) is :

$$L_\rho(x, y, z) = e'z_+ + y'(Cx + e - z) + \frac{\rho}{2}||Cx + e - z||^2 \tag{5.10}$$

As per the algorithm described in 5.1, (5.9) is minimized by alternating between updating x, z and the dual multipliers y of the equality constraints. We develop the updates for each variable below:

$x - update$

The $x - update$ requires minimizing the augmented Lagrangian (5.10) with respect to x.

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\ L_\rho(x, z^k, y^k) \tag{5.11}$$

By the first order optimality conditions of 5.11

$$\begin{aligned} \nabla_x L_\rho(x, z^k, y^k) &= C'y + \rho C'(Cx + e - z) \\ &= 0 \end{aligned}$$

The new value of x defines the separating plane. It can then be obtained by solving

$$\begin{aligned} Hx &= \psi \\ \text{where } H &= C'C \\ \psi &= \rho C'(-e + z - v) \\ \text{and } y &= \rho v \end{aligned} \tag{5.12}$$

$z - update$

$$\begin{aligned} z^{k+1} &= \underset{z}{\text{argmin}} e'z_+ - y'z + \frac{\rho}{2}||Cx + e - z||^2 \\ &= \underset{z}{\text{argmin}} \frac{e'|z| + e'z}{2} - y'z + \frac{\rho}{2}(z'z - 2(Cx + e)z) \\ &= \underset{z}{\text{argmin}} \frac{e'|z|}{2} - \rho(v + Cx + e - \frac{e}{2\rho})z + \frac{\rho}{2}z'z \\ &= \underset{z}{\text{argmin}} \frac{e'|z|}{2} - \frac{\rho}{2}||z - \theta||^2 \\ &\text{where } \theta = v + Cx + e - \frac{e}{2\rho} \end{aligned} \tag{5.13}$$

A closed form solution to problems of this form is defined as follows [85]

$$z = S_{\frac{1}{2\rho}}(\theta)$$

where the soft-thresholding operator is a shrinkage operator defined as below:

$$S_\kappa(a) = \begin{cases} a - \kappa & a > \kappa \\ 0 & |a| \leq \kappa \\ a + \kappa & a < -\kappa \end{cases} \tag{5.14}$$

Therefore the algorithm for finding the separating planes can be summarized as follows:

```
for $k = 1 \ldots$ until convergence do
    $x^{k+1} = H \backslash \psi$ {from(5.12)}
    $z^{k+1} = S_{\frac{1}{2\rho}}(\theta)$ {from (5.13)}
    $y^{k+1} = y^k + \rho(Cx^{k+1} + e - z^{k+1})$
end for
```

5.2.2 ADMM for Proximity Preservation and Overlap Minimization in Graph Visualizations

In Chapter 4, we described a penalty method to minimize stress subject to constraints for no edge-crossings. We can rewrite the *no-crossings penalized stress problem*

(4.7) that used quadratic penalties using *L1-penalties* as follows:

$$\min_{X,u,\gamma} \sigma(X,\bar{X}) + \sum_{i=1}^{m} \frac{\lambda_i}{2}[||(-A^i(X)u^i + (\gamma^i+1)e)_+||_1 + ||(B^i(X)u^i - (\gamma^i-1)e)_+||_1] \quad (5.15)$$

Here X represents the co-ordinates and $\bar{X}$ represents the supporting point (the value of X in the previous iteration). Such penalty functions are known as an *exact* penalty functions based on their desirable property that a single minimization can yield the *exact* solution if the penalty parameters λ_i is large enough[78]. This is in contrast to inexact penalty methods using quadratic penalties where the performance depends on the penalty parameter update strategy. In order to handle the nonsmoothness resulting from the use of *L1-penalties*, (5.15) can be formulated as an ADMM problem. Note, this strategy can be used to minimize any form of overlap e.g. node-node overlap as described in subsequent sections. As an illustration, consider the problem of minimizing stress and edge-crossings. We can rewrite (5.15) as follows:

$$\begin{aligned} \min_{x,z}.\quad & \frac{1}{2}x'L^w x - x'L^{\bar{x}}\bar{x} + \sum_{i<j} w_{ij}d_{ij}^2 + \lambda' z_+ \\ \text{s.t.}\quad & Ux + r + e = z \end{aligned}$$

$$\text{where } L^w = \begin{bmatrix} \bar{L}^w & 0 \\ 0 & \bar{L}^w \end{bmatrix} \text{ and } L^{\bar{x}} = \begin{bmatrix} \bar{L}^{\bar{X}} & 0 \\ 0 & \bar{L}^{\bar{X}} \end{bmatrix}$$

and $\bar{L}^w$ and $\bar{L}^{\bar{X}}$ are as defined in (3.2).

Here $x \in \mathbb{R}^{2|V|}$ and represents the concatenated x and y co-ordinates of the nodes. z is the newly introduced variable used to represent the equality constraint and transform the problem into ADMM form (5.8). $U \in \mathbb{R}^{4numcross \times 2|V|}$ is a sparse matrix and $r \in \mathbb{R}^{4numcross}$ is a vector suitably defined to represent equation 4.4 that defines the separating plane for each edge-crossing, such that the non-zero terms in U correspond to the terms u, and r represents γ.

The augmented Lagrangian is then defined as follows

$$L_\rho = \frac{1}{2}x'L^w x - x'L_{\bar{x}}\bar{x} + \sum_{i<j} w_{ij}d_{ij}^2 + \lambda' z_+ + y'(Ux + r - z) + \frac{\rho}{2}||Ux + r - z||_2^2 \quad (5.16)$$

The algorithm involves computing the following three steps: the *x-update*,*y-update* and the *dual variable update*.

x-update

$$\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}} L_\rho(x, z^k, y^k) && (5.17)\\ &= \underset{x}{\operatorname{argmin}} x' L^w x - 2x' L^{\bar{x}} \bar{x} + \lambda' z_+ && (5.18)\\ &+ y'(Ux + r + e - z) + && (5.19)\\ &\frac{\rho}{2} ||Ux + r + e - z||_2^2 \end{aligned}$$

By first order optimality conditions of (5.17),

$$\begin{aligned} \nabla_x L_\rho &= L^w x - L^{\bar{x}} \bar{x} + U'y + \rho U'(Ux + r + e - z)\\ &= 0 \end{aligned}$$

Therefore, $\underset{x}{\operatorname{argmin}} L_\rho(x, z^k, y^k)$ can be obtained by simply solving

$$\begin{aligned} Hx &= \psi\\ \text{where } H &= L^w x + \rho U'U\\ \psi &= L^{\bar{x}} \bar{x} - U'y - \rho U'(r + e - z) && (5.20) \end{aligned}$$

$z - update$

$$\begin{aligned} z^{k+1} &= \underset{z}{\operatorname{argmin}} L_\rho(x^{k+1}, z, y^k)\\ &= \underset{z}{\operatorname{argmin}} \frac{\lambda'}{2}(|z| + z) - y'z + \frac{\rho}{2} ||Ux + r + e - z||_2^2\\ &= \underset{z}{\operatorname{argmin}} \frac{\lambda'}{2}|z| + \frac{\lambda'}{2} z - y'z - \rho(Ux + r + e)'z + \frac{\rho}{2}||z||_2^2\\ &= \underset{z}{\operatorname{argmin}} \frac{\lambda'}{2}|z| - \rho(v + Ux + r + e - \frac{\lambda}{2\rho})'z + \frac{\rho}{2}||z||_2^2 \qquad \text{where } v = \frac{y}{\rho} && (5.21) \end{aligned}$$

$$z^{k+1} = \underset{z}{\text{argmin}} \frac{\lambda'}{2}|z| + \frac{\rho}{2}||z - \theta||_2^2 \tag{5.22}$$
$$\text{where } \theta = v + Ux + r + e - \frac{\lambda}{2\rho}$$

From [15], a closed form solution to problems of this form is defined as follows:

$$z = S_{\frac{\lambda}{2\rho}}(\theta)$$

where the soft-thresholding operator is a shrinkage operator defined as in (5.14).

dual variable update

This step, based on the dual ascent method, involves updating the dual variable y by taking a step along the gradient of the dual defined by the residual $Ux + r + e - z$

$$y^{k+1} = y^k + \rho(Ux^{k+1} + r + e - z^{k+1})$$

The algorithm can then be summarized as

```
for $k = 1 \ldots$ until convergence do
    $x^{k+1} = H\backslash\psi$ {from (5.20)}
    $z^{k+1} = S_{\frac{\lambda}{2\rho}}(\theta)$ {from (5.22)}
    $y^{k+1} = y^k + \rho(Ux^{k+1} + r + e - z^{k+1})$
end for
```

5.3 Results and Interpretations

In this section, we demonstrate the efficacy of the method on three sets of graphs. First, we generate spoligoforests for the CDC dataset. This dataset contains the genotypes of MTBC strains obtained from samples of the greater than 39K patients diagnosed with TB in U.S. from 2006 to 2010. This dataset was obtained from the CDC and is collected as part of TB surveillance measures applied in the U.S. A detailed description of the test set is provided in Chapter 1.

The distance and adjacency matrix for each spoligoforest was generated by a standalone spoligoforest generating tool described in Chapter 7. The adjacency matrix for the spoligoforest was constructed using the algorithm specified in [89]. The distance matrix used is based on both spoligotypes and MIRU types. Common deletions between spoligotypes are used to define a spoligotype similarity matrix, and the minimum Hamming distance between MIRUs associated with each spoligotype is used to define the *MIRU-component* of the distance matrix. The spoligotype similarity matrix and MIRU distance matrix are used in equal proportion to generate a combined distance matrix. This process is described in further detail in Chapter 7.

The inital layout produced by stress majorization was provided as the starting point to MAA+. Alternate layouts for each of these spoligoforests were also generated using three other algorithms: MDS, Laplacian Eigenmaps and Graphviz Twopi. The images of each of these spoligoforests generated by the four different methods are represented in Figure 5.2 and 5.1 and in the appendix in Figures A.7, A.9 and A.8. Nodes are colored by labels assigned by the classification tool provided by TB-Lineage. The stress and number of crossings for each of these layouts is represented in Table 5.1. Comparing the images generated using MDS alone with MAA+, we can see that MAA+ layouts have a significantly reduced number of crossings as compared with the MDS layout, with only marginal increase in stress. In contrast, other dimensionality reduction techniques have a large number of edge-crossings. Whereas conventional graph drawing methods for trees such as Graphviz Twopi that pay no attention to proximity preservation have high stress.

The ROMA graphs are used as a standard benchmark for studies in exact crossing minimization. While the motivation of this work is not necessarily to find a layout with the minimum number of crossings, we use this set of non-planar graphs to show the potential of the method to be adapted for use in laying out more general graphs. These graphs were collected from a variety of applications [32]. No distance matrix was specified for these graphs but one was constructed by setting the distance between every pair of nodes to be equal to the shortest path between them. An initial layout was computed using MDS and subsequently MAA+ was applied to generate layouts that minimize crossings while keeping stress low. A comparison of stress and number of crossings for layouts generated by MDS, MAA+, Laplacian Eigenmaps and Spring Embedding are represented in Table 5.3.

The suite of randomly generated planar graphs described in Chapter 4 was also used as a test set for MAA+. Computational results comparing the average stress and number of crossings for layouts computed by MDS, MAA+, Spring and Orthogonal Embedding are represented in Table 5.2. Each row represents results for a set of of twenty different graphs of the size specified. Results are averaged over all twenty graphs in the set. MAA+

Table 5.1: Metrics for spoligoforests generated with MAA+, MDS, Twopi and Laplacian Eigenmaps.

Spoligoforest	**|V|**	**|E|**		**MAA+**	**MDS**	**Twopi**	**Lap. Eig.**
M. africanum	82	58	stress	0.09	0.08	0.45	0.95
			# cross.	6	17	0	194
X	426	373	stress	0.08	0.05	0.47	0.41
			# cross.	186	1120	0	3497
T	706	654	stress	0.17	0.09	0.44	0.46
			# cross.	2340	12242	0	23592
East-african Indian	188	176	stress	0.20	0.11	0.48	1.05
			# cross.	126	899	0	2162
M. bovis	79	73	stress	0.08	0.08	0.42	0.64
			# cross.	9	58	0	438
LAM	652	582	stress	0.13	0.11	0.47	1.04
			# cross.	1447	6802	137	21800
Haarlem	565	513	stress	0.08	0.05	0.50	0.53
			# cross.	1093	3826	0	8416
Indo-Oceanic	687	627	stress	0.08	0.05	0.44	0.29
			# cross.	632	3863	0	6326
EuroAm-African	143	126	stress	0.10	0.09	0.44	0.66
			# cross.	29	314	0	824

Table 5.2: Comparison of average stress and crossings for MAA+, MDS, Spring and Orthogonal Embedding for 5 sets of 20 randomly generated graphs.

|V|	**|E|**	metric	**MAA+**	**MDS**	**Spring**	**Orth.**
50	80	stress	0.16±0.01	0.14±0.00	0.54±0.02	0.56±0.04
		# cross.	255.50±84.33	416.30±125.11	34.60±14.34	0.00±0.00
100	100	stress	0.15±0.01	0.12±0.01	0.70±0.03	0.68±0.03
		# cross.	362.40±125.73	683.50±205.47	39.00±16.43	0.00±0.00
120	140	stress	0.15±0.01	0.12±0.00	0.67±0.04	0.69±0.04
		# cross.	643.10±218.42	1203.15±346.49	64.10±24.83	0.00±0.00
100	120	stress	0.16±0.02	0.12±0.01	0.67±0.03	0.69±0.04
		# cross.	478.45±205.02	894.70±362.05	53.30±25.67	0.00±0.00
120	160	stress	0.16±0.01	0.12±0.00	0.65±0.04	0.67±0.05
		# cross.	907.45±295.92	1554.25±469.73	97.30±33.61	0.00±0.00

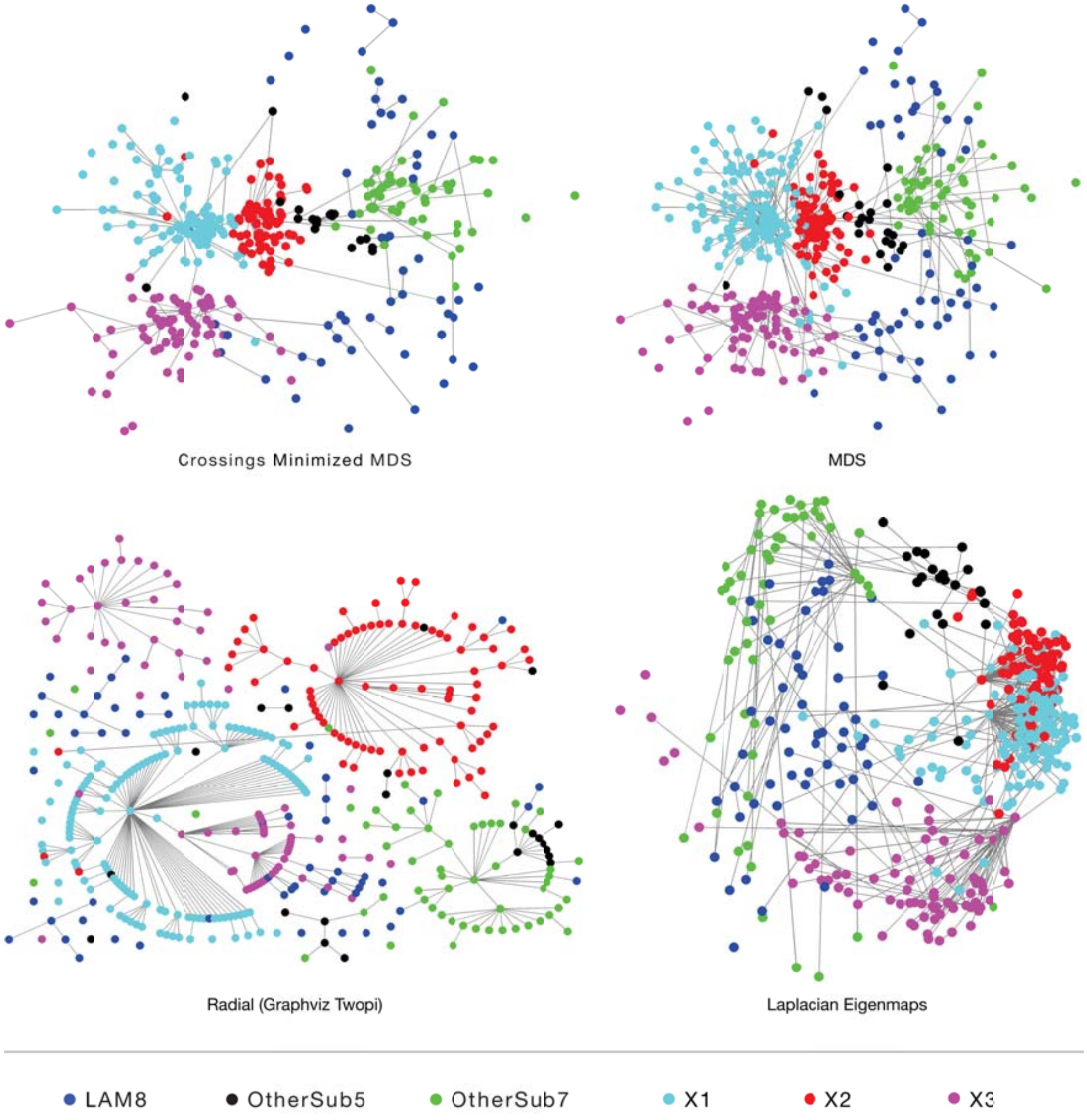


Figure 5.1: Spoligoforest for all sublineages predicted to belong to the Euro-American X lineage. Layout generated using (a) MAA+ (b)MDS (c) GraphViz Twopi (d) Laplacian Eigenmaps. MAA+ improves on MDS in number of crossings with marginal increase in stress.

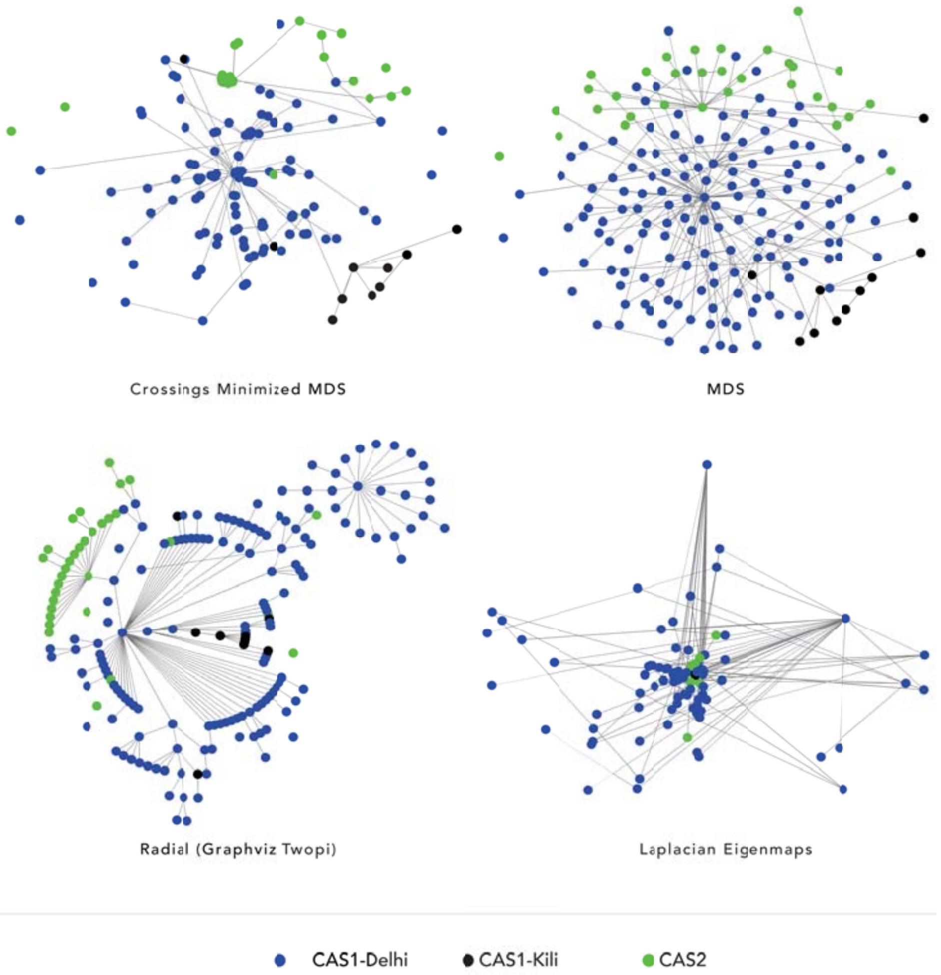


Figure 5.2: Spoligoforest for all sublineages CAS1-Delhi, CAS1-KILI and CAS2 of the East-african Indian lineage. Layout generated using (a) MAA+ (b)MDS (c) GraphViz Twopi (d) Laplacian Eigenmaps. MAA+ improves on MDS in number of crossings with marginal increase in stress.

Table 5.3: Comparison of average stress and crossings for MAA+, MDS, Spring Embedding and Laplacian Eigenmaps for 4 sets of ROMA graphs, each set ranging in size from 60 to 100 graphs.

# graphs	—V—	—E—		MAA+	MDS	Spring	Lap. Eigmap
59	50	65.42±5.42	stress	0.12±0.07	0.12±0.07	1.22±0.19	1.81±0.27
		65.42±5.42	# cross.	17.19±11.68	39.90±26.74	441.46±102.53	350.20±88.48
99	60	79.59±5.89	stress	0.11±0.05	0.11±0.05	1.20±0.13	1.92±0.22
		79.59±5.89	# cross.	27.82±17.64	64.22±39.69	647.67±127.36	592.11±127.49
81	70	94.17±6.18	stress	0.11±0.04	0.11±0.04	1.17±0.13	2.34±0.28
		94.17±6.18	# cross.	39.04±19.91	89.58±44.42	909.90±158.52	744.84±140.61
99	80	108.85±7.10	stress	0.12±0.05	0.12±0.05	1.12±0.10	2.06±0.21
		108.85±7.10	# cross.	56.74±25.17	128.93±53.86	1226.64±220.33	819.26±186.84

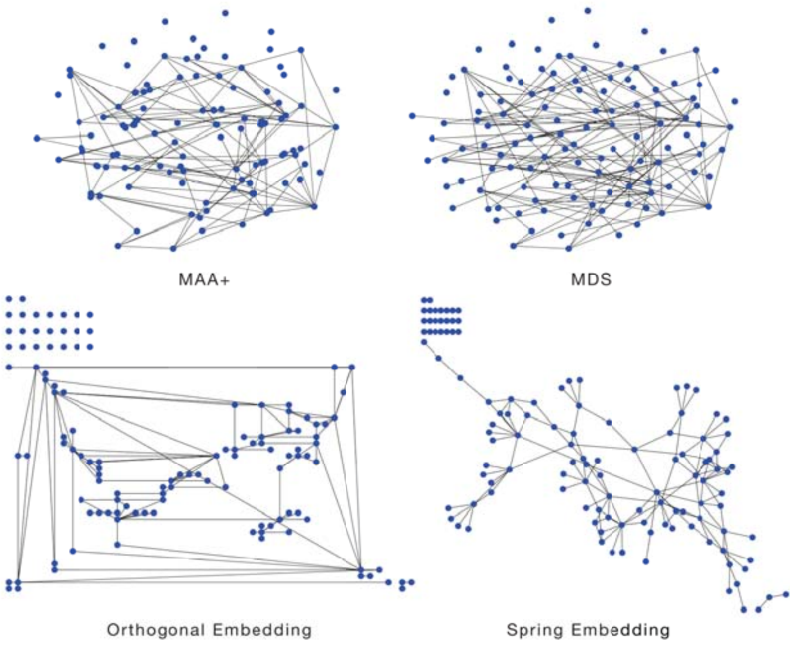


Figure 5.3: Comparison of spoligoforest layouts by MAA+, MDS, Spring and Orthogonal Embedding for randomly generated graph with 120 nodes and 140 edges. MAA+ reduces number of crossings to 431 from 1113 in MDS layout, while stress increases to 1.4 times that of MDS solution. Stress of Spring and Orthogonal Embeddings are 5.9 and 5.4 times the MDS layout.

was applied on graph layout obtained by stress majorization (MDS). MAA+ provided a drastic reduction in number of crossings with marginal increase in stress as compared to the starting layout. Spring and Orthogonal Embedding are presented as alternatives that have low number of crossings but significantly higher stress. These solutions would be undesirable in a setting where proximity preservation is an important aspect of the graph visualization. In figure 5.3, we represent layouts using these four layouts for a sample randomly generated graph with 120 nodes and 140 edges.

5.4 Discussion

ADMM is known to converge to modest accuracy quickly, but can be very slow to converge to high accuracy. Therefore, setting the right stopping criteria for ADMM poses a challenge. In practice, using a relatively low tolerance $1e^{-3}$ in each iteration of the algorithm produces better results overall.

Another consideration is the choice of norm used. L1-penalties were used in the MAA+ algorithm, while quadratic penalties were used in Chapter 4. Using L1-penalties results in an exact minimization problem as described earlier. This will lead to removal of edge-crossings but may also result in sharper increases in stress. The use of inexact penalty methods using quadratic penalties for intersections would require gradually increasing the penalty parameter in each iteration, thus requiring multiple iterations to reach the minima. The use of gentle penalties means that small changes are made in the co-ordinates of the nodes resulting in lower stress. Thus, these variations can be used to define objectives with varying emphasis on the stress and intersections between components resulting in different layouts.

The ADMM method for finding separating planes provides a significant (up to 10x) improvement in performance compared to MATLAB LP and QP solvers for each subproblem as used in Chapter 4. This is a significant improvement as a large number of these subproblems $|O(|E|^2)|$ need to be solved in every iteration. Further improvements are obtained by warm-starting the solution from that obtained in previous iterations. In practice however closed form solutions based on geometry of the problem provide even greater savings in time. These solutions while not being optimal provide a reduction in objective value during the *u-stage*. A detailed description of the closed form solutions is provided in Chapter 7.

CHAPTER 6
Analysis

In this chapter, we provide a theoretical analysis of the alternating algorithm for the nonconvex nonsmooth optimization problems. We begin by recalling some definitions and basic properties of biconvex optimization problems in 6.1. In Section 6.2, we describe a biconvex nonsmooth optimization problem that serves as a template for our constrained embedding problems of interest. We develop optimality conditions for this problem. In section 6.3, we describe an alternating strategy for solving it that exploits the convexity of its subproblems. The algorithms MAA and MAA+ in Chapters 4 and 5 are based on such an alternating strategy. We show that the alternating algorithm can produce monotonic decrease in objective value in every iteration. Convergence properties of the alternating algorithm are discussed. The analysis provided applies to the use of an alternating algorithm for the general class of nonconvex nonsmooth optimization problems. In section 6.4, we will show how MAA and MAA+ that employ alternating strategies for solving the constrained MDS problem are special cases of this alternating algorithm.

6.1 Notation and Definitions

Let us recall the following definitions from [55]. Let $X \subseteq \mathbb{R}^n$ and $U \subseteq \mathbb{R}^l$ be two non-empty convex sets and let $\mathcal{C} \subseteq X \times U$. Let $\mathcal{C}_x = \{u \in U : (x, u) \in \mathcal{C}\}$ and $\mathcal{C}_u = \{x \in X : (x, u) \in \mathcal{C}\}$

Definition 1. ***Biconvex set:*** *The set $\mathcal{C}$ is a biconvex set, if $\mathcal{C}_x$ is a convex set for every $x \in X$ and $\mathcal{C}_u$ is a convex set for every $u \in U$.*

Definition 2. ***Biconvex function:*** *A function $f : \mathcal{C} \to \mathbb{R}$ on a biconvex set $\mathcal{C}$ is called a biconvex function on $\mathcal{C}$, if the function $f(\bar{x}, u)$ is a convex function on $\mathcal{C}_{\bar{x}}$ for every fixed $\bar{x} \in X$, and $f(x, \bar{u})$ is a convex function on $\mathcal{C}_{\bar{u}}$ for every fixed $\bar{u} \in U$.*

Definition 3. ***Biconvex optimization problem:*** *An optimization problem of the form*

$$\underset{x,u}{minimize}\{f(x, u) : (x, u) \in \mathcal{C}\}$$

is said to be a biconvex optimization problem, if the objective function is biconvex on $\mathcal{C}$, where $\mathcal{C}$ is a biconvex set on $X \times U$.

See [55] for more detailed definitions and properties of biconvex problems.

We now introduce some notation and recall some definitions from [87] used to describe the optimality conditions in this chapter.

Definition 4. ***Subgradient:*** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function and let $x \in dom\ f$. A vector $g \in \mathbb{R}^n$ is called the subgradient of f at x, if*

$$f(y) \geq f(x) + < g, y - x > \forall y \in \mathbb{R}^n$$

Definition 5. ***Subdifferential:*** *The set of all subgradients of $f : \mathbb{R}^n \to \mathbb{R}$ at x is called the* subdifferential *of f at x and is denoted by $\partial f(x)$*

The subgradient generalizes the concept of the gradient as applied to convex functions that are not necesarily differentiable. For a continuously differentiable function, the subdifferential consists of a single unique element, the gradient, denoted by $\nabla f(x)$. The plus function introduced in previous chapters is used to denote the component-wise maximum of the vector x and 0 as:

$$x_+ = max(x, 0)$$

We denote the subdifferential $D_x(x_+)$ of x_+ as the set of subgradients of x_+ taken with respect to x. The set is defined as

$$D_x(x_+) = \{\delta \in \mathbb{R}^n\},$$

where for each element δ of the set, the ith component of the vector δ is set based on the ith component of x, as follows

$$\begin{aligned} \delta_i &= 1 \text{ if } x_i > 0 \\ &\in [0, 1] \text{ if } x_i = 0 \\ &= 0 \text{ if } x_i < 0 \end{aligned}$$

Further, optimality conditions for constrained optimization problems are subject to the satisfaction of certain regularity conditions or constraint qualifications (CQ). One example of a CQ is Slater's constraint qualification. If $X = \{x | g(x) \leq 0\}$ defines the convex feasible region, then the function g is said to satisfy *Slater's CQ* if $\exists \bar{x}$ such that $g(\bar{x}) < 0$. Another example of a CQ is Robinson's condition which is equivalent to metric regularity. A sufficient condition for metric regularity at x_0 is that the gradients of all equality constraints are linearly independent and there exists an interior point x_M, such that for all inequality constraints $g_i(x) = 0$, $< \nabla g_i(x_0), x_M - x_0 >< 0$ and for all equality constraints $h_i(x) = 0$, $< \nabla h_i(x_0), x_M - x_0 >= 0$ [87].

6.2 Optimality Conditions

Let us now consider a nonconvex nonsmooth optimization problem of the following form:

$$\begin{aligned} \underset{x,u}{\text{minimize}}\; f(x,u) = \quad & \phi(x) + \sum_{i=1}^{m} (p_i(x,u))_+ \\ \text{subject to} \qquad\quad & r_i(u) \leq 0,\ i = 1..m \end{aligned} \tag{6.1}$$

where $\phi(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ is a continuous function, differentiable in x and bounded below. Further, $\phi(x)$ is a coercive function i.e. $\lim_{k \rightarrow +\infty} \phi(x_k) = +\infty$ whenever $||x_k|| \rightarrow +\infty$. We assume, that $\phi(x)$ is not necessarily convex. We assume, $r_i(u) : \mathbb{R}^l \rightarrow \mathbb{R}$ is a continuous function, differentiable in u, and is strictly convex. $p_i(x,u)$ is a biconvex function on $\mathcal{C} \subseteq X \times U$; it is continuous and differentiable in x and u. Overall, this function is nonsmooth, with nonsmoothness arising from the *plus* function applied to the penalty terms. Constrained embedding problems with biconvex constraints in $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^l$ imposed as penalties can be expressed in the form of problem 6.1. In the following sections, we will discuss this formulation in the context of constrained MDS. Similarly, $\phi(x)$ may be a second order approximation of any embedding objective.

We define the following convex subproblems obtained by fixing one variable and minimizing with respect to the other.

x-subproblem: The Problem (6.1) for a fixed value of $u = u^*$:

$$\underset{x}{\text{minimize}} \quad f(x,u^*) = \phi(x) + \sum_{i=1}^{m} (p_i(x,u^*))_+ \tag{6.2}$$

u-subproblem: The Problem (6.1) for a fixed value of $x = x^*$:

$$\begin{aligned} &\underset{u}{\text{minimize}} \quad f(x^*, u) = \sum_{i=1}^{m} (p_i(x^*, u))_+ \\ &\text{subject to} \quad r_i(u) \leq 0, \; i = 1..m \end{aligned} \tag{6.3}$$

Definition 6. ***Partial Optimum:*** *A point (x^*, u^*) is called a partial optimum of a biconvex function f on $\mathcal{C}$, if*

$$f(x^*, u^*) \leq f(x, u^*) \; \forall x \in \mathcal{C}_{u^*} \; \textit{and} \; f(x^*, u^*) \leq f(x^*, u) \; \forall u \in \mathcal{C}_{x^*}$$

We now develop optimality conditions for problem (6.1). For this we transform the optimality conditions defined for a problem with a differentiable objective and possibly nonsmooth but convex constraints in [87] to apply to problem with a nonconvex nonsmooth objective:

Theorem 1. ***First Order Necessary Conditions for optimality of problem (6.1):*** *Assume, $z^* = (x^*, u^*)$ is a minimum of problem (6.1) and a constraint qualification is satisfied, then $\exists \alpha_i$, such that*

$$0 = \nabla_x \phi(x^*) + \sum_{i=1}^{m} \hat{\delta}_i \frac{\partial p_i(x^*, u^*)}{\partial x} \tag{6.4}$$

$$0 = \sum_{i=1}^{m} \tilde{\delta}_i \frac{\partial p_i(x^*, u^*)}{\partial u} + \alpha_i \nabla r_i(u^*) \tag{6.5}$$

$$r_i(u^*) \leq 0$$

$$\alpha_i r_i(u^*) = 0, \; i = 1..m, \; \alpha \geq 0. \tag{6.6}$$

$$\textit{where } \hat{\delta}_i \in D_x(p_i(x, u)) \textit{ and } \tilde{\delta}_i \in D_u(p_i(x, u))$$

Proof. To obtain the above optimality conditions, we transform the problem to an equivalent problem with a differentiable objective. We introduce new variables v_i and s_i to replace the nondifferentiable terms in the objective.

$$\begin{aligned}
\underset{x,u}{\text{minimize}} \quad & f(x,u) = \phi(x) + \sum_{i=1}^{m} v_i \\
\text{subject to} \quad & r_i(u) \leq 0,\ i = 1..m \\
& s_i = p_i(x,u),\ i = 1..m \\
& (s_i)_+ \leq v_i,\ i = 1..m
\end{aligned} \tag{6.7}$$

Problem (6.7) is equivalent to (6.1). For every feasible point (x, u) in problem (6.1), the variables $x, u, s_i = p_i(x, u), v_i \geq s_i$ are feasible in this problem, and the objective values of both problems are equal. If $x^*, u^*, s_i^* = p_i(x^*, u^*), s_{i+}^* \leq v_i^*$ is a minimum of the Problem (6.7), then x^*, u^* is a solution of (6.1) with objective value of problem (6.1) equal to objective value of Problem (6.7). We can derive the optimality conditions for (6.1) from the optimality conditions of Problem (6.7) which are as follows: (x^*, u^*) is a minimum of the following problem, if Robinson's Constraint Qualification holds and $\exists \alpha \geq 0, \mu_i \geq 0$

$$\left.\begin{aligned}
r_i(u^*) &\leq 0 \\
s_i^* &= p_i(x^*, u^*) \\
s_{i+}^* &\leq v_i^*
\end{aligned}\right\} \text{Primal feasibility}$$

$$\left.\begin{aligned}
\alpha_i r_i(u^*) = 0,\ i = 1..m,\ \alpha_i \geq 0. \\
\mu_i(s_{i+} - v_i) = 0,\ i = 1..m,\ \mu_i \geq 0.
\end{aligned}\right\} \text{Complementarity}$$

$$\left.\begin{aligned}
0 = \nabla_x \phi(x^*) + \sum_{i=1}^{m} \lambda_i \frac{\partial p_i(x^*, u^*)}{\partial x} \\
0 = + \sum_{i=1}^{m} \lambda_i \frac{\partial p_i(x^*, u^*)}{\partial u} + \alpha_i \nabla r_i(u^*) \\
0 = 1 - \mu_i,\ i = 1..m, \\
0 = -\lambda_i + \mu_i \delta_i,\ i = 1..m, \\
\text{where } \delta \in D_s(s_+)
\end{aligned}\right\} \text{Dual feasibility}$$

By substitution of $\mu_i = 1$ and $\delta_i = \lambda_i$ we obtain the FONC for optimality of (6.1) listed above. □

We now make some observations about the nature of problem (6.1) based on FONC for optimality.

Theorem 2. ***Partial optimality $\implies$ FONC satisfied, if problem(6.1) biconvex:*** *If $z^* = (x^*, u^*)$ is a partial optimum of f on $\mathcal{C}$, $\exists alpha$ such that (x^*, u^*) satisfies FONC in Theorem 1.*

Proof. By definitions of *x-subproblem* in (6.2), x^* is a minimum of the *x-subproblem*, if and only if condition (6.4) is satisfied. Similarly by definitions of *u-subproblem*, u^* is a minimum of the *u-subproblem* and Slater's Constraint Qualification is satisfied, if $\exists alpha$, such that conditions (6.5) and (6.6) are satisfied. Therefore, since any partial optimum point $z^* = (x^*, u^*)$ satisfies optimality conditions for *x-subproblem* (6.2) and *u-subproblem* (6.3), it satisfies optimality conditions (6.4), (6.5) and (6.6), therefore it satisfies all the FONC for optimality defined in Theorem 1. □

Theorem 3. ***If problem (6.1) biconvex, FONC satisfied $\implies$ partial optimality:*** *If $f(x, u)$ is biconvex, every point that satisfies FONC of problem (6.1) as defined in Theorem 1 is a partial optimum of f on $\mathcal{C}$.*

Proof. By definition of biconvex functions, for fixed u^*, the function $f(x, u^*) : \mathcal{C}_{u^*} \to \mathbb{R}$ is convex. Therefore, $\exists d \in \frac{\partial f(x^*,u^*)}{\partial x}$ and $\exists b \in \frac{\partial f(x^*,u^*)}{\partial u}$

$$f(x^*, u^*) + d'(x - x^*) \leq f(x, u^*) \; \forall x \in \mathcal{C}_{u^*}$$

From condition (6.4), $0 \in \frac{\partial f(x^*,u^*)}{\partial x}$, therefore

$$f(x^*, u^*) \leq f(x, u^*) \; \forall x \in \mathcal{C}_{u^*}$$

Similarly, from convexity of function $f(x^*, u) : \mathcal{C}_{x^*} \to \mathbb{R}$ for fixed x^*, we have

$$f(x^*, u^*) + b'(u - u^*) \leq f(x^*, u) \; \forall u \in \mathcal{C}_{x^*}$$

From conditions (6.5) and 6.6, $-\sum_{i=1}^{m} \alpha_i \nabla r_i(u^*) \in \frac{\partial f(x^*,u^*)}{\partial u}$, therefore

$$f(x^*, u^*) + (-\sum_{i=1}^{m} \alpha_i \nabla r_i(u^*))'(u - u^*) \leq f(x^*, u) \; \forall u \in \mathcal{C}_{x^*}$$

By convexity and differentiability of each constraint $r_i(u) \leq 0 \; i = 1..m$,

$$\begin{aligned}
\nabla r_i(u^*)'(u-u^*) &\leq r_i(u) - r_i(u^*) \; i = 1..m \\
-\sum_{i=1}^{m} \alpha_i \nabla r_i(u^*)'(u-u^*) &\geq \sum_{i=1}^{m} \alpha_i r_i(u^*) - \sum_{i=1}^{m} \alpha_i r_i(u) \\
&= -\sum_{i=1}^{m} \alpha_i r_i(u) \text{ from complementarity} \\
&\geq 0 \text{ Since } \alpha_i \geq 0 \text{ and } r_i(u) \leq 0 \; \forall i
\end{aligned}$$

Therefore, we have $f(x^*, u^*) \leq f(x^*, u) \; \forall u \in \mathcal{C}_{x^*}$. Therefore, by definition of partial optimality (x^*, u^*) is a partial optimum. □

From theorems 2 and 3, we have the following:

Corollary 1. ***If*** $f(x,u)$ ***biconvex, FONC satisfied*** $\Leftrightarrow$ ***partial optimality for problem (6.1):*** *If* $f(x,u)$ *of problem (6.1) is biconvex, a point* $z^* = (x^*, u^*) \in \mathbb{R}^{n+l}$ *satisfies its FONC if and only if it is a partial optimum.*

6.3 Alternating Algorithm for Nonconvex Nonsmooth Optimization Problems

In this section, we develop an algorithm for minimizing a nonconvex nonsmooth function like Problem (6.1), with the nonconvex objective with additional biconvex penalty terms. The constrained embedding problem for MDS with biconvex edge-crossing penalties is an example of this type of mathematical program. Instead of using general global optimization methods for the problem directly, we define a related biconvex optimization problem called the *auxiliary problem*, and we develop an algorithm that exploits convex substructures of the auxiliary problem. Let us first define the following two programs, both of the form defined in problem (6.1)

$$\left.\begin{array}{lll}
\underset{x,u}{\text{minimize}} \; \tilde{f}(x,u) = & \tilde{\phi}(x) + \sum_{i=1}^{m} (p_i(x,u))_+ \\
\text{subject to} & r_i(u) \leq 0, \; i = 1..m
\end{array}\right\} \text{Original problem} \qquad (6.8)$$

This problem is identical to Problem (6.1) with the additional assumption that $\tilde{\phi}(x)$ is a differentiable but nonconvex function of x. Specifically, $\tilde{\phi} : \mathbb{R}^n \to \mathbb{R}$ is nonconvex, continuous, differentiable in x, coercive and bounded below. $r_i(u),\ i = 1..m$ is a strictly convex function, continuous and differentiable in u. $p_i(x,u)$ is a biconvex function on $\mathcal{C} \subseteq X \times U$; it is continuous and differentiable in x and u.

Now, consider the following auxiliary problem:

$$\left.\begin{array}{lll} \underset{x,u}{\text{minimize}}\ \hat{f}(x,u,s) = & \hat{\phi}(x,s) + \displaystyle\sum_{i=1}^{m}(p_i(x,u))_+ \\ \text{subject to} & r_i(u) \leq 0,\ i = 1..m \end{array}\right\}\text{Auxiliary problem} \tag{6.9}$$

This problem is identical to Problem (6.1) with the additional assumption that $\hat{f}(x,u,s)$ is biconvex and differentiable in x and u for any fixed s. Moreover, we have $\hat{\phi}(x,\bar{s})$ is strictly convex for fixed $\bar{s}, \forall x \in X$. $\hat{\phi}(x,s)$ is a coercive function, it is continuous and bounded below. $p_i(x,u)$ and $r_i(u)$ are defined exactly as in the original problem 6.8.

Further, let us define the assumptions that define the relation between $\hat{f}$ and $\tilde{f}$.

Definition 7. ***Majorization Assumption:*** *We say the auxiliary function $\hat{f}(x,u,s)$ majorizes the function $\tilde{f}(x,u)$ when the following hold:*

$$\tilde{f}(x,u) = \hat{f}(x,u,x)$$
$$\tilde{f}(x,u) < \hat{f}(x,u,s)\ \forall x,s,s \neq x$$

By this assumption, the function $\hat{f}$ provides an upper bound on $\tilde{f}$. From equality of $\hat{f}(x,u,s)$ and $\tilde{f}(x,u)$ at $s = x$, we also have $\frac{\partial \tilde{f}(x,u)}{\partial x} = \frac{\partial \hat{f}(x,u,x)}{\partial x}$ and $\frac{\partial \tilde{f}(x,u)}{\partial u} = \frac{\partial \hat{f}(x,u,x)}{\partial u}$, i.e. $\hat{f}$ is tangential to $\tilde{f}$ at the point (x,u,x).

Lemma 1. ***Equivalance of FONC of original and auxiliary problems*** *If point (x^*,u^*,x^*) satisfies FONC for optimality of problem (6.9) then (x^*,u^*) also satisfies FONC for problem (6.8).*

Proof. From the majorization assumption, we know that the gradient of the auxiliary function $\hat{f}(x_k,u_k,x_k)$ taken at x_k is equal to the gradient of the original function $\tilde{f}(x_k,u_k)$. Therefore, from Theorem 1, we can see that the FONC conditions for the two problems: the original problem (6.8) and the auxiliary function (6.9), majorized at x^* are equivalent. □

The algorithm performs alternating minimization over the variables x and u as demonstrated in [109] for the generic class of problems with nonseparable objective functions. MAA and MAA+ algorithms for solving constrained embedding described in subsequent sections are based on such an alternating strategy. The algorithm performs iterative updates as follows:

$$u_{i+1} = \underset{u}{\operatorname{argmin}} \hat{f}(x, u_i, x_i)$$
$$x_{i+1} = \underset{x}{\operatorname{argmin}} \hat{f}(x, u_{i+1}, x_i)$$

In this section, we discuss convergence results of the alternating strategy. For this we first define some properties of functions $\tilde{f}$ and $\hat{f}$. Let $\mathcal{C} \subseteq \mathbb{R}^{n \times l}$, and let $z_i = (x_i, u_i)_{i \in \mathbb{N}} \in \mathcal{C}$ be the sequence of points obtained by the alternating algorithm. Let both the optimization subproblems arising at each iteration i be solvable. Then,

Lemma 2. ***Every step of alternating algorithm is a descent step:***

$$\hat{f}(x_{i+1}, u_{i+1}, x_i) < \hat{f}(x_i, u_i, x_i)$$

where i and $i+1$ are successive iterations in the alternating algorithm.

Proof. Consider iteration i of the algorithm such that $x_i \neq x_{i+1}$ and $u_i \neq u_{i+1}$. Let (x_i, u_i, x_i) and (x_{i+1}, u_{i+1}, x_i) not be partial optimum points of the auxiliary problem (6.9). By strict convexity of the objective function of the *x-subproblem* of problem (6.9), we have $\hat{f}(x_{i+1}, u_i, x_i) < \hat{f}(x_i, u_i, x_i)$. And by convexity of the objective function of the *u-subproblem* of problem (6.9), we have $\hat{f}(x_{i+1}, u_{i+1}, x_i) \leq \hat{f}(x_{i+1}, u_i, x_i)$. Therefore,

$$\hat{f}(x_{i+1}, u_{i+1}, x_i) < \hat{f}(x_i, u_{i+1}, x_i) \leq \hat{f}(x_i, u_i, x_i)$$

□

Theorem 4. ***Monotonically decreasing function values:*** *Let $\{\tilde{f}(z_i)\}_{i \in \mathbb{N}}$ be the sequence of function values generated by the alternating algorithm, where $z_i = (x_i, u_i)$, then $\tilde{f}(z_{i+1}) < \tilde{f}(z_i)$ $i = 1, 2....$*

Proof. Consider iteration i of the algorithm. Let $x_i \neq x_{i+1}$ and $u_i \neq u_{i+1}$, and (x_i, u_i, x_i) and (x_{i+1}, u_{i+1}, x_i) not be partial optimum points of $\hat{f}(x, u, x_i)$. From the majorization assumption we know,

$$\hat{f}(x_i, u_i, x_i) = \tilde{f}(x_i, u_i). \tag{6.10}$$

Let $z_i = (x_i, u_i)$ and $z_{i+1} = (x_{i+1}, u_{i+1})$. Lemma 2 shows $\hat{f}(x_{i+1}, u_{i+1}, x_i) < \hat{f}(x_i, u_i, x_i)$, since $\hat{f}$ is biconvex, the *u-subproblem* is convex and the *x-subproblem* is strictly convex. From Lemma (2) and Equation (6.10) obtained from the majorization assumption, we have, $\hat{f}(x_{i+1}, u_{i+1}, x_i) < \tilde{f}(x_i, u_i)$. Further from majorization assumption, we have $\tilde{f}(x_{i+1}, u_{i+1}) \leq \hat{f}(x_{i+1}, u_{i+1}, x_i)$ since $x_i \neq x_{i+1}$. Therefore, combining the two inequalities, $\tilde{f}(z_{i+1}) < \tilde{f}(z_i)$. Thus, the sequence of function values generated by the alternating algorithm are monotonically decreasing. □

Theorem 5. ***Function values converge:*** *The sequence of function values* $\{f(z_i)\}_{i\in\mathbb{N}}$ *generated by the alternating algorithm converges monotonically.*

Proof. Function $\tilde{f}$ is bounded below by assumption. The sequence of function values generated by the alternating algorithm are monotonically decreasing as seen in Theorem 4. Therefore, the sequence $\{\tilde{f}(z_i)\}_{i\in\mathbb{N}}$ converges to a limit value $a \in \mathbb{R}$. □

Lemma 3. ***Existence of at least one accumulation point:*** *The sequence* $\{z_i\}_{i\in\mathbb{N}}$ *generated by the alternating algorithm has at least one accumulation point.*

Proof. The sequence $\{z_i\}_{i\in\mathbb{N}}$ generated by the alternating algorithm is contained in a compact set. For a coercive function, such as $\tilde{\phi}$, every non-empty lower level set $L_c = \{x|\tilde{\phi}(x) \leq c\}$ is bounded. The constraint $r_i(u) \leq 0$ restricts the set of feasible u to a compact set. Similarly, if we consider the variant of Problem 6.8 with $||u||^2$ in the objective rather than constrained, the corresponding functions $\tilde{\phi}(x)$ and $\tilde{\theta}(u)$ are coercive and have bounded level sets. Therefore, by the Bolzano-Weierstrass theorem, the sequence $\{z_i\}_{i\in\mathbb{N}}$ has at least one accumulation point $z^* \in \mathcal{C}$. □

Let us now introduce the concept of an algorithmic map as used in [9] to prove convergence. We restate the definition as applied to the alternating algorithm:

Definition 8. ***Algorithmic Map*** *Let* $\mathcal{C} \subseteq \mathbb{R}^{n\times l}$ *and let* $f : \mathcal{C} \to \mathbb{R}$. *Let* $z_i = (x_i, u_i) \in \mathcal{C}$ *for* $i = 1, 2, ...$ *The map* $\mathcal{M} : \mathcal{C} \to \mathcal{C}$ *defined by* $z_{i+1} \in \mathcal{M}(z_i)$ *is called the algorithmic map iff* $\hat{f}(x_k, u_{k+1}, x_k) \leq \hat{f}(x_k, u, x_k)$ $\forall u \in \mathcal{C}_{x_k}$ *and* $\hat{f}(x_{k+1}, u_{k+1}, x_k) \leq \hat{f}(x, u_{k+1}, x_k)$ $\forall x \in \mathcal{C}_{u_{k+1}}$.

Each iteration of the alternating algorithm is therefore a transformation from z_i to z_{i+1}, and the algorithm is an iterative selection of points $z_{i+1} \in \mathcal{M}(z_i)$. Thus, the alternating algorithm is an iterative application of map $\mathcal{M}$.

Further, as X and U are closed sets, and $\hat{f}$ and $\tilde{f}$ are continuous functions, we show the algorithmic map $\mathcal{M}$ is closed on $X \times U$.

Lemma 4. ***Closed map***

$$\left.\begin{array}{ll} z_i = (x_i, u_i) \in X \times U, & (x_i, u_i) \to (x^*, u^*) = z^* \\ z_i' = (x_i', u_i') \in \mathcal{M}(z_i), & (x_i', u_i') \to (\bar{x}', \bar{u}') = \bar{z}' \end{array}\right\} \implies \bar{z}' \in \mathcal{M}(z^*).$$

Proof. $z_i' \in \mathcal{M}(z_i)\ \forall i \in \mathbb{N} \implies \hat{f}(x_i, u_i', x_i) \leq \hat{f}(x_i, u, x_i)\ \forall u \in U$ and $\hat{f}(x_i', u_i', x_i) \leq \hat{f}(x, u_i', x_i)\ \forall x \in X$. Due to continuity of $\hat{f}$, this would be true $\forall i$. Taking limits, $\lim_{i\to+\infty} \hat{f}(x_i, u_i', x_i) = (x^*, \bar{u}', x^*)$ and $\lim_{i\to+\infty} \hat{f}(x_i, u, x_i) = (x^*, u, x^*)$. Therefore,

$$x^*, \bar{u}', x^*) \leq (x^*, u, x^*).$$

Similarly,

$$\hat{f}(\bar{x}', \bar{u}', \bar{x}') = \lim_{i\to+\infty} \hat{f}(x_i', u_i', x_i') \leq \lim_{i\to+\infty} \hat{f}(x_i, u_i', x_i) = (x^*, \bar{u}', x^*)$$

Hence, by definition of the algorithmic map, $\bar{z}' \in \mathcal{M}(z^*)$. □

Further, we can make the following conclusions regarding convergence of the algorithm and optimality of solution obtained.

Theorem 6. ***Accumulation point*** $\implies$ ***FONC*** *Every accumulation point $z^* = (x^*, u^*)$ of the sequence $\{z_i\}_{i\in\mathbb{N}}$ is a partial optimum of auxiliary function $\hat{f}(x, u, x^*)$ and satisfies FONC for Optimality for original problem defined in Theorem 1.*

Proof. The detailed proof of the first part of the statement follows from theorems pertaining to general biconvex problems as discussed in survey in [55]. Here we reproduce a proof sketch from [55]. From Lemma 3, the sequence $\{z_i\}_{i\in\mathbb{N}}$ is contained in a compact set and it has at least one accumulation point. Therefore, there exists a convergent subsequence $\{z_k\}_{k\in\mathcal{K}}$ that converges to z^*. Similarly, as the subsequences $\{z_k\}_{k\in\mathcal{K}+\infty}$ and $\{z_k\}_{k\in\mathcal{K}-\infty}$ are also contained in compact sets, they have accumulation points z^+ and z^-, respectively. As $\tilde{f}$ is continuous, and the algorithmic map $\mathcal{M}$ is closed, we know $z^+ \in \mathcal{M}(z^*)$ and $f(z^+) = f(z^*)$, and also $z^* \in \mathcal{M}(z^-)$ and $f(z^-) = f(z^*)$. Therefore, z^* must be a partial optimum. Otherwise by Theorem 4 showing strictly decreasing function values, we would have $f(z^*) < f(z^-)$ or $f(z^*) < f(z^+)$, and therefore have a contradiction. From Theorem 2, every partial optimum of the auxiliary problem satisfies FONC. □

Theorem 7. ***Convergence of sequence $\implies$ partial optimum of auxiliary problem:*** *If the sequence $\{z_i\}_{i\in\mathbb{N}}$ generated by the alternating algorithm converges to z^*, then z^* is a partial optimum of auxiliary function $\hat{f}(x, u, x^*)$.*

Proof. This can be seen from Lemma 4, as the algorithmic map $\mathcal{M}$ is closed, $z_{i+1} \in \mathcal{M}(z_i)$ for all points in the sequence $\{z_i\}_{i\in\mathbb{N}}$. We have $z^* \in \mathcal{M}(z^*)$. Hence,

$$\hat{f}(x^*, u^*, x^*) \leq \hat{f}(x, u^*, x^*),\ \forall x \in X \text{ and } \quad \hat{f}(x^*, u^*, x^*) \leq \hat{f}(x^*, u, x^*),\ \forall u \in U$$

Therefore by definition of partial optimality, z^* is a partial optimum. □

Theorem 8. ***Sequence convergence and FONC for original problem (6.8)*** *If $(x_k, u_k) = (x_{k+1}, u_{k+1})$, FONC for optimality of original problem in (6.8) are satisfied.*

Proof. If sequence $\{z_i\}_{i\in\mathbb{N}}$ converges, from Theorem 7, we know (x_k, u_k, x_k) is a partial optimum point for $\hat{f}(x, u, x_k)$. From Theorem 2, partial optimum of a biconvex problem such as auxiliary problem 6.9 also satisfies its FONC. From Lemma 1, showing equivalence of FONC for optimality of auxiliary and original problems, we can conclude that FONC for optimality of (6.8) are satisfied. □

Let us define Ω as the set of all solution points that satisfy FONC defined in Theorem 1. Combining the above theorems we can make the following conclusions.

Corollary 2. *Either the sequence of points generated by the algorithm converges in a finite number of steps with $(x_k, u_k) = (x_{k+1}, u_{k+1})$, and $(x_k, u_k) \in \Omega$, or it generates an infinite sequence $\{z_k\}$, where $z_k = (x_k, u_k)$, such that i) All accumulation points of $\{z_k\}$ belong to Ω ii) $\tilde{f}(x_k, u_k) \to \tilde{f}(\bar{x}, \bar{u})$ for some $(\bar{x}, \bar{u}) \in \Omega$.*

This statement can also be made using Zangwill's convergence theorem [111] based on the following properties of $\tilde{f}$ and the algorithmic map $\mathcal{M}$: (a) $\tilde{f}$ is a continuous function (b) $\tilde{f}$ is a descent function i.e. there is a strict decrease in function value at every iteration as seen in Lemma 4, and (c) The algorithmic map $\mathcal{M}$ is closed as shown in Theorem 4

6.4 Discussion: Alternating Algorithms MAA and MAA+ for Constrained MDS

In this section, we will show that MAA and MAA+ which employ alternating strategies for solving the constrained MDS problem are special cases of the general alternating algorithm described in the previous section. We first show that the MDS objective augmented with biconvex penalty terms for edge crossings follows the template of the original problem (6.8). Similarly, the corresponding auxiliary problem matches the template of program (6.9). We show that the majorization assumption defining the relation between the MDS objective augmented with penalties and an auxiliary function. Therefore, all convergence properties described in Section 6.3 apply to MAA and MAA+ for solving constrained MDS.

Let us first recall the original MDS objective subject to edge-crossing penalties. Let $X \in \mathbb{R}^{n\times 2}$ represent a configuration of points, with X_{i1} and X_{i2} representing the x and y coordinates of point i. Let δ be the pairwise distance matrix. For convenience, let us define function $p(X,U)$ representing all penalty terms for edge-crossings determined for layout given by X.

$$\begin{aligned}\sigma(X,U) &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij}-d_{ij}(X))^2 + p(X,U)\\ &= \frac{1}{2}\Big(\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij})^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}\delta_{ij}d_{ij}(X)\\ &+ \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(d_{ij}(X))^2\Big) + p(X,U)\end{aligned}$$

In order to write the above equation in matrix notation, note that we can express $d_{ij}^2(X)$ as follows

$$\begin{aligned}d_{ij}^2(X) &= (X_i - X_j)(X_i - X_j)'\\ &= X'(e_i - e_j)(e_i - e_j)'X\\ &= tr(X'E_{ij}X)\end{aligned}$$

where e_i is the ith column of the identity matrix and $E_{ij} = (e_i - e_j)(e_i - e_j)'$. Including the multiplier w_{ij} and summing over all i, j, we can define the following matrices

$$\bar{L}^w_{i,j} = \begin{cases} -w_{ij} & i \neq j,\\ \sum_{i\neq j} w_{ij} & i = j \end{cases}$$

and

$$\bar{L}_{i,j}(X) = \begin{cases} \frac{-w_{ij}\delta_{ij}}{d_{ij}(X)} & i \neq j, \\ -\sum_{i \neq j} \bar{L}_{i,j}(X) & i = j \end{cases}$$

Therefore, the stress function can be represented as follows:

$$\tilde{\sigma}(X,U) = \eta_\delta^2 - tr\big(X'\bar{L}(X)X\big) + \frac{1}{2}tr\big(X'\bar{L}^w X\big) + p(X,U) \tag{6.11}$$

where

$$\eta_\delta^2 = \sum_{i=1}^{n}\sum_{j=i}^{n} w_{ij}(\delta_{ij})^2$$

Similar to the strategy introduced in the seminal work by De Leeuw reproduced in [30], we can now define an auxiliary function $\hat{\sigma}(X,U,\hat{X})$ and show that it provides an upper bound on $\tilde{\sigma}(X,U)$.

$$\hat{\sigma}(X,U,\hat{X}) = \eta_\delta^2 - tr\big(X'\bar{L}(\hat{X})\hat{X}\big) + \frac{1}{2}tr\big(X'\bar{L}^w X\big) + p(X,U) \tag{6.12}$$

We now show that $\hat{\sigma}(X,U,\hat{X})$ majorizes $\tilde{\sigma}(X,U)$. This follows from applying Cauchy-Schwartz inequality to each pair of vectors $X_i - X_j$ and $\hat{X}_i - \hat{X}_j$. Recall the Cauchy-Schwartz inequality for any two vectors s and t is:

$$s't \leq ||s||||t||$$

Applied to vectors $X_i - X_j$ and $\hat{X}_i - \hat{X}_j$ we have,

$$\begin{aligned} (X_i - X_j)'(\hat{X}_i - \hat{X}_j) &\leq ||X_i - X_j||||\hat{X}_i - \hat{X}_j|| \\ tr(X'E_{ij}\hat{X}) &\leq ||X_i - X_j||||\hat{X}_i - \hat{X}_j|| \\ \frac{tr(X'E_{ij}\hat{X})}{||\hat{X}_i - \hat{X}_j||} &\leq ||X_i - X_j|| = \frac{tr(X'E_{ij}X)}{||X_i - X_j||} \\ \frac{1}{||\hat{X}_i - \hat{X}_j||}tr(X'E_{ij}\hat{X}) &\leq \frac{1}{||X_i - X_j||}tr(X'E_{ij}X) \end{aligned}$$

Multiplying both sides by w_{ij}, and summing over all i, j,

$$tr(X'\bar{L}(\hat{X})\hat{X}) \leq tr(X'\bar{L}(X)X)$$

Therefore, *for any* $\hat{X}$,

$$\hat{\sigma}(X, U, \hat{X}) \geq \tilde{\sigma}(X, U) \tag{6.13}$$

with equality when $X = \hat{X}$

For convenience let us define the following notation: x is X represented as a vector, $L^w = \begin{bmatrix} \bar{L}^w & \bar{0} \\ \bar{0} & \bar{L}^w \end{bmatrix}$ and $L(X) = \begin{bmatrix} \bar{L}(X) & \bar{0} \\ \bar{0} & \bar{L}(X) \end{bmatrix}$ and $\bar{0}$ is a matrix of zeros of appropriate size. The stress function augmented with penalty terms can be rewritten as

$$\begin{aligned} \underset{x,u}{\text{minimize}} \tilde{f}(x,u) = \quad & \frac{1}{2}x'L^w x - x'L(x)x + \sum_{i=1}^{m} \rho_i[(-u'A^i x + 1)_+ + (u'B^i x + 1)_+] \\ \text{subject to} \quad & ||R^i u||^2 \leq \delta, \; i = 1..m. \end{aligned} \tag{6.14}$$

The penalty function $p(X, U)$ is expressed using matrices A^i and B^i are matrices in $\mathbb{R}^{(3m+1)\times(2n+1)}$ that serve as indicator matrices. They select appropriate elements of u defining the separating plane for the ith crossing, and elements of x corresponding to the nodes of the edges in the ith crossing. For convenience of notation, assume the x and u vectors in the penalty terms are concatenated with a 1. R^i is a diagonal matrix for selecting the first two components of the separating plane defined for crossing i. Note that the objective function $\tilde{f}(x, u)$ is coercive in x for any fixed u, as for any arbitrarily large coordinates x, the stress and penalty terms go to infinity. The constraints restrict the set of feasible u to a compact set. The function $\tilde{f}(x, u)$ is bounded below as the stress terms and each penalty term is bounded below by 0. Let us assume that the distance matrix defined by δ has non-zero elements for all $i \neq j$, therefore the function $\tilde{f}(x, u)$ is differentiable in x and in u. We also know that the stress function and penalty terms are continuous. Therefore, problem (6.14) complies with the template of a nonconvex nonsmooth optimization problem defined in Problem (6.8).

Let us now compare the original constrained MDS problem with its auxiliary problem (6.15)

$$\begin{aligned} \underset{x,u}{\text{minimize}} \hat{f}(x,u,\hat{x}) = \quad & \frac{1}{2}x'L^w x - x'L(\hat{x})\hat{x} + \sum_{i=1}^{m} \rho_i[(u'A^i x + 1)_+ + (u'B^i x + 1)_+] \\ \text{subject to} \quad & ||R^i u||^2 \leq \delta, \; i = 1..m. \end{aligned} \tag{6.15}$$

This auxiliary problem is similar to (6.14). The penalty terms $p_i(x, u), \; i = 1..m$ augmented to the objectives of both problems are identical. All the properties described above hold, with the additional property that problem (6.15) is biconvex. It can be shown, if we fix the coordinates of one point as in [49] and consider the problem of finding the configuration of the remaining $n-1$ points, $L^w$is diagonally dominant and therefore positive definite. The *x-subproblem* of (6.15) is *strictly convex* and has a unique solution, thus making the configuration translation-independent. Therefore, $\hat{\phi}(x,\hat{x}) = \frac{1}{2}x'L^w x - x'L(\hat{x})\hat{x}$ is biconvex with the *x-problem* being strictly convex. Therefore, problem (6.15) complies with the template of a biconvex nonsmooth optimization problem defined in 6.9.

From inequality (6.13), we know the objective function $\hat{f}(x,u,\hat{x})$ of problem (6.15) also majorizes function $\tilde{f}(x,u)$ of problem (6.14), i.e.

$$\hat{f}(x,u,\hat{x}) \geq \tilde{f}(x,u) \tag{6.16}$$

with equality at $\hat{x} = x$.

Therefore, we can apply the alternating algorithm developed in Section 6.3 to solve the original constrained MDS problem (6.14).

For the sake of completion, let us define the First Order Necessary Conditions (FONC) for optimality of (6.15) using the conditions derived in Theorem 1 for problems (6.14) and (6.1).

Definition 9. ***FONC for optimality of original constrained embedding problem (6.14):*** *Assume, x^*, u^* is a minimum of problem (6.14) and a constraint qualification is satisfied, then $\exists\alpha$, such that*

$$\begin{aligned} L^w x^* - L^{x^*}x^* + \sum_{i=1}^{m} \rho_i(\hat{g}_i A^{i\prime}u^* + \hat{h}_i B^{i\prime}u^*) = 0 \\ \sum_{i=1}^{m}[\rho_i(\tilde{g}_i A^{i\prime}x^* + \tilde{h}_i B^{i\prime}x^*) + 2\alpha_i u^*] = 0 \\ ||R^i u^*||^2 \leq \delta \; i = 1..m \\ \alpha_i(||R^i u^*||^2 - \delta) = 0, \; \alpha_i \geq 0, \; i = 1..m. \end{aligned}$$

where

$$\begin{aligned} \hat{g} \in D_x(u^{*'}A^i x^* + 1), \hat{h} \in D_x(u^{*'}B^i x^* + 1) \\ \tilde{g} \in D_u(u^{*'}A^i x^* + 1) \; \textit{and} \; \tilde{h} \in D_u(u^{*'}B^i x^* + 1) \end{aligned}$$

Definition 10. ***FONC for optimality of auxiliary function(6.15):*** *Assume, x^*, u^**

is a minimum of problem (6.15) and a constraint qualification is satisfied, then $\exists\alpha$*, such that*

$$L^w x^* - L(\hat{x})\hat{x} + \sum_{i=1}^{m} \rho_i(\hat{g}_i A^{i\prime} u^* + \hat{h}_i B^{i\prime} u^*) = 0$$

$$\sum_{i=1}^{m} [\rho_i(\tilde{g}_i A^{i\prime} x^* + \tilde{h}_i B^{i\prime} x^*) + 2\alpha_i u^*] = 0$$

$$||R^i u^*||^2 \leq \delta \; i = 1..m$$

$$\alpha_i(||R^i u^*||^2 - \delta) = 0, \; \alpha_i \geq 0, \; i = 1..m.$$

where

$$\hat{g} \in D_x(u^{*'} A^i x^* + 1), \, \hat{h} \in D_x(u^{*'} B^i x^* + 1)$$

$$\tilde{g} \in D_u(u^{*'} A^i x^* + 1) \; \textit{and} \; \tilde{h} \in D_u(u^{*'} B^i x^* + 1)$$

It can be seen from the above two definitions of optimality conditions that if x^*, u^* satisfies FONC of problem (6.14), x^*, u^*, x^* also satisfies FONC of (6.15), the auxiliary problem majorized at x^*. This follows from Lemma 1.

CHAPTER 7
Variations and Extensions

In this chapter we further describe two tools: spoligoforests (and its variations and extensions) and host-pathogen maps, that help further the application of visual analytics for TB epidemiology. They serve as a testbed for the solutions described in Chapter 4 and 5. We will look at variations of the algorithm and concepts discussed in this thesis for edge-crossing minimization to other problems in information visualization in the context of TB epidemiology. . In Section 7.1, we describe closed-form solutions for the *u-subproblem* based on the geometry of the problem. We then discuss applying the alternating algorithm for reducing intersections between pairs of edges to apply to node-node and node-edge intersections in Section 7.2. We apply this method to generate host-pathogen maps with no node-overlaps. We extend the idea to apply to removing overlaps between objects of arbitrary shapes and size.

7.1 Closed-form solutions

In this section we develop the closed form solutions for the *u-subproblem*. The *u-subproblem* is essentially the 1-norm SVM problem for the four end-points of the edges. While we have described methods to find the optimal separating plane in chapter 4 and 5, it may not in fact be necessary to find the *optimal* separating plane. By exploiting the geometry of the problem, we can find closed-form solutions significantly faster.

Assume the x and y co-ordinates of a node n are represented by a vector $n \in \mathbb{R}^2$. Consider edge A from node a to b and edge B from node c to d. Denote these as $\overline{ab}$ and $\overline{cd}$ respectively. Equation of line containing edge A is:

$$x' u_{ab} = \beta_{ab}. \tag{7.1}$$

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Web tools for molecular epidemiology of tuberculosis*, Infect. Genet. Evol., 12 (2012), pp. 767-781.
(2) K. P. BENNETT, C. OZCAGLAR, J. RANGANATHAN, ET AL., *Visualization of tuberculosis patient and Mycobacterium tuberculosis complex genotype data via host-pathogen maps*, IEEE BIBM Workshop on Computational Advances in Molecular Epidemiology, Atlanta, November 2011., (2011).

with $u_{ab} = [b_2 - a_2, a_1 - b_1]'$ and $\beta_{ab} = a_1(b_2 - a_2) + a_2(a_1 - b_1) = a'u$. The distance from c to $\overline{ab}$ is

$$\frac{|c'u_{ab} - \beta_{ab}|}{\|u_{ab}\|}. \tag{7.2}$$

Note that $c'u_{ab} - a'u_{ab} > 0$ then c is on the "greater than" half space defined by $\overline{ab}$ and similarly $c'u_{ab} - a'u_{ab} < 0$ then c is the "less than" half space yielding the following Lemmas.

Lemma 5. *The points c and d are contained in the same half space defined by $\overline{ab}$ if and only if*

$$(c'u_{ab} - a'u_{ab}) * (d'u_{ab} - a'u_{ab}) > 0. \tag{7.3}$$

Theorem 9. *The line segments $\overline{ab}$ and $\overline{cd}$ intersect if and only if*

$$(c'u_{ab} - a'u_{ab}) * (d'u_{ab} - a'u_{ab}) \leq 0. \tag{7.4}$$

and

$$(a'u_{cd} - c'u_{cd}) * (b'u_{cd} - c'u_{cd}) \leq 0. \tag{7.5}$$

Proof. Assume $\overline{ab}$ and $\overline{cd}$ intersect at point p. There exists $0 \leq \beta \leq 1$, such that $p = c + \beta(d - c)$ and $p = d - (1 - \beta)(d - c)$. Therefore

$$(c'u_{ab} - a'u_{ab}) = (p - \beta(d - c))'u_{ab} - a'u_{ab} = -\beta(d - c)'u_{ab}$$

and

$$(d'u_{ab} - a'u_{ab}) = (p + (1 - \beta)(d - c))'u_{ab} - a'u_{ab} = (1 - \beta)(d - c)'u_{ab}$$

Thus

$$(c'u_{ab} - a'u_{ab})(d'u_{ab} - a'u_{ab}) = -\beta(1 - \beta)((d - c)'u_{ab})^2 \leq 0.$$

Swapping the role of $\overline{ab}$ and $\overline{cd}$ yields equation 7.5.

Conversely without loss of generality, assume equation 7.4 does not hold. Then $(c'u_{ab} - a'u_{ab}) * (d'u_{ab} - a'u_{ab}) > 0$. By the Lemma 5, c and d are contained in a half space defined by $\overline{ab}$, thus $\overline{ab}$ and $\overline{cd}$ do not intersect. □

Note this provides a closed form procedure for determining if the edges are separable. One can simply calculate both u_{ab} and u_{cd} and see if equation of any one edge defines a half-space containing both endpoints of the other edge.

7.1.1 No Intersection Case

In this section we discuss possible closed-form solutions for finding the equation of the separating line between non-intersecting edges. Including constraints for enforcing a minimum separation between non-intersecting edges prevents the algorithm from introducing new edge-crossings while adjusting the layout in the $x - stage$. It can be seen that if we simply include penalties for only currently observed edge-crossings in a given layout, we could potentially introduce new crossings as there would be no penalty for its occurrence in the current iteration. However introducing penalty terms for all pairs of edges is computationally expensive, and may in fact be unnecessary as the $x - stage$ makes gentle updates in the layout. Also the more constraints that are in the place the less flexibility in the acceptable co-ordinates of a layout. Therefore, a compromise must be made in allowing enough flexibility in the positions of the nodes that reduce existing crossings while also preventing introduction of new crossings.

One way of avoiding the introduction of new crossings in the $x - stage$ is to ensure minimum separation between edges that are relatively close to each other but are as yet uncrossed. We can formulate this problem enforcing minimum separation between edges $\overline{ab}$ and $\overline{cd}$ as follows:

$$\begin{array}{ll} \min_{u,\beta} & (a'u - \beta + 1)_+ + (b'u - \beta + 1)_+ (-c'u + \beta + 1)_+ (-d'u + \beta + 1)_+ \\ s.t. & \|u\|^2 \leq \delta \end{array} \tag{7.6}$$

Note that the distance between the separating planes $u = \beta + 1$ and $u = \beta - 1$ is $\frac{2}{||u||}$. If the optimal objective value is 0, then the edges are separable and presence of the constraint guarantees the two edges are at least $\frac{2}{\delta}$ apart. However, to solve each of these problems to optimality may be needlessly expensive. We therefore consider ways of finding approximate solutions that can be computed relatively easily.

One possible solution would be to use the equations of lines defined by either edge with an appropriately shifted intercept. For instance, consider the case when the edges are separable, and exactly one of the edges, say edge $\overline{ab}$, defines a half-space in which the other edge $\overline{cd}$ lies. Then we can construct candidate separating planes using the equation of edge ab given by $x'u_{ab} = \beta_{ab}$. The norm of $||u_{ab}||$ can be set to enforce the desired minimum separation η as follows,

$$u_{ab} = \frac{u_{ab}}{||u_{ab}||} * \frac{2}{\eta}$$

Further, we know that if both points c and d lie in exactly one half-space defined by edge ab, then $(c'u_{ab} - a'u_{ab}) > 0(d'u_{ab} - a'u_{ab}) > 0$. Let $m = min(c'u_{ab}, d'u_{ab})$. The intercept can be set to be exactly midway between the edge ab and the end-point of edge cd that is closer to edge ab as follows:

$$\beta_{ab} = \frac{m + a'u_{ab}}{2}$$

If u_{ab} and β_{ab}, found by this procedure result in a non-zero value of the objective in problem (7.6), then in the *x-subproblem*, the algorithm tries to find a layout that enforces the requisite minimum separation between the edges. This can produce an aesthetically pleasing layout. However, choosing an unduly large η in proportion to the scale of the drawing makes the problem increasingly difficult as several constraints may be conflicting.

Other variations are possible based on the geometry of the problem that have differing effects on the behavior of the *x-stage* of the algorithm. For instance, the intercept β_{ab} can be set to be equal to $a' * u_{ab} - 1$. Thus $(a'u - \beta + 1)_+ + (b'u - \beta + 1)_+$ would be exactly 0. Therefore, if there is a violation of the minimum separation requirement, the non-zero penalties would be associated only with edge $\mathcal{B}$, and the onus of shifting to satisfy this constraint in the *x-stage* of the algorithm would be entirely on edge $\mathcal{B}$.

Note also the following technical aside. The separating plane obtained in the *u-subproblem* is used to create the penalties in the *x-subproblem* and therefore, must use the same convention of signs. i.e. The sign of u_{ab} must be adjusted so that the edge that is considered as $\mathcal{A}$ lies in the "greater than" half space defined by $\overline{ab}$ and edge $\mathcal{B}$ must lie in the "less than" half space.

If both $\overline{ab}$ and $\overline{cd}$, are acceptable, i.e. each edge define a half-space in which the other edge lies entirely, then equation of either edge may be used as separating line. In this case, the line that allows for the largest $||u||$ while keeping the objective value at 0, would be preferred. The larger the value of $||u||$, the greater flexibility is allowed in the positioning of coordinates. If the projection of one of the end-points of edge A lies on edge B, the separating line can be constructed midway between the endpoint and its projection. If the projection of none of the endpoints of the edges lies on the opposite edge, one possibility is to use the perpendicular bisector of the line connecting the two nearest points of edge $\mathcal{A}$ and edge $\mathcal{B}$. Some closed form solutions for finding separating lines discussed in this section are illustrated in Figure 7.1.

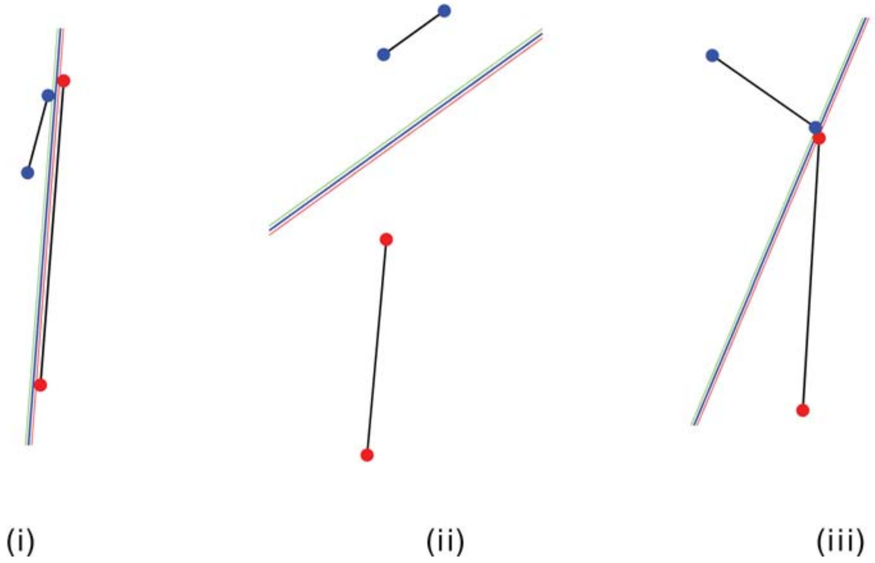


Figure 7.1: Various possible closed form solutions for finding separating line between edges when they are separable. Edge $\mathcal{A}$ is denoted by red nodes, and edge $\mathcal{B}$ by blue nodes. The separating line $xu = \beta$ is given by the blue line, while corresponding line $x'u = \beta + 1$ is in red, and $x'u = \beta - 1$ in green. (i) Both equations of edge $\mathcal{A}$ and $\mathcal{B}$ can be used, separating line based on equation of edge $\mathcal{A}$, (ii) Only a line parallel to $\mathcal{B}$ may be used. (iii) Separating line midway between two nearest points. Minimum separation requirement not satisfied even though there is no crossing, will result in non-zero penalty.

7.1.2 Intersection Case

In the separable case too, there are several possibilities for closed form solutions. Let us consider equations of separating lines derived from the equations of one of the edges. Consider edge A given by $x'u_{ab} = \beta_{ab}$. If in addition to removing the crossing we want a minimum separation of exactly η, we can adjust $||u||$ as follows:

$$u = \frac{u_{ab}}{||u_{ab}||} * \frac{2}{\eta}$$

The intercept can be set in several ways creating various consequences in the behaviour of the *x-stage*. One possibility is to set it such that $\mathcal{A}$ satisfies the constraint exactly. This can be done by setting $\beta = \beta_{ab} * \frac{2}{\eta * ||u_{ab}||} - 1$. This would imply a single node of the opposite edge has a relatively large non-zero penalty. In the *x-stage*, this node would be required to move the most, while other nodes may remain stationary in order to satisfy the no-crossing constraint corresponding to this edge-pair. Another possibility is to set the intercept midway between β_{ab} and $c * u_{ab}$ or $(d * u_{ab})$. This may result in all three nodes a, b, c violating the constraint, but the violation caused by each node is relatively small. Therefore, three nodes would be required to adjust their coordinates, but the shift in each node required is relatively small. Overall, the former choice resulted in slightly better performance and is what is included in the implementation. The two possibilities are illustrated in Figure 7.2.

Further, we can use heuristics to decide which point c or d of edge $\mathcal{B}$ is on the "wrong" side of edge $\mathcal{A}$. For instance, the point whose distance from edge $\mathcal{A}$ is smaller can be chosen as the point on the "wrong" side. Consider the case when c is closer to edge $\mathcal{A}$, than d is. If $(-c'u + \beta + 1) < 0$, i.e. c is on the correct side, then setting $u = -u$, would obtain the desired effect. This choice results in the smallest possible adjustment in coordinates to satisfy the crossing constraint. Alternatively, this choice can be made based on the degree of nodes c and d. A leaf node has greater flexibility in placement as compared to a non-leaf node, as a change in the position of a non-leaf node could also alter the coordinates of all nodes connected to it. Therefore, setting a leaf node to be on the "wrong" side, even if it causes a larger penalty, and forcing it to move might be beneficial as it does not cause other edges connected to non-leaf nodes to move.

7.2 Intersections between Arbitrary Shapes in Graph Drawings

We also explore various genetic distances that serve as proximity measures for spoligotypes.

A strategy similar to that for minimizing edge-crossings as proposed earlier can be used to minimize overlaps between various components in a graph drawing e.g. node-node overlap, node-edge overlap, label overlaps and intersections between subgraphs. We describe the optimization challenges posed by these visualization tasks involving minimizing overlap while preserving proximity relations. Given that each object in the graph is a convex polyhedron (considering convex hulls of arbitrary shapes), we can define the condition for no overlap between these objects. Allowing $\mathcal{A}$ and $\mathcal{B}$ to be of an arbitrary number of extreme points, recall corollary (4.4) that can be easily proven to apply to intersections between any convex polyhedrons expressed as a convex combination of their extreme points. Two polyhedrons intersect only if the condition specified in (4.5) is sat-

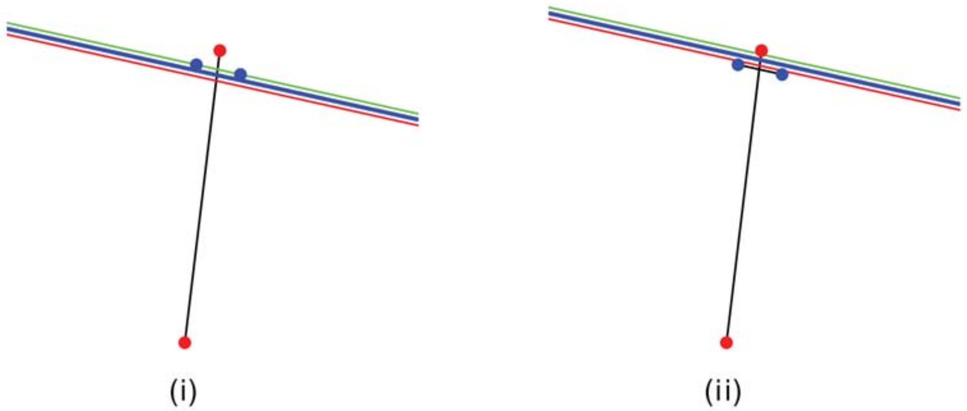


Figure 7.2: Various possible closed form solutions for finding separating line between edges when they are separable. Edge $\mathcal{A}$ is denoted by red nodes, and edge $\mathcal{B}$ by blue nodes. The separating line $xu = \beta$ is given by the blue line, while corresponding line $x'u = \beta + 1$ is in red, and $x'u = \beta - 1$ in green. (i) Intercept set such that $\mathcal{B}$ satisfies constraint exactly, non-zero penalty for only single node of edge $\mathcal{A}$ (ii) Intercept set such that 3 nodes have non-zero penalties, although penalty of each node is relatively small. Requires adjustment of three points to satisfy constraint.

isfied. If the edges do cross, then the optimal objective of (4.5) will be strictly greater than 0. The solution then defines the half-spaces in which the two graph objects should lie, such that the error is minimized. This is analogous to soft-margin SVM classifiers for non-separable data where the goal is to find the separating plane that best separates the data while allowing for the possibility of misclassified points. A cost is incurred for every misclassified point, therefore the solution tries to strike a balance between finding the maximum margin separating plane and minimizing training error. From the visualization perspective, the soft-margin has a desirable aesthetic consequence. The least adjustment is required in the location of the objects so that there is no overlap. This implies that no overlaps between objects can be obtained with only small changes in stress. Fig. 7.3 represents this concept of finding the separating plane for a pair of nodes.

There can be several variations that have interesting aesthetic consequences. Observe that the node object $\mathcal{A}$ in Fig. 7.3 is a nonconvex set of points, and the convex envelope of the node is used in the illustration to represent the concept of linear separability of the convex polyhedrons $\mathcal{A}$ and $\mathcal{B}$. Another possibility is to split the nonconvex

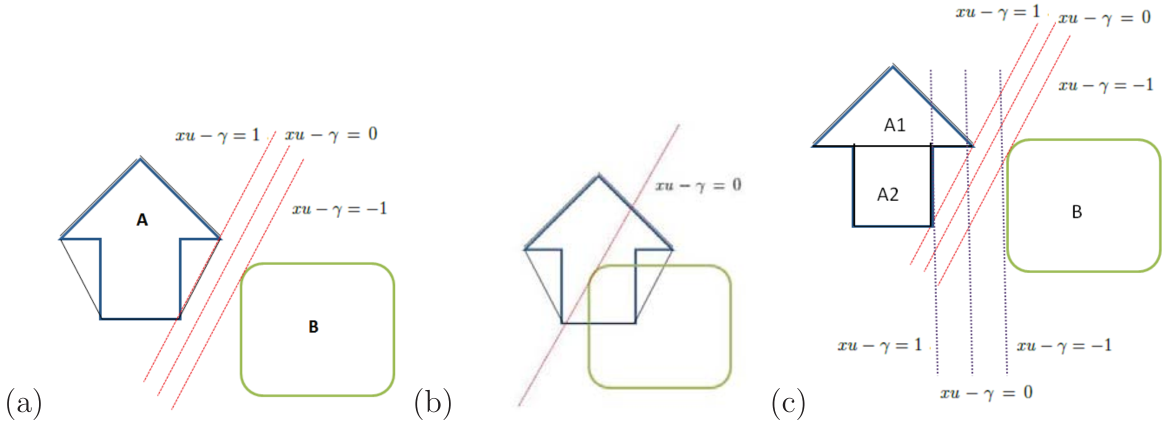


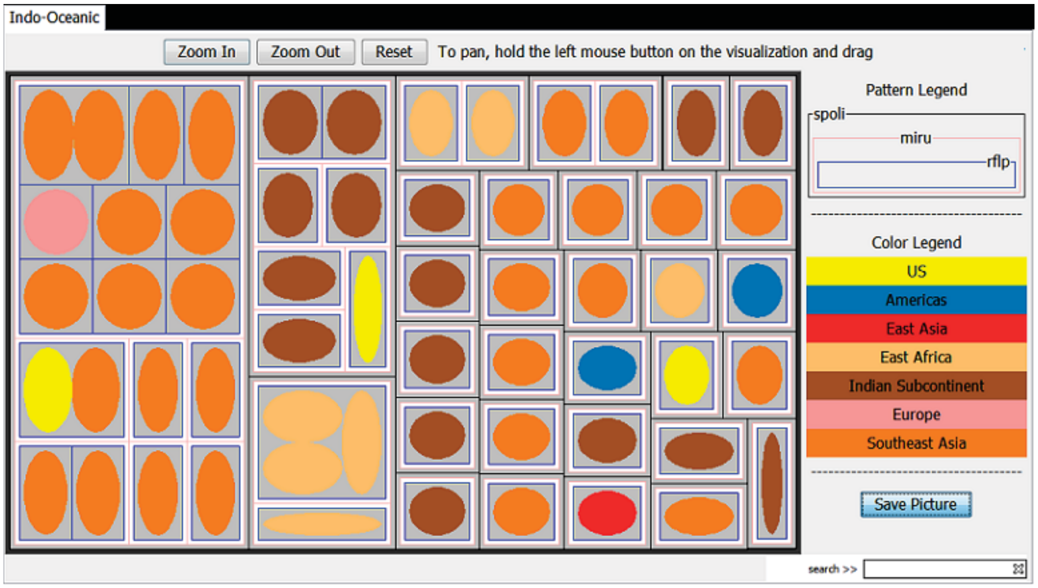
Figure 7.3: Condition for no-overlap: (a) Objects $\mathcal{A}$ and $\mathcal{B}$ do not overlap. Any line between $xu - \gamma = 1$ and $xu - \gamma = -1$ strictly separates the edges. Using a soft margin, the plane in (b) $xu - \gamma = 0$ separates the plane into half spaces that should contain each object. (c) Nodes that are nonconvex sets can be decomposed into convex parts if prior knowledge about the shape of the node is known.

object into its convex parts, if we have knowledge of the special structure of the shape of the node. This is represented in Fig. 7.3 c. Decomposing a node into simple-shaped convex sets could imply lesser adjustment in the placement of objects to satisfy the no-overlapping requirement. However, it comes at the additional cost of computing more separating planes for more pairs of objects and having more penalty terms.

In the following three sections, we show various applications that illustrate the concept of removing overlaps between arbitrary components of a graph.

7.2.1 Minimizing Overlaps in Host-pathogen Maps

These concepts of minimizing node-node, node-label overlap can be tested in a practical application: generating spoligoforests and host-pathogen maps for tuberculosis. Spoligoforests are described in greater detail in Chapter 1 and host-pathogen maps described in detail in [11] are introduced in this section. Host-pathogen map visualizations provide a graphical representation of strain and patient associations. Patients are represented as nodes within the nested boxes depicting strains. The visualization as shown in Fig. 7.4 depicts each strain by telescopic boxes depending on the number of biomarkers uploaded. In Fig. 7.4, the nested boxes represent the spoligotype, MIRU type and RFLP pattern, respectively. Other biomarkers such as SNPs may also be used. Patient char-



To pan, hold the left mouse button on the visualization and drag

Figure 7.4: Host-pathogen maps of patients from New York State infected with strains of the Indo-Oceanic lineage that visualize associations between the genotype and host characteristics. Strains are represented by triples of spoligotype, MIRU and RFLP patterns and are depicted by nested boxes. Patients are depicted as nodes colored by region of birth. The visualization shows the predominance of strains of the Indo-Oceanic lineage in patients from South-East Asia and the Indian subcontinent. Clusters of cases with identical associated genotype appear in bigger boxes, thus bringing attention to possible outbreaks.

acteristics such as birth-place are represented by color coding the nodes by continent of birth. This visualization provides a means of tracking trends in transmissions between patients infected with the strains of interest. It can help reveal previously unrecognized epidemiological links between patients. Anomalous behavior of strain groups can also be identified. Epidemiological investigations require the investment of significant time and resources. Therefore, identifying suspicious clusters using such visual tools will help towards the efficient allocation of efforts for case investigations.

Previously, tree-maps were used to visualize host-pathogen information [11]. Tree-maps are a popular and effective approach for visualizing hierarchically structured information. However, the design allows for very little flexibility in the placement of boxes. An important goal of any effective information visualization is to create visual appeal while minimizing cognitive load. Tree maps can capture both structural information as well as

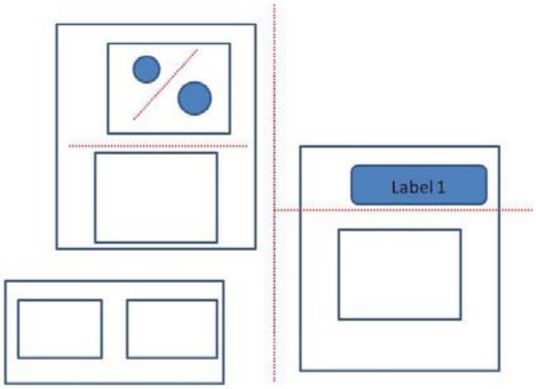


Figure 7.5: Illustration of the concept of eliminating overlaps between nested boxes, and boxes and labels in host-pathogen maps.

details of content of individual nodes (provided the graph is not too dense). The amount of information to be displayed, and hence space requirements, for traditional representations of hierarchical information e.g. listings and outlines increases linearly with the number of nodes [91]. While traditional tree diagrams represent structure quite effectively, greater than 50% of space goes into the background. The construction of tree maps implicitly captures structure, and provides ample space to represent content information. They are therefore very good tools to depict the overall view of trends in host-pathogen associations.

However, the main drawback with generating such a layout is there is not enough flexibility in the placement of nodes. It is desirable to have a visualization where the hierarchical structure is maintained, but boxes be placed such that proximity relations are preserved. For instance, distances between boxes could reflect genetic distances between strains. Distances between nodes within boxes can be used to reflect ”epidemiological distances” based on shared risk factors or epi-links. Temporal information e.g. time of infection in order of occurrence could also be represented as distances between nodes placed along a single axis, with the earliest infection shown as the leftmost point on the axis. Additional bounds constraints would have to be incorporated to ensure the nodes are placed within the boxes. With these additional desiderata, the requirement of having absolutely no background space would have to be relaxed. Moreover, since boxes have arbitrary positions, overlaps between boxes and between boxes and labels will have to be handled. This is illustrated in Fig. 7.5. Such depictions that capture the distances between nodes and boxes can be especially useful when providing a detailed view of smaller sets of patients of interest. The distances can be used to capture multiple pieces of information e.g. genetic distances, epi-distances and time. These visualizations can complement the big-picture overview provided by treemap based host-pathogen maps.

In Figure 7.6, we represent a host-pathogen map based on some of these ideas. Each colored circle represents an infected individual, with the color representing the ethnicity of the person. Each box represents a MIRU type. Patients with identical spoligotype and MIRU type are placed in the same box, just as in Host-Pathogen Treemaps described in Chapter 2. The size of the box is therefore determined by the number of patients, therefore the boxes are not uniformly sized. Distances between boxes represent the MIRU Hamming distances. MIRU types that differ in exactly one locus, known as *one-offs* are connected by an edge. One-offs are of special interest to epidemiologists. They could represent cases in which the MIRU type has evolved. While these occurrences are not frequent, the number of repeats at a locus have been observed to change over a period of $2-3$ years, resulting in different genotyping results for the same patient over a span of some years. Or they could simply be instances when the number of repeats at a given locus were misread. Therefore the cluster may be larger than perceived and there may be a greater number of active transmissions than believed. Therefore, there is a need for these cases to be highlighted in a visualization.

The host pathogen map in Figure 7.7 shows all patients in New York City infected with strains that have spoligotype 1111111111110110000000000000000000000000000 of the Euro-American lineage. The large box at the top left (associated with MIRU type 223327133228) and filled with yellow circles is one example of a case that is interesting to epidemiologists. Firstly, because the patients with these strains are predominantly US-born. It is largely believed by health care workers that a majority of US-born individuals represent cases of active transmission within the U.S.. While foreign-born patients possibly acquired infections outside the country but have only become actively infected within the U.S [20]. Therefore, the former cases are of greater interest in terms of stopping an active chain of transmission. Furthermore, the one-offs (223327163228 and 223327173228) associated with this MIRU type are also of interest as they may represent cases of mutation in MIRU type or technical error and in fact could be cases of active transmission of MIRU type 223327133228.

Figure 7.7 represents a host-pathogen map of the virulent Beijing strain with spoligotype 0000000000000000000000000000000000111111111 from the New York City dataset of patients. While this strain is largely associated with people of East Asian descent as

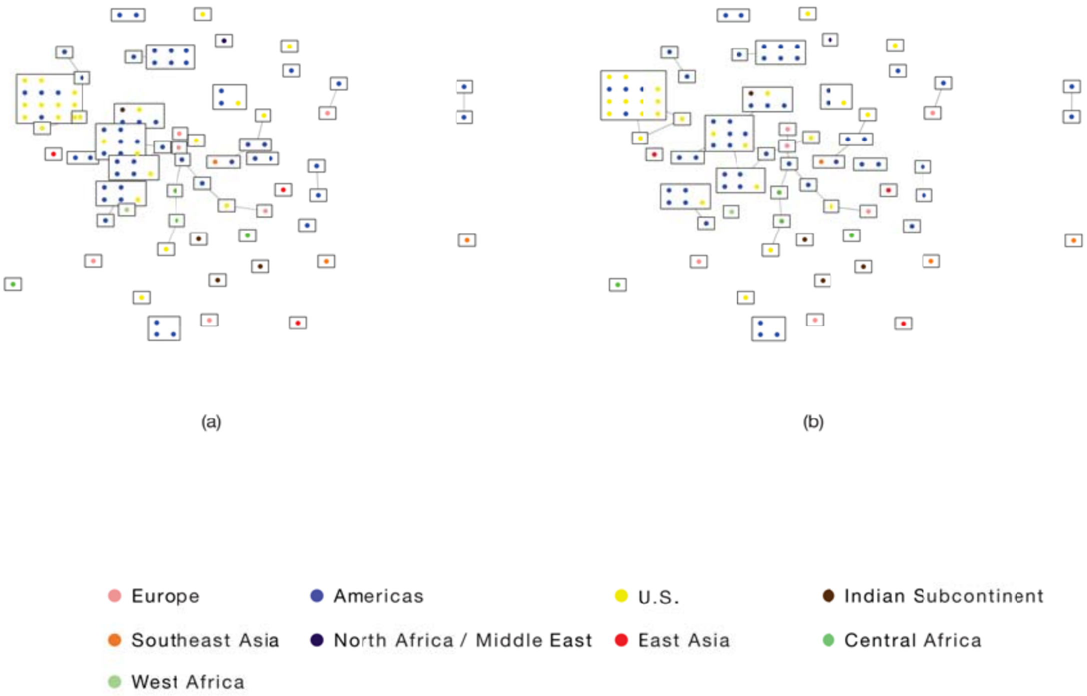


Figure 7.6: Host-pathogen map for Euro-American strains (a) before and (b) after node overlap removal.

can be seen by the predominantly red circles in the image, there are cases of US-born patients as well, possibly representing cases of active transmission.

Thus host-path maps show the potential of removing overlaps between arbitrarily sized nodes using the alternating strategy. There are existing methods such as in [34, 50], that also perform node overlap removal by scanning the graph from left to right and top to bottom and imposing constraints based on the minimum and maximum x and y coordinates of each node. A formal comparison between these methods is left as future work. However these methods may not be suited for more arbitrarily shaped objects in a graph drawing. In the following two subsections, we illustrate two such applications where overlaps between arbitrarily shaped objects in a graph drawing are eliminated.

7.2.2 Node-Edge Overlaps

Constraints to avoid node-edge overlaps may be required in cases, when we have nodes of arbitrary shape and size. When nodes are not simply represented as points and have a significant width and height relative to edges, this can lead to overlaps between edges and nodes. If constraints that eliminate node-edge intersection are imposed as

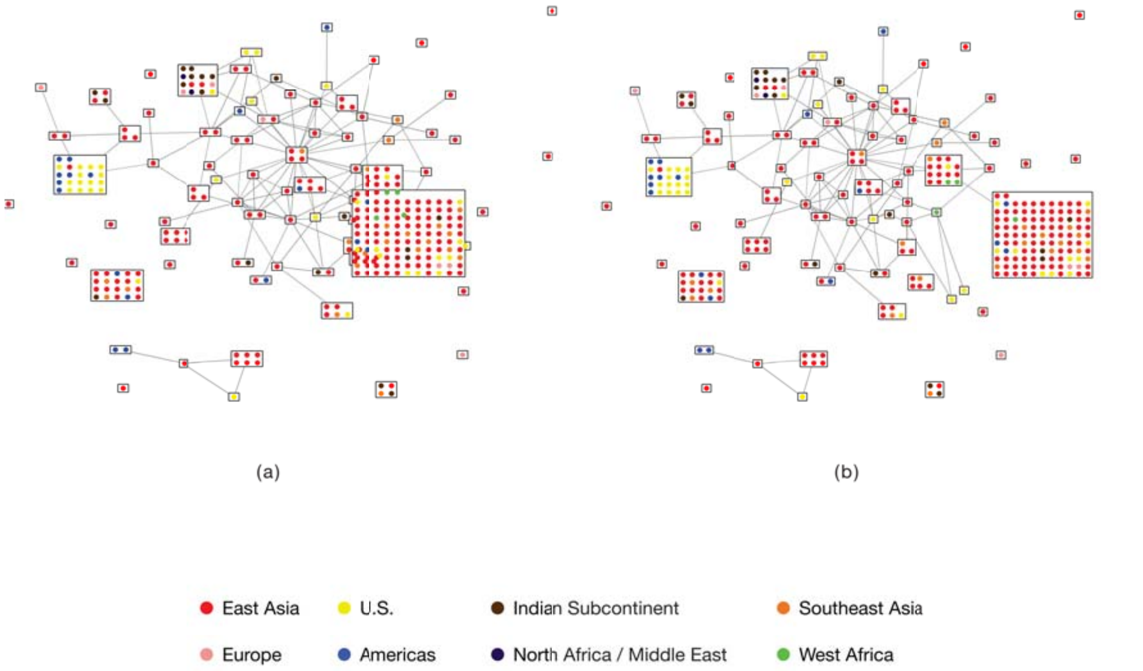


Figure 7.7: Host-pathogen map for East-Asian strains (a) before and (b) after node overlap removal.

penalties in addition to intersections between edges, we can get an aesthetically more appealing layout with marginal increase in stress. A possible application of minimizing node-edge intersections arises in generating spoligoforests where nodes are of arbitrary shapes and sizes. For example, when generating interactive spoligoforests, where users may zoom into nodes to view an embedded host-pathogen map showing all patients infected with the corresponding strains. This idea is represented in the sample spoligoforest in Fig. 7.8.

Note also that as a side-effect of imposing edge-crossing constraints alone the algorithm can reduce the length of "offending" edges that cause one or more crossings. We show how a minimum separation can be enforced between nodes and edges allowing the use of larger nodes. To generate the image in Figure 7.9 (b) two types of constraints were imposed as penalties, the edge-crossing constraints as described in Chapter 4, as well as similar constraints preventing overlap between pairs of a node A and an edge B.

$$Au \geq \gamma + 1$$

$$Bu \leq e'\gamma - 1$$

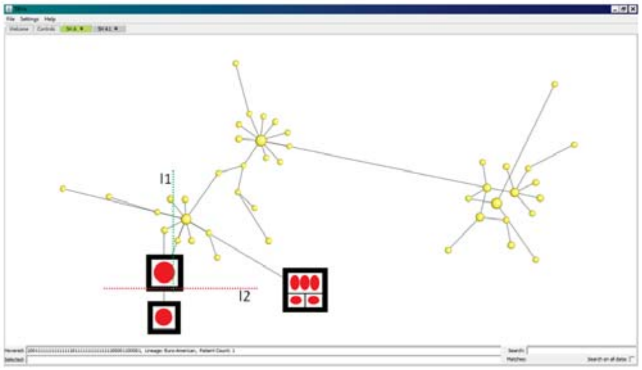


Figure 7.8: Illustration of the concept of eliminating node-node and node-edge intersections resulting from nodes of arbitrary shapes and sizes in a spoligoforest. l1 is the separating plane that defines the halfspaces in which the node and edge must lie, while l2 defines the halfspaces that should contain the two nodes.

where $A = \begin{pmatrix} A_x & A_y \end{pmatrix}$ with A_x and A_y represents x and y co-ordinates of the node, and $B = \begin{pmatrix} S_x & S_y \\ T_x & T_y \end{pmatrix}$, where B is an edge between nodes S and T.

7.2.3 Convex Envelope around Subgraphs

Another possible application of this methodology is in modifying the placement of individual subgraphs in existing layouts to respect some proximity-preserving criteria, by applying penalties for overlapping subgraphs. In Figure 7.10, we modify existing layout as computed by Graphviz Twopi to respect lineage-wise distances. We define inter-lineage distance as the average genetic distance between strains belonging to each lineage. While we do not modify the layout of each individual component, we change the overall layout by altering the relative positions of the components to reflect the inter-lineage distance. Thus in Figure 7.10, a spoligoforest of EuroAm-African and *M. africanum* lineages, the two components corresponding to the T-Uganda sublineage are placed adjacent to each other, and all the EuroAm-African sublineages are placed closer together than to the *M. africanum* sublineages. This system can be adapted for use in "packing" algorithms as in [51] that strives to find the most compact arrangement of a set of disconnected components. Another possible application of this concept is in parallelizing the algorithm to suitably partition the graph, find crossings minimized layouts for each of the disconnected components individually and merge them by disallowing overlaps between any components. We

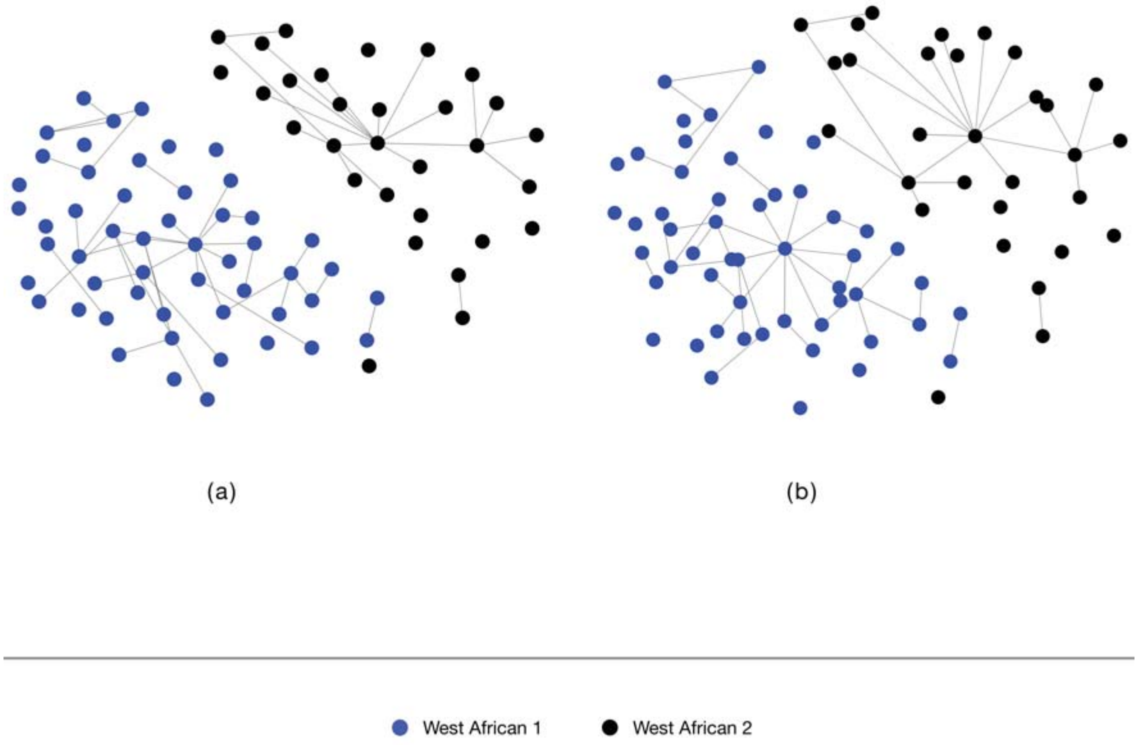


Figure 7.9: Enforcing minimum separation between nodes and edges results in a clear representation of both nodes and edges, allowing larger nodes to be used, with marginal increase in stress. (a) Initial layout, stress=0.103 and (b) Final layout, stress=0.112.

leave this promising direction as future work.

7.3 Tool for Spoligoforest Generation

In this section, we describe a standalone tool for spoligoforest generation. This software is a Java implementation of the MAA+ algorithm, various genetic distance computation and spoligoforest generating algorithm described in [89]. Options to use other graph layouts generated by MDS using stress majorization and SNE [62] are also provided to the user. The interface is easy to use and allows users to upload a comma separated text file of genotypes(Spoligotypes and optionally MIRU types) along with putative labels to generate spoligoforest images. The adjacency matrix for the user-uploaded dataset is generated using the algorithm described in [89]. Several options are provided to generate

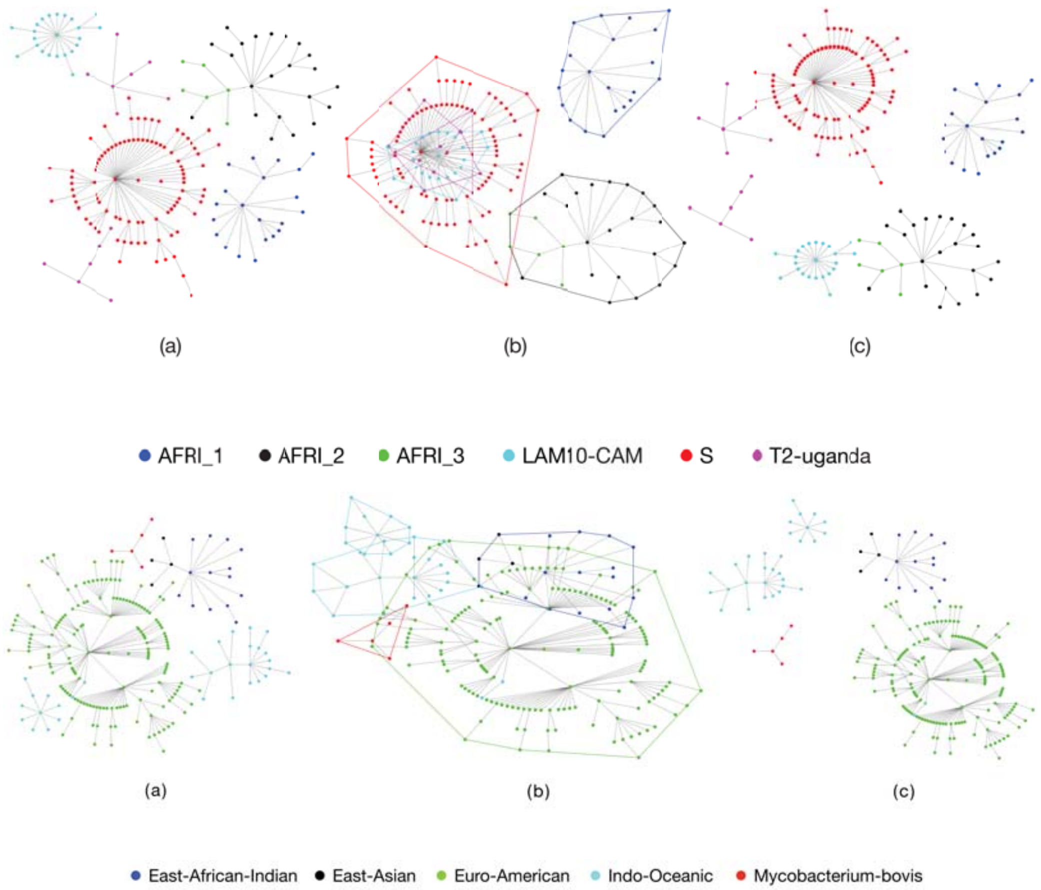

(a)
(b)
(c)
● AFRI_1 ● AFRI_2 ● AFRI_3 ● LAM10-CAM ● S ● T2-uganda
(a)
(b)
(c)
● East-African-Indian ● East-Asian ● Euro-American ● Indo-Oceanic ● Mycobacterium-bovis

Figure 7.10: Spoligoforest layouts computed using Graphviz Twopi modified so distance between components reflects inter-lineage distance. (a) Original layout by Twopi (b) Convex hull of each component computed, MDS using inter-lineage distances results in overlapping components. (c) Overlaps between convex shapes removed while keeping stress low.

the distance matrix. Users may choose between one or more combinations of similarity measures for spoligotypes and MIRU types. Spoligotype distances may be combined with Hamming, l1-norm or l2-norm distances between MIRU types. While the user may choose to weight each distance measure differently, the recommended choice is equally weighted Common (Spoligotype) Deletions and MIRU Hamming distances. The MAA+ algorithm is implemented to create an embedding that preserves the pairwise distances as per the generated distances matrix while keeping the number of crossings low. Separating planes between each pair of edges are found using closed-form solutions described in Section 7.1. An ADMM-based minimization routine is used for the X-stage as described in 5.2.2. Users may generate multiple spoligoforest images for the same input data file using the various combinations of distance measures and embedding algorithms (MAA+, MDS, SNE). Each spoligoforest is generated in a separate tab. The user may export one or all of the images in Portable Network Graphics (PNG) format, individually or combined in a zip file, for later use. Some spoligoforest images generated for equally weighted Common (Spoligotype) Deletions and MIRU Hamming distance matrices using MDS and MAA+ algorithms are represented in figure 7.11. Further this tool is integrated with host-pathogen maps described in Chapter 2. Clicking on a node in the spoligoforest provides the user with a context menu to generate a host-pathogen map for all MIRU types associated with the spoligotype corresponding to the node. This interacton is represented in figure 7.8. The software will be shortly made available on the TB-Insight website as part of the host of tools to aid studies in molecular epidemiology of TB. This software provides the TB epidemiology community with a useful tool to visualize and analyze the genetic diversity in an MTBC population of interest.

Also note that since spoligoforests are used to visualize genetic diversity of an MTBC population, it is necessary for the proximity measures to represent evolutionary distances between strains. In this section, we describe proximity measures offered by the tool, that are based on known mutation mechanisms of the DR region. Distance matrices derived from these similarity measures are used to draw proximity preserving visualizations of the spoligotype strain dataset. These similarity measures were used in [14] for clustering MTBC strains into lineages. These measures take into account the domain knowledge about evolution of the DR region – one or more adjacent spacers may be lost in a single mutation event [108]. Therefore, contiguous deletions or the absence thereof is a good measure of the evolutionary distance between spoligotypes, rather than looking at differences in individual spacers as with traditional Hamming distances. In the tool we provide implementations of the distance metrics defined in [14].

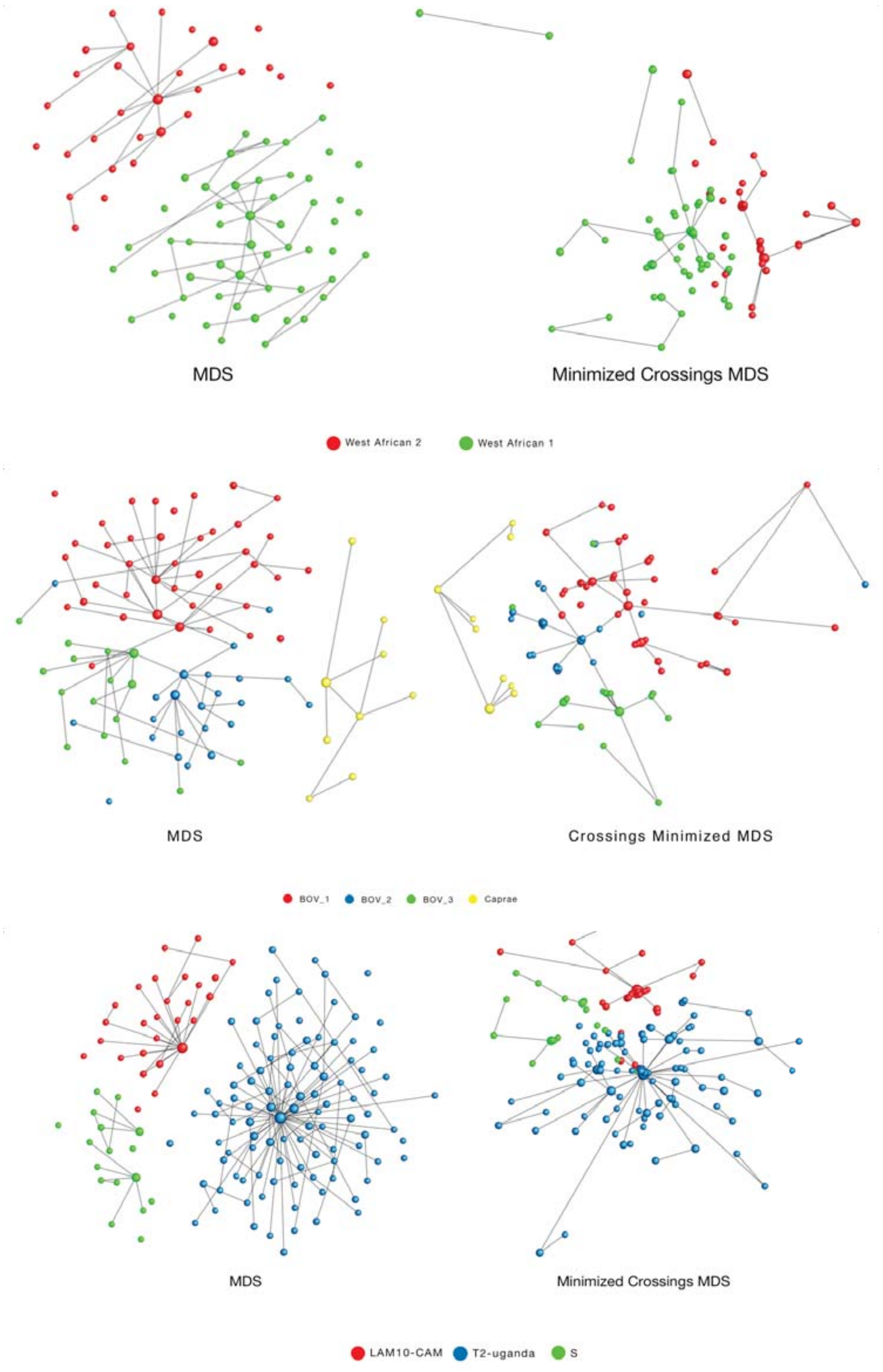


Figure 7.11: Spoligoforest generated by MAA+ using the TB-Vis spoligoforest tool for *M. africanum*, *M. bovis* and EuroAm-African lineages.

CHAPTER 8
Conclusions

In this chapter, we summarize the main contributions of this work :

- Formulation of molecular epidemiology data visualization challenges as optimization problems that represent proximity relations between objects in a graph layout while minimizing overlap.
- An alternating algorithm to solve these nonconvex nonsmooth optimization problems. Implementations of the alternating algorithm MAA and MAA+.
- We provide an analysis of this algorithm, describe optimality conditions of the problem and convergence properties of the algorithm. We show the alternating algorithm guarantees an improvement in the objective relative to the starting configuration of a graph.
- Tools for TB molecular epidemiology:
  - Effective information visualizations: spoligoforests and host-pathogen maps using MAA and MAA+.
  - Development of TB-Lineage: a rule-based lineage classification tool.

We discuss these with reference to the growing role of DNA fingerprint data in building analytics tools for effective tracking and control of TB. This work has created many opportunities and future directions for representing and solving visual analytics tasks as optimization problems. We discuss these briefly.

The application of molecular methods for the epidemiology of TB complement traditional approaches used in public health, and have helped to better understand TB dynamics and the characteristics of its causative agent, *Mycobacterium tuberculosis* complex

Portions of this chapter previously appeared as: (1) A. SHABBEER, C. OZCAGLAR, AND K. P. BENNETT, *Crossing minimization within graph embeddings*, arXiv preprint arXiv:1210., (2012).
(2) A. SHABBEER, C. OZCAGLAR, B. YENER, ET AL., *Preserving proximity relations and minimizing edge-crossings in high dimensional graph visualizations*, IEEE Symposium on Large Data Analysis and Visualization (LDAV), Providence, RI, 2011, pp. 131-132.
(3) A. SHABBEER AND K. P BENNETT, *Proximity preservation and crossing-minimization for graph embedding*, in The Snowbird Learning Workshop, 2012.

(MTBC). DNA fingerprinting methods are now routinely employed in TB surveillance programs and have been used primarily to detect recent transmissions and to investigate outbreaks. DNA fingerprinting methods such as spoligotyping, Mycobacterial Interspersed Repetitive Units - Variable Number Tandem Repeats (MIRU-VNTR) typing and IS6110 Restriction Fragment Length Polymorphism typing have provided insights into the genetic diversity of the population structure of the MTBC [74]. Primarily, these typing methods aid traditional epidemiological approaches to detect unsuspected transmission links. These tools address the shortcomings of standard contact tracing methods in identifying transmission events. Since epidemiologically linked patients have identical fingerprints, the fingerprint can serve as a basic tool to distinguish between reactivation of latent infections and recent transmissions and in identifying chains of transmissions[23]. In the long term, however, DNA fingerprint data have been useful in population-based studies and helped develop a deeper understanding of the disease dynamics. In addition, there is great potential in further insights that can be created using routinely collected genotype information.

In addition to tracking individual strains, classification of strains into genetic groups can help provide insights into the characteristics of the strain such as host association, drug resistance, immunity to vaccines, virulence, mutation and transmission rates, latency periods, and pathogen fitness [26, 48, 29, 80, 63]. Groups of emerging strains can also be revealed by an analysis of trends by lineage. Recent studies have established a correlation between host ethnicity and pathogen strains [46, 22, 63, 47]. In this thesis, we first presented a survey of existing classification and visualization tools used for TB molecular epidemiology. We describe our own publicly available classification tool TB-Lineage http://tbinsight.cs.rpi.edu/run_tb_lineage.html that classifies strains of the MTBC into major lineages based on DNA fingerprint data. This tool was developed by synthesizing existing literature and knowledge from experts in the CDC.

Our attempts to build classification tools and analyze the performance of new and existing classification tools exposed new challenges and motivated the need for visual analytics tools for TB. Visualization of public health data is emerging as a popular aid to traditional methods of epidemiology. Modeling and visualizing genetic relatedness and patterns of mutation over relatively short periods of time as provided by spoligoforests are crucial for epidemiological studies. Tools that help in identifying previously unrecognized epi-links and associations between patient and strain groups, such as host-pathogen maps, help analyze recent transmission trends and focus public health efforts in an effective manner.

Generating these visualizations poses a challenging optimization problem. Creating the visualization requires preserving proximity relations while also minimizing intersections between nodes and edges. While existing methods can successfully accomplish each of these embedding objectives individually, combining these (sometimes contradictory) goals poses interesting challenges. Crossing minimization techniques leave little flexibility in the placement of nodes. It is therefore difficult to combine these existing methods with proximity preserving embedding techniques. In this work, we developed an alternate formulation to represent crossings. We showed that the condition for non-intersection of graph "objects" (subgraphs, nodes, edges, labels) can be expressed as the feasibility of a system of nonlinear inequalities. These constraints can then easily be included as penalties in the original proximity preserving embedding objective. Thus, we propose a novel continuous optimization based approach to generating graph embeddings that both preserves proximity relations and minimizes intersection of nodes and edges.

We propose an alternating approach to handle the nonconvexity of this resulting objective. For a fixed layout we compute the separating planes that define the conditions that must be satisfied so that there are no intersections between nodes and edges. We then optimize the layout with respect to proximity preserving criteria and the conditions identified in the previous stage. Both problems (i) finding the separating planes and (ii) minimizing the penalized proximity preserving objective are nonsmooth. We propose an elegant optimization framework: Alternating Directions of Multiplier Methods (ADMM) to handle the nonsmoothness. ADMM is an iterative method that involves solving simple problems which have known closed-form solutions at every iteration improving computational costs.

We describe the optimality conditions for the problem and describe the convergence properties of the algorithm. We also show that the sequence of function values generated by the algorithm is monotonically decreasing and converges. Thus we propose using nonconvex nonsmooth optimization techniques for the development of two visual analytics tools for the molecular epidemiology of TB: spoligoforests and host pathogen maps.

This work opens up many avenues for future research at the intersection of machine learning and data visualization. Here we focused on elimination of edge crossings and stress minimization (MDS). The theorems and algorithms are directly applicable to the intersection of any graph components that are convex polyhedrons. As illustrated in host-pathogen maps, the method can also be used to eliminate node-node overlaps and node-edge crossings. Constraints can be developed to apply to various components of a graph drawing such as between subgraphs, labels, or annotations. The general multi-objective approach for minimizing overlaps between graph components is applicable to any optimization-based dimensionality reduction, graph drawing or embedding methods [103, 33] used for data visualization. While the method described was motivated by the need to minimize edge crossings and simultaneously preserve pairwise distances in heterogeneous graph data as defined by the MDS objective, it can be used to eliminate edge crossings with any embedding objective. Our work was limited to spoligoforests to solve a relevant problem in TB molecular epidemiology. The penalty approach however can be used to reduce crossings in nonplanar graphs as well. Another direction to be pursued is parallelizing the algorithm by a suitable partitioning of the graph, removal of overlaps in resulting components, and subsequently merging the modified layouts of components. We leave these promising research directions as future work.

REFERENCES

[1] C. Allix-Beguec, D. Harmsen, T. Weniger, et al., *Evaluation and strategy for use of MIRU-VNTRplus, a multifunctional database for online analysis of genotyping data and phylogenetic identification of Mycobacterium tuberculosis complex isolates*, J. Clin. Microbiol., 46 (2008), pp. 2692–2699.

[2] M. Aminian, A. Shabbeer, and K.P. Bennett, *Determination of major lineages of Mycobacterium tuberculosis complex using Mycobacterial Interspersed Repetitive Units*, in 2009 IEEE International Conference on Bioinformatics and Biomedicine, Washington D.C., 2009, IEEE, pp. 338–343.

[3] M. Aminian, A. Shabbeer, and K. Bennett, *A conformal bayesian network for classification of Mycobacterium tuberculosis complex lineages*, BMC Bioinforma., 11 (2010), p. S4.

[4] M. Aminian, A. Shabbeer, K. Hadley, et al., *Knowledge-based bayesian network for the classification of Mycobacterium tuberculosis complex sublineages*, in 2011 ACM Conference on Bioinformatics, Computational Biology and Biomedicine, Orlando, FL, 2011, ACM-BCB.

[5] L. Baker, T. Brown, M. C. Maiden, et al., *Silent nucleotide polymorphisms and a phylogeny for Mycobacterium tuberculosis*, Emerg. Infect. Dis., 10 (2004), pp. 1568–77.

[6] P.F. Barnes and M.D. Cave, *Molecular epidemiology of tuberculosis*, New Engl. J. Med., 349 (2003), pp. 1149–1156.

[7] C. Batini, M. Talamo, and R. Tamassia, *Computer aided layout of entity relationship diagrams*, J. Syst. Softw., 4 (1984), pp. 163–173.

[8] G.D. Battista, P. Eades, R. Tamassia, et al., *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall PTR, Englewood Cliffs, NJ, 1998.

[9] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, Wiley-Interscience, New York, 2006.

[10] M. Belkin and P. Niyogi, *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural Comput., 15 (2003), pp. 1373–1396.

[11] K. P. Bennett, C. Ozcaglar, J. Ranganathan, et al., *Visualization of tuberculosis patient and Mycobacterium tuberculosis complex genotype data via host-pathogen maps*, in 2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW), Atlanta, GA, 2011.

[12] Dimitri P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*, Athena Scientific, Nashua, NH, 1 ed., 1996.

[13] I. Borg and P.J.F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, Springer Verlag, Berlin, Germany, 2005.

[14] C. Borile, M. Labarre, S. Franz, et al., *Using affinity propagation for identifying subspecies among clonal organisms: Lessons from M. tuberculosis*, BMC Bioinforma., 12 (2011), p. 224.

[15] S. Boyd, N. Parikh, E. Chu, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations Trends Mach. Learn., 3 (2011), pp. 1–122.

[16] J.M. Boyer and W.J. Myrvold, *On the cutting edge: Simplified O(n) planarity by edge addition*, J. Graph Algorithms Appl., 8 (2004), pp. 241–273.

[17] U. Brandes and T. Willhalm, *Visualization of bibliographic networks with a reshaped landscape metaphor*, in Proceedings of the Symposium on Data Visualisation, Aire-la-Ville, Switzerland, 2002, Eurographics Association, p. 159.

[18] R. Brosch, S. V. Gordon, M. Marmiesse, et al., *A new evolutionary scenario for the Mycobacterium tuberculosis complex*, Proc. Natl. Acad. Sci., 99 (2002), pp. 3684–3689.

[19] K. Brudey, J. R. Driscoll, L. Rigouts, et al., *Mycobacterium tuberculosis complex genetic diversity: Mining the fourth international spoligotyping database (SpolDB4) for classification, population genetics and epidemiology*, BMC Microbiol., 6 (2006), p. 23.

[20] Kevin P Cain, Stephen R Benoit, Carla A Winston, et al., *Tuberculosis among foreign-born persons in the United States*, The J. Am. Méd. Assoc., 300 (2008), pp. 405–412.

[21] L.L. Cavalli-Sforza and A.W.F. Edwards, *Phylogenetic analysis. Models and estimation procedures*, Am. J. Hum. Genet., 19 (1967), p. 233.

[22] Maxine Caws, Guy Thwaites, Sarah Dunstan, et al., *The influence of host and bacterial genotype on the development of disseminated disease with Mycobacterium tuberculosis*, PLoS Pathog., 4 (2008), p. e1000034.

[23] CDC, *Guide to the Application of Genotyping to Tuberculosis Prevention and Control*, 2011. Date last accessed Aug. 23, 2013 http://www.cdc.gov/tb/programs/genotyping/manual.htm.

[24] A. Civril, M. Magdon-Ismail, and E. Bocek-Rivele, *SDE: Graph drawing using spectral distance embedding*, in Graph Drawing, Karlsruhe, Germany, 2006, Springer, pp. 512–513.

[25] P. Constant, E. Perez, W. Malaga, et al., *Role of the pks15/1 gene in the biosynthesis of phenolglycolipids in the Mycobacterium tuberculosis complex. evidence that all strains synthesize glycosylated p-hydroxybenzoic methyl esters and that strains devoid of phenolglycolipids harbor a frameshift mutation in the pks15/1 gene*, J. Biol. Chem., 277 (2002), pp. 38148–38158.

[26] M. Coscolla and S. Gagneux, *Does M. tuberculosis genomic diversity explain disease diversity?*, Drug Discov. Today: Dis. Mech., 7 (2010), pp. e43–e59.

[27] L. S. Cowan, L. Mosher, L. Diem, et al., *Variable-number-tandem repeat typing of Mycobacterium tuberculosis isolates with low copy numbers of IS6110 by using mycobacterial interspersed repetitive units*, J. Clin. Microbiol., 40 (2002), pp. 1592–1602.

[28] T.F Cox and M.A.A. Cox, *Multidimensional Scaling,Second Edition*, Chapman and Hall, London, UK, 2001.

[29] B. C. de Jong, P. C. Hill, A. Aiken, et al., *Progression to active tuberculosis, but not transmission, varies by Mycobacterium tuberculosis lineage in the gambia*, J. Infect. Dis., 198 (2008), pp. 1037–1043.

[30] J. De Leeuw, *Applications of convex analysis to multidimensional scaling*, Recent Dev. Stat., (1977), pp. 133–146.

[31] A. Denise, M. Vasconcellos, and D.J.A. Welsh, *The random planar graph*, Congr. Numerant., (1996), pp. 61–80.

[32] G. Di Battista, A. Garg, G. Liotta, et al., *An experimental comparison of four graph drawing algorithms*, Comput. Geom., 7 (1997), pp. 303–325.

[33] T. Dwyer, Y. Koren, and K. Marriott, *Drawing directed graphs using quadratic programming*, IEEE Transactions on Vis. Comput. Graph., 12 (2006), pp. 536–548.

[34] T. Dwyer, K. Marriott, and P. J. Stuckey, *Fast node overlap removal*, in Graph Drawing, Karlsruhe, Germany, 2006, Springer, pp. 153–164.

[35] P. Erdös and R.K. Guy, *Crossing number problems*, The Am. Math. Mon., 80 (1973), pp. 52–58.

[36] J. D. Ernst, G. Trevejo-Nunez, and N. Banaiee, *Genomics and the evolution, pathogenesis, and diagnosis of tuberculosis*, J. Clin. Investig., 117 (2007), pp. 1738–1745.

[37] Z. Fang, N. Morrison, B. Watt, et al., *IS6110 transposition and evolutionary scenario of the direct repeat locus in a group of closely related Mycobacterium tuberculosis strains*, J Bacteriol., 180 (1998), pp. 2102–2109.

[38] I. Fáry, *On straight line representation of planar graphs*, Acta Sci. Math. Szeged., 11 (1948), pp. 229–233.

[39] S. Ferdinand, G. Valetudie, C. Sola, et al., *Data mining of Mycobacterium tuberculosis complex genotyping results using mycobacterial interspersed repetitive units validates the clonal structure of spoligotyping-defined families*, Res. Microbiol., 155 (2004), pp. 647–654.

[40] I. Filliol, J. R. Driscoll, D. van Soolingen, et al., *Global distribution of Mycobacterium tuberculosis spoligotypes*, Emerg. Infect. Dis., 8 (2002), pp. 1347–1349.

[41] I. Filliol, J. R. Driscoll, D. van Soolingen, et al., *Snapshot of moving and expanding clones of Mycobacterium tuberculosis and their global distribution assessed by spoligotyping in an international study*, J. Clin. Microbiol., 41 (2003), pp. 1963–1970.

[42] I. Filliol, A. S. Motiwala, M. Cavatore, et al., *Global phylogeny of Mycobacterium tuberculosis based on single nucleotide polymorphism (SNP) analysis: Insights into tuberculosis evolution, phylogenetic accuracy of other DNA fingerprinting systems, and recommendations for a minimal standard SNP set*, J. Bacteriol., 188 (2006), pp. 759–772.

[43] L. Flores, T. Van, S. Narayanan, et al., *Large sequence polymorphisms classify Mycobacterium tuberculosis strains with ancestral spoligotyping patterns*, J. Clin. Microbiol., 45 (2007), pp. 3393–3395.

[44] R. Frothingham, H. G. Hills, and K. H. Wilson, *Extensive DNA-sequence conservation throughout the Mycobacterium tuberculosis complex*, J. Clin. Microbiol., 32 (1994), pp. 1639–1643.

[45] T. Fruchterman and E. Reingold, *Graph drawing by force-directed placement*, Software-Practice Exp., 21 (1991), pp. 1129–1164.

[46] S. Gagneux, *Host–pathogen coevolution in human tuberculosis*, Philos. Transactions Royal Soc. B: Biol. Sci., 367 (2012), pp. 850–859.

[47] S. Gagneux, K. DeRiemer, T. Van, et al., *Variable host-pathogen compatibility in Mycobacterium tuberculosis*, Proc. Natl. Acad. Sci., 103 (2006), pp. 2869–2873.

[48] S. Gagneux and P. M. Small, *Global phylogeography of Mycobacterium tuberculosis and implications for tuberculosis product development*, Lancet Infect. Dis., 7 (2007), pp. 328–337.

[49] E Gansner, Y Koren, and S North, *Graph drawing by stress majorization*, in Graph Drawing, vol. 3383, New York City, NY, 2004, pp. 239–250.

[50] E. R. Gansner and Y. Hu, *Efficient node overlap removal using a proximity stress model*, in Graph Drawing, Chicago, IL, 2009, Springer, pp. 206–217.

[51] E. R. Gansner, Y. Hu, and S. North, *Visualizing streaming text data with dynamic graphs and maps*, in Graph Drawing, Bordeaux, France, 2013, Springer, pp. 439–450.

[52] M.R. Garey and D.S. Johnson, *Crossing number is NP-complete*, SIAM J. on Algebraic Discret. Methods., 4 (1983), p. 312.

[53] L. Getoor, J. T. Rhee, D. Koller, et al., *Understanding tuberculosis epidemiology using structured statistical models*, Artif. Intell. Med., 30 (2004), pp. 233–256.

[54] D. B. Goldstein, L. A. Ruiz, L.L. Cavalli-Sforza, et al., *Genetic absolute dating based on microsatellites and the origin of modern humans*, Proc. Natl. Acad. Sci., 92 (1995), pp. 6723–6727.

[55] J. Gorski, F. Pfeuffer, and K. Klamroth, *Biconvex sets and optimization with biconvex functions: A survey and extensions*, Math. Methods Oper. Res., 66 (2007), pp. 373–407.

[56] M. M. Gutacker, B. Mathema, H. Soini, et al., *Single-nucleotide polymorphism-based population genetic analysis of Mycobacterium tuberculosis strains from 4 geographic sites*, J. Infect. Dis., 193 (2006), pp. 121–128.

[57] M. M. Gutacker, J. C. Smoot, C. A. Migliaccio, et al., *Genome-wide analysis of synonymous single nucleotide polymorphisms in Mycobacterium tuberculosis complex organisms: Resolution of genetic relationships among closely related microbial strains*, Genet., 162 (2002), pp. 1533–1543.

[58] C. Gutwenger and P. Mutzel, *An experimental study of crossing minimization heuristics*, in Graph Drawing, New York City, NY, 2004, Springer, pp. 13–24.

[59] S. Hachul and M. Jünger, *Drawing large graphs with a potential-field-based multilevel algorithm*, in Graph Drawing, Limerick, Ireland, 2005, Springer, pp. 285–295.

[60] D. Harel and Y. Koren, *Graph drawing by high-dimensional embedding*, in Graph Drawing, Irvine, CA, 2002, Springer, pp. 207–219.

[61] R. Hershberg, M. Lipatov, P. M. Small, et al., *High functional diversity in Mycobacterium tuberculosis driven by genetic drift and human demography*, PLoS Biol., 6 (2008), p. e311.

[62] G. Hinton and S.T. Roweis, *Stochastic neighbor embedding*, Adv. Neural Inf. Process. Syst., 15 (2002), pp. 833–840.

[63] A. E. Hirsh, A. G. Tsolaki, K. DeRiemer, et al., *Stable association between strains of Mycobacterium tuberculosis and their human host populations*, Proc. Natl. Acad. Sci. United States Am., 101 (2004), pp. 4871–4876.

[64] M. Jünger and P. Mutzel, *Maximum planar subgraphs and nice embeddings: Practical layout tools*, Algorithmica., 16 (1996), pp. 33–59.

[65] M. Junger and P. Mutzel, *2-layer straightline crossing minimization: Performance of exact and heuristic algorithms*, J. Graph Algorithms Appl., 1 (1997), pp. 1–25.

[66] J. Kamerbeek, L. Schouls, A. Kolk, et al., *Simultaneous detection and strain differentiation of Mycobacterium tuberculosis for diagnosis and epidemiology*, J. Clin. Microbiol., 35 (1997), pp. 907–914.

[67] M. Kato-Maeda, EY Kim, L. Flores, et al., *Differences among sublineages of the east-asian lineage of Mycobacterium tuberculosis in genotypic clustering*, The Int. J. Tuberc. Lung Dis., 14 (2010), pp. 538–544.

[68] Y. Koren, *On spectral graph drawing*, Comput. Comb., (2003), pp. 496–508.

[69] K. Kremer, C. Arnold, A. Cataldi, et al., *Discriminatory power and reproducibility of novel DNA typing methods for Mycobacterium tuberculosis complex strains*, J. Clin. Microbiol., 43 (2005), pp. 5628–5638.

[70] K. Kremer, D. van Soolingen, R. Frothingham, et al., *Comparison of methods based on different molecular epidemiological markers for typing of Mycobacterium tuberculosis complex strains: Interlaboratory study of discriminatory power and reproducibility*, J. Clin. Microbiol., 37 (1999), pp. 2607–2618.

[71] K. Kuratowski, *Sur le probleme des courbes gauches en topologie*, Fund. Math., 15 (1930), p. 79.

[72] E. Legrand, I. Filliol, C. Sola, et al., *Use of spoligotyping to study the evolution of the direct repeat locus by IS6110 transposition in Mycobacterium tuberculosis*, J. Clin. Microbiol., 39 (2001), pp. 1595–1599.

[73] M. Mäkelä, *Survey of bundle methods for nonsmooth optimization*, Optim. Methods Softw., 17 (2002), pp. 1–29.

[74] B. Mathema, N. E. Kurepina, P. J. Bifani, et al., *Molecular epidemiology of tuberculosis: Current insights*, Clin. Microbiol. Rev., 19 (2006), pp. 658–685.

[75] J. M. Musser, A. Amin, and S. Ramaswamy, *Negligible genetic diversity of Mycobacterium tuberculosis host immune system protein targets: Evidence of limited selective pressure*, Genet., 155 (2000), pp. 7–16.

[76] P. Mutzel, *An alternative method to crossing minimization on hierarchical graphs*, in Graph Drawing, Rome, Italy, 1997, Springer, pp. 318–333.

[77] Y. Nesterov, *Smooth minimization of non-smooth functions*, Math. Program., 103 (2005), pp. 127–152.

[78] J. Nocedal and S. Wright, *Numerical Optimization, Series in Operations Research and Financial Engineering*, Springer, New York, NY, 2006.

[79] C. Ozcaglar, A. Shabbeer, S. Vandenberg, et al., *Sublineage structure analysis of Mycobacterium tuberculosis complex strains using multiple-biomarker tensors*, BMC Genom., 12 (2011), p. S1.

[80] D. Portevin, S. Gagneux, I. Comas, et al., *Human macrophage responses to clinical isolates from the Mycobacterium tuberculosis complex discriminate between ancient and modern lineages*, PLoS Path., 7 (2011), p. e1001307.

[81] H. Purchase, *Which aesthetic has the greatest effect on human understanding?*, in Graph Drawing, Rome, Italy, 1997, Springer, pp. 248–261.

[82] M. B. Reed, P. Domenech, C. Manca, et al., *A glycolipid of hypervirulent tuberculosis strains that inhibits the innate immune response*, Nat., 431 (2004), pp. 84–87.

[83] M. B. Reed, V. K. Pichler, F. McIntosh, et al., *Major Mycobacterium tuberculosis lineages associate with patient country of origin*, J. Clin. Microbiol., 47 (2009), pp. 1119–1128.

[84] J. F. Reyes, A. R. Francis, and M. M. Tanaka, *Models of deletion for visualizing bacterial variation: An application to tuberculosis spoligotypes*, BMC Bioinforma., 9 (2008), p. 496.

[85] R.T. Rockafellar, *Convex Analysis*, vol. 28, Princeton University Press, Princeton, NJ, 1997.

[86] S. T. Roweis and L. K. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, Sci., 290 (2000), p. 2323.

[87] A. Ruszczynski, *Nonlinear Optimization*, vol. 13, Princeton University Press, Princeton, NJ, 2011.

[88] M. Sebban, I. Mokrousov, N. Rastogi, et al., *A data-mining approach to spacer oligonucleotide typing of Mycobacterium tuberculosis*, Bioinforma., 18 (2002), pp. 235–243.

[89] A. Shabbeer, L. Cowan, C. Ozcaglar, et al., *TB-Lineage: An online tool for classification and analysis of strains of Mycobacterium tuberculosis complex*, Infect. Genet. Evol., 12 (2012), pp. 789–797.

[90] A. Shabbeer, C. Ozcaglar, B. Yener, et al., *Web tools for molecular epidemiology of tuberculosis*, Infect. Genet. Evol., 12 (2012), pp. 767–781.

[91] Ben Shneiderman, *Tree visualization with tree-maps: 2-d space-filling approach*, ACM Transactions on graphics (TOG)., 11 (1992), pp. 92–99.

[92] M. D. Shriver, L. Jin, E. Boerwinkle, et al., *A novel measure of genetic-distance for highly polymorphic tandem repeat loci*, Mol. Biol. Evol., 12 (1995), pp. 914–920.

[93] C. Sola, I. Filliol, M. C. Gutierrez, et al., *Spoligotype database of Mycobacterium tuberculosis: Biogeographic distribution of shared types and epidemiologic and phylogenetic perspectives*, Emerg. Infect. Dis., 7 (2001), pp. 390–396.

[94] S. Sreevatsan, X. Pan, K. E. Stockbauer, et al., *Restricted structural gene polymorphism in the Mycobacterium tuberculosis complex indicates evolutionarily recent global dissemination*, Proc. Natl. Acad. Sci., 94 (1997), pp. 9869–9874.

[95] E. M. Streicher, T. C. Victor, G. van der Spuy, et al., *Spoligotype signatures in the Mycobacterium tuberculosis complex*, J. Clin. Microbiol., 45 (2007), pp. 237–240.

[96] K. Sugiyama and K. Misue, *Graph drawing by the magnetic spring model*, J. Vis. Lang. Comput., 6 (1995), pp. 217–231.

[97] T. Sun and S. Chen, *Locality preserving cca with applications to data visualization and pose estimation*, Image Vis. Comput., 25 (2007), pp. 531–543.

[98] Y. J. Sun, R. Bellamy, A. S. Lee, et al., *Use of mycobacterial interspersed repetitive unit-variable-number tandem repeat typing to examine genetic diversity of Mycobacterium tuberculosis in Singapore*, J. Clin. Microbiol., 42 (2004), pp. 1986–1993.

[99] P. Supply, C. Allix, S. Lesjean, et al., *Proposal for standardization of optimized mycobacterial interspersed repetitive unit-variable-number tandem repeat typing of Mycobacterium tuberculosis*, J. Clin. Microbiol., 44 (2006), pp. 4498–4510.

[100] P. Supply, S. Lesjean, E. Savine, et al., *Automated high-throughput genotyping for study of global epidemiology of Mycobacterium tuberculosis based on mycobacterial interspersed repetitive units*, J. Clin. Microbiol., 39 (2001), pp. 3563–3571.

[101] R. Tamassia, *Handbook of graph drawing and visualization*, Chapman & Hall/CRC, London, UK, 2007.

[102] R. Tamassia and I.G. Tollis, *Planar grid embedding in linear time*, IEEE Transactions on Circuits Syst., 36 (1989), pp. 1230–1234.

[103] LJP Van der Maaten, EO Postma, and HJ Van Den Herik, *Dimensionality reduction: A comparative review*, J. Mach. Learn. Res., 10 (2009), pp. 1–41.

[104] G. D. van der Spuy, K. Kremer, S. L. Ndabambi, et al., *Changing Mycobacterium tuberculosis population highlights clade-specific pathogenic characteristics*, Tuberc. (Edinb)., 89 (2009), pp. 120–125.

[105] J. D. van Embden, M. D. Cave, J. T. Crawford, et al., *Strain identification of Mycobacterium tuberculosis by DNA fingerprinting: Recommendations for a standardized methodology*, J. Clin. Microbiol., 31 (1993), pp. 406–409.

[106] D. Van Soolingen, *Molecular epidemiology of tuberculosis and other mycobacterial infections: Main methodologies and achievements*, J. Intern. Med., 249 (2001), pp. 1–26.

[107] I. Vitol, J. Driscoll, B. Kreiswirth, et al., *Identifying Mycobacterium tuberculosis complex strain families using spoligotypes*, Infect. Genet. Evol., 6 (2006), pp. 491–504.

[108] R. M. Warren, E. M. Streicher, S. L. Sampson, et al., *Microevolution of the direct repeat region of Mycobacterium tuberculosis: Implications for interpretation of spoligotyping data*, J. Clin. Microbiol., 40 (2002), pp. 4457–4465.

[109] R. E. Wendell and A. P. Hurter, *Minimization of a non-separable objective function subject to disjoint constraints*, Oper. Res., 24 (1976), pp. 643–657.

[110] G.J. Wills, *Nicheworks: Interactive visualization of very large graphs*, J. Comput. Graph. Stat., 8 (1999), pp. 190–212.

[111] W. I. Zangwill, *Convergence conditions for nonlinear programming algorithms*, Manag. Sci., 16 (1969), pp. 1–13.

[112] X. Zhu, S. Chang, K. Fang, et al., *MyBASE: A database for genome polymorphism and gene function studies of mycobacterium*, BMC Microbiol., 9 (2009), p. 40.

APPENDIX A
APPENDIX

A.1 Graphs and Spoligoforests Generated by MAA

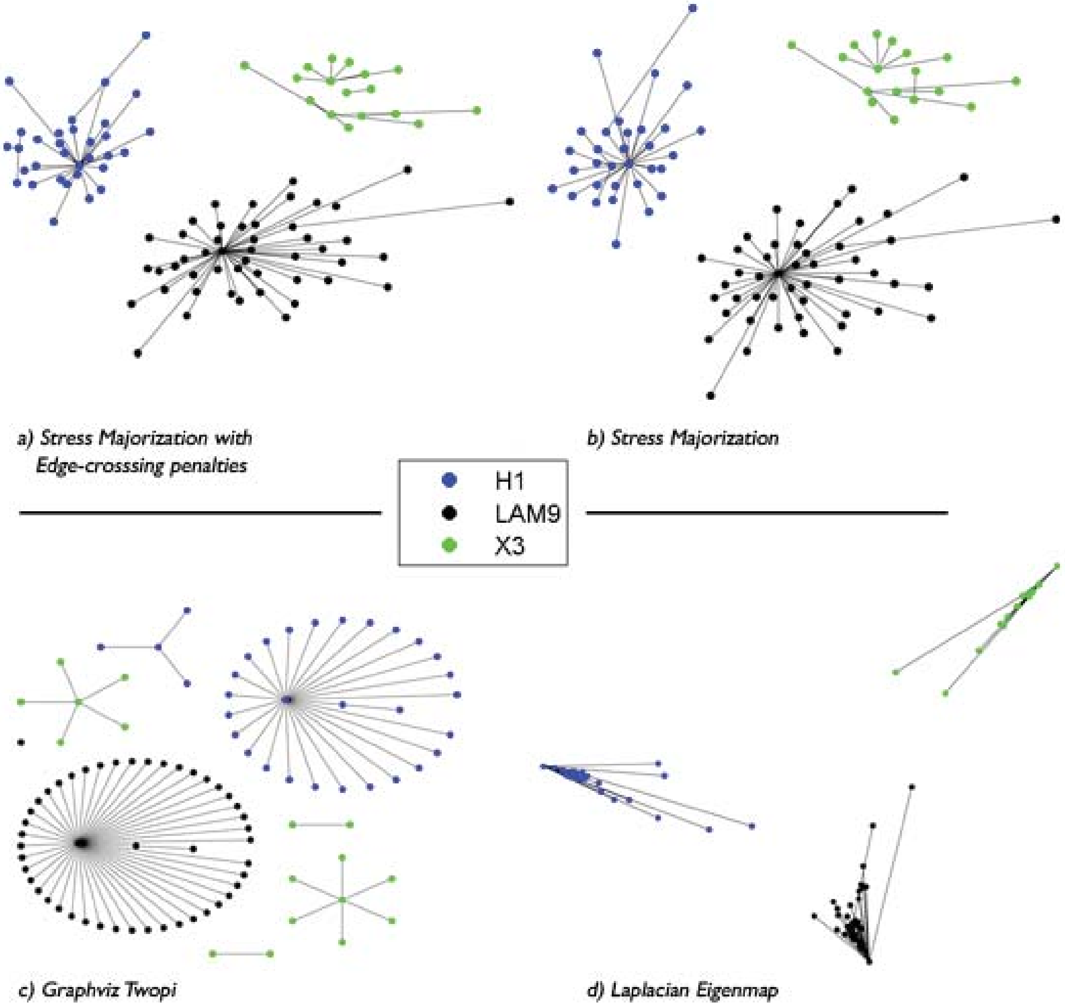
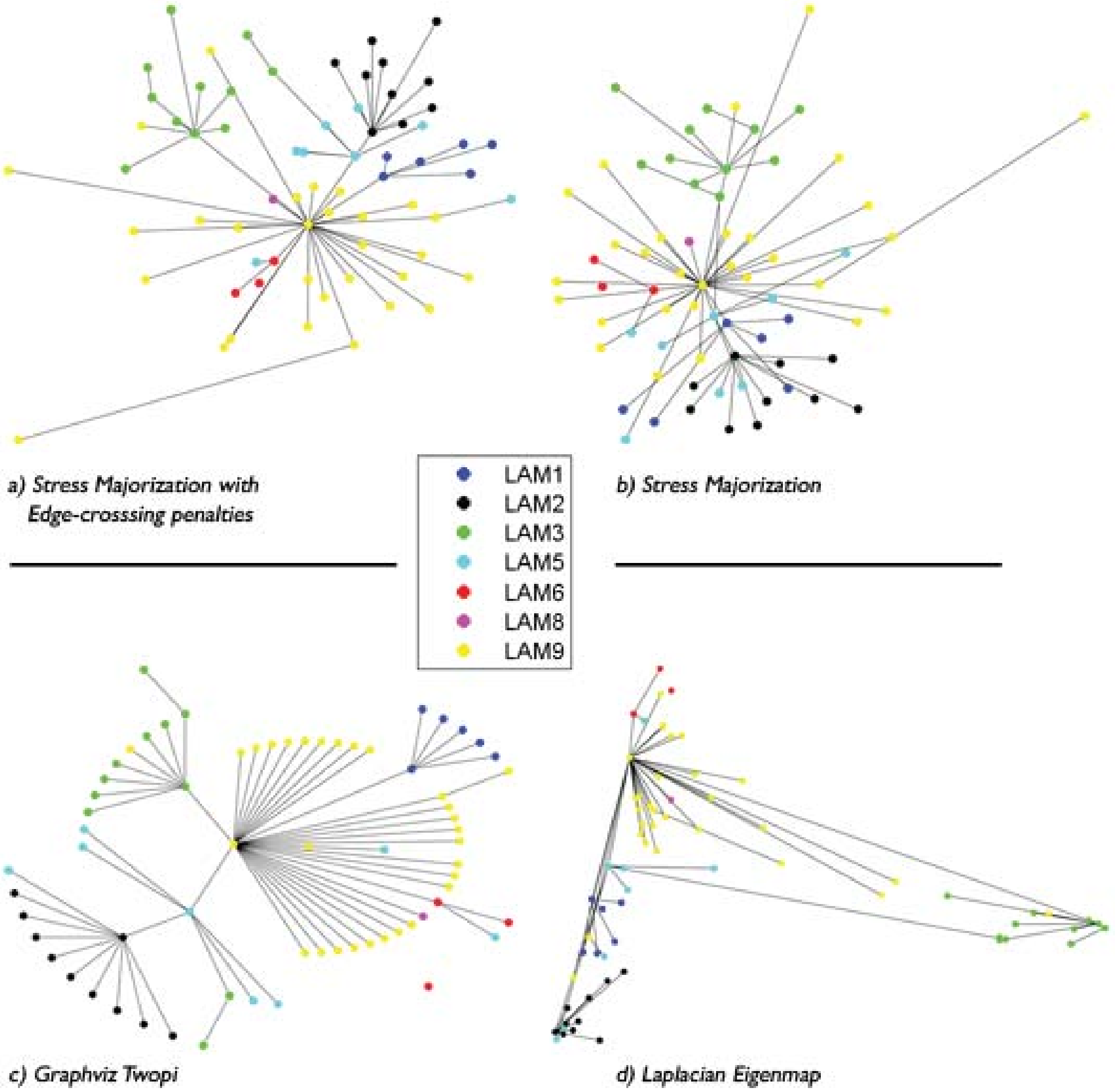


Figure A.1: Embeddings of spoligoforests of Haarlem, X, and LAM sublineages. Graph (b), that optimizes the MDS objective and generated using Neato, preserves proximity relations but has edge-crossings. In graph (a), the proposed approach eliminates all edge crossings with little change in the overall stress.



a) Stress Majorization with Edge-crosssing penalties

b) Stress Majorization

c) Graphviz Twopi

d) Laplacian Eigenmap

Figure A.2: Embeddings of spoligoforests of LAM (Latin-American-Mediterranean) sublineages using (a)MAA, (b)Neato, (c)Twopi and (d) Laplacian Eigenmaps. The proposed method, MAA, eliminates all edge crossings and shows the most genetically relevant arrangements within the sublineages.

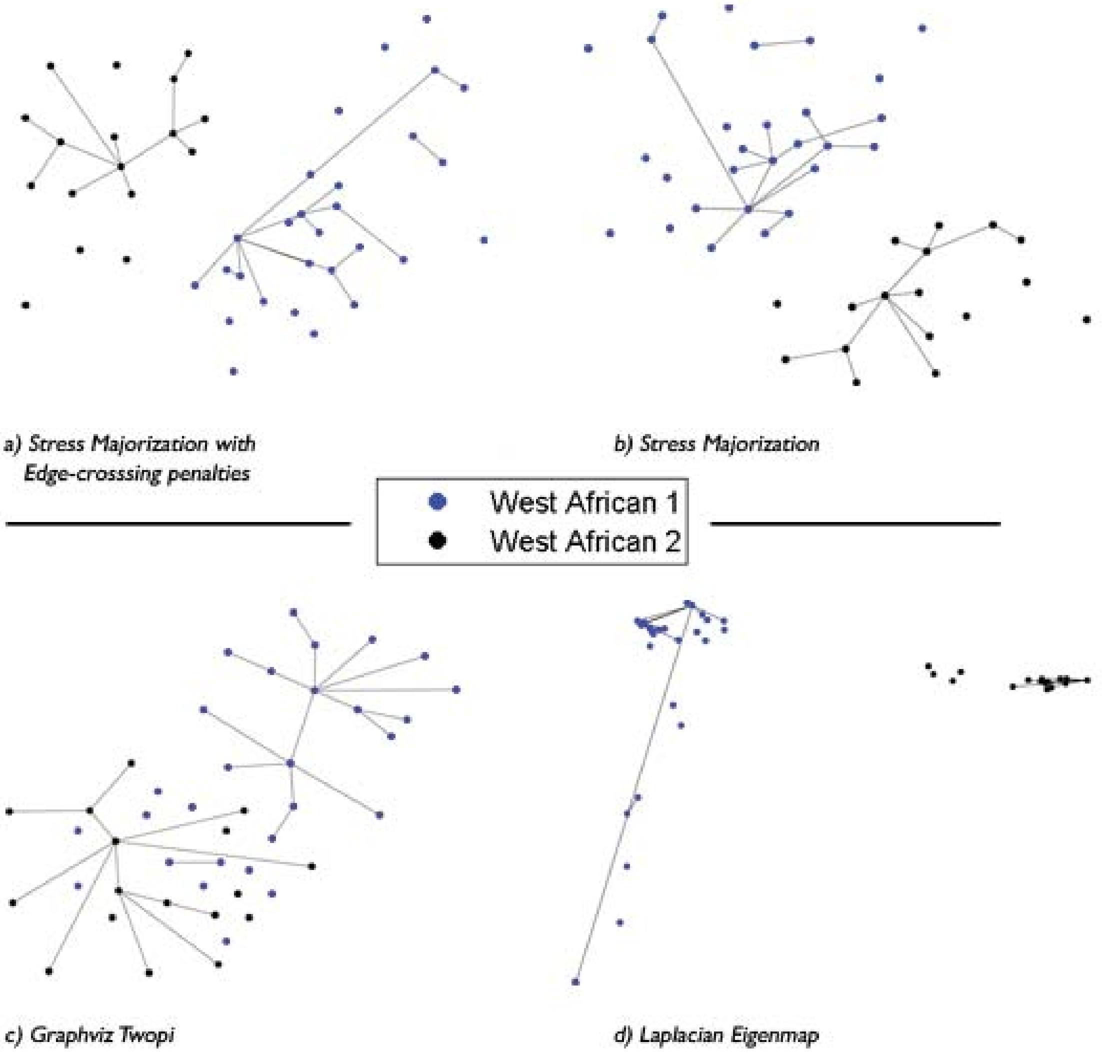


Figure A.3: Embeddings of spoligoforests of *M. africanum* sublineages. The *M. africanum* lineage is divided into two distinct sublineages. However, the distinction between the two sublineage is not visible in graph (c) produced using the radial graph drawing algorithm Twopi. Graph (b), drawn using Neato tool with stress majorization, clearly shows the separation, but is difficult to understand because of edge crossings. Graph (d), drawn using Laplacian eigenmaps preserves proximity, but the edge-crossings are not eliminated. Graph (a), drawn using the proposed approach eliminates all edge crossings with little change in the overall stress, while preserving proximity between genetically related strains.

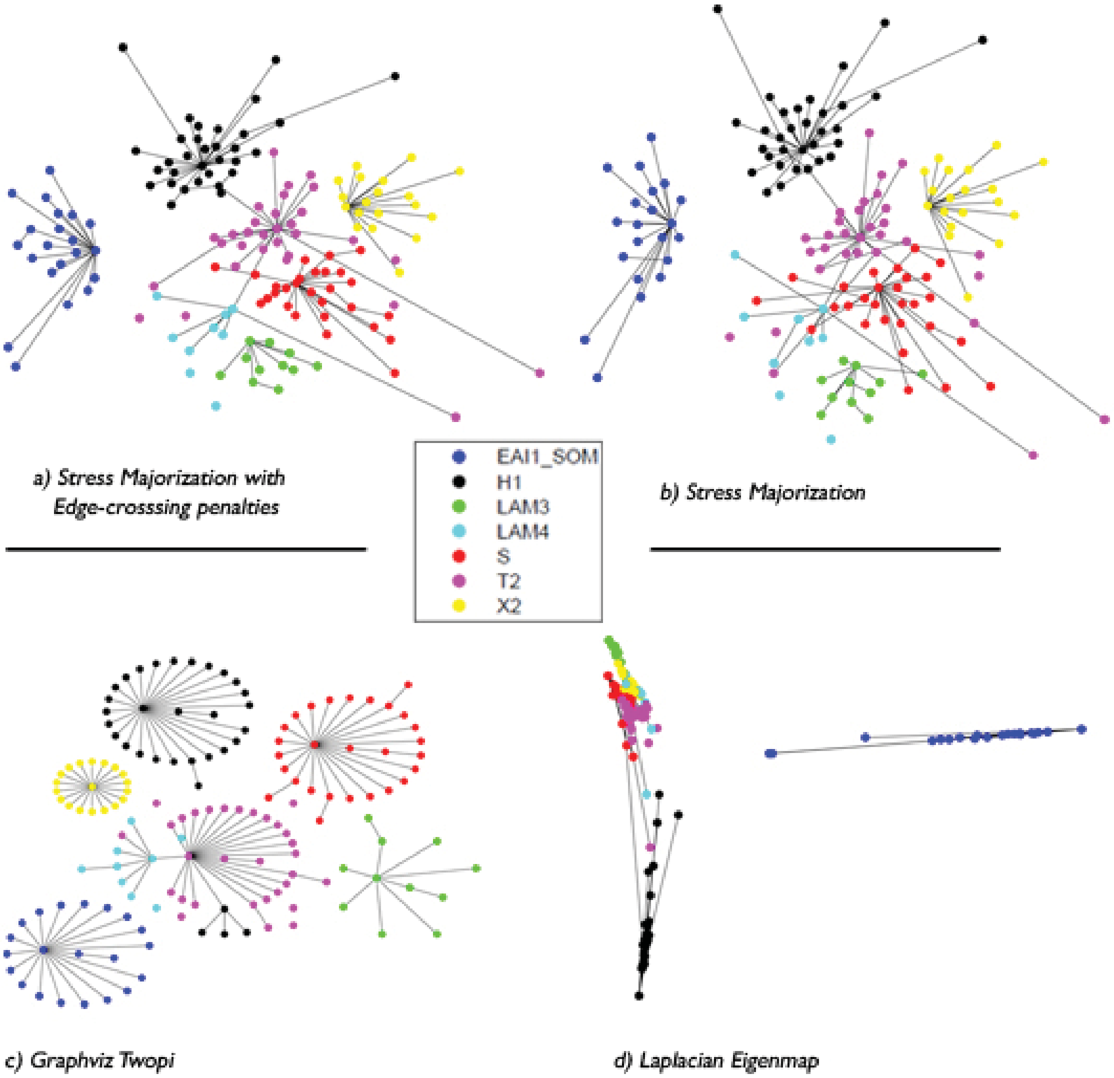
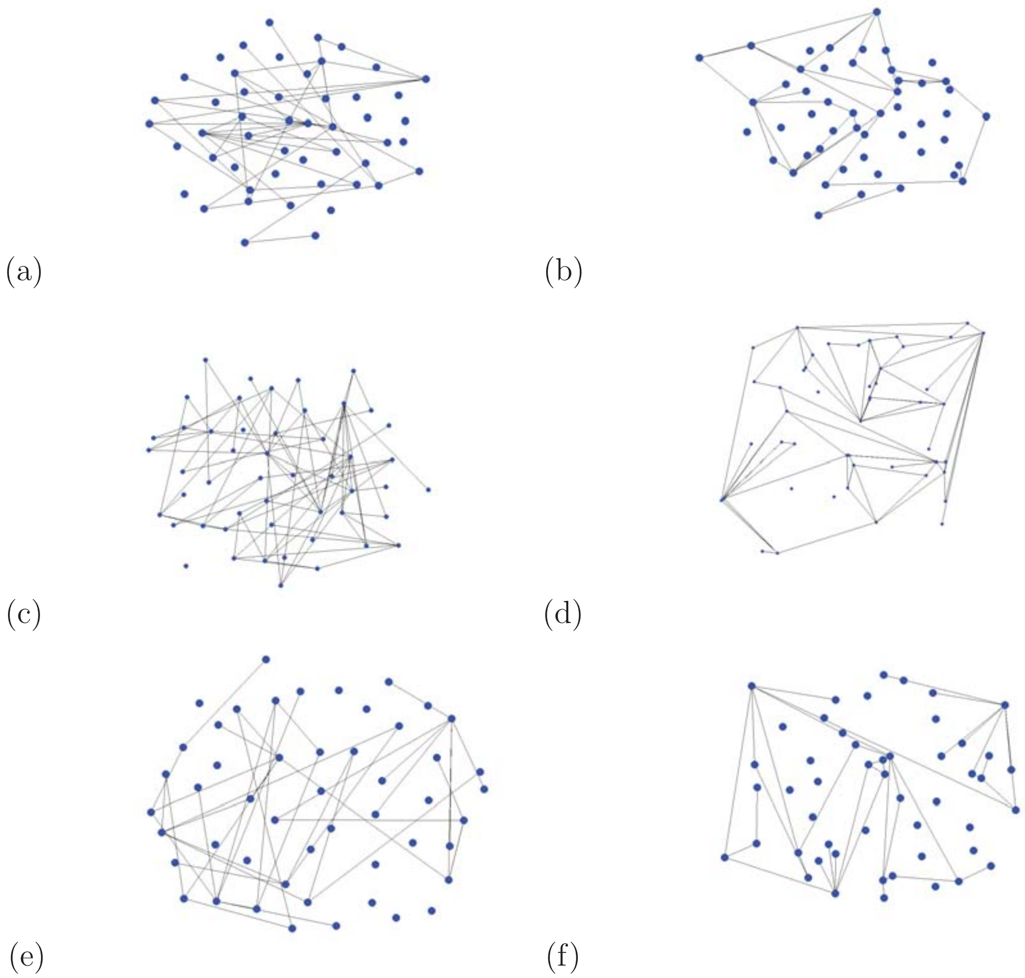


Figure A.4: Embeddings of spoligoforests of 7 SpolDB4 sublineages. Notice that there are many connected components in the graph. Graph (b), drwan using Neato with stress majorization preserves proximity, but otroduces edge crossings. Graph (c), drawn using radial layout implemented in Graphviz Twopi tool, eliminates edge crossings, but the proximity of genetically related strains belonging to different connected components is not preserved. In graph (d), Laplacian Eigenmap embedding based on weighted Laplacian groups genetically related strainsclosely, but there are edge crossings which makes it harder to distinguish the nodes. In graph (a), the proposed approach eliminates all edge crossings with little change in the overall stress.



(a)

(b)

(c)

(d)

(e)

(f)

Figure A.5: Embeddings for randomly generated graph in $\mathbb{R}^7$ with 50 nodes and 40 edges using (a) Stress majorization and (b) MAA. Embeddings for randomly generated graph in $\mathbb{R}^7$ with 50 nodes and 40 edges using (c) Stress majorization and (d) MAA. Embeddings for randomly generated graph in $\mathbb{R}^7$ with 50 nodes and 40 edges using (e) Stress majorization and (f) MAA.

A.2 Spoligoforests Generated by MAA+

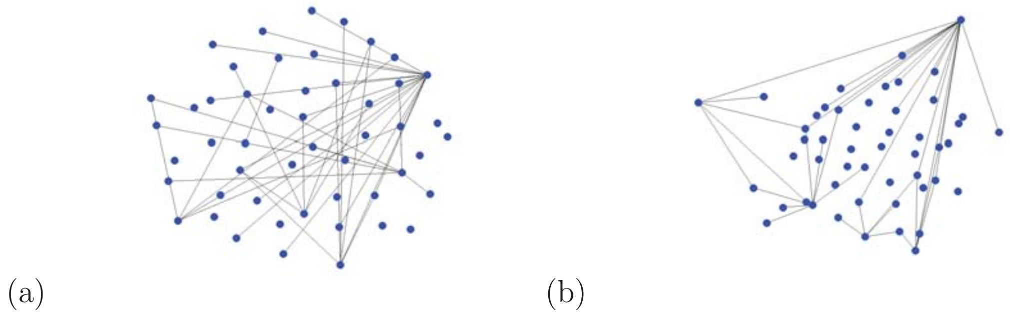


Figure A.6: Embeddings for randomly generated graph in $\mathbb{R}^7$ with 50 nodes and 40 edges using (a) Stress majorization and (b) MAA.

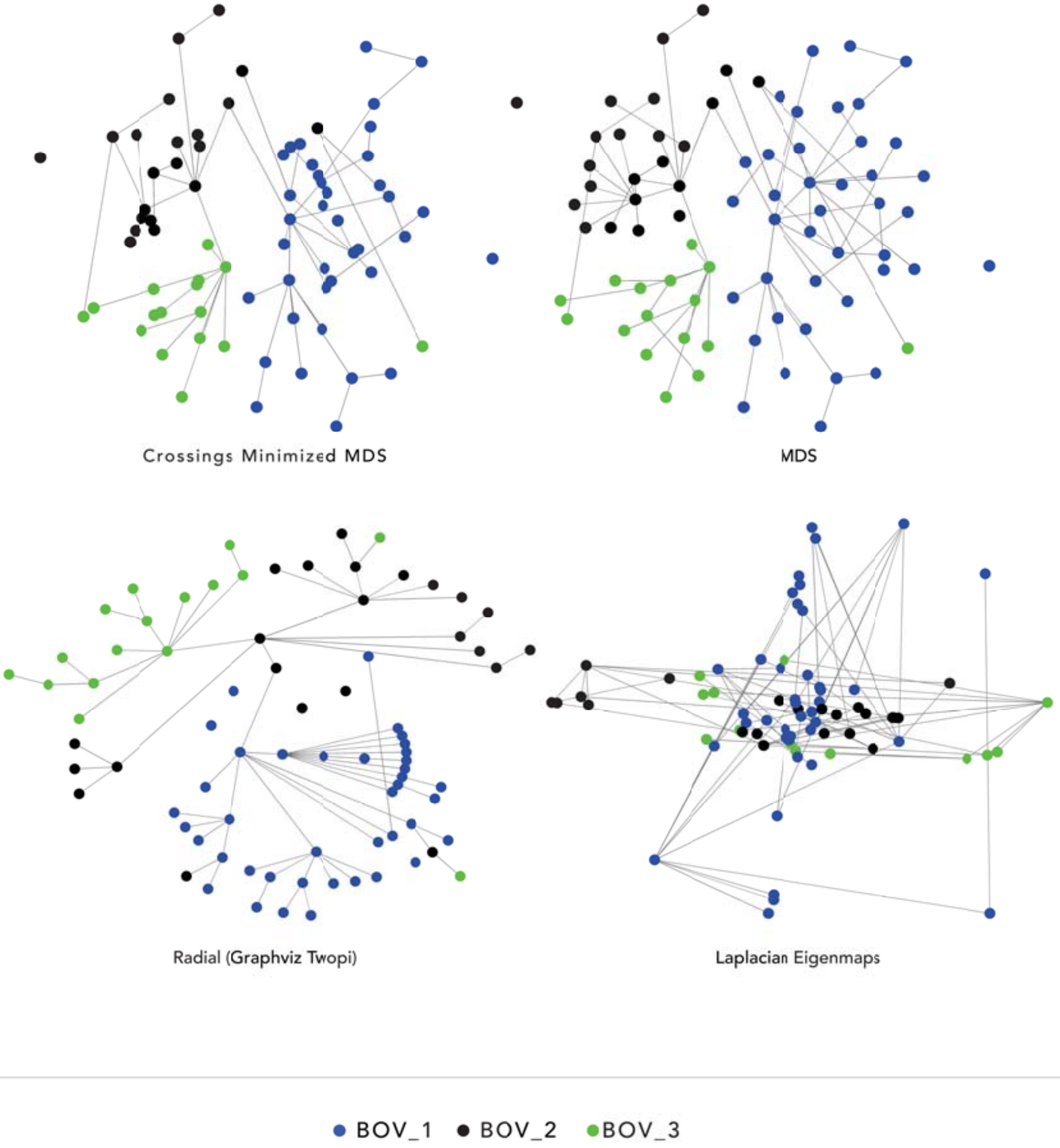


Figure A.7: Spoligoforest for all sublineages BOV_1, BOV_2 and BOV_3 of *M. bovis*. Layout generated using (a) MAA+ (b)MDS (c) GraphViz Twopi (d) Laplacian Eigenmaps. MAA+ improves on MDS in number of crossings with marginal increase in stress.

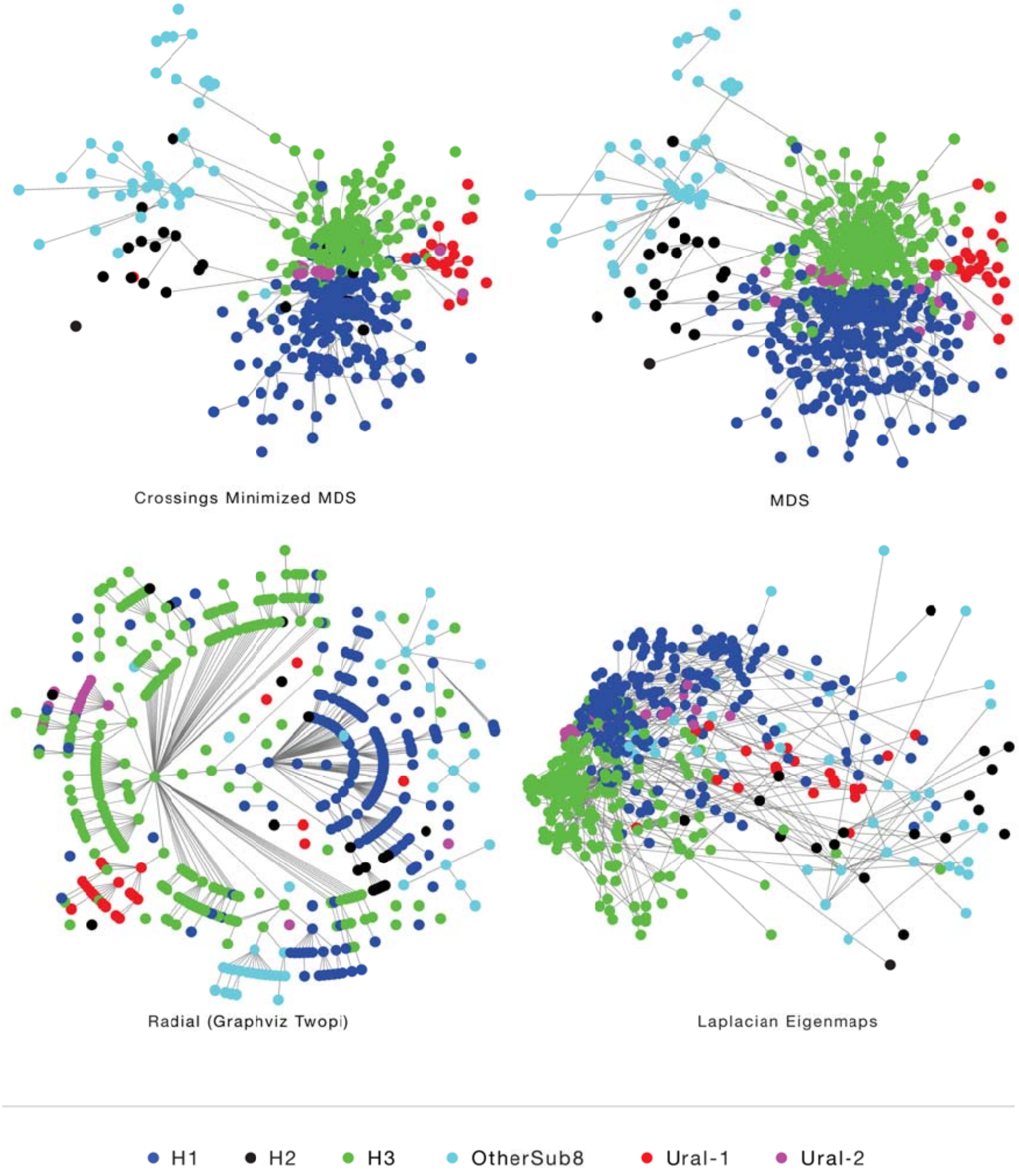


Figure A.8: Spoligoforest for all sublineages predicted to belong to the Euro-American Haarlem lineage. Layout generated using (a) MAA+ (b)MDS (c) GraphViz Twopi (d) Laplacian Eigenmaps. MAA+ improves on MDS in number of crossings with marginal increase in stress.

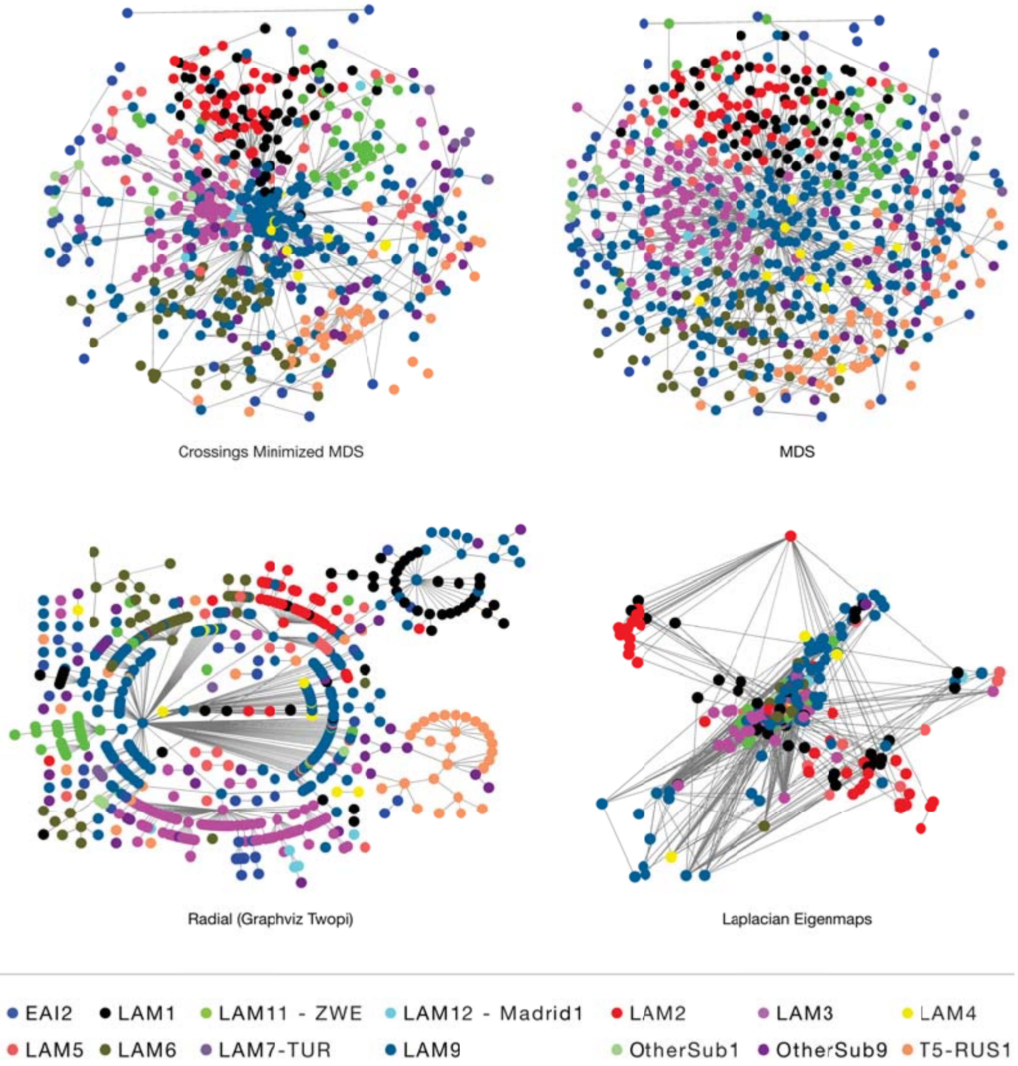


Figure A.9: **Spoligoforest for all sublineages predicted to belong to the Euro-American LAM lineage. Layout generated using (a) MAA+ (b)MDS (c) GraphViz Twopi (d) Laplacian Eigenmaps. MAA+ improves on MDS in number of crossings with marginal increase in stress.**