UNIVERSITY OF CALIFORNIA, SAN DIEGO

Optimization Algorithms for Biological Data

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Banu Dost

Committee in charge:

Professor Vineet Bafna, Chair
Professor Steven P. Briggs
Professor Trey Ideker
Professor Pavel A. Pevzner
Professor Lawrence Saul

2010

The dissertation of Banu Dost is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2010

DEDICATION

To my parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First and foremost, I would like to gratefully acknowledge the support and encouragement of my advisor Prof. Vineet Bafna. This dissertation would not have been possible without your guidance and inspiration. I have learned and benefited a lot from your professional expertise, wisdom, patience, and kindness. Thank you very much for sharing them so generously with me along the way.

I would also like to thank all the members of my committee: Prof. Steven Briggs, Prof. Trey Ideker, Prof. Pavel Pevzner and, Prof. Lawrence Saul. Their comments and suggestions during various committee and individual meetings have been instrumental in improving my dissertation.

I have worked with various collaborators, and my thesis has benefited greatly from their insights. Specifically, Dr. Nuno Bandeira who has always been available for long discussions has been vital to my research. Thanks are also due to my other collaborators Roded Sharan and Tomer Shlomi from Tel Aviv University, Andrew Su and Chunlei Wu from Genomics Research Foundation of Novartis, Shaojie Zhang, Buhm Han, Steven Briggs, Xianglian Li, and Zhouxin Shen from UC, San Diego. I am grateful to all my colleagues in the Bafna and Pevzner laboratories for their suggestions and comments in weekly lab meetings. I give special thanks to Neil Jones, Steven Tanner, Ali Bashir and fellow students Julio Ng and Sangtae Kim for discussions. I am grateful to all our collaborating labs for providing their data without which our work would not have been possible.

Finally, I am forever indebted to my parents Mr. Abdulkadir Dost and Mrs. Solmaz Dost for unconditionally providing their love, support, guidance, and encouragement at every stage of my life. I am also grateful to my brother Ilkay, my sisters Ebru and Feyza, and my friends Aslihan, Murat, Gokce, Christena, Ines and many others. Last, I am more grateful to Vasileios – my closest friend, inspiration, support in the last two years – for always being there for me and for your wonderful sense of humor even at very tough times.

Chapter 2 is, in full, a reprint of the paper "Structural alignment of pseudoknotted RNA. B. Han, B. Dost, S. Zhang and V. Bafna (2008). Journal of Computational Biology. 15(5): 489-504". The dissertation author was the primary investigator and first author of this paper jointly with Buhm Han. This research was, in part, previously published in the conference proceedings of Recomb 2006.

Chapter 3 is, in full, a reprint of the paper “QNet: a tool for querying protein interaction networks. B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan (2008). Journal of Computational Biology. 15(7):913-25”. The dissertation author was the primary investigator and author of this paper jointly with Tomer Shlomi. This research was, in part, previously published in the conference proceedings of Recomb 2007.

Chapter 4 is, in full, a reprint of the paper “A fast algorithm for clustering genome-scale expression data. B. Dost, A. Su, C. Wu, and V.Bafna (2008). EEE/ACM Transactions on Computational Biology and Bioinformatics. In press”. The dissertation author was the primary investigator and author of this paper.

Chapter 5 is, in full, a reprint of the submitted paper “Accurate Mass Spectrometry Based Protein Quantification via Shared Peptides. B. Dost, N. Bandeira, X. Li, Z. Shen, S. Briggs and V. Bafna”. The dissertation author was the primary investigator and author of this paper. This research was also previously published in the conference proceedings of Recomb 2009.

Chapter 6 is in preparation for publication as “Identification and Quantification of Post-translational Modification Variants and False Discovery Rates”. B. Dost, V. Bafna and N. Bandeira (2010). In preparation”. The dissertation author was the primary author of this paper.

VITA

2004	B. S. in Computer Science and Engineering, Sabanci University, Istanbul, Turkey
2010	Ph. D. in Computer Science, University of California, San Diego, USA

PUBLICATIONS

B. Dost, V. Bafna, N. Bandeira. Identification and Quantification of Post-translational Modification Variants and False Discovery Rates. *In preparation*.

B. Dost, N. Bandeira, X. Li, Z. Shen, S. Briggs, V. Bafna. Accurate Mass Spectrometry Based Protein Quantification via Shared Peptides. *Submitted*

B. Dost, A. Su, C. Wu, V.Bafna (2009). TCLUST: A fast algorithm for clustering genome-scale expression data. *In press. IEEE/ACM Transactions on Computational Biology and Bioinformatics*

B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, R. Sharan (2008). QNet: a tool for querying protein interaction networks. *Journal of Computational Biology*. 15(7):913-25.

B. Han, B. Dost, S. Zhang, V. Bafna (2008). Structural alignment of pseudoknotted RNA. *Journal of Computational Biology*. 15(5): 489-504.

B. Dost, N. Bandeira, X. Li, Z. Shen, S. Briggs, V. Bafna (2009). Shared peptides in mass spectrometry based protein quantification. RECOMB 2009, also published in Lecture Notes in Bioinformatics, 5541, p. 356-371.

B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, R. Sharan (2007). QNet: a tool for querying protein interaction networks. RECOMB 2007, also published in Lecture Notes in Bioinformatics, 4453, p. 1-15.

B. Dost, B. Han, S. Zhang and V. Bafna (2006). Structural alignment of pseudoknotted RNA. RECOMB 2006, also published in Lecture Notes in Bioinformatics, 3909, p. 143-158.

ABSTRACT OF THE DISSERTATION

Optimization Algorithms for Biological Data

by

Banu Dost
Doctor of Philosophy in Computer Science

University of California San Diego, 2010

Professor Vineet Bafna, Chair

High-throughput techniques in biology have enabled the generation of enormous amount of data allowing researchers to reveal systems level information deciphering the underlying dynamics and mechanisms of the cell. In the last few decades, the immense databases containing DNA, RNA and protein sequences, structures and abundance estimates have been available to researchers. Research in bioinformatics necessitates the use of advanced efficient algorithms to analyze and interpret those biological data. A common characteristic of high-throughput biological data is that it is often incomplete, noisy and inconsistent due to the biases and inefficiencies induced by the laboratory methods. That is why several of the problems defined on biological data can be viewed as constrained optimization problems.

In this dissertation, I address different optimization problems that arise in the analysis of biological data: RNA structural alignment, protein interaction network querying, micro-array expression data clustering, protein quantification and protein modification site assignment. The dissertation begins with an overview of the basic concepts of molecular biology and an introduction to the optimization problems to be addressed. Then, each problem is discussed in detail in a separate chapter along with our contribution in the solution of the problem and our results on biological data opening a way for biological discoveries.

Chapter 1

Introduction

Following the discovery of the DNA structure by Watson and Crick, many interesting advances have been made in the area of biology and biotechnology. In particular, high-throughput techniques in biology have enabled the generation of vast amount of data allowing researchers to reveal systems level information deciphering the underlying dynamics and mechanisms of the cell. The immense databases containing DNA, RNA and protein sequences and structures have been available to researchers. Bioinformatics has risen from the need of analyzing and interpreting those biological data which would be impossible with the traditional biological experiments and it has become an essential counterpart to modern biology by enabling important biological discoveries.

The main goal of bioinformatics is to develop methods using algorithms, statistics and many more mathematical techniques to utilize and interpret biological data. A common characteristic of high-throughput biological data is that it is often incomplete, noisy and inconsistent due to the biases and inefficiencies induced by the laboratory methods. That is why several of the problems defined on biological data are constrained optimization problems. Sequence alignment of bio-polymers such as DNA, RNA and protein, determining RNA/protein structure, biological network comparisons, protein identification and quantification can all be viewed as optimization problems. The solutions to those problems usually need to be computationally efficient and smart in the way they reason with the incompleteness and noise in the data.

In this dissertation, I address five different optimization problems that arise in the analysis of biological data (See Figure 1.1). It should be remarked that this number of problems is minuscule considering all of the possible optimization problems of bioinformatics. However, the approaches used to address optimization problems in bioinformatics are quite similar in nature. This allows researchers to adopt developed efficient algorithms to solve many different problems.

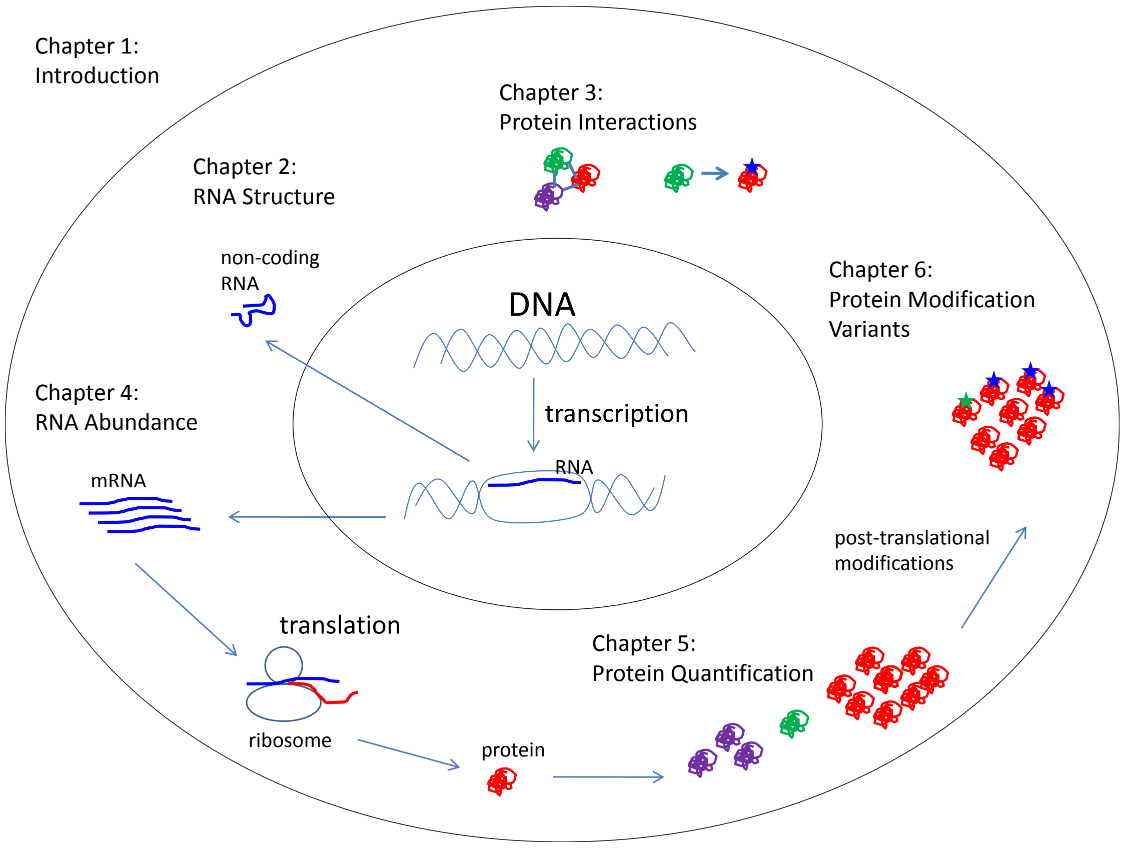


Figure 1.1: Overview of the dissertation. Five different optimization problems that arise from different facets of bioinformatics are discussed in this dissertation: RNA structural alignment, protein interaction network querying, clustering genome-scale expression data, protein identification and quantification and protein modification site assignment.

In the subsequent sections, I first provide a very high level biological background necessary to understand the optimization problems addressed in this dissertation. Then, for each problem, I explain the basic biological concepts specific to the problem, describe the nature of the existing biological data and underline the computational challenges and our contribution in the area that would be discussed in detail in the following chapters.

1.1 Brief Description of A Cell

In this section, I give a minimal set of biological facts needed to understand the material in the subsequent sections. This section is necessarily coarse. I refer the reader to the book ”Cell Biology” [115] for a detailed explanation of the fundamentals of cell and molecular biology.

All living organisms are composed of one or more cells. Cells in different organisms or within the same organism vary significantly in structure, reproduction, and metabolism. However, they all share a few common characteristics. The cell is made up of molecular components, which can be viewed as 3D-structures of various shapes. There are three types of molecules that are essential for life: DNA, RNA and proteins.

DNA stores the genetic code describing how the cell works. DNA is packaged as long molecules consisting of four types of nitrogenous bases called adenosine (A), thymine (T), guanine (G), and cytosine (C). It can be thought as a pair of complementary strings that are written in a four-letter alphabet A, T, G, C and twisted together such that each letter on one string pairs up with a letter on the other string in a certain way. The base A always pairs with the T, and the C always pairs with the G. The order and composition of these bases is what makes the DNA and therefore the organism ‘unique’. During cell division, DNA replicates itself by using each strand as a template to build its complementary strand. This way, an exact copy of DNA is passed on to new cells. DNA strings are extraordinarily long strings consisting of thousands of genes. Each gene is a short region of DNA storing the code necessary to determine the protein composition.

RNA is central in the flow of information from genes to the place where proteins are synthesized. When proteins are needed in the cell, the corresponding genes are transcribed into RNA (transcription). RNA molecule is a single strand consisting of A,T,G, C. A copying enzyme, RNA polymerase, copies the letters on a DNA strand to build the RNA molecule. After the completed messenger RNA (mRNA) is pre-processed so that its non-coding regions are removed, it is transferred to a cellular compartment called ribosome. The ribosome synthesizes a protein by using the messenger RNA as a template (translation). (See Figure 1.1.)

Proteins perform important tasks for the cell functioning such as performing biochemical reactions, signalling to other cells and serving as building blocks. A protein is also a long molecule like DNA and RNA, however it is built from a twenty-letter alphabet. These letters are called amino-acids. Each of the many different kinds of protein has its own unique order of amino-acids. There are always plenty of these amino acids floating around in the cytoplasm. The transfer RNAs bring amino-acids to the ribosome where they are attached to each other in the right order determined by the letters making up the messenger RNA.

1.2 RNA Structural Alignment

Not all RNA molecules are translated into a protein as described above. The DNA sequence from which a non-coding RNA is transcribed as the end product is often called an RNA gene or non-coding RNA gene (ncRNA). Non-coding RNA molecules include highly abundant RNAs such as transfer RNA (tRNA), ribosomal RNA (rRNA), and siRNAs performing essential functions in the cell. Discoveries of novel ncRNA families [3, 19, 23, 25, 28] points to the possibility that RNA molecules are as abundant, and diverse as protein molecules [10].

RNA molecules fold back onto themselves forming three dimensional complex shapes called secondary structure. Secondary structure is formally defined by the hydrogen bonding between the nitrogenous bases. The remarkable functional properties of ncRNAs lie in their capacity to fold into many shapes. For many RNA molecules, the secondary structure is highly important for the correct function of the RNA often more than its actual sequence.

High-throughput techniques in biology have generated vast amount of not only sequence information of thousands of biological molecules but also structural information. For instance, RFam, a ncRNA database, includes sequence and secondary structures of 379 ncRNA families. Due to the availability of sequence and structural information, biological sequence and structural comparison has been at the heart of research for a few decades. Many efficient tools such as BLAST , CLUSTAL , MFOLD, and RNAfold, have been developed to compare sequences [101, 102] and structures [122, 124]. These tools are used to identify conserved subsequences, substructures and motifs, which convey functional, structural, and evolutionary information. Pairwise and multiple sequence and structural alignment tools are very useful in designing experiments to test and modify the function of specific RNAs, in predicting the function and structure, and in identifying new members of RNA families. For detailed discussion of structural alignment of RNA sequences, previous work and challenges, see Chapter 2.

In Dost et al. [121], we provide an efficient algorithm for computing an optimum structural alignment of an RNA sequence with complex structures - which have not been handled before - against a genome. We also use our implemented tool PAL to search entire viral genome and mouse genome for novel homologs of some viral, and eukaryotic pseudoknotted RNAs respectively. In each case, we demonstrate strong support for novel homologs. In Chapter 2, the problem of discovering novel ncRNAs using sequence and secondary structure conservation is revisited and the details of our work on this problem are given.

1.3 Querying Protein-Protein Interaction Networks

In a cell, the molecules interact with each other. The functioning of the majority of molecules within cellular structures depends on their highly specific interactions with other molecules. By interaction it is meant that two or more molecules are combined to form one or more new molecules, that is, new 3D-structures with new shapes. An interaction may also reflect mutual influence among molecules. These interactions are due to attractions and repulsions that take place at the atomic level. For instance, if two or more proteins have surface regions that are complementary, they may stick to each other forming a protein complex. Proteins may also interact by chemically modifying each other by attaching (phosphorylation) or removing (dephosphorylation) small phosphate groups at specific sites. Many processes in the cell occur as a series of such protein protein interactions. Proteins are known to work in slightly overlapping, co-regulated functional modules such as pathways and complexes.

By the recent developments in molecular biology, a new kind of experimental data, relationships and interactions between molecules, has been generated. Biological networks that abstract those data convey information to understand several mechanisms in the cell, such as signaling pathways, transcriptional regulation mechanisms and metabolic processes. High-throughput methods such as two-hybrid analysis [103, 104] and genome-wide chromatin-immunoprecipitation [105, 108] provide large amounts of data on the protein protein interactions (interactome) of an increasing number of species.

It has been revealed that protein protein interaction networks evolve at a modular level [109] as biological sequences, motivating the study of conserved sub-network modules through comparative analysis. Consequently, the study of biological networks shows a significant parallelism with biological sequence comparison. Biological network comparison is the process of contrasting and comparing two or more interaction networks, representing different species, conditions, or time points. This process aims to detect conserved sub-network modules that are likely to reflect functional or computational units which combine to regulate the cellular behavior as a whole. The computational problem of biological network comparison in the form of alignment and querying that captures the underlying biological phenomena has been an interest to researchers.

Querying biological subnetworks in a large protein interaction network is a challenging computational problem, and recent efforts have been limited to simple queries. In Dost et al. [119], we extend the class of subnetworks that can be efficiently queried to the case of trees, and graphs of bounded tree-width. We also use our implemented tool QNet to perform the first large scale cross-species comparison of protein complexes, by querying known yeast complexes against a fly protein interaction network. This comparison points to strong conservation between the two species, and underscores the importance of our tool in mining protein interaction networks. This study is presented in Chapter 3 with a detailed discussion of network querying problem and previous efforts.

1.4 Clustering Expression Data

The different cell types of a multicellular organism contain the same DNA, however they differ dramatically in both structure and function. That cell differentiation generally depends on changes in gene expression, production of mRNA by a given gene, rather than the changes in the sequence of the genome. The cell types in a multicellular organism become different from one another because they synthesize different sets of proteins at different abundance levels. It has been reported that a typical human cell, at any one time, expresses approximately 10K-20K of its approximately 30K genes. When the patterns of mRNAs in a series of different human cell lines are compared, it is found that the level of expression of almost every active gene varies. Moreover, cells can change pattern of mRNA expression in response to environmental changes. The patterns of mRNA abundance are characteristic of cell type and/or condition that they can be used to type diseased cells (e.g. cancer) and determine the genes that are involved in certain functions. Thus, comparing mRNA expression levels across cell types and conditions has been a great interest to researchers.

Developed in 1990s, DNA microarrays has allowed researchers to study mRNA levels in large scale. Using microarrays, mRNA levels of tens of thousands of genes across hundreds of conditions can be measured simultaneously. As we previously mentioned, this amount corresponds to gene expression and presumably the amount of mRNA generated by the various genes establishes an estimate for the corresponding protein levels. Genes with a common function are often hypothesized to have correlated expression levels across different conditions. Thus, clustering algorithms have been intensively studied for analyzing gene expression data in order to detect the groups of genes with correlated expression patterns under different conditions or at different time points. Among the most popular algorithms for clustering gene expression data are K-means, SOM, hierarchical clustering, and CAST [54, 62, 69].

In clustering of genome-scale microarray expression data, it is important that the method is able to deal with its noisy nature of the data and that it is scalable to large datasets. In Chapter 4, the limitations of the current approaches are explained in detail and a novel clustering method TCLUST we have developed in Dost et al. [120] to efficiently cluster genome-scale expression data is presented along with the results on simulated data and genome-scale mouse gene expression data set.

1.5 Protein Identification and Quantification

Even though the differences in mRNA expression patterns among the different cell types are dramatic, they nonetheless underestimate abundance differences in the whole range of proteins in the cell. There are many steps after transcription of mRNA at which gene expression can be regulated. For instance, alternative splicing of mRNA can produce a whole family of proteins from a single gene by reorganizing the primary transcript to produce a different sequence. Alternative splicing can be thought as a mechanism that increases the possible proteins that are produced from a single gene. Especially in complex organisms like human, mRNA expression level might be very different from corresponding protein abundances. It has been estimated that at least 40-60% of all genes in the human genome are alternatively spliced and there are more than 120,000 protein slice variants from about 30,000 genes [123]. Therefore, a better way of detecting the differences in gene expression between cell types is through the use of proteomics technologies that directly measure protein levels.

In the last two decades, mass spectrometry (MS) has been the major technology used to identify and quantify proteins from a biological sample. Mass spectrometry allows one to determine the precise mass of each molecule in the sample along with their intensities which is a measurement for the abundance. The mass information can then be used to search databases, in which the masses of all possible molecules have been listed to identify the present molecules. Similarly, peak intensities can be used to estimate the relative abundances of different molecules within the sample. Mass spectrometric techniques are critically important for the large scale studies to identify and quantify all of the proteins present in biological samples.

It is often peptides derived from proteins by enzymatic or chemical cleavage rather than intact proteins that is being analyzed by mass spectrometers. MS analysis of peptides has several advantages over the analysis of proteins. First, MS analysis of peptides are, in general, more sensitive than analysis of proteins. Second, before the MS analysis contaminants (salts and detergents) need to be removed and the removal is easier for peptides than proteins [116].

In MS analysis, the identification and quantification of a peptide is used as a proxy for the parent protein. However, this holds only when the peptide sequence is unique to the protein. When a peptide is shared across proteins (Ex: proteins that share domains), its abundance depends upon contributions from multiple proteins. For this reason, shared peptides have been traditionally disregarded in protein-level identification and quantification analysis. However, in most of the mass spectrometric data sets, about half of the identified peptides are shared and thus discarded. This significantly decreases the number of proteins for which abundance estimates can be obtained.

In Dost et al. [118], we propose a novel optimization problem that uses shared peptides in protein identification and quantification. This study is the first attempt to use the ubiquitous shared peptides increasing the sensitivity in protein identification and accuracy in protein quantification. We validate our approach on simulated data and apply to a model of Arabidopsis nematode infection elucidating the differential role of many protein family members in mediating host response to the pathogen. In Chapter 5, more detailed information about the challenges and current approaches and their limitations on protein identification and quantification along with the details of our work in [118] are presented.

1.6 Protein Modification Site Assignment

As we have mentioned previously, mRNAs are subject to post-transcriptional events enabling the production of multiple protein sequences from a single single. Within the cell, the variety of protein sequences are further increased by post-translational modifications (PTMs) that change the properties of a protein by addition or removal of a modifying group to one or more amino-acid residues [93]. The structural diversification enabled by post-translational modification increases the molecular variants of proteins in cells by a few orders of magnitude over the number encoded in the genome [99]. If there are some 30,000 genes transcribed into RNAs and translated into proteins, there may be 300,000 to millions of protein variants at any one time in cells. These protein *modification variants* may differ in modification content and location at one or more amino acid residues within any given protein.

PTMs greatly impact the function of proteins by changing their activity state, localization, turnover, and interactions with other proteins. Identification of proteins with all their post-translational variants is crucial to biologists for understanding the mechanisms of cell regulation. Biological effects are often due to changes on the level of modification, therefore quantitative study of modifications is also of great interest [92, 93].

Mass spectrometry is also used to sequence of individual peptides obtained after digestion of the protein of interest. This method is particularly useful when proteins contain modifications through attached biochemical functional groups such as acetate, phosphate, various lipids and carbohydrates. The exact residue/amino-acid site where the modification occurs can be determined from sequence information obtained by tandem mass spectrometry (MS/MS). In tandem mass spectrometry, two mass spectrometers are required to run in tandem. The first one separates peptides by mass and outputs a spectrum of peak masses and intensities where each peak corresponds to a peptide. Then, one peptide at a time is picked for further analysis. This peptide is fragmented by collision with high energy gas atoms. This method of fragmentation beraks the peptide bonds and generates a ladder of fragments, each differing by a single amino-acid. The second mass spectrometer then separates the fragments and displays their masses and intensities as 'tandem mass spectrum'. The amino-acid sequence can be deduced from the differences in mass between peaks. Post-translational modifications are identified when the amino-acid to which they are attached show a diagnostic mass shift.

Accurate identification of large numbers of PTM sites by tandem mass spectrometry (MS/MS) remains a major challenge in proteomics. The simple approach described above to identify modification site works only when the tandem mass spectrum is obtained from a single peptide as intended. However, mass spectrometer separates molecules only by mass. When there are multiple modification variants of a protein/peptide with the same mass value in the sample, 'a mixture tandem spectrum' that is the overlay of tandem mass spectrum of multiple peptide sequences is acquired.

In Dost et al. [117], we propose a novel computational framework to accurately identify and quantify peptide modification variants from a mixture tandem mass spectrum and demonstrate our approach on both simulated and mass spectrometric data. The challenges and the limitations of current approaches for modification site assignment and details of this work are discussed in Chapter 6.

Chapter 2

Structural Alignment of Pseudoknotted RNA

Abstract

In this chapter, we address the problem of discovering novel non-coding RNA (ncRNA) using primary sequence, and secondary structure conservation, focusing on ncRNA families with pseudo-knotted structures. Our main technical result is an efficient algorithm for computing an optimum structural alignment of an RNA sequence against a genomic substring. This algorithm finds two applications. First, by scanning a genome, we can identify novel (homologous) pseudoknotted ncRNA, and second, we can infer the secondary structure of the target aligned sequence. We test an implementation of our algorithm (PAL), and show that it has near-perfect behavior for predicting the structure of many known pseudoknots. Additionally, it can detect the true homologs with high sensitivity and specificity in controlled tests. We also use PAL to search entire viral genome and mouse genome for novel homologs of some viral, and eukaryotic pseudoknots respectively. In each case, we have found strong support for novel homologs.

2.1 Introduction

Ribonucleic acid (RNA) is the third, and (until recently) most underrated of the trio of molecules that govern most cellular processes: the other two being proteins and DNA. While much of cellular RNA carries a message encoding an amino-acid sequence, other, ‘non-coding’ RNA participate directly in performing essential functions. Recent and unanticipated discoveries of novel ncRNA families [3, 19, 23, 25, 28] point to the possibility of a ‘Modern RNA world’ in which RNA molecules are as abundant, and diverse as protein molecules [10]. The analog of the computational gene-finding problem: “given genomic DNA, identify all substrings that encode ncRNA” is increasingly relevant, and relatively unexplored.

While potentially abundant, RNA signals are weaker than proteins making them harder to identify computationally. Possibly, the strongest clue is from secondary structure. Being single-stranded, the base-pairs stabilize by forming hydrogen bonds, leading to a characteristic secondary and tertiary structure. With a few exceptions, the base-pairs are *non-crossing*, and form a tree-like structure. This recursive structure is the basis for efficient algorithms to predict RNA structure [14,31]. With this extensive work in structure prediction, it is natural to expect that novel non-coding RNA could be discovered simply by looking for genomic sub-strings that fold into low-energy structures. Unfortunately, that idea doesn’t work. Rivas and Eddy (2000) showed that random DNA (usually with high GC-content) can also ’fold’ into low-energy configurations, making it unlikely for a purely *de novo* approach to be successful. Therefore, a comparative approach is employed, often typified by the question: ”Given a query RNA with known structure, and a genome, identify all genomic sub-strings that match the query sequence and structure”. The query itself can be either a single molecule or a model (covariance model/stochastic context free grammar) of an RNA structure. This approach has been quite successful and single queries as well as covariance based models are routinely used to annotate genomes with ncRNA [13, 16]. Central to these approaches is an algorithm for computing a local alignment between a query structure and a DNA string. The search itself is simply a scan of the genome to obtain all high scoring local alignments.

Here we pose a related question: *Given a query RNA with known structure, allowing for pseudoknots, and a genome, identify all genomic sub-strings that match the* *query sequence and structure*. Without being precise, pseudoknots are base-pairs that violate the non-crossing rule (See Figure 2.1). While not as common as other substructures (bulges,loops), they are often critically important to function. Pseudoknotted RNAs are known to be active as ribozymes [21], self-splicing introns [1], and participate in telomerase activity [24]. They have also been shown to alter gene expression by inducing ribosomal frame-shifting in many viruses [18]. However, understanding the extent and importance of these molecules is partially handicapped by the difficulty of identifying them (computationally). The algorithm presented here will facilitate identification.

In order to compute a local structural alignment, we must start with a formal definition of a pseudoknot in Section 3.2. Many definitions of pseudoknots have been postulated [2,8,11,15,22], and recent research investigates the power of these definitions in describing real pseudoknots [7]. We start here with Akutsu's formalism (*simple* pseudoknots) [2], which has a clean recursive structure and encompasses a majority of the known cases [7,20]. We also present algorithms that extend this class of allowed pseudoknots (standard pseudoknots). Section 2.3 describes the chaining procedure which is key to the alignment algorithm that follows (Section 2.4). However, the simple pseudoknots usually do not occur independently, but are embedded in regular RNA structures. In Section 2.5, we extend the algorithm to handle these cases. Other extensions are considered in Section 2.7. It has been brought to our attention that a recent publication [17] considers the identical problem using the formation of tree adjoining grammars to model pseudoknots. The pseudoknots considered by them are a restricted version of our simple pseudoknots. Furthermore, our alignment combines sequence and structural similarity.

The local alignments can be used in two ways. First, they can be used to infer the structure of the aligned substring that is conserved with the query. We show in Section 2.8.1 that in a majority of the cases, this leads to a perfect prediction of secondary (pseudoknotted) structure. Next, they can be used to predict novel ncRNA in genomic sequences. While our algorithms are computationally intensive, they can be used in combination with database filtering approaches to search large genomic regions. In Section 2.8.2, we validate our approach on real sequences embedded in random sequence. Finally, in Section 2.9, we identify (putative) novel pseudoknotted ncRNA in a search of viral and eukaryotic genomes.

2.2 Definitions and Preliminary Information

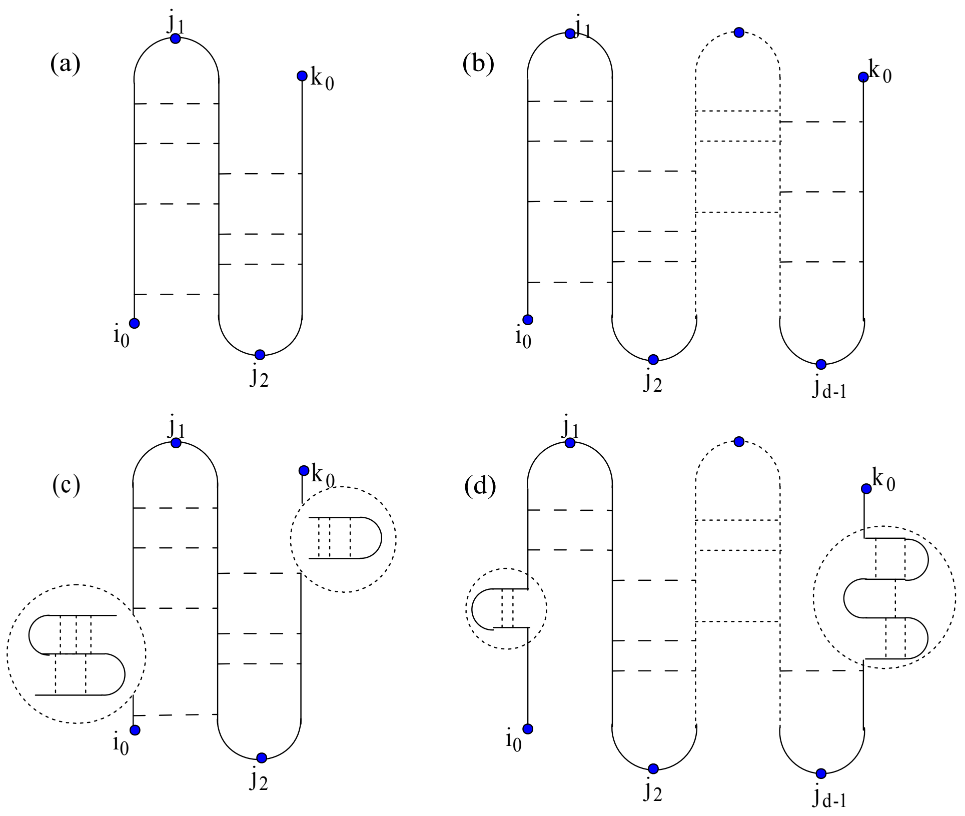


Figure 2.1: (a) Simple pseudoknot. (b) Standard pseudoknot of degree d. (c) Recursive simple pseudoknot. (d) Recursive standard pseudoknot of degree d.

Let $A = a_1 ... a_n$ be an RNA sequence. The secondary structure is represented simply as the set of base-pairs

$$M = \{(i, j) | 1 \leq i < j \leq n, (a_i, a_j) \text{ is a base pair}\}$$

Also, let $M_{i_0,k_0} \subseteq M$ be defined by $M_{i_0,k_0} = \{(i, j) \in M | i_0 \leq i < j \leq k_0\}$. The secondary structure, in the absence of crossing or interweaving base-pairs is called *regular*, and has the following recursive definition.

Definition 1: An RNA secondary structure M_{i_0,k_0} is regular if and only if $M_{i_0,k_0} = \phi$ or $\exists(i,j) \in M_{i_0,k_0}$ such that

- $M_{i_0,k_0} = M_{i_0,i-1} \cup M_{i+1,j-1} \cup M_{j+1,k_0} \cup \{(i,j)\}$ (No base-pairs cross the partitions).
- Each of $M_{i_0,i-1}, M_{i+1,j-1}, M_{j+1,k_0}$ is regular.

Next, we can define the class of allowed pseudoknots ([2]).

Definition 2: M_{i_0,k_0} is a simple-pseudoknot (see Figure 2.1a) if and only if M_{i_0,k_0} is regular or $\exists j_1, j_2 \in \mathbb{N}$ $(i_0 \leq j_1 < j_2 \leq k_0)$ such that the resulting partition, $D_1 = [i_0, j_1 - 1], D_2 = [j_1, j_2 - 1], D_3 = [j_2, k_0]$, satisfies the following:

- $M_{i_0,k_0} = (S_L \cup S_R)$, where $S_L = \{(i,j) \in M_{i_0,k_0} | i \in D_1, j \in D_2\}$ and $S_R = \{(i,j) \in M_{i_0,k_0} | i \in D_2, j \in D_3, \}$.
- S_L and S_R are regular.

Definition 3: M_{i_0,k_0} is a standard-pseudoknot with degree d ($d \geq 3$, see Figure 2.1b) if and only if M_{i_0,k_0} is regular or $\exists j_1, ..., j_{d-1} \in \mathbb{N}$ $(i_0 \leq j_1 < ... < j_{d-1} \leq k_0)$ which divide $[i_0, k_0]$ into d parts, $D_1 = [i_0, j_1 - 1], D_2 = [j_1, j_2 - 1], ..., D_d = [j_{d-1}, k_0]$, and satisfy the following:

- $M_{i_0,k_0} = \bigcup_{l=1}^{d-1} S_l$, where $S_l = \{(i,j) \in M_{i_0,k_0} | i \in D_l, j \in D_{l+1}\}$ for all $1 \leq l < d$.
- S_l is regular for all $1 \leq l < d$,

Note that a simple-pseudoknot is a standard-pseudoknot of degree 3.

Definition 4: M_{i_0,k_0} is recursive-standard-pseudoknot with degree d ($d \geq 3$, see Figure 2.1d) if and only if M_{i_0,k_0} is a standard pseudoknot of degree d or $\exists i_1, k_1, ..., i_t, k_t \in \mathbb{N}$ $(i_0 \leq i_1 < k_1 < i_2 < k_2 < ... < i_t < k_t \leq k_0, t \geq 1)$, which satisfy the following:

- $(M_{i_0,k_0} - \bigcup_{l=1}^{t} M_{i_l,k_l})$ is a standard pseudoknot of degree $\leq d$.
- $M_{i_l,k_l} (1 \leq l \leq t)$ is a recursive standard pseudoknot of degree $\leq d$.

A recursive-simple-pseudoknot is a recursive-standard-pseudoknot of degree 3 (Figure 2.1c). While we can devise algorithms to align recursive-standard-pseudoknots, they are computationally expensive, and most known families have a simpler structure. Therefore, we will limit our description and tests to a simpler structure (with a single level of recursion), defined as follows:

Definition 5: M_{i_0,k_0} is embedded-simple-pseudoknot if and only if $\exists i_1, k_1, ..., i_t, k_t \in \mathbb{N}$ $(i_0 \leq i_1 < k_1 < i_2 < k_2 < ... < i_t < k_t \leq k_0, t \geq 1)$, which satisfy the following:

- $(M_{i_0,k_0} - \bigcup_{l=1}^{t} M_{i_l,k_l})$ is regular.
- $M_{i_l,k_l}(1 \leq l \leq t)$ is a simple-pseudoknot.

2.2.1 Structural Alignment Preliminaries

For alignment purposes, we do not distinguish between RNA and DNA, as every substring in the genome might encode an RNA string. Let $q[1 \cdots m]$ and $t[1 \cdots n]$ be two RNA strings over the alphabet $\sum = \{A, C, G, U\}$ where q has a known structure M. An alignment of q and t is defined by a 2-rowd matrix A, in which row 1 (respectively, 2) contains q (respectively, t) interspersed with spaces, and for all columns j, $A[1, j] \neq' -'$ or $A[2, j] \neq' -'$. For $r \in \{1, 2\}$, define $\iota_r[i] = i - |\{l < i \text{ s.t. } A[r, l] =' -'\}|$. In other words, if $A[1, i] \neq' -'$, it contains the symbol $q[\iota_1[i]]$. The score of alignment A is given by

$$\sum_j \gamma(A[1, j], A[2, j]) + \sum_{i,j s.t. (\iota_1[i],\iota_1[j]) \in M} \delta(\iota_1[i], \iota_1[j], \iota_2[i], \iota_2[j])$$

The function γ scores for sequence similarity, while δ scores for conservation of structure. While this formulation encodes a linear gap penalty, we note here that alignments of RNA molecules may contain large gaps, particularly in the loop regions, and we implement affine penalties for gaps (details omitted). Naturally, we wish to compute alignments with the maximum score.

The key ideas are as follows: First, note that regular and pseudoknotted structures have a recursive formulation. Therefore, the problem of structurally aligning an RNA structure against a subsequence, can be decomposed into the problems of (recursively) aligning its sub-structures against the appropriate sub-sequences, and combining the results. For regular-structures, the structure is tree-like, and the recursion follows the nodes of the tree. For simple-pseudoknots, the structure is more complex, and will be described in Section 2.4. The structure for embedded-simple-pseudoknots is simply a combination of the two (See Section 2.7).

However, it is not sufficient to consider structural elements alone, as we wish to score for sequence conservation as well. The recursive structure described only contains a subset of the nucleotides that participate in structure. Therefore, we employ a second trick of introducing spurious structural elements (base-pairs) to M. The augmented structure M' must have the following properties:

- Each nucleotide i appears in M'.
- $|M'| = O(m)$, so that the size of the structure does not increase too much.
- The recursive structure of M is maintained.

Pseudoknots and regular structures have very different recursive structure, and require different augmentation procedures. In Section 2.3, we present *chaining*, a novel augmentation procedure for simple pseudoknots. An augmentation for regular structures, *binarization* was presented in [4], and is implicit in the covariance models used to align regular RNA [9]. Here, we extend binarization to include chaining for embedded-simple-pseudoknots (Figure 2.6). These augmentations are used in the alignment algorithms for simple-pseudoknots (Section 2.4), embedded-simple-pseudoknots (Section 2.5), and standard-pseudoknots (Section 2.7).

2.3 Chaining

Before describing the chaining procedure, we revisit the problem of aligning a simple pseudoknot to a genomic sub-string. Unlike regular structures, we cannot partition the genome into contiguous substrings, because of interweaving base pairs. Thus, we need a new substructure for simple pseudoknot structures. We start by defining a total ordering among the base pairs of a simple pseudoknot. Recall (Definition 3) that a simple-pseudoknot structure M_{i_0,k_0} can be divided into 3 parts: $D_1 = [i_0, j_0 - 1]$, $D_2 =$

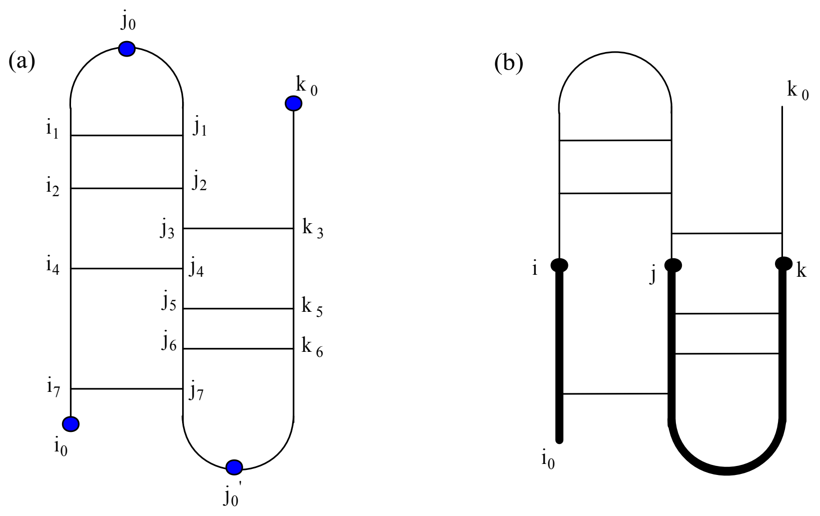


Figure 2.2: (a) Base pairs in a simple pseudoknot are ordered according to the index of the endpoint along $[j_0, j_0']$. Therefore, $(i_1, j_1) > (i_2, j_2) > (j_3, k_3) > (i_4, j_4) > (j_5, k_5) > (j_6, k_6) > (i_7, j_7)$. (b) Subpseudoknot structure.

$[j_0, j_0' - 1], D_3 = [j_0', k_0]$. (See Figure 2.2a) For each base pair $(i, j) \in M$, exactly one of i and j is in D_2 part. We define an ordering of the base pairs in M by sorting the coordinate in D_2. Formally, define $D_2(i, j)$ for all $(i, j) \in M$ as follows: $D_2(i, j) = i$ if $(i, j) \in S_R$, and $D_2(i, j) = j$ otherwise. For each $(i, j), (i', j') \in M$,

$$(i, j) \geq_p (i', j') \text{ iff } D_2(i, j) \geq D_2(i', j')$$

. As distinct base-pairs do not share any coordinates, $\geq_p$ defines a total ordering on the actual base-pairs, and can be used to define a partial order on substructures that we can recurse on. Define a *subpseudoknot* $\mathcal{P}(i, j, k)$ as the union of two subintervals $\mathcal{P}(i, j, k) = [i_0, i] \cup [j, k]$ (Figure 2.2b). Denote the triple (i, j, k) as the *frontier* for $\mathcal{P}(i, j, k)$. Note that i_0 is implicit from the context. Suppose that we are aligning frontier (i', j', k') of the query against frontier (i, j, k) of the target, with the score represented by $B[i, j, k, i', j', k']$. A naive algorithm would need to consider $O(m^3 n^3)$ pairs of frontiers. We improve this as follows: consider the special case of $(i', j') \in M$ where $(i', j') \in S_L$. The following recursion gives the score for B (proof omitted):

Theorem 1

$$B[i, j, k, i', j', k'] = \max\{\text{ MATCH,INSERT,DELETE}\}$$

$$\begin{aligned}
\text{MATCH} &= B[i-1, j+1, k, i'-1, j'+1, k'] + \delta(q[i'], q[j'], t[i], t[j]) \\
&\quad + \gamma(q[i'], t[i]) + \gamma(q[j'], t[j]), \\
\text{DELETE} &= \max \begin{cases} B[i-1, j, k, i'-1, j'+1, k'] + \gamma(q[i'], t[i]) + \gamma(q[j'], '-'), \\ B[i, j+1, k, i'-1, j'+1, k'] + \gamma(q[i'], '-') + \gamma(q[j'], t[j]), \\ B[i, j, k, i'-1, j'+1, k'] + \gamma(q[i'], '-') + \gamma(q[j'], '-') \end{cases} \\
\text{INSERT} &= \max \begin{cases} B[i-1, j, k, i', j', k'] + \gamma('-', t[i]), \\ B[i, j+1, k, i', j', k'] + \gamma('-', t[j]), \\ B[i, j, k-1, i', j', k'] + \gamma('-', t[k]) \end{cases}
\end{aligned}$$

Note that in every sub-case of MATCH and DELETE, we move from the query frontier (i', j', k') to the frontier $(i'-1, j'+1, k)$, because if either i' or j' is not used, we cannot score for the pair (i', j'). In the INSERT case, we stay at the frontier (i', j', k'). The situation is symmetric when $(j', k') \in S_R \subseteq M$, but is not defined when $(i', j') \notin M \wedge (j', k') \notin M$. The key idea for the chaining procedure is that we can define a unique frontier to move to in all cases, and still ensure that each nucleotide is touched by at least one frontier. By starting with a fixed frontier, and always moving to a fixed child, we only have $O(m)$ frontiers to consider.

From Definition 2, there exist indices j_1, j_2 which divide the simple pseudoknot structure into D_1, D_2 and D_3. We choose $(j_1 - 1, j_1, k_0)$ as the *root* frontier. Note that $\mathcal{P}(j_1 - 1, j_1, k_0)$ represents the entire simple-pseudoknot (See Figure 2.3a). We maintain the invariant that if (i, j, k) is a frontier and j participates in a base-pair, then the base-pair must be 'below' or within the frontier. In other words, if $(i', j) \in S_L$, then $i' \leq i$. Likewise, if $(j, k') \in S_R$, then $k' \leq k$. For a frontier (i, j, k), we have different cases: for example, if $(i', j) \in S_L$, we add spurious base pairs $(i, j), (i-1, j), \ldots (i', j)$. These base pairs define an ordered set of frontiers $(i, j, k) \geq (i-1, j, k) \geq \ldots, (i', j, k) \geq (i'-1, j+1, k)$. Likewise, if $(j, k') \in S_R$, we add spurious base-pairs $(j, k), (j, k-1), \ldots, (j, k')$, which define the frontiers $(i, j, k) \geq \ldots \geq (i, j+1, k'-1)$. The chaining algorithm, with a complete listing of cases is described in Figure 2.3. The output of chaining is a directed path of 'frontiers'. The number of nucleotides in a frontier (i, j, k) is given by the expression $((i - i_0 + 1) + (k - j + 1)) \leq m$. Further, this number decreases by at least 1 for each adjacent frontier. Thus the number of nodes in the chain is $O(m)$.

We still need to consider $O(n^3)$ target frontiers in aligning, for a complexity of $O(mn^3)$.

2.4 Alignment Algorithm for Simple Pseudoknots

Figure 2.4 describes the algorithm ALIGN-SP for aligning a simple-pseudoknot to a DNA substring. Its input is a chain of query sub-pseudoknots, which is aligned to all sub-pseudoknots $\mathcal{P}(i, j, k)$ of the target sequence $t[1 \dots n]$. Let M_L (respectively M_R) be the set of solid nodes representing subpseudoknots $\mathcal{P}(i, j, k)$ where $(i, j) \in S_L$ (respectively, $(j, k) \in S_R$). Let M_S be set of the nodes representing subpseudoknots $\mathcal{P}(i, j, k)$ where neither $(i, j) \notin S_L$, and $(j, k) \notin S_R$.

As an example, suppose we are aligning sub-pseudoknot $\mathcal{P}(i, j, k)$ in t to the subchain rooted at v. Let $B[i, j, k, v]$ be the score of the optimal alignment. First, we have cases involving insertion of target nucleotides: $t[i], t[j]$, and $t[k]$, as described by the recurrence in Figure 2.4 (Line 15). Next, we have the cases corresponding to match or deletion of v. We consider the case $v \in M_L$ corresponding to the subpseudoknot $\mathcal{P}(l_v, m_v, r_v)$ in q. The following cases can occur

1. $(t[i], t[j])$ is a pair in t corresponding to the pair $(q[l_v], q[m_v])$ in q.
2. $q[l_v]$ is substituted with $t[i]$ and $q[m_v]$ is deleted.
3. $q[m_v]$ is substituted with $t[j]$ and $q[l_v]$ is deleted.
4. $q[l_v]$ and $q[m_v]$ are both deleted.

The corresponding recurrences are shown on Line 6 of the procedure in Figure 2.4. The other cases are handled in an analogous fashion and are described in Figure 2.4.

```
CHAINING$(i, j, k)$
if $i = i_0 - 1$ and $j > k$
  then return NIL
if $(i, j) \in S$
  then $v = $ CHAINING$(i - 1, j + 1, k)$;
       return CREATENODE$(i, j, solid, move(1, 1, 0), v)$
if $(j, k) \in S$
  then $v = $ CHAINING$(i, j + 1, k - 1)$;
       return CREATENODE$(j, k, solid, move(0, 1, 1), v)$
if $j \in V_L$
  then $v = $ CHAINING$(i - 1, j, k)$;
       return CREATENODE$(i, j, empty, move(1, 0, 0), v)$
if $j \in V_R$
  then $v = $ CHAINING$(i, j, k - 1)$;
       return CREATENODE$(j, k, empty, move(0, 0, 1), v)$
if $i \in V_L$
  then $v = $ CHAINING$(i, j + 1, k)$;
       return CREATENODE$(i, j, empty, move(0, 1, 0), v)$
if $k \in V_R$
  then $v = $ CHAINING$(i, j + 1, k)$;
       return CREATENODE$(j, k, empty, move(0, 1, 0), v)$
if $i > i_0$
  then $v = $ CHAINING$(i - 1, j, k)$;
       return CREATENODE$(i, j, empty, move(1, 0, 0), v)$
if $i = i_0$
  then $v = $ CHAINING$(i - 1, j + 1, k)$;
       return CREATENODE$(i, j, empty, move(1, 1, 0), v)$
if $i = i_0 - 1$
  then $v = $ CHAINING$(i, j + 1, k)$;
       return CREATENODE$(j, k, empty, move(0, 1, 0), v)$
```

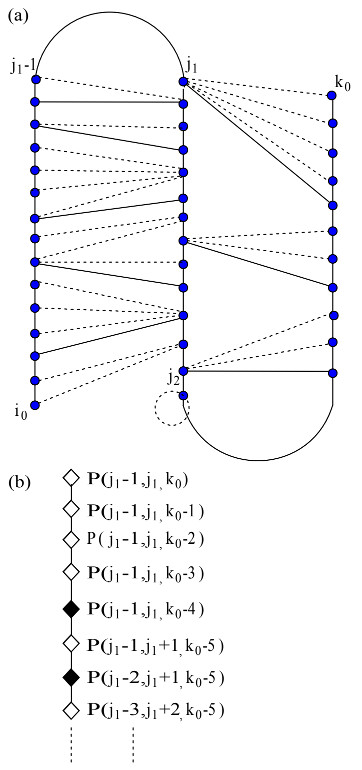


Figure 2.3: The chaining procedure on a simple pseudoknot structure M_{i_0,k_0}. (a) Solid base pairs are the actual base pairs, dotted ones are the spurious base pairs. (b) Chain structure representing the simple pseudoknot structure M_{i_0,k_0}. Solid nodes represents a sub-pseudoknot with frontier (i, j, k) where (i, j) or (j, k) is an actual base pair. Empty nodes represents a sub-pseudoknot with frontier (i, j, k) where neither (i, j) nor (j, k) is an actual base pair.

```
ALIGN-SP($M', t[1...n]$)
1    // $M'$ is the chain representing the simple pseudoknot region to be aligned in query $q$
2    for all intervals $(i_0, k_0)$ in $t[1...n]$
3    do for all $(i, j, k), i_0 \le i < j \le k \le k_0$
4        do for all nodes $v \in M'$
5            do if $v \in M_L$
6                then $B[i,j,k,v] = \max \begin{cases} B[i-1,j+1,k,child(v)] + \delta(q[l_v], q[m_v], t[i], t[j]) \\ \qquad +\gamma(q[l_v], t[i]) + \gamma(q[m_v], t[j]), \\ B[i-1,j,k,child(v)] + \gamma(q[l_v], t[i]) + \gamma(q[m_v], '-'), \\ B[i,j+1,k,child(v)] + \gamma(q[l_v], '-') + \gamma(q[m_v], t[j]), \\ B[i,j,k,child(v)] + \gamma(q[l_v], '-') + \gamma(q[m_v], '-') \end{cases}$
7                if $v \in M_R$
8                then $B[i,j,k,v] = \max \begin{cases} B[i,j+1,k-1,child(v)] + \delta(q[m_v], q[r_v], t[j], t[k]) \\ \qquad +\gamma(q[m_v], t[j]) + \gamma(q[r_v], t[k]), \\ B[i,j,k-1,child(v)] + \gamma(q[m_v], '-') + \gamma(q[r_v], t[k]), \\ B[i,j+1,k,child(v)] + \gamma(q[m_v], t[j]) + \gamma(q[r_v], '-'), \\ B[i,j,k,child(v)] + \gamma(q[m_v], '-') + \gamma(q[r_v], '-') \end{cases}$
9                if $v \in M_S$ and $move(v) = (1,0,0)$
10               then $B[i,j,k,v] = \max \begin{cases} B[i-1,j,k,child(v)] + \gamma(q[l_v], t[i]), \\ B[i,j,k,child(v)] + \gamma(q[l_v], '-') \end{cases}$
11               if $v \in M_S$ and $move(v) = (0,0,1)$
12               then $B[i,j,k,v] = \max \begin{cases} B[i,j,k-1,child(v)] + \gamma(q[r_v], t[k]), \\ B[i,j,k,child(v)] + \gamma(q[r_v], '-') \end{cases}$
13               if $v \in M_S$ and $move(v) = (0,1,0)$
14               then $B[i,j,k,v] = \max \begin{cases} B[i,j+1,k,child(v)] + \gamma(q[m_v], t[k]), \\ B[i,j,k,child(v)] + \gamma(q[m_v], '-') \end{cases}$
15               $B[i,j,k,v] = \max \begin{cases} B[i,j,k,v] \\ B[i-1,j,k,v] + \gamma('-', t[i]), \\ B[i,j+1,k,v] + \gamma('-', t[j]), \\ B[i,j,k-1,v] + \gamma('-', t[k]) \end{cases}$
16
17       $B_{SP}[i_0, k_0, i_{SP}, k_{SP}] = \max_{j=i+1,k=k_0}\{B(i,j,k,\text{ROOT}(M'))\}$
```

Figure 2.4: Align-SP procedure for alignment of a simple pseudoknot structure to a target sequence $t[1...n]$.

```
IMPROVED ALIGN-SP()
1  for all $v \in M'$
2  do for $i_0 = 1$ to $n - 1$
3      do for $i = i_0 - 1$ to $n - 1$
4          do for $j = n + 1$ downto $i + 1$
5              do for $k = j - 1$ to $n$
6                  do Compute $B[i, j, k, v]$
```

Figure 2.5: Improved Align-SP procedure.

2.5 Alignment Algorithm for Embedded Simple Pseudoknots

We consider now the special case of aligning recursive-simple-pseudoknots in which simple-pseudoknots are embedded in a regular structure. This is by far the most common occurrence of pseudoknots. While it is relatively easy to extend our algorithms to handle the full generality of recursive-pseudoknots, the complexity increase makes the algorithms untractable for real problems. Thus, this special case offers a compromise between generality and practicality.

The first step in the procedure is to *binarize* the query RNA, so that every nucleotide is in a base-pair, and can be represented by a binary tree of size $O(m)$ [4]. The main difference is that we invoke the chaining procedure whenever a simple-pseudoknot is encountered. Thus, in the binary tree, the simple pseudoknot substructure appears as a chain rooted at a *pseudo-node* (See Figure 2.6).

After the binary tree structure M' of query sequence q is created, target sequence t is aligned to this tree. The following procedure ALIGN aligns a given subsequence $(t[i \ldots j])$ in target sequence to a subtree of M'. The scores of optimal alignments are stored in matrix A. The entry $A[i, j, v]$ keeps the optimal alignment of the subproblem of aligning a subsequence $(t[i], t[j])$ to the subtree rooted at the node v, in other words to the subinterval $(q[l_v], q[r_v])$ of the query sequence.

```
BINARIZE-SP$(i, j)$
 1  if $(i, j)$ is a simple pseudoknot structure
 2    then return CHAINING$(i, j, pseudo - node, Nil)$;
 3  if $i = j$
 4    then return CREATENODE$(i, j, empty, Nil)$;
 5  if $(i, j) \in M$
 6    then $v =$ BINARIZE-SP$(i + 1, j - 1)$;
 7         return CREATENODE$(i, j, solid, v)$;
 8  if $(k, j) \in M$ for some $i < k < j$
 9    then
10         $vl =$ BINARIZE-SP$(i, k - 1)$;
11         $vr =$ BINARIZE-SP$(k, j)$;
12         (A empty node with 2 children, vl and vr.)
13         return CREATENODE$(i, j, empty, vl, vr)$;
14  if $i < j$
15    then $v =$ BINARIZE-SP$(i, j - 1)$;
16         return CREATENODE$(i, j, empty, v)$;
```

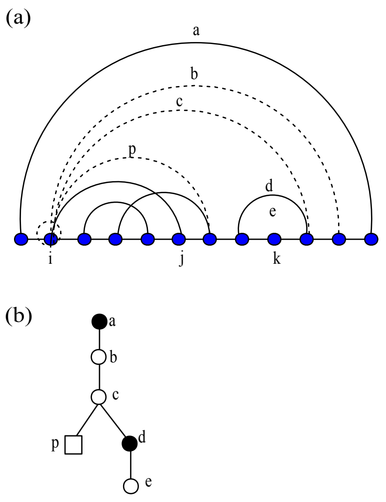


Figure 2.6: Binarization procedure revised for embedded-simple-pseudoknots and an illustration. (a) An embedded-simple-pseudoknot with spurious base pairs added. (b) Resulting binary tree. Solid nodes correspond to actual base pairs while empty (circular) nodes correspond to spurious base pairs. A '□' represents a pseudonode and subtree rooted at a pseudonode is formed by Chaining procedure.

2.6 Complexity

In Align-SP, lines $3-15$ runs in $O(n^3)$ time to align all subpseudoknots in target to a node. Those lines are executed for each subinterval (i_0, k_0) in target and for each node in the query tree. Then, time complexity of procedure Align-SP becomes $O(mn^5)$. However, we do not need to compute $O(n^3)$ scores for each subinterval (i_0, k_0). Since k_0 does not appear in the recurrences of Align-SP procedure and $B[i, j, k]$ does not depend on $B[i', j', k']$ such that $k' > k$, $B[i, j, k]$ does not depend on k_0. Thus, it is enough to compute $O(n^3)$ scores for each i_0 as shown in Figure 2.5. Then, total running time of Align-SP is $O(mn^4)$.

In Align procedure in Figure 2.7, we first call Binarization-SP procedure which runs in $O(m)$ time. We also call Align-SP procedure whenever we encounter with a pseudo-node in the binary tree formed. Let m_p be the length of the pseudoknot regions in $q[1 \cdots m]$, m_1 and m_2 be the number of the nodes with one child and two children in the binary tree of q representing the regions with regular structure. Then, the total running time of Align procedure will be $O(m_p n^4 + m_1 n^2 + m_2 n^3)$. It is useful to note that very often, $m_p, m_2 \in o(m)$, and so the true complexity is better than the worst case complexity. Also, in computing good alignments, we can often bound the gap-lengths. To take advantage of this, we employ a banding procedure (details not shown).

```
ALIGN($q[1...m], t[1...n]$)
1   $M' = $ BINARIZE-SP(Q)
2   for all intervals $(i, j)$ in $t$ and all nodes $v$ in $M$
3   do if $v$ is NIL
4          then $A[i, j, NIL] = \sum_{l=i}^{j} \gamma(t[l], '-')$
5      if $v$ is a pseudo node
6          then $A[i, j, v] = $ return ALIGN-SP$(i, j, v)$
7      if $v \in M$
8          then $A[i,j,v] = \max \begin{cases} A[i+1, j-1, \text{child}(v)] + \delta(t[i], t[j], q[lv], q[rv]) \\ A[i, j-1, v] + \gamma('-', t[j]) \\ A[i+1, j, v] + \gamma('-', t[i]) \\ A[i+1, j, \text{child}(v) + \gamma(q[lv], t[i]) + \gamma(q[rv], '-') \\ A[i, j-1, \text{child}(v)] + \gamma(q[lv], '-') + \gamma(q[rv], t[j]) \\ A[i, j, v] + \gamma(q[lv], '-') + \gamma(q[rv], '-') \end{cases}$
9      if $v \in M' - M$ and v has one child
10         then $A[i,j,v] = \max \begin{cases} A[i, j-1, \text{child}(v)] + \gamma(q[rv], t[j]) \\ A[i, j, \text{child}(v)] + \gamma(q[rv], '-') \\ A[i, j-1, v] + \gamma('-', t[j]) \\ A[i+1, j, v] + \gamma('-', t[i]) \end{cases}$
11     if $v \in M' - M$ and v has two children
12         then $A[i, j, v] = \max_{i \le k \le j} \{A[i, k-1, \text{left\_child}(v)] + A[k, j, \text{right\_child}(v)]\}$
```

Figure 2.7: Alignment Algorithm for aligning an embedded-simple-pseudoknot $q[1...m]$ to a target sequence $t[1...n]$.

2.7 Alignment Algorithm for Standard Pseudoknots

It is possible to extend the algorithm for aligning a simple pseudoknot to an alignment algorithm for a standard pseudoknot with degree $d > 3$. In this section, we present an extension of our algorithm for standard pseudoknot structures with degree 4, and achieve the following result:

Theorem 2 The optimal alignment for a standard pseudoknot with degree 4 can be computed in $O(mn^4)$ time which is identical to the degree 3 case (simple pseudoknots). In general, standard pseudoknots of degree $2k-1$ and $2k$ can be aligned in $O(mn^{2k})$ time.

To handle this kind of the pseudoknots, we firstly have to modify our substructure and chaining procedure accordingly. In a standard pseudoknot structure, the substructure will be union of three subintervals as shown in Figure 2.8 and each will be identified by a quadruple (i_v, j_v, k_v, l_v). We denote the substructure identified with quadruple (i_v, j_v, k_v, l_v) with $\mathcal{P}(i_v, j_v, k_v, l_v)$.

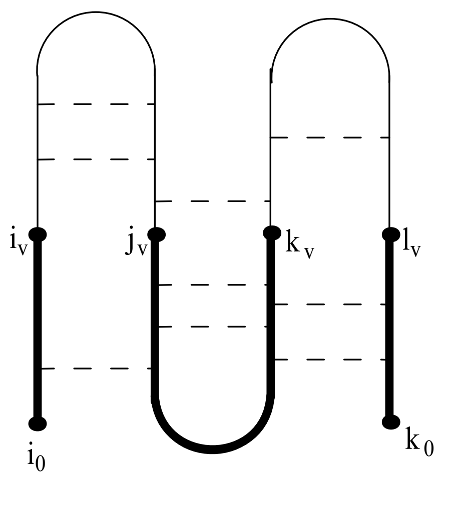


Figure 2.8: Substructure in a standard pseudoknot structure.

Chaining procedure again creates a chain of nodes where each corresponds to a substructure. There will be 3 different solid nodes corresponding to a quadruple (i_v, j_v, k_v, l_v). Let M_L,M_M and M_R be the sets of nodes in which $(q[i_v], q[j_v])$, $(q[j_v]$, $q[k_v])$ and $(q[k_v], q[l_v])$ is a pair, respectively and let M_S be the set of empty nodes.

Align-SPd4 procedure, as shown in Figure 2.9, is very similar to the one for simple pseudoknot structure which is also a standard pseudoknot but with degree 3. Now, we need to keep 4 indices i, j, k, and l along the target sequence. In the recursion, we slide the indices i, j, k and l down along the regions D_1, D_2,D_3, and D_4 respectively. To align a node $v \in M_L$ representing substructure $\mathcal{P}(i_v, j_v, k_v, l_v)$ in query to substructure $\mathcal{P}'(i, j, k, l)$ in target, it is enough to consider the same 7 cases we did for simple pseudoknot structure. However, we have to extend the algorithm to also account for the solid nodes in M_M and M_R. Cases for the empty nodes are also handled in the same as in alignment procedure for simple pseudoknot structures.

Similar to what we did for simple pseudoknots, we can control the indices i,j,k and l as shown in Figure 2.10 to improve running time. It turns out that aligning a standard pseudoknot with degree 4 has the same complexity with degree 3, i.e. $O(mn^4)$ where m is the length of the standard pseudoknot structure. Because, we limit the subinterval by only i_0 and the complexity of the for loop for a fixed i_0 is $O(n^3)$. Since the order of nodes for the tree of a standard pseudoknot of length m is $O(m)$, total complexity is $O(mn^4)$.

```
ALIGN-SPD4($M'$ in $M$ of $q[1...m], t[1...n]$)
1    // $M'$ is the chain representing the standard pseudoknot region to be aligned in query $q$
2   for all intervals $(i_0, k_0)$ in $t[1...n]$
3   do for all $(i,j,k,l), i_0 \le i \le j < k \le l \le k_0,$ all nodes $v \in M'$
4       do
5           if $v \in M_L$
6             then $B[i,j,k,l,v] = \max \begin{cases} B[i+1,j-1,k,l,child(v)] + \delta(q[i_v],q[j_v],t[i],t[j]) \\ \qquad +\gamma(q[i_v],t[i]) + \gamma(q[j_v],t[j]), \\ B[i+1,j,k,l,child(v)] + \gamma(q[i_v],t[i]) + \gamma(q[j_v],'-'), \\ B[i,j-1,k,l,child(v)] + \gamma(q[i_v],'-') + \gamma(q[j_v],t[j]), \\ B[i,j,k,l,child(v)] + \gamma(q[i_v],'-') + \gamma(q[j_v],'-') \end{cases}$
7           if $v \in M_M$
8             then $B[i,j,k,l,v] = \max \begin{cases} B[i,j-1,k+1,l,child(v)] + \delta(q[j_v],q[k_v],t[j],t[k]) \\ \qquad +\gamma(q[j_v],t[j]) + \gamma(q[k_v],t[k]), \\ B[i,j,k+1,l,child(v)] + \gamma(q[j_v],'-') + \gamma(q[k_v],t[k]), \\ B[i,j-1,k,l,child(v)] + \gamma(q[j_v],t[j]) + \gamma(q[k_v],'-'), \\ B[i,j,k,l,child(v)] + \gamma(q[j_v],'-') + \gamma(q[k_v],'-') \end{cases}$
9           if $v \in M_R$
10            then $B[i,j,k,l,v] = \max \begin{cases} B[i,j,k+1,l-1,child(v)] + \delta(q[k_v],q[l_v],t[k],t[l]) \\ \qquad +\gamma(q[k_v],t[k]) + \gamma(q[l_v],t[l]), \\ B[i,j,k,l-1,child(v)] + \gamma(q[k_v],'-') + \gamma(q[l_v],t[l]), \\ B[i,j,k+1,l,child(v)] + \gamma(q[k_v],t[k]) + \gamma(q[l_v],'-'), \\ B[i,j,k,l,child(v)] + \gamma(q[k_v],'-') + \gamma(q[l_v],'-') \end{cases}$
11          if $v \in M_S$ and $move(v) = (1,0,0,0)$
12            then $B[i,j,k,l,v] = \max \begin{cases} B[i+1,j,k,l,child(v)] + \gamma(q[i_v],t[i]), \\ B[i,j,k,l,child(v)] + \gamma(q[i_v],'-') \end{cases}$
13          if $v \in M_S$ and $move(v) = (0,1,0,0)$
14            then $B[i,j,k,l,v] = \max \begin{cases} B[i,j-1,k,l,child(v)] + \gamma(q[j_v],t[j]), \\ B[i,j,k,l,child(v)] + \gamma(q[j_v],'-') \end{cases}$
15          if $v \in M_S$ and $move(v) = (0,0,1,0)$
16            then $B[i,j,k,l,v] = \max \begin{cases} B[i,j,k+1,l,child(v)] + \gamma(q[k_v],t[k]), \\ B[i,j,k,l,child(v)] + \gamma(q[k_v],'-') \end{cases}$
17          if $v \in M_S$ and $move(v) = (0,0,0,1)$
18            then $B[i,j,k,l,v] = \max \begin{cases} B[i,j,k,l-1,child(v)] + \gamma(q[l_v],t[l]), \\ B[i,j,k,l,child(v)] + \gamma(q[l_v],'-') \end{cases}$
19          $B[i,j,k,l,v] = \max\{B[i+1,j,k,l,v] + \gamma('-',t[i]), B[i,j-1,k,l,v] + \gamma('-',t[j]),$
20            $B[i,j,k+1,l,v] + \gamma('-',t[k]), B[i,j,k+1,l-1,v] + \gamma('-',t[l], B[i,j,k,l,v])\}$
21          $B_{SP}[i_0,k_0,i_{SP},k_{SP}] = \max_{i=i_0,k=j+1,l=k_0}\{B(i,j,k,l,\text{ROOT}(M'))\}$
```

Figure 2.9: Alignment procedure (Align-SPd4) for a standard pseudoknot with degree $d = 4$.

```
IMPROVED ALIGN-SPD4()
1 for all $v \in S'$
2 do for $i_0 = 1$ to $n-1$
3     do for $j = i_0$ to $n-2$
4         do for $i = j$ downto $1$
5             do for $k = n$ downto $j+1$
6                 do for $l = k$ to $n$
7                     do Compute $B[i,j,k,v]$
```

Figure 2.10: Improved Align-SPd4 procedure.

2.8 Results

A C++ implementation of the algorithm given for simple pseudoknots (PAL) is done. PAL takes an RNA query and target sequence, and returns all high scoring structural local alignments in the target sequence. All tests were performed on a PC (3.4 Ghz, 1 GB RAM) unless otherwise stated. The structure of the target sub-sequence is inferred from the alignment (Ex: Figure 2.12). In order to assess the performance of PAL, we tested 6 RNA families from Rfam database: UPSK, Antizyme, Parecho CRE, Corona-FSE, Corona-pk3 and IFN-gamma. Each of these families has an embedded-simple-pseudoknot structure. General information about these families are shown in Table 2.1.

Table 2.1: Six simple Pseudoknotted RNA families. Avg Id stands for the average sequence identity between two members, n_s for the number of seed members, n_f for the number of whole known family members, n_u for the number of non-redundant (unique) family members that conserve pseudoknot, L for the length, L_P for the length of the pseudoknot region and t for the average time PAL takes for the alignment of a pair.

RNA Family	**Rfam Id**	Avg Id	n_s	n_f	n_u	L	L_P	$t(sec)$
UPSK	RF00390	92.75%	4	25	4	$23-23$	~ 22	0.1
Antizyme	RF00381	81.74%	13	41	18	$57-59$	~ 54	5.9
Parecho CRE	RF00499	80.32%	5	5	5	$102-115$	~ 33	2.4
Corona-FSE	RF00507	64.91%	18	143	18	$79-85$	~ 76	31.5
Corona-pk3	RF00165	68.37%	14	76	17	$62-64$	~ 56	9.9
IFN-gamma	RF00259	78.47%	5	11	7	$166-169$	~ 113	124.0

2.8.1 Predicting Structure with PAL

To test structural inference, we select a pair of members from a family as the query and target. PAL is used to align the query to the target. The inferred structure of the target is compared against the annotated structure in the Rfam database. We evaluate the predicted structure by computing *TP* (true positives), *FP* (false positives) and *FN* (false negatives), defined as follows: *TP* is the number of base pairs in inferred target structure that are correct: *FP* is the number of base pairs in the inferred structure that are not in the true structure, and *FN* is number of base pairs in the true structure that are not inferred. We define *Specificity = TP / (TP + FP)* and *Sensitivity = TP / (TP + FN)*. Good performance is indicated by both being close to 1. Table 2.2 summarizes the result of testing each pair in the 6 families. As the results show, PAL is a strong predictor of structure, with mean sensitivity and specificity of 0.95. We also investigated the few cases in which the prediction was away from the mean. In most of those cases, the target had stem loops that were longer than the query. As they were not aligned to the query structure, they were not inferred. In practice, we would augment the inferred structure by a local extension of stem loops in both directions. A second source of errors was incorrect annotation in Rfam. Other than these two scenarios, the structure inference was essentially correct.

There is a second caveat in these results which is not apparent. Many (but not all) of the sequences have high sequence similarity, which might be making the alignment task easier. We believe this is because a sequence search tool like Blast is used to fish out candidates, which are then manually aligned, and experimentally validated. We will show in the following sections that our tool can pick out candidates that BLAST cannot find, and also align them structurally. Also, in the cases where there isn't high sequence similarity, the structure inference was just as good.

Table 2.2: Pairwise tests: Statistics for Specificity and Sensitivity values. Mean is the average of Specificity (Sensitivity) values and median is the mid-point of Specificity (Sensitivity) values over all pairs of non-redundant members in an RNA family.

	Specificity				**Sensitivity**			
RNA Family	Mean	StdDev	Median	Range	Mean	StdDev	Median	Range
UPSK	1.000	0.000	1.000	(1.000-1.000)	1.000	0.000	1.000	(1.000-1.000)
Antizyme	0.994	0.018	1.000	(0.941-1.000)	0.993	0.018	1.000	(0.941-1.000)
Parecho	0.943	0.052	0.969	(0.848-1.000)	0.939	0.060	0.969	(0.844-1.000)
Corona-FSE	0.921	0.126	1.000	(0.579-1.000)	0.920	0.126	1.000	(0.579-1.000)
Corona-pk3	0.933	0.178	1.000	(0.000-1.000)	0.931	0.178	1.000	(0.000-1.000)
IFN-gamma	0.943	0.083	0.982	(0.691-1.000)	0.940	0.083	0.982	(0.691-1.000)

2.8.2 Searching for Structural Homologs

In this test, we use one of the members of an RNA family as a query, and look for its homolog in a large random sequence, with the other members inserted. Figure 2.11 shows the results for the Corona-FSE family, in which 17 members were embedded in a 19kb random sequence. The windowed scores are shown by solid lines. The actual positions of the remaining 17 members are denoted by '+'. We note that the true hits are easily the highest scoring regions along the sequence, and that all true positives score higher than all the false hits. The lowest scoring TP has a score of 769 and the highest scoring FP has a score of 557.

We performed a random test to assess significance of these result. We generated a bulk (10^6) of random sequences of equal size with the query member of the RNA family, and determined the p-value of a score as the proportion of the sequences which exceeds the score. However, since we performed many (965) tests shifting windows in the previous test on 19kb sequence, there is high chance that we assessed the score by chance; we should correct multiple hypothesis. We use Bonferroni correction method to adjust p-value to a correct value. Note that Bonferroni correction is conservative here because windowed tests on near-by regions are correlated and the score we assess p-value is not always the best score chosen among all tests. (Bonferroni correction assumes that we select the best score among independent result.) We performed three experiments assuming different GC contents containment (25%, 50%, 75%) in random sequences (Table 2.3), and in all cases, the lowest TP shows significant p-value even after adjusted by conservative Bonferroni correction.

Table 2.3: P-value of the structural homolog search test, as the proportion of random tests which exceeds the score among 1 million random tests. Adjusted p-value is Bonferroni corrected value.

GC contents	Description	Score	# of tests exceeding the score	p-value (unadjusted)	p-value (adjusted)
25%	Lowest TP	769	3	3.0×10^{-6}	2.9×10^{-3}
	Highest FP	557	2912	2.9×10^{-3}	1.0
50%	Lowest TP	769	1	1.0×10^{-6}	9.6×10^{-4}
	Highest FP	557	633	6.3×10^{-4}	6.0×10^{-1}
75%	Lowest TP	769	0	1.0×10^{-6}	9.6×10^{-4}
	Highest FP	557	360	3.6×10^{-4}	3.4×10^{-1}

In contrast, Blastn (E-value 10, Word-size 7) is able to locate only 4 of the members. These results also show the significance of the secondary structure for searching homologue in addition to the primary structure. We repeat the same experiment for RNA families, UPSK, Antizyme, Parecho, Corona-FSE, Corona-pk3 and IFN-gamma. In all cases, PAL locates all members as the topmost hits (See Table 2.4). We agree that Blast is not the most appropriate tool for comparison as other tools such as RSEARCH, and our own tool FastR can search for structural homologs of RNA [16,30]. However, these other tools cannot align psuedoknotted RNA and the search must be followed up with a correct alignment to determine homologs. Also, the complexity of these methods often force a use of Blast to determine initial candidates. In the next section, we show that our tool used in conjunction with RNA filters can efficiently search large genomes.

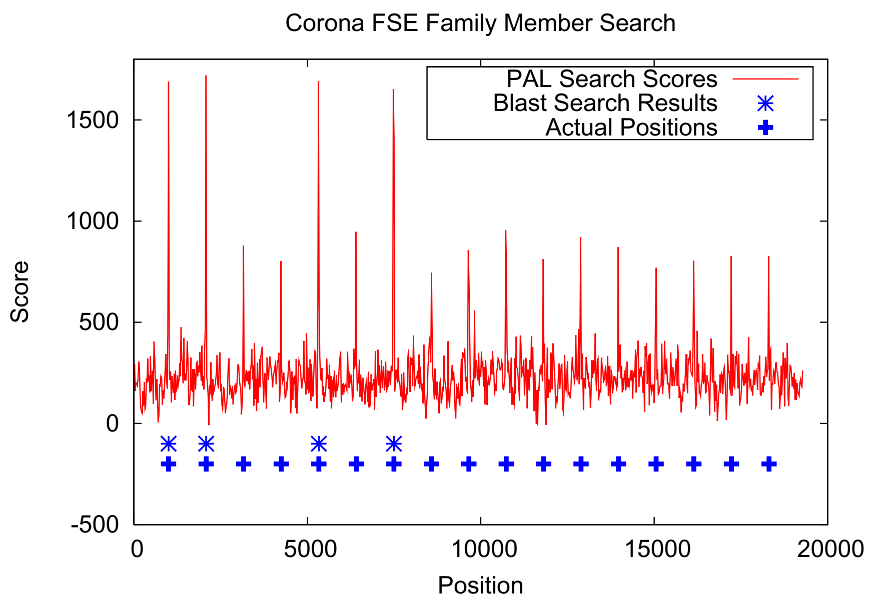


Figure 2.11: The Score plot for Corona-FSE homologue search using PAL as a pseudoknot RNA search tool. '*' denotes actual positions of the members and '+' denotes the members located by Blastn.

Table 2.4: Comparison PAL against BLAST on pseudoknotted RNA families.

RNA Family	**# Found** BLAST	PAL
UPSK	3	3
Antizyme	12	12
Parecho CRE	4	4
Corona-FSE	4	17
Corona-pk3	5	13
IFN-gamma	4	4

```
Query:    Human chromosome 12, minus, 66839786 - 66839618
Subject:  Mouse chromosome 10, plus, 118018890-118019061

                .......AAAAAAA<<<<<<<<         ..<<<<<.  .  <<<<<....<<<<<<<<..
Query:          CAUUGUUCUGAUCAUCUGAAGA---------UCAGCUAU--U--AGAAGAGAAAGAUCAGUUA
                ||      +**++ +*+*+++*         ||+***+   |  ** **  | *  **** |
Sbjct:          CA-----GAGAGGUGCAGGCUAUAGCUGCCAUCGGCUGACCUAGAGAAG--ACACAUCAGCU-
                .......AAAAAAA<<<<<<<<...........<<<<<......<<<<<....<<<<<<<<..


                <<<..<<<<....aaaaaaa .......>>>>.>>>>>>>>>>>...>>>>>.>>>>>....>
Query:          AGUCCUUUGGACCUGAUCAG-CUUGAUACAAGAACUACUGAUUUCAACUUCUUUGGCUUAAUU
                ++*||****||   ++**+  |||||  |**++|++* ++*+  *|  ** ** +***+    *
Sbjct:          GAUCCUUUGGA--CCCUCUGACUUGAGACAGAAGUUCUGGGCUUCUCCUCCUGCGGCC----U
                <<<..<<<<....aaaaaaa........>>>>.>>>>>>>>>>>...>>>>>.>>>>>....>


                >>.>>>>><<<<<<<..  <<<<<....>>>>>...<<<<....>>>>...    >>>>>>>
Query:          CUCUCGGAAACGAUGAA--AUAUACAAGUUAUAUCUUGGCUUUUCAGCUCUG---CAUCGUU
                ++|**+**+ *+***||   * +*||   *+ *  |||****|| |****||    ***+* *
Sbjct:          AGCUCUGAGACAAUGAACGCUACACA--CUGCAUCUUGGCUUUGCAGCUCUUCCUCAUGGCU
                >>.>>>>><<<<<<<....<<<<<....>>>>>...<<<<....>>>>.......>>>>>>>
                            Start codon
```

Figure 2.12: Structural alignment of the Human Interferon-γ pseudoknot against mouse upstream genomic DNA. The structure of the query is denoted by parenthesis $<, >$" , and "A,a" for the pseudoknot. The symbols describe the conservation: (*) sequence and structure is conserved. (+) structure is conserved but not sequence. (|) sequence is conserved, but not structure.

2.9 Discussion

While PAL is accurate in fishing for structural homologs, it is computationally intensive, making genome scale searches intractable. However, there has been much recent research (including our own work) on *computational filters* for RNA, which quickly eliminate much of the database, while retaining the true homologs [26, 30]. We used PAL in conjunction with sequence based filters [29] to search genomes, for the 3 most interesting families.

The Corona-FSE family (RF00507) is a conserved pseudoknot in Coronaviruses which can promote ribosomal frameshifting [5]. We searched the entire Viral genome (79 Mb) for homologs of this family in 33.8 CPU hours on 1.6GHz AMD Opteron Grid, and identified 11 novel members of the sub-family. Like other known members, these are found in coronaviruses, murine hepatitis virus, and Avian flu viruses. Only 2 of the 11 were similar enough in sequence to be identified by BLAST. The alignments can be retrieved from (http://www.cse.ucsd.edu/∼bdost/RF00507.htm). A similar result was obtained for Corona-pk3. This family has a conserved $\sim$ 55nt pseudoknot structure which has been shown to be necessary for viral genome replication [27]. We identified 20 novel members of this family with significant scores (See http://www.cse.ucsd.edu/∼bdost/RF00165.htm). Only 1 of the 20 was similar enough in sequence to be identified by BLAST.

The Interferon-gamma family is an interesting example of a pseudoknot that is found in the 5'UTR of the Interferon-gamma gene. It regulates translation of the downstream gene by binding to the kinase PKR, a known regulator of IFN-gamma translation [6]. After its discovery in 2002, the pseudoknot was found to be conserved in many mammals. Its presence in rodents was speculated, but the homolog was not located. We searched in mouse and rat genomic DNA, and in the complete gene of gerbil. In all 3 species, we clearly identified the homologs as the top-scoring alignment. The alignment of human and mouse pseudoknots are shown in Figure 2.12. The conserved location in the two species, just upstream of the start codon, and conservation of key elements validates the hit. We are working with collaborators on experimental validation, and to locate more members of this family.

In conclusion, we demonstrate that the algorithm for aligning pseudoknots, implemented as PAL represents a viable tool for searching for novel homologs, and for structural inference. We hope that our tool will help increase the impact and influence of pseudoknotted RNA in cellular function. PAL and supplemental data are available upon request.

Chapter 2 is, in full, a reprint of the paper "Structural alignment of pseudoknotted RNA. B. Han, B. Dost, S. Zhang and V. Bafna (2008). Journal of Computational Biology. 15(5): 489-504". The dissertation author was the primary investigator and first author of this paper jointly with Buhm Han. This research was, in part, previously published in the conference proceedings of Recomb 2006.

Chapter 3

Querying Protein Interaction Networks

Abstract

Molecular interaction databases can be used to study the evolution of molecular pathways across species. Querying such pathways is a challenging computational problem, and recent efforts have been limited to simple queries (paths), or simple networks (forests). In this chapter, we significantly extend the class of pathways that can be efficiently queried to the case of trees, and graphs of bounded treewidth. Our algorithm allows the identification of non-exact (homeomorphic) matches, exploiting the color coding technique of Alon et al. We implement a tool for tree queries, called QNet, and test its retrieval properties in simulations and on real network data. We show that QNet searches queries with up to 9 proteins in seconds on current networks, and outperforms sequence-based searches. We also use QNet to perform the first large scale cross-species comparison of protein complexes, by querying known yeast complexes against a fly protein interaction network. This comparison points to strong conservation between the two species, and underscores the importance of our tool in mining protein interaction networks.

3.1 Introduction

The study of biological networks has gained substantial interest in recent years. In particular, technological advances, such as the yeast two-hybrid [42] and co-immuno precipitation assays [46], have enabled the large-scale mapping of protein-protein interactions (PPIs) across many model species. The newly available PPI networks present a host of new challenges in studying protein function and evolution. Key to addressing these challenges is the development of efficient tools for network database searches, much the same as sequence searches have been instrumental in addressing similar problems at the genome level.

Network queries call for searching a "template" subnetwork within a network of interest. Commonly, the query is a known pathway, and the network is searched for subnetworks that are similar to the query. Similarity is measured both in terms of protein sequence similarity and in terms of topological similarity. The hardness of the problem stems from the non-linearity of a network, making it difficult to apply sequence alignment techniques for its solution.

Several authors have studied the network querying problem, mostly focusing on queries with restricted topology. Kelley et al. [44] devised an algorithm for querying linear pathways in PPI networks. While the problem remains NP-hard in this case as well (as, e.g., finding the longest path in a graph is NP-complete [38]), an efficient algorithm that is polynomial in the size of the network and exponential in the length of the query was devised for it. Pinter et al. [48] enable fast queries of more general pathways that take the form of a tree. However, their algorithm is limited to searching within a collection of trees rather than within a general network. Sohler and Zimmer [37] developed a general framework for subnetwork querying, which is based on translating the problem to that of finding a clique in an appropriately defined graph. Due to its complexity, their method is applicable only to very small queries. Recently, some of us have provided a comprehensive framework, called QPath, for linear pathway querying. QPath is based on an efficient graph theoretic technique, called color coding [32], for identifying subnetworks of "simple" topology in a network. It improves upon [44] both in speed and in higher flexibility in non-exact matches.

In this chapter, we greatly extend the QPath algorithm to allow queries with more general structure than simple paths. We provide an algorithmic framework for handling tree queries under non-exact (homeomorphic) matches (Section 3.3.1). In this regard, our work extends [48] to querying within general networks, and the results in [32] to searching for homeomorphic rather than isomorphic matches. More generally, we provide an algorithm for querying subnetworks of bounded treewidth (Section 3.3.2). We implemented a tool for tree queries which we call QNet. We demonstrate that QNet performs well both in simulation of synthetic pathway queries, and when applied to mining real biological pathways (Section 6.3). In simulations, we show that QNet can handle queries of up to 9 proteins in seconds in a network with about 5,000 vertices and 15,000 interactions, and that it outperforms sequence-based searches. More importantly, we use QNet to perform the first large scale cross-species comparison of protein complexes, by querying known yeast complexes in the fly protein interaction network. This comparison points to strong conservation of protein complexes structures between the two species. For lack of space some algorithmic details are omitted in the sequel.

3.2 The Graph Query Problem

Let $G = (V, E, w)$ be an undirected weighted graph, representing a PPI network, with a vertex set V of size n, representing proteins, an edge set E of size m, representing interactions, and a weight function $w : E \to \mathcal{R}$, representing interaction reliabilities.

Let $G_Q = (V_Q, E_Q)$ denote a query graph with k vertices. We reserve the term *node* for vertices of G_Q and use the term *vertex* for vertices of G.

Let $h(q, v)$ denote a similarity score between query node $q \in V_Q$ and vertex $v \in V$. In our context, vertices correspond to proteins, and their similarity score is a function of their sequence similarity. A query node q is referred to as *homologous* to a graph vertex v, if the corresponding similarity score $h(q, v)$ exceeds a predefined threshold.

A *subdivision* of an edge (u, v) in a graph $H = (U, F)$ replaces it with two edges (u, w) and (w, v), where $w \notin U$, i.e., creating a new graph $H' = (U \cup \{w\}, F \cup \{(u, w), (w, v)\} \setminus \{u, v\})$. H is considered *extendable* to a graph G, if G can be obtained from H by a series of subdivisions. In particular, H is then homeomorphic to G.

An *alignment* of the query graph G_Q to G is defined as a pair of: (i) a subgraph $G_A = (V_A, E_A)$ of G, referred to as the *alignment subgraph*; and (ii) a bijection, $\sigma : V_Q^S \to V_A^S$, between a subset of query nodes, $V_Q^S \subseteq V_Q$, and homologous vertices in the alignment subgraph, $V_A^S \subseteq V_A$. The vertices in $V_Q^S \cup V_A^S$ are called *skeleton* vertices. Pairs of associated vertices $(q, \sigma(q)) \in V_Q^S \times V_A^S$ are called *aligned*.

An alignment is *proper* if there exists a pair of *skeleton* graphs $S_Q = (V_Q^S, E_Q^S)$ and $S_A = (V_A^S, E_A^S)$ that satisfy the following conditions: (i) there is an isomorphism between S_Q and S_A which respects the alignment (i.e., there is an edge $(u, v) \in E_Q^S$ iff there is an edge $(\sigma(u), \sigma(v)) \in E_A^S$); and (ii) S_Q is extendable to G_Q and S_A is extendable to G_A. In particular, this means that G_Q and G_A are required to be homeomorphic. In the rest of the chapter we discuss proper alignments only. An example of such an alignment is given in Figure 3.1a.

Query nodes that are not aligned with vertices in the alignment subgraph are considered to be *deleted*. Conversely, vertices in the alignment subgraph that are not aligned with query nodes are considered to be *inserted*. Insertions and deletions are also referred to as *indels*. From the above definitions, inserted and deleted vertices must be of degree 2 in their respective graphs. An alignment which involves no insertions or deletions is considered *simple*. The weight of an alignment is the sum of: (i) similarity scores of aligned vertices, (ii) weights of edges in the aligned subgraph, (iii) a penalty score, δ_d, for each node deletion, and (iv) a penalty score, δ_i, for each vertex insertion.

The *graph query problem* is formally defined as follows: Given a query graph G_Q, a graph G, a similarity score h, and penalty scores for insertions and deletions, find a proper alignment of G_Q in G with maximal weight. In practice, we would also like to limit the number of insertions and deletions in the alignment, to control the evolutionary distance between the two subnetworks. To this end, we also consider a variant of the problem in which the number of insertions is limited by N_{ins}, and the number of deletions is limited by N_{del}.

3.3 Graph Query Algorithms

The complexity of the graph query problem depends on the topology of the query graph G_Q, the topology of the graph G, and the similarity function h. In the general case, the problem of finding simple alignments is in general equivalent to subgraph isomorphism [59], which is computationally hard. In this chapter, we focus on efficient query algorithms by exploiting the underlying biological constraints. Specifically, motivated by known pathways in KEGG [43], we consider restricted query topologies, i.e., the query graph being a *tree*, and a graph of *bounded treewidth* (see also [48]). For these special structures, we adapt the color coding method of Alon et al. [32] to make the problem tractable.

Color coding is a randomized technique for finding simple paths and simple cycles of a specified length k within a given graph of size n. The basic idea is to randomly assign k colors to the vertices of the graph and then search for *colorful* paths in which each color is used exactly once. Thus, rather than having to maintain a list of vertices visited so far (of size $O(n^k)$), one can maintain a list of colors at considerably lower complexity ($O(2^k)$).

The use of the color coding technique within a query algorithm is intuitively similar. We construct an optimal alignment by extending optimal sub-alignments using dynamic programming. Adding a network vertex to the optimal alignment can be done only if this vertex is not already contained in the sub-optimal alignment. Thus, naively, each potential sub-optimal alignment should maintain the list of at most k vertices already matched. This yields $O(n^k)$ potential alignments. In color coding, we apriori color each network vertex randomly with one of k colors, looking for a colorful alignment. Consequently, we only need to maintain a list of used colors (of size $O(2^k)$), which significantly reduces the computation time. However, the computation returns a correct answer only if the optimum alignment is colorful, which happens with probability $\frac{k!}{k^k} \simeq e^{-k}$. Therefore, if we repeat the experiment $\ln(\frac{1}{\epsilon})e^k$ times, we get the optimum alignment with probability at least $1 - \epsilon$ for any desired value of ϵ.

3.3.1 Tree Query

We describe an algorithm for solving the graph query problem assuming that the query graph is a tree. For ease of presentation, we start by presenting a simplified version of the algorithm that limits the number of insertions only. The proper treatment of limiting both the number of insertions and deletions is deferred to the end of the section.

First, we root G_Q arbitrarily at a node r with degree 1. For each query node q, denote its children by $q_1, \ldots, q_{n_q}$, where n_q denotes their number. Let $T_{q,j}$ denote the tree that includes q and the subtrees rooted at each of its first j children, for $1 \leq j \leq n_q$. The algorithm proceeds in a series of trials in which every vertex $v \in V$ is independently assigned a color $c(v)$ drawn uniformly at random from the set $C = \{1, 2, \ldots, k+N_{ins}\}$. Given the random vertex colors, we employ dynamic programming to identify an optimal colorful alignment. Let $W^M(q, v, S, j)$ denote the maximal score of an alignment of $T_{q,j}$ in G, such that query node q is aligned with graph vertex v, with the aligned subgraph receiving distinct colors from $S \subseteq C$. The recursion is initialized by setting $W^M(q, v, S, 0) = h(q, v)$ for leaf nodes q, and is formulated as follows:

$$W^M(q,v,S,j) = \max_{\substack{(u,v) \in E \\ S' \subset S}} \begin{cases} (\star \;\; \textit{Match, child}\, j \;\; \star) \\ W^M(q,v,S',j-1) + W^M(q_j,u,S-S',n_{q_j}) + w(u,v), \\ \\ (\star \;\; \textit{Insertion, vertex}\, u \;\; \star) \\ W^M(q,v,S',j-1) + W^I(q_j,u,S-S') + w(u,v), \\ \\ (\star \;\; \textit{Deletion, child}\, j \;\; \star) \\ W^M(q,v,S',j-1) + W^D(q_j,v,S-S') \end{cases}$$

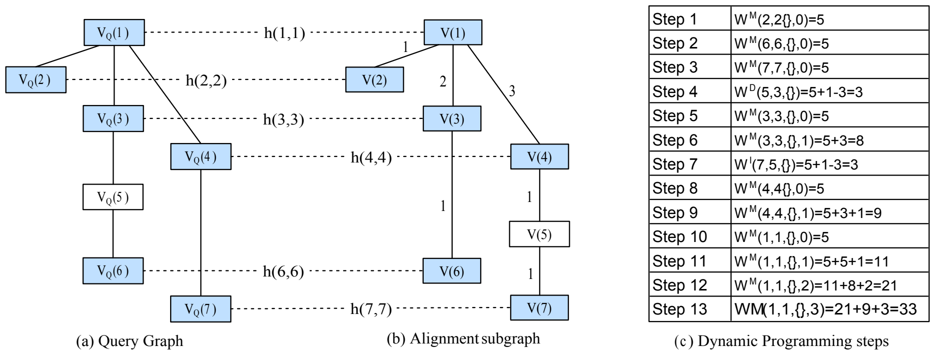
Here $W^I(q, v, S)$ denotes the optimal score of an alignment of T_{q,n_q} in G, such that q is aligned with some vertex u that is a descendant of v in the aligned subgraph. $W^D(q, v, S)$ denotes the optimal score of the alignment of $T_{q,1}$ in G, such that q is deleted and v is aligned with an ancestor of q. The recursions for the insertion and deletions cases are given below. For query nodes q of degree other than 2, we set

$W^D(q, v, S) = -\infty.$

$$W^I(q, v, S) = \max_{u : (u,v) \in E} \begin{cases} W^M(q, u, S - \{c(v)\}, n_q) + w(u, v) + \delta_i, \\ W^I(q, u, S - \{c(v)\}) + w(u, v) + \delta_i \end{cases}$$

$$W^D(q, v, S) = \max_{u : (u,v) \in E} \begin{cases} W^M(q_1, u, S, n_{q_1}) + w(u, v) + \delta_d, \\ W^I(q_1, u, S) + w(u, v) + \delta_d, \\ W^D(q_1, v, S) + \delta_d \end{cases}$$

The maximal score of the alignment is $\max_{v,S} W^M(r, v, S, 1)$. The optimal alignment is obtained through standard dynamic programming backtracking. An application of the dynamic programming recursions to a sample query is demonstrated in Figure 3.1.



Step 1	W^M(2,2{},0)=5
Step 2	W^M(6,6,{},0)=5
Step 3	W^M(7,7,{},0)=5
Step 4	W^D(5,3,{})=5+1-3=3
Step 5	W^M(3,3,{},0)=5
Step 6	W^M(3,3,{},1)=5+3=8
Step 7	W^I(7,5,{})=5+1-3=3
Step 8	W^M(4,4{},0)=5
Step 9	W^M(4,4,{},1)=5+3+1=9
Step 10	W^M(1,1,{},0)=5
Step 11	W^M(1,1,{},1)=5+5+1=11
Step 12	W^M(1,1,{},2)=11+8+2=21
Step 13	WM(1,1,{},3)=21+9+3=33

Figure 3.1: (a) An example of a tree query graph and the corresponding alignment subgraph. Numbers on the query graph's edges represent an arbitrary ordering of children nodes. Aligned query nodes and graph vertices are connected with dashed lines. Nodes in the skeleton graphs appear in gray. (b) A simulation of the dynamic programming recursions. For simplicity, we denote color sets as $\{\}$. Matched vertices are awarded by $+5$, insertions and deletions are penalized by -3 and edge weights are as shown.

The running time of each trial is $2^{O(k+N_{ins})}m$. The probability of receiving distinct colors for the vertices of the optimal matching tree is at least $e^{-k-N_{ins}}$. Thus, the running time of the algorithm is $2^{O(k+N_{ins})}m\ln(\frac{1}{\epsilon})$ for any desired success probability

$1 - \epsilon$ (where $\epsilon > 0$). We note that it is straightforward to limit the number of deletions to N_{del} by incorporating an additional variable in the recursions to count the number of deletion in the optimal sub-alignment. The cost in terms of running time is multiplicative in N_{del}. When incorporating such a variable, it is also easy to limit the number of insertions to N_{ins} by choosing the optimum solution based on its number of deletions and the cardinality of its color set.

3.3.2 Bounded Treewidth Graph Query

The algorithm for matching trees can be extended to subgraphs that have tree-like properties. We present an algorithm for the simpler case where no indels are allowed and defer the description of an algorithm for the general case to the appendix. Intuitively, the treewidth of a graph indicates how close the graph is to being a tree, where a tree has treewidth 1. The maximal treewidth value for a graph with n vertices is $n - 1$ and this value is attained by an n-vertex clique. A formal definition of a treewidth and the associated tree-like structure follows.

A *tree decomposition* (X, T) of the query graph $G_Q = (V_Q, E_Q)$ is defined as follows (see, e.g., [45]): $T = (I, F)$ is a rooted binary tree, and $X = \{X_i \subseteq V_Q : i \in I\}$ is a collection of subsets of V_Q, such that $\bigcup_{i \in I} X_i = V_Q$ and the following conditions are satisfied:

1. For each edge $(u, v) \in E_Q$ there exists $i \in I$ such that $u, v \in X_i$.

2. If $i, j, k \in I$ and j is on the path from i to k in T, then $X_i \bigcap X_k \subseteq X_j$.

The *treewidth* of the tree decomposition is $max_{i \in I}|X_i| - 1$. An example of a graph and its tree decomposition is given in Figure 3.2a,b.

Let t denote a bound on the treewidth of G_Q. We add a dummy node d as a parent of the root of T, with $X_d = \emptyset$. To avoid confusion, we call the nodes of T, *super-nodes*. For a non-leaf tree super-node $X_i \in X$, denote its two children by X_{i_1} and X_{i_2}. Let T_i denote the subtree of T that is rooted at X_i. The algorithm proceeds in a series of trials in which every vertex $v \in V$ is independently assigned a color $c(v)$ drawn uniformly at random from the set $\{1, 2, \ldots, k\}$. Given the random vertex colors, we employ dynamic programming to identify an optimal colorful alignment.

The properties of the tree decomposition enable us to identify the optimal alignment by recursing on T and maintaining sub-optimal alignments of query nodes spanned by subtrees of T, similar to the tree query algorithm described above. However, there are two main difficulties to tackle: (i) A set of query nodes, X_i, may have an arbitrary topology (e.g., forming a clique), potentially requiring an exhaustive $O(n^{t+1})$-time search of an alignment subgraph for it. (ii) A query node v may appear in more than a single super-node.

For the first issue, we exploit the fact that the treewidth is bounded by t. Large values of t would make the algorithm impractical. To cope with the second difficulty, we note that by definition, if $v \in X_{i_j}$ and $v \notin X_i$, then $v \notin X_l$ for all super-nodes X_l that are not descendants of X_i in the tree. Thus, when visiting a certain super-node X_{i_j}, it contains *active* query nodes $X_{i_j}^A = X_i \cap X_{i_j}$ that are yet to be handled, and *non-active* nodes $X_{i_j}^N$ that can be removed from consideration when traversing up the tree (Figure 3.2b). We define a *non-active* edge at a super-node X_i, as a query edge touching a non-active node in X_i. We let E_i^N denote the set of non-active edges in super-node X_i.

We need some more notation before giving the main recurrence of the algorithm. For each $X_i \in X$, let Σ_i denote the $O(n^{t+1})$-size set of all mappings $\sigma : X_i \to V$ such that: (i) for all distinct $q_1, q_2 \in X_i$, $c(\sigma(q_1)) \neq c(\sigma(q_2))$; and (ii) if $(q_1, q_2) \in E_Q$ then $(\sigma(q_1), \sigma(q_2)) \in E$. Figure 3.2b,c shows an example of mappings between query nodes and graph vertices.

For computing the weight of an alignment, it is convenient to credit each super-node i (when traversing up the tree) with the similarity scores associated with its non-active nodes and the edge weights corresponding to its non-active edges. The node term is $W^S(i, \sigma) = \sum_{u \in X_i^N} h(u, \sigma(u))$. The edge term is $W^E(i, \sigma) = \sum_{(u_1,u_2) \in E_i^N} w(\sigma(u_1), \sigma(u_2))$.

Let $W(i, \sigma, S)$ be the maximum weight of an alignment of a subgraph of G_Q that includes all super-nodes in $T_i - X_i$, identifies on the active query nodes in super-node i with the assignment $\sigma \in \Sigma_i$, and uses the colors in $S \subseteq C$. $W(i, \sigma, S)$ can be recursively

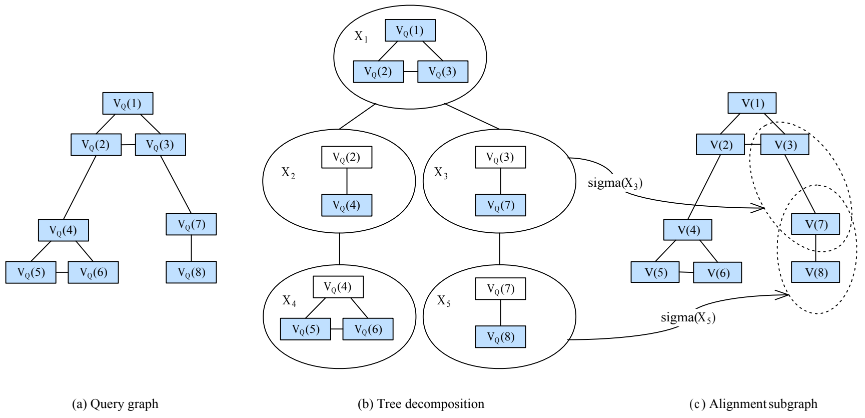


Figure 3.2: (a) An example of a query graph with a treewidth of 2. (b) A tree decomposition of the query graph such that each super-node has no more than 3 query nodes associated with it. Non-active query nodes are grayed. (c) An alignment subgraph. $\sigma(X_3)$ and $\sigma(X_5)$ are mappings of the query nodes in X_3 and X_5 to graph vertices, respectively, that identify on the active query node $V_Q(6)$ in X_5.

computed as follows. For a leaf i, $W(i, \sigma, S) = 0$. For all other super-nodes:

$$W(i,\sigma,S) = \max_{\substack{S_1 \uplus S_2 = S \\ \sigma_1,\sigma_2}} \sum_{j=1}^{2} \Big[\; W(i_j,\sigma_j,S_j) + W^S(i_j,\sigma_j) + W^E(i_j,\sigma_j) \; \Big]$$

where σ is consistent with $\sigma_1 \in \Sigma_{i_1}$ and $\sigma_2 \in \Sigma_{i_2}$.

The score of an optimal alignment of G_Q is thus $\max_S W(d, \emptyset, S)$. The total running time is $2^{O(k)} n^{t+1}$.

3.4 Implementation Notes

We implemented a tool, QNet, for querying a given network with a tree subnetwork, following the algorithm given in Section 3.3.1. Bounded treewidth queries will be supported in future versions. To allow higher flexibility in matching a query, we slightly generalized the tree query algorithm to enable also deletions of query nodes of degree 1 (leaves of the tree). We also included in QNet a heuristic that exploits the structure of the homology function to reduce the number of color coding iterations needed. In the following we describe this heuristic and the parameter setting employed in QNet.

Restricted Color Coding. We present a heuristic approach to color coding that tries to take advantage of queries whose protein members tend to have non-overlapping sets of homologs. First, we assign each query node a distinct *match color*, and choose N_{ins} additional *insertion colors*. Now, we color the network vertices using the following rule: For each network vertex v, if v is not homologous to any query protein, then assign it with a random insertion colors. Otherwise, toss a coin with probability $p_t = \frac{N_{ins}}{k+N_{ins}}$. If HEADS, choose a random insertion color for it, else if TAILS, assign it with a random color from the set of query nodes it is homologous to.

The probability P_s to obtain a colorful alignment subgraph is at least the probability that: (i) each aligned vertex is given a match color, and each inserted vertex is given an insertion color; and (ii) all colors are distinct. Let p_m be the probability that aligned vertices are colorful, and p_i be the probability that insertion vertices are colorful. Then

$$P_s = (1 - p_t)^k p_t^{n_i} p_i p_m = \left(\frac{k}{N_{ins} + k}\right)^k \left(\frac{N_{ins}}{k + N_{ins}}\right)^{N_{ins}} p_i p_m$$

where $p_i \geq \frac{N_{ins}!}{{N_{ins}}^{N_{ins}}}$. It remains for us to compute a lower bound for p_m. To this end, we form a graph on the set of query nodes, in which for every pair q, q' of query nodes, we add the edge (q, q') if there exists a network vertex v that is homologous to both. We then partition the query vertices into connected components $Q_1, Q_2, \ldots, Q_{k'}$, and use the following bound: $p_m \geq \prod_{u=1}^{k'} \frac{|Q_u|!}{|Q_u|^{|Q_u|}}$. We expect p_m to be high since often query nodes are homologous to a single vertex. When the probability of success with restricted coloring is greater than the probability of success with the standard color coding (i.e., $\frac{(k+N_{ins})!}{(k+N_{ins})^{k+N_{ins}}}$), we use this procedure, and otherwise we use the standard color coding.

Parameter Setting. QNet involves several parameters controlling sequence similarity, insertion/deletion penalties, and the relative weights of edge- and node-terms. The current settings are as follows: we used blastp with an E-value threshold of 10^{-7} to compute sequence similarity, and set $h(q, v) = -log(\texttt{E-value})$. Interaction reliabilities $p(u, v)$ are assigned using a logistic regression scheme based on the experimental evidences for the interactions, as described in [49]. We use $w(u, v) = c \cdot r(u, v)$, where c is chosen to ensure the same scale for the reliability and homology values. We allow at most two insertions and two deletions per query, i.e., $N_{ins} = N_{del} = 2$. Indel penalties are set to $\delta_d = \delta_i = -100$. We empirically tested a range of penalties by querying perturbations of subtrees in the yeast network (see Section 3.5.1). A small set of queries were examined and the results did not change over the range as long as the net influence of a deletion or insertion were kept negative. In all runs reported below, the number of color coding iterations was set to ensure success probability ≥ 0.99.

3.5 Experimental Results

To evaluate the performance of QNet we measure its running time and accuracy under various configurations. We start by applying QNet to query a set of synthetic trees in the PPI network of yeast, measuring its running time and accuracy. Next, we show examples of querying known yeast and human signal transduction pathways in the PPI network of fly. Finally, we apply QNet to query known yeast complexes in fly.

Protein-protein interaction data for yeast *S. cerevisiae* and fly *D. melanogaster* were obtained from the Database of Interacting Proteins (DIP) [51] (April 2005 download). The fly data was complemented by PPI interactions from [50] and by genetic interactions from FlyGRID (see also [49]). Altogether, the yeast network consists of 4,738 proteins and 15,147 interactions, and the fly network consists of 7,481 proteins and 26,201 interactions.

3.5.1 Synthetic Query Trees

To measure the running time and estimate the accuracy of QNet, we applied it to query the PPI network of yeast with a set of synthetic query trees. This set consists of 20 randomly chosen subtrees of sizes ranging from $k = 5$ to $k = 9$ from the yeast PPI network. Each query tree was perturbed with up to 2 node insertions and deletions, and by a pre-specified amount of point mutations in its proteins' sequences of average length ~ 500. QNet was applied to identify a match for each query tree.

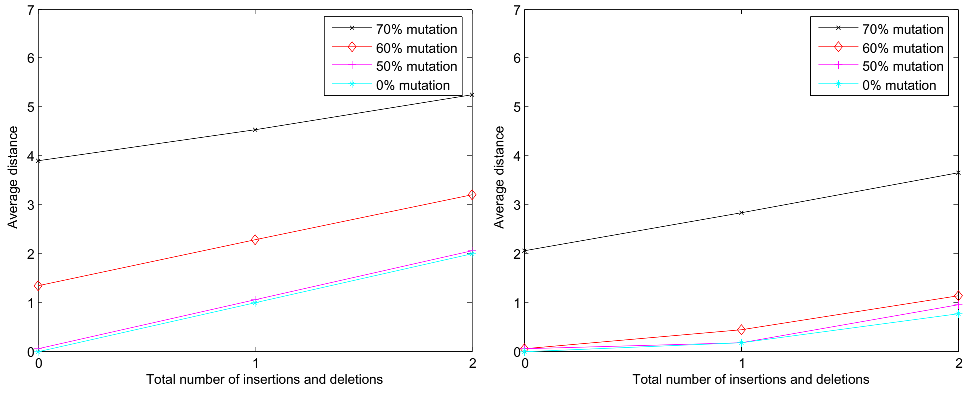
The running time measurements were performed on a standard PC (2GHz, 1Gb). We find that the running time of QNet is a few seconds in all cases, reaching an average of 11 seconds for the largest tree queries with 9 nodes (Table 3.1). To measure the

Table 3.1: Number of color coding iterations and timing statistics for QNet. The last two columns show the average time per query. The algorithm's parameters are set as follows: $N_{ins} = 2$, $N_{del} = 2$, and the probability of success is set to 0.99.

	#Iterations		Avg. time (sec.)	
Query size (k)	Standard color coding	Restricted color coding	Standard color coding	Restricted color coding
5	752	603	1.71	1.58
6	1916	917	6.36	4.73
7	4916	1282	20.46	6.24
8	12690	1669	61.17	9.08
9	32916	2061	173.88	11.03
10	85720	2509	1463	21.74
11	223990	2987	5501	41.39
12	1891868	4623	50455	97.93

improvement in running time introduced by the restricted color coding heuristic, we applied QNet also without this heuristic. We find that restricted color coding significantly reduces the number of iterations required to identify the optimal match, while the running time of each iteration remains similar. Overall, restricted color coding reduces the running time by an order of magnitude on average (Table 3.1). The running time of the algorithm is significantly affected by the number of insertions allowed. If no insertions are allowed, the average number of iterations required for queries of size 9 is less than 100. When increasing the number of allowed insertions to above 2, the restricted color coding heuristic becomes less effective (data not shown).

To evaluate the accuracy of the matched trees, we computed the symmetric difference between the protein set of a query and its match, termed their *distance* herein. The results show that when perturbing protein sequences in up to 60% of the residues, the average distance between the matched tree and the original tree is lower than 1 (Figure 3.3b). Moreover, we compared the accuracy of matches obtained by QNet to matches that are based only on best BLAST hits. We found that matches obtained by QNet are markedly more accurate than purely sequence-based matches, showing that the topology of the query tree carries important signal (Figure 3.3a). Evidently, the advantage of QNet over a sequence-based approach becomes more pronounced when the mutation rate increases.



(a) (b)

Figure 3.3: The average distance of the matched tree from the original tree is plotted against the total number of insertions and deletions introduced to the query for 4 different mutation levels. (a) Performance of a sequence-based approach. (b) Performance of QNet.

3.5.2 Cross-Species Comparison of MAPK Pathways

The mitogen-activated protein kinase (MAPK) pathways are a collection of related signal transduction pathways, which play a critical role in mediating the cellular response to various toxic stresses [36]. The pathways are known to be conserved across species and, hence, serve as controlled tests to QNet.

We queried MAPK pathways from the KEGG database [43] in the PPI network of fly. The first pathway is a classical human MAPK pathway involved in cell proliferation and differentiation. Querying this pathway in fly resulted in detecting a known MAPK pathway involved in dorsal pattern formation (Figure 3.4a). Specifically, 6 out of the 8 matched proteins in the target are members of the known MAPK pathway in fly. Similar results were obtained by querying the yeast MAPK pathways from KEGG against the fly network. As an example, the top output for the starvation response pathway query (Figure 3.4b) is a fly MAPK pathway with a putative MAPK cascade (fray,Dsor1,rl), which includes the GTPases Cdc42, Ras64b that are homologous to the two GTPases in the query. These results support the fidelity of QNet.

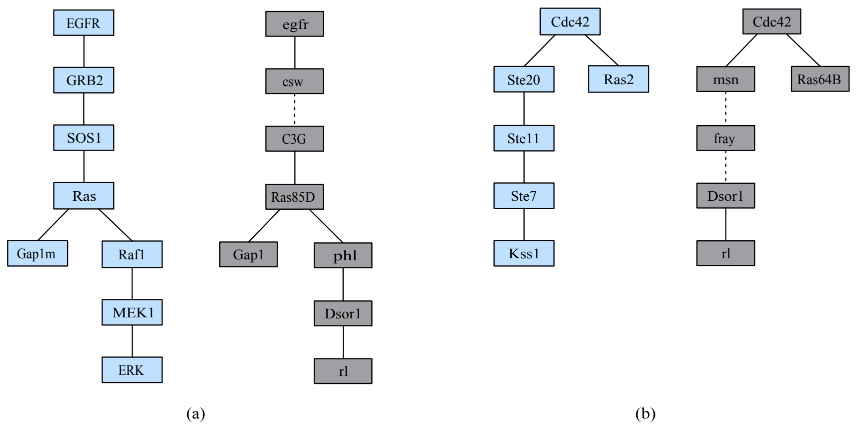


Figure 3.4: [
Querying the fly network using (a) a human MAPK pathway, and (b) a yeast MAPK pathway induced by starvation]Querying the fly network using (a) a human MAPK pathway, and (b) a yeast MAPK pathway induced by starvation, taken from the KEGG database [43]. Matched nodes appear on the same horizontal line. A dotted edge represents inserted proteins (not shown).

3.5.3 Cross-Species Comparison of Protein Complexes

As a large-scale validation of QNet we systematically queried known yeast protein complexes, obtained from the MIPS database [40, 47], in the fly network, and tested the biological plausibility of the identified matches. We included all hand curated complexes in MIPS, which are considered a reliable data source, excluding complexes that were identified via high throughput measurements (category 550 in MIPS). Overall, we considered 94 complexes consisting of at least 4 proteins each. As MIPS does not contain information on the topology of the complexes, we mapped each complex to the yeast network and used the induced subnetworks as queries. More accurately, for each complex, we extracted an average of 40 random query trees of size in the range $3 - 8$ from its induced subnetwork. We applied QNet to systematically query all of the induced query trees in fly. The resulting query matches were used to construct a *consensus match*, consisting of all proteins that appeared in at least half of the matches.

The biological plausibility of an obtained consensus matches was tested based on functional enrichment of their member proteins w.r.t. the fly gene ontology (GO) process

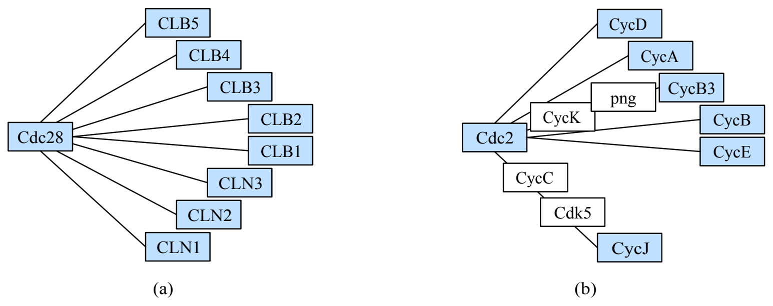


Figure 3.5: (a) The MIPS Cdc28p complex. (b) The consensus match in fly. Matched nodes appear on the same horizontal line. Inserted proteins appear in white.

annotation [33]. Specifically, let $n(t)$ denote the number of genes in the consensus match that are annotated with term t. We compute the probability $p(t)$ of obtaining a random set of genes, of the same size as the original pathway, with at least $n(t)$ genes annotated with term t, assuming a hypergeometric distribution. Having found a term t_0 with minimal probability $p(t_0)$, we compute a p-value for the enrichment under term t_0 by comparing $p(t_0)$ with similar values computed for $10,000$ random sets of genes. The latter p-values are further corrected for multiple match testing via the false discovery rate procedure [34].

36 of the yeast complexes resulted in a consensus match with more than one protein in fly. We find that 72% of these consensus matches are significantly functionally enriched ($p < 0.05$). For comparison, we computed the functional enrichment of randomly chosen trees from the fly PPI network that have the same distribution of sizes and interactions scores as the consensus matches. We find that only 17% of the random trees are functionally enriched, and that the mean enrichment p-values is significantly lower for the true consensus matches (Wilcoxon rank test p-value$< 6.5e-9$).

Figure 3.5 illustrates the result of querying the Cdc28p complex. This complex is composed of cyclin-dependent kinases involved in regulating the cell cycle in yeast. The consensus match obtained in fly consists solely of cyclin-dependent kinases and significantly overlaps the cyclin-dependent protein kinase holoenzyme complex (GO:0000307).

3.6 Discussion

Data sets of protein-protein interactions are increasingly common, and will continue to increase in number and complexity. In this chapter, we address the problem of searching such data for specific pathways of interest. We provide efficient algorithms for querying trees and graphs of bounded treewidth within PPI networks. We implement the tree query algorithm, QNet, and demonstrate its efficiency and accuracy. QNet can handle queries of up to 9 proteins in seconds on current networks, and is shown to outperform sequence-based homology searches. More importantly, we use QNet to perform a large scale cross-species comparison of protein complexes, by querying known yeast complexes in the fly network. This comparison points to strong conservation between the two species.

While our work has helped in clarifying some algorithmic questions regarding efficient querying of biological networks, and has shown promising results in practice, it leaves many aspects open for future research. One important direction is the development of appropriate score functions to better identify conserved pathways. Research in this direction could gain from probabilistic models of network evolution [35, 41]. A second important direction is the application of the methods developed here to queries of more general structure. This entails both the implementation and testing of a tool for querying bounded treewidth graphs, and the use of such a tool for querying arbitrary structures, perhaps in a way similar to that presented in Section 3.5.2.

Chapter 3 is, in full, a reprint of the paper "QNet: a tool for querying protein interaction networks. B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan (2008). Journal of Computational Biology. 15(7):913-25". The dissertation author was the primary investigator and author of this paper jointly with Tomer Shlomi. This research was, in part, previously published in the conference proceedings of Recomb 2007.

Appendix: A General Alignment Algorithm for Bounded Treewidth Queries

In Section 3.3.2 we described an algorithm for identifying optimal simple alignments of a bounded treewidth query graph. To generalize the algorithm to support deletions, we modify the mapping σ to allow mapping to '0'. To support insertions, we allow σ to map connected query nodes to non-connected graph vertices, and use additional N_{ins} color (as in Section 3.3.1).

Given the new definition of σ, the node term is modified as follows:

$$W^S(i,\sigma) = \delta_d |\{u \in X_i^N : \sigma(u) = 0\}| + \sum_{u \in X_i^N, \sigma(u) \neq 0} h(u, \sigma(u))$$

The edge term is more problematic as it depends on the subset of colors used for insertions, and requires some preprocessing. For a pair of vertices $u, v \in V$ and a set of colors $S \subseteq C - \{c(u), c(v)\}$, we denote by $W_P(u,v,S)$ the maximum weight of a path between u and v that visits the colors in S. Given a set of vertex pairs $R = R(l) = \{(r_1^1, r_2^2), \ldots, (r_l^1, r_l^2)\}$, we define $W_P(R,S)$ as the maximum weight of $|R|$ simple paths between all vertex pairs that visit distinct colors from S:

$$W_P(R,S) = \max_{\substack{S^1, S^2, \ldots S^q \\ \biguplus S^l = S}} \sum_{l=1}^{q} W_P(r_l^1, r_l^2, S^l)$$

In order to compute $W_P(R,S)$ efficiently, we use the following recurrence:

$$W_P(R(l),S) = \max_{S' \subset S} \Big[\; W_P((r_i^1, r_i^2), S') + W_P(R(l-1), S - S') \; \Big]$$

Define $E_i(\sigma)$ as the set of graph vertex pairs that are mapped from non-active edges in super-node i:

$$E_i(\sigma) = \{(u,v) \in E : (u',v') \in E_i^N, \sigma(u') = u, \sigma(v') = v\}$$

The edge term for super-node i under the mapping σ and colors S, is:

$$W^E(i,\sigma,S) = W_P(E_i(\sigma), S)$$

Finally, we modify the main recursion as follows:

$$W(i,\sigma,S) = \max_{\substack{S_1 \uplus S_2 = S,\\ S'_1 \subset S_1,\\ S'_2 \subset S_2,\\ \sigma_1,\sigma_2}} \sum_{j=1}^{2} \Big[\ W(i_j,\sigma_j,S_j - S'_j) + W^S(i_j,\sigma_j) + W^E(i_j,\sigma_j,S'_j)\ \Big]$$

To compute the running time of the preprocessing stage, note that $W_P((u,v),S))$ can be pre-computed for all S in $O(n^2 2^k)$ time. Therefore, $W_P(E_i(\sigma),S)$ can be pre-computed in $2^{O(k)}n^{t+1}$ time, and hence the total running time is $2^{O(k)}n^{t+1}$.

Chapter 4

Clustering Genome-Scale Expression Data

Abstract

Genes with a common function are often hypothesized to have correlated expression levels in mRNA expression data, motivating the development of clustering algorithms for gene expression data-sets. We observe that existing approaches do not scale well for large data-sets, and indeed did not converge for the data-set considered here. We present a novel clustering method TCLUST, that exploits co-connectedness to efficiently cluster large, sparse expression data.

We compare our approach with two existing clustering methods CAST and K-means which have been previously applied to clustering of gene expression data with good performance results. Using a number of metrics, TCLUST is shown to be superior or at least competitive with the other methods, while being much faster. We have applied this clustering algorithm to a genome-scale gene expression data set and used gene set enrichment analysis to discover highly significant biological clusters.

4.1 Introduction

With the current mRNA expression profiling technology, expression levels of tens of thousands of genes across hundreds of conditions are measured simultaneously. Genes with a common function are often hypothesized to have correlated expression levels across different conditions. It is not surprising that clustering algorithms have been intensively studied for analyzing gene expression data in order to detect the groups of genes with correlated expression patterns. However, current clustering algorithms commonly used in the domain of gene expression, do not scale well for genome-scale expression data.

In the context of gene expression data, it is convenient to think of clustering on a graph in which the vertices correspond to genes (or, probe sets), and edge weights reflect the similarity/correlation between the expression profiles of the probe sets. Generally, the correlation graph is a complete graph with $n \sim 10^5$ nodes, and $m \sim 10^9$ edges. It is a common strategy to sparsify the graph by discarding the edges with edge weights smaller than a chosen threshold. In our experiments, we observe that even after filtering with a reasonable threshold, the number of edges remain $m \sim 10^7$, resulting in a graph of complexity $mn \sim 10^{11}$. Therefore, it is critical to devise a clustering algorithm that will scale well with very large graphs.

In this paper, we propose a fast method to identify the dense subgraphs in correlation graph defined on genes (or, probe sets). If a cluster of functionally related genes were all co-expressed, and those were the only co-expressed genes, the correlation graph should look like a collection of disjoint cliques. Errors and other biological variation will add additional edges and remove some true edges. To simplify exposition, we will consider these extra and missing edges as erroneous data (False positives (FP), and negatives (FN), respectively), even though they might be encoding a true biological phenomenon. The resulting graph is therefore a sparse graph with 'dense subgraphs' embodying functionally related clusters.

Assuming the graph has an underlying clique structure, identification of such dense-subgraphs is known as 'cluster editing problem' in literature. The problem has been proven to be NP-hard for arbitrary FP rates [58, 72], even when FN=0 (The clique problem; [59]). In practice it is hard for even moderate error rates. Fortunately, in the specific domain of gene-expression, the error rates appear to be low enough for the approach to be viable.

Among the more popular algorithms for clustering gene expression data are K-means, SOM, hierarchical clustering, and CAST [54, 62, 69]. Previous studies comparing traditional clustering methods in microarray data have shown that K-means and SOM have superior performance to hierarchical clustering [63]. CAST has also been applied on expression data with good results [54]. However, in our experiments, they do not appear to scale well for large expression data-sets, as discussed later in the text. Both K-means and CAST are similar in one sense as they both dynamically update clusters by assigning and removing vertices iteratively, according to a stated objective. In K-means, the objective is to reduce the intra-cluster variation, while increasing the inter-cluster variation. However, the number of clusters (K) is an important parameter, and must be specified in advance.

CAST updates clusters with no prior knowledge of the number and size of the clusters. It constructs one cluster at a time by iteratively examining each vertexs relationship to an open, nonstabilized, cluster. CAST then uses affinity of a vertex to the cluster to determine whether the vertex belongs or not. Affinity of a vertex v to an open cluster C is measured by the sum of the edge weights going from v to the current members of C. CAST alternates between adding and removing vertices until all members but not non-members of C have high affinity to C. The key parameter here is *affinity* which determines the number and sizes of the clusters. For very large data-sets, it takes a lot of time as it will take many iterations for the vertex$\leftrightarrow$cluster relationship to stabilize.

The weighted cluster editing problem (defined below) has also been directly studied by Rahmann et al. [70]. The authors define a cost function, and describe a fixed-parameter algorithm which reaches its limit above 50 vertices. They also suggest an alternative fast, $O(n^2)$, layout-based heuristic for large graphs. While faster, it is still computationally intensive. In self reported results, the time increased exponentially, requiring $\sim 10^5$ s on graphs of complexity of 10^8. Additionally, it requires setting of upto 5 input parameters.

We observe that existing approaches do not scale well for large data-sets, and indeed did not converge for the data-set considered here. In our experiments, we use microarray data acquired from 11 tissues in each of 29 inbred strains of mice Affymetrix MOE430v2 GeneChips (45101 probe sets). This data set represents a 'genetical genomics' or 'eQTL' microarray experiment [71]. While the vertex-set is large, it is sparse in the edges. Of the $\sim 10^9$ possible edges, 4.2×10^7 exceed a correlation coefficient of 0.3, resulting in graphs of complexity 10^{11}.

4.2 Weighted Cluster Editing Problem

Unweighted cluster editing problem is to make the fewest number of changes to the edge set of an input graph such that the resulting graph is a disjoint union of cliques. In this paper, we address 'weighted cluster editing problem' in the context of gene expression data which also takes the edge weights into account.

4.2.1 Definitions and Notation

Following Rahmann et al. [70], we consider a set of genes V and a symmetric function $s : V^2 \to R^+$ that reflects the similarity between the expression profiles of the genes in V. Given a weighted undirected correlation graph $G = (V, E)$ where $E = \{(u, v) | s(u, v) > 0\}$, our goal is to edit G by removing and adding edges in such a way that it becomes a union of disjoint cliques where each clique corresponds to subsets of genes with highly correlated expression profiles.

Each operation incurs a nonnegative cost: If $(u, v) \in E$, the edge removal cost of (u,v) is $c^- = s(u, v)$. If $(u, v) \notin E$, the edge addition cost of (u,v) is $c^+ = -s(u, v)$. Note that for similar vertex pairs u and v, removal of (u, v) will incur a non-negative cost. Likewise, for distant u and v pairs, addition of (u, v) will incur a non-negative cost.

Consequently, the cost to transform the initial graph $G = (V, E)$ into a graph $G' = (V, E')$ is defined as

$$cost(G \to G') = \sum_{(u,v) \in E \setminus E'} c^-(u, v) + \sum_{(u,v) \in E' \setminus E} c^+(u, v).$$

4.2.2 Problem Statement

Given a similarity function $s : V^2 \to R^+$ and a weighted undirected graph $G = (V, E, s)$, find a union of cliques graph G^* such that $cost(G \to G^*) = \min\{cost(G \to G'|G'\text{is a union of disjoint cliques})$.

In our method to tackle this problem, we do not explicitly use the cost function, but use it as an evaluation criteria for our simulation results. We show that the output of our algorithm minimizes the cost function when the error rate is low, and its corresponding cost is close to the optimal when the error rate is high. (See Section 4.5.3, Figure 4.5). As noted earlier, algorithms with explicit theoretical guarantee on performance are intractable for the large data-sets. Further, assuming an underlying 'corrupted-clique' model to explain real data, the method is guaranteed (Section 4.3) to improve in each iteration. This provides a theoretical foundation for its performance on real data.

4.3 Co-connectedness Based Heuristic

In this section, we propose a heuristic algorithm based co-connectedness to tackle weighted cluster editing problem. Co-connectedness is described as the fraction of neighbors shared by the pair and has been used in many different graph-theoretical problems [65, 73]. To illustrate using an example, consider Figure 4.1. The underlying clique structure of the input graph has two independent cliques formed by the vertices $1, 2, 3, 4, 5, 6$ and $7, 8, 9, 10, 11$. The observed graph has some noise as some of the edges within the cliques are missing and there are some edges between the cliques.

For any vertex u, define the neighborhood vector $\vec{u}$ as the bit-vector describing the set of neighbors. For example, $\vec{\mathbf{1}} = [1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0]$. We use the Tanimoto coefficient (TC) to describe co-connectedness of two vertices u and v as

$$TC(u, v) = \frac{\vec{u} \cdot \vec{v}}{\vec{u} \cdot \vec{u} + \vec{v} \cdot \vec{v} - \vec{u} \cdot \vec{v}} \tag{4.1}$$

Note that the TC here is identical to the *Jaccard Coefficient* ($JC(u, v) = \frac{|N_u \cap N_v|}{|N_u \cup N_v|}$), but also works for weighted graphs (Section 4.4.2). It measures not only if two vertices are

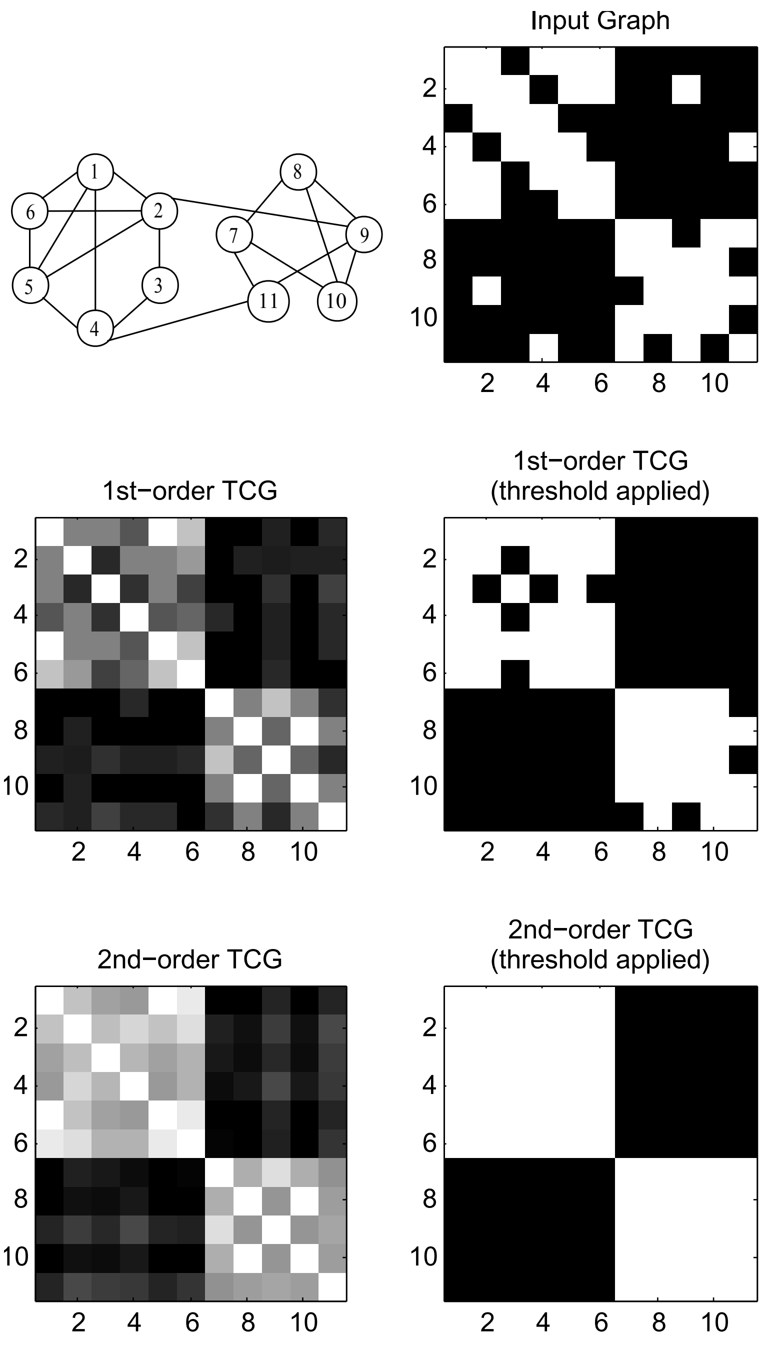

Input Graph
1st−order TCG
1st−order TCG (threshold applied)
2nd−order TCG
2nd−order TCG (threshold applied)

Figure 4.1: Illustration of co-connectedness based heuristic for an unweighted graph. Graph and its adjacency matrix as an image are shown in the top panel. Color scale is from 0 (black) to 1 (white). In the second and third panel, TCG^1 and TCG^2 are shown, respectively. The threshold-applied and binarized version of each is also shown to indicate that TCG get closer to the underlying clique structure in each iteration.

connected to the similar set of vertices but also if they are connected with similar edge weights.

In Figure 4.1, for visualization purposes, we display the adjacency matrices of the graphs as images. Observe that for a spurious edge $(2, 9)$, TC$(2, 9)$ is low, while for the missing edge $(2, 4)$, TC$(2, 4)$ is high. Therefore, computing TC, and applying a threshold, we get a new sparse graph, TC graph, with fewer errors. Within two iterations, the graph reverts to a collection of two cliques. Note that TC of two vertices does not have direct interpretation of their original edge weight. It is possible for two vertices with high edge weight to have low TC and for two vertices with low edge weight to have high TC. We treat a high edge weight as FP if measured TC is low. Similarly, if the measured TC is high, we treat a low edge weight as FN.

We can generalize this idea and show calculations that help to show convergence. Following Ben-Dor et al. [54], define a (d, α)-*corrupted clique graph* as a collection of disjoint cliques, each of size d, in which intra-clique edges are removed independently with probability α', and inter-clique edges are independently added with probability α' for an arbitrary $\alpha' \in [0, \alpha]$.

Let G be a (d, α)-corrupted-clique graph. If

$$\alpha < \min\left\{\frac{\ln d}{3d}, \frac{d}{n}\right\},$$

we can show that there exists a *tcg-threshold* such that the TC graph (TCG) is a (d, α')-corrupted-clique graph with $\alpha' < \alpha$. Note that if this is true then in each iteration, we will converge towards the underlying clique structure. In practice, we only need to do this for a few iterations and then output the connected components.

Consider two vertices u, v from the same clique. Then, assuming that edges in the clique are missing with probability α

$$E(\vec{u} \cdot \vec{v}) \geq (1 - \alpha)^2 d$$

The inequality is because some spurious edges may add to the dot-product. Note that the dot-product can be computed as the sum of d independent binary variables, each of which is 1 with probability $(1 - \alpha)^2$. Consequently, we can use Chernoff's bound (e.g. [67]) to compute deviations from the mean as

$$Pr[\vec{u} \cdot \vec{v} < (1 - \delta)(1 - \alpha)^2 d] \leq e^{-\delta^2 (1-\alpha)^2 d/2} \tag{4.2}$$

for any $\delta > 0$. Correspondingly, for vertices u, v from different cliques

$$E(\vec{u} \cdot \vec{v}) = 2\alpha(1-\alpha)d + \alpha^2(n-2d) \leq 3\alpha d$$

and, an equality similar to Equation 4.2 holds

$$Pr[\vec{u} \cdot \vec{v} > (1+\delta)3\alpha d] \leq e^{-\delta^2 3\alpha d/4} \tag{4.3}$$

Finally, we have

$$E(\vec{u} \cdot \vec{u}) = (1-\alpha)d + \alpha(n-d)$$

The condition $\alpha \leq d/n$ implies that

$$(1-\alpha)d \leq E(\vec{u} \cdot \vec{u}) \leq (2-\alpha)d$$

and, with high probability

$$(1-\delta)(1-\alpha)d \leq \vec{u} \cdot \vec{u} \leq (1+\delta)(2-\alpha)d$$

Finally, consider the expression $\vec{u} \cdot \vec{u} + \vec{v} \cdot \vec{v} - \vec{u} \cdot \vec{v}$. Using Cauchy-Schwartz inequality $\vec{u} \cdot \vec{v} \leq \max\{\vec{u} \cdot \vec{u}, \vec{v} \cdot \vec{v}\}$. Therefore, with high probability

$$(1-\delta)(1-\alpha)d \leq \vec{u} \cdot \vec{u} + \vec{v} \cdot \vec{v} - \vec{u} \cdot \vec{v} \leq 2(1+\delta)(2-\alpha)d$$

Thus, if u, v are in the same clique, with high probability

$$\begin{aligned} TC(u,v) &\geq \frac{\vec{u} \cdot \vec{v}}{\vec{u} \cdot \vec{u} + \vec{v} \cdot \vec{v}} \\ &\geq \frac{(1-\delta)(1-\alpha)^2}{2(1+\delta)(2-\alpha)} \end{aligned}$$

If u, v are in different cliques, then with high probability

$$TC(u,v) \leq \frac{(1+\delta)3\alpha}{(1-\delta)(1-\alpha)}$$

If we can choose a threshold in between these two numbers, then with high probability, we will get rid of the spurious edges, and add missing edges. This imposes the following condition on δ

$$\frac{(1+\delta)3\alpha}{(1-\delta)(1-\alpha)} \leq \frac{(1-\delta)(1-\alpha)^2}{2(1+\delta)(2-\alpha)}$$

implying

$$\frac{(1+\delta)^2}{(1-\delta)^2} \leq \frac{(1-\alpha)^3}{6\alpha(2-\alpha)}$$

or

$$\delta \geq \frac{(1-\alpha)^{3/2} - (6\alpha(2-\alpha))^{1/2}}{(1-\alpha)^{3/2}(6\alpha(2-\alpha))^{1/2}} \tag{4.4}$$

Finally, we require that the probability that a spurious edge remains in the TCG, or a true edge is missing is lower than α. This is bounded using Equations 4.2,4.3, to be

$$4e^{-\delta^2 3\alpha d/4} < \alpha$$

implying

$$\delta^2 < \frac{4\ln(4/\alpha)}{3d\alpha} \tag{4.5}$$

Equations 4.4 and 4.5 are satisfied when $\alpha < \frac{\ln d}{3d}$, implying that error is reduced in a single iteration of TCG generation. In practice, our approach works well for much higher error rates ($\alpha < 0.5$), with results comparable to CAST, but extending to larger data-sets.

4.3.1 Running Time Analysis

If the average degree of vertex is d, computation of TC for a pair of vertices costs $O(d)$. In each iteration of TCG generation, we compute TC for only pairs with distance at most 2, since otherwise TC will be zero. Thus, the running time complexity of the algorithm is $O(|V|d^3)$ per iteration and scales linearly with the number of iterations.

The number of iterations, k, depends on the size of the underlying cliques and the error rate. At a fixed error rate the neighborhood similarity for a pair of vertices in a small clique is more influenced than a pair of vertices in a large clique. Thus, it takes more iterations for smaller cliques to edit all FP and FN edges. In our experiments, at error rate $\epsilon = 0.2$, in a corrupted graph of 640 vertices, we recover the cliques of size 128 in TCG^1, while we recover the cliques of size 16 in TCG^3. (See Figure 4.4).

4.4 TCLUST

Clustering of large gene expression data-sets is challenging. Our clustering method TCLUST is based on two ideas. The first is an assumption that the expression clusters resemble corrupted cliques, and co-connectedness can be used for clustering. This of-course cannot be proved. However, we evaluate our clustering results using independent biologically meaningful metrics, and show that whether the underlying model is true or not, we can use co-connectedness.

The second idea, motivated in part by Gibson et al. [64], is based on fingerprinting which suggests that co-connectedness can be exploited looking only at a subset of genes to which a particular gene is highly correlated. As this subset is small, our algorithm can comfortably handle large input sizes. Any loss of performance is handled by repeated iterations of the co-connectedness heuristic from Section 4.3.

Figure 6.2 provides an overview of TCLUST. A fingerprinting strategy is used to generate a correlation coefficient graph(CCG) from gene expression data. This is followed by iterative computation of TC graphs (TCG) until connected components of the TCG are clique-like. Note that once a connected component is dense enough, it will only get denser in subsequent iterations and no two disjoint connected components will be merged. Thus, it is possible to output a connected component as a cluster at any iteration as its edge density, number of edges/number of pairs exceeds a threshold.

While CAST and TCLUST both have similar goal of identifying underlying cliques, TCLUST should converge faster as vertex pairs with strong (respectively, weak) co-connectedness are quickly identified as such, and remain as edges (respectively, non-edges) for the remainder of the iterations. Specifically, if the two vertices end up in different connected components, their TC will always be 0, and they will never converge into one component. This implies that we only need to compute TC for vertex pairs in the same connected component, and a few iterations should suffice to establish the clusters.

We apply our method on a mouse gene expression dataset with 45101 probesets and 295 samples. (See Appendix for details.) In our experiments we show that TCLUST is very efficient while it is still able detect biologically meaningful gene groups.

In the following subsections, we discuss the different steps spelled out in the flowchart in detail. In Section 4.5, we demonstrate the performance of our algorithm on simulated data. In Section 4.6, we compare performance of TCLUST, CAST and K-means on filtered expression data. Finally, in Section 4.6.3, we apply TCLUST to our

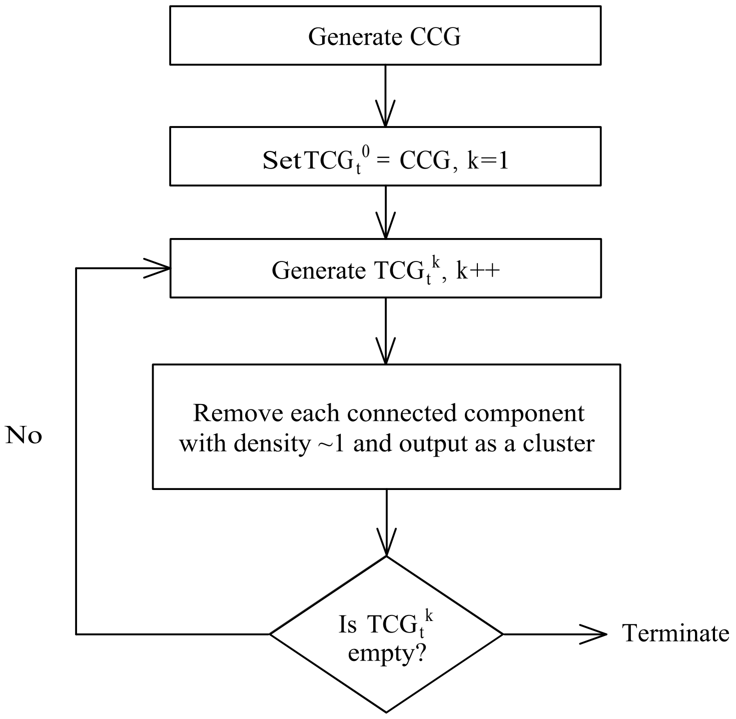


Figure 4.2: Flow chart for TCLUST algorithm.

large gene expression data set for gene set enrichment analysis.

4.4.1 Generating Correlation Coefficient Graph (CCG)

We use the Pearson's correlation coefficient as a measure of co-expression of genes/probesets. Let x_{ik} denote the expression level of p_i in the kth sample. For a pair of probes p_i, p_j

$$s(p_i, p_j) = \frac{\sum_{k=1}^{n}(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)}{\sqrt{\sum_{k=1}^{n}(x_{ik} - \bar{x}_i)^2 \sum_{k=1}^{n}(x_{jk} - \bar{x}_j)^2}}$$

Clearly $s(p_i, p_j) = s(p_j, p_i)$. Define *CCG* as a complete undirected, weighted graph (V, E, s), with s defining edge weights. We use a threshold on the weights, only including edges that exceed the threshold. Other results have suggested that rank based correlations are more robust to noise. However, in our own experiments, the data-set was of sufficiently high quality that we continued with Pearson's correlation, which was required by our collaborators on the project.

It is a common strategy to reduce the complexity of CCG by discarding the edges with edge weights smaller than a chosen threshold. In our experiments, we observe that correlation coefficients of a probe-set with the rest of the probe-sets follow a unimodal distribution with a small tail. (See Figure 4.3 for the distribution for 4 randomly chosen probe-sets.) However, the shape of the distribution differ from probe-set to probe-set. This makes it difficult to set a fixed threshold that would not be biased for any probe-sets.

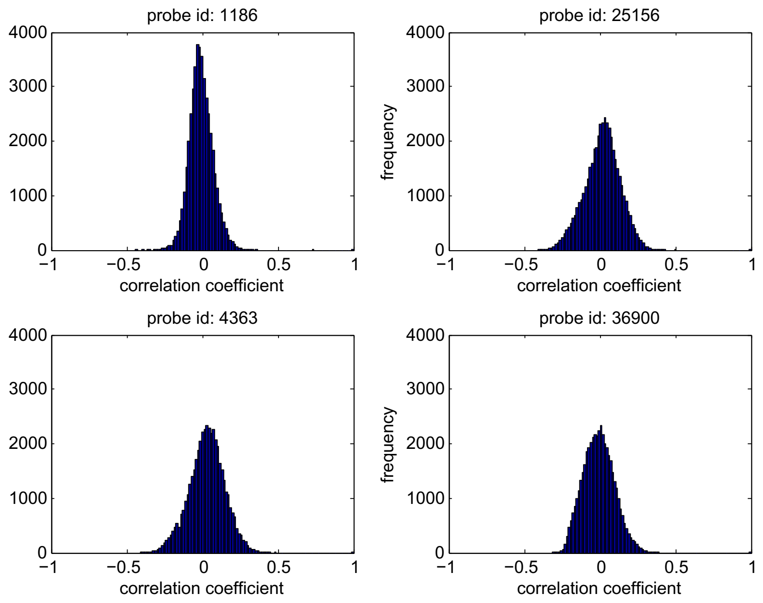


Figure 4.3: Distribution of correlation coefficients with the rest of $45K$ probe-sets for 4 randomly chosen probe-sets.

We use a novel *per-vertex-threshold* scheme to determine edges. First, define the *top-neighbors* of vertex p_i, $N(p_i)$ as follows: let μ_i and σ_i be the mean and standard deviation of the correlation coefficients between vertex p_i and all other vertices in V, respectively. Denote $p_j \in N_\theta(p_i)$ if and only if

$$\mathrm{s}(p_i, p_j) \geq \mu_i + \theta\sigma_i$$

Define CCG$_\theta = (V, E_\theta, s)$, where V is the set of probe-sets, and

$$E_\theta = \{(p_i, p_j) | p_i \in N_\theta(p_j), p_j \in N_\theta(p_i)\}$$

Per-vertex thresholding can be thought as a fingerprinting scheme where each vertex/probe set is fingerprinted with the set of its top neighbors. It keeps the graph sparse, and also controls for certain vertices (genes) having higher overall expression values compared to other genes. We use CCG$_3$ (determined empirically) as the input to TCLUST. However, note that no assumption is made w.r.t the distribution of scores of correlated genes.

We reiterate that the sparsification by edge and node thresholding is provided as an option which allows us to handle large graphs. It is neither required, nor assumed that the input is sparse. Indeed while evaluating cluster quality, we compare against the original (complete) graph. It is possible that the sparsification could reduce the quality of our clusters. However, the co-connectedness property is mostly preserved by sparsification, and our results show that there is no loss of quality by introducing this.

4.4.2 Generating Tanimoto Coefficient Graph (TCG)

The Tanimoto coefficient (TC) is a measure of similarity between two real-valued vectors, defined by Equation 4.1. For a weighted graph, $G = (V, E, w)$, denote

$$\vec{W}_i = [w(p_i, p_j) : p_j \in V]$$

as the vector of edge weights incident on vertex p_i. We define the TC between a pair of vertices (p_i, p_j) as

$$TC(p_i, p_j) = TC(\vec{W}_i, \vec{W}_j)$$

This reweighing of edges allows us to define a family of *Tanimoto Coefficient Graphs* TCG$_t^k$ for arbitrary $k \in Z^+$, and a real valued tcg-threshold t. Specifically, TCG$_t^0 = (V, E, w)$ and $\vec{W}_i^0 = \vec{W}_i$.

For $k > 0$, TCG$_t^k = (V, E^k, w^k)$, where

$$E^k = \left\{(p_i, p_j) | \text{TC}(\vec{W}_i^{k-1}, \vec{W}_j^{k-1}) \geq t, \forall p_i, p_j \in V\right\}$$

$$w^k(p_i, p_j) = \begin{cases} \text{TC}(\vec{W}_i^{k-1}, \vec{W}_j^{k-1}), & (p_i, p_j) \in E^k, \\ 0, & \text{otherwise} \end{cases}$$

As we iterate over TCG^k for $k = 0, 1, \ldots$, the connected components should resemble the underlying clique structure. Therefore, we output the connected components as the final clusters.

4.4.3 Implementation of TCLUST

We implemented the core algorithm in C++. The input to the program is the CC graph, a tcg-threshold $t \in R$, and $k \in Z^+$, the number of iterations (See Figure 6.2).

We use R [68] statistics package to generate the CCG for input to TCLUST. We compute the correlation matrix and filter the matrix using per-vertex thresholding technique.

4.5 Results on Simulated Data

4.5.1 Corrupted Random Weighted Graph Model (CRWG)

Ben-Dor et al. [54] proposed a natural corrupted random graph model to test the performance of CAST in retrieving the underlying clique structure from 'corrupted' graphs.

We extend their model to include weighted graphs. The Corrupted Random Weighted Graph (CRWG) model has 3 input parameters (S, T, ε) defined as follows: S is the underlying clique structure denoted by the size of the underlying cliques. As an example $S = [2 \times 8, 2 \times 16, 1 \times 32]$ denotes a structure consisting of 2 cliques of size 8, 2 cliques of size of 16 and one clique of size 32. T is the intra-clique threshold, i.e. any edge weight within a clique is larger than T and any edge weight between cliques is less than or equal to T. Finally, ε is the error range of the observed edge weights.

In the ideal case, we assume a weighted complete graph with an underlying clique structure where $\varepsilon = 0$. Therefore, when a threshold T is applied, we can retrieve the cliques. In CRWG model, this ideal case is corrupted. Each edge weight w is replaced by a random weight uniformly distributed in the range $[w - \varepsilon, w + \varepsilon]$. Thus, when the threshold T is applied on the observed graph, an inter-clique edge may appear since $(w + \varepsilon)$ may exceed t, and an intra-clique edge may not appear since $(w - \varepsilon)$ may be smaller than T. The effect of changing weight is the weighted analog of adding spurious edges, or deleting true edges in the corrupted-clique model. An edge with actual weight $T < w \leq T+\epsilon$ is not in the thresholded observed graph G with probability $\frac{T-w+\epsilon}{2\epsilon} \in [0, 0.5)$. Similarly, an edge with actual weight $T - \epsilon < w < T$ is in G with probability $\frac{w+\epsilon-T}{2\epsilon} \in (0, 0.5)$.

The output of TCLUST is a set of clusters. In a corrupted graph model, it is reasonable to think of each cluster as a complete subgraph. However, data sets may only have an underlying dense subgraphs which are not complete. Therefore, we can use either the TCG^k itself as the output or we can use the collection of cliques induced by the vertices of each connected component of TCG^k as the output. We test both of these outputs in the following sections. In the second case, we also compare against a version of CAST, reimplemented for the weighted case.

4.5.2 Recovering Underlying Structure by TCG

To test performance of iterative computation of TCG in recovering the underlying cliques, we generate 20 random graphs using CWRG model with $S = [16 \times 8, 8 \times 16, 4 \times 32, 2 \times 64, 1 \times 128]$, $T = 0.6$ and $0.1 \leq \varepsilon \leq 0.5$. Note that each graph has 31 underlying cliques of sizes $8, 16, 32, 64, 128$ and the number of vertices participating in different size cliques are the same. The input graph G to our clustering algorithm is the threshold graph that is obtained by filtering the edges with weights smaller than $T = 0.6$. Thus, G has a number of extra and missing edges depending on the error rate ϵ.

If an edge in TCG is indeed between two clique members, it is considered as a TP, otherwise a FP. TN and FN are defined accordingly. We use two similarity measures: *sensitivity* and *specificity*. We define sensitivity as $TP/(FN + TP)$ and specificity as $TN/(FP + TN)$.

Figure 4.4 shows our simulation results for the comparison of underlying clique structure S with filtered TCG^1, TCG^2 and TCG^3 by applying tcg-threshold. The tcg-threshold is varied between 0.1 and 0.9 to select the setting with better sensitivity and specificity results. In Figure 4.4, the results with only the best settings are shown.

The results show us that iteratively generating TCG allows us to get closer to the underlying clique structure S. For example, in the case where $\varepsilon = 0.2$, sensitivity and specificity values for G are improved from $(87\%, 92\%)$ to $(99\%, 99\%)$ by TCG^1 and

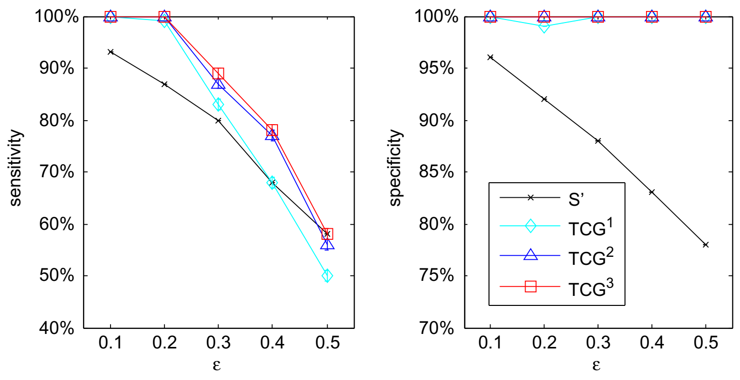


Figure 4.4: Comparison of underlying clique structure and filtered TCG^1, TCG^2 and TCG^3. Average cost, sensitivity and specificity are plotted with error bars. Random graphs are generated by CRWG model with parameters S=[16×8, 8×16, 4×32, 2×64, 1×128], $T = 0.6$ and ε ranging from 0.1 to 0.5. TCG^k is the kth order TCG computed from S' iteratively. Sensitivity = $TP/(FN+TP)$ and specificity = $TN/(FP+TN)$ are computed according to S where TP, FP, TN and FN are defined based on the existence of edges after applying the tcg-threshold on the generated TCG. Tcg-threshold is set to its optimal values among $[0.1, 0.2, \cdots, 0.9]$.

to $(100\%, 100\%)$ by TCG^2 revealing S exactly. In each case, we improve significantly upon the corrupted graph.

4.5.3 CAST versus TCLUST

In this section, we compare the performance of TCLUST with CAST [54]. As the code of CAST was not available, and we need a version that works for weighted graphs, we implemented TCLUST and CAST algorithms in C++ using the same framework. While there is the real caveat of comparing our own implementation of CAST, we took care to maintain the fidelity of the approach. Indeed, the project started out by attempting to use CAST to cluster the data-sets. However, in the interest of fairness, the results below are best interpreted by considering the CAST running times generically. Both return a collection of cliques. We evaluate the quality of the output clique graphs by comparing their associated cost from the input graph with the cost of obtaining the underlying clique structure S.

We compare the results for CAST and TCLUST of degree $k = 2, 3$. Both CAST and TCLUST take a parameter as input: affinity-threshold and tcg-threshold, respectively. We vary both parameters from 0.1 to 0.9. In Figure 4.5, we show the simulation results of CAST and TCLUST with only the setting which is optimal in terms of cost.

This simulation show us that both CAST and TCLUST closely approximates the cost of the underlying clique structure even with high error rate. However, TCLUST has better sensitivity and specificity results with a much lower standard deviation at each error rate. Note that each connected component is treated as a complete graph in this test. This changes the FP, and FN rate, and so the results in Figure 4.4 and Figure 4.5 are not directly comparable.

4.6 Comparison of Different Clustering Methods on Microarray Expression Data

We compare TCLUST with two existing methods: K-means and CAST on microarray data. We explore the performance of these methods over a wide range of their parameters (K, affinity-threshold, tcg-threshold). We limit the data set to 5000 probe sets with the highest variation across samples, as K-means and CAST did not converge for larger sizes. In the next section, we will present results of TCLUST on the complete dataset ($45K$ probes).

We use a weighted version of CAST for comparison [54]. We applied a pre-filter to the pairwise similarity matrix, discarding all edges less than 0.3. In the absence of pre-filter, CAST did not converge and output only five clusters of almost equal sizes, none of which were functionally enriched.

4.6.1 Functional Enrichment

In clustering genes according to the expression data, a common goal is to cluster functionally related genes, and we use this as a test for the three methods. We extracted

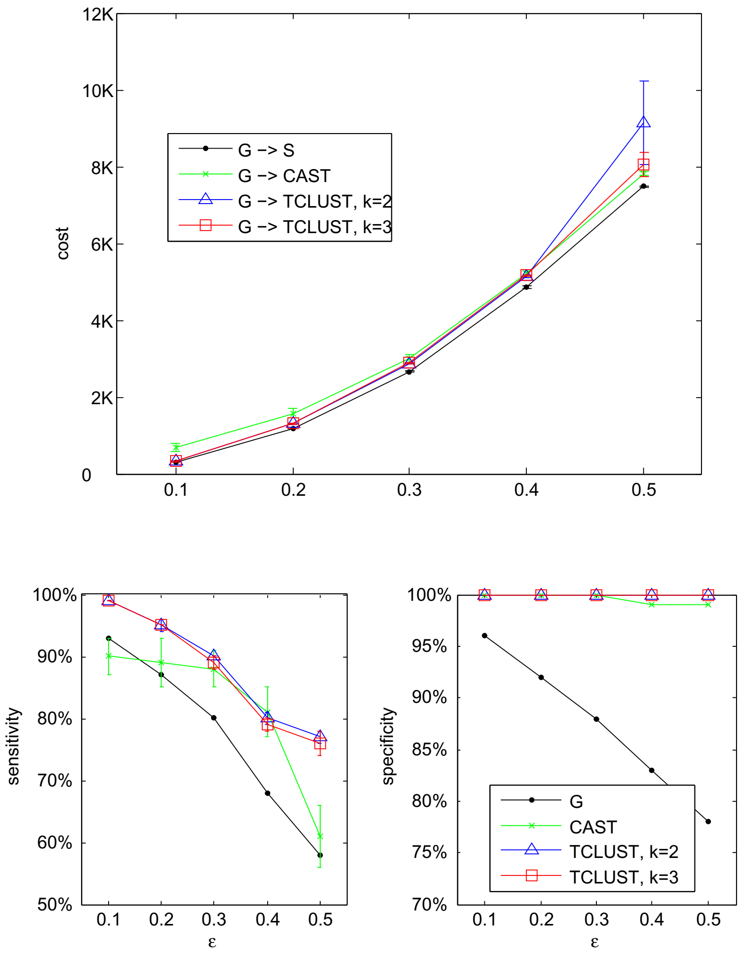


Figure 4.5: Comparison of CAST and TCLUST with $k = 2, 3$. Random graphs are generated by CRWG model with parameters S=$[16 \times 8$,8×16, 4×32, 2×64, $1 \times 128]$, $T = 0.6$ and ε ranging from 0.1 to 0.5. Sensitivity = $TN/(FP + TN)$ and specificity = $TP/(FN + TP)$ are computed according to S. While computing sensitivity and specificity, clusters obtained by the method are treated as a clique. Affinity-threshold and tcg-threshold are set to 0.5 and 0.2 respectively, which are their best setting among the values $[0.1, 0.2, \cdots, 0.9]$.

all *Functional Gene Sets* (FGS) of size ≤ 500 from 6 different databases: KEGG pathways database (KEGG) [66], Ingenuity pathways database (ING) [74], Gene Ontology (GO) database [53] in categories cellular component (CC), molecular function (MF), and biological process (BP) and mouse phenome database (MPD) [55]. The chosen 5000 probe sets map to 3776 genes, of which 23.36% are annotated in KEGG, 21.21% in ING, 31.89% in CC, 64.27% in MF and 58.16% in BP and 30.27% in MPD.

The first step is to decide if a predicted cluster C is functionally enriched in an FGS F. We compute a p-value for enrichment of F in C, using a hypergeometric test [75]. For details, see Supplemental Data Section 4.8.3. We set the significance threshold to $p \leq 0.001$. Also, we say that a cluster is functionally enriched if it is significant for at least one FGS.

As the parameters for clustering are changed, we get differing number/sizes of clusters. Larger cluster sizes include more genes, but are less likely to be enriched in a single FGS. Therefore, we use *gene-coverage*–defined as the fraction of genes in functionally enriched clusters–to compare the different methods. However, some of the functional databases may not include all genes. Therefore, a related measure is *db-coverage*, defined as the fraction of genes annotated in the specific database that end up in a functionally enriched cluster.

In Figure 4.6, we compare TCLUST, CAST and K-means in terms of gene-coverage and db-coverage. For each clustering method, we show 7 different parameter settings exploring a wide range. The *tcg*-threshold for TCLUST varies from 0.2 to 0.8, affinity threshold for CAST varies from 0.3 to 0.9, and the number of clusters for K-means varies from 200 to 3000 so that all three methods give similar ranges for the number of clusters. Each choice of a method, and a parameter setting is plotted according to its gene, and db-coverage. The best parameter settings for all three methods are highlighted by larger data points and labels in the plots.

The results show that TCLUST does generally better than CAST and K-means. The best setting for TCLUST has better gene, and db-coverage compared to best setting of CAST and K-means, and also many other settings show high coverage. Note that the gene coverage in the functionally enriched clusters is limited to 30% for all methods because of incomplete gene annotation.

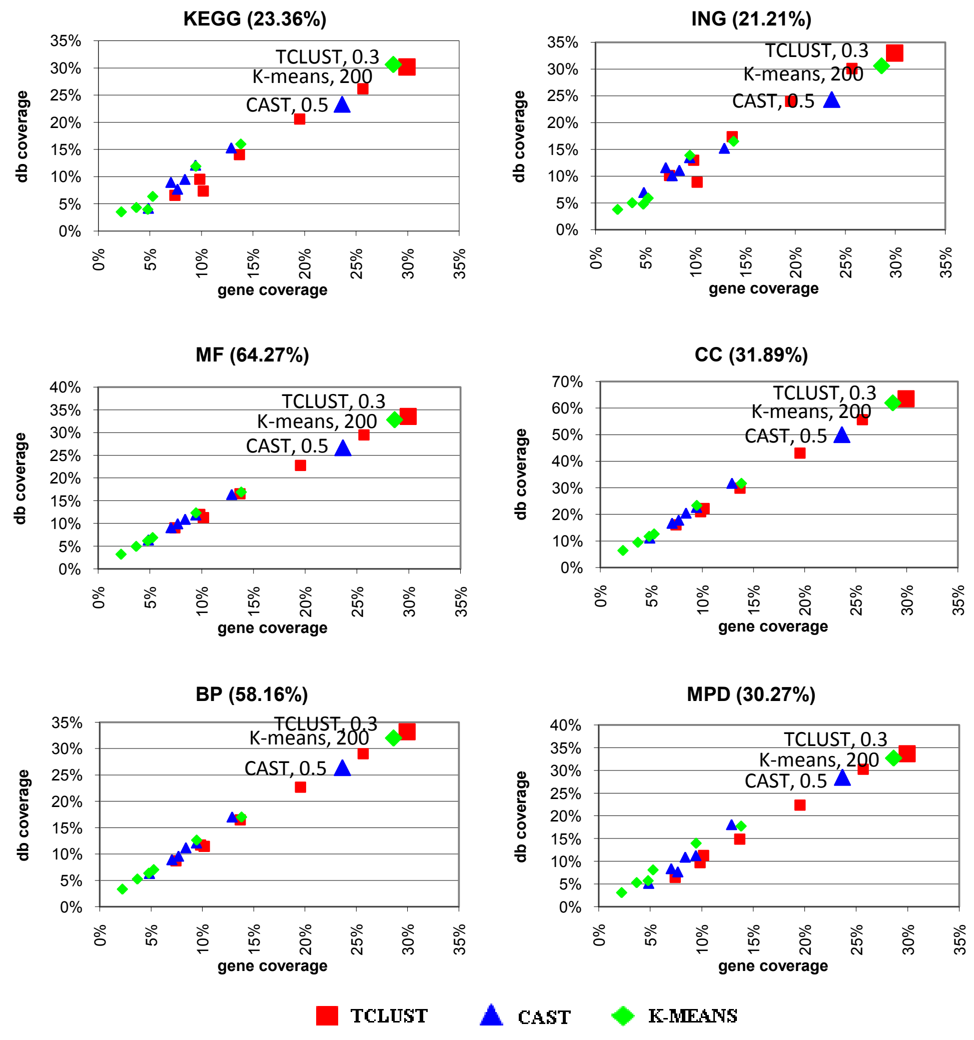


Figure 4.6: Comparison of TCLUST, CAST and K-means in terms of gene and database coverage in the functionally enriched clusters. For each clustering method, 7 different parameter settings are shown. The number of clusters for K-means varies from 200 to 3000, affinity-threshold for CAST varies from 0.3 to 0.9 and tcg-threshold for TCLUST varies from 0.2 to 0.8. The best settings for TCLUST, CAST and K-means (tcg-threshold=0.3, *aff-threshold*=0.5 and $K = 200$) are highlighted by larger data points in the plots.

We use a False Discovery Rate (FDR) approach to evaluate the significance of the number of functionally enriched clusters obtained by the clustering methods. We picked the best parameter setting that achieves the highest database coverage for each method. We computed many random permutations of gene annotation labels. Each method/parameter setting pair is applied to the randomized data in which the number and the size distribution of the clusters remain the same. Any enriched cluster discovered in a randomized data is a false positive. The ratio of the average number of false positives to the actual number of functionally enriched clusters describe the FDR.

In Table 4.1, we give thw db-coverage, number of functionally enriched clusters and the associated FDR results for the clusterings obtained by TCLUST, CAST and K-means with their best parameter settings. As we see, FDR for TCLUST is substantially lower than the FDR for CAST and K-means while it achieves better db-coverage and higher number of functionally enriched clusters. This observation suggests that larger database coverage of TCLUST is not due to the size distribution or number of the clusters.

Table 4.1: False discovery rates associated with the number of functionally enriched clusters (#fec) for TCLUST, CAST and K-means. The parameter setting which is best in terms of gene and database coverage is used for each of the clustering method. At high gene and database coverage, TCLUST still has lower FDR.

Database	K-means			CAST			TCLUST		
	db-coverage	#fec	FDR	db-coverage	#fec	FDR	db-coverage	#fec	FDR
KEGG	30.61%	2	78.6%	23.36%	13	6.2%	30.16%	13	5.9%
ING	30.59%	2	59.4%	24.34%	8	8.9%	32.96%	8	8.8%
CC	61.96%	8	36.6%	50.08%	9	15.5%	63.46%	13	8.3%
MF	32.80%	11	51.5%	26.70%	17	14.4%	33.54%	24	7.6%
BP	31.97%	12	40.0%	26.41%	23	12.6%	33.15%	26	8.6%
MPD	32.72%	10	40.6%	28.52%	11	25.9%	33.68%	14	14.9%

4.6.2 Timing

In this section, we compare K-means, CAST and TCLUST with $k = 0, 1, 2$ in terms of timing for different number of probe sets to cluster. (See Table 4.2.) K-means takes the whole similarity matrix as input while TCLUST and CAST take an input graph using similarities. The timing for generation of the similarity matrix and the input graphs are excluded and only the time required for the algorithm itself are given. For TCLUST, we give three timing results varying the degree k of the algorithm from 0 to 2. We give three timing results for also CAST, running on the complete, filtered at a cut-off and per-node thresholded similarity graphs. Originally CAST takes the whole similarity matrix as input. However, as the graph gets larger, CAST becomes intractable. Thus, we assess also the timing for CAST after applying a cut-off=0.3 on the similarity matrix. In order to clarify gain in running time by per-node thresholding, we also report running time of CAST on the per-node thresholded similarity graph.

The number of probe sets is varied from 100 to $10K$ by selecting the probe sets with the highest variation across the samples. The timing table was extended until $N = 45K$ for TCLUST only as it is not feasible to run K-means and CAST on large data sets. The parameters for the methods are chosen as the setting that performed best on the 5000 probe sets in terms of gene and database coverage in section 4.6.1(aff. threshold=0.5, tcg-threshold=0.3, K = $N/25$). The programs were run on machine with 32GB memory and a single 2.2GHz AMD Opteron Processor.

Results clearly indicate the speed advantage of TCLUST.

Table 4.2: Timing results in seconds as a function of N, the number of input probe sets to cluster, for TCLUST, CAST and K-means. The parameters for the methods are chosen as the setting that performed best on the 5000 probe sets in terms of gene and database coverage (aff. threshold=0.5, tcg-threshold=0.3, K = $N/25$). We give two timing results for CAST with and without a cut-off. Originally CAST takes the whole similarity matrix as input. We also assess the time by applying a cut-off=0.3 on the similarity matrix since as the graph gets larger CAST becomes intractable. For TCLUST, we give three timing results varying the degree k of the algorithm from 0 to 2. (* CAST does not converge and reaches the upper limit for the number of iterations.)

N	K-means	CAST			TCLUST		
		w/o cutoff	w/ cutoff	per-vertex threshold	k=0	k=1	k=2
100	0.02	0.32	0.30	0.04	0.01	0.04	0.06
500	1.08	138	7.06	0.11	0.02	0.12	0.17
$1K$	9.31	2144	35.18	0.69	0.04	0.45	0.65
$2K$	71.91	7653	113.36	3.41	0.10	1.91	2.59
$5K$	1392	444405*	3442	91.64	0.32	18.43	24.30
$10K$	19693	-	8240	526	1.44	159	203
$20K$	-	-	-	5220	5.93	1125	1363
$45K$	-	-	-	109847	49.08	13223	15676

4.6.3 Results on complete dataset

We ran TCLUST on the entire corpus of expression data with the goal of identifying functionally related gene sets using tcg-threshold=0.3. This threshold was optimized (as described earlier) by using a reduced data-set of randomly chosen 5000 probes (out of the 45000) probes. As different data-sets vary in quality, such an optimization is desirable. It is also worth noting that (a) other tools, like CAST do not provide any guidance on how to set parameters; and, (b) our tool, when run with default parameters does not do much worse on the data-sets we have tried (data not shown).

All 45101 probe sets mapping onto 21452 genes were clustered. Of the 25172 resulting clusters, majority were singletons, 550 had size in range $[3, 500]$ covering 5994 probe sets. The clusters in size $[3, 500]$ were mapped to Entrez gene IDs and then analyzed in terms of functional enrichment in each of the gene annotation databases. There were 32 out of the 550 clusters found to be functionally enriched at $p \leq 0.001$.

A cluster can be seen as a collection of two types of members: annotated and unannotated. The annotated genes are used to detect the FGSs in which the cluster is enriched in. This information can be used as a quality measure for the cluster and also a basis for function prediction of the unannotated members.

Among the clusters with the most significant functional enrichments was one set of 231 genes dominated with genes of muscle related functions. It was functionally enriched in at least one FGS in each database: muscle contraction (BP, $p < 10^{-37}$), contractile fiber (CC, $p < 10^{-39}$), structural constituent of cytoskeleton (MF, $p < 10^{-14}$), and calcium signaling pathway (KEGG, $p < 10^{-25}$). There were 181 genes in this cluster which were annotated in at least one of the enriched categories. The enrichment in muscle contraction is defined by 28 genes which are primarily in the actin, actinin, myosin, tropomyosin, and troponin protein families, genes which are well known to play important roles in muscle tissue [52]. This enrichment is also supported by 32 members of the actin, actinin, myosin, troponin protein families which had no previously annotated role in muscle function. In addition, there are also some members of this cluster which also had no annotated role in muscle function but have been previously discussed in literature. For example, the expression of leiomodin 3 (Lmod3) shows high correlation with the other members of the muscle contraction cluster, suggesting that it might have a related function. This hypothesis is further supported by known roles for the other leiomodin family members, Lmod1 and Lmod2, in smooth muscle and cardiac muscle function, respectively [57].

Another cluster we detected has 36 genes and it is enriched in genes involved in hormone activity (MF, $p < 10^{-13}$), and cysteine type endopeptidase activity (MF, $p < 10^{-3}$). Nine of the genes are annotated in hormone activity, and three genes are annotated in cysteine type endopeptidase activity. Among the other unannotated genes in this cluster are cathepsin 3 (Cts3). A putative role for Cts3 in hormone activity and peptidase function is supported by evidence for extracellularly acting cathepsins mediating thyroid hormone liberation in thyroid epithelial cells [56].

Although the examples highlighted here represent only two of the many functionally enriched clusters discovered, they illustrate the tremendous potential of functional inference based on genome-wide clustering.

4.7 Discussion

We introduce here a new method, TCLUST, for clustering large, genome-scale data sets. The algorithm is based on measures of co-connectedness to identify dense subgraphs present in the data. We have applied this method to a large reference gene expression data set, and showed that the resulting clusters show strong enrichment in known biological pathways.

Although TCLUST has been shown to perform as good as or better than existing methodologies, as with any methodology, certain caveats must be noted. A possible shortcoming might be that once two vertices end up in different clusters, they are never reconnected. On the one hand, this makes the algorithm converge faster, on the other hand, it might lead to some loss of sensitivity for higher error-rates. In principle, this could be adjusted, by applying the tcg thresholds more judiciously, gaining some FN edges at the cost of some FP edges, and increasing the number of iterations. We will explore this, and similar directions in future research. Also, the theoretical justification will be clarified and extended to more general settings. Specifically, while we present sufficient conditions on error rate α, we do not report any necessary conditions. These will be the subject of future investigation.

While the development of TCLUST was motivated by the lack of a suitable clustering tool for large gene-expression data-sets, its performance on smaller data-sets is superior or at least, competitive with established methods. Moreover, the method is based on the relatively broad assumptions that the clusters behave like dense-subgraphs of an appropriate sparse sub-graph. Therefore, TCLUST should be applicable in a variety of biological settings, and offers a new approach, complementing existing methods. As biological data-sets continue to grow in scale, the importance of efficient algorithms for clustering genome-scale will become paramount, requiring continued development of efficient algorithms.

Chapter 4 is, in full, a reprint of the paper “A fast algorithm for clustering genome-scale expression data. B. Dost, A. Su, C. Wu, and V.Bafna (2008). EEE/ACM Transactions on Computational Biology and Bioinformatics. In press”. The dissertation author was the primary investigator and author of this paper.

4.8 Supplemental Data

4.8.1 Expression Profiling and Preprocessing

Eleven tissues (adipose, amygdala, dorsal root ganglia, frontal cortex, hippocampus, hypothalamus, liver, nucleus accumbens, pituitary, skeletal muscle, and spleen) were dissected from a panel of 29 diverse inbred strains. Gene expression analysis was performed using Affymetrix MOE430v2 GeneChips. After samples which did not pass quality control were removed, data for 295 samples remained. (See Table 1 in Supplemental Data.) Each microarray measured expression for 45101 probe sets targeting 21452 unique mouse genes. Each expression measurement was summarized by gcRMA (bioconductor package; [60, 61]) from the quantile-normalized probe intensities of a probe set.

We treat each set of the same tissue samples from up to 29 diverse inbred mouse strains as a separate tissue specific data set. We preprocess each of these 11 data sets separately by centering the log-transformed intensity values at zero, to highlight the variation in expression across strains and emphasize genetic background. We then merge the data sets so that we have a single dataset with 45101 probe sets and 295 samples.

4.8.2 Biological knowledge represented in gene sets

The Gene Ontology (GO) database [53] was downloaded from http://www.geneontology.org/ontology/geneontology.obo. The snap-shot of April 03, 2006 was used in this analysis, which contains 21,316 GO terms in three categories for biological process (BP), molecular function (MF) and cellular component (CC). Three unknown categories, ”GO: 0000004”, ”GO: 0005554” and ”GO: 0008372”, were removed for the analysis. The mapping from Entrez Gene IDs to GO terms was obtained from NCBI’s gene2go table (April 03, 2006 snapshot from ftp://ftp.ncbi.nih.gov/gene/DATA/gene2go.gz). In addition, we utilized two databases of manually-annotated metabolic and signaling pathways. The KEGG pathway database [66] was downloaded from ftp://ftp.genome.jp/pub/kegg/pathways/mmu/. The snapshot of April 26, 2006 was used, which contains 174 pathways for mouse. Ingenuity pathways database (ING) [74] was obtained from Ingenuity Systems, which contains 137 pathways for mouse. Finally, Mouse phenome database (MPD) [55] which is a repository of phenotypic and genotypic data on diverse inbred strains was downloaded on May 17, 2007 from http://phenome.jax.org/phenome. All flat-file formatted databases were parsed by individual python scripts for the use in the functional analysis.

4.8.3 Functional analysis of clusters

For each cluster and functional gene set (FGS) pair, the enrichment p-value p is calculated as:

$$p = \frac{\binom{N_F}{k}\binom{N-N_F}{n-k}}{\binom{N}{n}}$$

where N_F is defined as the number of genes assigned to the FGS, n is the number of genes in the cluster, k is the number of genes in the cluster that are annotated in the FGS, and N is the total number of genes [75].

Table 4.3: Eleven tissues dissected from 29 strains: adipose (AD), amygdala (AM), dorsal root ganglia (DR), frontal cortex (FC), hippocampus (HC), hypothalamus (HT), liver (LV), nucleus (NC), pituitary (PT), skeletal muscle (SM), spleen (SP). The samples for which gene expression data is available are indicated by "*".

strains	tissues										
	AD	AM	DR	FC	HC	HT	LV	NC	PT	SM	SP
129S1/SvImJ		*	*	*	*	*	*	*	*	*	*
A/J	*	*	*	*	*	*	*	*	*	*	*
AKR/J	*	*	*	*	*	*	*	*	*	*	*
BALB/cByJ	*	*	*	*	*	*	*	*	*	*	*
BTBR T+ tf/J	*	*	*	*	*	*	*	*	*	*	*
BUB/BnJ	*	*	*	*	*	*	*	*	*	*	*
C3H/HeJ	*	*	*	*	*	*	*	*	*	*	*
C57BL/6J	*	*	*	*	*	*	*	*	*	*	*
C57BR/cdJ	*	*	*	*		*	*	*	*	*	*
C58/J	*	*	*	*	*	*	*	*	*	*	*
CBA/J	*	*	*	*	*	*	*	*	*	*	
CE/J		*		*	*	*	*	*	*	*	*
DBA/2J	*	*	*	*	*	*	*	*	*	*	*
FVB/NJ	*	*	*	*	*	*	*	*	*	*	*
I/LnJ	*	*		*	*	*	*	*	*	*	*
KK/HlJ	*	*	*	*	*	*	*	*	*	*	*
MA/MyJ	*	*		*	*	*	*	*	*	*	*
MRL/MpJ	*			*	*		*		*	*	
NOD/LtJ	*	*	*	*	*	*	*	*	*	*	*
NON/LtJ	*	*	*	*	*	*	*	*	*	*	*
NZO/HlLtJ	*			*	*		*			*	*
NZW/LacJ	*	*	*	*	*	*	*	*	*	*	*
P/J	*		*	*	*		*	*		*	*
PL/J	*	*	*	*	*	*	*	*	*	*	*
RIIIS/J	*	*	*	*	*	*	*	*	*	*	*
SJL/J	*	*	*	*		*	*	*	*		*
SM/J		*	*	*	*	*	*	*	*	*	*
SWR/J	*	*	*	*	*	*	*	*	*	*	*
WSB/EiJ		*		*	*	*	*	*	*	*	*
#strains	25	26	23	29	28	26	29	27	26	28	28

4.8.4 Randomization procedure for FDR calculation

We compute the FDR associated with the number of functionally enriched clusters obtained by a clustering as follows. We generate 100 random clusterings with the same number of clusters and cluster size. This is achieved by simply permuting the probe set or gene labels. For each random clustering, the number of functionally enriched clusters was recorded so that we obtain a null distribution for the number of functionally enriched clusters. We compute the FDR for the actual number of functionally enriched clusters by:

$$FDR(n) = \frac{n_R}{n}$$

where n is the actual number of functionally enriched clusters and n_R is the mean number of functionally enriched clusters in the null distribution.

Chapter 5

Mass Spectrometry Based Protein Quantification

Abstract

In analyzing the proteome using mass spectrometry, the mass values help identify the molecules, and the intensities help quantify them, relative to their abundance in other samples. Peptides that are shared across different protein sequences are typically discarded as being uninformative w.r.t each of the parent proteins.

In this chapter, we investigate the use of shared peptides which are ubiquitous ($\sim 50\%$ of peptides) in mass spectrometric data-sets. In many cases, shared peptides can help compute the relative amounts of different proteins that share the same peptide. Also, proteins with no unique peptide in the sample can still be analyzed for relative abundance. Our work is the first attempt to use shared peptides in protein quantification, and makes use of combinatorial optimization to reduce the error in relative abundance measurements. We describe the topological and numerical properties required for robust estimates, and use them to improve our estimates for ill-conditioned systems. Extensive simulations validate our approach even in the presence of experimental error. We apply our method to a model of Arabidopsis root knot nematode infection, and elucidate the differential role of many protein family members in mediating host response to the pathogen.

5.1 Introduction

The analysis of the proteome using mass spectrometry involves the separation of molecules (often, enzymatically digested peptides from expressed proteins) followed by accurate measurement of mass of each molecule, termed as the mass-spectrum. Together with mass, the spectrum also measures peak-intensity for each molecule. For any constituent peptide from a protein sequence, its spectral intensity is a measurement of *abundance*, the amount of the expressed protein. However, the actual value is hard to interpret, as it depends upon a number of poorly understood factors, including instrument types, energetics of the process, and physico-chemical properties of the peptide itself. Consequently, it is often the *relative-abundance* of a peptide, measured as the ratio of intensities of a peptide across samples, that is investigated [79, 83]. By the same token, intensity values of different peptides are usually not comparable.

The relative abundance of a peptide is a proxy for the relative abundance of the parent protein. This is acceptable only when the peptide sequence is unique to the protein. By contrast, when a peptide is shared across proteins (Ex: proteins that share domains), its abundance (and relative abundance) depends upon contributions from multiple proteins. For this reason, shared peptides have been traditionally disregarded in protein-level quantification analysis. However, this may significantly decrease the number of proteins for which abundance estimates can be obtained. While often unreported, a significant portion of the data (as much as 50%) is ignored. In our own experiment with Arabidopsis proteins, $4,145(48\%)$ of the $8,584$ expressed proteins were not represented by a unique peptide and would normally be discarded.

The goal of this work is to demonstrate that shared peptides are a resource that adds value, and we make the point with two simple examples. Consider a case with two proteins p_1, p_2, and 3 constituent peptides s_1, s_2, s_3, where s_1, s_2 are unique, and s_3 is shared. See Figure 5.1a, where a peptide is connected to a protein by an edge only if it is contained in it. Consider an experiment that revealed the relative abundances (r_1, r_2, r_3) of the 3 peptides over two samples B and A as $16, 1, 4$ respectively. The typical approach is to discard the shared peptide s_3, and to assert that p_1 is $16\times$ over-expressed, while p_2 is unchanged. Formally, if Q_j^A, Q_j^B represent the actual abundance of protein j in samples A, B respectively, then

$$\frac{Q_1^A}{Q_1^B} = r_1 = 16 \quad \text{and} \quad \frac{Q_2^A}{Q_2^B} = r_2 = 1.$$

Our point is that the ignored peptide s_3 also provides information because

$$r_3 = 4 \quad \simeq \quad \frac{Q_2^A + Q_1^A}{Q_2^B + Q_1^B} = \frac{Q_2^B + 16Q_1^B}{Q_2^B + Q_1^B} = \frac{16 + \frac{Q_2^B}{Q_1^B}}{1 + \frac{Q_2^B}{Q_1^B}}.$$

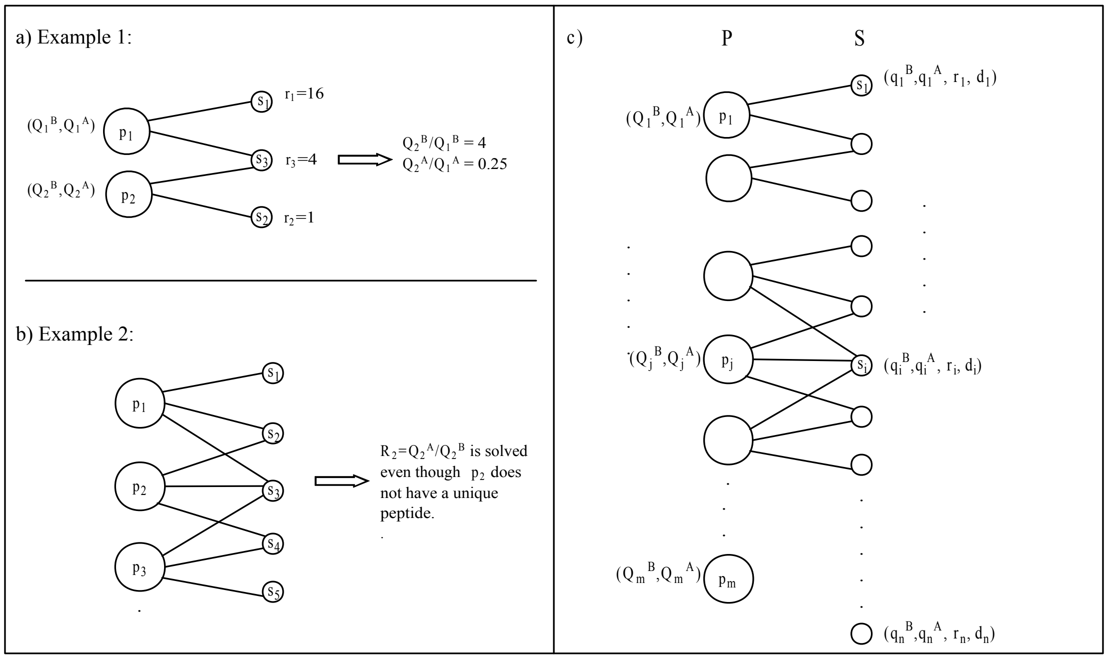


Figure 5.1: (a, b) Two examples illustrating our approach for protein quantification via shared peptides. (c) Protein-peptide bi-partite graph $G = (P \cup S, E)$ representing the mapping between m proteins and n peptides.

Solving, we learn that $\frac{Q_2^B}{Q_1^B} = 4$, indicating that p_2 is $4\times$ more abundant that p_1 in sample B. Here, we have 4 unknowns from the 2 proteins, and 3 constraints, one from each of the peptide. By using ratios, (Ex: $R_j = Q_j^A/Q_j^B$), we reduce the number of unknowns, and can solve to get the extra information. Note that the unit of measurement for Q_j^A, Q_j^B is immaterial. For this reason, we always reduce one degree of freedom, typically by adding the constraint $\sum_j Q_j^B = 100$, or solving for ratios, as we do here. As long as the number of constraints matches the number of unknowns, we can solve to get the relative abundances of different proteins, possible only with shared peptides.

Consider a second example, a more complex one this time, with 3 proteins p_1-p_3 and 5 peptides s_1-s_5, as shown in Figure 5.1b. Here, protein p_2 does not have any unique peptide, and would normally be discarded. However, the system has $5 + 1$ constraints, and 6 unknowns. Therefore, solving the system gives us Q_2^A, Q_2^B, and therefore, the relative abundance $\frac{Q_2^A}{Q_2^B}$ of p_2.

To summarize, shared peptides provide extra information in protein quantification. Under certain conditions, they allow us to a) compute relative abundance of a protein even when it does not contain a unique peptide, and b) compute relative abundance values of two different proteins in a sample. To our knowledge, this is the first work to exploit shared peptides in this manner. However, the simple idea is confounded by the realities of missing data, and error in experiments. Here, we lay out the theoretical foundations and practical considerations in determining when the shared peptide abundances can be used reliably. We show that the solvability must depend upon the topological properties of the peptide-protein relationships as well as numerical properties of the experimentally determined intensity values. It is often the case that interesting cases cannot be resolved because of missing data, or numerical instability.

As an extension to our approach, we also consider some intrinsic properties of peptides. Informally, define the *detectability* of a peptide as the probability that it will be detected via MS, when the parent protein is expressed. We propose an alternative formulation that estimates the peptide detectabilities in addition to absolute and relative abundances of proteins when appropriate data is available.

Furthermore, we suggest two improvements to increase the number of cases that can be solved. First, we describe a algebraic technique based on singular value decomposition to make robust inferences for numerically ill-conditioned systems. Recent results have shown that detectability is indeed an intrinsic characteristic of peptides that can be computed in independent experiments, and maintained for future use [76]. We also point out that incorporating detectabilities as known variables in our formulation, it is possible to solve a much larger number of cases.

In Section 5.2, we describe the theoretical and empirical considerations for shared peptide analysis. In Section 5.3.1, we validate our approach with extensive simulations. We apply our methods to data from ITRAQ experiments comparing an Arabidopsis model of root-knot infection versus wild-type in Section 5.3.2. Our results elucidate the relative abundance among different members of a family in over 55 Arabidopsis protein families.

5.2 Protein quantification via shared peptides

We represent the protein quantification data using a bipartite graph $G = (P \cup S, E)$ where P is the set of proteins and S is the set of peptides. For all $p \in P, s \in S$, $(p, s) \in E$ if and only if peptide s is a substring of the protein sequence p. Note that different connected components of G do not influence each other, and we treat each component independently. W.l.o.g, assume that G is connected, and let $|P| = m$ and $|S| = n$. See Figure 5.1c. Consider the case where only two samples are involved. In many experiments, the abundances are measured before and after a treatment, so we denote the samples as B, and A. We associate two variables (Q_j^B, Q_j^A) corresponding to the 'before' and 'after' abundance for each protein $p_j \in P$. As mentioned earlier, we also add the constraint $\sum_j Q_j^B = 100$.

Analogous to proteins, we associate values q_i^B, q_i^A, r_i with each peptide $s_i \in S, i = 1 : n$ where $r_i = q_i^A/q_i^B$ denotes the ratio of the peptide s_i abundance between samples. It is possible to generalize the representation for the data with more than two samples. While this abstraction hides many of the complexities of protein quantification via mass spectrometry, it is useful to present our approach which can be applied to many different quantification protocols, including labeled and label-free approaches.

Key to our computation are equations that connect all proteins p_j which contain a single peptide s_i. In the absence of experimental error, the abundance values must satisfy the following $n + 1$ constraints over $2m$ variables.

$$\begin{array}{l} \sum_{(p_j,s_i)\in E} Q_j^A - r_i \times \sum_{(p_j,s_i)\in E} Q_j^B = 0 \quad \text{for all } s_i \in S \\ \sum_j Q_j^B = 100 \end{array}$$

With no errors, we can solve this equation uniquely as long as $n + 1 \geq 2m$. To incorporate errors, we consider a *linear-programming* formulation that minimizes the total error. (See Figure 5.2, F_1 formulation.)

a)

Input	Output	F_1Formulation
$r_i, \forall s_i \in S$	Q_j^B, Q_j^A, $\forall p_j \in P$	$\min \sum_{i=1}^{n} \lvert\varepsilon_i\rvert$ s.t. $\sum_{p_j \in P} Q_j^B = 100$ $\varepsilon_i = \sum_{(p_j,s_i)\in E} Q_j^A - r_i \times \sum_{(p_j,s_i)\in E} Q_j^B \quad \forall s_i \in S, r_i \geq 1$ $\varepsilon_i = r_i \times \sum_{(p_j,s_i)\in E} Q_j^A - \sum_{(p_j,s_i)\in E} Q_j^B \quad \forall s_i \in S, r_i \leq 0$ $Q_j^B \geq 0, Q_j^A \geq 0 \quad \forall p_j \in P.$

b)

Input	Output	F_2Formulation
q_i^B, q_i^A, r_i, $\forall s_i \in S$	Q_j^B, Q_j^A, $\forall p_j \in P$ $d_i, \forall s_i \in S$	$\min \sum_{i=1}^{n} \lvert\varepsilon_i^B\rvert + \lvert\varepsilon_i^A\rvert$ s.t. $\sum_{p_j \in P} Q_j^B = 100$ $\varepsilon_i^B = \sum_{(p_j,s_i)\in E} Q_j^B - q_i^B f_i, \quad \forall s_i \in S$ $\varepsilon_i^A = \sum_{(p_j,s_i)\in E} Q_j^A - q_i^A f_i, \quad \forall s_i \in S$ $Q_j^B \geq 0, \; Q_j^A \geq 0, \quad \forall p_j \in P.$

Figure 5.2: Input, output, and computation summary of two LP formulations for protein quantification via shared peptides. a) F_1: A formulation that does not include peptide detectability, and; b) F_2: using peptide detectabilities. We use $f_i = 1/d_i$ as the reciprocal of detectability to maintain linear constraints.

Note that the ratios are not symmetric about 1, so we always choose a constraint where the ratio contribution is greater than 1. To simplify notation, we will also represent the LP formulation in a matrix form as

$$\min \sum_i |\varepsilon_i| \text{ where } \varepsilon = \mathcal{A}x - b, x \geq 0 \tag{5.1}$$

where x is vector of dimension $2m$, given by $x = [Q_1^B, \ldots, Q_m^B, Q_1^A, \ldots, Q_m^A]^T$, b is a $(n+1)$-dimensional vector described by $b = [100, 0, \ldots, 0]^T$, and $\mathcal{A}$ is a $(n+1) \times 2m$ matrix. While this LP is not in standard form, it can easily be transformed into one.

The formulation of the linear program is natural in that the LP seeks for protein abundances that optimally fit the observed peptide ratios. Nevertheless, it raises questions about our confidence in the estimates of Q_j. Note first that a low value for the objective does not necessarily result in robust estimates of Q_j. Consider an underdetermined system, where $n + 1 < 2m$. By setting an arbitrary subset of $2m - (n + 1)$ variables to 0, and solving for the remaining, we obtain multiple solutions, each with 0 error. A simple illustration of this is found in the notion of *symmetric proteins*. Define proteins $p_1 \in P$ and $p_2 \in P$ as symmetric if and only if the set of incident peptides $S_1 = \{s|(p_1, s) \in E\}$ and $S_2 = \{s|(p_2, s) \in E\}$ are identical. Two symmetric proteins imply two identical columns in $\mathcal{A}$, which means that any linear combination of abundances for these 2 proteins will lead to an identical solution. Certainly, we can solve this problem as a special case: simply merge the two identical columns (proteins) into one, effectively reducing m. However, more complex dependencies might arise which are harder to detect.

Generalizing, when rank$(\mathcal{A}) < 2m$, we get multiple solutions with zero-error. If however, $\mathcal{A}$ has full column rank, then by parsimony arguments, the LP solution is likely to provide accurate estimates of protein abundance values. Even if the system is full-rank, it might be ill-conditioned, resulting in poor estimates. We define a *rank-threshold* function to characterize the solvability, a quantity that is closely related to the condition number of the matrix. Start with the singular-value decomposition of $\mathcal{A}$. Using standard approaches, compute matrices U, V, Σ such that

$$\mathcal{A} = V \Sigma U^T$$

Σ is a $(n + 1) \times 2m$ diagonal matrix with nonnegative real numbers $\sigma_1, \ldots, \sigma_p$ on the diagonal. These p diagonal entries describe the singular values of $\mathcal{A}$ in decreasing order of magnitude, where $p \leq \min\{n+1, 2m\}$. U is orthonormal with dimensionality $2m \times 2m$, and V is orthonormal with dimensionality $n+1 \times n+1$, respectively. The rank of $\mathcal{A}$ is given by the number of non-zero singular values. We define a related concept, *rank-threshold* of $\mathcal{A}$ as

$$\mathcal{R}(\mathcal{A}) = \min\{t | \sigma_j > 10^{-t} \forall j\}$$

$\mathcal{R}(\mathcal{A}) = t$ being low implies that all its singular values are large ($\geq 10^{-t}$), implying that estimates of protein abundance values should be robust. In our experiments, we will show that the rank-threshold is a good way to characterize the reliability of the final solution.

Robust estimates for ill-conditioned systems: This formalism allows us to distinguish high rank systems $\mathcal{A}$ for which we can estimate protein abundance reliably, but it also provides a handle into under-determined systems. Using our notation, we can describe an under-determined system as one in which $\mathcal{R}(\mathcal{A})$ is high. Specifically, $\mathcal{R}(\mathcal{A}) = \infty$ implies the case when some of the singular values are 0. For a rank threshold t, define the *rank$_t$* of $\mathcal{A}$ as

$$\text{rank}_t(\mathcal{A}) = \max\{j | \sigma_j > 10^{-t}\}$$

This 'thresholded' rank allows us to get the true dimensionality of a system for which we could get robust results. For all j, let U_j denote the $2m \times j$ matrix formed by taking the first j columns of U (corresponding to the dominant singular values). Likewise, let V_j denote the matrix formed by the first j columns of V, and $\Sigma_j = \text{diag}[\sigma_, \ldots, \sigma_j]$. This implies that if $\text{rank}_t(\mathcal{A}) = k$, then

$$\mathcal{R}(\mathcal{A}U_k) = \mathcal{R}(V_k\Sigma_k) = t$$

We choose $B = \mathcal{A}U_k$, and solve the linear program for the k dimensional vector y

$$\min \sum_i |\varepsilon_i| \text{ where } \varepsilon = \mathcal{B}y - b, U_k y \geq 0 \tag{5.2}$$

Note that $\mathcal{R}(B) = t$ implying that the estimates of y are robust. The reason to keep y unconstrained, but impose $U_k y \geq 0$ is the following: The values y cannot be interpreted directly, but can be used to retrieve protein abundance values x by solving

$$x = U_k y$$

Our constraints ensure that the protein abundance values are non-negative.

5.2.1 Incorporating peptide detectability:

Here we consider an alternative formulation that builds on different assumptions to improve robustness to measurement errors and potentially greatly increase the numbers of components that can be solved. Assuming one is able to estimate the *absolute* peptide abundances q_i^B and q_i^A (as previously described [77]), this formulation allows one to relate the absolute peptide abundance with the total abundance of its parent proteins and thus make inferences about peptide *detectabilities* in addition to relative protein abundances.

We define the detectability of a peptide s_i as a quantity $d_i \in [0, 1]$ that relates peptide abundance to the abundances of its parent proteins. In the absence of experimental error, for each peptide $s_i \in S$,

$$\begin{array}{c} q_i^B = d_i \times \sum_{(p_j, s_i) \in E} Q_j^B \\ q_i^A = d_i \times \sum_{(p_j, s_i) \in E} Q_j^A \end{array} .$$

In dealing with errors, we use a linear programming formulation that is similar to F_1, but with $2n + 1$ constraints and $2m + n$ variables. (See Figure 5.2, F_2 formulation.) We use $f_i = \frac{1}{d_i}$ as the reciprocal of detectability to maintain linearity of equations. The previous discussion regarding reliable estimates of abundance values is unchanged from the previous section.

The ITRAQ data does not provide peptide abundance values that can be used directly for F_2. However, recent developments indicate that the absolute peptide abundances can be experimentally estimated [77]. Also, recent results have shown that peptide detectabilities can be reliably estimated with very little variability across mass spectrometry runs [76, 83]. This observation is especially important in F_2. The knowledge of peptide detectabilities implies $2m$ variables instead of $2m + n$ and greatly increases the number of cases that can be solved.

5.3 Results

Data-set: We choose an Arabidopsis model of root-knot nematode infection. The root-knot nematodes are worm-like, microscopic plant-parasites that infect a multitude of plants, including all major crops, turf, and many ornamental plants. The diversity and extent of infection makes it economically significant to explore. The typical mode of infection is via the root. The female nematode lays its eggs at the root tip. The juveniles infect via the root tip, and move up. Inside, they manipulate the cellular machinery to create specialized *feeding* cells, which grow and multi-nucleate, but do not divide, eventually forming giant cells that provide nutrients to the parasite [78, 85]. As the nematodes exploit the Arabidopsis cellular machinery to create the giant cell phenotype, an analysis of proteins that are differentially expressed in infected versus non-infected host cells can help elucidate the underlying mechanism [79]. As the Arabidopsis genome is sequenced, with extensive annotation on the known genes and pathways, it is an appropriate model for the host.

An ITRAQ method was used to collect protein abundance information. A brief overview of the method is given here (See [86] for details). The samples are enzymatically digested into short peptides. Peptides from different samples are N-terminally covalently labeled with tags of different mass, but then pooled and analyzed together using tandem mass spectrometry. Each spectrum contains both the fragment masses used to identify the peptide, and the intensities of the differential tags for abundance computation. In our terminology, for every peptide s_i, we read the intensities of the two tags as q_i^A q_i^B, and compute the ratio $r_i = q_i^A/q_i^B$, which approximates the ratio of the peptide abundance values in the two samples.

Our data-set is a collection of $118,426$ spectra, encoding $27,728$ peptides mapping onto $8,584$ protein sequences. Each protein is mapped to at least one peptide and vice versa. The number of peptides mapping to a protein sequence varies considerably, ranging from from 1 to 59. The distribution of the number of peptides per protein, is shown in Figure 5.3. Close to half of the proteins $(4,145$ out of $8,584)$ do not have a unique peptide. The number of unique peptides per protein range from 0 to 51. The distribution of the number of unique peptides per protein is shown in Figure 5.3. Likewise,

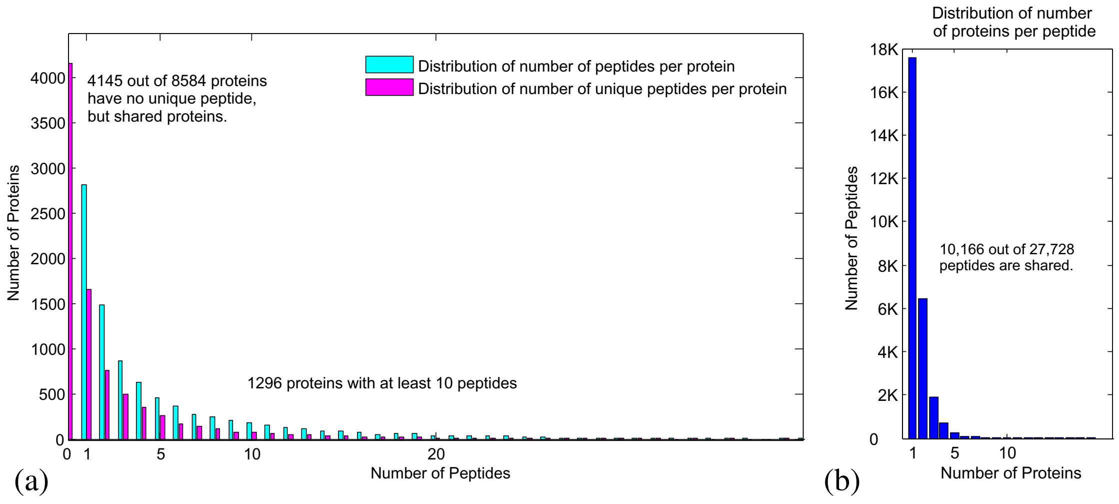


Figure 5.3: Mapping of peptides to proteins in Arabidopsis root-knot nematode infection ITRAQ data. (a) Distribution of the number of peptides and unique peptides per protein. (b)Distribution of the number of proteins per peptide.

there is tremendous spectral redundancy among peptides, with the number of spectra encoding a peptide ranging from 1 to 975. Close to half of the peptides $(10, 166)$ are shared by multiple protein sequences. We reduce the data by merging *symmetric* peptides, or peptides that belonged to an identical subset of proteins. The redundancy helps understand the measurement error, and the merging removes artificial dimensionality, giving a better measure of rank, and rank-threshold. Likewise, we also merge the symmetric proteins for reasons mentioned earlier.

After merging, we obtain a protein-peptide bi-partite graph $G = (P \cup S, E)$, where $|P| = 6,998$, $|S| = 8,069$, $|E| = 13,055$. G has 4119 connected components projecting onto 257 non-isomorphic topologies with size ranging from 2 to 127. In this study, we consider only the 1190 non-trivial components, with at least 2 proteins.

In addition to testing on this data, we also perform a series of controlled experiments by simulating data-sets based on the topologies of the Arabidopsis data-set.

Generation of simulation data: We start with 257 topologically distinct components of the Arabidopsis data, and generated 100 data-sets from each topology with different values. For each component, we do the following:

1. Sample protein amounts $\vec{Q}^B = [Q_1^B, \ldots, Q_m^B]$ at random from the collection of

ITRAQ tag intensities.

2. Generate ratio R_j by sampling from a log-normal N$(0, \sigma_R)$ distribution. σ_R is set to 0.7 which is the estimated standard deviation of the log peptide ratios in the Arabidopsis data. Compute $Q_j^A = R_j Q_j^B$.

3. For each peptide s_i, generate d_i uniformly from $(0, 1]$, and compute q_i^A and q_i^B as $d_i \sum_{(p_j,s_i)\in E} Q_j^A$ and $q_i^B = d_i \sum_{(p_j,s_i)\in E} Q_j^B$, respectively. When detectabilities are not incorporated, choose $d_i = 1$.

4. Compute peptide ratios r_i, and perturb according to a log-normal $N(0, \sigma)$, over a range of values σ. Denote σ as *perturbation level*.

We consider the *system* of constraints for each data-set.

Once the data is generated, only the peptide ratios r_i are used as inputs. The linear programs are solved for $\vec{Q}^{B\prime} = [Q_1^{B\prime}, \ldots, Q_m^{B\prime}]$, and $\vec{Q}^{A\prime} = [Q_1^{A\prime}, \ldots, Q_m^{A\prime}]$ using ILOG OPL Development Studio 6.1. The reliability of the estimates is tested using three measures.

Validation statistics: Recall that the value of the optimized objective is a weak indicator of the quality of results. For the simulations, as the protein abundances are known, we can compute the error in the estimate as the protein-abundance-distance, PAD:

$$\text{PAD}(\vec{Q}^{B\prime}, \vec{Q}^{B}) = \frac{\| \vec{R^B} \|_1}{m} \tag{5.3}$$

where $\vec{R^B} = \left[\ln \frac{Q_i^{B\prime}}{Q_i^B}\right]$. While any norm can be used as a valid measure of distance, the choice of the 1-norm, averaged over the dimensions can be loosely interpreted as average fold difference between actual and estimated protein abundances. The true protein abundances are not available for the Arabidopsis ITRAQ data, so we compute an indirect measure LRD, defined as

$$\text{LRD}(\vec{r}, \vec{r\prime}) = \frac{||\vec{r} - \vec{r}\prime||_1}{n} \tag{5.4}$$

where $\vec{r} = [\ln(r_1), \ldots, \ln(r_n)]$ is the vector of experimental peptide log-ratios for the n peptides, and $\vec{r}\prime$ describe the peptide log ratios computed by using the estimated protein abundances. Intuitively, if the protein abundance estimates are accurate, the computed peptide log-ratios should match the experimental log-ratios. In a similar way, we compute the peptide log detectability distance LDD as the average 1-norm of the logs of detectabilities. We use PAD, LRD, and LDD to test performance on simulations.

5.3.1 Results of Simulation

As the reliability of the estimates depend upon rank of $\mathcal{A}$, we loosely group each of 100×257 simulated systems into three categories according to $rank(\mathcal{A})$ for a fixed rank-threshold t, as follows:

Category I : $rank_t(\mathcal{A}) = 2m \leq n + 1$ (Over-determined, full-rank systems).

Category II: $rank_t(\mathcal{A}) < 2m \leq n + 1$ (Ill-conditioned systems).

Category III : $rank_t(\mathcal{A}) \leq n + 1 < 2m$ (Under-determined systems).

At rank-threshold 1, we obtain 1074, 3926, and 20700 systems in Categories I, II, and III for the F_1 simulation, and similar distributions for F_2. As the rank-threshold is increased, some of the Category II systems move into Category I (Table 5.1). Additionally, the performance of under-determined systems is uniformly worse than the other two (data not shown). Therefore, we will focus on Category I evaluation using different rank-thresholds. For each category, and each validation statistic, we compute *cumulative-probability* as the fraction of systems in that category that achieve a certain distance or lower. The ideal case is when the cumulative probability is 1 at distance 0.

Table 5.1: Simulation Category I systems grouped according to their rank thresholds.

	$\mathcal{R}(\mathcal{A})$				
	1	2	4	8	16
F_1 - Category I	1074 (4.2%)	2514 (9.8%)	3044 (11.8%)	3980 (15.5%)	4388 (17.1%)
F_2 - Category I	663 (2.6%)	2251 (8.8%)	2955 (11.5%)	3104 (12.1%)	4989 (19.4%)

In the absence of noise, we achieve the ideal case, zero PAD and LRD, for all Category I systems at rank-threshold 4. As noise is introduced to data and less stringent rank-thresholds is used, we deviate from the ideal case. Figure 5.4a shows the cumulative probability distribution against PAD, and LRD at perturbation level 0.01. Out of 1074 Category I systems at rank-threshold 1, 75% have PAD error of less than 0.16, and an LRD error of less than 0.01. The performance degrades for higher rank-thresholds. Note that in all cases, the optimized objective is very close to 0 ($\sim 10^{-4}$). However, in the ill-conditioned and under-determined systems, multiple solutions will lead to a low-error solution, and an arbitrarily picked solution will have high PAD and LRD error. The performance also degrades with an increase in perturbation error. Figure 5.4b plots the cumulative ratio for Category I systems at rank-threshold 1 under increasing perturbation levels. Thus while 93% of systems show LRD of at most 0.1 at perturbation 0.01, the number falls to 55% at perturbation 0.15.

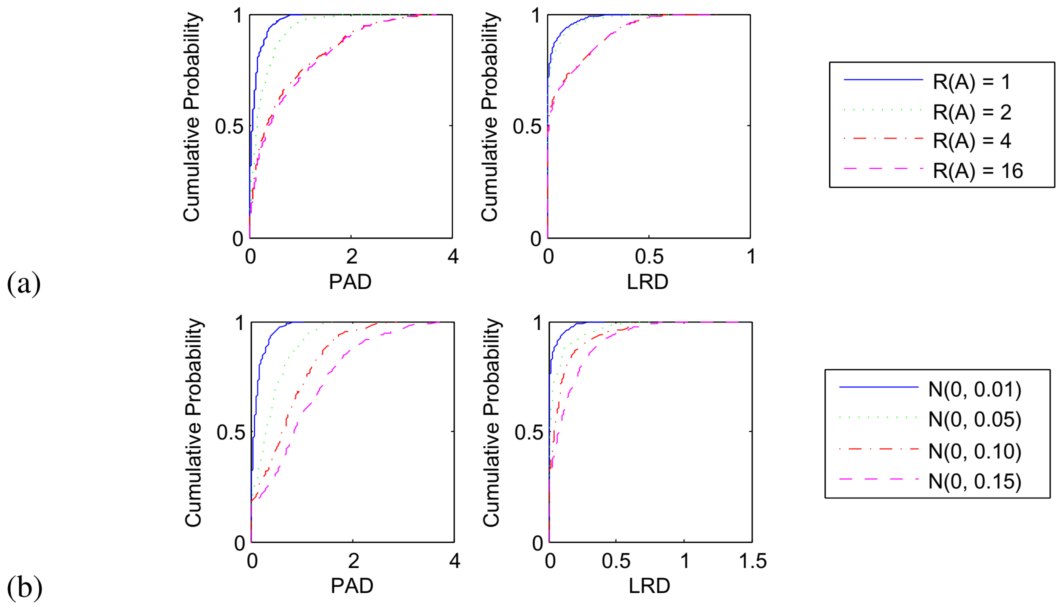


Figure 5.4: Simulation results using F_1 formulation. Cumulative probability of PAD and LRD for Category I systems (a) perturbation level 0.01, but different rank-thresholds (b) at rank-threshold 1, but different perturbation levels. In all cases, we measure the fraction of systems that were estimated within a certain distance.

Ill-conditioned systems We identified a number of Category II systems where the rank-threshold was poor, but only because a small number of singular values were close to 0. For example, in a simulation with perturbation level 0.05, we observed 339 systems for which fewer than 3 singular values were at most 10^{-16}, and $\text{rank}_1(\mathcal{A}) \geq 2m - 3$ (remaining s.v. $\geq 10^{-1}$). Our results show that the revised LP, suggested for ill-conditioned systems in Section 5.2, indeed provides better estimates of protein abundance values of these systems. (See Figure 5.5.) For example, over 88% of the systems from the revised LP achieve an LRD of 0.25 or better, compared to 65% from the original formulation.

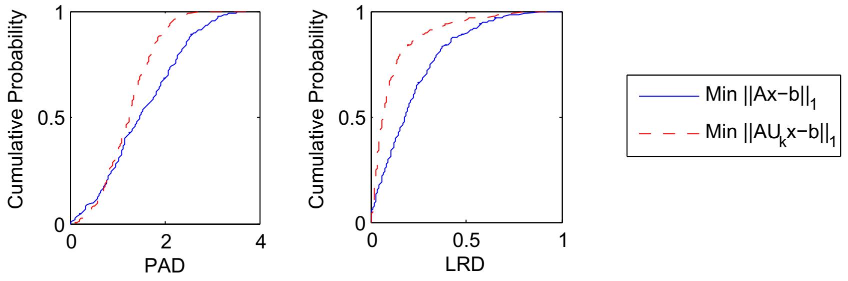
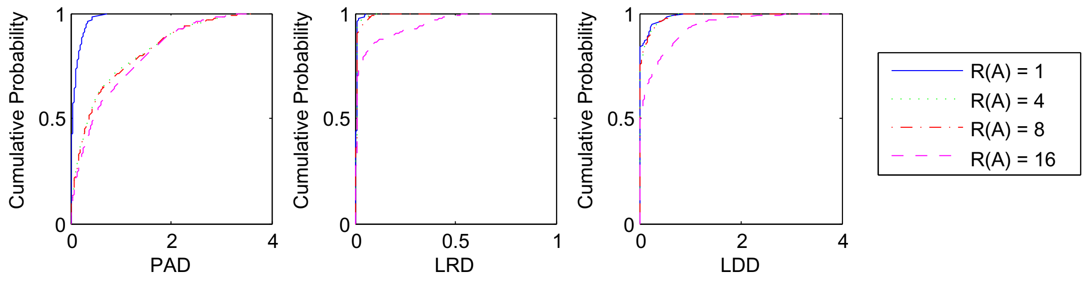


Figure 5.5: Improved estimates of protein abundance values using SVD based projection on Category II (over-determined but ill-conditioned systems).

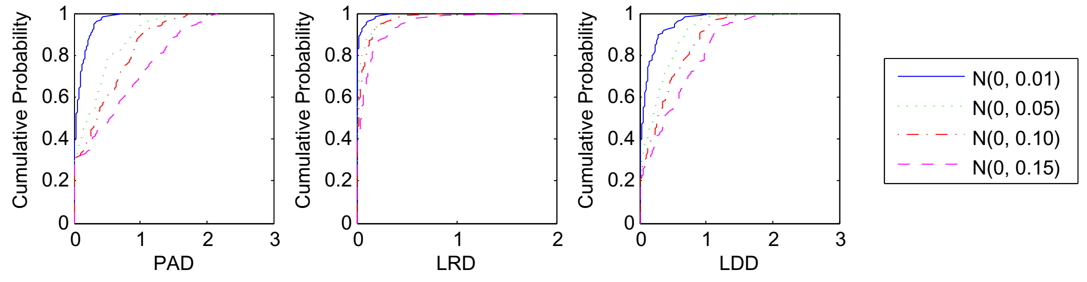
Peptide detectabilities A similar behavior is observed during estimation of peptide detectabilities (Figure 5.6a,b). The performance of PAD, and LRD surprisingly does not change with the addition of an extra n unknowns (also, n new constraints get added). We also plot the performance of detectability estimates. As expected, the performance is acceptable for low rank-threshold systems and low perturbations, but degrades for higher rank-thresholds, and higher perturbation levels. The detectability estimates are robust as well, and degrade in a similar manner.

5.3.2 Arabidopsis ITRAQ data

We focus on the 1190 non-trivial systems from the ITRAQ data comparing infected samples to non-infected ones. ITRAQ data is not appropriate to get peptide abundance values reliably, so we only use the F_1 formulation on this data-set. The



(a)



(b)

Figure 5.6: Simulation results using F_2 formulation. Cumulative probability of PAD, LRD and LDD for Category I systems (a) at different rank-thresholds (b) at rank-threshold 1, but different perturbation levels. In all cases, we measure the fraction of systems that were estimated within a certain distance.

distribution of systems in different rank categories is described in Figure 5.7a. A total of 99 systems fall into Category I with the most stringent rank-threshold 1, covering 219 proteins and 357 peptides. In addition to relative protein abundances, we estimate abundance ratios across samples for 4 proteins which do not have a unique peptide.

As actual protein abundances are not known, we use LRD to evaluate the estimates. The LRD statistic of different categories is as expected with this group performing better than the other groups. Within the 99 systems, 79 have LRD smaller than 10^{-1}, and 55 have LRD smaller than 10^{-4}. The list of systems, constituent peptides, and protein abundance values are shown in online supplemental data. Here, we cherry-pick a few representative examples that point to the differential expression of individual sequences in response to the infection.

Ca^{2+} ATPases: One of the systems is encoded by 3 proteins from the P-type Ca^{2+} ATPase super-family involved in Ca^{2+} transport. The three members are the plasma-membrane bound AT5G57110 (ATPase 8), AT4G2990 (ATPase 10), and AT3G21189

(a)

$\mathcal{R}(\mathcal{A})$	Cat. I	Cat. II
1	99 (8.3%)	191 (16%)
2	249 (20.9%)	41 (3.4%)
4	276 (23.2%)	14 (1.2%)
8	277 (23.3%)	13 (1.1%)
16	282 (23.7%)	8 (0.7%)

(b)

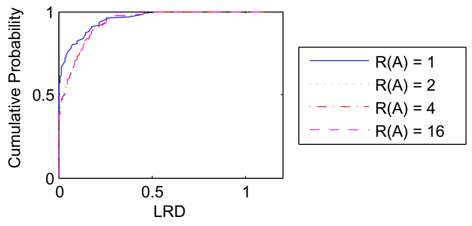
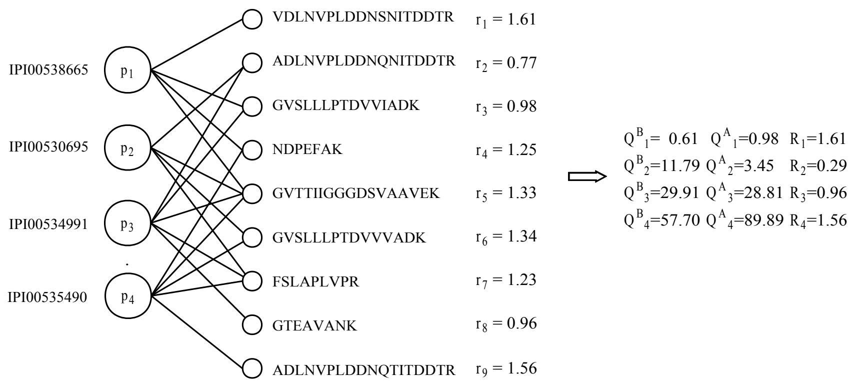


Figure 5.7: Arabidopsis root-knot nematode infection ITRAQ data. (a) Number of systems in Category I and II at different rank-thresholds. (b) Empirical cumulative probability distribution of LRD for Category I systems at different rank-thresholds.



IPI00538665
IPI00530695
IPI00534991
IPI00535490
p_1
p_2
p_3
p_4
VDLNVPLDDNSNITDDTR $r_1 = 1.61$
ADLNVPLDDNQNITDDTR $r_2 = 0.77$
GVSLLLPTDVVIADK $r_3 = 0.98$
NDPEFAK $r_4 = 1.25$
GVTTIIGGGDSVAAVEK $r_5 = 1.33$
GVSLLLPTDVVVADK $r_6 = 1.34$
FSLAPLVPR $r_7 = 1.23$
GTEAVANK $r_8 = 0.96$
ADLNVPLDDNQTITDDTR $r_9 = 1.56$
Q^B_1= 0.61 Q^A_1=0.98 R_1=1.61
Q^B_2=11.79 Q^A_2=3.45 R_2=0.29
Q^B_3=29.91 Q^A_3=28.81 R_3=0.96
Q^B_4=57.70 Q^A_4=89.89 R_4=1.56

Figure 5.8: PhosphoGlycerate kinase family members sharing peptides.

(ATPase 9). Earlier reports have suggested that ATPase8,10 are co-expressed evenly over all vegetative tissues, while ATPase 9 is expressed almost exclusively in pollen [84]. In our data, the 3 form a confected component with 6 peptides, and our analysis showed the relative wild-type expression of the 3 to be $34.3\%, 57.5\%, 8.22\%$ respectively, confirming this observation. Further, we find that ATPase 8 is $3\times$ over-expressed in the infected state.

Profilins: The Profilin family encodes proteins that regulate actin cytoskeleton formation. Five profilins are known. The two that are identified in our data (PRF-1,2) are constitutively expressed in all vegetative organs, and a regulatory element in their first intron is suspected to mediate this expression, differentiating them from the other Profilins [80]. In our data-set, the two are in a component with 3 peptides, one shared. Our analysis shows that Profilin-2 has only (13%) of the total abundance, and is further reduced two-fold upon infection.

Cinnamyl-alcohol dehydrogeneases: A number of genes in Arabidopsis have been annotated as part of the CAD family, an assertion which has subsequently been challenged, pointing instead to the central role of two members (AtCAD4, and AtCAD5) in the CAD metabolic network. These two molecules have expression patterns consistent with lignification at stem tissues. Interestingly, expression was also observed in various non-lignifying zones (e.g. root caps) indicative of a possible role in plant defense [82]. Our results have a single connected component with AtCAD4,5 and 3 peptides. The analysis shows that both proteins are equally abundant, with AtCAD5 (AT4G34230) at 56% in non-infected cells. However, AtCAD5 is significantly ($1.5\times$) over-expressed, while AtCAD4 (AT3G19450) is $2\times$ under-expressed.

Phosphoglycerate kinases: Phosphoglycerate kinases have been previously shown to be differentially expressed during defense response of Arabidopsis [81]. In our data, four proteins from this family are in a component with nine peptides as shown in Figure 5.8. In this example, along with the relative protein abundances, we also compute the abundance ratio across-sample for IPI00530695 even though it does not have a unique peptide. Our analysis suggests that both IPI00538665 and IPI00535490 are $1.5\times$ over-expressed, but IPI00535490 is much more abundant in both samples. IPI00530695 is $3\times$ under-expressed while the abundance of IPI00534991 does not change.

5.4 Discussion

The extent of peptide sharing in proteomics is under-estimated; consequently, shared peptides, and proteins with non-unique peptides are typically discarded, corresponding to as much as 50% of the data in our experience. As mass spectrometry based protein quantification becomes routine, shared peptide analysis will be increasingly important. Our results are the first to show that a careful analysis not only helps in recovering abundance values of some of these proteins, but also helps quantify the relative levels of different proteins. These across-protein relative abundance computations can help elucidate these differential regulation of the proteins from a family. We investigate topological and numerical considerations in estimating reliability of our computations. Nevertheless, the final quality of the results does depend upon the accuracy of the experimental abundance computations [77, 83]. As the mass spectrometers become more accurate, experimental variation in relative abundance computations will decrease, increasing the power of our methods. In a similar fashion, the estimation of peptide detectabilities is in an early stage of development. Our results attest to the viability of using shared peptides for detectability computation, but also point to the importance of detectability values in extending the scope of shared peptide analysis. The model's ability to automatically estimate peptide detectabilities may result in an ongoing cycle of self-refinement where different systems resulting from different experimental conditions may allow one to continuously expand the set of known detectabilities, which in turn would allow for the resolution of more complicated systems. In fact, we note that this progressive convergence towards an extensive database of peptide detectabilities may even allow one to learn more about systems that were previously not solvable in a given experiment by adding information from different or even additional targeted experiments aimed at estimating the necessary detectabilities.

A final contribution of this study is the use of novel evaluation methods for shared peptide computations. Clearly, different algorithms can be used to optimize the error in estimation, including non-linear optimization and other machine learning approaches. We have experimented using simulated annealing approach with a non-linear cost function that minimizes the absolute sum of differences between observed and expected peptide ratios. While such approach is more time consuming, it provides better estimates for systems with unbalanced protein abundances as linear programming formulation is biased towards the error terms associated with the more abundant proteins. Details of that study will be discussed somewhere else. This chapter describes a systematic simulation based framework to compare, and develop improved methods for shared peptide analysis.

Chapter 5 is, in full, a reprint of the submitted paper "Accurate Mass Spectrometry Based Protein Quantification via Shared Peptides. B. Dost, N. Bandeira, X. Li, Z. Shen, S. Briggs and V. Bafna". The dissertation author was the primary investigator and author of this paper. This research was also previously published in the conference proceedings of Recomb 2009.

Chapter 6

Relative Quantification of Peptide Modification Variants

Abstract

Accurate identification of large numbers of post translational modification (PTM) sites by nanoscale reverse-phase liquid chromatography/tandem mass spectrometry (LC-MS/MS) remains a major challenge in proteomics. One of the common limitations of mass spectromety-based techniques in PTM identification is co-elution of different positional PTM variants with similar mass and retention times resulting in *mixture* tandem mass spectrum. In this study, we propose a novel computational framework to accurately identify and quantify co-eluting peptide modification variants. Our approach utilizes the presence and absence of fragment ions in MS/MS data as well as some intrinsic properties of fragment ions, *detectabilities*, that cause the variability in the observed peak intensities. We either report a precise site for modification or a mixture of modification variants with accurate relative abundances along with the ambiguities in the site assignment if there is no/little evidence in the tandem mass spectrum.

For evaluation of our quantification results on a MS/MS dataset, we propose a novel target-decoy FDR strategy to measure error rate in PTM site assignment. This strategy allows us to control the FP rate in PTM identifications. We have demonstrated the robustness of our framework on a human lens dataset with a number of different

PTMs. At an estimated $< 2\%$ FDR, we confirm InsPecTs PTM site assignment for 82.1% of cases. In 13.4% of the cases, InsPecT had the right region but the site assignment is actually ambiguous. In the remaining 4.5% of cases, we report a mixture of variants or a site different than Inspect's assignment. In those cases, manual inspection also mostly reveals multiple sites or that InsPecT did not consider the correct site. We validate our framework also on simulated mixtures of up to 3 variants and report high performance of identification and quantification of variants at 1% FDR.

Key words: identification, quantification, post-translational modifications, variants, mass spectrometry, false discovery rate.

6.1 Introduction

Many proteins undergo post-translational modifications (PTMs) that change the properties of a protein by addition or removal of a modifying group to one or more amino-acid residues [93]. The structural diversification enabled by post-translational modification increases the molecular variants of proteins in cells by a few orders of magnitude over the number encoded in the genome [99]. If there are some 30,000 genes transcribed into RNAs and translated into proteins, there may be 300,000 to millions of protein variants at any one time in cells. These protein *modification variants* may differ in modification content and location at one or more amino acid residues within any given protein.

PTMs greatly impact the function of proteins by changing their activity state, localization, turnover, and interactions with other proteins. Identification of proteins with all their post-translational variants is crucial to biologists for understanding the mechanisms of cell regulation. Biological effects are often due to changes on the level of modification, therefore quantitative study of modifications is also of great interest [92, 93].

Mass spectrometry and tandem mass spectrometry (MS/MS) experiments are major tools used in protein identification. It is also suited for detecting post-translational modifications, because modifications produce a diagnostic mass shift on some of the fragment ions in the mass spectrum [94]. However, accurate identification of large numbers of post translational modification (PTM) sites by nanoscale reverse-phase liquid chromatography/tandem mass spectrometry (LC-MS/MS) still remains a major challenge in proteomics.

For instance, precise site localization can be difficult when there are multiple *valid* residues, on which modification can occur, within a single peptide. To resolve the ambiguity between potential sites, fragment ions exclusive to a specific site location must be identified to uniquely assign the modification to a specific site. We refer to these specific fragment ions as *distinguishing ions*. Beausoleil et al. [88] presented a probability-based ambiguity score, the Ascore, that measures the probability of correct modification site localization based on the presence of distinguishing ions in MS/MS spectra. The authors score and rank the modification sites by the likelihood of identifying distinguishing ions compared to random chance. Ascore is in essence the delta score of the top two candidates revealing the certainty/ambiguity in assignment of modification to the top candidate. The authors reported better accuracy and sensitivity in PTM site assignment than Sequest and Mascot at various thresholds chosen for Ascore.

A more complicated challenge in mass spectromety-based PTM identification is co-elution of different positional PTM variants with similar mass and retention times resulting in *mixture* tandem mass spectrum. It is crucial to develop a computational approach to confidently identify all present variants in a mixture tandem mass spectrum with precise site localizations to the extent that MS/MS data allows.

In previous approaches, the presence of the distinguishing ions/peaks has also been used as a proxy for identification of a modification variant present in a mixture tandem spectrum of multiple co-eluting variants. This assumption holds when there are peaks that are uniquely associated with a single ion fragment from a single peptide variant. However, for highly modified peptides, the present modification variants might not have a distinguishing peak, and their abundances might only contribute to shared peaks. Consider a mixture tandem mass spectrum of 3 isobaric peptide modification variants E**H**SSP, EH**S**SP, EHS**S**P with the same modification but on residues H2, S3 and S4, respectively. For representation purposes, MS/MS spectrum data is shown in a bipartite graph where each modification form is a vertex on the left, each peak is a vertex on the right as in Figure 6.1. For simplicity, only the peaks due to b- and y-ions are considered. In each modification variant, the modified residue is highlighted in red and the peaks corresponding to a modified fragment ion is marked with '*'. There is an edge/link between a modification variant and a peak if and only if the variant has a fragment ion with mass corresponding to that peak. Note that one peak may correspond to multiple modified fragment ions as modification at different residues induce the same mass shift in the fragment ions.

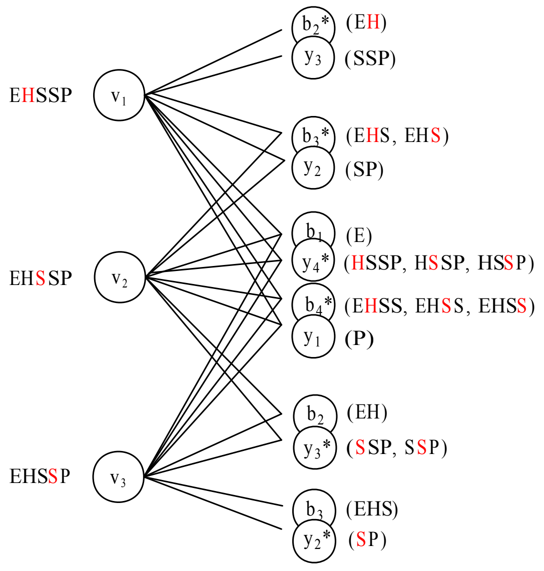


Figure 6.1: A simple example with three isobaric peptide modification variants E**H**SSP, EH**S**SP, EHS**S**P. In each modification variant one of H2, S3 or S4 is modified with the same modification group (e.g. +28). The modified residue is highlighted in red. There are 12 potential peaks induced by b/y ions of the variants. There is an edge between a modification variant and a peak if and only if the variant has a fragment ion with the mass corresponding to that peak. Note that one peak may correspond to multiple fragment ions as the modification can occur at multiple sites. b2* and y3 are the distinguishing peaks of variant E**H**SSP since no other variant have a fragment ion that can lead to that peak. Similarly, b3 and y2* distinguish variant EHS**S**P from the others. However, there are no peaks that would distinguish variant EH**S**SP. All of the peaks induced by EH**S**SP are shared with other variants. Thus, from the mixture tandem mass spectrum of these 3 variants, presence of EH**S**SP would be inconclusive by just looking at the mapping between the peaks and the modification variants.

In this example, b2* and y3 are the distinguishing peaks of variant E**H**SSP since no other variant have a fragment ion that can lead to that peak. Thus, if we see a peak at b2* and/or y3 in the mass spectrum, these peaks can be used as a proxy for the presence of variant E**H**SSP. Similarly, b3 and y2* distinguish variant EHS**S**P from the others. However, there are no peaks that would distinguish variant EH**S**SP. All of the peaks induced by EH**S**SP are shared with other variants. In this simple example, the identification of present modification variants are inconclusive by just looking at the mapping between the peaks and the modification variants.

In this chapter, we address the problem of identification and relative quantification of peptide modification variants given a tandem mass spectrum. Recently, DiMaggio et al. [89] proposed a mixed integer linear optimization (MILP) framework to address the same problem. The authors first solve an NP-hard MILP problem iteratively until they enumerate all of the modification variants and then score each modification variant according to the presence of distinguishing peaks of the variant. In their approach, the highest scoring variant is treated as the dominant present variant. To identify and quantify other present variants, the authors formulate another NP-hard MILP model that superposes multiple spectra with unknown weights to get the mixture tandem mass spectrum. This model is also solved in an iterative fashion until no more variants are identified. However, as we mentioned in the example above, the present variant(s) do not have to induce the distinguishing peaks in the mixture spectrum. Thus, building superposition problem based on the assumption that the modification variant that explains the most of the observed peaks might lead to the incorrect identifications and quantification.

We propose a novel computational framework to address this problem. Given an MS/MS spectrum, peptide sequence and mass of the precursor ion mass, we propose an efficient pseudo-polynomial DP algorithm to enumerate all of the modification variants that satisfy the precursor mass. We then formulate a linear programming (LP) problem to solve for the relative abundances of modification variants. Note that LP problem can be solved efficiently in polynomial time. In our LP formulation, we utilize the presence and absence of fragment ions in MS/MS data as well as some intrinsic properties of fragment ions that cause the variability in the observed peak intensities. We informally define the *detectability* of a fragment ion as the probability that it will be detected via MS, when its parent peptide ion is present in the mixture. We estimate ion detectabilities as an intermediate step to our approach and incorporate them in our LP formulation. Another novel aspect of our approach is that we report ambiguities in the assignment of modification when there is no/little evidence in the MS/MS data for particular variant(s). In other words, we group the variants based on presence or absence of peaks distinguishing between the variants. If two or more variants are indistinguishable with the given data, we report a quantification value for the group instead of the individual variants.

We further extend our approach to address another important proteomics problem: Estimation of False Discovery Rates (FDRs) for PTM site localizations. Note that existing approaches [88, 89, 95] score each PTM site or modification variant but provide little to no guidance on how to set the threshold and how to estimate the FP rates. We propose a new way to define decoy databases specifically for this purpose, allowing one to assign FDRs to site-localization score thresholds. In brief, our approach first ignores the knowledge of which residues are valid modification sites (i.e., decoy = invalid PTM sites), assigns a score to every residue as a putative modification site and then estimates how often each score threshold results in invalid site assignments. Use of target-decoy based FDR strategy not only helps to control the FP rate in the site assignments, but also allows us to identify putative novel modification sites.

6.2 Method

In our recent publication in [90], we have proposed a computational framework for relative quantification of distinct proteins via their shared peptides. Here, we extend this framework to identify and quantify isobaric peptide modification variants with stable modifications in MS/MS data.

Figure 6.2 provides an overview of our approach. The input to our approach is a mixture MS/MS spectrum S_{mix}, an unmodified MS/MS spectrum S, unmodified peptide p, precursor mass M and a set of modifications $Mods$ where

1. p is the dominant unmodified peptide identification of S_{mix}
2. M is the precursor mass of S_{mix} from MS^1
3. $Mods$ is the set of modifications that might occur on the residues of p
4. S is an unmodified MS/MS spectrum identified as p.

Given the precursor ion mass M and the set of modifications $Mods$, we first enumerate all of the modification variants of p that satisfy the precursor mass M within some tolerance. As a second step, we infer the detectabilities of fragment ions of modification variants using the unmodified spectrum S. Given the mixture spectrum S_{mix}, modification variants and the inferred ion detectabilities, we estimate the relative abundances of variants in the third step using a linear programming formulation. Finally, we group the indistinguishable modification variants based on the mixture spectrum S. When there is no or very little detected peaks distinguishing two modification variants in the mixture spectrum, it is impossible for LP to differentiate between the two variants as well. In such case, the individual variant abundances are not accurate but the total abundance for the group. We output the relative abundances of variant groups instead of individual variant groups.

In the following subsections, we discuss the different steps spelled out in the flowchart in detail.

6.2.1 Enumeration of Modification Variants

As a first step to our method, given the dominant peptide identification p from MS^2, the precursor mass M from MS^1, and the set of modifications $Mods$, we enumerate all modification variants of p that satisfy the precursor ion mass M. This can be done for each MS/MS spectrum by simple combinatorial enumeration of variants in exponential running time in the number of modifications. We propose a novel efficient algorithm to solve this problem.

We consider a peptide $p = a_1a_2 \ldots a_L$ of length L, and a set of modifications $Mods = \{x_j = (a_{i_j}, m_j)\}$, each defined by its amino acid specificity a, and induced non-zero mass offset $m \in \mathbb{R}$. Let $mass(p) = mass(a_1, a_2, \ldots, a_L)$ be the total mass of peptide p, defined as $\sum_{i=1..L} mass(a_i)$ for unmodified peptides. Using $a_i^{m_i}$ to indicate a modification of mass m_i on the i-th amino acid, the mass of a modified peptide p is defined as $\sum_{i=1..n} mass(a_i) + m_i$ where $m_i = 0$ for unmodified amino acids.

Given a peptide p, set of modifications $Mods$, and precursor mass $M \in \mathbb{R}$, the problem of modification variant enumeration is to list all modified peptide sequences

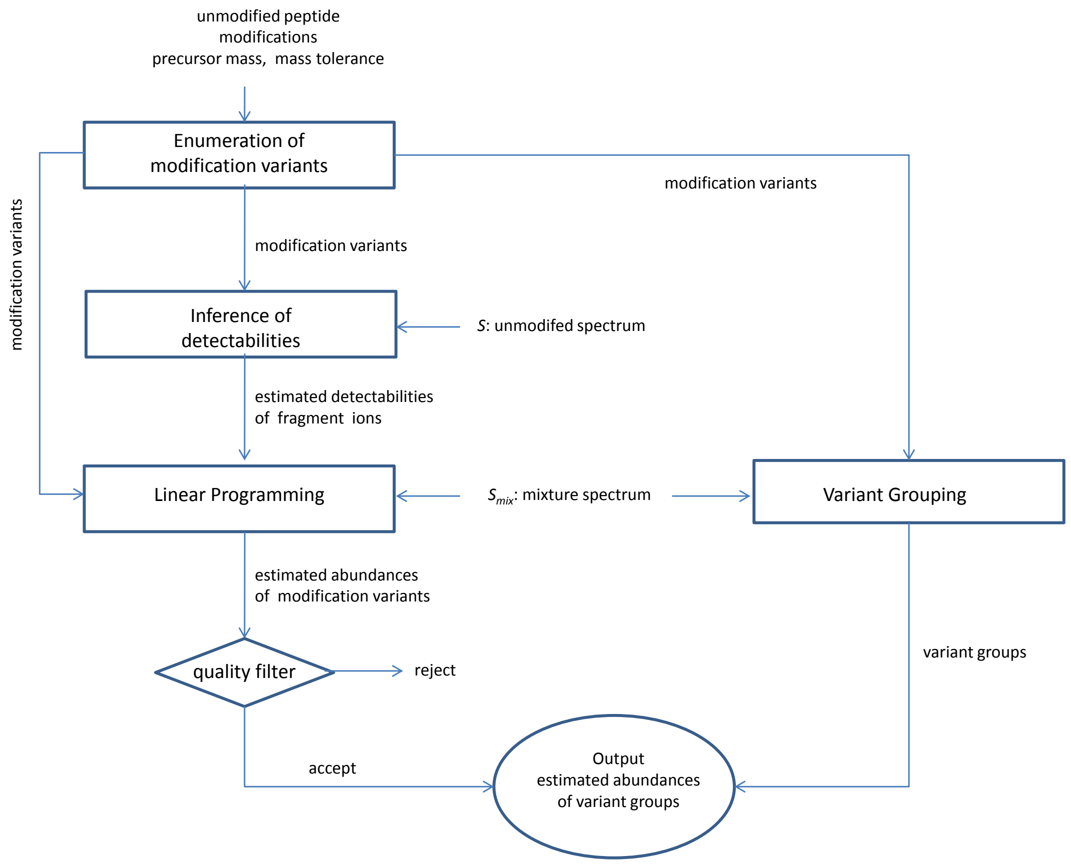

unmodified peptide
modifications
precursor mass, mass tolerance
Enumeration of modification variants
modification variants
modification variants
modification variants
Inference of detectabilities
S: unmodifed spectrum
estimated detectabilities of fragment ions
Linear Programming
S_{mix}: mixture spectrum
Variant Grouping
estimated abundances of modification variants
variant groups
quality filter
reject
accept
Output estimated abundances of variant groups

Figure 6.2: Flowchart of our approach.

$\Pi(p, Mods, M) = \{\pi^1, \pi^2, \ldots, \pi^{n_\pi}\}$ such that

- π_i can be derived from p by modifying amino acids with modifications in $Mods$,
- $mass(\pi_i)$ is $\in [M - t, M + t]$ for a mass tolerance $t \in \mathbb{R}$, and
- the number of modifications per amino-acid a_i is at most 1.

Each of these modified peptide sequences is referred to as a *modification variant* of p.

This problem is a more generalized version of Subset-Sum Problem which is known as an NP-Complete Problem. In this problem, the items are the possible modifications in $Mods$ with values m_j that correspond to modification masses and the target sum to be achieved is the difference between the precursor mass M and the mass of the unmodified peptide p. However, the modification masses are not limited to positive integers, they can be non-integers and negative as well. Also, we need to consider the error in the estimation of the precursor mass M by allowing the target sum to be within range $[\Delta M - t, \Delta M + t]$ where $\Delta M = M - mass(p)$.

In the worst case, $|\Pi(p, Mods, M)| = 2^{|Mods|}$ and a naive exponential-time recursive solution that goes through all subsets of $Mods$ will enumerate all the variants that satisfy the given precursor mass in $O(2^{|Mods|})$ time. However, in practice the number of variants that satisfy the precursor mass M is much less than $2^{|Mods|}$. We propose the pseudo-polynomial time DP algorithm, shown in Figure 6.2.1, where the subproblems are defined on the subsets of $Mods$ and target values smaller than ΔM. In the algorithm, before starting to solve subproblems, we first scale up the problem with a factor f so that the modification masses m_j, mass difference M and tolerance t are all integers. The running time of this algorithm is $O((M + t) \cdot f \cdot |Mods| \cdot n_\pi)$ where $n_\pi = |\Pi(p, Mods, M)|$ is the size of the solution.

Note that the algorithm given in Figure 6.2.1 works for positive modification masses. It can be easily generalized for also negative values by solving the subproblems for the target values from T^- to T^+ where T^- is the sum of negative modification masses and T^+ is the sum of positive modification masses. Then, the running time becomes $O((T^+ - T^-) \cdot f \cdot |Mods| \cdot n_\pi)$ which is still polynomial in the number of modifications and the size of the solution.

```
ENUMERATE-MODIFICATION-VARIANTS()
1  $Input:$ peptide $p$, Mods $= \{x_1, \cdots, x_n\}$, precursor mass $M$,
2   mass tolerance $t$, scaling factor $f$
3  **Output:** all $S \subset$ Mods s.t. mass$(p) + \sum_{x_k \in S} m_k \in [M - t, M + t]$
4  $\Delta M \leftarrow M - mass(p)$
5  $\Delta M \leftarrow \lfloor \Delta M \cdot f \rfloor$
6  $t \leftarrow \lfloor t \cdot f \rfloor$
7  **for** $j = 1$ to $n$
8  **do** $m_j \leftarrow \lfloor m_j \cdot f \rfloor$
9   Let A be an array of size $\Delta M + t + 1$
10  $A[m_1] \leftarrow \{\{x_1\}\}$, $A[i] \leftarrow \emptyset$ for all $i \neq m_1$
11 **for** $j = 2$ to $n$
12 **do for** $i = \Delta M + t$ down to $m_j$
13     **do for** $S \in A[i - m_j]$
14         **do** $A[i] \leftarrow A[i] \cup \{S \cup \{x_j\}\}$
15 **return** $\bigcup_{i=(\Delta M - t):(\Delta M + t)} A_i$
```

Figure 6.3: Pseudo-polynomial dynamic programming algorithm to enumerate all modification variants.

6.2.2 Inference of Fragment Ion Detectabilities

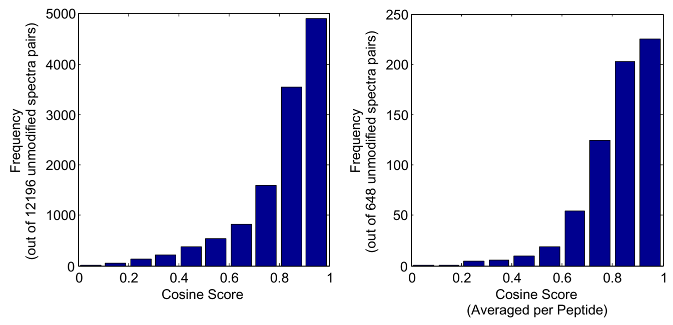
The point of our approach is that the observed intensity of a peak is directly related to the total abundance of the theoretical fragment ions that have the same corresponding m/z value, thus to the total abundance of the modification variants that contain those isobaric ions. Simply, if there is a peak in the spectrum which is associated with a single modification variant through a single ion, the peak intensity is a direct measurement for the abundance of the modification variant. However, the fragment ions have different detectabilities in the mass spectrometry depending on the physico-chemical properties of the fragment ions. For our purposes, only the relative contributions of the fragment ions to the peak intensities per unit abundance of its parent peptide are important. So, we define *detectability* of a fragment ion as its expected intensity as a percentage of most intense/detectable ion of the peptide.

For inference of fragment ion detectabilies, we are motivated by conservation of spectral shapes of a peptide sequence across different unmodified and modified tandem mass spectra. Previously, Sniatynski et al. [96] showed that delay-series correlation between the mass spectra of modified and unmodified peptides revealed significant spectral overlap at an offset indicative of the modification. This observation has been confirmed in many other publications including one of ours [87]. We observe similar behavior in also our test MS/MS dataset from human lens proteins here. We measure the similarity between two spectra using *cosine score* which is a widely accepted measure for spectral similarity. Given two unmodified spectra S_1, S_2 and an unmodified peptide sequence p, we compute the cosine score of S_1 and S_2 with respect to p as follows. We first list all of the theoretically possible ions from the unmodified peptide p and therefore the theoretical m/z values. We then form two vectors v_1 and v_2 from S_1 and S_2 respectively where each vector denotes the peak intensities observed at the theoretical m/z values in the corresponding spectrum. The cosine score is computed as

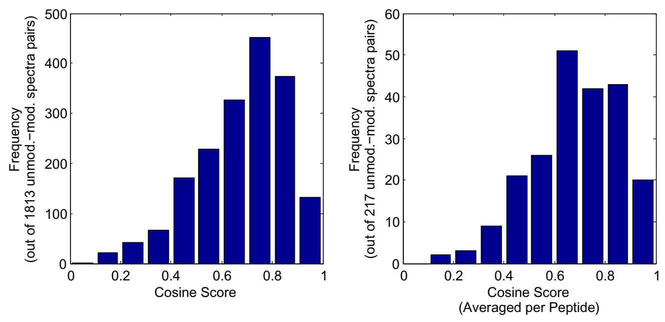
$$cos_p(S_1, S_2) = \frac{v_1 \cdot v_2}{\|v_1\| \cdot \|v_2\|}.$$

In Figure 6.4a, we show the distribution of the cosine score similarities of unmodified spectra pairs that are identified as the same peptide using Inspect. 93.6% of the pairs have more than 0.5 cosine similarity confirming that the fragment ion detectabilities are conserved across unmodified spectra. In a similar way, we compute the cosine similarity of an unmodified spectrum of a peptide with a modified spectrum of the same peptide after shifting the modified fragment ions back to their m/z values without the modifications. The distribution of spectral similarities of modified-unmodified spectra pairs is shown in Figure 6.4b. We see that even after modification, there is a significant spectral overlap between the spectra pairs of the same peptides. 83.3% of the pairs shows more than 0.5 cosine similarity.

The second step of our method is to list all (modified/unmodified) theoretical fragment ions and to estimate their detectabilities. Let $\mathcal{I}(\Pi) = \{\iota_i : (f_i, m_i)\}$ be the set of theoretical fragment ions expected from the modification variants in Π. Each fragment is defined by its sequence(with modifications if any) f_i and theoretical mass m_i.



a)



b)

Figure 6.4: Distribution of cosine score (a) two unmodified spectra of the same peptide (b) an unmodified spectrum and a modified spectrum with of the same peptide with a single modifications from the set (+16 on M or W; -17 on N-terminal Q;+43 on N-terminal *, R and K)

The sequence and mass information can be easily obtained from modification variants by enumerating all subsequences of the variants in $O(kL)$ time where k is the number of variants and L is the length of the peptide.

To infer the fragment ion detectabilities, we recruit an unmodifed MS/MS spectrum S_{unmod} of the peptide p which is the dominant peptide identification of S_{mix} from MS^2. Since there is only one possible parent peptide for the ions detected in an unmodified MS/MS spectrum, we know that the direct ratio of the annotated peak intensities denotes the relative detectabilities of their corresponding unmodified fragments. We first annotate S_{unmod} with peptide p. If a peak is annotated with a single fragment ion, we assign the peak intensity to the detectability of the fragment ion. If a peak is annotated by multiple fragment ions, it is possible to choose from several strategies such as splitting the peak intensity among the ions according to their estimated ion probabilities or PRM scores, etc. Our results did not differ much with different strategies, so we adopted a simpler strategy. If a peak is annotated by multiple fragment ions, we assign the whole intensity to the ion with largest ion probability. For every other fragment ion not detected in the unmodified spectrum, we assign a detectability of $\epsilon > 0$. In our tests we used $\epsilon = 1$.

High spectral overlap indicates that the relative detectability of the modified and unmodified versions of fragments are conserved with respect to the other fragments. We thus assume that the detectability of different modified forms of a fragment ion are roughly the same. Therefore, for every theoretical modified fragment ion, we assign the same detectability value that we computed for its unmodified form.

6.2.3 Quantification of Modification Variants via Linear Programming

The mixture spectrum of isobaric modification variants is the superposition of individual modified spectra from all of the modification variants. Thus, a peak intensity does not necessarily correspond to fragment ion(s) from a single parent peptide. Often, multiple fragment ions from one or more variants have the same m/z value and they contribute to the same peak. For instance, in the simple example shown in Figure 6.1, in the presence of all variants, 3 different b4 ions from 3 variants are contributing to the same peak. The overlap gets even more when other types of ions and the ions with neutral losses get into the picture. It is crucial to see the mapping between the peaks and the theoretical fragment ions as well as the mapping between variants and the fragment ions. Each observed intensity value in the mixture spectrum at a theoretical m/z value gives information about the total abundance of the fragment ions from variants contributing to that peak.

We first compile a set of observed peaks where each peak corresponds to a pair of m/z and intensity value. Let $S = (z_i, X_i)$ be a *spectrum* defined as a set of *peaks* with mass and observed intensity values. Given the set of theoretical ions $\mathcal{I}$, the mixture tandem mass spectrum S_{mix}, we form the consensus set of observed peaks as follows. Let $\mathcal{P}(\mathcal{I}, S_{mod}) = \{(z_i, X_i)\}$ be the set of peaks, each defined by the mass value z_i and observed intensity value X_i. For each ion $\iota_i = (f_i, m_i)$ in $\mathcal{I}$ where f_i is the sequence of the ion and m_i is the theoretical mass,

- if there is a peak (z, X) in S_{mod} such that $|z - m_i| \leq t$, add (z, X) to $\mathcal{P}$
- otherwise, add $(m_i, 0)$ to $\mathcal{P}$.

Note that we add an m/z value with intensity zero to the set of observed peaks for each peaks that we would expect to see if the associated fragment ion(s) were present in the sample.

In order to visualize the one-to-many mapping from variants to fragment ions and many-to-one mapping from ions to peaks, we represent the tandem spectrum data using a variant-ion-peak graph $G = (\Pi \cup \mathcal{I} \cup \mathcal{P}, E)$ where Π is the set of modification variants and $\mathcal{I}$ is the set of theoretical ion fragments and $\mathcal{P}$ is the set of peaks observed in MS^2.

We associate a variable Q_i corresponding to the abundance of $\pi^i \in \Pi$ in the mixture. For all $\pi^i \in \Pi$ and $\iota_j : (f_j, m_j) \in \mathcal{I}$, $(\pi^i, \iota_j) \in E$ if and only if fragment ion f_j is contained as a substring of modification variant v_i. For all $\iota_j \in \mathcal{I}, \rho_k : (z_k, X_k) \in \mathcal{P}$, $(\iota_k, \rho_k) \in E$ if and only if the mass z_k of peak ρ_k is within mass tolerance t of the theoretical mass m_j of ion fragment ι_j, i.e. $|z_k - m_i| \leq t$.

For each peak ρ_k, we would expect

$$X_k \approx \sum_{(\iota_j, \rho_k) \in E} \sum_{(\pi^i, \iota_j) \in E} d_j \times Q_i$$

where d_j is the estimated detectability of ion ι_j. To account for noise in mass spectrometric data, we associate each peak with an error term ε_j as

$$\varepsilon_k = X_k - \sum_{(\iota_j,\rho_k)\in E} \sum_{(\pi^i,\iota_j)\in E} d_j \times Q_i$$

and we consider a linear-programming formulation that minimizes the total error. Given detectabilities d_j and abundance measurements X_k, for all peaks, we solve for optimal abundance variables Q_i for all modification variants that minimizes the sum of the error terms ε_j. The complete formulation is shown in Figure 6.5.

Input	Output	Formulation
variant-ion-peak graph G d_j for every ion ι_j X_k for every peak ρ_k	Q_i for every variant v_i	$\min \sum_{k=1}^{r} \lvert\varepsilon_k\rvert$ s.t. $\varepsilon_k = X_k - \left(\sum_{(\iota_j,\rho_k)\in E} \sum_{(\pi^i,\iota_j)\in E} d_j \times Q_i \right)$ for all ρ_k $Q_i \geq 0$ for all π^i. $Q_i s$ are normalized prior to output so that $\sum_i (Q_i) = 1$

Figure 6.5: Input, output, and computation summary of LP formulation for quantification of variants. π^i denotes a isobaric modification variant, ι_j denotes an ion and ρ_k denotes a peak. Q_i is the abundance variable for modification form π^i. X_k and d_j are the intensity of the peak ρ_k and detectability of the ion ι_j, respectively.

6.2.4 Evaluation of an Individual Result

In case of incorrect peptide identifications, the unmodified and the modified spectrum are substantially different. In such a case, LP is not able to find a solution that fits the observed mixture spectrum and the inferred detectabilities well. We apply a quality filter on the LP solution to eliminate such cases.

If the variant abundance estimates fit the observed mixture spectrum well, we expect a high spectral similarity between the observed mixture and the theoretical mixture spectrum induced by the estimated variant abundances. Thus, once we solve for the relative abundances of the modification variants, we compute the cosine score between the observed mixture and the theoretical mixture. If the score is low, we reject the solution. In our tests, we use a threshold of 0.4, since more than 90% of the cases, as seen in Figure 6.4, the cosine score between two spectrum of the same peptide is above 0.4.

6.2.5 Grouping Modification Variants

LP outputs an estimated abundance per modification variant. However, depending on the completeness of the fragmentation pattern, there might be no or very little *distinguishing peaks* between two modification variants. In that case, it is impossible for LP to distinguish between the variants in its abundance assignment. In absence of distinguishing ions, some of the estimates for individual variant(s) will not be accurate, but for the groups of variants. Therefore, we group the *indistinguishable variants*, the variants that do not have enough/any distinguishing peaks in between, and report total estimated abundance per variant group instead of individual variants.

Given a spectrum S, modification variants $\pi_1, \cdots, \pi_n$ and a threshold t, we output the groups of variants as follows. For every pair of variants (π_i, π_j), we compute distance d_{ij} as the sum of the intensities of the peaks that are annotated with only one of the variants π_i and π_j divided by the total intensity of all peaks in the spectrum S. Based on the computed variant distances, we do single-linkage hierarchical clustering of the variants and output the clusters at threshold t as the groups of indistinguishable variants. In our experiments, we used a variant distance threshold of 5%.

Grouping variants allows us to report ambiguous site identification of modifications when it is impossible to distinguish between the sites by the detected peaks in the spectrum.

6.3 Results

6.3.1 Evaluation of Dataset Results

In peptide identification, to evaluate the performance of search algorithms on a dataset, use of target and decoy databases is widely practiced. The target-decoy search strategy permits an impartial assessment of search results and by applying a score cutoff, FPs and false discovery rate (FDR) can be controlled at a desired level. We adopt a similar target-decoy strategy to determine the global false-discovery rate (FDR) for modification variant identifications.

Ideally, we would run LP formulation on only modification variants that are *valid*, i.e. variants that assign the modification(s) to valid residues/sites. In order to assess FDR, we create a set of variants comprising of both valid and invariant variants, and run LP on this composite set of variants and output the quantification results for groups of those variants. These variant groups construct our target and decoy database as we quantify variant groups instead of individual variants. If a group has at least one valid variant, it is called *valid group* and it is added to the target. If a group does not have any valid variant, it is an *invalid group* and added to the decoy. Thus, if an invalid group is assigned abundance by LP, it is considered a FP identification while an valid group identified is considered a TP.

FDR estimation by target-decoy strategy is coupled with a scoring scheme. Each instance in target and decoy is assigned a score. For a chosen score cutoff, FDR measures the percentage of incorrectly identified instances (FPs) in the final set accepted at that cutoff. In our target-decoy approach for FDR, we use the estimated abundances assigned by LP to score each variant group. This is a reasonable choice since we would have more confidence in the presence of a group if its estimated abundance is high. If the estimated abundance is low, it is more likely that the abundance is assigned due to contaminant peaks or noise. Note that it is possible to devise different scoring schemes but it is not the focus of this chapter.

FDR estimation is based on the assumption that the probability that a random instance is identified incorrectly from the decoy database is expected to be the same as the probability that it is identified incorrectly from the target database when the sizes of the decoy database and the target database are the same. Thus, FDR at a score/estimated abundance cutoff is calculated as the number (N_d) of variant group identifications from decoy divided by the number (N_t) of group identifications from target , i.e., FDR = N_d/N_t. N_d is an estimate of the number of FPs resulting from random identifications in target database.

Use of target-decoy strategy not only helps to control the FP rate in the variant group identifications, but also allows us to identify putative novel modification sites. Decoy hits might give information about novel modification sites.

6.3.2 Application on Lens Dataset

In this section, we validate our approach on MS/MS dataset of peptides with known modification sites. We blind ourselves to the knowledge of the modification site and evaluate how well our approach assigns all the abundance to the only correct modification site.

The dataset consists of human lens proteins from a 93 year old human male with nuclear cataract. A major component of the lens proteome comprises of crystallins, which have very little turnover, and acquire modifications with age. When a person ages, the crystallins become insoluble, and the tissue increasingly opaque often leading to cataract. Post-translational modifications are known to play a major role in the process [98]. Mass spectrometry data ($160,940$ spectra) from human lens proteins were acquired on a ESI ion trap mass spectrometer. This dataset was also a subset of that used by Wilmart et al. [100] and will be referred as 'lens dataset'. The

The spectra were searched for peptide identifications by running $\mathrm{Inspect}$ [97], against human lens protein database with parent mass tolerance 2 Da and fragment mass tolerance 0.5; 1% false discovery rate was enforced using a standard target/decoy strategy [91]. We allowed at most one modification from the set of modifications previously identified in the dataset by Wilmart et al [100] and commonly occurring modifications. In total, $22,011$ spectra were annotated. For the reported results, we used only the

$15,487$ spectra that were identified as tryptic charge +2 peptides. $13,094$ of them were identified as unmodified peptides and $2,393$ of them as modified peptides. The unmodified peptide identifications mapped onto 898 unique peptides and the modified ones onto 297 unique peptides. For 217 of these unique modified peptides, unmodified version of the peptide was also identified by at least one spectrum. In our analysis, we focused on those 217 modified spectra. The breakdown of these spectra by the identified modification is shown in Table 6.1.

Table 6.1: Dataset by modifications

Modification	Number of Cases	Valid Sites
+16	68	M or W
-17	20	N-terminal *
+43	129	N-terminal *, R or K
Total	217	

We apply our method on each identified modified peptide p' of which unmodified version is also identified. Let p be the unmodified version of p', m be the mass of the modification. We choose the highest-scoring (MQScore) unmodified and modified spectra pair (S, S') that were identified as p and p' by $\mathrm{Inspect}$, respectively. We blind ourselves to p'. Given (S, S'), p and m, we would like to identify p' as the only present modification variant.

In order to estimate FDR as described in Section 6.3.1, we also blind ourselves to the site specificity of the modification m. We enumerate all possible modification variants of p with a single modification m allowing it to occur on any residue. Thus, we obtain L modification variants $p'_1, \cdots, p'_L$ where L is the length of the peptide p. Then, we enumerate all theoretical ion fragments, infer fragment ion detectabilities and then generate the LP system based on the observed peak intensities and detectabilities as described in Section 6.2.3. We solve each LP system estimating the relative abundance of modification variants. Finally, we group the modification variants using variant distance threshold of 5% and output the estimated abundances per variant group.

The variants and variant groups are marked as valid or invalid according to their validity in the assignment of modification. The number of cases and valid sites con-

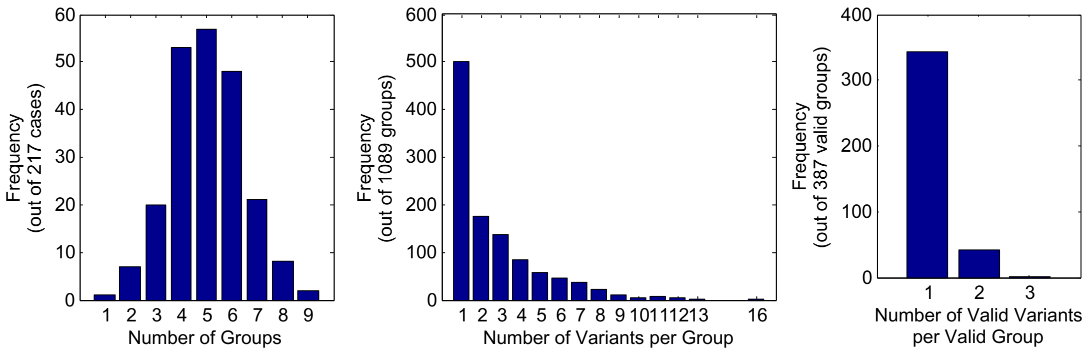


Figure 6.6: Distribution of (a) number of variant groups (b) size of variant groups (c) number of valid variants per valid group.

sidered per modification are shown in Table 6.1. The distribution of number of variant groups per LP system, number of variants per group and number of valid variants per valid group are shown in Figure 6.6. Average number of valid variants per valid group is 1.11. Out of 387 valid variant groups, 344 of them had only 1 valid variant, 42 had 2 valid variants and 1 had 3. In majority (88.9%) of the cases, when we locate the true valid group we also locate the true valid variant.

Based on the valid and invalid groups, which form the target and the decoy, we estimate FDRs for a range of score/estimated abundance cutoffs. In Table 6.2, FDRs by score cutoffs and the percentage of the cases with $1 - 4$ identified groups are listed. As seen, FDR goes up only up to 4% while the threshold for the estimated abundance can go as low as 20%. Also, even at a very low FDR rate, 1.1%, we are able to identify at least one valid variant group for more than 96.5% of the cases.

For further investigation, we choose the abundance cutoff of 32% at which we achieve less than 2% FDR. In Table 6.3, we present the detailed results of modification variant identification at $< 2\%$ FDR. In 95.5% of the cases $(192/201)$, we successfully identify the modification variant group identified by $\mathrm{Inspect}$ as the *only* abundant group. In $51/192$ cases, there is only one valid site for the modification. In all of those cases, we identify the valid variant as the *only* abundant variant. Out of $141/192$ cases with more than one valid site, we report

- precise site assignment for $80.9\%(114/141)$ of the cases

Table 6.2: FDR Results on lens dataset. At cosine score cutoff = 0.4, 16 cases rejected, 201 cases accepted.

Estimated Abundance of Variant Group	FDR	#Cases with k groups identified			
		$k \geq 1$	$k = 1$	$k = 2$	$k = 3$
$\geq 55\%$	0.011	96.5%	96.5%	0	0
$\geq 50\%$	0.014	97.0%	97.0%	0	0
$\geq 45\%$	0.012	97.0%	96.5%	0.5%	0
$\geq 40\%$	0.016	97.5%	97.0%	0.5%	0
$\geq 35\%$	0.016	98.0%	97.0%	1.0%	0
$\geq 34\%$	0.016	98.5%	97.5%	1.0%	0
$\geq 32\%$	0.019	98.5%	97.0%	1.5%	0
$\geq 30\%$	0.024	98.5%	96.5%	2.0%	0
$\geq 25\%$	0.024	99.0%	96.5%	2.5%	0
$\geq 20\%$	0.035	99.5%	96.0%	3.5%	0

- ambiguous site assignment for $19.1\%(27/141)$ of the cases.

Note that this result suggests that FP rate can be reduced by up to 19.1% when ambiguous site assignments are removed. Using our method, we not only report more accurate and trustworthy site assignments, but we are also able to identify mixtures of variants and estimate relative abundances. Our method also has the potential to find novel modification sites through the decoy hits.

Below, we discuss the rest of the cases $(9/201)$ where we did not agree with $\mathrm{Inspect}$'s site assignment in 3 categories: (1) mixtures, (2) corrected site assignments and (3) incorrect site assignments.

Mixtures At FDR 2%, we identify 3 mixtures of 2 variant groups. In all mixture cases, each identified group has only one valid variant. So, we unambiguously assign the modifications to the sites. The identified modification variants and the estimated abundances for those mixture cases are listed in Table 6.4. The annotated mixture spectra are presented in Figures 6.7, 6.8 and **??**.

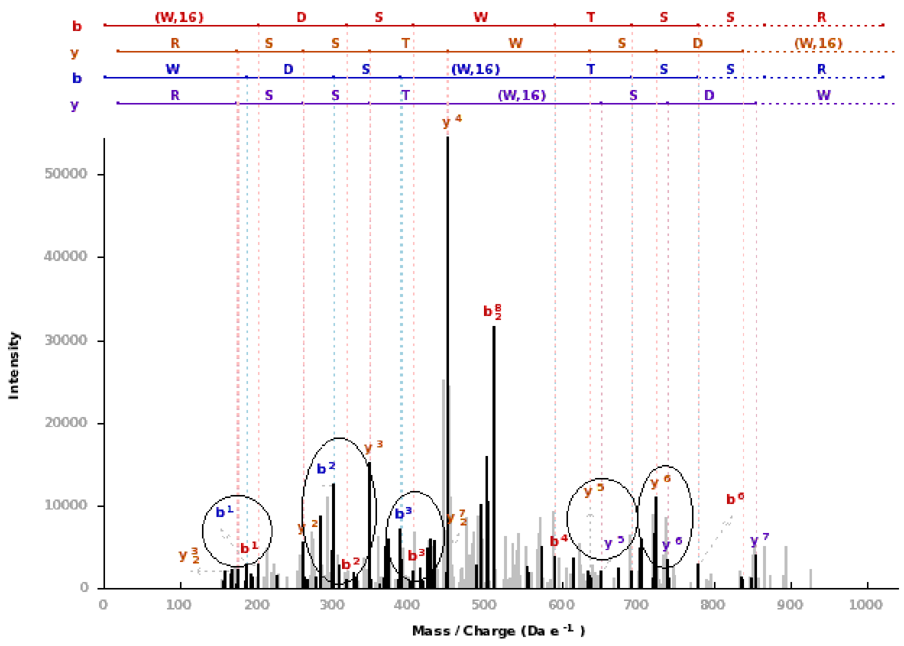


Figure 6.7: An identified mixture of 2 modification variants of WDSWTSSR. The variant with +16 on N-terminal W and the variant with +16 on W4 are identified at ratio 64%:36%. For simplicity, only b- and y- ions are shown. The distinguishing peaks supporting the co-existence of the two variants are circled.

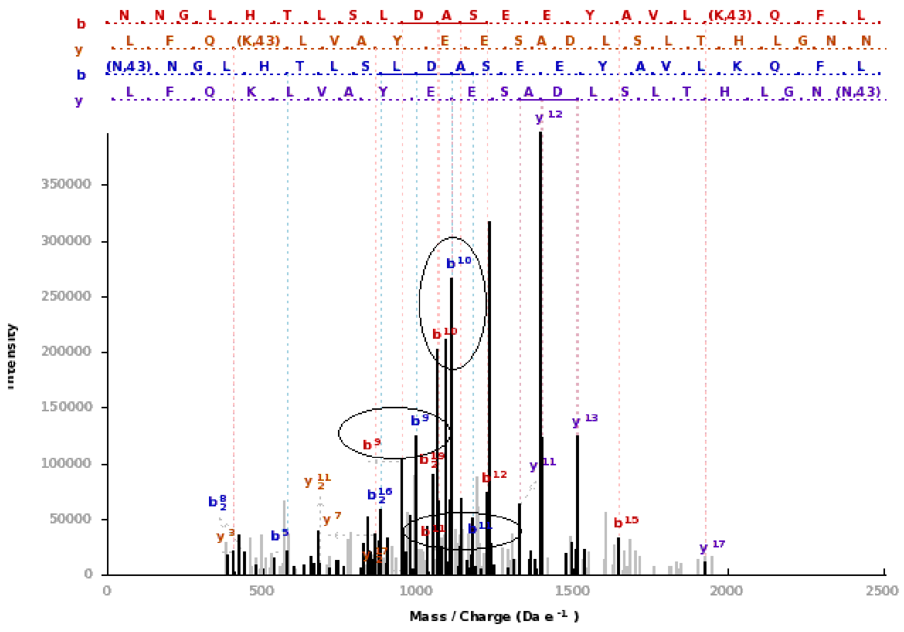


Figure 6.8: An identified mixture of 2 modification variants of NNGLHTLSLDASEEYAVLKQFL. The variant with +43 on N1 and the variant with +43 on K19 are identified at ratio 60%:40%. For simplicity, only b- and y- ions are shown. The distinguishing peaks supporting the co-existence of the two variants are circled.

Table 6.3: Results for Modification Variant Identification at 2% FDR.

	mod.	#valid groups	#valid groups identified	rank of the Inspect id.	#valid variants in the top group	#cases
precise site assignments	16	1	1	1	1	33
	-17	1	1	1	1	17
	43	1	1	1	1	1
	16	2	1	1	1	18
	16	3	1	1	1	1
	43	2	1	1	1	84
	43	3	1	1	1	11
ambiguous site assignments	16	1	1	1	2	7
	16	2	1	1	2	2
	16	2	1	1	3	1
	43	1	1	1	2	2
	43	2	1	1	2	15
mixtures	16	2	2	2	1	1
	43	3	2	2	1	1
	43	4	2	2	1	1
corrected assignments	43	3	1	-	1	2
	43	3	1	-	2	1
incorrect assignments	16	1	0	-	-	1
	43	1	0	-	-	1
	43	2	0	-	-	1

Corrected Assignments In 3 cases, we identify a variant group different than Inspect's identification. In each case, the identified group has only one valid variant, thus we unambiguously assign the modification to a single distinct variant. Inspect's and our site assignment of the modification for these cases are listed in Table 6.5. For one of the cases, the modified spectrum annotated with both of the identifications is also presented in Figure 6.10. The distinguishing peaks supporting the existence of our assignment and the ones supporting Inspect's assignment are enclosed in circles and squares, respectively. As we can see from the figure, there is more evidence for our identification. Note that, in this case, LP assigns 80% to our identification while it assigns 20% to Inspect's. Thus, it can be also interpreted as a mixture at a higher FDR (4%).

Table 6.4: Identified mixtures at FDR 2%

peptide	modification variant	estimated abundances
NNGLHTLSLDASEEYAVLKQFL	+43 on N-terminal +43 on K19	60% 40%
VLGDVIEVHGKHEER	+43 on N-terminal +43 on R15	55% 45%
WDSWTSSR	+16 on N-terminal +16 on W4	64% 36%

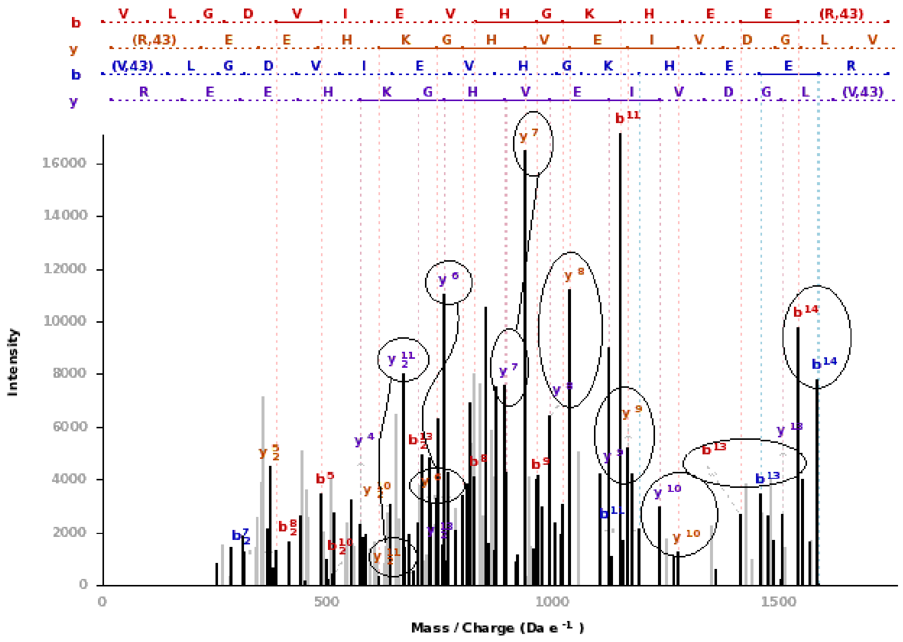


Figure 6.9: An identified mixture of 2 modification variants of VLGDVIEVHGKHEER. The variant with +43 on V1 and the variant with +43 on R15 are identified at ratio 56%:44%. For simplicity, only b- and y- ions are shown. The distinguishing peaks supporting the co-existence of the two variants are circled.

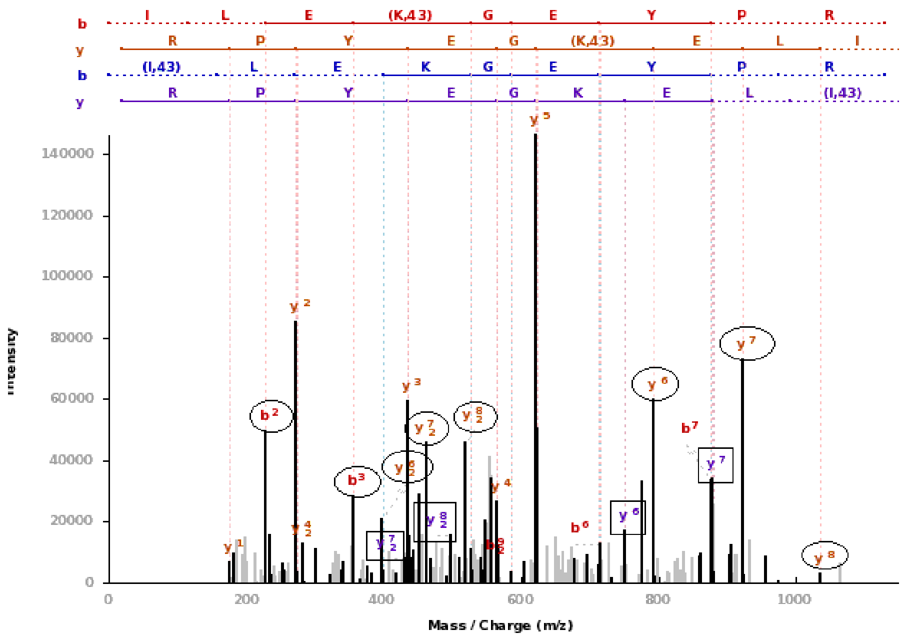


Figure 6.10: A corrected Inspect site assignment. The modified spectrum is identified as I[+43]LEKGEYPR by Inspect. At $<$ 2% FDR, we identify the variant ILEK[+43]GEYPR instead. The distinguishing peaks supporting the existence of ILEK[+43]GEYPR and the ones supporting Inspect's assignment are enclosed in circles and squares, respectively.

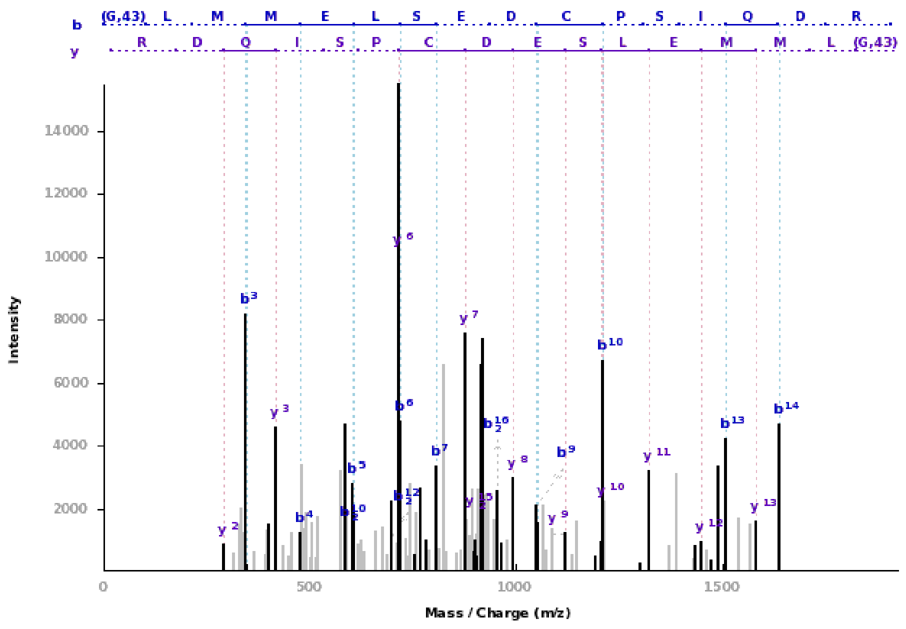
Table 6.5: Corrected Inspect Identification at FDR $< 2\%$

peptide	Inspect identification	our identification
ILEKGEYPR	+43 on N-terminal	+43 on K4
TDSLSSLRRPIKVDSQEHK	+43 on N-terminal	+43 on K11
SDRDKFVIFLDVK	+43 on N-terminal	+43 K5

Incorrect Site Assignments In 3 cases, LP assigns all of the abundance to the invalid groups. Thus, we can not identify any variant for those cases. The reasons of these incorrect identifications are listed below:

1. Discrepancy between unmodified spectrum and mixture spectrum:
   The distinguishing fragment ions between the actual variant group and its neighboring group are not identified in the unmodified spectrum. So, the inferred detectability is very low or none. Then, in LP optimization, the distinguishing ions do not influence the solution, and in the optimal solution some/all of the abundance is given to the neighboring group. The 2 of the false negative cases fall in this category. The modified spectrum and the unmodified spectrum used for one of those two cases are shown in Figure 6.11a,b. Inspect identifies the variant that assigns the modification to N-terminal G. In our grouping, based on the modified spectrum, the variants that assign the modification to N-terminal G, L2, M3 form the 1st group and the variants that assign the modification to M4, E5, L6 form the 2nd group. LP solution assigns all of the abundance to the 2nd group instead of the 1st group even though there is an intense peak at b3* that distinguishes between those two variant groups. Because, b3 ion is not detected in the unmodified spectrum, its detectability is very low. LP can not distinguish between the two groups and incorrectly assigns the abundance to the 2nd group.

2. Contaminant peaks or novel modification site:
   If LP assigns abundance to an invalid variant group and there are peaks supporting the existence of the invalid group, the solution can be interpreted in two ways. Either (1) LP identifies the group incorrectly due to contaminant peaks or (2) There is indeed a variant that assigns the modification to a novel residue. In one of the false negative cases, we do not identify the Inspect's pick that assigns +43 mod-



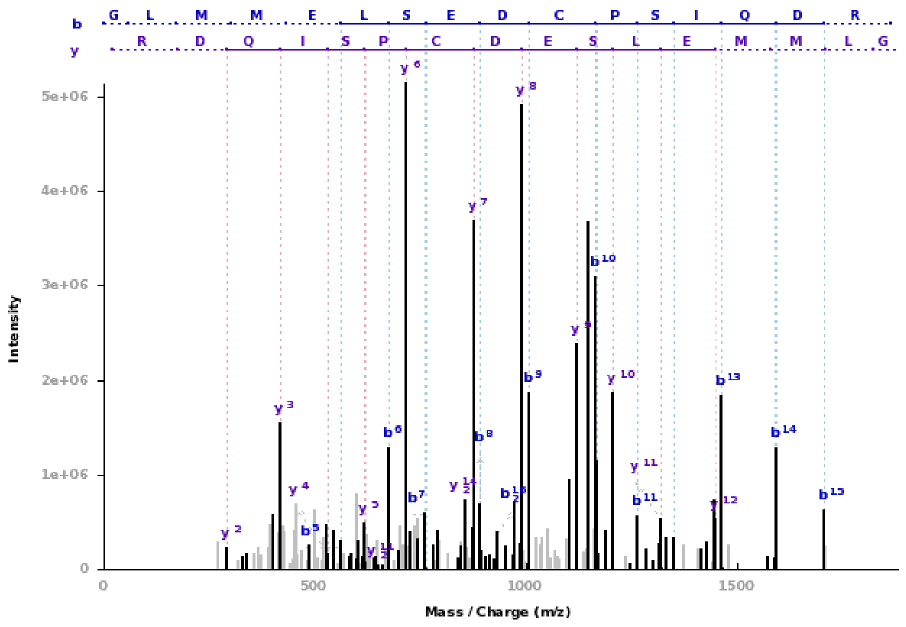


Figure 6.11: An incorrect site assignment due to discrepancy between the modified spectrum and the unmodified spectrum. (a) Modified spectrum identified as G+43LMMELSEDCPSIQDR and (b) unmodified spectrum identified as GLMMELSEDCPSIQDR by Inspect. In our grouping based on the modified spectrum, the variants that assign the modification to N-terminal G, L2, M3 form the 1st group and the variants that assign the modification to M4, E5, L6 form the 2nd group. LP solution assigns all of the abundance to the 2nd group instead even though there is an intense peak at b3* that distinguishes between those two variant groups. Because, since b3 ion is not detected in the unmodified spectrum, its detectability is very low, and the LP can not distinguish between the two groups.

ification on N-terminal R but the variant that assigns the modification to F4. The modified spectrum annotated with both of the variants are shown in Figure 6.12. The peaks that are supporting the presence of the variant that assign modification to F4 are circled. This case can be interpreted as an incorrect identification due to contaminant peaks or an identification of ”F” as a novel modification site for +43.

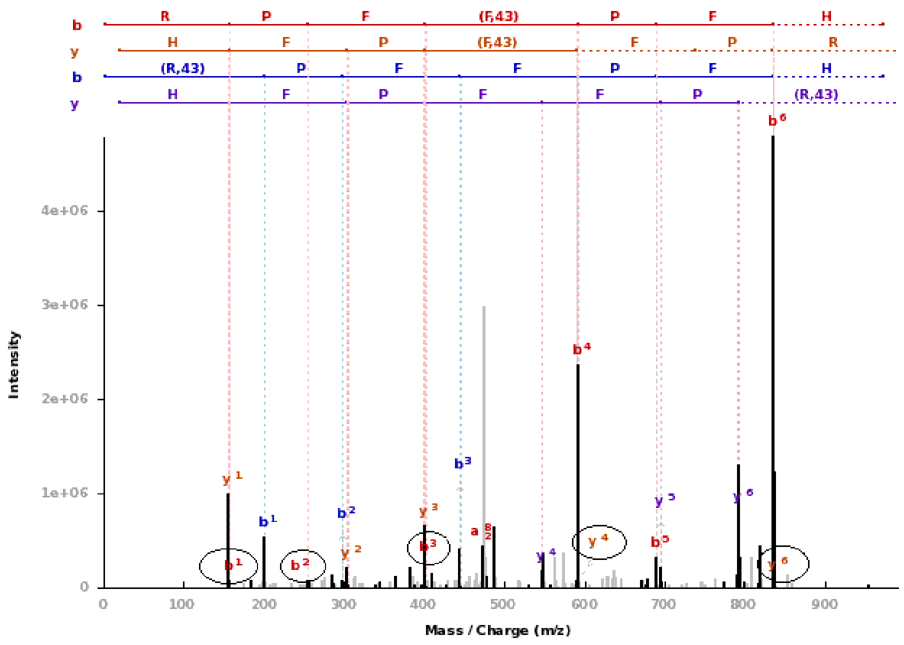


Figure 6.12: An incorrect identification due to contaminant peaks or identification of a novel modification site. The modified spectrum is identified as R[+43]PFFPFH by Inspect. LP assigns all of the abundance to the variant that assigns the modification to F4. The annotation of the spectrum as R[+43]PFFPFH and RPFF[+43]PFH are shown in one plot. The peaks that support the presence of 2nd variant are circled. This case can be interpreted as a misidentification due to contaminant peaks or an identification of ”F” as a novel modification site for +43.

6.4 Simulation

We apply our framework on simulated data to test how well we are able to identify the present modification variants and recover their actual relative abundances from a mixture MS/MS spectrum. We generate simulated data based on the 212 unique unmodified peptides which were identified from the lens dataset by at least two spectra. For each unmodified peptide, we create mixture spectra of its variants which are singly modified with $+43$. We allow the modification to occur on only valid sites (N terminal *, R, K) to account for also the distribution of the number and position of the sites in real data. The distribution of the number of valid sites of $+43$ for our set of peptide sequences are shown in Table 6.6.

Table 6.6: Number of valid sites of +43 in the set of peptide sequences chosen for simulation.

#valid sites	#cases
≥ 1	212
≥ 2	208
≥ 3	58
$== 4$	4

Each simulated dataset consists of an unmodified peptide sequence, an unmodified spectrum to be used for inference of detectabilities, and a mixture spectrum. Given unmodified peptide p, we generate 100 different such datasets of $k \leq 3$ co-eluting peptide variants as follows:

1. Among the MS/MS spectra identified with the same unmodified peptide p, we choose top two highest scoring ($\mathrm{Inspect}$ MQscore) spectra S_1 and S_2. We use S_1 to generate mixture spectrum and S_2 to be used for inference of detectabilities later.

2. Let n be the number of valid sites in p. For $k = 1$ to n, we do the following 100 times to generate a series of mixture spectra of k variants.

    (a) We select k random modification sites on p and k random relative abundances $\geq 20\%$ (summing to 1), one for each modification variant.

    (b) We create k versions of S_1, each with total intensity multiplied by its relative abundance. For each version, we shift all ion masses to the new mass positions induced by modification and randomly redistribute the noise (non-annotated peaks).

    (c) We add up all k versions of S_1 to create the mixture spectrum $M(S_1)$

In total, we obtain $21200(44.7\%)$, $20800(43.9\%)$, $5400(11.4\%)$ mixture spectra of 1,2, and 3 variants, respectively. Note that since some of the present variants might be grouped together, the percentages of cases with actual $1, 2, 3$ variant groups might be different from those.

We then solve the LP using S_2 as the unmodified spectrum to infer detectabilities and $M(S_1)$ as the mixture spectrum to obtain the relative abundances for each of the k variants. Note that in order to estimate FDR we blind ourselves to the site specificity of the modification. In the LP formulation, we include all possible singly modified variants of p allowing modification to occur on any residue. We group indistinguishable variants using variant distance threshold of 5% and output the estimated abundances per variant group.

The distribution of number of variant groups per LP system, number of variants per group and number of valid variants per valid group are shown in Figure 6.13. Note that the average number of valid variants in a valid variant group in the simulations is 1.1. When we identify the correct variant group, we identify the correct variant in 90.0% of the cases. In the rest, the modification site is ambiguous and the group has 2 (9.7% of cases) or 3 (0.3% of cases) variants.

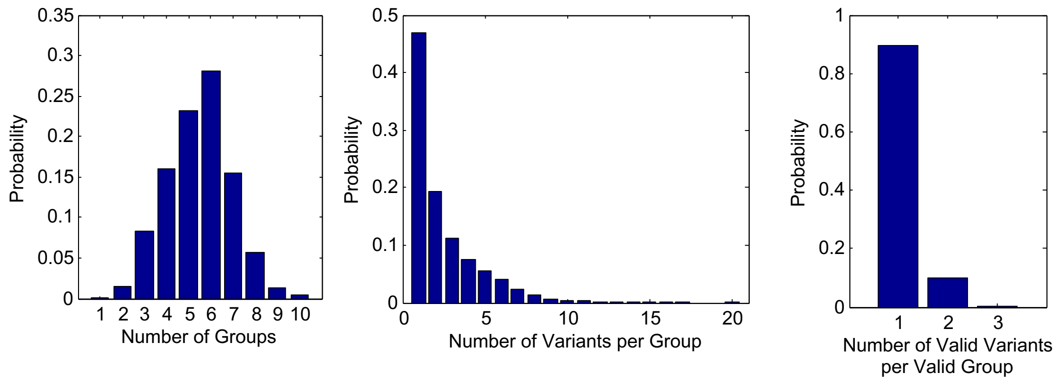


Figure 6.13: Distribution of (a) number of variant groups (b) size of variant groups (c) number of valid variants per valid group in the simulated data.

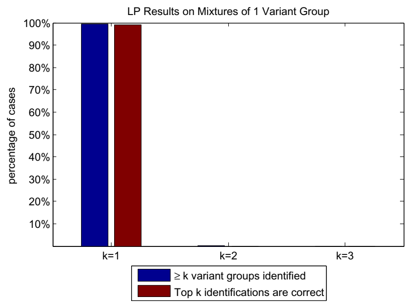
We pool results from all of the datasets with mixtures of $k = 1, 2, 3$ variants and evaluate our method on (1) identification (2) quantification of variant groups.

As described earlier, we are able to control FP rate on variant identification using target-decoy strategy. In Table 6.7, we show the FDRs for a range of cutoffs on the estimated abundance of the variant groups. We also show the percentage of the cases with $k = 1 - 4$ identified groups (estimated abundance $\geq$ cutoff) and the percentage of the cases with k actual groups (actual abundance $\geq$ cutoff). In ideal sensitivity, the two percentages should be equal. As we see, the percentages inferred from the LP results closely follow the actual percentages. For example, at 1% FDR (estimated abundance $\geq 20\%$) cutoff, we identify at least one variant group for 98.9% of the cases; 1 variant group in 51.0%, 2 variant groups in 45.2% and 3 groups in 2.7% of the cases. In the ideal case, we would identify exactly $1, 2$ and 3 variant groups in $47.0\%, 47.6\%$ and 5.7% of the cases, respectively. This result suggests that the sensitivity of our approach is close to the optimal.

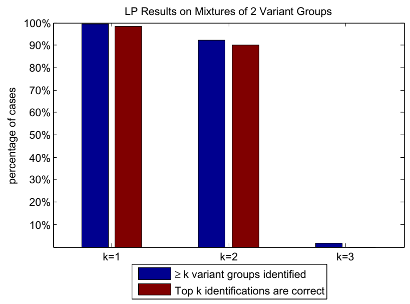
Table 6.7: Estimated FDRs of our approach on simulated data for a range of cutoffs on the estimated abundance of the variant groups. At each FDR level, the percentage of the cases with $k = 1 - 4$ identified groups (estimated abundance $\geq$ cutoff) and (in parenthesis) the percentage of the cases with k actual groups (actual abundance $\geq$ cutoff) are displayed.

Estimated Abundance	FDR	#Cases with k groups identified				
of Variant Group		$k \geq 1$	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\geq 60\%$	0.002	74.9% (79.2%)	74.9% (79.2%)	0.0% (0.0%)	0.0% (0.0%)	0.0%
$\geq 55\%$	0.002	82.7% (87.1%)	82.7% (87.1%)	0.0% (0.0%)	0.0% (0.0%)	0.0%
$\geq 50\%$	0.003	90.9% (95.6%)	90.8% (95.6%)	0.1% (0.0%)	0.0% (0.0%)	0.0%
$\geq 45\%$	0.003	94.6% (96.9%)	89.6% (89.1%)	5.0% (7.8%)	0.0% (0.0%)	0.0%
$\geq 40\%$	0.003	96.7% (98.6%)	84.5% (83.2%)	12.1% (15.4%)	0.0% (0.0%)	0.0%
$\geq 35\%$	0.004	98.1% (99.9%)	77.7% (75.7%)	20.4% (24.2%)	0.0% (0.0%)	0.0%
$\geq 30\%$	0.005	98.7% (100.0%)	68.8% (65.5%)	29.7% (34.1%)	0.1% (0.3%)	0.0%
$\geq 25\%$	0.007	98.9% (100.0%)	59.2% (55.9%)	38.7% (42.0%)	1.0% (2.1%)	0.0%
$\geq 20\%$	0.010	98.9% (100.0%)	51.0% (47.0%)	45.2% (47.6%)	2.7% (5.4%)	0.0%
$\geq 15\%$	0.019	99.0% (100.0%)	48.7% (47.0%)	46.6% (47.6%)	3.7% (5.4%)	0.0%
$\geq 10\%$	0.036	99.0% (100.0%)	47.9% (47.0%)	46.7% (47.6%)	4.4% (5.4%)	0.0%
$\geq 5\%$	0.077	99.0% (100.0%)	47.6% (47.0%)	46.4% (47.6%)	4.9% (5.4%)	0.0%

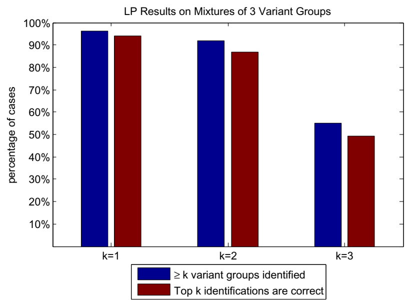
In order to measure the accuracy of the variant identifications, we calculate how often the variant groups identified are correct. We call a variant group identification *correct*, if the group was initially assigned a non-zero abundance (i.e. $\geq$ min. abundance$= 20\%$). In Figure 6.14, we compare the percentage of the cases where we identify at least k variant groups and the cases where *top* (most abundant) k identifications are correct at FDR 1% for $k = 1, 2, 3$. As shown in Figure 6.14a, when there is only one variant group present in the sample, we identify a single variant group in almost all (99.3%) of the cases with very high accuracy (99.9%). In Figure 6.14b,c the identification results on mixtures of 2 and 3 variant groups are shown. We achieve similarly very good performance in identification of both variant groups from a mixture of 2 groups. We are able to correctly identify both groups in 90.0% of the cases. In presence of 3 variants, percentage of identification of all variants drops to 55.2% but with 89.2% accuracy.



a)



b)



c)

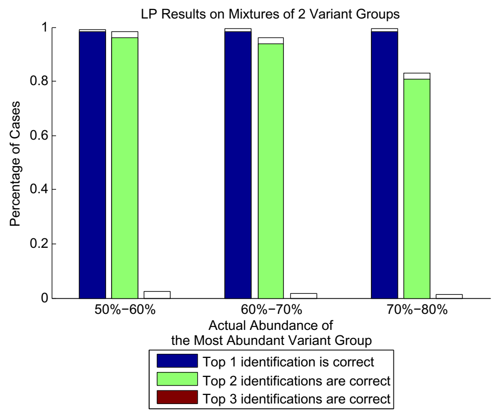
Figure 6.14: Identification results on mixtures of a) 1, b) 2 and c) 3 variant groups at FDR=1%. The percentage of the cases where we identify at least k variant groups and the cases where *top* (most abundant) k identifications are correct for $k = 1, 2, 3$.

We further investigate how the accuracy of the identifications change by the abundance of the variant groups. Figure 6.15 demonstrate the percentage of $\geq k(1-3)$ identifications (white bars) and the correct top k identifications (blue, green, red bars for k=1,2, and 3, respectively) grouped by the abundance of actual most abundant variant group. As we can see, in the mixtures of 2 variant groups, the percentage of the correct top 2 identifications decreases from 95.9% to 80.6% while the percentage of correct top 1 identification is almost the same ($> 98\%$) suggesting that as the relative abundance of the 2nd most abundant variant gets smaller, we are more likely not to identify the $2nd$ variant group. Note that the accuracy is still more than 90% for the correct 2nd variant group identification. Similarly, in the mixtures of 3 variant groups, as the relative abundances of the 2nd and 3rd most abundant variants gets smaller, the percentage of the identifications of 2 or more groups decreases from 92.1% to 88.9% and of 3 variants from 60% to 43.5. However, in all those cases, accuracy is above 85%.

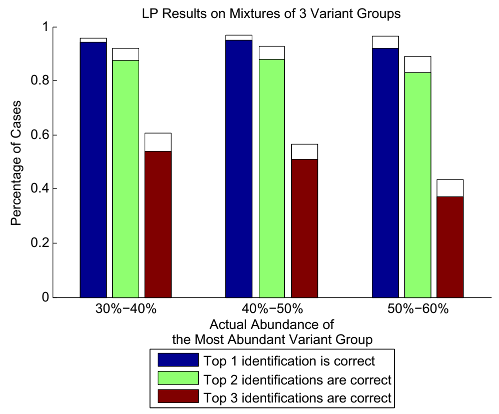
Besides identification of the present variants, our approach also reports relative abundances of the identified variant groups. We evaluate our quantification results by comparing the estimated and actual abundance of the identified variant groups in mixtures of 2 and 3 groups. The boxplots of log ratios of the estimated abundances of identified variant groups to their actual abundances, shown in Figure 6.16ab, are constructed with 5, 25, 50, 75, and 95 percentiles. We observe that 90% of the variant groups are accurately quantified with less than 0.5 and 1 fold difference of their actual abundance in mixtures of 2 and 3 variant groups, respectively.

6.5 Discussion

Large scale post translational modification (PTM) site assignment using mass-spectrometry based techniques is a challenging problem in proteomics. One of the common limitations of in PTM identification is co-elution of different positional PTM variants with similar mass and retention times resulting in mixture tandem mass spectrum. We propose a novel computational framework to accurately identify and quantify co-eluting peptide modification variants. Given a mixture spectrum, our method assigns

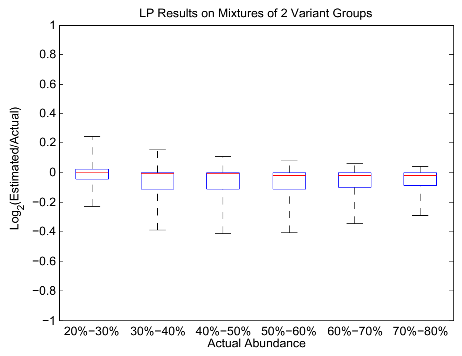


a)

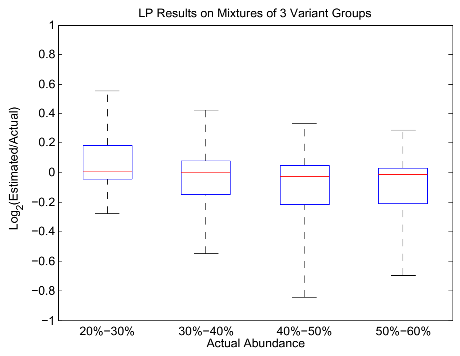


b)

Figure 6.15: Identification results on mixtures of a) 2 and b) 3 variant groups at FDR=1%. Colored bars show the percentage of the cases where *top* (most abundant) k $(1-3)$ identifications are correct grouped by the actual abundance of the most abundant the variant group. Non-colored (white) bars indicate the percentage of the cases with $\geq k$ group identifications.



a)



b)

Figure 6.16: Quantification results on mixtures of a) 2 and b) 3 variant groups at FDR=1%. Box plot displays the distribution of the log ratio of the estimated abundance to the actual abundance of the variant groups. The bottom and top of the box are the 25th and 75th percentile (the lower and upper quartiles, respectively), and the band near the middle of the box is the 50th percentile (the median). The ends of the whiskers represent the 5th and 95th percentiles.

the modification(s) to specific site(s) with confidence or determines a mixture of multiple modification variants with accurately estimated relative abundances.

In PTM identification and quantification, our approach is first to utilize absence of the peaks in addition to the presence of peaks and their intensities in tandem mass spectrum based on the inferred detectabilities of the fragment ions in mass spectrometers. We recruit an unmodified spectra of the peptide sequence of which fragment ion detectabilities will be inferred. Relative detectabilities are assigned to ions based on their relative peak intensities in the spectra. We resolve ambiguity in detectability assignment when a peak is annotated by multiple fragment ions by simply discarding the annotation with lower ion probability. However, detectability inference can be improved by recruiting multiple unmodified spectra and/or devising better strategies to resolve the cases with ambiguous peak annotations.

Another novel aspect of our approach is that we report ambiguities in the assignment of modification when there is no/little evidence in the MS/MS data for particular variant(s) and it can not assign the modification confidently. In such a case, we cluster the variants based on the peaks distinguishing between the variants and output identification and quantification results for the clusters/groups instead of individual variants.

Grouping of the modification variants can be studied more systematically by exploring different distance metrics and many existing clustering techniques.

Traditional methods in PTM identification do not provide a way to measure error rates in large scale PTM analysis. We propose a novel target-decoy strategy to estimate false discovery rates (FDRs) for site-localization score thresholds. In our FDR strategy, we used the estimated abundances of variant groups as our scoring scheme but it is possible to devise different scoring schemes for different applications.

Use of target-decoy based FDR strategy not only helps to control the FP rate in the site localizations, but also has great potential to detect novel sites of less studied or unknown modifications. Site specificity blind-quantification of PTM variants will elucidate potential sites through decoy hits.

Chapter 6 is in preparation for publication as "Identification and Quantification of Post-translational Modification Variants and False Discovery Rates". B. Dost, V. Bafna and N. Bandeira (2010). In preparation". The dissertation author was the primary author of this paper.

References

[1] P. L. Adams, M. R. Stahley, A. B. Kosek, J. Wang, and S. A. Strobel. Crystal structure of a self-splicing group I intron with both exons. *Nature*, 430(6995):45–50, Jul 2004.

[2] T. Akutsu. Dynamic programming algorithm for RNA secondary structure prediction with pseudoknots. *Disc. Appl. Math.*, 104:45–62, 2000.

[3] L. Argaman et al. Novel small RNA-encoding genes in the intergenic regions of *Escherischia coli*. *Curr. Biol.*, 11:941–950, 2001.

[4] V. Bafna, S. Muthukrishnan, and R. Ravi. Computing similarity between RNA strings. *Combinatorial Pattern Matching*, 937:1–14, 1995.

[5] P. V. Baranov, C. M. Henderson, C. B. Anderson, R. F. Gesteland, J. F. Atkins, and M. T. Howard. Programmed ribosomal frameshifting in decoding the SARS-CoV genome. *Virology*, 332(2):498–510, Feb 2005.

[6] Y. Ben-Asouli, Y. Banai, Y. Pel-Or, A. Shir, and R. Kaempfer. Human interferon-gamma mRNA autoregulates its translation through a pseudoknot that activates the interferon-inducible protein kinase PKR. *Cell*, 108(2):221–232, Jan 2002.

[7] A. Condon, B. Davy, B. Rastegari, F. Tarrant, and S. Zhao. Classifying RNA Pseudoknotted Structures. *Theoretical Computer Science*, 320(1):35–50, 2004.

[8] R. M. Dirks and N. A. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *J Comput Chem*, 24(13):1664–1677, Oct 2003.

[9] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis*, chapter 10.3 Covariance models: SCFG-based RNA profiles. Cambridge University Press, 1998.

[10] S. Eddy. Non-coding RNA genes and the modern RNA world. *Nature Reviews in Genetics*, 2:919–929, 2001.

[11] P. Evans. *Algorithms and Complexity for Annotated Sequence Analysis*. PhD thesis, University of Victoria, Victoria BC, Canada, 1964.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W.H. Freeman and Company, 1979.

[13] S. Griffiths-Jones, S. Moxon, M. Marshall, A. Khanna, S. R. Eddy, and A. Bateman. Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res*, 33(Database issue):121–124, Jan 2005.

[14] J. Jaeger, D. Turner, and M. Zuker. Improved prediction of secondary structures for RNA. *Proceedings of the National Academy of Sciences*, 86:7706–7710, 1989.

[15] T. Jiang, G. Lin, B. Ma, and K. Zhang. A general edit distance between RNA structures. *J Comput Biol*, 9:371–388, 2002.

[16] R. Klein and S. Eddy. Rsearch: Finding homologs of single structured rna sequences. *BMC Bioinformatics*, 4(1):44, 2003.

[17] H. Matsui, K. Sato, and Y. Sakakibara. Pair stochastic tree adjoining grammars for aligning and predicting pseudoknot RNA structures. *Bioinformatics*, 21:2611–2617, 2005.

[18] P. L. Nixon, A. Rangan, Y.-G. Kim, A. Rich, D. W. Hoffman, M. Hennig, and D. P. Giedroc. Solution structure of a luteoviral P1-P2 frameshifting mRNA pseudoknot. *J Mol Biol*, 322(3):621–633, Sep 2002.

[19] C. D. Novina and P. A. Sharp. The RNAi revolution. *Nature*, 430(6996):161–164, Jul 2004. News.

[20] B. Rastegari and A. Condon. Linear time algorithm for parsing rna secondary structure. In *5th Workshop on Algorithms in Bioinformatics (WABI)*, 2005.

[21] T. Rastogi, T. L. Beattie, J. E. Olive, and R. A. Collins. A long-range pseudoknot is required for activity of the Neurospora VS ribozyme. *EMBO J*, 15(11):2820–2825, Jun 1996.

[22] E. Rivas and S. Eddy. A Dynamic Programming Algorithm for RNA Structure Prediction Including Pseudoknots. *Journal of Molecular Biology*, 285:2053–2068, 1999.

[23] G. Storz. An expanding universe of noncoding RNAs. *Science*, 296(5571):1260–1263, May 2002.

[24] C. A. Theimer, C. A. Blois, and J. Feigon. Structure of the human telomerase RNA pseudoknot reveals conserved tertiary interactions essential for function. *Mol Cell*, 17(5):671–682, Mar 2005.

[25] A. Vitreschak, D. Rodionov, A. Mironov, and M. Gelfand. Riboswitches: the oldest mechanism for the regulation of gene expression? *Trends in Genetics*, 20(1):44–50, 2003.

[26] Z. Weinberg and W. L. Ruzzo. Faster genome annotation of non-coding rna families without loss of accuracy. In *Proceedings of the Annual Intl. Conference on Computational Biology (RECOMB)*, 2004.

[27] G. D. Williams, R. Y. Chang, and D. A. Brian. A phylogenetically conserved hairpin-type 3' untranslated region pseudoknot functions in coronavirus RNA replication. *J Virol*, 73(10):8349–8355, Oct 1999.

[28] W. C. Winkler and R. R. Breaker. Genetic control by metabolite-binding riboswitches. *Chembiochem*, 4(10):1024–1032, Oct 2003.

[29] S. Zhang, I. Borovok, Y. Aharonowitz, R. Sharan, and V. Bafna. A Sequence-Based Filtering Method for ncRNA Identification and its Application to Searching for Riboswitch Elements. Manuscript, 2005.

[30] S. Zhang, B. Hass, E. Eskin, and V. Bafna. Searching genomes for non-coding rna using fastr. *IEEE Transactions on Computational Biology and Bioinformatics*, 2(4):366–379, 200.

[31] M. Zuker and D. Sankoff. RNA secondary structures and their prediction. *Bull. Math. Biol.*, 46:591–621, 1984.

[32] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995.

[33] M. Ashburner et al. The gene onthology consortium. gene onthology: Toll for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[34] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. B*, 57:289–300, 1995.

[35] J. Berg, M. Lassig, and A. Wagner. Structure and evolution of protein interaction networks: A statistical model for link dynamics and gene duplications. *Bio. Med. Center Evolutionary Biology*, 4:51, 2001.

[36] P. Dent, A. Yacoub, P. B. Fisher, M. P. Hagan, and S. Grant. Mapk pathways in radiation responses. *Oncogene*, 22(37):5885–5896, Sep 2003.

[37] S. F and Z. R. Identifying active transcription factors and kinases from expression data using pathway queries. *Bioinformatics*, 21(Suppl 2):ii115–ii122, Sep 2005.

[38] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., San Francisco, 1979.

[39] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W.H. Freeman and Company, 1979.

[40] U. Guldener, M. Munsterkotter, M. Oesterheld, P. Pagel, A. Ruepp, H.-W. Mewes, and V. Stumpflen. MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Res*, 34(Database issue):436–441, Jan 2006.

[41] E. Hirsh and R. Sharan. Identification of conserved protein complexes based on a model of protein network evolution. In *Fifth European Conference on Computational Biology (ECCB'06)*, 2006. To appear.

[42] T. Ito, T. Chiba, and M. Yoshida. Exploring the yeast protein interactome using comprehensive two-hybrid projects. *Trends Biotechnology*, 19:23–27, 2001.

[43] M. Kanehisa, S. Goto, S. Kawashima, Y. Okuno, and M. Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Res*, 32(Database issue):277–280, Jan 2004.

[44] B. P. Kelley, R. Sharan, R. M. Karp, T. Sittler, D. E. Root, B. R. Stockwell, and T. Ideker. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc Natl Acad Sci U S A*, 100(20):11394–9, 2003.

[45] T. Kloks. *Treewidth: computations and approximations*. Springer-Verlag, 1994.

[46] M. Mann, R. Hendrickson, and A. Pandey. Analysis ures of proteins and proteomes by mass spectrometry. *Annu. Rev. Biochem*, 70:437–473, 2001.

[47] H. W. Mewes, D. Frishman, K. F. Mayer, M. Munsterkotter, O. Noubibou, P. Pagel, T. Rattei, M. Oesterheld, A. Ruepp, and V. Stumpflen. MIPS: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Res*, 34(Database issue):169–172, Jan 2006.

[48] R. Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–8, 2005.

[49] T. Shlomi, D. Segal, E. Ruppin, and R. Sharan. QPath: A Method for Querying Pathways in a Protein-Protein Interaction Network. *BMC Bioinformatics*, 7(199), 2006.

[50] C. A. Stanyon, G. Liu, B. A. Mangiola, N. Patel, L. Giot, B. Kuang, H. Zhang, J. Zhong, and J. Finley, R. L. A Drosophila protein-interaction map centered on cell-cycle regulators. *Genome Biol*, 5(12):R96, 2004.

[51] I. Xenarios, D. W. Rice, L. Salwinski, M. K. Baron, E. M. Marcotte, and D. Eisenberg. DIP: the database of interacting proteins. *Nucleic Acids Res*, 28(1):289–91, 2000.

[52] E. Homsher A. M. Gordon and M. Regnier. Regulation of contraction in striated muscle, 2000.

[53] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25(1):25–29, May 2000.

[54] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.

[55] Molly A. Bogue, Stephen C. Grubb, Terry P. Maddatu, and Carol J. Bult. Mouse phenome database (mpd). *Nucleic Acids Research*, 35(Database-Issue):643–649, 2007.

[56] K Brix, P Lemansky, and V Herzog. Evidence for extracellularly acting cathepsins mediating thyroid hormone liberation in thyroid epithelial cells. *Endocrinology*, 137(5):1963–1974, May 1996.

[57] Angels Almenar-Queralt Velia M. Fowler Catharine A. Conley, Kimberly L. Fritz-Six. Leiomodins: Larger members of the tropomodulin (tmod) gene family. *Genomics*, 73(1):127–139, May 2001.

[58] Steven Delvaux and Leon Horsten. On best transitive approximations to simple graphs. *Acta Inf.*, 40(9):637–655, 2004.

[59] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W.H. Freeman and Company, 1979.

[60] Laurent Gautier, Leslie Cope, Benjamin M. Bolstad, and Rafael A. Irizarry. affy—analysis of affymetrix genechip data at the probe level. *Bioinformatics*, 20(3):307–315, 2004.

[61] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. Yang, and J. Zhang. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol*, 5(10), 2004.

[62] M. Gerstein and R. Jansen. The current excitement in bioinformatics - analysis of whole genome expression data: how does it related to protein structure and function, 2000.

[63] F D Gibbons and F P Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res*, 12(10):1574–1581, Oct 2002.

[64] David Gibson, Ravi Kumar, and Andrew Tomkins. Discovering large dense subgraphs in massive graphs. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 721–732. VLDB Endowment, 2005.

[65] Curtis Huttenhower, Avi I Flamholz, Jessica N Landis, Sauhard Sahi, Chad L Myers, Kellen L Olszewski, Matthew A Hibbs, Nathan O Siemers, Olga G Troyanskaya, and Hilary A Coller. Nearest neighbor networks: clustering expression data based on gene neighborhoods. *BMC bioinformatics*, 8:250, 2007 2007.

[66] Minoru Kanehisa, Michihiro Araki, Susumu Goto, Masahiro Hattori, Mika Hirakawa, Masumi Itoh, Toshiaki Katayama, Shuichi Kawashima, Shujiro Okuda, Toshiaki Tokimatsu, and Yoshihiro Yamanishi. Kegg for linking genomes to life and the environment. *Nucl. Acids Res.*, pages gkm882+, December 2007.

[67] R. Motwani and P. Raghavan. *Randomized algorithms.* Cambridge University Press, 1995.

[68] R. Project. *The R project for statistical computing*, 2003.

[69] J. Quackenbush. Computational analysis of microarray data. *Nat Rev Genet*, 2(6):418–427, June 2001.

[70] Sven Rahmann, Tobias Wittkop, Jan Baumbach, Marcel Martin, Anke Truss, and Sebastian Böcker. Exact and heuristic algorithms for weighted cluster editing. *Comput Syst Bioinformatics Conf*, 6:391–401, 2007.

[71] E. E. Schadt, S. A. Monks, T. A. Drake, A. J. Lusis, N. Che, V. Colinayo, T. G. Ruff, S. B. Milligan, J. R. Lamb, G. Cavet, P. S. Linsley, M. Mao, R. B. Stoughton, and S. H. Friend. Genetics of gene expression surveyed in maize, mouse and man. *Nature*, 422(6929):297–302, March 2003.

[72] Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Appl. Math.*, 144(1-2):173–182, 2004.

[73] Nan Song, Jacob M. Joseph, George B. Davis, and Dannie Durand. Sequence similarity network reveals common ancestry of multidomain proteins. *PLoS Comput Biol*, 4(4), Apr 2008.

[74] Ingenuity Systems. Ingenuity systems pathway analysis. www.ingenuity.com.

[75] Zhou Yingyao, A. Young, Santrosyan Andrey, Chen Kaisheng, Yan S. Frank, and A. Winzeler. In silico gene function prediction using ontology-based pattern identification. *Bioinformatics*, 21(7):1237–1245, April 2005.

[76] P. Alves, R.J. Arnold, M.V. Novotny, P. Radivojac, J.P. Reilly, and H. Tang. Advancement in protein inference from shotgun proteomics using peptide detectability. *Pac Symp Biocomput*, pages 409–420, 2007.

[77] M Bantscheff, M Schirle, G Sweetman, J Rick, and B Kuster. Quantitative mass spectrometry in proteomics: a critical review. *Anal Bioanal Chem*, 389(4):1017–1031, Oct 2007.

[78] *An Advanced Treatise on Meloidogyne: Volume I.* North Carolina State University Graphics, 1985.

[79] F. Jammes, P. Lecomte, J. de Almeida-Engler, F. Bitton, ML Martin-Magniette, JP. Renou, P. Abad, and B. Favery. Genome-wide expression profiling of the host response to root-knot nematode infection in Arabidopsis. *The Plant Journal*, 44:447458, August 2005.

[80] Y.M. Jeong, J.H. Mun, I. Lee, J.C. Woo, C.B. Hong, and S.G. Kim. Distinct roles of the first introns on the expression of Arabidopsis profilin gene family members. *Plant Physiol.*, 140:196–209, Jan 2006.

[81] A M Jones, M H Bennett, J W Mansfield, and M Grant. Analysis of the defence phosphoproteome of arabidopsis thaliana using differential mass tagging. *Proteomics*, 6(14):4155–4165, Jul 2006.

[82] S.J. Kim, K.W. Kim, M.H. Cho, V.R. Franceschi, L.B. Davin, and N.G. Lewis. Expression of cinnamyl alcohol dehydrogenases and their putative homologues during Arabidopsis thaliana growth and development: lessons for database annotations? *Phytochemistry*, 68:1957–1974, Jul 2007.

[83] P Mallick, M Schirle, S S Chen, M R Flory, H Lee, D Martin, J Ranish, B Raught, R Schmitt, T Werner, B Kuster, and R Aebersold. Computational prediction of proteotypic peptides for quantitative proteomics. *Nat Biotechnol*, 25(1):125–131, Jan 2007.

[84] A. Marmagne, M.A. Rouet, M. Ferro, N. Rolland, C. Alcon, J. Joyard, J. Garin, H. Barbier-Brygoo, and G. Ephritikhine. Identification of new intrinsic proteins in Arabidopsis plasma membrane proteome. *Mol. Cell Proteomics*, 3:675–691, Jul 2004.

[85] http://www.nematology.umd.edu/rootknot.html.

[86] S. Wiese, K. A. Reidegeld, H. E. Meyer, and B. Warscheid. Protein labeling by itraq: a new tool for quantitative mass spectrometry in proteome research. *Proteomics*, 7(3):340–350, February 2007.

[87] Nuno Bandeira, Dekel Tsur, Ari Frank, and Pavel A Pevzner. Protein identification by spectral networks analysis. *PNAS*, 104(14):6140–6145, April 2007.

[88] S A Beausoleil, J Villén, S A Gerber, J Rush, and S P Gygi. A probability-based approach for high-throughput protein phosphorylation analysis and site localization. *Nat Biotechnol*, 24(10):1285–1292, Oct 2006.

[89] P A DiMaggio, N L Young, R C Baliban, B A Garcia, and C A Floudas. A mixed integer linear optimization framework for the identification and quantification of targeted post-translational modifications of highly modified proteins using multiplexed electron transfer dissociation tandem mass spectrometry. *Mol Cell Proteomics*, 8(11):2527–2543, Nov 2009.

[90] Banu Dost, Nuno Bandeira, Xiangqian Li, Zhouxin Shen, Steve Briggs, and Vineet Bafna. Shared peptides in mass spectrometry based protein quantification. In *RECOMB*, pages 356–371, 2009.

[91] J E Elias and S P Gygi. Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry. *Nat Methods*, 4(3):207–214, Mar 2007.

[92] S Loughrey Chen, M J Huddleston, W Shou, R J Deshaies, R S Annan, and S A Carr. Mass spectrometry-based methods for phosphorylation site mapping of hyperphosphorylated proteins applied to net1, a regulator of exit from mitosis in yeast. *Mol Cell Proteomics*, 1(3):186–196, Mar 2002.

[93] Matthias Mann and Ole N. Jensen. Proteomic analysis of post-translational modifications. *Nature Biotechnology*, (21):255 – 261, 2003.

[94] Matthias Mann, Shao-En Ong, Mads Grnborg, Hanno Steen, Ole N. Jensen, and Akhilesh Pandey. Analysis of protein phosphorylation using mass spectrometry: deciphering the phosphoproteome. *Trends in Biotechnology*, 20(6):261 – 268, 2002.

[95] D Phanstiel, J Brumbaugh, W T Berggren, K Conard, X Feng, M E Levenstein, G C McAlister, J A Thomson, and J J Coon. Mass spectrometry identifies and quantifies 74 unique histone h4 isoforms in differentiating human embryonic stem cells. *Proc Natl Acad Sci U S A*, 105(11):4093–4098, Mar 2008.

[96] M J Sniatynski, J C Rogalski, M D Hoffman, and J Kast. Correlation and convolution analysis of peptide mass spectra. *Anal Chem*, 78(8):2600–2607, Apr 2006.

[97] S Tanner, H Shu, A Frank, L C Wang, E Zandi, M Mumby, P A Pevzner, and V Bafna. Inspect: identification of posttranslationally modified peptides from tandem mass spectra. *Anal Chem*, 77(14):4626–4639, Jul 2005.

[98] D Tsur, S Tanner, E Zandi, V Bafna, and P A Pevzner. Identification of post-translational modifications via blind search of mass-spectra. *Proc IEEE Comput Syst Bioinform Conf*, pages 157–166, 2005.

[99] Christopher T. Walsh, Sylvie Garneau-Tsodikova, and J. Gregory Gatto. Protein posttranslational modifications: The chemistry of proteome diversifications. *Angewandte Chemie International Edition*, (45):7342–7372, 2005.

[100] P A Wilmarth, S Tanner, S Dasari, S R Nagalla, M A Riviere, V Bafna, P A Pevzner, and L L David. Age-related changes in human crystallins determined from comparative analysis of post-translational modifications in young and aged lens: does deamidation contribute to crystallin insolubility? *J Proteome Res*, 5(10):2554–2566, Oct 2006.

[101] S F Altschul, T L Madden, A A Schäffer, J Zhang, Z Zhang, W Miller, and D J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, Sep 1997.

[102] J D Thompson, D G Higgins, and T J Gibson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res*, 22(22):4673–4680, Nov 1994.

[103] P Uetz, L Giot, G Cagney, T A Mansfield, R S Judson, J R Knight, D Lockshon, V Narayan, M Srinivasan, P Pochart, A Qureshi-Emili, Y Li, B Godwin, D Conover, T Kalbfleisch, G Vijayadamodar, M Yang, M Johnston, S Fields, and J M Rothberg. A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature*, 403(6770):623–627, Feb 2000.

[104] T Ito, T Chiba, R Ozawa, M Yoshida, M Hattori, and Y Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Natl Acad Sci U S A*, 98(8):4569–4574, Apr 2001.

[105] V R Iyer, C E Horak, C S Scafe, D Botstein, M Snyder, and P O Brown. Genomic binding sites of the yeast cell-cycle transcription factors sbf and mbf. *Nature*, 409(6819):533–538, Jan 2001.

[106] Y Ho, A Gruhler, A Heilbut, G D Bader, L Moore, S L Adams, A Millar, P Taylor, K Bennett, K Boutilier, L Yang, C Wolting, I Donaldson, S Schandorff, J Shewnarane, M Vo, J Taggart, M Goudreault, B Muskat, C Alfarano, D Dewar, Z Lin, K Michalickova, A R Willems, H Sassi, P A Nielsen, K J Rasmussen, J R Andersen, L E Johansen, L H Hansen, H Jespersen, A Podtelejnikov, E Nielsen, J Crawford, V Poulsen, B D Sorensen, J Matthiesen, R C Hendrickson, F Gleeson, T Pawson, M F Moran, D Durocher, M Mann, C W Hogue, D Figeys, and M Tyers. Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. *Nature*, 415(6868):180–183, Jan 2002.

[107] R J Cho, M J Campbell, E A Winzeler, L Steinmetz, A Conway, L Wodicka, T G Wolfsberg, A E Gabrielian, D Landsman, D J Lockhart, and R W Davis. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell*, 2(1):65–73, Jul 1998.

[108] B Ren, F Robert, J J Wyrick, O Aparicio, E G Jennings, I Simon, J Zeitlinger, J Schreiber, N Hannett, E Kanin, T L Volkert, C J Wilson, S P Bell, and R A

Young. Genome-wide location and function of dna binding proteins. *Science*, 290(5500):2306–2309, Dec 2000.

[109] S Wuchty, Z N Oltvai, and A L Barabási. Evolutionary conservation of motif constituents in the yeast protein interaction network. *Nat Genet*, 35(2):176–179, Oct 2003.

[110] S Bandyopadhyay, R Sharan, and T Ideker. Systematic identification of functional orthologs based on protein network comparison. *Genome Res*, 16(3):428–435, Mar 2006.

[111] S Suthram, T Sittler, and T Ideker. The plasmodium protein network diverges from those of other eukaryotes. *Nature*, 438(7064):108–112, Nov 2005.

[112] T R Hughes, M J Marton, A R Jones, C J Roberts, R Stoughton, C D Armour, H A Bennett, E Coffey, H Dai, Y D He, M J Kidd, A M King, M R Meyer, D Slade, P Y Lum, S B Stepaniants, D D Shoemaker, D Gachotte, K Chakraburtty, J Simon, M Bard, and S H Friend. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–126, Jul 2000.

[113] R Milo, S Shen-Orr, S Itzkovitz, N Kashtan, D Chklovskii, and U Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, Oct 2002.

[114] A C Gavin, M Bösche, R Krause, P Grandi, M Marzioch, A Bauer, J Schultz, J M Rick, A M Michon, C M Cruciat, M Remor, C Höfert, M Schelder, M Brajenovic, H Ruffner, A Merino, K Klein, M Hudak, D Dickson, T Rudi, V Gnau, A Bauch, S Bastuck, B Huhse, C Leutwein, M A Heurtier, R R Copley, A Edelmann, E Querfurth, V Rybin, G Drewes, M Raida, T Bouwmeester, P Bork, B Seraphin, B Kuster, G Neubauer, and G Superti-Furga. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415(6868):141–147, Jan 2002.

[115] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell, Fourth Edition*. Garland, 2002.

[116] Christoph H. Borchers, Roopa Thapar, Evgeniy V. Petrotchenko, Matthew P. Torres, J. Paul Speir, Michael Easterling, Zbigniew Dominski, and William F. Marzluff. Combined top-down and bottom-up proteomics identifies a phosphorylation site in stem-loop-binding proteins that contributes to high-affinity rna binding. *Proceedings of the National Academy of Sciences of the United States of America*, 103(9):3094–3099, 2006.

[117] Banu Dost, Vineet Bafna, and Nuno Bandeira. Identification and quantification of post-translational modification variants and false discovery rates. In preparation, 2010.

[118] Banu Dost, Nuno Bandeira, Xiangqian Li, Zhouxin Shen, Steve Briggs, and Vineet Bafna. Shared peptides in mass spectrometry based protein quantification. In *RECOMB 2'09: Proceedings of the 13th Annual International Conference on Research in Computational Molecular Biology*, pages 356–371, Berlin, Heidelberg, 2009. Springer-Verlag.

[119] Banu Dost, Tomer Shlomi, Nitin Gupta 0002, Eytan Ruppin, Vineet Bafna, and Roded Sharan. Qnet: A tool for querying protein interaction networks. *Journal of Computational Biology*, 15(7):913–925, 2008.

[120] Banu Dost, Chunlei Wu, Andrew Su, and Vineet Bafna. Tclust: A fast algorithm for clustering genome-scale expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. In press.

[121] Buhm Han, Banu Dost, Vineet Bafna, and Shaojie Zhang. Structural alignment of pseudoknotted rna. *Journal of Computational Biology*, 15(5):489–504, 2008.

[122] Ivo L. Hofacker, Walter Fontana, Peter F. Stadler, Sebastian L. Bonhoeffer, Manfred Tacker, and Peter Schuster. Fast folding and comparison of rna secondary structures. *Monatsh. Chem.*, 125:167–188, 1994.

[123] H Kim, R Klein, J Majewski, and J Ott. Estimating rates of alternative splicing in mammals and invertebrates. *Nat Genet*, 36(9):915–916, Sep 2004.

[124] M Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*, 31(13):3406–3415, Jul 2003.