

University of Nebraska - Lincoln

DigitalCommons@University of Nebraska - Lincoln

Theses, Dissertations, and Student Research from
Electrical & Computer Engineering

Electrical & Computer Engineering, Department of


Summer 8-2012

Classification of Genomic Sequences by Latent Semantic Analysis

Samuel F. Way

University of Nebraska-Lincoln, samfway@gmail.com

Follow this and additional works at: <http://digitalcommons.unl.edu/elecengtheses>

 Part of the [Bioinformatics Commons](#), [Computational Biology Commons](#), and the [Other Electrical and Computer Engineering Commons](#)

Way, Samuel F, "Classification of Genomic Sequences by Latent Semantic Analysis" (2012). *Theses, Dissertations, and Student Research from Electrical & Computer Engineering*. 42.

<http://digitalcommons.unl.edu/elecengtheses/42>

This Article is brought to you for free and open access by the Electrical & Computer Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in Theses, Dissertations, and Student Research from Electrical & Computer Engineering by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.

CLASSIFICATION OF GENOMIC SEQUENCES
BY LATENT SEMANTIC ANALYSIS

by

Samuel F. Way

A THESIS

Presented to the Faculty of
The Graduate College at the University of Nebraska
In Partial Fulfilment of Requirements
For the Degree of Master of Science

Major: Electrical Engineering

Under the Supervision of Professor Khalid Sayood
and Professor Ozkan Ufuk Nalbantoglu

Lincoln, Nebraska

August, 2012

CLASSIFICATION OF GENOMIC SEQUENCES
BY LATENT SEMANTIC ANALYSIS

Samuel F. Way, M. S.

University of Nebraska, 2012

Adviser: Khalid Sayood and Ozkan Ufuk Nalbantoglu

Evolutionary distance measures provide a means of identifying and organizing related organisms by comparing their genomic sequences. As such, techniques that quantify the level of similarity between DNA sequences are essential in our efforts to decipher the genetic code in which they are written.

Traditional methods for estimating the evolutionary distance separating two genomic sequences often require that the sequences first be aligned before they are compared. Unfortunately, this preliminary step imposes great computational burden, making this class of techniques impractical for applications involving a large number of sequences. Instead, we desire new methods for differentiating genomic sequences that eliminate the need for sequence alignment.

Here, we present a novel approach for identifying evolutionarily similar DNA sequences using a theory and collection of techniques called latent semantic analysis (LSA). In the field of information retrieval, LSA techniques are used to identify text documents with related underlying concepts, or “latent semantics”. These techniques capture the inherent structure within a collection of documents, and in much the same way, we extend these approaches to infer the biological structure through which a collection of organisms are related. In doing so, we develop a computationally efficient means of identifying evolutionarily similar DNA sequences that is especially well-suited for partitioning large sets of biological data.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my committee and academic advisors for their help throughout my graduate studies. In particular, no words could ever express my appreciation for Dr. Khalid Sayood and the impact that he has had on my life, both academically and personally. I feel very fortunate to have found an advisor who is as patient and caring as he is, and I can't thank him enough for his support and guidance.

I would also like to thank Dr. Michael Hoffman for his invaluable insight and help in preparing this research. He and Dr. Sayood have shown me how absolutely fun research can be, and I thank them both for making my time at UNL incredibly enjoyable. Next, I would like to thank Dr. Ufuk Nalbantoglu for being a great friend and constant source of ideas throughout my graduate career.

Beyond my committee, I want thank Dr. David Russell, who has been an amazing role model, teacher, and friend over the past few years. His high expectations have motivated me to always strive to do quality work, and with him, I also thank my other co-workers at Red Cocoa, especially Mark Nispel, David McCreight, Chris Hruby, and Andy Lenhart. I also need to thank Dr. Mark Bauer for his incredible support and guidance, as well as our collaborators at Names for Life and the University of Nebraska Medical Center. I owe a special thanks and many boxes of cookies to Cheryl Wemhoff and Terri Eastin for helping me get everything lined up to graduate this semester. They helped me to overcome a number of obstacles this summer, and I thank them for all of their patience. I also need to thank Jay Carlson for his help with Adobe Illustrator and for letting me vent my frustrations.

Finally, my thanks also go to my family and friends for their amazing support throughout my academic journey. Since day one, my parents have provided me with

the means and inspiration to accomplish great things, and they've always been there to help me in any way they can. Words can't express how grateful I am for their many sacrifices and undying support. Lastly, I would like to thank my beautiful girlfriend, Laura Norris, whose love and encouragement has carried me through the last nearly eight years of my life. She inspires me to become a better person every day, and this work would not have been possible without her support.

Contents

Contents	v
1 Introduction	1
2 Biology Background	3
2.1 Introduction to DNA Structure	3
2.2 Function of DNA	6
3 Comparing DNA Sequences	8
3.1 Alignment-based Distance Measures	8
3.1.1 Jukes-Cantor	9
3.1.2 Kimura	13
3.2 Alignment-free Distance Measures	15
3.2.1 Relative Complexity Metric	16
3.2.2 K-mer Based Genomic Signatures	19
3.2.2.1 The Average Mutual Information Profile	20
3.2.2.2 The Relative Abundance Index	22
4 Latent Semantic Analysis	25
4.1 Introduction to LSA	27

4.2	Dimensionality Reduction Techniques	31
4.2.1	Singular Value Decomposition	32
4.2.1.1	Mathematical Derivation	33
4.2.1.2	Dimensionality Reduction via SVD	35
4.2.2	Non-negative Matrix Factorization	36
4.2.2.1	Mathematical Derivation	38
4.2.2.2	Dimensionality Reduction via NMF	40
4.3	Adaptation to Sequence Data	41
5	Visualizing High-Dimensional Datasets	43
5.1	Method	45
5.2	Implementation	47
5.3	Results and Discussion	48
5.3.1	Validation of Taxonomy Data	49
5.3.2	Map Creation From Proximity Data	50
5.3.3	Discussion	51
6	Clustering of 16s Ribosomal Genes	54
6.1	Background Information	54
6.2	Method	55
6.3	Results and Discussion	56
7	Identification and Removal of Host DNA Fragments from Metage- nomics Datasets	67
7.1	Construction of a synthetic dataset	68
7.2	Method	70
7.3	Results	73

7.4	Discussion	77
8	Using LSA-NMF to Design Microarray Probes	79
8.1	Background Information	79
8.2	Method	81
8.3	Results and Discussion	82
9	Conclusion and Recommendations for Future Work	89
9.1	Recommendations for Future Work	90
	Bibliography	94

Chapter 1

Introduction

Since the days of the Human Genome Project, the world has witnessed an ongoing revolution in the field of biology. In fact, the changes in this area are so profound that many believe we are currently experiencing the advent of a “New Biology”, one that relies on concepts and approaches from engineering, computational sciences, mathematics, and many other fields[1]. The integration of these disciplines signifies the emergence of biology as an informational science and is necessitated by the introduction of many new technologies for producing biological data.

Some of the most important examples of these new technologies are platforms for massively parallel, next-generation DNA sequencing. Next-generation sequencing (NGS) replaces the traditional Sanger sequencing method with automated, high-throughput technologies that produce unprecedented amounts of sequence data at ever-increasing speeds. What’s more, the cost to purchase and operate these platforms has declined rapidly over the years, and as a result, the number of research centers using these technologies has skyrocketed.

Given the ability to quickly sequence genetic material, scientists are constantly discovering new applications for next-generation platforms. However, with so many

scientists generating such massive amounts of sequence data, it is becoming increasingly difficult to make sense of all this new information. As such, it is imperative that we devise efficient techniques for classifying and organizing genomic sequences such that they may be quickly identified and retrieved.

Traditional methods for sequence comparison are often computationally inefficient and therefore should not be used in large-scale implementations. Instead, in this research, we present a new approach for differentiating genomic sequences based on latent semantic analysis, a theory and collection of techniques born out of the fields of natural language processing and information retrieval. Using this new means comparison, we explore several experiments that will demonstrate the approach as an attractive alternative to traditional distance measures that is especially well-suited for large-scale applications.

Chapter 2

Biology Background

Before we go about introducing methods for comparing DNA sequences, we should first provide some biological background to establish what these sequences are and why they are important. It is commonly understood that DNA is a sort of genetic code or blueprint that is used to construct living things. For the purposes of our discussion, we will require only a slightly more detailed explanation, which we present here. In addition, this section will be used to introduce most of the terminology and conventions that will be used throughout subsequent chapters.

2.1 Introduction to DNA Structure

DNA stands for *deoxyribonucleic acid* and is a double-stranded macromolecule comprised of four basic structural units called *nucleotides*. Nucleotides can be thought of as the building blocks for creating DNA sequences, and much like how we form sentences using strings of letters from our alphabet, we will see that DNA is essentially a string of nucleotides.

A nucleotide molecule consists of three main parts: a five-carbon sugar, a phos-

phate group, and a nitrogenous base or *nucleobase*. As previously mentioned, there are four different nucleotides, each differing only in the nucleobase. The four nucleobases are Adenine (A), Guanine (G), Thymine (T), and Cytosine (C). When referring to a nucleotide, we typically mean to indicate the attached base, which we denote using its one letter abbreviation. Accordingly, this set of four letters forms the basis for our “DNA alphabet”.

In 1952, scientists James Watson and Francis Crick noticed that certain bases naturally pair or bind to one another. Specifically, adenine pairs with thymine (A-T), and guanine with cytosine (G-T). These complementary nucleotides are held together by hydrogen bonds which link the two molecules, forming a *base pair*. Because DNA is essentially a string of these pairs, we typically express the length of a sequence in terms of its number of base pairs or “bp” for short. These chemical properties ensure that each nucleotide bonds only to its pair, which we refer to as its complementary

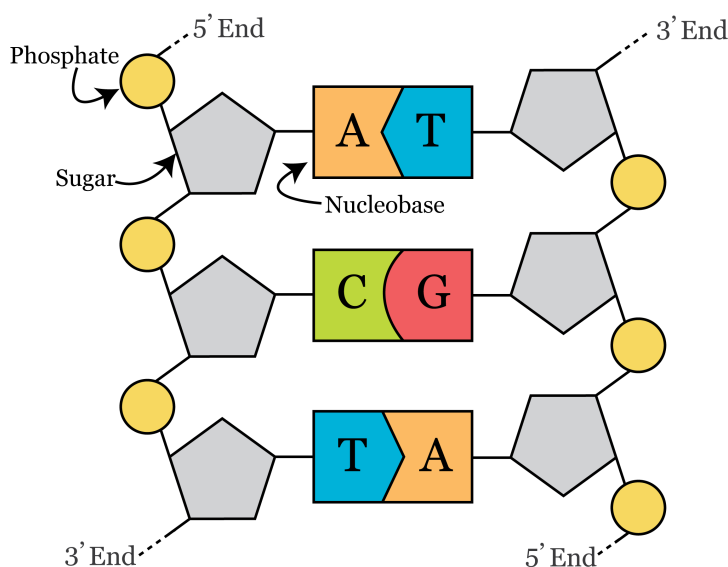


Figure 2.1: A section of DNA. This particular section features three base pairs, for a total of six nucleotides. Labeled features include the individual nucleotide components as well as the 5' and 3' ends of each strand.

base or simply its *complement*.

Figure 2.1 illustrates a small section of DNA, consisting of three base pairs. As shown in the figure, the DNA macromolecule includes two, complementary strands of nucleotides which are held together by hydrogen bonds. Also labeled in this image are the so-called 5' (read as “five prime”) and 3' ends of each strand. As it turns out, the chemical structure of the macromolecule provides directionality in the way that DNA is constructed. Appropriately, when we talk about reading a DNA sequence, we are simply naming off the bases from the 5' end towards the 3' end on one of the strands.

In order differentiate between the two complementary strands of DNA, we typically label one as the *forward strand* and refer to the other as the *reverse strand*. To demonstrate, let's call the left strand shown in Figure 2.1 as the forward strand. Using the 5' end as our starting point, the forward strand is read as *ACT*. Similarly, starting from its 5' end, the reverse strand is read as *AGT*. Conventionally, for a given segment of DNA, we refer to the corresponding portion of the opposite strand as the *reverse complement*. In the example above, we would say that *AGT* is the reverse complement of *ACT*.

Finally, in Figure 2.1, we note that the DNA macromolecule looks a lot like a ladder, with the nucleobases forming rungs which connect the two sugar-phosphate backbones. However, due to the opposite or *antiparallel* orientation of the two strands and the stacking forces between nucleobases, a DNA macromolecule actually twists around itself. This twisting forms the familiar double-helix structure which we often associate with DNA. Figure 2.2 illustrates this structure.

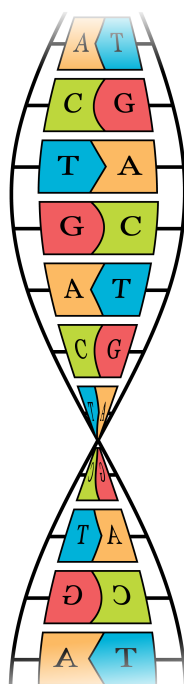


Figure 2.2: Double-helix structure of DNA

2.2 Function of DNA

The primary function of an organism’s *genome* (or complete DNA sequence) is to store the genetic code or set of instructions that determine how the organism is to be built. Portions of the genome, called *genes*, contain specific instructions for how to create proteins and can also be used to send signals within the cell or even control other genes.

Proteins can be thought of as the machinery inside every cell of every living thing. They perform virtually every function from converting food into energy to defending our bodies from harmful bacteria. In fact, the word “protein” itself comes from the Greek word “prota” meaning “of primary importance” [2]. Naturally, because they’re so important, we’re very interested in studying proteins and, in our case, the DNA

used to create them.

Because of the macromolecule's double-helix structure, DNA sequences are in general very well-protected, and the likelihood of a change occurring is very low. When a change or *mutation* does occur, it can have a wide variety of impact on the organism. Typically, mutations have little impact and are either ignored or corrected by error-checking proteins inside the cell. Less frequently, a mutation will significantly affect some function within the organism. This change could be advantageous, or as is the case with certain types of cancer, the mutation could be lethal. If the mutation is advantageous or has little to no impact on the organism, it stands a chance of being permanently incorporated into the organism's genome and being passed on to future generations.

When a change is incorporated into a genome, particularly one that benefits the organism, we say that the organism has *evolved* in some capacity. These changes, however slowly, contribute to the level of genetic diversity between species. From this, assuming that all forms of life have evolved from a common ancestor, it follows that the level of similarity in the genomic sequences for two species provides an indication of their evolutionary distance and, thus, functional similarity.

The focus of this research is to discover computationally efficient techniques for estimating the evolutionary distance between DNA sequences. By clustering or grouping sequences together based on evolutionary similarity, we are able to identify and investigate the similarities and differences between related organisms. In turn, this ability to compare sequences provides us with a means of deciphering the genetic code in which they are written.

Chapter 3

Comparing DNA Sequences

Many biological applications are reliant on our ability to identify and investigate groups of related genomic sequences. In effect, these applications require a means of approximating the evolutionary distance separating two organisms. As one might expect, because this task is so important, a number of approaches have been developed over the years. Here we introduce several popular distance measures, dividing them into two categories: ones that require the sequences to be aligned prior to comparison and others which are “alignment-free” distance measures.

3.1 Alignment-based Distance Measures

Of the many approaches for approximating evolutionary distance, a large number of them require sequence alignment as a preliminary step. When two sequences are said to be aligned, in general, they are thought to be positioned such that we have maximized the amount of overlapping regions in common to the two sequences. Table 3.1 depicts two sequences that have been aligned, with *’s indicating matching bases.

		*	*		*	
T	A	C	G	A	A	T
-	-	C	G	T	A	-

Table 3.1: Two aligned sequences

As can be seen, this particular arrangement of the two sequences produces the maximum amount of overlap between the pair, and at this point we can begin to evaluate their level of similarity. While these particular sequences are somewhat trivial to align, it's not hard to imagine that sequence alignment can be a very difficult problem, especially as the length of the sequences increases. As such, there are a number of different algorithms addressing this issue, each with its own interpretation of and procedure for finding maximal overlap. Popular algorithms include the classic Needleman-Wunsch algorithm [3], Smith-Waterman [4], and the more recent (and more efficient) algorithms GramAlign [5] and MAFFT [6].

At this point, we turn our attention to a few classical techniques for approximating evolutionary distance. The following distance measures will assume that sequences have been aligned using one of these algorithms, however, one should not underestimate the computational burden and complexity of this preliminary operation.

3.1.1 Jukes-Cantor

In the 1960's, before DNA sequencing was really even possible, scientists Charles R. Cantor and Thomas H. Jukes hypothesized about ways of estimating the evolutionary distance between two organisms using their DNA sequences. The method that they developed is known as the Jukes-Cantor model [7], and despite being one of the simplest metrics for approximating evolutionary distance, it is still widely used today. In fact, as new models are developed, they are often compared to Jukes-Cantor as

the de facto standard for measuring evolutionary similarity.

The simplicity behind the Jukes-Cantor model comes from a number of assumptions. First, the model assumes that each site in which the two sequences differ has evolved or mutated independently of all other sites. In addition, it is assumed that all bases are equally likely (i.e. $p_A = p_C = p_G = p_T = \frac{1}{4}$) and that substitution from one base to any other base occurs with the equal probability and mutation rate.

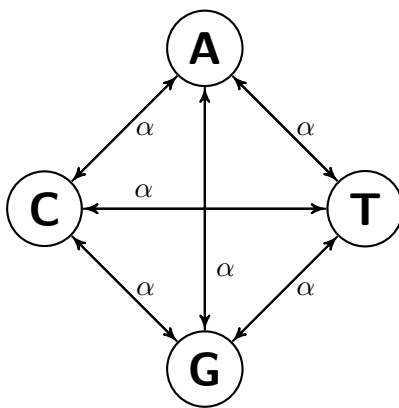


Figure 3.1: Probability of base substitutions in the Jukes-Cantor model

The state transition diagram shown in Figure 3.1 illustrates the probabilities with which a single base changes to any other base, according to the Jukes-Cantor model. As indicated, the probability P_{XY} of a single nucleotide changing from any base X to any base Y where $X \neq Y$ is α .

In order to devise a distance metric using this model, let us define $q(t)$ to be the proportion of nucleotides in the two sequences S_1 and S_2 that are the same at some time instance t . Naturally, it follows that $p(t)$, the fraction of differences between the two sequences is given by

$$p(t) = 1 - q(t). \tag{3.1}$$

At time $t + 1$, the proportion of nucleotides that are the same between S_1 and S_2 includes the nucleotides that were the same at time t and remained the same plus any nucleotides that were different but are now the same. The proportion of nucleotides that stayed unchanged since time instance t is given by the product of $q(t)$ with the probability of no change occurring in the corresponding bases of S_1 and the probability of no change in those nucleotides in S_2 , which is simply $q(t)(1 - 3\alpha)^2$.

On the other hand, for the nucleotides that were different but have since become the same, this change could have occurred in any of three ways. First, the nucleotide in sequence S_1 could have changed to match the corresponding base in S_2 . Similarly, the nucleotide in S_2 could have changed to match the corresponding base in S_1 . Lastly, the nucleotide could have changed in both sequences, and now we have a match. Putting everything together, the proportion of similarities between the two sequence at time $t + 1$ is

$$q(t + 1) = q(t)(1 - 3\alpha)^2 + (1 - q(t))(\alpha(1 - 3\alpha) + \alpha(1 - 3\alpha) + \alpha^2). \quad (3.2)$$

Because α is very small, we can ignore terms involving powers of α , leaving us with

$$\begin{aligned} q(t + 1) &= q(t)(1 - 6\alpha) + (1 - q(t))2\alpha \\ &= q(t) + 2\alpha - 8\alpha q(t) \end{aligned}$$

or

$$q(t + 1) - q(t) = 2\alpha - 8\alpha q(t).$$

As we make the interval smaller, in the limiting case we get

$$\frac{dq}{dt} = 2\alpha - 8\alpha q$$

Solving this differential equation, we get the integrating factor

$$\mu = ke^{8\alpha t}$$

and the solution for $q(t)$ as

$$q(t) = \frac{1}{4} + ce^{-8\alpha t} \tag{3.3}$$

Since we assume that all organisms evolved from a common ancestor, we can say that S_1 and S_2 were at one point identical, and therefore we can solve for c using the initial value $q(0) = 1$. Obtaining $c = \frac{3}{4}$, we are left with

$$q(t) = \frac{1}{4} + \frac{3}{4}e^{-8\alpha t}, \tag{3.4}$$

and

$$\begin{aligned} p(t) &= 1 - q(t) \\ &= \frac{3}{4}(1 - e^{-8\alpha t}). \end{aligned} \tag{3.5}$$

Returning to our transition diagram, we notice that the probability of a change in a nucleotide per time unit is 3α . Since this is the case for both sequences, the

distance d for a time period t will be $6\alpha t$. Rearranging Equation 3.5, we obtain

$$e^{-8\alpha t} = 1 - \frac{4}{3}p$$

or

$$-8\alpha t = \ln\left(1 - \frac{4}{3}p\right).$$

By substituting $d = 6\alpha t$, we finally obtain

$$d = -\frac{3}{4}\ln\left(1 - \frac{4}{3}p\right) \tag{3.6}$$

Looking at Equation 3.6, it is easy to imagine why the Jukes-Cantor metric has remained so popular throughout the years – it’s remarkably simple. After alignment, all that remains is to count the number of mismatches between the sequences, divide by the length of the overlapping portion, and enter this number into the provided equation. What’s more, despite having some rather egregious assumptions in its derivation, the metric yields a surprisingly useful estimate for evolutionary distance.

3.1.2 Kimura

After the introduction of the Jukes-Cantor metric, a number of similar distance measures emerged, attempting to replace some of its underlying assumptions with more biologically accurate predictions. One such measure is the Kimura model which was published in 1980 and is named after its creator, Motoo Kimura [8].

Like Jukes-Cantor, the Kimura model also assumes that all bases are equally likely, however Kimura makes an important distinction in the probabilities with which any one base changes to another. When a single nucleotide undergoes a point mutation,

the substitution is classified as either a *transition* or a *transversion*, which as we are about to see, do not occur with equal probabilities.

A transition occurs when a purine changes to another purine ($A \leftrightarrow G$) or a pyrimidine to another pyrimidine ($C \leftrightarrow T$). A transversion, on the other hand, occurs when a purine is replaced by a pyrimidine or vice versa. Transversions involve a more drastic change in the chemical structure of the DNA molecule and therefore occur much less frequently than transitions. Knowing this, Kimura modified the Jukes-Cantor distance measure to accept two parameters that account for the mutation rate differences between transitions and transversions.

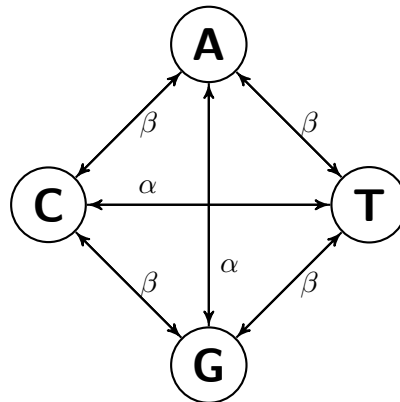


Figure 3.2: Probability of base substitutions in the Kimura model

As shown in Figure 3.2, the Kimura model contains two parameters to account for the differing mutation rates for transitions(α) and transversions(β). From here, the derivation closely follows that of the Jukes-Cantor metric, resulting in the equation below for estimating evolutionary distance.

$$d = -\frac{1}{2}\ln(1 - 2f_{transition} - f_{transversion}) - \frac{1}{4}\ln(1 - 2f_{transversion}) \quad (3.7)$$

Because this model distinguishes between the types of substitutions that can occur,

it follows that the resulting approximation relies on two parameters. First, $f_{transition}$ denotes the fraction of differences due to transition, and similarly, $f_{transversion}$ represents the fraction of differences due to transversion.

Like the Jukes-Cantor model, Kimura is inherently simple to implement and produces very similar yet arguably more biologically relevant results. Somewhat unsurprisingly, since the release of the Kimura metric, several other models have been introduced, attempting to further replace the simplifying assumptions made by Jukes, Cantor, and Kimura. These models include the work of Felsenstein in 1981 [9]; Hasegawa, Kishino, and Yano in 1985 [10]; Tavar in 1986 [11]; and Tamura in 1992 [12] and 1993 [13]. Many of these metrics, like Kimura, distinguish between mutation rates for transitions and transversions but add additional parameters to account for unequal base frequencies and other biases.

3.2 Alignment-free Distance Measures

In each of the distance measures presented thus far, DNA sequences must be aligned before they are compared. As was noted, this preliminary step imposes great computational burden upon each of these metrics, despite the relatively simple steps that follow.

To appreciate this burden, consider the case in which you would like to compare the genomes of two eukaryotes and estimate their evolutionary distance. Potentially billions of bases in length, these two sequences would be horribly difficult to align and would require incredible computational resources. In addition, while alignment may reveal the majority of similarities between the two sequences, this class of distance measures fails to address the issue of genomic rearrangements. A rearrangement occurs when a portion of the genome, for a variety of reasons, is relocated to a different

position along the sequence. As a result, when using alignment-based approaches, large portions of similar genomes might be incorrectly identified as differences due to rearrangement.

In hopes of overcoming some of the limitations of alignment-based distance measures, we now introduce several alignment-free metrics. These approaches, in general, attempt to construct statistical characterizations or profiles for DNA sequences which are then compared to profiles of other sequences in order to approximate their evolutionary distance.

3.2.1 Relative Complexity Metric

Our first example of an alignment-free distance measure is the relative complexity measure, which was originally devised by Hasan Otu and Khalid Sayood in 2003 [14]. In a recent and pending publications by the authors, the relative complexity measure has been demonstrated to produce results similar to what is obtained using the Jukes-Cantor model but with a huge computational performance advantage.

The relative complexity measure is based on an approximation of Kolmogorov complexity [15], achieved using the Lempel-Ziv complexity measure [16]. The Lempel-Ziv (LZ) complexity of a sequence is given by the number of distinct phrases in a left-to-right parsing of its bases. A distinct phrase is one which has not been encountered in the history of the sequence. As an example, consider the sequence $Q = gagacagt$. Initially the history of the sequence is the empty string. As g is not in the empty string, g becomes the first distinct phrase. The history of the sequence now consists of the single letter g . Continuing, because a is also not found in the history, a becomes the second unique phrase. The history of the sequence is now ga . The next two letters are g and a . Because the sequence ga already exists in the history we keep

building the phrase until we get to *gac* which is not in the history of the sequence. Thus, *gac* is the third unique phrase. The history is now the sequence *gagac*. The next unique phrase in the sequence is *agt*, as *ag* is found within in the history. Thus the sequence *gagacagt* can be represented by four unique phrases and therefore has an LZ complexity of four. We can represent the parsing by

$$Q = g \cdot a \cdot gac \cdot agt \quad (3.8)$$

and the LZ complexity by $c(Q)$, or the number of distinct elements in Q . We can see that sequences with more diversity will result in higher LZ complexity, and sequences that are more uniform will result in smaller LZ complexity. Further, if we concatenate a sequence with another, similar sequence, the increase in complexity will be quite small. On the other hand, if the appended sequence is, in general, dissimilar from Q , we will observe a large increase in the LZ complexity. To demonstrate, consider the sequence $R = gagacat$ and the parsing for the concatenated sequence QR :

$$QR = g \cdot a \cdot gac \cdot agt \cdot gagacat \quad (3.9)$$

Despite being nearly twice as long as the original sequence Q , the concatenated sequence QR has an LZ complexity of just five, only one more than the LZ complexity of Q alone. If, on the other hand, we were to concatenate Q with a sequence with notably dissimilar composition, we observe a greater increase in LZ complexity. To demonstrate, consider the sequence $T = tcgctta$ and the parsing for the concatenated sequence QT :

$$QT = g \cdot a \cdot gac \cdot agt \cdot tc \cdot gc \cdot tta \quad (3.10)$$

As expected, the LZ complexity of QT goes up to seven, notably higher than the complexity of QR and an increase which is proportional to the length of the concatenated sequence.

In light of these observations, we can construct a distance metric by comparing the relative complexities of two DNA sequences. To do this, consider the difference between the magnitudes of $c(Q)$ and $c(QR)$, where QR is the sequence formed by appending R to the end of Q . As previously stated, if the sequences are similar, $c(QR)$ will be only slightly larger than $c(Q)$. We can therefore estimate the two sequences' relative complexity by observing the difference in these two values. Finally, in order to produce a symmetric measurement, we must consider the differences between $c(RQ)$ and $c(R)$. Putting all of this together, the relative complexity measure is defined by

$$d(Q, R) = \frac{c(QR) - c(Q) + c(RQ) - c(R)}{\frac{c(QR) + c(RQ)}{2}} \quad (3.11)$$

where the denominator provides normalization to account for differences in sequence length. Recalling that similar sequences will exhibit small relative complexities, smaller values produced by this metric can be used to indicate stronger similarity, at least in terms of the LZ-based complexity estimate.

A highly efficient implementation of the relative complexity measure can be accessed as part of the GramAlign software package [5]. Given the impressive performance of the application, it is clear that the relative complexity measure provides an excellent alternative to alignment-based distance metrics. In addition, since the algorithm uses LZ complexity to approximate evolutionary distance, this approach offers a distinct advantage over alignment-based measures when comparing entire genomes or sequences long enough to be affected by genomic rearrangement.

3.2.2 K-mer Based Genomic Signatures

In the relative complexity measure which we have just discussed, a distance metric was constructed by parsing DNA sequences into lists of distinct subsequences of variable length. Now, we focus our attention on a large collection of alignment-free distance measures that are based on the frequency of occurrence of fixed-length subsequences or “k-mers”.

A k-mer refers to an oligonucleotide (“oligo”) or polymer of length k that typically denotes a portion or subsequence of some larger sequence. As an example, we could say that *gagacagt* from our example above is a k-mer of length eight or, simply, an “8-mer”. Going a step further, we could say that *gagacagt* contains the 3-mers (or “trigrams”) *gag*, *aga*, *gac*, *aca*, *cag*, and *agt*. Since we have four bases in our alphabet, these trigrams are just a small subset of the 4^k or, in this case, $4^3 = 64$ possible k-mers.

Thinking about how we could use k-mers the basis for a distance metric, recall our previous discussion on the relative complexity measure. There we concluded that similar sequences would have many phrases or subsequences in common, and thus, a low relative complexity. Similarly, we can expect sequences with similar composition to have a large number of k-mers in common.

To demonstrate this property and justify using k-mers as a suitable basis for creating distance measures, let us return for a moment to the example from our previous discussion in which we had defined the sequences $Q = \textit{gagacagt}$, $R = \textit{gagacat}$, and $T = \textit{tcgctta}$. Again, QR will be used to denote the sequence created by appending R to the end of Q . Ultimately, we would like to see that Q has more k-mers in common with QR than QT , so we begin by constructing the sets of k-mers contained within each sequence, which we will denote using Z .

$$\begin{aligned}
Z_Q &= \{gag, aga, gac, aca, cag, agt\} \\
Z_{QR} &= \{gag, aga, gac, aca, cag, agt, gtg, tga, cat\} \\
Z_{QT} &= \{gag, aga, gac, aca, cag, agt, gtt, ttc, tcg, cgc, gct, ctt, tta\}
\end{aligned}$$

We can now consider the number of additional k-mers found going from Q to QR and QT . Once again, ideally, more similar sequences will have more k-mers in common, so the number of additional trigrams will be smaller.

$$|Z_{QR} \setminus (Z_R \cap Z_{QR})| = 3$$

$$|Z_{QT} \setminus (Z_T \cap Z_{QT})| = 7$$

As was expected, in comparing the sets of k-mers used to construct each concatenated sequence, QR required fewer additional k-mers than QT . Based on this observation, we can once again correctly assume that Q is more similar to R than T . More importantly, though, we have given reason for using k-mers as a basis for constructing distance measures.

3.2.2.1 The Average Mutual Information Profile

In the simple example above, we saw that two sequences with similar composition are comprised of very similar sets of distinct k-mers. If we were to increase the length of the sequences being compared, however, this information alone might not be enough to differentiate between the two. As a result, in order to build a more effective characterization or profile for a piece of DNA, we will need to incorporate more of

the information that is embedded within the sequence.

A very natural example of such a characterization is the average mutual information (AMI) profile by Bauer et al. [17]. The AMI profile is an information theoretic measure that describes the amount of information that an individual nucleotide carries about the base that will occur k positions downstream. In other words, if we come across a c in our sequence, does this make us any more certain about what to expect three bases away?

As a simple example, consider the following sequence:

$$S = gagatctga$$

Moving along S , we notice that whenever we encounter a g , the next base is always an a . Similarly, whenever we encounter an a , we always see a t three bases downstream. In terms of mutual information, the appearance of a g completely erases any uncertainty that we might have about the next letter being an a . In this case, we see that a g carries a great deal of information about the next base in the sequence.

The AMI profile extends this idea by computing the average amount of information carried by each base about every other base separated by k positions. Repeating this process for a range of distances from 1 to k , we obtain a vector of length k that can be used as a statistical characterization for the sequence. The elements I_k of this vector can be expressed as

$$I_k = \sum_{X \in \mathcal{A}} \sum_{Y \in \mathcal{A}} p_k(X, Y) \log \frac{p_k(X, Y)}{p(X)p(Y)} \quad (3.12)$$

where \mathcal{A} is our DNA alphabet, consisting of the four nucleotides $\{A, C, G, T\}$. Finally, a distance measure can be formed by simply computing the correlation coefficient for

two AMI vectors.

In their studies, Bauer et al. demonstrated the AMI profile as an effective yet computationally inexpensive means of comparing DNA sequences. The profile is pervasive in the sense that a small fragment will produce a very similar AMI profile to what would be generated for the entire sequence from which it was taken. In addition, the size of the vector used to characterize a sequence is typically small (<50 elements), making the profile very compact and inexpensive to store in memory.

All of these properties make the AMI profile a very attractive method for comparing genomic sequences. We should note, however, that in the interest of forming a compact profile, each element in the vector represents the average of sixteen different measures. By taking this average, we are effectively throwing away information. Consider the case where we have a number of measurements with high values and an equal number low values. The average of these measurements will fall somewhere in the middle, but through this process, we have lost the information of *which* entries had high values—information that might be useful in distinguishing the two sequences that we are comparing. For this reason, as we go about developing new distance measures, we should be mindful of the type and amount of information that we are willing to sacrifice.

3.2.2.2 The Relative Abundance Index

The relative abundance index by Nalbantoglu et al. is an example of k-mer based comparison metric that, unlike the AMI profile, forfeits compactness in favor of preserving more information about the sequence [18]. For a chosen k-mer size k , this model begins by measuring the frequencies or abundances of each of the 4^k possible k-mers. Next, each of these frequencies is compared to an expected value, indicating whether each particular k-mer is over- or underrepresented relative to some expected

frequency.

The expected frequency in this measure is based on Markov assumptions. To demonstrate, Nalbantoglu offers the example in which we start with a particular k-mer, x_1, x_2, \dots, x_k whose probability is $p(x_1, x_2, \dots, x_k)$. Using Markov assumptions, we can rewrite this probability as

$$p(x_1, x_2, \dots, x_k) = p(x_k | x_1, x_2, \dots, x_{k-1}) p(x_1, x_2, \dots, x_{k-1}). \quad (3.13)$$

Assuming that each base occurs independently of each other, the conditional probability on the right-hand side of Equation 3.13 can be re-written as

$$p(x_k | x_1, x_2, \dots, x_{k-1}) = p(x_k). \quad (3.14)$$

Under this assumption, we can now compute the log odds ratio to form the relative abundance index of zeroth order, rai_0 :

$$rai_0(x_1, x_2, \dots, x_k) = \log_2 \frac{p(x_1, x_2, \dots, x_k)}{p(x_k) p(x_1, x_2, \dots, x_{k-1})} \quad (3.15)$$

If we were to instead assume that nucleotides follow a n th order Markov model, it follows that the corresponding relative abundance index would be written as

$$rai_n(x_1, x_2, \dots, x_k) = \log_2 \frac{p(x_1, x_2, \dots, x_k) p(x_{k-1}, \dots, x_{k-n})}{p(x_k, x_{k-1}, \dots, x_{k-n}) p(x_1, x_2, \dots, x_{k-1})} \quad (3.16)$$

In the final step, we combine multiple relative abundance indices, computed for a range of orders to form the final RAI profile,

$$rai(x_1, x_2, \dots, x_k) = \sum_{n=0}^{k-2} rai_n(x_1, x_2, \dots, x_k). \quad (3.17)$$

At this point, having compared each k-mer's frequency to an expected value, the scores in the RAI profile will be positive if the corresponding k-mer appears more frequently than expected and negative if it appears less frequently. Once again, as was the case with the AMI profile, we can form a distance metric by simply correlating or otherwise comparing two profile vectors.

As demonstrated by its authors[18], the relative abundance index profile performs quite well when used as a similarity measure. Classifying sequenced reads of 400bp in length, RAiphy, an open-source implementation of the RAI profile, outperformed other popular similarity-search-based binning programs CARMA [19], MEGAN [20], and PhymmBL [21] in nearly all tests. In addition, it should be noted that the RAI profile was demonstrated using DNA fragments 100bp to 1kbp in length. For comparison, the AMI profile was evaluated on fragments from 1kbp up to 10kbp. Because of this ability to differentiate such small fragments, the RAI metric is a very attractive distance measure for partitioning short-read datasets like those generated by high-throughput, next-generation sequencing platforms.

Chapter 4

Latent Semantic Analysis

Having just introduced a variety of techniques for comparing genomic sequences, we must now ask ourselves: *where do we go from here?* Looking at the collection of distance measures presented above, each technique offered a unique advantage over the rest. For instance, the alignment-based measures were ideal for directly comparing specific, highly-related genomic sequences. On the other hand, the alignment-free metrics were much more computationally efficient and provided a means of differentiating even very short fragments of DNA. As we look to devise a new method for sequence comparison, perhaps a better question might be: *what do we need?*

In answering this question, we should be mindful of a relatively new complication facing today's biologists that didn't exist when we first started comparing genetic sequences. This complication, which has been dubbed the "biological data explosion" is, of course, the sheer volume of data that is currently being generated by labs all across the world [22]. Fueling this explosion are a number of high-throughput technologies like next-generation DNA sequencers that are currently capable of producing on the order of 100 *megabases* of data in one hour [23].

To put this number in perspective, consider the Human Genome Project, which

took a team of biologists from twenty research centers in six different countries nearly thirteen years to assemble [24]. Using next-generation sequencing technologies, this same amount of data can be generated in a matter of hours, and as demonstrated in 2009 by Li et al., a human genome can now be assembled in about two days[25]. What's more is that while the speed of sequencing improves, operating costs continue to fall.

In report issued by the National Institutes of Health's National Human Genome Research Institute, we see that the cost of obtaining a megabase of sequence data has plummeted since the days of the Human Genome Project, which was completed in 2003 [26]. The findings of this report are shown in Figure 4.1.

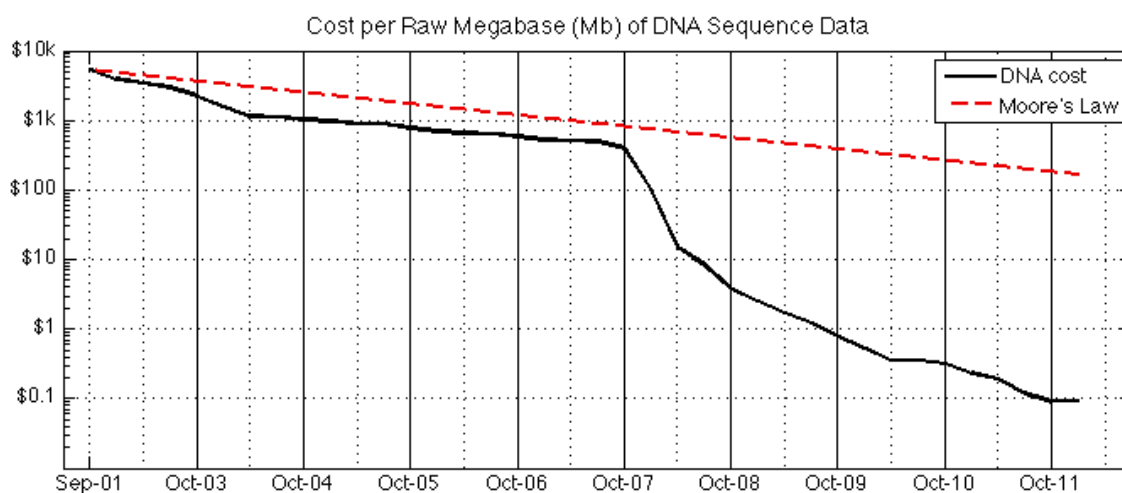


Figure 4.1: The cost of obtaining genomic sequence data over the past decade. Moore's law is shown to provide a growth reference. Source: [26]

As demonstrated in the figure, the cost of obtaining DNA sequence data is decreasing at an astonishing rate, especially since the introduction of next-generation technologies in early 2008. For comparison, the graph also shows Moore's law, which describes the rate of progress in integrated circuit technologies. The cost of generating sequence data is declining at a rate that completely overshadows this already-

impressive rate of growth, and as prices fall, the volume of a data being generated is skyrocketing.

For this reason, biologists today are tasked with managing an overwhelming amount of data, an amount that promises to continue to grow in the years to come. As a result, returning to our question, what we *need* are tools that will help us to conquer the mountains of data that we are creating. We need ways of partitioning and organizing this data so it can be utilized more efficiently.

With this need in mind, in order to partition sets of genomic sequences, a logical first step would be to investigate existing methods for organizing large collections of data. Perhaps the largest dataset that we can imagine is the Internet, which consists of billions of webpages or text documents written on every conceivable topic and in nearly every language. And, yet, in a matter of seconds, search engines are able to retrieve entire lists of relevant documents for a provided word search. Imagine if we could design a similar system to organize our genomic sequences!

In this chapter, we introduce latent semantic analysis (LSA), a theory and collection of techniques traditionally used to identify and partition similar text documents. For our purposes, we hope to extend these techniques to identify and partition evolutionarily similar genomic sequences.

4.1 Introduction to LSA

Latent Semantic Analysis refers to a theory and standard collection of techniques that attempt to identify similar documents in a text corpus based on their underlying concepts or knowledge content [27]. The theory was introduced in 1988 by Dumais, Furnas, Landauer, Deerwester, and Harshman as method for improving the field of Information Retrieval [28]. At that time, most search engines relied on simple lexical

matching to gather documents for a user’s query. However, when we use a search engine, what we really want to discover are documents that discuss a particular topic, regardless of whether or not they contain the exact terms that we’ve entered into our search. As such, these authors devised a method of retrieving documents based on their semantics using classical dimensionality reduction techniques.

LSA, in general, involves these four steps [29]:

1. **Formation of a term-document matrix.**
2. **Transformation/modified weighting of term-document matrix.**
3. **Dimensionality reduction.**
4. **Clustering of documents in the reduced space.**

As shown, the basis of all LSA approaches starts with the formation of a term-document matrix, which uses a vector space model to describe text documents based on the words they contain. To demonstrate, imagine for a moment that we have a language L , consisting of the set of words

$$L = \{“good”, “nice”, “play”, “The”, “was”\}. \quad (4.1)$$

Although not very many, we can construct a few sentences using this language. For example, let’s define the following couple sentences:

$$S_1 = “The play was good”$$

$$S_2 = “The play was nice”$$

In a simple LSA model, we could represent these sentences with binary vectors whose elements indicate the presence or absence of every word in our language. This

type of representation is known as a “bag-of-words model”, in which a sentence or entire text document is characterized by an unordered collection of words. Accordingly, using the language L , we could describe S_1 with the vector

$$\begin{aligned} v_{S_1} &= \langle \text{good?}, \text{nice?}, \text{play?}, \text{The?}, \text{was?} \rangle \\ &= \langle 1, 0, 1, 1, 1 \rangle \end{aligned}$$

and S_2 with

$$v_{S_2} = \langle 0, 1, 1, 1, 1 \rangle .$$

Next, these vectors are compiled into a matrix called a *term-document matrix* in which the rows represent individual terms or words, and the columns represent our collection of documents. This matrix forms the starting point for all LSA techniques, and from here, we can begin comparing document vectors. Before moving any further, however, there are a number of issues that we need to address.

First off, much like was the case in our discussion of k-mer based distance measures, a binary weighting used to denote the presence or absence of individual terms might not be enough information to distinguish between two documents. Moreover, at least in the case of natural languages, we expect certain words to appear in every document. For instance, words like “and”, “a”, and “the” are statistically very likely to occur in every document that we’re comparing. As a result, in order to highlight the differences between documents, many approaches start by removing frequently occurring “stop words”. In addition, most all techniques replace the binary weighting

scheme with a more detailed measure.

A common alternative to binary is the “term frequency” (*tf*) weighting scheme. This weighting simply counts the number of occurrences for every term in the language and normalizes the term-document vector by the total number of words in the document. After normalization, each element in the vector indicates the fraction of the document represented by the corresponding term.

An extension to the *tf* weighting and one of the most popular weighting schemes is the “term frequency–inverse document frequency” or *tf-idf* measure [30]. This weighting attempts to emphasize the presence of words that occur infrequently and deemphasize words that appear throughout the corpus. To compute the inverse document frequency for a term t_i , we define n_i to be the number of documents in the corpus that contain t_i [31]. If the total number of documents is N , the inverse document frequency for t_i is given by

$$idf_i = \log \frac{N}{n_i}. \quad (4.2)$$

By multiplying each element of a term-document vector by the corresponding term’s inverse document frequency, we reduce the strength of terms that appear frequently throughout the corpus, and in doing so, we identify terms that can be used to differentiate our documents. A nice property of this technique is that, because stop words occur so frequently, they are essentially removed by the weighting, eliminating the need to manually detect and remove stop words.

A final and perhaps most important issue before we attempt to compare document vectors is one of word choice. In a study conducted by the founders of LSA, it was discovered that when tasked with assigning something a name, people very rarely agree on what to call things [32]. In fact, the authors noted that two people will

favor the same term with a probability of less than 20%. This issue of multiple words sharing the same meaning is known as *synonymy*, and as one can imagine, it poses an enormous problem for information retrieval systems, particularly those based on lexical matching. To demonstrate, consider our sentences S_1 and S_2 from above. By searching for a “good play”, a system based solely on lexical matching would return S_1 but would leave out S_2 , despite its similar, unenthusiastically positive review.

As a second half to our word choice issue, consider that not only can multiple words share a single meaning, but single words can have multiple meanings. This is known as *polysemy* and can be a major problem even for semantics-based searches. In our sentences S_1 and S_2 , depending on how you spend your free time, you might have assumed that these sentences describe some theatrical production. But what if S_1 is talking about a play in the first half of a football game? Assuming S_2 is talking about something completely different, how do we separate these items?

Clearly, synonymy and polysemy present significant hurdles that must be resolved before we can move any further in our discussion. Fortunately, latent semantic analysis was formulated with these very issues in mind. To overcome them, LSA utilizes various dimensionality reduction techniques to identify sets of correlated words that are used to describe a similar topic. In doing so, we achieve a means of comparing and retrieving documents that is immune to discrepancies in word choice.

4.2 Dimensionality Reduction Techniques

Dimensionality reduction is the heart of any LSA-based technique. By mapping term-document vectors into some lower-dimensional space, we resolve two very important issues. First, dimensionality reduction reduces the size of our problem. Natural languages contain a very large number of words, and for practical reasons, we would

like to avoid vectors of this length. Secondly, as previously mentioned, dimensionality reduction provides us with a method of grouping correlated words into a single dimension. This helps us to address issues that arise out of linguistic ambiguities, or difference in word choice.

Here we present two popular methods for dimensionality reduction: singular value decomposition (SVD) and non-negative matrix factorization (NMF). To indicate which dimensionality reduction routine is used, LSA techniques will often use the notation “LSA-SVD” or “LSA-NMF” to denote LSA via SVD or NMF, respectively. Because SVD was the reduction technique used in the original paper, approaches are typically assumed to be using SVD, unless otherwise noted [28]. Finally, to avoid possible confusion, “latent semantic indexing” (LSI) is a term used to imply LSA-SVD but is often mistakenly treated as a synonym for LSA.

4.2.1 Singular Value Decomposition

In their 1988 paper, the founders of LSA presented singular value decomposition as a powerful method for identifying and combining contextually related words into a reduced-dimensional space [28]. In LSA-based information retrieval systems, recall that from Section 4.1 that we begin with a $t \times d$ term-document matrix X , where t represents the number of terms in our language and rows in X , and d is the number of documents in our corpus. Using dimensionality reduction techniques like singular value decomposition, we can find an approximation for X that groups collections of correlated terms into a small number of dimensions. Using SVD, we transform our data into a new set of fewer variables that capture the majority of the variance in our original variables, highlighting the primary characteristics of the data. Here we provide a mathematical derivation for SVD, followed by a word on its application to

latent semantic analysis.

4.2.1.1 Mathematical Derivation

Using SVD, a $t \times d$ matrix X is factorized as

$$X = U\Sigma V^T. \tag{4.3}$$

Here, U is a $t \times n$ unitary matrix in which the columns represent orthonormal eigenvectors of XX^T , where $n \leq d$ and typically $d \ll t$. Similarly, the rows of V^T are orthonormal eigenvectors of $X^T X$, forming an $n \times d$ matrix, and Σ is an $n \times n$ diagonal matrix with entries sorted in descending order. The diagonal entries of Σ are called the *singular* values or *principal values* of X , and they represent the necessarily real, non-negative square roots of the eigenvalues of $X^T X$ and XX^T [33].

The factorization itself is based on the notion that we can construct n eigenpairs $\{\lambda_i, \mathbf{v}_i\}_{i=1..n}$ for the matrices $X^T X$ and XX^T , where \mathbf{v}_i is an eigenvector with corresponding eigenvalue λ_i . Recall for a moment that the eigenpair $\{\lambda_i, \mathbf{v}_i\}$ is defined such that the following equation holds:

$$A\mathbf{v}_i = \lambda_i\mathbf{v}_i \tag{4.4}$$

In other words, by transforming or multiplying vector \mathbf{v}_i by the matrix A , we obtain a scalar multiple (λ_i) of the vector itself. Replacing A in Equation 4.4 with the matrix $X^T X$, we notice the following property:

$$\begin{aligned}
(X^T X)\mathbf{v}_i &= \lambda_i \mathbf{v}_i \\
X(X^T X)\mathbf{v}_i &= X\lambda_i \mathbf{v}_i \\
(XX^T)X\mathbf{v}_i &= \lambda_i X\mathbf{v}_i
\end{aligned} \tag{4.5}$$

From Equation 4.5, we see that for an eigenvector \mathbf{v}_i of $X^T X$, $X\mathbf{v}_i$ is an eigenvector of XX^T with the same eigenvalue λ_i . As such, we can construct the singular value decomposition of X by finding the eigenvectors of $X^T X$ and XX^T and arranging them into matrices U and V^T such that the i^{th} column of U and i^{th} row of V^T correspond to the same singular value σ_i in Σ . Again, for convenience, we arrange these matrices such that the singular values of Σ are listed in descending order of magnitude.

Most modern algorithms and implementations of SVD compute these eigenpairs based on an approach that was introduced by Golub and Kahan in 1965 [33]. In this method, a matrix X is first reduced to a bidiagonal matrix, which then gets reduced to a diagonal matrix containing the singular values. Typically, for performance reasons, QR factorization is added as a preliminary step before bidiagonalization, resulting in the following steps for SVD [34, 35]:

1. Compute QR factorization of X . ($X = QR$)
2. Reduce R into a bidiagonal matrix using orthogonal transformations. ($R = U_1 B V_1$)
3. Reduce B to a diagonal matrix Σ using an iterative approach.

After computing these steps, our original matrix X will be factorized as $X = U\Sigma V^T$. Essentially, this leaves us with an encoding for our original set of documents

that uses an orthogonal basis set to capture the majority of the variance in X . The diagonal matrix Σ provides us with an indication of which of these basis vectors correspond to dimensions with the most variance. These vectors will correspond to the highest singular values in Σ , and as such, we will see that by ignoring dimensions with relatively low variance, we achieve a means of dimensionality reduction that captures the underlying structure of our data while minimizing the impact of correlated words.

4.2.1.2 Dimensionality Reduction via SVD

After performing singular value decomposition on a term-document matrix X , our set of document vectors are approximated by linear combinations of orthogonal basis vectors or “pseudo-documents”. Some of these basis vectors are considered to be more important than others in the sense that they capture more of the variance in X and are therefore more essential in reconstructing its rows and columns. As a result, if we take only the k most important vectors, corresponding to the k largest singular values in Σ , we are left with a “truncated SVD” that can be used to approximate X .

$$X \approx U_k \Sigma_k V_k^T. \quad (4.6)$$

This truncated SVD forms a least squares approximation for X that uses k dimensions to describe a majority of the variance in the original matrix. Through this process, correlated dimensions in the original term-space are collapsed into a single dimension in the reduced, “LSA space”. As a result, we are left with the rows U_k forming a collection of uncorrelated, basis vectors that can be used to reconstruct term-vectors in the reduced space. Similarly, the columns of V_k^T form a basis set for reconstructing document vectors.

As it turns out, by expressing text documents as linear combinations of these basis

vectors, the impact of linguistic disparities like synonymy and polysemy is greatly reduced, and nearby vectors in the LSA space are related based on their conceptual content [28]. Finally, an information retrieval system can be defined by projecting a query vector into this space and gathering or clustering nearby documents.

4.2.2 Non-negative Matrix Factorization

If we examine the set of matrices produced by the singular value decomposition of a term-document matrix, we notice an interesting property. In this method, when approximating a document vector in an LSA space, the contribution of a basis vector (which indicates the presence of a group of one or more correlated terms) is allowed to take on a negative value. We can trace this property back to the requirements for orthogonality in the basis set, but it's certainly odd to imagine a word or set of words having a *negative contribution* in the reconstruction of a document.

Perhaps a more intuitive method for dimensionality reduction would allow for the reconstruction of items using only non-negative multiples of a set of basis vectors. In this way, vectors either possess a certain quality or not—there is no such thing as the negative presence of a feature. A collection of algorithms implementing this very type of decomposition is known as “non-negative matrix factorization” (NMF).

Non-negative matrix factorization was originally introduced in 1994 by Paatero et al. [36] (as “positive matrix factorization), but the technique was popularized in the late 1990's by Lee and Sebastian when they presented NMF as a method for learning parts of objects from digital images [37, 38]. At that time, principal components analysis was the primary analysis tool used for image processing applications like face detection and facial recognition. In a facial recognition system, for example, an image would be encoded using a set of basis images designed to capture the variance

in a collection of training images [39]. These basis images, called *eigenfaces*, represent the principal components of the training images and appear as distorted, “ghostly” outlines of whole faces.



Figure 4.2: Original facial image to be reconstructed. Source: [38]

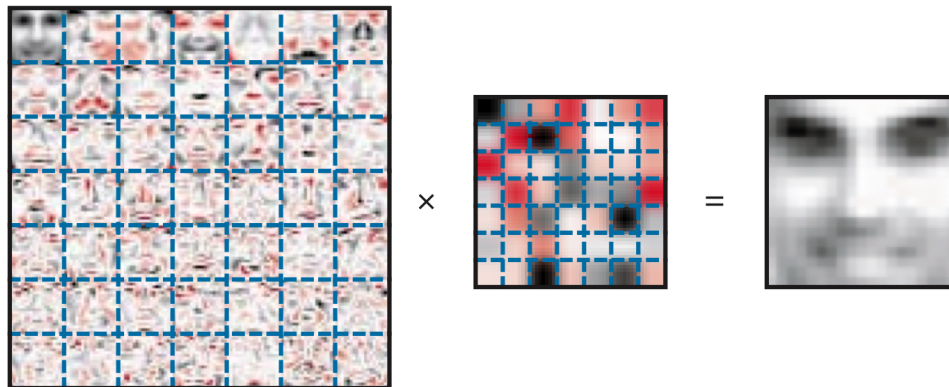


Figure 4.3: Facial image reconstruction as a combination of principal components basis images (or “*eigenfaces*”). Source: [38]

Figure 4.3 shows the reconstruction of Figure 4.2 using a linear combination of the eigenfaces/basis images shown on the left. The middle matrix indicates the weights or contribution amounts for each basis image. In the figure, red is used to indicate negative values, while black represents a positive value, and the intensity of either color signifies its magnitude.

As can be seen in the final image, this approach works rather well, but the set of basis images lacks any sort of intuitive interpretation. This goes against how we

believe humans recognize objects, which indicates that our perception of an object is based on our perception of its parts [40, 41, 42]. Accordingly, the authors of non-negative matrix factorization aspired to devise an algorithm that would learn the essential parts of, say, a collection of facial images and encode the images as a combination of these parts.

4.2.2.1 Mathematical Derivation

NMF works by approximating a non-negative matrix X (which can be a collection of 2D images or term-document vectors) as the product of two non-negative factors A and Y ,

$$X \approx AY \tag{4.7}$$

where A is an $t \times k$ matrix of basis elements, and Y is a $k \times d$ matrix of coefficients/encodings. Here, d indicates the number of vectors being approximated, t is the size of these vectors, and k is the number of elements in the basis set. Each column in X is therefore approximated by a linear combination of k basis vectors, using the weights found in the corresponding column of Y . As indicated, NMF imposes a non-negativity constraint on the matrices A and Y . The result of this constraint is that basis elements are not allowed to have a negative contribution in the approximation of a vector, and as a result, we are left with, essentially, a set of “feature vectors” as our basis set.

The approach of finding reduced rank non-negative factors to approximate the non-negative matrix X , according to Berry and Browne[43], can be stated generically as

The NMF problem:

“Given a non-negative matrix $X \in R^{t \times d}$ and a positive integer $k < \min\{t, d\}$ find non-negative matrices $A \in R^{t \times k}$ and $Y \in R^{k \times d}$ to minimize the functional”:

$$\min_{A, Y} f(A, Y) \equiv \frac{1}{2} \|X - AY\|_F^2, \text{ s.t. } A, Y \geq 0, \quad (4.8)$$

where $\|\cdot\|_F$ is the Frobenius norm and $A, Y \geq 0$ means that every element of A and Y is non-negative. The product AY is therefore a k -dimensional approximation for X , where typically $k \ll \min(t, d)$. Naturally, our choice for k will have a great impact on the quality of the approximation, but as we will see in later chapters, this number will be application- and even data-specific.

Some of the earliest and indeed simplest algorithms for computing the non-negative matrix factorization of a matrix are based alternating least squares (ALS) methods. This class of algorithms begins by constructing a random or otherwise initialized set of basis vectors and continues by applying pairs of “alternating” least squares steps to iteratively refine the starting matrix. These methods are based on the fact the optimization problem presented in Equation 4.8 is convex in *either* A or Y separately but not simultaneously. Instead, given one matrix, these methods optimize the other using a simple least squares computation in alternating fashion [43].

In addition to providing the generalized description of the NMF presented shown in Equation 4.8, Berry and Browne also supplied the following pseudocode to describe a basic algorithm for computing NMF using ALS.

Algorithm 1: Basic NMF algorithm using alternating least squares

```

A = rand(t, k) ;                               /* randomize/initialize A */
for  $i = 1$  to numIterations do
    Solve for  $Y$  using  $A^T A Y = A^T X$ ;          /* (Least squares step 1) */
    Set negative entries of  $Y$  to zero;          /* (Enforce non-negativity) */
    Solve for  $A$  using  $Y Y^T A^T = Y X^T$  ;      /* (Least squares step 2) */
    Set negative entries of  $A$  to zero;          /* (Enforce non-negativity) */
end

```

As can be seen in Algorithm 1, NMF via alternating least squares is a remarkably simple procedure. In fact, because of their simplicity, ALS algorithms lend themselves to fast implementations and have been found to outperform other NMF techniques and even SVD. Finally, since the introduction of the first ALS techniques, a number of algorithms have emerged offering modifications and improvements, typically involving the initialization of the starting matrix A [44, 43].

4.2.2.2 Dimensionality Reduction via NMF

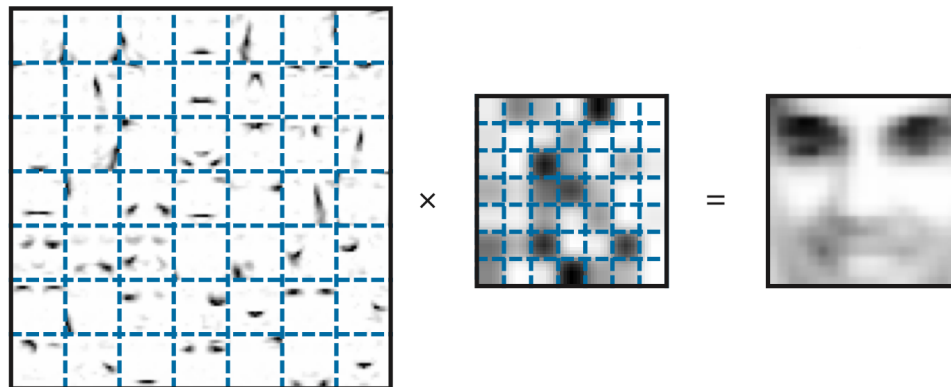


Figure 4.4: Facial image reconstruction as a combination of basis images formed using non-negative matrix factorization on a set of training images. Source: [38]

To compare the results of non-negative matrix factorization versus singular value decomposition, Figure 4.4 demonstrates our facial image from above, reconstructed using a set of basis images that were obtained by performing non-negative matrix factorization on a set of training images. As can be seen, this method also successfully recreates the original test image, but just as importantly, it does so using a set of basis images that have some meaningful interpretation. For instance, the basis set contains individual elements roughly corresponding to eyes, a nose, and a mouth. This is a result of the sparse, non-negative basis set produced by our factorization, and appropriately, the test image is recreated with a combination of each of these elements.

As an end result, we are left with a means of approximating a collection of vectors or images in terms of the parts from which they are constructed. In the facial recognition example, we saw that basis images loosely corresponded to facial features. Returning to our discussion on information retrieval, by using NMF on a term-document matrix, we find that the resulting set of basis vectors defines a k -dimensional latent semantic space in which each axis signifies a particular topic [45]. In other words, we can now represent text documents as a linear combination of a set of base topics, and we can identify related documents as neighboring vectors in the latent semantic space.

4.3 Adaptation to Sequence Data

Up until this point, we have presented latent semantic analysis as a means of identifying semantically similar text documents. Recall, however, that our aim is to modify these techniques to identify evolutionarily similar sequences. To do this, consider our previous discussion on k -mer based distance measures.

With each technique that was presented, a DNA sequence was characterized by some statistical interpretation of the k-mers from which it was comprised. In effect, these techniques model a sequence as an unordered collection of distinct k-mers, a “bag-of-k-mers”, if you will. Since latent semantic analysis techniques are based on the bag-of-words model for representing text documents, we can easily modify them by treating k-mers as the words in our genetic language. As a result, in the methods presented above, we can simply replace the notion of *term-document matrices* with *k-mer-sequence matrices*.

In the interest of discovering possible new information about the construction of genomic sequences, from this point forward, we limit our discussion to using LSA-NMF to identify evolutionarily similar sequences. By using NMF to profile sequences, we anticipate the set of basis vectors to be indicative of some type of “biological signals”. If these signals can be used to differentiate organisms, they themselves may be worth investigating.

As a final concern before discussing potential biological applications of LSA-NMF, in order to evaluate the results of our experiments, we will require a means of visualizing high-dimensional data. In each of our experiments, we will be projecting k-mer-sequence vectors into high-dimensional latent semantic spaces (which we will simply call “feature spaces” because we are using LSA-NMF). To evaluate the closeness of sequences in these feature spaces, we will require a means of visualizing this information in fewer dimensions. In the following chapter, we present nSpect, an exploratory visualization tool for analyzing and inspecting high-dimensional data in three dimensions. We will be making extensive use of this utility to visualize the results of our attempts to differentiate genomic sequences.

Chapter 5

Visualizing High-Dimensional Datasets

Visualization provides valuable insight into the overall structure and defining characteristics of a system by reorganizing and mapping data to a visual reference. A problem arises, however, when the data describing our system involves more than two or three attributes. Such is the case with the high-dimensional feature spaces in which we hope to evaluate k-mer-sequence vectors. In this example, our entire collection of information cannot be represented perfectly in an intuitive, Euclidean space. To address this problem, much research has been done to efficiently reduce the dimensionality of a dataset such that as much information as possible can be represented graphically, using two or three dimensions [46, 47, 48, 49, 50].

For the purposes of our discussion, we will consider only data that can be described using a dissimilarity matrix (proximity data). A dissimilarity matrix (or *distance matrix*) is a square, symmetric matrix containing scores which indicate the similarity of each pair of objects in a collection. Since most applications typically involve more than just two or three items, in order to visualize the dissimilarity matrix, we will

require some form of dimensionality reduction.

The collection of techniques used to embed an $N \times N$ dissimilarity matrix into a lower-dimensional space is called Multidimensional Scaling (MDS). These techniques attempt to map high-dimensional data to a low-dimensional representation while preserving pairwise distances as best as possible [47]. Traditionally, the most popular MDS technique is Principal Component Analysis (PCA), which attempts to reduce the dimensionality of a dataset consisting of a large number of interrelated variables by transforming the data to a new set of uncorrelated variables called *principal components* [51]. Principal components are ordered or ranked such that the first few variables capture most of the variation present in the original set of attributes. Dimensionality reduction is typically accomplished using the assumption that these first few components convey the majority of the information contained in the original data, and, thus, the remaining components can be ignored.

One key problem in using PCA for information visualization stems from the fact that the data is now being described in terms of its principal components. Through this transformation of our data, we lose any sort of intuitive explanation for the dimensions being presented, as we are visualizing derived variables that lack any meaning that was carried by the original set of attributes. In addition, dimensionality reduction using PCA often assumes that since the first few principal components capture most of the variance in the data, the rest are thought to contain little information and are discarded. This assumption, however, is not always the case, as low variance for a component does not necessarily imply that the corresponding component is unimportant or uninformative [52, 53]. As a result, by throwing away principal components with low variance, we risk throwing away the very information that we wish to visualize. A more recent and particularly interesting MDS algorithm is Relational Perspective Mapping (RPM), which arranges objects on a closed surface in

accordance with their pairwise similarity measures [49]. The algorithm treats each item in the dissimilarity matrix as an object in a force-directed, multi-particle system with mutual repulsive forces between each pair of objects. Items with larger relational distances between them exhibit larger repulsive forces, which propel the two objects away from each other on the surface of a torus. Once the objects have reached a stable configuration, the torus is unwrapped to create a two-dimensional relationship mapping.

Because this model assigns repulsive forces between every pair of objects in the dataset, the resulting visualization incorporates information from all of the original N dimensions. Here, we present nSpect, an exploratory visualization tool which uses a similar, repulsive force-driven system to visualize high-dimensional proximity data in three dimensions. This application is an adaptation of a visualization technique presented in 2008 by Bauer et al. [17]. Unlike RPM, which maps objects onto a closed surface, nSpect treats each element in the visualization as a particle in a three-dimensional free space. The resulting visualization allows users to view and interact with the 3D model as it progresses in real-time.

5.1 Method

nSpect requires as input a dissimilarity matrix in the standard, PHYLIP format [54]. This matrix serves as a table of values indicating the distances between every pair of objects in the visualization. An entry, $t_{i,j}$, then denotes the relative, ideal distance separating the i th and j th elements. The value of $t_{i,j}$ should range from 0.0 to 1.0, where 1.0 indicates maximum dissimilarity, and 0.0 suggests equivalency.

Using these distances, nSpect computes repulsive forces between the collection of objects such that the movement produced by these forces will result in a new, more

appropriate configuration at the next time instance.

In order to compute the force between two objects, nSpect first evaluates the error in actual displacement versus a scalar multiple of the ideal distance, $t_{i,j}$, separating the pair.

$$e_{i,j} = (S \times t_{i,j}) - d_{i,j} \quad (5.1)$$

Here, the scalar S has been empirically chosen to produce an appropriate size for the visualization. Using this equation, we see that the error indicates the quality of the current arrangement of objects. As such, we define the repulsive force separating two items with the following equation.

$$\mathbf{f}_{i,j} = e_{i,j} \times \mathbf{x}_{i,j} \quad (5.2)$$

Using the error as a weighting, the force vector $\mathbf{f}_{i,j}$ acts to repel the i th and j th elements along the three-dimensional direction vector $\mathbf{x}_{i,j}$ which separates the two objects. A force equal in magnitude and opposite in direction is applied in the second element. Finally, the net force acting upon the i th element is computed as

$$\mathbf{f}_i = \sum_j \mathbf{f}_{i,j} \quad (5.3)$$

This force vector is calculated for each object in the visualization and is used to determine an appropriate direction and velocity with which to move the particle during the next time instance. The velocity of an object is calculated by

$$\mathbf{v}_i^{(n+1)} = \alpha(\mathbf{v}_i^{(n)} + \beta \times \mathbf{f}_i^{(n)}) \quad (5.4)$$

Here, the superscripts denote the time iteration in the visualization. As is shown,

the velocity vector at the next time instance $\mathbf{v}_i^{(n+1)}$ is determined by the object's current velocity plus some additional amount induced by its force vector.

$$\mathbf{l}_i^{(n+1)} = \mathbf{l}_i^{(n)} + \gamma \mathbf{v}_i^{(n+1)} \quad (5.5)$$

The object's new location $\mathbf{l}_i^{(n+1)}$ is determined using its current location and computed velocity vector $\mathbf{v}_i^{(n+1)}$. In the equations above, the constant α introduces a drag force or decay in velocity. If chosen to be too strong, the drag force could keep objects from escaping local minima, and if too weak, the system is slow to converge. The parameter β is similarly chosen to calculate an appropriate velocity for an object given the amount of force applied to it. If chosen to be too strong, the system will not reach a stable condition. If too weak, the rate of convergence suffers considerably. Finally, γ represents a default step size, which determines how far an object should move. If the step size is chosen to be too large, the system will not be able to reach a stable condition. If too small, the system will be slow to converge. These constants were experimentally determined to provide a good tradeoff between rate of convergence and jitter.

Lastly, as the visualization progresses and objects move into a stable arrangement, one must consider the possibility of objects settling in local minima. To address this issue, objects are randomly displaced by a series of perturbations occurring when the average velocity of the system falls below a threshold. These perturbations diminish in intensity until their effect is negligible.

5.2 Implementation

nSpect is built as a standard, makefile-based project written in C++. It makes use of the freely available OpenGL Utility Toolkit (GLUT) framework for display-rendering

and input event processing. In addition, nSpect has been multi-threaded using the POSIX threading API, “Pthreads”.

When starting the application, the only required input file is a standard, PHYLIP-formatted dissimilarity matrix. The user does have the option, however, to provide additional configuration files to customize the visualization. These files can be used to define the name, color, size, and shape of each object in the collection.

Upon launching the program, the user is presented with a view of the three-dimensional free space in which the visualization takes place. Using the controls described in the on-screen window, the user can explore the space by rotating, panning, and zooming in or out. Clicking on an object “pops” the item, removing it from the visualization and printing its name to the console window. Popped items can be returned to the visualization as described on the controls menu. To pop multiple objects, simply click and drag to select a group of items. Upon releasing the mouse button, the names of all items underneath the selection will be printed to the console window, and the corresponding objects will be removed from the visualization.

Because the objects in the visualization are given randomized starting locations and perturbations, the orientation of the final arrangement is also non-deterministic. Acknowledging this fact, the user has the ability to restart the visualization by “jumbling” or randomizing the starting locations of the objects. In addition, the user can manually issue perturbations or shakes to the collection of items.

5.3 Results and Discussion

Here, we present two example applications for nSpect. The first makes use of the relative complexity measure to compare a set of DNA sequence. To demonstrate the versatility of this technique and provide perhaps a more intuitive application, the

second example shows the reconstruction of a map using only the pair-wise distances between a collection of cities.

5.3.1 Validation of Taxonomy Data

nSpect has successfully been used to identify inconsistencies and misclassifications in a taxonomy database. Occasionally when organisms are given a scientific name, they are incorrectly classified in the sense that there exists another taxonomy group that is evolutionally more similar than the one that was chosen or, in the rare case, the organism constitutes a new group altogether.

To approximate the evolutionary distance from one organism to another, we used the relative complexity measure that was introduced in Section 3.2.1. The program GramAlign provides a fast implementation of this algorithm and, in this example, was used to obtain the distance matrix for a FASTA file containing 16s sequences for organisms representing five genera [5]. The five genera represented in this experiment are Bulkholderia, Chryseobacterium, Desulfovibrio, Nocardioidea, and Shewanella. By preparing the optional display configuration files, we were able to color the items in the visualization according to the genus to which they are assigned in the taxonomy database. If all organisms in the experiment are appropriately labeled, they should cluster together in free space with other organisms from the same genus. What we see in Figures 5.1 & 5.2, however, is that certain objects do not cluster as expected and appear to be misclassified.

Figures 5.1 and 5.2 are screenshots of nSpect, taken after the visualization reached a stable configuration. Figure 5.2 is simply a rotated view of the same scene shown in Figure 5.1. In both figures, several regions of interest have been noted. Labeled regions 1 and 2 show outliers from Chryseobacterium and Desulfovibrio respectively.

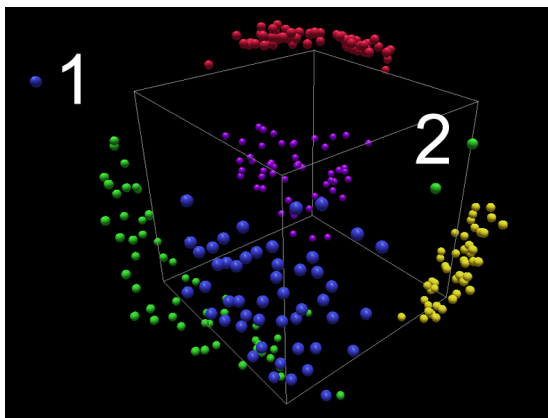


Figure 5.1: Front view of the five-genera taxonomy visualization

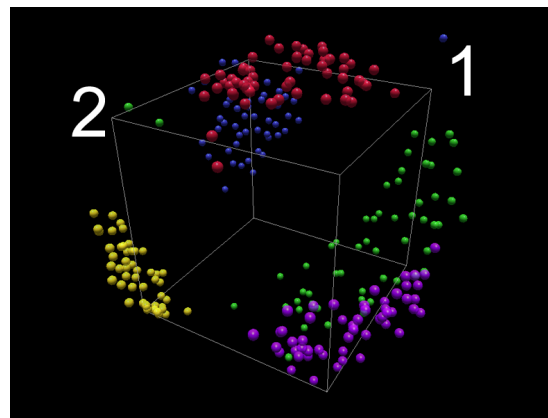


Figure 5.2: Rotated view of the five-genera taxonomy visualization

These objects have been repelled by objects of the same genus, implying these organisms are evolutionally distant from their assigned taxonomic group and, therefore, might have been misclassified or mislabeled.

5.3.2 Map Creation From Proximity Data

As a second example, we attempt to illustrate the versatility of nSpect and its ability to visualize proximity data from any source. One very natural example of proximity data is that of physical proximity. Consider the case where we would like to visualize the distances between cities in the world but lack any sort of geographic coordinate system. In other words, how do we construct a map without using the standard longitude and latitude coordinates for each location?

Despite not having this global coordinate structure, it turns out that, using nSpect, we can recreate a map using only the pairwise distances between cities. For demonstration, we have selected primarily coastal cities from the Western Hemisphere and computed all city-city distances. After formatting this data into a dissimilarity matrix and creating a configuration file to color North and South American cities, our

results are shown in Figures 5.3 & 5.4.

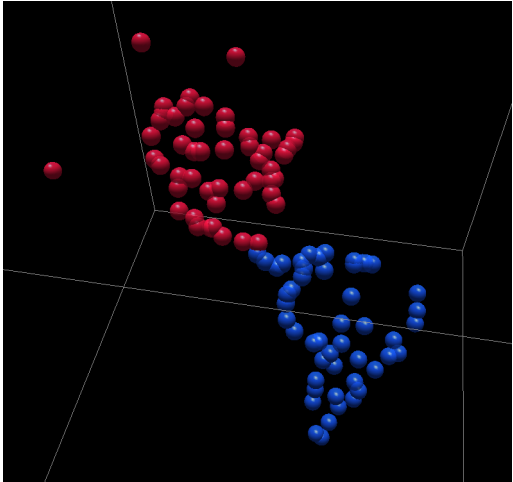


Figure 5.3: Zoomed-out view of the visualization for pair-wise city data

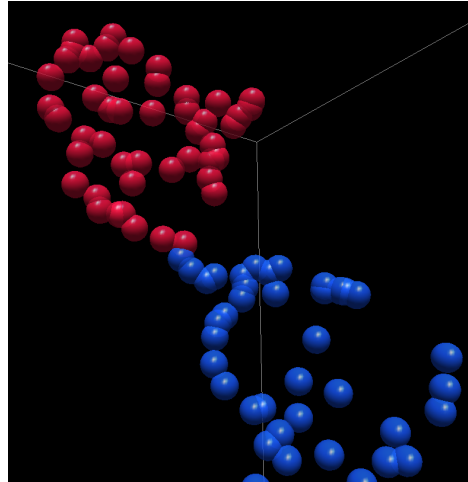


Figure 5.4: Zoomed-in view of the visualization for pair-wise city data

As can be seen in the figures, nSpect accurately positions the city objects, revealing the outline of the continents and the United States in particular. Figure 5.3 shows a zoomed-out view, in which Honolulu, Hawaii and several northern Canadian cities can be seen. This reconstruction of city-to-city data provides a very natural interpretation of how, given a set of pair-wise distances, this approach can be used to analyze the overall structure and underlying relationships in our data.

5.3.3 Discussion

With nSpect, high-dimensional proximity data is visually approximated in a simulated, three-dimensional free space. In order to evaluate the quality of this approximation, we can compute the percent error between the original set of distances and the distances separating objects in the free space. Ideally, if our data were perfectly represented, the distances in the visualization space would be a scaled version of the original data.

To determine this scale factor, let us first define X to be the dissimilarity matrix formed by compiling the distances separating each pair of objects in the visualization space, and Y is the distance matrix driving the visualization. In other words, X is an approximation of Y . Since these matrices are symmetric, we can determine an appropriate scale factor by adjusting the elements of X such that S_X , the sum of the upper triangular portion of X is equal to S_Y , the sum of the upper triangular portion of Y .

$$S_X = \sum_{i=1}^N \sum_{j=i+1}^N X_{i,j}$$

$$S_Y = \sum_{i=1}^N \sum_{j=i+1}^N Y_{i,j}$$

Using S_X and S_Y , we define the scale factor α to be

$$\alpha = \frac{S_Y}{S_X}.$$

Finally, we compute the percent difference between matrices X and Y as

$$\% \text{ error} = \frac{\sum_{i=1}^N \sum_{j=i+1}^N \|\alpha \times Y_{i,j} - X_{i,j}\|}{\sum_{i=1}^N \sum_{j=i+1}^N Y_{i,j}} \quad (5.6)$$

Computing the percent error for the examples presented above, we find that the 5 genera test has an error of 18.5%, while the map example has a percent error of only 0.7%. This difference in error rate should be expected, as in the case of the map example, our data was originally obtained from a three-dimensional model and

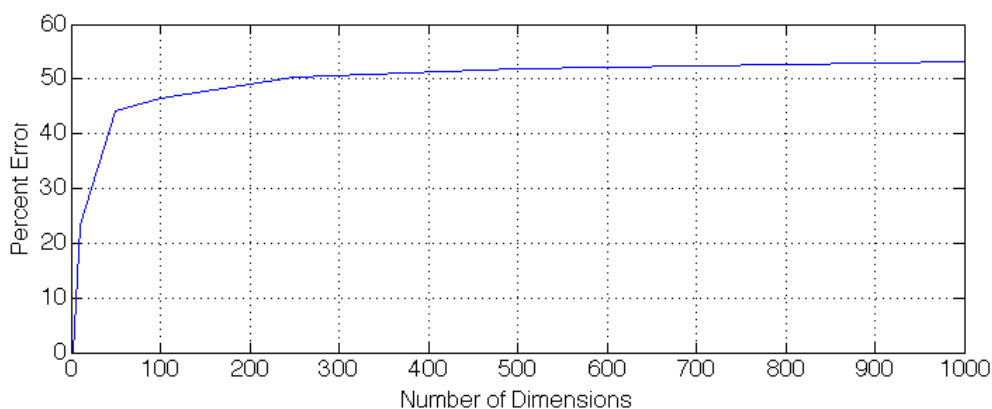


Figure 5.5: Percent Error for Visualizing Randomized Data of N Dimensions

thus lends itself nicely to a three-dimensional representation. In the 5 genera test, however, our data comes from a higher dimensional source and therefore is not as perfectly represented in three dimensions.

As one can imagine, perhaps the worst possible scenario for this visualization scheme would be attempting to approximate randomized data. In this case, the data does not necessarily lend itself to any amount of dimensionality reduction, and thus we expect high error rates for its visualization. In Figure 5.5, we see how the percent error increases with the number of elements in a randomized distance matrix. As expected, the error increases with dimensionality, but interestingly, the error seems to saturate beyond just a few hundred dimensions.

Moving forward, in our experiments, we will be analyzing data that has an expected underlying structure of relatively few dimensions. As such, we can use nSpect to visualize our results while anticipating low error rates. These visualizations will help us to evaluate the performance of our LSA-based techniques for partitioning genomic sequence data and will provide us with a means of comparing these techniques to standard approaches.

Chapter 6

Clustering of 16s Ribosomal Genes

To demonstrate how LSA-NMF could be used to partition a collection of genomic sequences, we begin by attempting to cluster a collection of 16s ribosomal genes. In Section 5.3.1, without providing much explanation as to why this was possible, we used the relative complexity measure and nSpect to estimate the evolutionary similarity of a small sample of these sequences. Here we explain why these sequences were chosen and perform a similar experiment using LSA to compare them.

6.1 Background Information

When we talk about a “16s sequence” (or, more formally, a “16s rDNA sequence”), we are actually referring to a specific gene that is used in the production of *ribosomes*. Ribosomes can be thought of as the machinery inside a cell which is responsible for assembling proteins. Naturally, these pieces of machinery are incredibly important, and every type of organism has them. In fact, because ribosomes are so essential, the genes used to produce them are typically very similar across species, and any differences in these genes can be used as a means of differentiating between the species [55].

The 16s gene itself is fairly small at around 1,500 bases in length. Although short, from our earlier discussion, we know that this length is sufficient for constructing functional statistical characterizations. Another attractive property of this particular sequence is that there are methods of sequencing it directly and therefore quickly and relatively inexpensively[56]. With these things combined, we have a short, informative sequence that is easy to obtain and inexpensive to store. For these reasons, 16s sequences are very popular for constructing phylogenies and taxonomic databases. As a matter of fact, ribosomal sequence databases like Ribosomal Database Project (RDP) are amongst the largest collections available for studying related organisms [57].

In this example, for comparison with the relative complexity measure, we will apply LSA-NMF to the same set of 16s sequences that were evaluated in Section 5.3.1. Recall that this collection consisted of 268 sequences taken from 5 genera: Bulkholderia, Chryseobacterium, Desulfovibrio, Nocardioidea, and Shewanella.

6.2 Method

We begin with the formation of a k-mer–sequence matrix, X . In a series of tests, we found that a k-mer size of 7 provides a good trade-off between specificity and profile size for this dataset. Lowering the k-mer size reduces the amount of information being used to compare sequences, effectively lowering our resolution. Going higher than 7, however, yields a case of diminishing returns in which the additional resolution is not sufficient to justify the increased memory requirements and computational burden. As such, our k-mer–sequence vectors are $4^7 = 16,384$ elements long, and X is a 16,384x268 matrix. After obtaining k-mer frequency counts for each of the 268 sequences, tf-idf weighting was applied to X in order to deemphasize any similarities and highlight any differences between the k-mer counts.

At this point, we are ready for the most important step in any LSA algorithm: dimensionality reduction. In this and subsequent experiments, we will be making use of the NMF MATLAB Toolbox by Li and Ngom [58]. This toolbox provides a collection of standard non-negative matrix factorization routines and was demonstrated as a means of analyzing gene expression data from microarray experiments. As we are just beginning to evaluate NMF, we will restrict ourselves to perhaps its most basic implementation, which is based on an alternating non-negative least squares algorithm. A number of other, more advanced techniques for non-negative factorization exist [59, 60, 61], but evaluating each of them individually will not be necessary for the purposes of this research.

6.3 Results and Discussion

In this example, we happen to already know the number of distinct groups in our dataset—there are five genera. Knowing this information, we will start by approximating X using 5 basis vectors. Using the standard, least squares-based algorithm, we obtain the factorization

$$X \approx AY, \tag{6.1}$$

where A is, again, our set of basis vectors compiled into a matrix of size 16,384x5. Similarly, Y is the 5x268 matrix of encodings from which we can reconstruct the original k-mer–sequence vectors using the basis set A .

If we focus on Y for a moment, effectively, these encodings are short profiles that represent our original set of vectors and, thus, can be used as a basis for comparing them. As a result, we can form a distance measure by simply computing the Euclidean distance or correlation coefficient for each pair of vectors. With this, we have finally

arrived at an LSA-NMF-based distance metric.

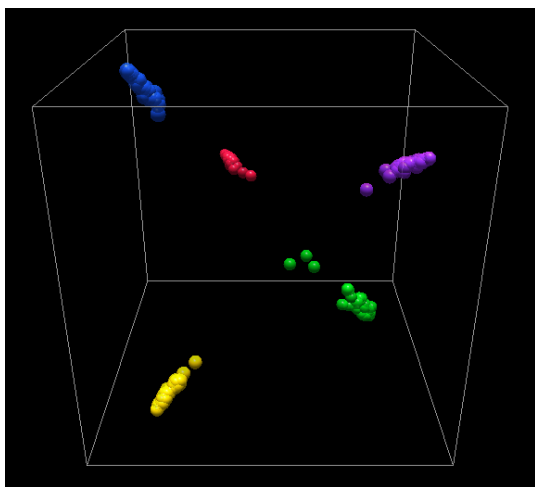


Figure 6.1: Five-genera clustered using Euclidean distance of LSA encodings.

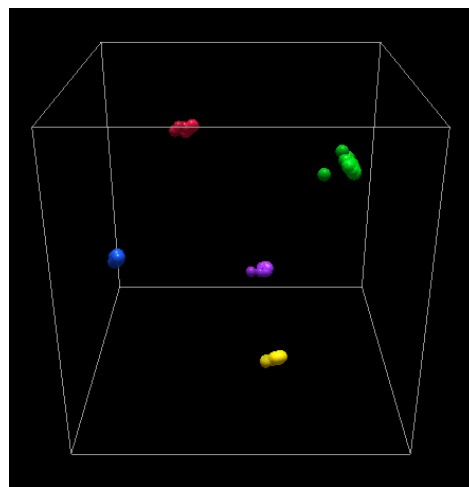


Figure 6.2: Five-genera clustered using correlation coefficient of LSA encodings.

To evaluate our new metric, we have constructed two distance matrices by computing the Euclidean distance and correlation coefficient of each pair of sequences. Using nSpect, we obtain Figures 6.1 and 6.2, which approximate how these profiles cluster in high-dimensional space. Again, the colors of the objects in each figure indicate the genus to which each sequence belongs (red=*Burkholderia*, blue=*Chryseobacterium*, green=*Desulfovibrio*, yellow=*Nocardioides*, and violet=*Shewanella*). As can be seen, these measures provide a great deal of separation in the 5-dimensional feature space. We can compare these images to Figure 6.3, in which sequences are clustered using their relative complexity measures.

Looking at Figure 6.3, we notice that our clusters are much more well-defined using the LSA-based techniques. This makes sense, as the dimensionality reduction stage in our approach attempts to capture the majority of the variance in our data. As a result, vectors which are heavily correlated, like ones representing organisms from the same genus, appear to be very similar when projected into the feature space.

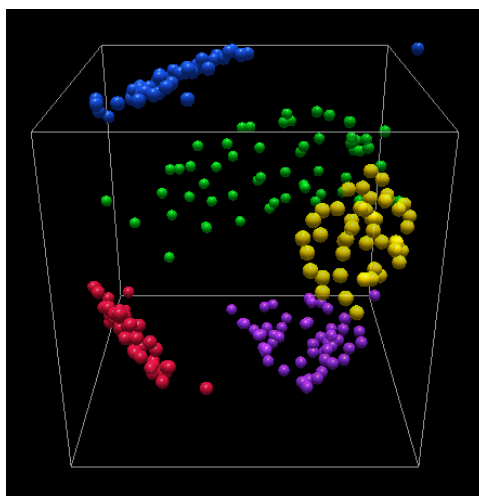


Figure 6.3: Five-genera example using relative complexity measure.

Recall that in the field of information retrieval, dimensionality reduction allowed us to recognize related documents in spite of large differences in word choice due to the authors' vocabulary and writing style. Here, this same effect provides a level of “noise-reduction” that makes our technique less sensitive to small differences between genomic sequences.

From our results, it appears that NMF has effectively identified a set of basis vectors that correspond to the centroid of each of the five genera. Remember that these basis vectors represent sets of k-mers whose presence denotes a feature in the projected space. Here, the features seem to indicate the genus of the organism.

Figure 6.4 demonstrates a sample of elements from the encoding matrix, Y . Each column in the figure presents a grayscale rendering of a column in Y . White indicates a maximum level of contribution for the feature vector. As can be seen, most sequences from a particular genus associate very strongly with just one feature vector. On the other hand, a few columns from the genus *Nocardioides* are not well-defined by any single feature. These columns, to our delight, correspond to the same outliers

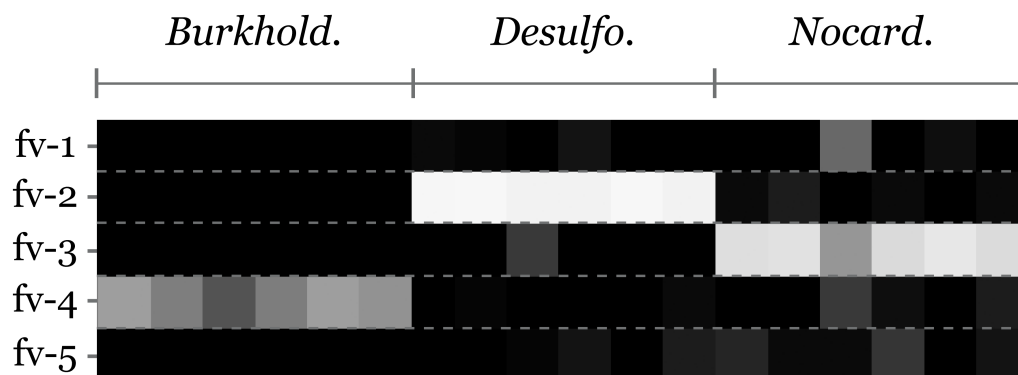


Figure 6.4: Level of contribution for each feature vector

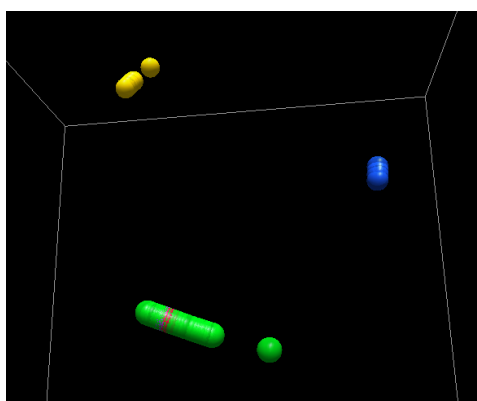


Figure 6.5: Five-genera example using LSA-NMF with 3 feature vectors.

discovered in Section 5.3.1!

Up to this point, we have made use of the fact that we know how many genera are present in our dataset. So what happens if this information is not available? How does the number of feature vectors affect our results? If we lower the number of feature vectors to $k = 3$, we see the clustering shown in Figure 6.5.

Using this new set of three basis vectors, we see that our genera have been divided into three relatively distinct groups. The largest of these three groups contains the three genera *Burkholderia*, *Desulfovibrio*, and *Shewanella*, while the other two genera

seem to correspond to their own feature vectors.

Upon inspection, the genera *Burkholderia*, *Desulfovibrio*, and *Shewanella* are all Gram-negative genera in the phylum of Proteobacteria. On the other hand, *Chryseobacterium* and *Nocardioide*s, the remaining genera, belong to the phyla of Bacteroidetes and Actinobacteria, respectively. As such, it appears that our new set of feature vectors loosely correspond to three taxa at the phylum level.

Suppose, instead, that we were to increase the number of feature vectors in our basis set. From what we have seen, we would expect that the clusters that corresponded to individual genera would now be split into smaller subgroups. As an example, we generate another set of feature vectors, this time with with $k = 7$ elements in the basis set.

As shown in Figure 6.6, by increasing the number of feature vectors, we further divide our collection of sequences. For instance, *Burkholderia* samples (shown in red) that were deemed similar in less-detailed feature spaces have now been separated by this new set of basis vectors. As can be seen in Figure 6.7, there appear to be three

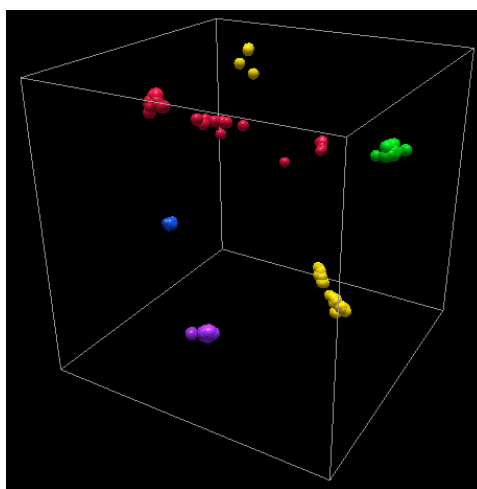


Figure 6.6: Five-genera example using LSA-NMF with 7 feature vectors.

distinct clusters within this genus, which have been labeled for convenience.

Inspecting the three clusters, we find that these groupings are consistent with the results of a 2005 study by Payne et al. [62]. In this work, a phylogenetic tree was constructed for the entire *Burkholderia* genus using the Jukes-Cantor model to measure variations in the *recA* gene. The study used the *recA* gene as they believed it provided finer resolution than the 16s gene for this particular genus. A portion of the phylogenetic tree from this study is shown in Figure 6.8.

From these results, it appears that our LSA-based method works rather well in differentiating between organisms at varying levels of evolutionary similarity, especially when we have a rough idea of how many groups into which we would like to partition our data. If we were to extend this approach, we could use our comparison method to construct a phylogenetic tree like the one shown in Figure 6.8.

As a reference, Figure 6.9 shows a phylogenetic tree constructed for the *Burkholderia* genus, using the Jukes-Cantor distance measure. This tree is noticeably different than the one shown in Figure 6.8, but this was to be expected based on the findings of Payne[62]. Figure 6.10 demonstrates a phylogenetic tree constructed for this same set of *Burkholderia* species but by correlating the encoding vectors used to represent sequences in an LSA-NMF feature space.

Comparing these trees, we see that, in general, our LSA-based method performs reasonably well and has captured the overall structure of the genus. Figure 6.10 does, however, expose a potential problem that should be noted. This tree was constructed using 5 feature vectors, and, as can be seen in the lower portion of the tree, this number probably should have been higher to increase resolution and provide more separation. As a result of this issue, species like *Burkholderia cepacia* and *Burkholderia vietnamiensis* appear misleadingly similar despite the fact that we know they can be separated.

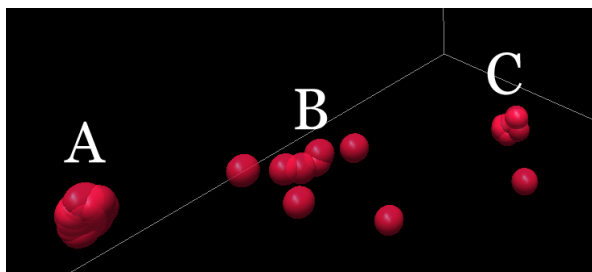


Figure 6.7: Three clusters of Burkholderia.

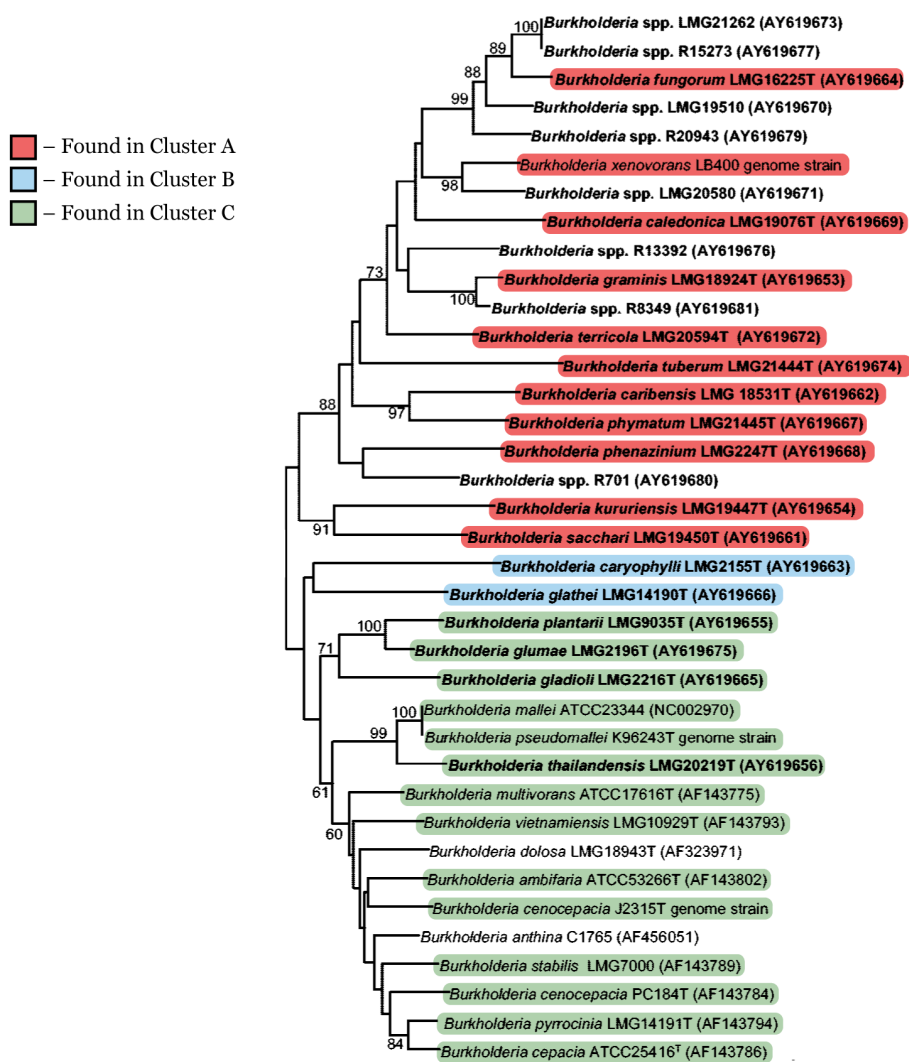


Figure 6.8: Phylogenetic tree for the entire Burkholderia genus. Source: [62]

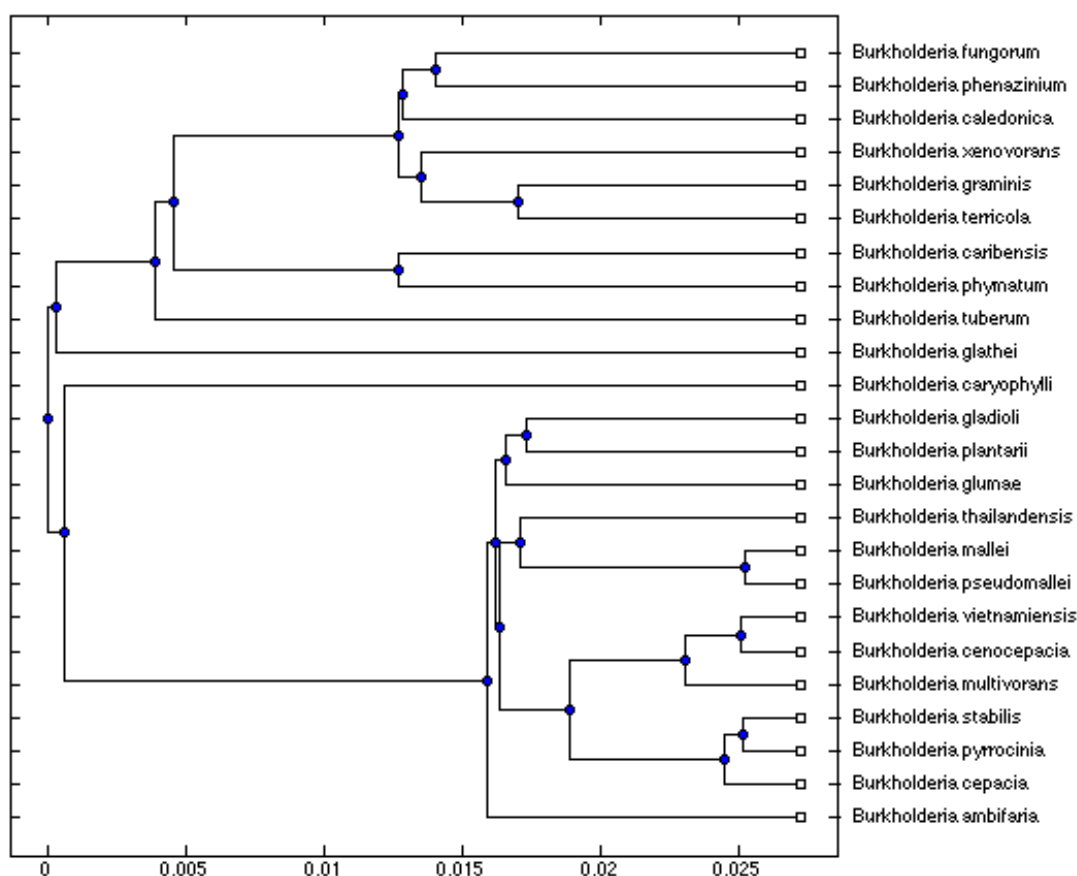


Figure 6.9: Phylogenetic tree constructed with Jukes-Cantor

Beyond constructing phylogenetic trees, the original partitioning of *Burkholderia* shown in Figure 6.7 leads us to a final observation for this example and an idea that will lead into our next experiment. Consider that the genus of *Burkholderia* contains a large number of pathogenic species, many of which are antibiotic resistant and are considered to be especially dangerous. In fact, some of the more hazardous species are feared as potential biological warfare agents and must be handled with extreme caution. Examining the lists of species within clusters A & C from the figure, we notice an interesting trend.

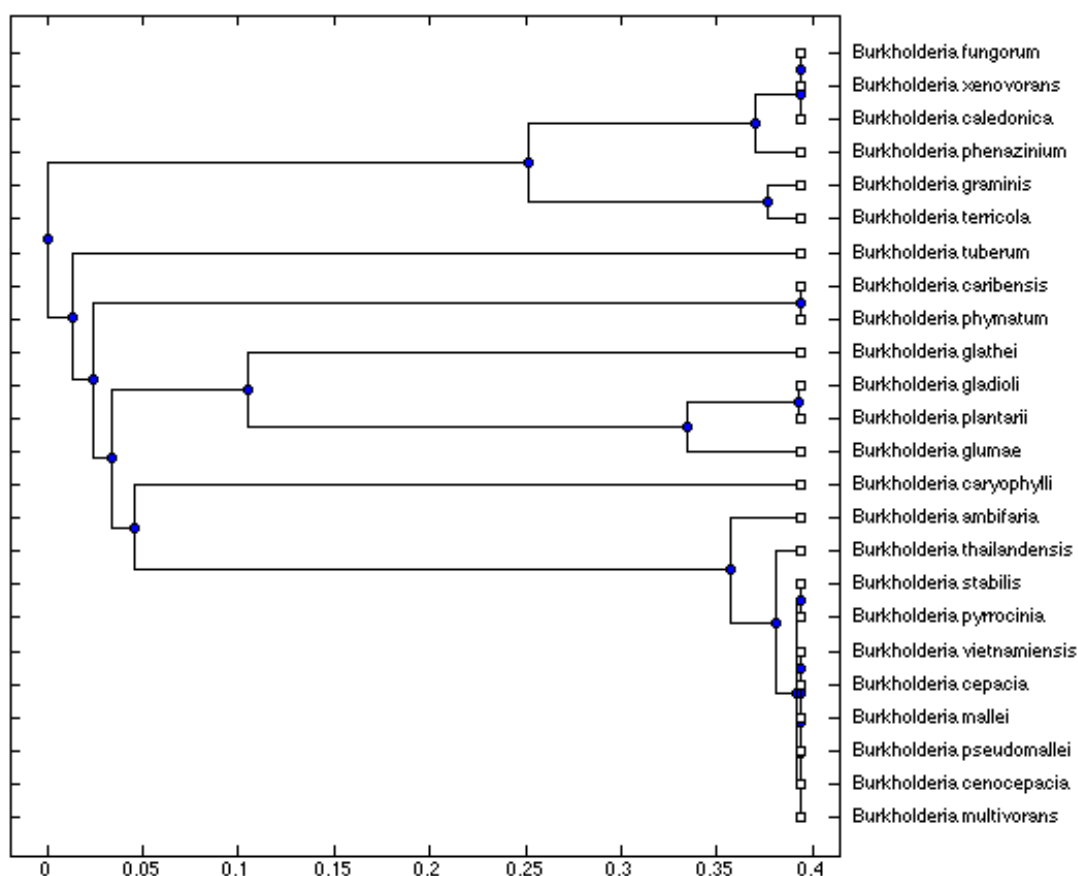


Figure 6.10: Phylogenetic tree constructed with LSA-NMF method

Tables 6.1 and 6.2 demonstrate the risk factor associated with each of the species found in clusters A & C. These risk factors are taken from the Technical Rules for Biological Agents (TRBA) from the German Federal Institute for Occupational Safety and Health (BAuA), and the levels indicate the risk of infection for an extensive list of species[63]. On the BAuA's scale, a risk factor of 1 denotes an agent with a relatively low risk of infection, while agents with risk factors of 2 or higher are particularly virulent.

From these tables, it is clear that cluster C is, in general, more dangerous than

cluster A. As a potential application for our LSA-based techniques, imagine a system that could be used to approximate an unknown sample's risk of infection. Naturally, we could try to compare the unknown sample to all known species, find the best match, and assign our estimate accordingly. A more efficient approach, however, could use our LSA-based technique to identify a set of basis vectors that have been trained to recognize a set of features that are indicative of harmful agents. Such a system would require the ability to *query* or project new items into a predetermined feature space in order to avoid recalculating the NMF for every test.

In the next chapter, we present an experiment which uses LSA-NMF to define high-dimensional feature spaces that allow for the differentiation and elimination of unwanted DNA samples from an environmental sample. This experiment introduces a simple technique for querying feature spaces that will be useful in a number of different applications.

Genus	Species	Risk Score
Burkholderia	bryophila	1
Burkholderia	caledonica	1
Burkholderia	caribensis	1
Burkholderia	ferrariae	1
Burkholderia	fungorum	1
Burkholderia	ginsengisoli	1
Burkholderia	graminis	1
Burkholderia	heleia	?
Burkholderia	hospita	1
Burkholderia	kururiensis	1
Burkholderia	megapolitana	1
Burkholderia	mimosarum	1
Burkholderia	nodosa	1
Burkholderia	phenazinium	1
Burkholderia	phenoliruptrix	1
Burkholderia	phymatum	1
Burkholderia	phytofirmans	1
Burkholderia	sabiae	?
Burkholderia	sacchari	1
Burkholderia	sartisoli	1
Burkholderia	sediminicola	1
Burkholderia	silvatlantica	1
Burkholderia	terrae	1
Burkholderia	terricola	1
Burkholderia	tropica	1
Burkholderia	tuberum	1
Burkholderia	unamae	1
Burkholderia	xenovorans	1

Table 6.1: Burkholderia Cluster A with risk scores.

Genus	Species	Risk Score
Burkholderia	ambifaria	2
Burkholderia	arboris	2
Burkholderia	cenocepacia	2
Burkholderia	cepacia	2
Burkholderia	cocovenenans	?
Burkholderia	diffusa	2
Burkholderia	gladioli	?
Burkholderia	glumae	1
Burkholderia	lata	?
Burkholderia	latens	2
Burkholderia	mallei	3
Burkholderia	metallica	2
Burkholderia	multivorans	2
Burkholderia	oklahomensis	2
Burkholderia	plantarii	1
Burkholderia	pseudomallei	3
Burkholderia	pyrocinia	1
Burkholderia	seminalis	2
Burkholderia	stabilis	2
Burkholderia	thailandensis	1
Burkholderia	ubonensis	1
Burkholderia	vandii	?
Burkholderia	vietnamiensis	2

Table 6.2: Burkholderia Cluster C with risk scores.

Chapter 7

Identification and Removal of Host DNA Fragments from Metagenomics Datasets

In the previous chapter, we demonstrated how LSA-NMF could be used to construct high-dimensional feature spaces in which the dimensions indicate the presence of specific biological “feature” present in a genomic sequence. Here we extend this idea and present an application which uses predefined feature spaces to filter out unwanted samples of DNA.

A perfect example of where such a “DNA filter” could be enormously beneficial is in the field of metagenomics. Metagenomics is a relatively new area of study which focuses on sequencing genetic material that has been taken directly from environment samples. This approach differs widely from traditional methods of sequencing an organism, which begin with the isolation and amplification of an organism’s cells. The problem with the traditional approaches is that we now know a very large number of organisms cannot be cultured in isolation and therefore cannot be sequenced by

conventional techniques. Metagenomic studies, on the other hand, do not attempt to remove an organism from its natural environment, but instead, sequence it directly along with anything else that happens to be in the sample. As a result, metagenomic studies often involve identifying and even assembling genomic sequences from a microbial colony containing a diverse mixture of organisms.

A common problem in metagenomic studies is that many of the microbial colonies that we wish to study are found inside other living organisms. As a result, when a sample is taken from, for example, the gut of an animal or a human, the sample will also contain DNA from the host organism. More often than not, the host will have a large and very complex genome that when sequenced will almost inevitably contain portions that look very similar to the bacterial samples that we wish to study. To make matters worse, typically a large percentage of the sequenced data will belong to the host, and we are left with a biological version of finding a needle in the haystack, except in our case, the needle looks a lot like the hay.

Faced with these difficulties, it would be extremely beneficial to be able to remove at least a portion of the host DNA from a sequenced metagenomic sample. In our approach to this problem, we will attempt to filter out unwanted DNA fragments by constructing feature spaces that have been trained to distinguish the host DNA from the bacterial samples that we wish to keep.

7.1 Construction of a synthetic dataset

Without access to an actual metagenomic dataset, our only option is to generate one synthetically. To do this, we have attempted to simulate the results of sequencing a gut sample from *Mus musculus*, the common house mouse. The species in our synthetic mixture are based on the findings of 2012 study by Chung et al. which

notes the types of bacteria found in a gut sample as well as the amounts in which they appear [64]. These bacteria are divided into three primary phyla: Bacteroidetes, Firmicutes, and Proteobacteria. For the purposes of our experiment, the two most dominant genera from each phylum were included in the mixture. These genera are *Alistipes* and *Bacteroides* from Bacteroidetes; *Bacillus* and *Clostridium* from Firmicutes; and *Acinetobacter* and *Enterobacter* from Proteobacteria.

The complete genomes for each of these bacteria as well as all chromosomes from *Mus musculus* were collected and randomly sampled to create 1kbp fragments, simulating shotgun sequencing. The fragments from each organism were then combined to form a collection of nearly 10,000 fragments, made up of approximately 90% mouse DNA and 10% bacterial DNA. Finally, the number of samples taken from each group of bacteria was chosen based on their average relative abundance amounts presented by Chung et al. and for the mouse, an equal number of samples were taken from each chromosome.

Group	Sample Count	% of Total
Host (<i>Mus musculus</i>)	8,771	89.74%
Bacteroidetes	693	7.09%
Firmicutes	260	2.66%
Proteobacteria	50	0.51%

Table 7.1: Amount of samples from each major group of organisms.

Phylum	Genus	Sample Count	% of Bacteria Total
Bacteroidetes	<i>Alistipes</i>	343	34.20%
Bacteroidetes	<i>Bacteroides</i>	350	34.90%
Firmicutes	<i>Bacillus</i>	65	6.48%
Firmicutes	<i>Clostridium</i>	195	19.44%
Proteobacteria	<i>Acinetobacter</i>	25	2.49%
Proteobacteria	<i>Enterobacter</i>	25	2.49%

Table 7.2: Amount of samples from bacterial genus.

7.2 Method

With the synthetic dataset assembled, we are now ready to start designing a filter. To begin, we will base this approach on a few important assumptions. First, we assume that, whether in whole or in part, we have access to the host organism’s genome. In addition, we also assume that we know of and have access to genomic sequences for at least one of the bacteria in the microbiome. As we are about to see, these sequences will be used to construct the feature spaces in which we will determine which fragments to keep and which to discard.

In the experiment demonstrated in Chapter 6, sequences were clustered in feature spaces that were obtained by performing non-negative matrix factorization on the entire collection of sequence profiles. In a real metagenomic dataset, however, our collection might contain millions of sequences, and as such, we would like to avoid this computationally inefficient step. Instead, we now introduce the concept of using a set of training data to define a feature space and use a projection matrix to map k-mer–sequence vectors into this space.

For this experiment, we would like to define a feature space that can be used to differentiate between host fragments and samples that belong to the microbial colony. Accordingly, we will perform NMF on a small set of known samples taken from the host’s DNA as well as the bacteria that we presume is in the mixture. In this example, we trained an initial feature space using a collection of just under 1,000 sequences, half of which were taken from *Mus musculus*. The other half were obtained by randomly sampling *Bacteroides fragilis*, which is an organism from the most populous genus in the mouse’s gut microbiome.

These sequences were then profiled as k-mer frequency vectors using a k-mer size of 7, and the results were compiled into a k-mer–sequence matrix, X_K , representing

the profiles of our known set. Next, we apply tf-idf weighting followed by the same standard, non-negative least squares implementation of NMF on X_K to obtain the approximation

$$X_K \approx A_K Y_K. \quad (7.1)$$

In anticipation of a diverse set of samples in our mixture, we chose to use $k = 25$ basis vectors for the factorization. Moving forward, recall that A_K is the set of basis vectors which define a high-dimensional feature space, and Y_K is a collection of encodings that can be used to reconstruct our original set of vectors using this basis set. At this point, we introduce a method of projecting new samples into a feature space that has been obtained through NMF. For the purposes of our filter, this will allow us to project new samples into the feature space and determine whether or not they appear to be from the mouse genome.

Given an unknown sample x , its approximation in the feature space defined by the set of basis vectors A can be written as

$$x \approx Ay, \quad (7.2)$$

where y is the encoding used to reconstruct x using the elements of A . In our eyes, y is a k -dimensional profile (a “feature profile”) that characterizes x , so we desire a means of transforming x into y . Solving for y in Equation 7.2, we obtain the projection matrix B as follows:

$$\begin{aligned}
x &\approx Ay \\
A^T x &\approx (A^T A)y \\
(A^T A)^{-1} A^T x &\approx (A^T A)^{-1} (A^T A)y \\
(A^T A)^{-1} A^T x &\approx y \\
(A^T A)^{-1} A^T x \triangleq Bx &\approx y
\end{aligned} \tag{7.3}$$

Now, given an unknown sample x , we can perform a simple matrix multiplication to project x 's k-mer-sequence vector into a predetermined feature space. For example, we can project an item into the feature space defined by A_K using the projection matrix $B = (A_K^T A_K)^{-1} A_K^T$. Once samples are in the feature space, we can identify similar objects, which in our case means determining whether or not a sequence appears to have been taken from the host's genome.

Having known sequences from both the host and the microbial colony, there are a number of ways in which we could partition the feature space in order to identify samples that should be removed. One approach would be to project and cluster known fragments along with our set of unknown samples. By noting how the known sequences cluster, we can determine which groups of sequences should be filtered out. For simplicity's sake, we will use k-means clustering to group elements in the feature space. Clearly, there are more accurate and efficient means of doing this, (a few of which we will discuss later), but this technique will suffice to demonstrate our point.

Finally, putting everything together, our approach for filtering a host's DNA from a sequenced metagenomic sample can be described by the following steps:

1. **Construct a projection matrix, B , using known samples.**

2. **Project known and unknown samples into the feature space with B .**
3. **Cluster the feature profiles.**
4. **Eliminate groups containing known-host samples.**
5. **Repeat steps 1-4 as needed, using different known samples.**

7.3 Results

As stated in the previous section, we began this experiment by constructing a k -mer-sequence matrix using a set of roughly 1,000 known fragments randomly chosen from the host's genome and the microbial colony. This matrix was then used to train an initial feature space with corresponding projection matrix B_0 . Next, a collection of 200 known-host fragments, 200 known-bacteria fragments, and the entire set of approximately 10,000 unknown samples were projected into the feature space defined by B_0 . It should be noted that the 200 known-host and known-bacteria fragments were different than those used to train B_0 .

Once in the feature space, all projected samples were clustered into 20 groups using k -means to identify sequences with similar feature profiles. The composition of these clusters are presented in Table 7.3. Note that this table presents several bits of information. First, there are separate columns denoting the clustering of known and unknown samples. If we were to implement this algorithm, the "Known" columns represent the information that we would use to determine whether or not remove the cluster. Supposing we require a large majority of known-host samples and perhaps a maximum number for the removal of known-bacteria samples, we could make the removal predictions shown in the right-most column. For our experiment, we will assume that we only have this information available and require a four-times majority and maximum of ten known-bacteria samples in the clusters we remove.

Cluster	Known Samples		Unknown Samples		Remove?
	Host	Bacteria	Host	Bacteria	
1	45	0	2060	0	Y
2	2	0	15	0	Y
3	12	0	537	0	Y
4	0	99	5	280	N
5	14	1	553	0	Y
6	3	0	111	0	Y
7	7	0	257	0	Y
8	10	0	192	0	Y
9	2	0	78	0	Y
10	0	68	2	243	N
11	1	0	54	0	Y
12	6	0	280	0	Y
13	31	17	1182	145	N
14	8	0	273	0	Y
15	3	0	209	0	Y
16	5	0	152	0	Y
17	51	9	2720	0	Y
18	0	6	22	168	N
19	0	0	6	167	N
20	0	0	64	0	N

Table 7.3: Round 1 Clustering of Mouse Metagenomic Data.

From Table 7.3, we can see the number of unknown samples that we removed in this round of filtering. Notice that we didn't lose any unknown bacterial samples but were able to remove 7,491 of the unknown mouse fragments. In just one pass of the filter, we successfully removed over 85 percent of the host DNA, leaving 1,280 samples. Looking at cluster 20, for example, we might have been able to remove even more if our set of known samples were larger. Also worth noting, looking at cluster 13, if we increased the number of known-bacteria samples that we were willing to throw away, we could have eliminated almost another 13.5% of the host DNA, which would have left just 98 samples or 1.1% of the host DNA while retaining 85% of the bacterial samples.

At this stage, we have essentially three options. We can decide that this is good enough and that we'd like to stop; we could repeat this same set of steps using a

Cluster	Known Samples		Unknown Samples		Remove?
	Host	Bacteria	Host	Bacteria	
1	78	1	0	0	Y
2	1	0	0	0	Y
3	7	0	29	0	Y
4	0	0	0	135	N
5	10	0	6	0	Y
6	16	4	75	0	Y
7	0	34	0	226	N
8	0	84	9	34	N
9	1	54	2	203	N
10	0	0	63	0	N
11	1	0	0	0	N
12	3	4	32	94	N
13	9	1	294	2	Y
14	0	1	0	141	N
15	10	7	273	43	N
16	0	2	0	117	N
17	33	8	470	7	Y
18	4	0	24	0	Y
19	10	0	0	0	Y
20	17	0	4	0	Y

Table 7.4: Round 2 Clustering of Mouse Metagenomics Data.

different projection matrix; or we could place everything that remains along with the known samples back into the feature space and repeat the clustering and removal process. Going with latter option, we repeat the clustering with the same set of known samples and obtain the round 2 results presented in Table 7.4.

In this round, we see that we lost 9 bacterial fragments but eliminated another 902 host fragments. After round 2, we are down to 378 samples or 4% of the original amount of host DNA, and we still have over 99% of the original 1,003 bacterial samples. Continuing in this fashion for a few more rounds we obtain the results shown in Table 7.5. This same information is presented graphically in Figure 7.1. In the table, a * next to the round number indicates that a projection matrix, trained on a new group of host and bacterial samples was computed prior to this stage. In addition, with each of these rounds, a new set of known samples was chosen to be

clustered in this and subsequent rounds.

Round	Remaining Host Samples	Remaining Host %	Remaining Bacterial Samples	Remaining Bacterial %
(Start)	8771	100.0	1003	100
1*	1280	14.59	1003	100
2	378	4.30	994	99.1
3	235	2.68	985	98.2
4*	68	0.775	974	97.1
5	65	0.741	973	97.0

Table 7.5: All Rounds of Filtering Mouse Metagenomics Data. (* indicates that a new projection matrix and set of known sequences was introduced in this round.)

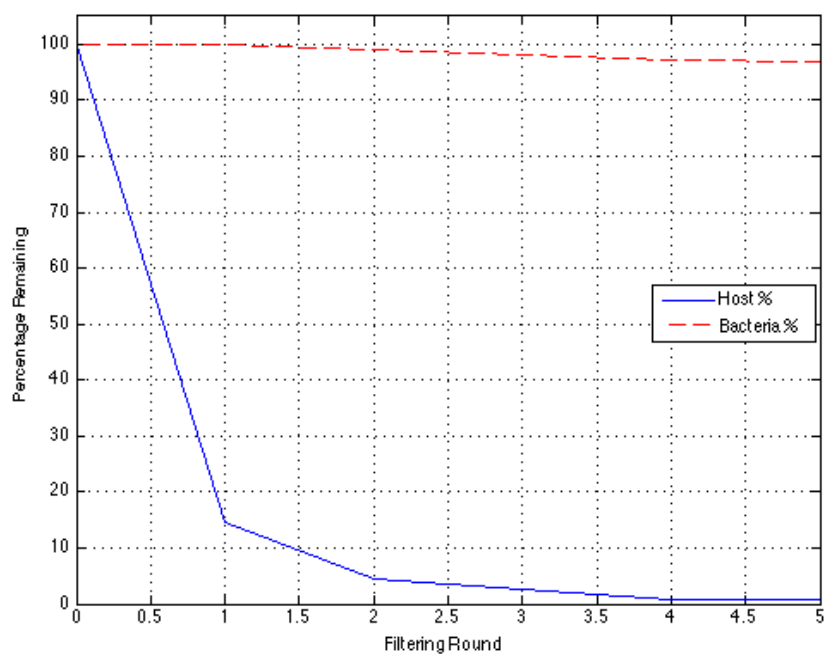


Figure 7.1: Remaining percentage of unclassified samples after each round of filtering.

Stopping after either round 4 or 5, the final percentage of host samples remaining is less than one percent of its starting amount, while preserving about ninety-seven percent of the original unclassified bacterial samples. In addition, throughout this

process, the composition of our mixture went from 90% host DNA down to about 6% in the final set.

7.4 Discussion

In this experiment, we demonstrated a method for filtering host DNA from a sequenced metagenomics dataset. In doing so, we were able to greatly reduce the amount of host DNA in the mixture. Had we been willing to sacrifice a larger number of bacterial samples, we could have completely eliminated all host fragments.

Earlier, we mentioned that there are a number of ways to partition a feature space in order to recognize and eliminate fragments which appear to be very similar to the host's DNA. In our example, we used k-means to accomplish this partitioning but noted that other methods might perform better or provide a more appropriate solution for large-scale implementation.

A possible solution to this problem might make use of an important property that stems from the non-negativity constraint that we have imposed on our dimensionality reduction procedure. Under LSA-NMF, a sample either exhibits a particular feature or not. Up until this point, however, we've made little use of the *strength* with which a sample associates with a given feature vector. Indeed, a crude method of clustering can be achieved by simply noting the feature vector with the greatest contribution for a collection of samples [58].

Unfortunately, in our tests, clustering items by their strongest feature was perhaps a bit too crude for most applications. There is hope, however, for a slight modification of this simple idea. To demonstrate, if we observe the contributions of each feature vector in the reconstruction of a sequence, we note that a few dimensions have very large components, and the rest die off rather quickly.

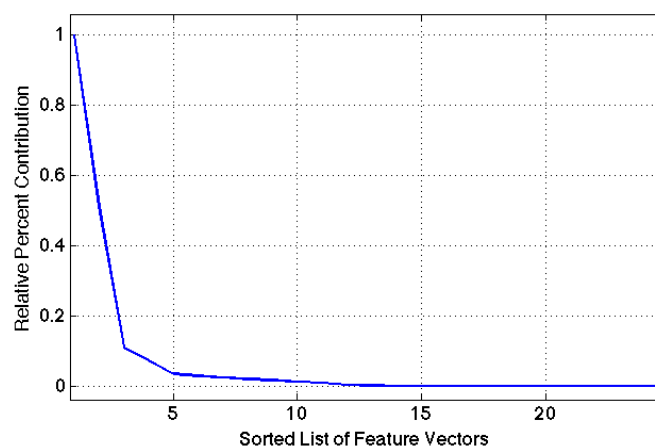


Figure 7.2: Distribution of feature vector strength sorted by magnitude.

Figure 7.2 demonstrates the average level of contribution for a collection of 500 feature profiles of length 25 that have been sorted by their magnitude. As can be seen, most sequences are almost completely characterized by five or fewer components. An extremely efficient means of clustering samples might be possible by devising a clever method of expressing sequences as a combination of features. In the same way that a semantic topic may be combination of multiple subtopics in the realm of information retrieval, we would express sequences as a combination of a set of biological features. As a result, feature profiles would serve as a sort of hyper-spectral coloring that could be used to distinguish between highly similar sequences with very low computational overhead.

Chapter 8

Using LSA-NMF to Design Microarray Probes

In this third and final experiment, we investigate yet another potential biological application for latent semantic analysis via non-negative matrix factorization. In the two previous experiments, LSA-NMF was used to construct high-dimensional feature spaces in which genomic sequences were classified in some manner. Up until this point, however, we have paid little attention to the feature vectors themselves, which, as we are about to discover, have a very natural application in the design of DNA microarray probes.

8.1 Background Information

A DNA microarray is a technology that uses a large collection of short DNA probes to detect the presence of specific sequences in a mixture. The probes are actually short pieces of complementary DNA that bind or *hybridize* to a matching sequence if it is found in the mixture. As one can imagine, this technology is incredibly useful

and has been incorporated into a wide variety of applications.

One of the most popular uses for DNA microarrays is for gene expression profiling. When a gene is transcribed or copied by the enzymes inside a cell, a piece of messenger RNA (mRNA) is created. This piece of mRNA is essentially a set of instructions that directs the machinery inside the cell to perform some function. By convention, when a gene is transcribed into mRNA, we say that the gene has been *expressed*. If a particular gene is used to perform some routine task, it might be expressed very frequently or in large amounts. On the other hand, if the gene gets used only under special conditions, it will have a relatively low rate of expression.

With a DNA microarray, we can design a set of probes to measure the expression of an entire collection of genes simultaneously. By noting when and how much genes are expressed, we observe an organism's "gene expression profile". As it turns out, this profile differs between individuals and even in different parts of the body. By comparing a set of expression profiles, we obtain information that can be used for a variety of purposes like predicting a gene's function or evaluating the status of a disease.

In addition to gene expression profiling, DNA microarrays are also frequently used to detect pathogenic species in an environmental sample. By designing a set of probes that indicate the presence of certain organisms, we can use microarrays to diagnose infectious diseases or even monitor the safety of our food, water, and air [65]. Thinking back to our previous experiments, the role that microarray probes play in each of these applications sounds remarkably similar our use of feature vectors to identify and differentiate between organisms. In this experiment, we attempt to devise a novel method for designing microarray probes by reverse engineering the feature vectors in an LSA-NMF space.

8.2 Method

In previous examples, we have used the feature vectors or dimensions of an LSA-NMF space as evidence of some presumed biological feature that is exhibited by a genomic sequence. In our first experiment, for instance, these features were clear indicators of the genus or some other taxonomic group to which the organism belonged. The presence of this feature, however, is signaled by the presence of a set of one or more k-mers in a k-mer-sequence vector. As a result, by investigating which k-mers are found in these sets, we will have recovered a collection of distinct k-mers that together can be used to indicate a biological feature.

Unfortunately, in the process of constructing a feature space using NMF, there is no simple method for observing which k-mers get mapped into each feature vector. We can, on the other hand, project individual k-mers into a predetermined space and note how strongly they associate with a particular feature. In an information retrieval system, this method would be the equivalent of querying a semantic space with individual words and noting how strongly they indicate a semantic group. In effect, this procedure identifies potential *keywords* or, in our case, distinct k-mers whose presence is indicative of a semantic group or biological feature.

Finally, by varying our choice of k-mer size and the number of dimensions in our factorizations, we may be able to devise a simple method for establishing collections of variable-length oligonucleotides that can be used to design efficient sets of microarray probes. As demonstrated by our first experiment, using a feature space of lower dimensions results in a less-specific partitioning of our collection of elements. In detecting the presence of a distinct organism, perhaps we can employ a series of feature spaces to search for a species at varying levels of precision.

8.3 Results and Discussion

To evaluate our proposed method of designing microarray probes, we will once again make use of the 5-genera dataset presented in our first experiment. An actual metagenomic dataset would almost certainly contain a much more diverse mixture of species, but this collection will suffice to demonstrate our concept.

Recall that by training a feature space of five dimensions on this collection of sequences, the elements in the resulting basis set roughly correspond to the five genera found in the mixture. This is demonstrated in Figure 8.1, which shows the amount of contribution for each the five feature vectors in reconstructing each sequence in the collection. A value of 1.0 in this figure means that a sample is best approximated using only one of the elements from the basis set.

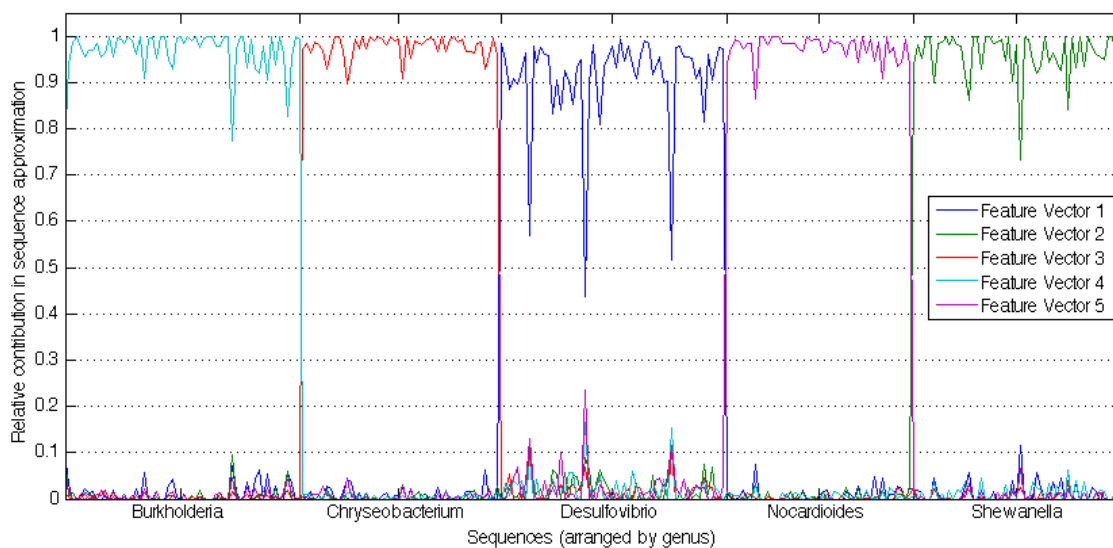


Figure 8.1: Relative contribution amounts used in the reconstruction of each sample from the set of five basis vectors.

Clearly, this set of feature vectors can be used to predict the genus of a sample, but how do we interpret the feature vectors themselves? Consider that when a basis

vector is used in the reconstruction or approximation of a sample, we indicate that the sample’s DNA sequence contains some distinct collection of k-mers. As a result, if we can determine which k-mers are in these collections, we can use them to construct a set of microarray probes that can detect this same set of features.

To determine which oligos best represent a particular feature (our “keywords”), we can project individual k-mers into our feature space and note how well they associate with the corresponding basis vector. Note that this is equivalent to simply observing the columns of the projection matrix $B = (A^T A)^{-1} A^T$ for a basis set A . In this matrix, the magnitude of each element $B_{i,j}$ provides an indication of the strength with which the j^{th} k-mer implies the i^{th} feature.

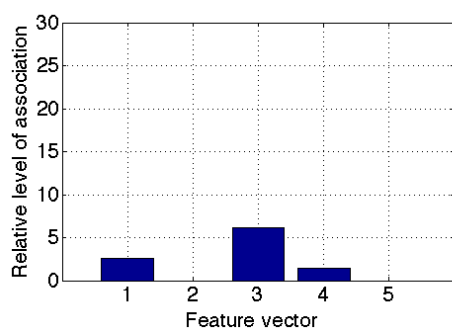


Figure 8.2: Levels of association for the k-mer *ctgttat*

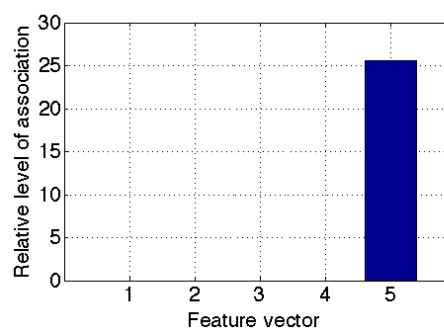


Figure 8.3: Levels of association for the k-mer *tggggtc*

Figures 8.2 and 8.3 demonstrate the relative levels of association for the k-mers *ctgtta* and *tggggtc* with each of our five basis vectors. In these figures, the “relative level of association” indicates the amount that each feature vector contributes in the approximation of the k-mer, relative to the average contribution amount (the mean of B). As noted, we can use this measure as an indication of k-mer’s association with each feature. Demonstrated in Figure 8.2, we can see that *ctgttat* is somewhat weak indicator of features 1, 3, and 4. On the other hand, *tggggtc* appears to be a clear indication of feature vector 5 and, thus, the genus *Nocardioides*.

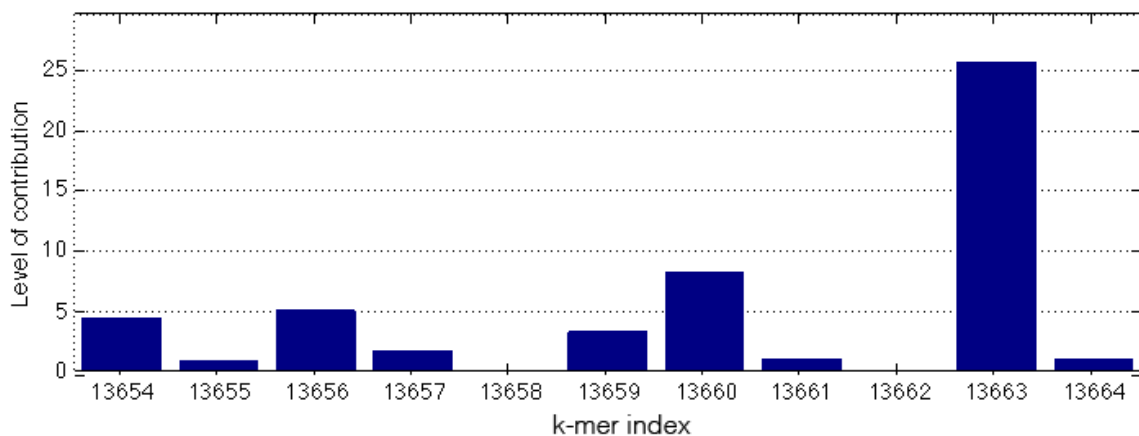


Figure 8.4: Relative contribution amounts used in the reconstruction of each sample from the set of five basis vectors.

In Figure 8.4, we see the levels of association for feature vector 5 over a range of the 16,384 possible 7-mers. Shown in the right, index number 13,663 corresponds to the k-mer *tggggtc*, which we just noted as a strong representative of this feature. From these figures, it seems that we can construct a set of microarray probes by simply finding the collection of k-mers with the highest levels of association for each of the features that we wish to detect. To test this theory, we select a set of candidate microarray probes for *Nocardiooides* by sorting the list of k-mers by their level of association and picking the top five.

Rank	k-mer	Level of Assoc.	TP	FP	TN	FN
1	<i>gaccca</i>	25	47	0	221	0
2	<i>tggggtc</i>	25	47	1	220	0
3	<i>agcaacg</i>	22	47	3	218	0
4	<i>cgttgct</i>	22	3	0	221	44
4*	<i>agcaacg</i>	22	47	3	218	0
5	<i>gcatgcg</i>	21	46	0	221	1

Table 8.1: Candidate probe oligos (* indicates the previous round was repeated with the reverse complement of the k-mer.)

Table 8.1 shows the results of searching for each candidate oligo in our collection

of 268 samples. A true positive (TP), then, indicates the number of the 47 possible *Nocardioides* species that contain the k-mer in question and thus should hybridize to the probe. The number of false positives (FP) indicates the number of species outside of *Nocardioides* that also contained the oligo. As can be seen in the table, this collection of probes works very well in detecting only the target sequences. We do, however, notice a cause for concern in the fourth k-mer.

Recall that when we build our k-mer–sequence vectors, we account for the fact that DNA contains two complementary strands by counting k-mers as well as their reverse complements when we profile the sequence. As an result, in the case of probe 4 in our table, we have actually found the reverse complement of the probe that we want. By using the probe’s complement in row 4*, we obtain much better performance and results that are consistent with the rest of the table.

If we are not trying to detect double-stranded genetic material, like is the case with RNA probes, we could instead profile sequences without counting reverse complements as we profile the sequence. Table 8.2 shows results for the same experiment repeated using this method of profiling.

Rank	k-mer	Level of Assoc.	TP	FP	TN	FN
1	<i>cgcagat</i>	35	47	1	220	0
2	<i>cagcaac</i>	34	47	3	218	0
3	<i>ttggcgcg</i>	34	47	3	218	0
4	<i>agcaacg</i>	34	47	3	218	0
5	<i>cgtcacg</i>	33	46	0	221	1

Table 8.2: Candidate probe oligos found by using alternative profiling method (without counting reverse complements in k-mer profile)

As can be seen, this approach yields similar results but with slightly higher levels of association. This makes sense, considering that because we’re not counting reverse complements in our profile, there is a lower chance of two unrelated sequences con-

taining the same k-mer. That said, despite using a different profiling scheme, both approaches produce similar sets of candidate oligos, with *agcaacg* being found in the top five for both.

Looking at the results from these two examples, in both cases we were able to identify a unique set of k-mers that reliably detect the presence of our target collection of *Nocardioides* samples. In addition, by requiring that a microarray contains a combination of these probes, we can maintain a high level of detection while further reducing our already-small number of false positives. As a final demonstration, we now apply this same technique to a larger data set.

Our new dataset contains a total of 750 16s sequences, with 75 taken from each of the following, randomly selected genera: *Bacillus*, *Burkholderia*, *Corynebacterium*, *Enterococcus*, *Halomonas*, *Nocardia*, *Pseudomonas*, *Streptococcus*, and *Vibrio*. We begin as before by training a projection matrix for this collection of sequences, using 10 feature vectors to define the LSA-NMF space.

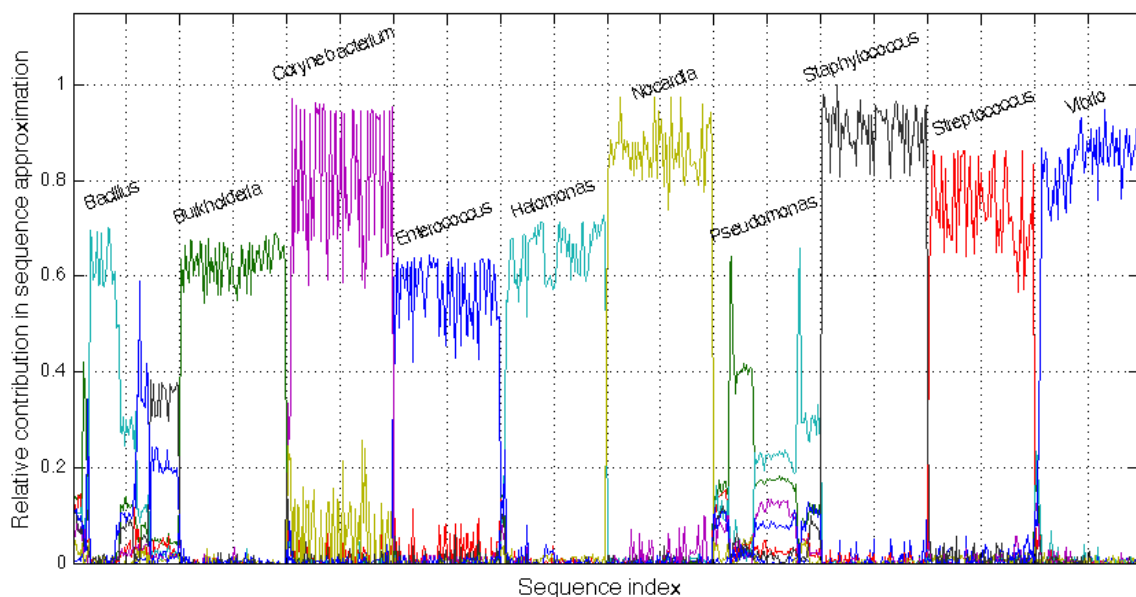


Figure 8.5: Relative contribution amounts used in the reconstruction of the larger dataset from the set of ten basis vectors.

Figure 8.5 demonstrates which feature vectors are used in the approximation of each sample in the collection. As can be seen, our feature vectors roughly correspond to the ten genera but not perfectly. The encodings corresponding to genera *Bacillus* and *Pseudomonas* are wildly unpredictable and do not seem to associate well with any one particular feature vector. Upon inspection, these results are consistent with biological research that indicates a high level of diversity in the 16s sequences for *Bacillus*[66] and *Pseudomonas*[67]. In fact, a number of species from *Pseudomonas* have even been relocated into other genera like *Burkholderia*[68]. This research points out that it may not be possible to perfectly reconstruct phylogenies using only 16s sequences. As a result, we should not expect to be able to construct feature vectors that reliably detect these genera.

With these considerations in mind, Figure 8.5 suddenly looks much better, and we are ready to select the oligos for our microarray probes. To demonstrate, we will use feature vector that seems to best indicate *Streptococcus*, shown in red in the figure.

Rank	k-mer	Level of Assoc.	TP	FP	TN	FN
1	<i>ctgaagt</i>	37	64	1	674	11
2	<i>taggtcc</i>	35	73	1	674	2
3	<i>tcggtga</i>	35	68	86	589	7
4	<i>aagggac</i>	33	74	1	674	1
5	<i>aggtccc</i>	33	73	0	675	2
6	<i>gtgctag</i>	33	75	1	674	0
7	<i>aggtggt</i>	33	74	2	673	1
8	<i>gttgtat</i>	33	71	0	675	4
9	<i>gtaggtc</i>	33	74	2	673	1
10	<i>ctttccg</i>	32	71	0	675	4

Table 8.3: Candidate probe oligos for *Streptococcus*

Table 8.3 shows the ten oligos with the highest levels of association for the *Streptococcus* feature vector. As shown, this set of probes very reliably detects the presence of our target samples, while producing an extremely low number of false positives in

nearly every case. Again, by requiring a combination of these k-mers, we can further reduce the likelihood of false positives.

To make a few final observations on these results, recall that when selecting candidate oligos for our probes, we made use of a k-mer's "relative level of association" with each feature vector. In our experiments, we simply chose the subset of oligos with the highest levels of association. By doing this, however, we have perhaps incorrectly assumed that the k-mer has little association with any other feature vector. This problem can be avoided by applying a weighting to deemphasize k-mers which are highly associated with multiple features.

Finally, in a large, diverse dataset, it might be the case that no oligo is highly representative of a feature. By increasing our k-mer size, though, we increase the number of possible keywords that can be used to define a feature, and by keeping track of the relative levels of association for a dataset, we can determine whether or not the k-mer size should be increased. While 7-mers were used in this small example, in practice, we presume that larger probe sizes will be required to keep false positive rates low. Naturally, by increasing the length of our probes, we achieve higher specificity, but by doing so only when and where necessary, we can design an efficient set of microarray probes of minimal length.

Chapter 9

Conclusion and Recommendations for Future Work

In this research, latent semantic analysis was evaluated as a potential collection of techniques for differentiating and classifying genomic sequences by modeling them as unordered sets of distinct, fixed-length words. In the presented experiments, dimensionality reduction was performed using non-negative matrix factorization to identify sets of basis vectors to approximate sequences in high-dimensional LSA-NMF spaces. These basis vectors, due to the non-negativity constraints of NMF, represent collections of oligonucleotides whose collective presence was shown to indicate latent biological features.

By projecting sequences into these high-dimensional feature spaces, we found that the encodings or contributions of each element in the basis set provides a short profile that can be used to identify groups of sequences with similar biological features. By computing the distance or correlation coefficient between pairs of these profiles, a new form of evolutionary distance measure was created and was used to construct phylogenetic trees, cluster 16s ribosomal genes, and remove unwanted host DNA fragments

from a synthetic metagenomic sample. Finally, the feature vectors themselves were found provide a collection of biological keywords, or sets of oligonucleotides whose collective presence is indicative of some biological feature.

Throughout each of these three experiments, we made reference to a number of considerations for large-scale implementation. At their core, each of the presented applications relies on linear algebra techniques to rotate, scale, or otherwise transform elements in a vector space. These elements were noted as being particularly sparse, and as a result, these techniques lend themselves to simple yet highly efficient implementations. In addition, many practical applications could be designed around pre-computed projection matrices to further minimize their level of computational burden.

For each of these reasons, we conclude that latent semantic analysis provides a collection of techniques that will be greatly beneficial in dividing and organizing the enormous amounts of biological information being generated by today's scientists. And as new technologies emerge and these amounts become even larger, LSA-based techniques provide efficient methods for retrieving and interpreting data in ways that enhance our efforts of decoding the genetic language in which we are written.

9.1 Recommendations for Future Work

In Chapters 6 and 7, it was shown that LSA-NMF may be used to construct high-dimensional feature spaces capable of distinguishing between relatively short (~ 1 kbp) genomic sequences. We might be able to extend this approach to allow for the classification of longer and even whole-genome sequences, but there are several considerations that we should bear in mind.

First, in the presented experiments, a k-mer size of 7bp was chosen, resulting in

k-mer-sequence profiles of 16,384 elements. It's not hard to image, however, that when comparing sequences that are several billion bases in length, this number of elements might not be enough to distinguish between two samples. As a result, in order to effectively distinguish between longer sequences, perhaps much larger k-mer sizes should be investigated.

Alternatively, because we have seen that LSA-NMF can be used to classify short sequences, it may be possible to use a sliding window approach to sample a whole genome, classify each windowed region, and use a list of the classified elements as a profile to characterize the sequence. For example, consider that certain genes like the 16s ribosomal sequences that we have seen are remarkably similar across species, and as such, we could theoretically train a set of feature vectors to detect such genes. With this ability, we could classify sequences not by unordered collections of k-mers but by unordered collections of *genes*.

Supposing that it is possible to reliably detect specific genes in this manner, we are led to another potential application involving the construction of phylogenetic trees. Recall that in Chapter 8 it was noted that the 16s ribosomal gene alone may not be enough to perfectly reconstruct phylogenies [66]. On the other hand, if we are able to detect and therefore extract other genes which carry phylogenetic signals (like the *recA* gene that was used to differentiate between species of *Burkholderia* [62]), we may be able to characterize samples using a combination of several genes in order to construct more accurate phylogenies. Such a system could attempt to classify an unknown sample using each gene separately and combine the results to identify the species.

This notion of using a series of classifications or clusterings to profile a genomic sequence, as effective as it may be, sounds incredibly inefficient at first mention. However, recall that when projecting a sample into an LSA-NMF space, due to our

non-negativity constraints, a crude form of clustering may be achieved by simply noting the feature vector (or perhaps subset of feature vectors) with the largest contribution in the reconstruction. Combining this with the fact that we can devise and precompute a collection of projection matrices, suddenly, the idea of using a series of classifications sounds much more feasible.

In fact, this leads us to a final and particularly exciting potential application for LSA-NMF which involves using a series of projection matrices to partition and search large collections of sequences. Suppose, for example, that given an unknown sample, we would like to find the most similar item in a very large database of known organisms. Using the method described above, it may be possible to construct a collection of projection matrices that classify a sequence at various levels of taxonomic specificity, providing a sort of hashing function into the database.

To illustrate this idea, consider Figure 9.1 which describes a system in which an incoming sample is profiled as a k -mer-sequence vector. This profile is then projected into an initial feature space using the projection matrix B_0 , which has been designed to differentiate organisms at, say, the phylum level. Next, having identified the sample

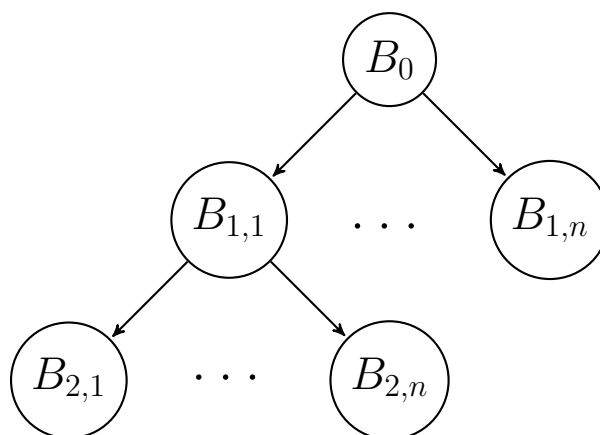


Figure 9.1: Using projection matrices to construct a divide and conquer databs

as belonging to phylum m , we select the corresponding projection matrix $B_{1,m}$ which breaks this phylum into smaller subgroups. This process is repeated for as many steps are necessary to classify the sample, providing a divide and conquer strategy for searching large sequence collections.

As a final remark, it is worth repeating that each of the techniques presented in this research are, at some level, reliant on linear algebra methods for factorizing and multiplying matrices. These types of operations lend themselves to extremely efficient implementations using recent advancements in parallel computing platforms like Nvidia's CUDA architecture [69]. As a suggested direction for future work, final implementations of the LSA-NMF based methods described in this work could benefit greatly from using the CUDA architecture and the CULA library of GPU-accelerated linear algebra routines [70].

Bibliography

- [1] Committee on and National Research Council. *A New Biology for the 21st Century*. National Academies Press, November 2009.
- [2] D. R. Mattoon and B. Schweitzer. Profiling protein interaction networks with functional protein microarrays. In Yuri Nikolsky, Julie Bryant, and John M. Walker, editors, *Protein Networks and Pathway Analysis*, volume 563 of *Methods in Molecular Biology*, pages 63–74. Humana Press, 2009.
- [3] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453, 1970.
- [4] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195 – 197, 1981.
- [5] D. J. Russell, H. H. Otu, and K. Sayood. Grammar-based distance in progressive multiple sequence alignment. *BMC Bioinformatics*, 9(306), July 10, 2008.
- [6] K. Katoh and H. Toh. Parallelization of the MAFFT multiple sequence alignment program. *Bioinformatics*, 26(15):1899–1900, 2010.
- [7] T. H. Jukes and C. R. Cantor. *Evolution of Protein Molecules*. Academy Press, 1969.

- [8] M. Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of molecular evolution*, 16(2):111–120, 1980.
- [9] J. Felsenstein. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.
- [10] M. Hasegawa, H. Kishino, and T. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 22:160–174, 1985.
- [11] S. Tavaré. *Some Probabilistic and Statistical Problems in the Analysis of DNA Sequences*, volume 17, pages 57–86. Amer Mathematical Society, 1986.
- [12] K. Tamura. Estimation of the number of nucleotide substitutions when there are strong transition-transversion and G+C-content biases. *Molecular Biology and Evolution*, 9(4):678–687, 1992.
- [13] K. Tamura and M. Nei. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular Biology and Evolution*, 10(3):512–526, 1993.
- [14] H. H. Otu and K. Sayood. A new sequence distance measure for phylogenetic tree construction. *Bioinformatics*, 19(16):2122–2130, 2003.
- [15] M. Li and P. Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 1997.
- [16] A. Lempel and J. Ziv. On the complexity of finite sequences. *Information Theory, IEEE Transactions on*, 22(1):75 – 81, 1976.

- [17] M. Bauer, S. M. Schuster, and K. Sayood. The average mutual information profile as a genomic signature. *BMC Bioinformatics*, 9(48), January 25, 2008.
- [18] O. U. Nalbantoglu, S. F. Way, S. H. Hinrichs, and K. Sayood. RAIphy: Phylogenetic classification of metagenomics samples using iterative refinement of relative abundance index profiles. *BMC Bioinformatics*, 12(41), July 31, 2011.
- [19] L. Krause, N. N. Diaz, A. Goesmann, S. Kelley, T. W. Nattkemper, F. Rohwer, R. A. Edwards, and J. Stoye. Phylogenetic classification of short environmental DNA fragments. *Nucleic Acids Research*, 36(7):2230–2239, 2008.
- [20] D. H. Huson, A. F. Auch, J. Qi, and S. C. Schuster. MEGAN analysis of metagenomic data. *Genome Research*, 17(3):377–386, 2007.
- [21] A. Brady and S. L. Salzber. Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models. *Nature Methods*, 6:673–676, 2009.
- [22] C. W. Schmidt. Data explosion: Bringing order to chaos with bioinformatics. *Environ Health Perspect*, 111(6), 05 2003.
- [23] J. Shendure and H. Ji. Next-generation dna sequencing. *Nature Biotechnology*, 26(10):1135 – 1145, 2008.
- [24] F. S. Collins, M. Morgan, and A. Patrinos. The human genome project: Lessons from large-scale biology. *Science*, 300(5617):286–290, 2003.
- [25] R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, S. Li, H. Yang, J. Wang, and J. Wang. De novo assembly of human genomes with massively parallel short read sequencing. *Genome Research*, 2009.

- [26] K. A. Wetterstrand. DNA sequencing costs: Data from the NHGRI large-scale genome. <http://www.genome.gov/sequencingcosts>, 2012.
- [27] T. K. Landauer, P. W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse Processes*, 25(2-3):259–284, 1998.
- [28] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI conference on Human factors in computing systems, CHI '88*, pages 281–285, New York, NY, USA, 1988. ACM.
- [29] S. T. Dumais. Latent semantic analysis. *Annual Review of Information Science and Technology*, 38(1):188–230, 2004.
- [30] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513 – 523, 1988.
- [31] S. Robertson. Understanding inverse document frequency: On theoretical arguments for IDF. In *Journal of Documentation*, volume 60, 2004.
- [32] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Commun. ACM*, 30(11):964–971, November 1987.
- [33] G. Golub and W. Kahan. Calculating the Singular Values and Pseudo-Inverse of a Matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.
- [34] G. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14:403–420, 1970. 10.1007/BF02163027.

- [35] S. Rajamanickam. *Efficient Algorithms for Sparse Singular Value Decomposition*. PhD thesis, University of Florida, 2009.
- [36] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.
- [37] D. D. Lee and H. S. Seung. Unsupervised learning by convex and conic coding. In *Advances in Neural Information Processing Systems 9*, pages 515–521. MIT Press, 1997.
- [38] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
- [39] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, January 1991.
- [40] S. E. Palmer. Hierarchical structure in perceptual representation. *Cognitive Psychology*, 9(4):441 – 474, 1977.
- [41] E. Wachsmuth, M. W. Oram, and D. I. Perrett. Recognition of objects and their component parts: Responses of single units in the temporal cortex of the macaque. *Cerebral Cortex*, 4(5):509–522, 1994.
- [42] N. K. Logothetis and D. L. Sheinberg. Visual object recognition. *Annu. Rev. Neurosci.*, 94:4577–621, 1996.
- [43] M. Berry, M. Browne, A. Langville, V. Pauca, and R. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis*, 52(1):155–173, September 2007.

- [44] P. Paatero. The multilinear engine: A table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *Journal of Computational and Graphical Statistics*, 8(4):854–888, 1999.
- [45] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval*, SIGIR '03, pages 267–273, New York, NY, USA, 2003. ACM.
- [46] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, 1999.
- [47] L.J.P. van der Maaten, E. O. Postma, and H. J. van den Herik. Dimensionality reduction: A comparative review. http://www.iai.uni-bonn.de/~jz/dimensionality_reduction_a_comparative_review.pdf, 2007.
- [48] R. L. Somorjai, B. Dolenko, A. Demko, M. Mandelzweig, A. E. Nikulin, R. Baumgartner, and N. J. Pizzi. Mapping high-dimensional data onto a relative distance plane—an exact method for visualizing and characterizing high-dimensional patterns. *Journal of Biomedical Informatics*, 37(5):366 – 379, 2004.
- [49] J. X. Li. Visualization of high dimensional data with relational perspective map. *Information Visualization*, 3(1):49–59, 2004.
- [50] J. Venna. *Dimensionality Reduction For Visual Exploration of Similarity Structures*. PhD thesis, Helsinki University of Technology, 2007. PhD Thesis.
- [51] E. S. Tzafestas, A. Nikolaidou, and S. G. Tzafestas. Performance evaluation and dynamic node generation criteria for ‘principal component analysis’ neural networks. *Mathematics and Computers in Simulation*, 51(3-4):145 – 156, 2000.

- [52] I. T. Jolliffe. *Principal Component Analysis*. Springer, 2nd edition, 2002.
- [53] I. T. Jolliffe. A Note on the Use of Principal Components in Regression. *Applied Statistics*, 31(3):300+, 1982.
- [54] J. Felsenstein. PHYLIP (Phylogeny Inference Package) version 3.5c. Distributed by the author. Department of Genetics, University of Washington, Seattle, 2003.
- [55] C. R. Woese. Bacterial evolution. *Microbiological Reviews*, 51(2):221–271, June 1987.
- [56] D. J. Lane, B. Pace, G. J. Olsen, D. A. Stahl, M. L. Sogin, and N. R. Pace. Rapid determination of 16S ribosomal RNA sequences for phylogenetic analyses. *Proceedings of the National Academy of Sciences*, 82(20):6955–6959, October 1985.
- [57] J. R. Cole, Q. Wang, E. Cardenas, J. Fish, B. Chai, R. J. Farris, A. S. Kulam-Syed-Mohideen, D. M. McGarrell, T. Marsh, G. M. Garrity, and J. M. Tiedje. The ribosomal database project: improved alignments and new tools for rRNA analysis. *Nucleic Acids Research*, 37(suppl 1):D141–D145, 2009.
- [58] Y. Li and A. Ngom. Non-negative matrix and tensor factorization based classification of clinical microarray gene expression data. In *Bioinformatics and Biomedicine (BIBM), 2010 IEEE International Conference on*, pages 438–443, dec. 2010.
- [59] J. Kim and H. Park. Fast nonnegative matrix factorization: An active-set-like method and comparisons. *SIAM J. Scientific Computing*, 33(6):3261–3281, 2011.

- [60] C. H. Q. Ding, T. Li, and M. I. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(1):45–55, January 2010.
- [61] C. Lin. Projected gradient methods for non-negative matrix factorization. In *Neural Computation*, volume 19, pages 2756–2779, 2007.
- [62] G. W. Payne, P. Vandamme, S. H. Morgan, J. J. LiPuma, T. Coenye, A. J. Weightman, T. H. Jones, and E. Mahenthiralingam. Development of a recA Gene-Based Identification Approach for the Entire Burkholderia Genus. *Applied and Environmental Microbiology*, 71(7):3917–3927, July 2005.
- [63] BAuA. Classification of bacteria and archaea in risk groups. <http://www.baua.de>, 2010.
- [64] H. Chung, S. J. Pamp, J. A. Hill, N. K. Surana, S. M. Edelman, E. B. Troy, N. C. Reading, E. J. Villablanca, S. Wang, J. R. Mora, Y. Umesaki, D. Mathis, C. Benoist, D. A. Relman, and D. L. Kasper. Gut immune maturation depends on colonization with a host-specific microbiota. *Cell*, 149(7):1578 – 1593, 2012.
- [65] R. B. Stoughton. Applications of DNA microarrays in biology. *Annual Review of Biochemistry*, 74(1):53–82, 2005.
- [66] G. E. Fox, J. D. Wisotzkey, and P. Jurtshuk. How close is close: 16S rRNA sequence identity may not be sufficient to guarantee species identity. *International Journal of Systematic Bacteriology*, 42(1):166–170, 1992.
- [67] F. Widmer, R. J. Seidler, P. M. Gillevet, L. S. Watrud, and G. D. Di Giovanni. A Highly Selective PCR Protocol for Detecting 16S rRNA Genes of the Genus

- Pseudomonas (Sensu Stricto) in Environmental Samples. *Applied and Environmental Microbiology*, 64(7):2545–2553, 1998.
- [68] B. W. Holloway. *Pseudomonas genetics and taxonomy*, pages 22–32. ASM Press, Washington, D. C., 1996.
- [69] NVIDIA Corporation. *NVIDIA CUDA Compute Unified Device Architecture Programming Guide*. NVIDIA Corporation, 2007.
- [70] EM Photonics. CULA Tools: GPU Accelerated Linear Algebra. <http://www.culatools.com/>, 2010.