UNIVERSITY OF CALIFORNIA,
IRVINE

Computational Approaches Toward a Polyadenylation Code

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Lingjie Weng

Dissertation Committee:
Professor Xiaohui Xie, Chair
Professor Yongsheng Shi
Professor Pierre Baldi

2014

DEDICATION

I dedicate this work to my parents and my sisters.
I am grateful for their unconditional love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

It has been a long journey, and numerous people over the years have helped me get here, so there are many people I would like to thank.

Firstly, I would like to thank my PhD advisor Dr. Xiaohui Xie for his support and guidance throughout my PhD training. I am thankful to him for taking me into computer science department from MCSB gateway program. Along the way I learned much from Dr. Xie, ranging from how to do critical thinking, discover, and investigate a scientific problem, ending with how to writing papers or preparing talks. I am grateful to him for teaching me all these. I also truly thank Dr. Yongsheng Shi for consistent help, both academically and financially. Dr. Shi acts as my co-advisor, his frequent insights and patience with me are always appreciated. I enjoyed working with him in an open and friendly environment. Without their consistent help and support, there will be no chance that I could make this far.

I would particularly like to thank professor Pierre Baldi for serving as my dissertation committee member, it's my great honor to have you there. Your 'Bioinformatics-The Machine Learning Approach' book is the first article that brought me into this field. I also learned a lot from your scientific writing class.

I would like to thank all members in Xie lab, thanks for being so nice to me, and always offering a lot of helps and suggestions. I enjoyed working in the same lab with you. I would also like to thanks all my collaborators, professor Chen Li and Hobbes team members, professor Zhaoxia Yu, professor Steve Potkin and people in his lab as well as people in Shi lab, I gained a lot from their helpful and useful discussions with me on research projects, I gained a lot. It's a great experience to work with all of you.

My beginnings at UCI were in MCSB program and I want to thank all faculty members there. I would particularly like to thank Karen Martin, I received warmest welcome from her even before I came to UCI, and she continuously helped me out till the very end of my PhD study.

Thanks to Eric Bax, my mentor during internship in Yahoo! Labs. Thank you for being so nice to me, for teaching me how to laugh at myself, friendship is more important than covering more works for your workmate, since friendship is life-long.

There are also many friends who have made an impact on my time at UC Irvine. Thank you for your patience and kindness. I'm so honored to have walked my graduate times with you and looking forward to many more to come.

The portion of the chapter two and five is a reprint of the materials as it appears in Nucleotide Acid Research and BMC bioinformatics. The co-authors listed in this publication helped and supervised research which forms the basis for this thesis. So many thanks to those two journals for publishing my works, and also thanks my collaborators for allowing me to incorporate the work into my thesis.

And finally, special thanks to my parents, my sisters and my boyfriend for their unconditional love and support.

CURRICULUM VITAE

Lingjie Weng

EDUCATION

Doctor of Philosophy in Computer Science **2014**
University of California, Irvine *Irvine, California*

Bachelor of Science in Bioinformatics **2008**
Zhejiang University *Hangzhou, Zhejiang, China*

RESEARCH EXPERIENCE

Graduate Research Assistant **2009–2014**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2014**
University of California, Irvine *Irvine, California*

WORKING EXPERIENCE

Intern Scientist **2013**
Yahoo! Lab *Burbank, California*

PROFESSIONAL SERVICE

ad hoc reviewing PLOS ONE Journal

REFEREED JOURNAL PUBLICATIONS

SNP-based pathway enrichment analysis for genome-wide association studies **2011**
BMC Bioinformatics

Hobbes: optimized gram-based methods for efficient read alignment **2012**
Nucleic Acids Research

Transcriptome-wide analyses of CstF64RNA interactions in global regulation of mRNA alternative polyadenylation **2012**
Proceedings of the National Academy of Sciences

Overlapping and distinct functions of CstF64 and CstF64 in mammalian mRNA 3 processing **2013**
RNA

Fip1 regulates mRNA alternative polyadenylation to promote stem cell selfrenewal **2014**
The EMBO journal

Global ProteinRNA Interaction Mapping at Single Nucleotide Resolution by iCLIP-Seq **2014**
Spliceosomal Pre-mRNA Splicing

CPSF30 and Wdr33 directly bind to AAUAAA in mammalian mRNA 3 processing **2014**
Genes & development

Validation of k-Nearest Neighbor Classifiers Using Inclusion and Exclusion **2014**
arXiv preprint arXiv:1410.2500

SOFTWARE

SSEA `https://cbcl.ics.uci.edu/SSEA/`
JAVA Package for SNP-based pathway enrichment analysis with user-friendly interface

Hobbes `https://hobbes.ics.uci.edu/`
Hobbes: optimized gram-based methods for efficient read alignment

ABSTRACT OF THE DISSERTATION

Computational Approaches Toward a Polyadenylation Code

By

Lingjie Weng

Doctor of Philosophy in Computer Science

University of California, Irvine, 2014

Professor Xiaohui Xie, Chair

Messenger RNA 3' polyadenylation (poly(A)) is an essential post-transcriptional processing step for most eukaryotic genes, significantly impacting many aspects of mRNA metabolism. The majority of eukaryotic genes present alternative poly(A) (APA), through which the same gene can have multiple alternative 3' ends due to the cleavage and poly(A) presence at distinct sites. APA results in RNA transcripts with different 3UTRs, which can influence transcript transport, localization, stability, and translation, or lead to different protein products. Many human diseases including cancer have been associated with abnormal poly(A) regulation, highlighting the importance of this process. However, the rules on how poly(A) sites are selected and regulated - the so called the poly(A) code - are not well understood.

Recent advances in high-throughput technologies have provided a great opportunity to elucidate the rules underlying APA. High-throughput sequencing(HTS) experiments yield a wealth of data regarding APA. Consequently, there is a need to develop computational techniques to mine these data. In this thesis, we present four major contributions furthering our understanding of the poly(A) code. The algorithms and computational methods we developed have all showed improved predictive and analytical capabilities over competing methods. They are as follows:

1) HTS reads need to be efficiently mapped back to a reference genome for further downstream analysis. To address this need, we developed a fast and accurate reads mapping package for identi-

fying all mapping locations for each read, called Hobbes. Hobbes outperforms most state-of-the-art all-mapping programs, including mrsFast and Razers2.

2) We developed a bioinformatics pipeline for identifying and profiling genes with significant APA switches from different biological or clinical conditions. The pipeline includes calling poly(A) sites, filtering artificial poly(A) sites, clustering heterogeneous poly(A) sites, and identifying and profiling genes with significant APA switches. This pipeline has already provided significant insights into many core polyadenylation factors.

3) The poly(A) code can be partially deciphered from the genome-wide modeling of tissue-specific APA. Consequently, extended existing Shannon entropy measuring to assess the tissue specificity for each poly(A) site, and applied an outlier detection method to identifying the tissue-specific pattern. With new mRNA features we explored, our ensemble predictive model successfully discriminated tissue-specific poly(A) sites from constitutive poly(A) sites, with test accuracy 84.5% (auRoc 0.92), which surpassed the previous model by more than 10%. Through an in-depth analysis of the most important features, we proposed a mechanism that controls the selection and regulation of tissue-specific APA.

4) Aberrant mRNA 3' polyadenylation has been implicated for a wide variety of complex diseases. We developed a novel statistical method for identifying disease-related pathway from genome-wide association studies (GWAS). We proposed to optimally select a representative SNP (single nucleotide polymorphism) set for each gene using adaptive truncated product statistic, and conducted enrichment analysis via the weighted Kolmogorov-Smirnov test to identify enriched pathways. By applying it to Schizophrenia GWAS SNPs, we showed our method identifies pathways highly associated with the disease. Moreover, the results are reproducible across large genetically distinct samples. This method can be used for detecting pathways involved in disease caused by APA, such as cancer.

Chapter 1

Introduction

The central dogma of molecular biology describes information in genes flowing from DNA to RNA (transcription) and from RNA to protein (translation). In a prokaryotic cell, transcription and translation are coupled; that is, translation begins while the messenger RNA (mRNA) is still being synthesized. However, in a eukaryotic cell, transcription occurs in the nucleus, while translation occurs in the cytoplasm. Genetic information from DNA is first transcribed into RNA, and this primary transcript mRNA (pre-mRNA) needs go through post-transcriptional processes to become a mature mRNA, which will then be translated into proteins. The post-transcriptional processes include three main modifications in the cell nucleus, including 5' capping, RNA splicing, and 3' polyadenylation(poly(A)). Polyadenylaton occurs at the 3' end of the pre-mRNA molecule, it typically involves a cleavage of its 3' end and then the addition of a long string of adenine residues to form a polyadenylated tail [27, 132, 21]. Poly(A) tails are essential structural and functional elements of eukaryotic mRNAs, significantly impacting many aspects of mRNA metabolism. For example, as the poly(A) tails are synthesized, they bind to multiple copies of poly(A) binding proteins, which helps regulate mRNA stability, localization and translation.

Recent genome-wide poly(A) analyses have indicated that alternative polyadenylation (APA), a

mechanism in which the same gene has multiple 3' ends due to the recognition of multiple distinct cleavage and poly(A) sites, is pervasive in eukaryotes [127, 112, 82]. $\sim 70\%$ of human genes have multiple potential poly(A) sites [33]. As a result, a larger portion of RNA transcripts have different 3' untranslated regions (UTRs), or encode different proteins. APA not only expands the proteomic and functional diversity, but also plays an important role in gene regulation regarding mRNA metabolism [104, 36]. Many studies have revealed that polyadenylation and APA regulation defects would cause or contribute to the development of a wide spectrum of human disease [41, 82, 55, 87, 32], highlighting the importance of this process. However, the rules on how poly(A) sites are recognized and how their recognitions are regulated – the so called the polyadenylation code – are not well understood.

Due to recent technological breakthroughs, next-generation sequencing (NGS) technologies are making sequence cheaper and quicker. For instance, in January 2014, Illumina announced its new top-of-the-line system, the HiSeq X Ten Sequencing System, which is capable of sequencing complete human genomes at $1000 each, and has a throughput of 600 billion base pairs per day. These NGS technological advances allow high-throughput sequencing to become a practical tool for many areas of biology and medicine. A number of high-throughput sequencing methods have been developed specifically for studying polyadenylation, including 3'-seq [75], PAS-seq [103], and direct RNA sequencing (DRS) [92]. The large scale of data arising from these genome-wide sequencing represent an unprecedented opportunity for the study of the poly(A) code, but also a major challenge. Consequently, there is a need to develop computational methods to mine hidden patterns from these data.

In this thesis, we present four major contributions furthering our understanding of the poly(A) code through computational and algorithmic methods developments. More specifically, we developed an efficient and scalable algorithm for mapping high-throughput sequencing reads to reference genome for further downstream analysis; a bioinformatics pipeline for identifying and profiling genes regulated by APA; an ensemble predictive model to explore the regulation rules of tissue-

specific APA regulation; and a robust and accurate statistical method for detecting disease-related pathways.

The thesis is outlined as follows:

To transform high-throughput sequencing reads into hypotheses and conclusions, an important first step is to efficiently map them back to a reference genome. In chapter 2, we describe a package for fast and accurate reads mapping called "Hobbes" that can find all mapping locations for a read. Because sequencing reads are short (36-100 bps), they can potentially map to multiple locations throughout the genome, which for human is about 3 gigabase pairs. We identified two bottlenecks that significantly affect the performance of gram-based read mapping, and proposed two novel techniques to overcome these two bottlenecks. As a result, Hobbes is faster than most state-of-the-art "all-mapper" programs, including mrsFast and Razers, while maintaining high mapping quality. In contrast, most read mappers focus only on uniquely mappable reads, which discards many reads and hence loses a large amount of information. Hobbes' all-mapping feature, on the other hand, provides a platform for inferring additional biological insights that would be otherwise missed. Hobbes is released as an open-source software package, which is available at http://hobbes.ics.uci.edu/. Portions of this chapter were published as part of [2].

In chapter 3, we introduce a bioinformatics pipeline for identifying and profiling genes with significant APA switches. The pipeline includes calling real poly(A) sites, filtering artificial poly(A) sites due to internal priming, clustering heterogeneous poly(A) sites and identifying and profiling genes with significant APA switches. This pipeline has been successfully applied to the APA analyses of many core polyadenylation factors, including Fip1, CstF64, CstF64-tau, and has already provided significant insights about their functionality and contributions to poly(A) associated diseases. An important step in the pipeline is the mapping of sequencing reads to a reference genome. For the read mapping step, the pipeline currently uses Bowtie, an efficient read mapper that only focuses on uniquely mappable regions of the genome. One obvious way to augment the bioinformatics pipeline is to replace the current read mapper with Hobbes in order to draw additional biological

insights that would otherwise be missed by focusing only on uniquely mappable portions of the genome.

The poly(A) code can be partially deciphered from the rules of tissue-specific APA regulation. In chapter 4, we present a predictive model for distinguishing tissue-specific and constitutive mRNA alternative polyadenylation. We extended existing Shannon entropy method to assess the tissue specificity for each poly(A) site, and applied an outlier detection method to identify the tissue-specific pattern for each tissue-specific poly(A) site. We explored many new RNA features, such as features describing transcript sequence structure into our model. In total, we assembled 658 RNA features, covering all major parameters known or with great potential to influence polyadenylation regulation. We then trained an ensemble predictive model, which successfully discriminated tissue-specific poly(A) sites from constitutive poly(A) sites, with test accuracy 84.5% (auROC 0.92). We evaluated the importance of different feature sets, identifying transcript sequence structure feature set is very important in determining tissue-specific APA. More specifically, for alternative poly(A) sites in the same exon, the distance between alternative poly(A) sites is a key feature discriminating tissue-specific and constitutive poly(A) sites; for alternative poly(A) sites in different exons, the distance from the intronic poly(A) sites to its closest upstream 5' splice site as well as the strength of the 5' splice site greatly affect tissue-specific APA regulation. In addition, we found the evolutionary conservation level surrounding poly(A) sites to be important for tissue-specific APA regulation. Based on the information we mined from genome-wide tissue-specific modeling, we proposed a mechanism that controls the selection and regulation of tissue-specific APA.

Aberrant mRNA 3' polyadenylation has been implicated for a wide variety of complex diseases. In chapter 5, we present our contribution to genome-wide association studies: a novel statistical method for identifying disease-related pathway. We propose to optimally select a representative SNP (single nucleotide polymorphism) set for each gene using adaptive truncated product statistic, and conduct enrichment analysis via the weighted Kolmogorov-Smirnov test to identify enriched pathways. By applying it to Schizophrenia GWAS SNPs, we showed our method identifies path-

ways highly associated with the disease, confirmed by experimental results on many published articles. Moreover, the results are reproducible across large genetically distinct samples. This method can be used for detecting pathways involved in disease caused by deregulation of APA, such as cancer. These contributions are released as an open-source software package called SSEA, available at https://cbcl.ics.uci.edu//SSEA/. Portions of this chapter were published as part of [122].

Chapter 2

Hobbes: optimzed gram-based methods for efficient read alignment

2.1 Introduction

Due to recent technological breakthroughs, whole genome sequencing using next-generation sequencing (NGS) technologies is becoming cheaper and quicker, and is gaining popularity in many areas of biology and medicine. Sequence alignment is an essential step in many of the biological applications like genome sequence variation, mapping of protein binding sites, gene prediction, etc. The enormous amount of reads produced from the sequencers poses a great challenge on the speed and the accuracy of read alignment programs for two major reasons. First, the reference sequence can be very large. For instance, the human genome is about 3 billion base pairs long. Mapping a billion reads to the human genome amounts to check 3×10^{18} candidate locations. Second, due to sequencing errors and/or genetic variations, many reads map to the reference sequence approximately but not exactly, and therefore, to map a read to the reference sequence, read mapping programs should allow a certain number of mismatches between the read and a candidate

location.

A considerable amount of time and effort have been dedicated to handle the problem of exact and inexact alignment. Many algorithms have been developed over the years that address compression and querying of read sequences. It still took days or even weeks (depending on different mapping criteria) to align billions of reads to a large reference genome on a single desktop. And for sequence alignment with insertion and deletions, many kinds heuristic approaches were used to speed up at the cost of moderate accuracy. Hence, a faster and accurate read alignment tool is highly in demand to cater to the changing needs of the sequencing domain.

Related work: Existing approaches to the read-mapping problem can be broadly classified into two categories: trie-based methods and gram-based methods. In the first group, they use tree data structures, like suffic tries, enhanced suffic arrays, FM-index [38] to build indices. FM-index is a careful combination of Burrows-wheeler transform [18] and the suffix array structure, to provide a reduced space usage and efficient method for finding all occurrences of a substring in a reference sequence with exact matching. Each character in the read is matched against a suffix trie of the reference generated using the BWT. This index gives a substantial compression without affecting the mapping times and most of popular packages like Bowtie [68], BWA [72], and SOAP2 [74] are implemented under this principle.

These packages have a very small memory footprint (about 2 GB), and are very efficient for finding a few mappings for short reads with not too many mismatches. They use backtracking to allow mismatches during the tree traversal, and therefore, their performance deteriorates as the read length and the number of mismatches increase. BWT-based packages are typically not designed for finding a large number of mappings per read.

The gram-based methods follow a filter-and-verify paradigm. Using grams, they first identify a set of candidate mappings, and then verify the true distance for those candidates to remove false positives. The candidate-generation step is often supported by an inverted index on grams (from

the reads and/or the reference sequence), leading to a relatively large memory footprint. Early packages like SSAHA [89] and BLAST [7] had long mapping times, infeasible for large data sets. Newer packages like Maq [73], RMAP [108], ZOOM [77], SHRiMP [99], RazerS [120], mrsFAST [46], and mrFAST-CO [6] offer significant improvements, but they do not consistently outperform BWT-based methods. We will show that Hobbes outperforms both existing gram-based and BWT-based methods.

Hobbes: Applications may differ in their requirements on a read-mapping package. Sometimes finding a couple of mappings per read is sufficient, but other times the application may need all mapping positions. For instance, in RNA-seq applications, due to the occurrence of homologous genes and multiple RNA isoforms originated from the same gene, finding all mapping positions will be necessary for quantifying the expression level of a particular gene isoform [57]. Similarly, in ChIP-seq applications, finding all mapping positions is a necessary step for characterizing protein binding patterns in repeat regions of a genome [88, 26].

Hobbes is a gram-based read mapper, supports Hamming and edit distance, and is efficient in both of those situations. Hobbes is about 2-10 times faster than state-of-the-art packages when finding all mappings per read, and performs comparably when looking for a few mappings. Hobbes is also at least as accurate as other packages.

In the following sections, we identify two performance bottlenecks of existing gram-based approaches, and make two major contributions to overcome them: First, we present a novel technique for judiciously choosing a small set of grams of each read to generate candidate mappings. Second, we develop a cache- and CPU-efficient filter for removing false positive mappings during the traversal of inverted lists.

2.2 Materials and Methods

We first discuss how to map reads to the reference sequence with a given Hamming-distance threshold, and later extend our solution to support edit distance. Our approach is based on generating overlapping q-grams of the reference sequence, and constructing an inverted index of those q-gram positions. To map a read, we generate its q-grams, and access the inverted index to compute a superset of all mapping positions. We then remove false-positive positions by computing the real distance of the read to the subsequences starting at those positions in the reference sequence. Next, we summarize the basic q-gram method focusing on Hamming distance, although some techniques directly apply to edit distance as well.

2.2.1 Basic Q-Gram Method

For a positive integer q, the q-grams of a sequence are all its overlapping substrings of length q. For example, the 3-grams of a sequence $\mathtt{s} = \mathtt{TGCCCTA}$ are $\mathtt{G(s)} = \{(\mathtt{1}, \mathtt{TGC}), (\mathtt{2}, \mathtt{GCC}), (\mathtt{3}, \mathtt{CCC}), (\mathtt{4}, \mathtt{CCT}), (\mathtt{5}, \mathtt{CTA})\}$.

Approximate subsequence matching using q-grams is based on the following intuition: If two sequences are similar, then they share a certain number of q-grams. For the Hamming distance this idea has been formalized as *count filtering* [115].

Count filtering: If two sequences r and s are within Hamming distance d, then their q-gram sets $G(s)$ and $G(r)$ share at least the following number of q-grams:

$$T = \max(|G(r)|, |G(s)|) - d * q. \tag{2.1}$$

The lower bound T on common grams in the above equation is based on the observation that a character substitution can affect at most q grams, hence, d substitutions can affect at most $d * q$

grams.

Gram filtering variants: Other variants of filtering use multiple patterns based on the pigeonhole principle [102], non-overlapping grams [89], gapped grams [11, 17] or variable-length grams [71].

Q-gram inverted index: Finding substrings in a reference sequence that share at least T q-grams with a given read can be facilitated with an inverted index on q-gram positions, explained as follows for Hamming distance. Figure 5.1 shows an example of a 5-gram inverted index. To map a read ACGGTCTTCCCTACGGT within Hamming distance $d = 2$ and $T = 17 - 5 + 1 - 2 * 5 = 3$, we first look up the read's 5-grams in the inverted index. Notice that only the grams ACGGT and CGGTC (underlined in the read) are present in the index. We traverse their inverted lists, and normalize each element relative to the position of the corresponding gram in the read. For example, the 5-gram CGGTC appears at position 2 in the read, so the relative position of the element on CGGTC's inverted list is $106 - 2 + 1 = 105$. In this way, we can count how many of the read's grams are contained in substrings of the reference sequence starting at a fixed position (position 105, in this example). The gram ACGGT appears twice in the read, and we treat each occurrence as a separate list. Its appearance at position 1 yields a normalized list of $\{105 - 1 + 1 = 105, 118 - 1 + 1 = 118\}$, and a list $\{105 - 14 + 1 = 92, 118 - 14 + 1 = 105\}$ for position 14. Next, we count the occurrences of each element on the normalized lists. The positions 92 and 118 are pruned according to the count filter, because their occurrences do not meet the lower bound of $T = 3$. Position 105 has a count of 3, and therefore, it is a candidate answer whose Hamming distance to the read still needs to be computed.

Performance issues of q-gram counting: For a long reference sequence, the above approach for mapping reads could suffer from the following performance problems:

1. *CPU intensive gram counting.* Using all of a read's relevant inverted lists for gram counting can be expensive if there are many of them, or if some of the lists are very long. The cost of gram counting is directly related to the total number of elements on those inverted lists.

2. *Cache misses during candidate verification.* CPU caches are very small but fast memories that act as intermediaries to main memory. Transferring new data into the cache (a "cache miss") can last hundreds of CPU cycles. Accessing random portions of a very long sequence has low locality, and hence, it can become a performance bottleneck due to cache misses.

In the Materials and Methods Section we present a new technique to judiciously select a few grams from a read to overcome the first performance issue. To tackle the second issue, we develop a novel filter based on augmenting the inverted lists with additional information.

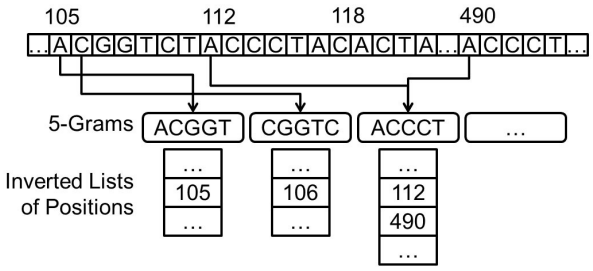


Figure 2.1: Excerpt of a reference sequence and a portion of its 5-gram inverted index. The inverted lists of the 5-grams ACGGT, CGGTC, and ACCCT are shown, each containing a sorted list of positions in the reference sequence at which the respective 5-gram appears.

2.2.2 Judiciously Selecting Q-Grams From Reads

In this section we aim to reduce the cost of processing inverted lists to generate candidate mappings. We present a new technique to judiciously select a few grams from a read to minimize the number of corresponding inverted lists, and the number of inverted-list elements that need to be considered for mapping the read.

Existing methods for q-gram selection

First, we briefly summarize the main ideas already developed in the literature to reduce the number of grams.

Prefix Filter: Consider a read s with a gram set $G(s)$ and the lower bound T in Equation 2.1. Let the "prefix gram set" of read s be the $|G(s)| - T + 1$ least frequent grams in $G(s)$, i.e., with the shortest inverted lists. A candidate mapping must share at least one gram with the prefix gram set of s, because otherwise it could only reach a maximum count of $T - 1$ [24].

Shortened Prefix: Xiao et al. [125] use the positions of q-grams to reduce the size of the prefix gram set in the context of edit-distance based joins. Their solution imposes a global ordering on the grams based on their frequency to achieve a consistent notion of prefix gram set across all strings, and constructs a q-gram inverted index on prefix grams on-the-fly. To improve the index-construction time (the dominating factor) they reduce the prefix gram set of each string to the first $i \leq |G(s)| - T + 1$ grams in the global ordering which contain $d + 1$ non-overlapping grams, where d is a given edit distance threshold. Inspired by their work, we develop a new method to judiciously select a few q-grams for probing our index.

Optimal q-gram prefix selection

Recall that a substitution can affect at most q grams (Equation 2.1). The insight that these q affected grams must be overlapping lead us to develop the following lower bound based on the positions of q-grams.

Lemma 1. *(Position-Based Prefix) Given a sequence s and its q-gram set $G(s)$, let P be a subset of $d + 1$ non-overlapping q-grams from $G(s)$. Then each sequence within Hamming distance d of s must have a gram in P.*

The intuition of the lemma is as follows. A substitution at position p can affect at most q overlapping grams, namely those starting from a position in $[p - q - 1, p]$. Since non-overlapping grams are at least q positions apart from each other, d substitutions can affect at most d non-overlapping grams. Among $d + 1$ non-overlapping grams, at least one gram remains unaffected by d substitutions. Since this analysis is true for every subset P of $d + 1$ non-overlapping grams of $G(s)$, we

can generalize the lower bound as follows.

Lemma 2. *(Generalized Position-Based Prefix) A sequence r within Hamming distance d of sequence s must share at least k grams with every subset of $d + k$ non-overlapping q-grams of s.*

Optimal prefix selection: We want to select a set of prefix grams that is optimal in the sense that (1) it refers to a minimum number of inverted lists, and (2) those inverted lists have the minimum total number of elements. The position-based prefix described above satisfies (1), but a read could have many possible sets of $d+1$ non-overlapping q-grams. To satisfy (2) we develop the following dynamic programming algorithm to select that set of $d+1$ non-overlapping q-grams from the read (Supplementary Data), which minimizes the total number of corresponding inverted-list elements.

Subproblem: Let $1 \leq i \leq d+1$ and $1 \leq j \leq |G(s)| - d * q$ be two integers. Let $M(i,j)$ be a lower bound on the sum of the lengths of the inverted lists of i non-overlapping grams starting from a position no greater than $j + (i-1) * q$. Our goal is to compute $M(d+1, |G(s)| - d * q)$.

Initialization: Let $L[p]$ denote the inverted list corresponding to the q-gram at position p, and $L[p].len$ its length. We initialize the row $M(0,j)$ to zero, and the column and $M(i,0)$ to infinity.

Recurrence function:

$$M(i,j) = \min \begin{cases} M(i,j-1) \\ M(i-1,j) + L[j + (i-1) * q].len \end{cases} \tag{2.2}$$

Example: Figure 2.2 shows an example of finding an optimal q-gram prefix of a read $\mathtt{s} = \mathtt{GGTCTCACCCTGAACTAA}$, gram length $q = 5$, and Hamming distance threshold $d = 2$. An optimal set of positions of the $d + 1 = 3$ non-overlapping q-grams are highlighted, including the cell in the matrix from which the q-gram position can be inferred. For example, "4" is the minimum value in the first row, and since its first appearance is in column 2, we can infer that the first optimal q-gram position is at $2 + (1-1) * q = 2$.

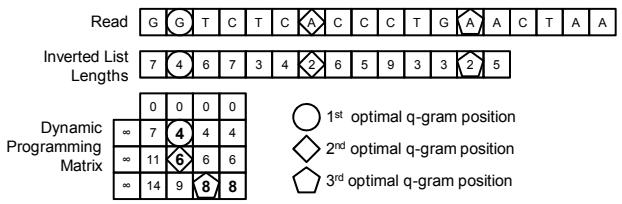


Figure 2.2: Example of our dynamic programming algorithm for finding an optimal set of prefix grams for a read GGTCTCACCCTGAACTAA, gram length $q = 5$, and Hamming distance $d = 2$. Optimal gram positions are highlighted with a circle, a diamond, and a pentagon.

Complexity: The complexity of the algorithm for finding an optimal prefix for a read s with length $|s|$ and Hamming distance d is $O(|s|d)$. Notice that the actual cost of the algorithm decreases when we increase d and q, because there are fewer sets of non-overlapping grams to choose from. For example, in Figure 2.2 we need to populate 12 matrix cells for a read of length 18. Our experiments show that the algorithm performs very well for good d and q values in real data sets.

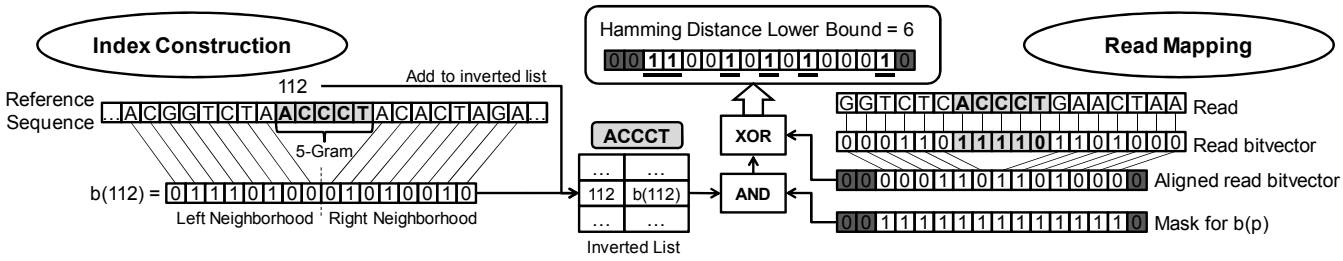


Figure 2.3: Adding bitvectors to a q-gram inverted index (left), and pruning candidates mappings with them (right), using a mapping of $A, T \Rightarrow 0$, and $C, G \Rightarrow 1$. The left portion shows how to encode the left and right neighborhood of a 5-gram ACCCT at position 112 in the reference sequence as a 16-bit bitvector, mapping $A, T \Rightarrow 0$, and $C, G \Rightarrow 1$. Both the position 112 and its bitvector $b(112)$ are inserted into ACCCT's inverted list. The right portion shows how to prune candidate mappings of a read GGTCTCACCCTGAACTAA from ACCCT's inverted list. The dark grey boxes indicate invalid bits we must ignore, based on ACCCT's position in the read. The light grey boxes highlight the matching q-gram ACCCT.

2.2.3 Cache-Efficient Filtering of Candidate Mappings

A straightforward implementation of the q-gram-based filter and verification procedure can lead to poor performance due to cache misses. Since the reference sequence is often much larger (e.g.,

3GB for the human genome) than the CPU cache (e.g., 12MB L3 cache for an Intel Xeon X5670), each verification of a candidate position likely causes a cache miss. Since most candidate positions typically are false positives, paying a cache miss for fetching an irrelevant substring of the reference sequence is very wasteful.

In this section, we present a cache- and compute-efficient filter for removing false-positive candidate mappings without accessing the reference sequence. The main idea is to augment the inverted lists with additional filtering data, such that it is in the CPU cache during the traversal of an inverted list.

Mapping q-gram neighborhoods to bitvectors: We attach to each inverted-list element an encoding of its corresponding neighboring characters in the reference sequence using 1 bit per character. The left hand side of Figure 2.3 illustrates this procedure on an exemplary 5-gram at position 112 in a reference sequence. We use 16 bits to encode the 8 characters to the left and right of the 5-gram ACCCT. Since we only access ACCCT's inverted list if ACCCT is also contained in the read we are processing, it is unnecessary to include ACCCT itself in the bitvector. The size of the bitvectors is a tunable parameter, and we use 16 bits for this example.

It might seem that using a single bit per character reduces the filtering capability by 50% because we map strings of a 4-letter alphabet $(\mathtt{A}, \mathtt{C}, \mathtt{T}, \mathtt{G})$ to a 2-letter alphabet $(0, 1)$. But, it is well known that not every character substitution is equally likely on real data [29]. For example, Table 2.3 shows the frequency of character substitutions we gathered in a simple experiment, as follows. For each of the 35bp reads we computed the optimal gram prefix and traversed the corresponding inverted lists to obtain candidate mapping positions. Next, we recorded the frequency of character substitutions during the verification of these candidate positions. Since "$\mathtt{A} \rightarrow \mathtt{G}$" and "$\mathtt{T} \rightarrow \mathtt{C}$" are the most frequent substitutions, our results suggest the following encoding: $\mathtt{A}, \mathtt{T} \Rightarrow 0$ and $\mathtt{C}, \mathtt{G} \Rightarrow 1$, such that the characters of the most frequent substitutions get different bit values.

Apart from representing more characters with a fixed number of bits, our bitvector encoding also

Table 2.1: Frequency of character substitutions using 2 million 35bp reads on hg18. The results suggest a mapping: $\texttt{A}, \texttt{T} \Rightarrow 0$ and $\texttt{C}, \texttt{G} \Rightarrow 1$.

Read character	hg18 character	# of substitutions
A	T	687,276,051
A	G	**1,382,950,075**
A	C	559,937,841
T	G	395,839,922
T	C	**1,232,657,183**
G	C	393,616,199

allows CPU-efficient filtering of candidate positions, as follows:

Candidate filtering using bitvectors: Let us revisit the example in Figure 2.2. After computing an optimal gram prefix we need to traverse their corresponding inverted lists to find candidate mappings. Suppose we begin with the list of gram ACCCT, since it is the shortest list of those in the optimal prefix.

The right-hand side of Figure 2.3 illustrates how we use the bitvectors for filtering. Before scanning ACCCT's inverted list, we map the read to a bitvector. Next, we "shift away" the bits of the matching q-gram ACCCT in the read's bitvector to align the positions of its bits with those in the reference-sequence bitvectors. Recall that we omitted the q-gram itself when generating bitvectors for the reference sequence. This shifting produces invalid bits at both ends of the read bitvector shown as dark boxes. These invalid bits represent portions of the bitvectors from the reference sequence that we cannot use for pruning candidates. For example, since there are only 6 characters to left of ACCCT in the read, we should ignore 2 of the 8 bits representing the left neighborhood of ACCCT's occurrences in the reference sequence. We generate a bitmask to remove those invalid bits from each bitvector in ACCCT's inverted list.

Now that we have aligned the read's bitvector with the bitvectors in the inverted list, and generated a bitmask to remove invalid bits from those bitvectors, we start scanning ACCCT's inverted list. For each candidate position, we use the read's bitvector and the candidate's bitvector to compute

a lower bound on the Hamming distance between their corresponding original sequences, as follows. First, we do a "bitwise-AND" operation between the bitmask and the candidate's bitvector. Then, we do a "bitwise-XOR" operation between the resulting bitvector and the read's bitvector to produce a final bitvector. In the final bitvector, a bit is set to 1 if and only if the original character at the corresponding position in the read is different from the corresponding character in the reference sequence. We prune a candidate if the number of 1-bits in the final bitvector exceeds our Hamming distance threshold. We determine the number of 1-bits in the final bitvector with a single CPU instruction, `popcount`, supported by most modern CPUs.

In summary, our new bitvector-filtering technique eliminates candidate mappings without accessing the reference sequence, thus avoiding expensive cache misses. In addition, our filter only requires a handful of CPU instructions per inverted-list element, namely bitwise-AND, bitwise-XOR, `popcount`, and a final comparison with the Hamming distance threshold.

2.2.4 Supporting Insertions and Deletions

Allowing insertions and deletions (indels) is important for mapping longer reads, because both sequencing errors and genetic variations can result in the deletion/insertion of bases and the chance of this happening increases as reads become longer. However, finding mappings with indels is computationally more challenging. Hobbes implements the following two methods for mapping reads with indels:

Hamming distance tends to be sufficient for shorter reads, whereas edit distance becomes important for long reads. Ultimately, the user must decide whether the added benefit of edit distance offsets its computational cost.

Non-heuristic mapping: To find all mappings of a read while allowing indels, we can use the optimal prefix grams as described in the preceding section for generating candidate mapping positions.

However, the bitvector filter mentioned above is specific to Hamming distance. A similar filter for indels is possible, and we leave this direction for future work. To verify a candidate position, we conceptually consider those substrings with all possible starting and ending positions (based on the edit-distance threshold), and compute their edit distances to the read. For each candidate position, we report the substring with the lowest edit distance. This approach tends to be very slow, and the following heuristics can significantly improve the mapping performance.

Seed extension approach: Again, we begin with the optimal prefix grams for finding candidate positions. Next, we introduce two heuristics to improve performance: First, we fix the starting position for verification, but shift it to the left once, if the initial position yielded no match. Second, we apply the bitvector filter to the neighborhood of matching grams in the reference sequence, as show in Figure 2.4.

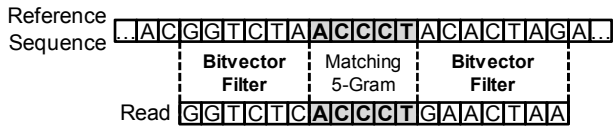


Figure 2.4: Seed extension approach with indels. We prune candidate positions by applying the bitvector filter on the neighborhood of matching grams.

Since most differences are due to substitutions, our intuition is that if the neighborhood already has a high Hamming distance to the corresponding substring in the read, then the candidate is probably not a match. The filter could remove valid mappings if those apparent substitutions are caused by inconveniently located indels. On the other hand, this effect is mitigated by using multiple grams. It is somewhat unlikely that the neighborhoods of all those grams have a high Hamming distance for true mappings. The bitvector-filter threshold on the neighborhood is a tuning parameter. We found that by setting it to 2/3 of the original edit distance, we capture most mappings while retaining high speed.

Letter count filter: Hobbes uses the letter-count filter described in [59] to quickly detect whether the two sequences can be within a given edit-distance threshold during the verification step. The

main idea of the filter is to count the number of occurrences of each character in both sequences, and establish a lower bound on the edit distance between the two sequences using the differences of those character counts, as follows. We divide the frequencies of all base-pairs into two groups. The first group contains the base-pairs that are more common in the read, and the second group contains those that are more common in the candidate. For each group, we create a sum of the frequency-differences of corresponding base-pairs in that group, denoted by Δ_1 and Δ_2, respectively. The grouping ensures that an edit operation can change each Δ by at most 1, establishing $max(\Delta_1, \Delta_2)$ as a lower bound on the edit distance between two sequences. For example, consider sequences $\mathtt{s_1} = \mathtt{AAACCTG}$ and $\mathtt{s_1} = \mathtt{CCCCTTGG}$, $\Delta_1 = (3 - 0) = 3$ (A is more frequent in s_1) and $\Delta_2 = (4 - 2) + (2 - 1) + (2 - 1) = 4$ (C, G, and T are more frequent in s_2). Based on the letter count filter, a lower bound on the edit distance between s_1 and s_2 is $max(3, 4) = 4$.

To prune candidates that survive the letter count filter, we also employ standard q-gram counting.

2.2.5 Supporting Paired-End Alignment

Hobbes supports the alignment of paired-end reads. Many next-generation sequencing technologies provide the user with paired-end reads that contain extra information about the relative position of two reads with respect to each other. To align paired-end reads, Hobbes initially considers each read of a pair separately and finds the set of candidate locations for each read. For example, given a read pair (r_1, r_2), Hobbes first finds the candidate location sets C_1 and C_2 corresponding to read r_1 and r_2 respectively. During the next step, for each candidate $c_1 \in C_1$ Hobbes performs verification only if there is a $c_2 \in C_2$ that satisfies the paired-end alignment constraints (appropriate orientation and distance) with respect to c_1.

Hobbes provides the option of reporting the alignments of each read in a paired-end read separately if no paired alignments are found.

2.2.6 Implementation Details

In this section, we discuss implementation details of Hobbes.

Treatment of `N` characters: We ignore q-grams with at least one `N` character. As a consequence, our inverted index does not contain those q-grams. When generating q-grams for a read, we may generate fewer than $d + 1$ non-overlapping q-grams (since we ignore q-grams with `N` characters). In our current implementation, we cannot find any mapping for such a read, although we could rely on those other q-grams to find *some* mappings (we leave this for future work). In all other cases, we treat `N`'s as mismatches. Note that we can deal with reads containing `N`'s (as long as there are enough non-overlapping q-grams), and a read can map to substrings in the reference sequence containing `N`'s even though the inverted index does not contain q-grams with `N`'s (we may reach such a position via a different, regular q-gram).

Hashing q-grams: We employ a collision-free hash function to map q-grams to integers, as follows. Each character $\{\texttt{A}, \texttt{C}, \texttt{T}, \texttt{G}\}$ is encoded as 2 bits, and the concatenation of all such 2-bits corresponding to a q-gram forms the q-gram's hash code. With this scheme, 32-bit integers can support hashing q-grams up to length $32/2 = 16$. For longer q-grams we use 64-bit integers.

Hamming-distance verification: Before computing the actual Hamming distance between two sequences using a character-by-character comparison, we do a significantly faster chunk-by-chunk comparison, typically with a chunk size of 4 bytes. If more than d chunks differ, then the two sequences cannot be within Hamming distance d, and we avoid the character-by-character comparison.

Edit-distance verification: After a candidate passes the letter count filter, we compute the real edit distance between two sequences using SeqAn's [35] implementation of Myer's bit-parallel algorithm [84].

Cache-prefetching during verification: Our bitvector filter can dramatically reduce the number

of cache misses by pruning false positives without accessing the reference sequence. However, the number of surviving candidates can still be in tens of thousands. Once a candidate has passed the bitvector filter, we cannot avoid a cache miss for the distance verification. But we can mitigate the cost by prefetching a future candidate's data into the cache, thus overlapping the verification of the current candidate and the data transfer from memory to cache for the future candidate. We have found that for our CPU architecture and our set of experiments the best performance is achieved when prefetching the data for candidate number $c + 2$ before verifying candidate number c.

2.3 Results

2.3.1 Implementation and Setup

We implemented Hobbes in C++, and compiled it with GCC 4.4.3. All experiments were run on a machine with 96GB of RAM, and dual quad-core Intel Xeons X5670 (8 cores total) at 2.93GHz, running a 64-bit Ubuntu OS. Note that Hobbes performs best on CPUs that support the `popcount` instruction. We used GCC's builtin functions for `popcount` and cache prefetching. Hobbes is freely available at http://hobbes.ics.uci.edu, and can output its results in SAM format for analysis with SAMTools.

2.3.2 Other Read Mappers and Data

We compared Hobbes with the following packages:

Bowtie [68] is a BWT-based short read aligner, and is efficient for finding few mappings per read (1 by default) with a very small memory footprint. Bowtie performs a DFS on the index and stops when the first qualified mapping is found.

BWA [72] is also a BWT-based program, and supports gapped alignment, while Bowtie does not. BWA uses a backtracking search similar to Bowtie's to handle mismatches. By default, BWA adopts an iterative strategy to accelerates its performance, at the price potentially losing mappings. To report all feasible mappings, we disable the iterative search ($-\texttt{N}$ option) in our experiments.

mrsFast [46] and **mrFast-CO** [6] are recent gram-based packages for gapped and ungapped alignment, respectively. They index both the reference genome and the reads. mrsFAST uses an efficient all-to-all list comparison algorithm, while mrFAST-CO follows a seed-and-extend strategy.

RazerS2 [120] builds a gram-based index on the reads, and performs gram counting while scanning over the reference sequence. RazerS2 has been reported to be very accurate in finding all mappings for typical read lengths. We set RazerS2's $\texttt{max} - \texttt{hit}$ parameter to 300,000,000 to get all mappings.

Data: We conducted our experiments using reads with 35, 51, 76, and 100 base pairs. The 35bp reads are taken from the YH database (http://yh.genomics.org.cn), the 51 and 76bp reads come from the DDBJ DNA Data Bank of Japan (DDBJ) repository (ftp://ftp.ddbj.nig.ac.jp) with entry DRX000359 and DRX000360 respectively, and the 100bp reads are from specimen HG00096 of the 1000 genome project (http://www.1000genomes.org/data). In all cases, we used the human genome with NCBI HG18 as our reference genome. As we do alignments read by read, the performance of Hobbes is almost linearly proportional to the number of reads. So in the following we mainly test the performance of Hobbes and other read mappers using datasets with 500K reads randomly chosen from the above mentioned databases.

2.3.3 Index Construction and Memory Footprint

We use an inverted index of overlapping q-grams on the reference genome. As described earlier, to avoid cache misses, we augment the inverted lists with bit vectors representing the neighbour-

ing characters of the corresponding q-grams. The index size is linearly dependent on the size of the reference sequence and the chosen bit-vector size. By default, Hobbes uses 16-bit vectors, resulting in a total index size of 21GB for hg18. We used bitvector sizes of 16 and 32 bits for our experiments with edit and Hamming distance, respectively. Using a single thread, it took Hobbes 20 minutes to build an index on hg18, whereas Bowtie and BWA needed 114 and 56 minutes, respectively. In addition, Hobbes has a tight-knit multi-threaded framework that parallelizes both indexing and mapping. On multi-core machines, users can build an index as large as hg18 in a few minutes. Because Hobbes does alignment read by read, its memory requirement is independent of the number of input reads.

2.3.4 Results Using Hamming Distance

All mappings: We configured the packages to find all mappings per read. Table 2.2 shows the mapping time, the fraction of reads with at least one mapping, and the total number of mappings for various read lengths and hamming distances. We observe that Hobbes is up to 5 times faster than other packages (even 100 times faster than RazerS2), while producing comparable mappings. For example, on 35bp reads, Hobbes is more than 4 times faster than Bowtie*, which is the fastest among all other listed programs; on 51bp and 76bp reads, Hobbes is about 3 times faster than our closest competitor BWA. Moreover, Hobbes maps slightly more reads than BWA in that setting. Among the tested programs, mrsFAST and RazerS2 consistenly achieved the best mapping quality. Hobbes delivers a similar quality, while outperforming mrsFAST and RazerS2 in mapping speed by a factor of up to 10 and 200, respectively.

Few mappings: Some applications may require all mappings per read, and others only a few mappings. Most tools are optimized for only one of those cases. For example, Bowtie focuses on finding a few mappings per read, and is very good at it. Therefore, the comparison in Table 2.2 somewhat disfavors those packages not designed for finding all mappings. To accommodate the

Table 2.2: Results of mapping 500K single-end reads against HG18.

Read length (Hamming)	35bp (2 errors)			51bp (3 errors)			76bp (3 errors)			76bp (4 errors)		
Algorithm	Time (h:m)	Reads mapped (%)	Total mappings (million)	Time (h:m)	Reads mapped (%)	Total mappings (million)	Time (h:m)	Reads mapped (%)	Total mappings (million)	Time (h:m)	Reads mapped (%)	Total mappings (million)
Bowtie*	0:28	76.61	492.6	0:34	91.93	317.1	0:16	91.44	73.4	NA	NA	NA
Bowtie	0:54	76.61	492.6	0:50	91.93	317.1	0:18	91.44	73.4	NA	NA	NA
BWA	0:30	76.61	492.6	0:24	91.61	277.6	0:10	91.36	71.1	0:18	92.47	115.2
mrsFAST	0:43	76.61	492.6	0:59	91.93	317.1	0:50	91.44	73.4	1:10	92.69	127.2
RazerS2	6:38	76.61	488.5	7:58	91.93	316.9	8:58	91.44	73.4	8:08	92.69	127.2
Hobbes	0:06	76.61	492.6	0:08	91.93	317.1	0:03	91.44	73.4	0:07	92.69	127.2

We used Bowtie in its default mode and an optimized mode (Bowtie*) where we set offrate=0 for maximum speed. *Reads mapped*: the fraction of reads with at least one mapping; *Total mappings*: the total number of mapped locations in the reference; *NA:* Bowtie does not support more than 3 mismatches.

Table 2.3: Frequency of character substitutions using 2 million 35bp reads on hg18. The results suggest a mapping: $\mathtt{A}, \mathtt{T} \Rightarrow 0$ and $\mathtt{C}, \mathtt{G} \Rightarrow 1$.

Read character	hg18 character	# of substitutions
A	T	687,276,051
A	G	**1,382,950,075**
A	C	559,937,841
T	G	395,839,922
T	C	**1,232,657,183**
G	C	393,616,199

few-mappings use case, Hobbes efficiently supports finding any number of mappings. Figure 2.5 shows the mapping times of BWA, Bowtie, and Hobbes for a varying number of requested mappings per read (k). We see that Hobbes performs comparably to Bowtie for very small k, but as k increases Hobbes begins to outperform other packages by a growing margin.

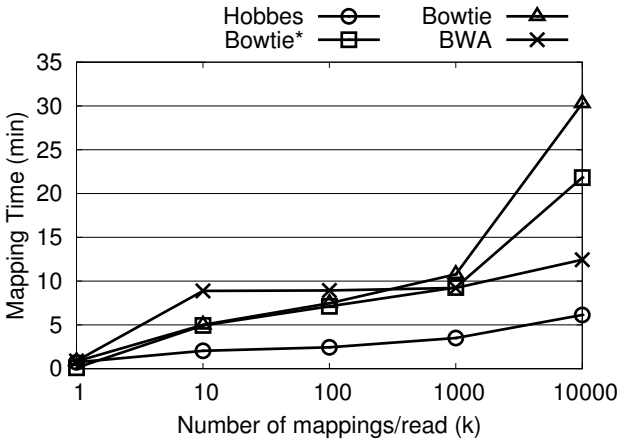


Figure 2.5: Maximun number of mappings k per read *vs*. mapping time on 51-bp reads with Hamming distance 3. We omitted RazerS2 due to its long mapping time, and mrsFAST because it only supports finding all mappings.

2.3.5 Results Using Edit Distance

Supporting edit distance is significantly more challenging than Hamming distance, and is becoming increasingly important as reads get longer and tend to contain indels. Hobbes implements a seed extension approach to align reads within a given edit distance threshold to take advantage of the two optimization strategies we have developed (Methods). Although unlike the Hamming distance mapping, the seed extension approach cannot guarantee to find all correct mapping locations, we will show that by using multiple seeds the mapping quality of Hobbes can be achieved to be nearly optimal. Bowtie does not support edit distance, so we compared Hobbes with BWA, mrFast-CO and RazerS2. We configured the packages to find all mappings per read.

All mappings: Table 2.4 lists our experimental results. We observed that Hobbes is about twice as fast as BWA and mrFast-CO, and over 7 times faster than RazerS2. Notice that the performance

gap increases with longer reads and higher edit distances. On 76bp reads, RazerS2 could map slightly more reads than the other packages, however, the mapping took an order of magnitude more time than Hobbes. The speed of mrFast-CO and BWA was similar, mapped slightly fewer reads. Hobbes was about twice as fast as mrFast-CO and BWA while mapping more reads. On 100bp reads, RazerS2 had the best quality but a comparably slow mapping speed; BWA become slower than RazerS2 and lost quality at the same time; the quality of Hobbes was very close to RazerS2. Compared to mrFast-CO, Hobbes could map more reads and was about 1.5 as fast. In addition, the current version of mrFast-CO is limited to edit distance 6, which is problematic for mapping even longer reads, while Hobbes does not have such a limitation.

Table 2.4: Results of mapping 500K single-end reads against HG18.

Read length (edit distance)	76bp (4 errors)			100bp (6 errors)		
Algorithm	Time (h:m)	Reads mapped (%)	Total mappings (million)	Time (h:m)	Reads mapped (%)	Total mappings (million)
BWA	02:33	94.06	141.1	22:54	92.16	79.4
mrFast-CO	02:45	94.28	142.3	03:47	92.39	96.3
RazerS2	12:26	94.32	143.6	17:14	92.50	96.4
Hobbes	01:46	94.30	145.8	02:48	92.47	100.7

2.3.6 Evaluation on Simulated Data

We simulated reads from the human genome using the wgsim program (http://github.com/lh3/wgsim), and then ran Hobbes to map those reads back to the same human genome. We used the default setting of wgsim, in which the mismatched bases are chosen randomly with a mismatch rate of 2% per base, and 15% of polymorphisms are indels with their sizes drawn from a geometric distribution with mean 1.43.

Since Hobbes is only guaranteed to be exact for Hamming distance, we use the simulated data to examine the mapping quality of Hobbes using edit distance. We use two metrics to measure

the accuracy of mapping: one is the fraction of reads with at least one mapping and the other is the mapping error rate. We say a read is aligned correctly if the true location of the read starts at the same location as one of its mappings. The mapping error rate is defined to be the fraction of mapped reads that are aligned incorrectly.

Table 2.5 shows the performance of Hobbes as compared to other programs in the case of edit distance. In terms of speed, Hobbes is the clear winner - about 3.5 times faster than BWA and 6 times faster than RaserS2 for 100bp reads. In terms of the fraction of reads mapped, Hobbes is slightly less than the best program, RaserS2, but the margin is small with a difference of only 0.02% for both 76bp and 100bp reads. Hobbes achieves the best mapping error rate of 0.22%. These results demonstrate that although Hobbes sacrifices some accuracy for speed, its mapping quality is comparatively better than the other packages tested.

Table 2.5: Results of mapping 500K simulated reads.

Read length (edit distance)	76bp (4 errors)			100bp (6 errors)		
Algorithm	Time (h:m)	Reads mapped (%)	Error rate (%)	Time (h:m)	Reads mapped (%)	Error rate (%)
BWA	01:22	96.05	2.17	07:55	97.09	1.78
mrFast-CO	02:15	97.84	3.43	03:22	99.43	3.63
RazerS2	10:08	97.90	0.98	12:59	99.50	1.15
Hobbes	01:08	97.88	0.22	02:20	99.48	0.22

2.3.7 Paired-End Alignment

We compared Hobbes with other state-of-the-art packages using paired-end reads.

Hamming distance: For Hamming distance, Hobbes is guaranteed to find all correct mappings. Table 2.6 summarizes the results of Hobbes and several programs for mapping reads of various lengths and Hamming distance thresholds to the human reference genome. We focus on the speed of mapping since the quality of mapping is similar among different programs. Hobbes is close

to Bowtie in the 35bp case, but substantially outperforms Bowtie (11 times faster) when the read length increases to 76bp and the Hamming distance increases to 3. Moreover, for the 100bp case with 4 errors, Bowtie was unable to provide answers because of too many backtracking steps required in the BWT-based algorithm. Hobbes is 24 times faster than the second-best program, mrsFAST in this case. These results suggest that Hobbes outperforms other programs in the Hamming distance case, especially for long reads while allowing many errors.

Edit distance: Our performance results are summarized in Table 2.7. The fraction of read pairs that can be aligned to the reference sequence is used as a surrogate of mapping quality. In terms of the fraction of mapped pairs, Hobbes is similar to RaserS2, both of which are significantly better than other programs. In terms of mapping speed, Hobbes is clearly the fastest in all three cases with big margins - 22 times faster than BWA, 3 times faster than mrFAST, and 15 times faster than RaserS2 in the 100bp case.

Table 2.6: Results of mapping 250K paired-end reads against HG18.

Read length (Hamming)	35bp (2 errors)		76bp (3 errors)		100bp (4 errors)	
Algorithm	Time (h:m)	Mapped pairs(%)	Time (h:m)	Mapped pairs(%)	Time (h:m)	Mapped pairs(%)
Bowtie	0:02	84.58	0:18	80.06	NA	NA
Bowtie*	0:20	84.68	0:25	80.06	NA	NA
mrsFAST	0:42	84.66	0:42	80.06	0:52	83.40
Hobbes	0:04	84.68	0:02	80.06	0:02	83.44

Mapped pairs: the fraction of read pairs mapped with at least one location satisfying the distance and orientation constraint. Bowtie*: running with -y option. Some entries contain `NA` because Bowtie does not support more than 3 mismatches.

2.3.8 Application in RNA-seq abundance analysis

In RNA-seq applications it is important to find all mapping positions of a read for quantifying the expression level of a particular gene isoform, due to the occurrence of homologous genes and multiple RNA isoforms originating from the same gene. To show the importance of finding

Table 2.7: Results of mapping 250K paired-end reads against HG18.

Read length (edit distance)	35bp (2 errors)		76bp (4 errors)		100bp (6 errors)	
Algorithm	Time (h:m)	Mapped pairs(%)	Time (h:m)	Mapped pairs(%)	Time (h:m)	Mapped pairs(%)
BWA*	1:02	84.93	2:15	84.45	22:48	88.14
mrFast-CO	2:32	78.02	2:32	81.61	03:36	85.96
RazerS2	3:57	84.53	10:52	84.49	15:02	88.18
Hobbes	0:26	85.35	0:21	84.60	0:50	88.37

* The raw number of mapped reads output by BWA are higher with 90.76%, 92.60% and 92.66% for 35, 76 and 100 reads, respectively. We removed those mappings that violated the edit distance constraints.

all mappings, we performed a transcript abundance analysis. We used the UCSC genes on the mm9 mouse (NCBI build 37) assembly as target transcripts; both 76bp paired-end RNA-seq reads were downloaded from Gene Expression Omnibus (Series GSE20846). We compared Bowtie and Hobbes for finding all mappings. The results in Table 2.8 show that Hobbes was 17-20x faster than Bowtie, and could even map slightly more reads.

Table 2.8: Results of mapping 76bp RNA-seq reads against 55,419 known mouse transcripts, using Hamming distance 3 and a minimum and maximum insert size of 76bp and 800bp, respectively.

Reads	SRR047951 (21.8 million)			SRR047953 (20 million)		
Algorithm	Time (h:m)	Reads mapped (%)	Total mappings (millions)	Time (h:m)	Reads mapped (%)	Total mappings (millions)
Bowtie	09:53	40.28	18.399	09:12	42.97	18.883
Hobbes	00:35	40.29	19.157	00:27	42.99	19.393

Next, we piped the result of Hobbes directly to eXpress (http://bio.math.berkeley.edu/eXpress), which is a streaming tool for quantifying the abundances of a set of target sequences from sampled subsequences. eXpress estimates the relative abundance by computing the fragments per kilobase per million (FPKM) value. We compared the FPKM values when finding at most $k = \{1, 10, 100\}$ mappings, with the FPKM when finding all mappings. To quantify their variation, we computed the ratio of the k-mappings FPKM to the all-mapppings FPKM (or its reciprocal). The results are shown in Table 2.9. We found that among 55,419 target transcripts, the percentage of transcriptions whose ratio was above 1.5 for $k = \{1, 10, 100\}$ was 42.39%, 0.92% and 0.07% respectively; there

was a lot of variability for $k = 1$, meaning the corresponding FPKM value is a poor estimator. The variability reduced as we increased k to 100. We included the corresponding scatter plots in Supplementary Data.

Table 2.9: Transcripts with FPKM ratio above 1.5 and 1.2 on 76bp RNA-seq reads within Hamming distance 3 against 55,419 known mouse transcripts.

	Transcriptions above ratio (%)	
k vs. All FPKM Ratio	1.5	1.2
k=1 vs. All	42.39	47.12
k=10 vs. All	00.92	01.42
k=100 vs. All	00.07	00.18

2.4 Discussion

Hobbes efficiently supports Hamming and edit distance while finding all mappings or few mappings per read. Our experiments have shown Hobbes to be just as accurate but significantly faster than current read mapping programs.

Hobbes has a large memory footprint (26GB in our experiments), but we believe its speed and mapping quality outweigh that drawback, especially considering today's low memory prices. We plan to reduce Hobbes memory requirement, possibly via compression, or by partitioning our index performing the read mapping one partition at a time (mrsFast and mrFast-CO follow this approach).

Given today's trend towards massively multi-core CPUs, we believe that good multi-thread support is of paramount importance. Both Hobbes' index-construction and read-mapping procedures support multiple threads and scale well (Supplementary Data). Some packages like mrsFast, mrFast-CO, and RazerS2 do not support multi-threading at all (we could not find this feature in their manuals). Other packages have certain limitations, e.g., in BWA only one of the two steps during read mapping is parallelizable.

We plan to further optimize Hobbes for the edit-distance based mapping, and account for read quality scores.

Chapter 3

Pipeline for analyzing PAS-seq reads

3.1 Introduction

The development of high-throughput sequencing technologies has changed nearly the entire field of genome biology. Several successful techniques have been developed to precisely sequence the 3' end of the mRNA based on high-throughput sequencing platforms, like Poly(A) Site Sequencing (PAS-seq) [103], Direct RNA Sequencing (DRS) [92]. PAS-seq is one the most popular protocols that are widely used in the study of polyadenylation. PAS-seq can quantitatively measure poly(A) profile. At first, mRNA is fragmented, a smart oligo (dT) primed reverse transcription (RT) reaction is carried out to capture the RNA fragment with a poly(A) stretch at the 3' end, the polymerase chain reaction (PCR) then amplifies the poly(A) junction, and the corresponding PCR products are subject to illumina sequencing [103]. We developed a bioinformatics pipeline for analyzing PAS-seq reads, for the purpose of globally profiling poly(A) sites, and identifying genes with significant APA switches between two biological conditions.

3.2 Bioinformatics pipeline to detect APA

The pipeline includes mapping reads back to a reference genome and calling real poly(A) site, filtering artificial poly(A) sites due to internal priming, clustering heterogeneous poly(A) sites and assigning representative poly(A) sites, identifying genes with statistically significant APA changes. This pipeline has been successfully applied to the APA analyses of many core polyadenylation factors, including Fip1 [67], CstF64 [128], CstF64τ [129], and has already provided significant insights about their functionality and contributions to poly(A) associated diseases. The flowchart of this pipeline is shown in Figure 3.1.

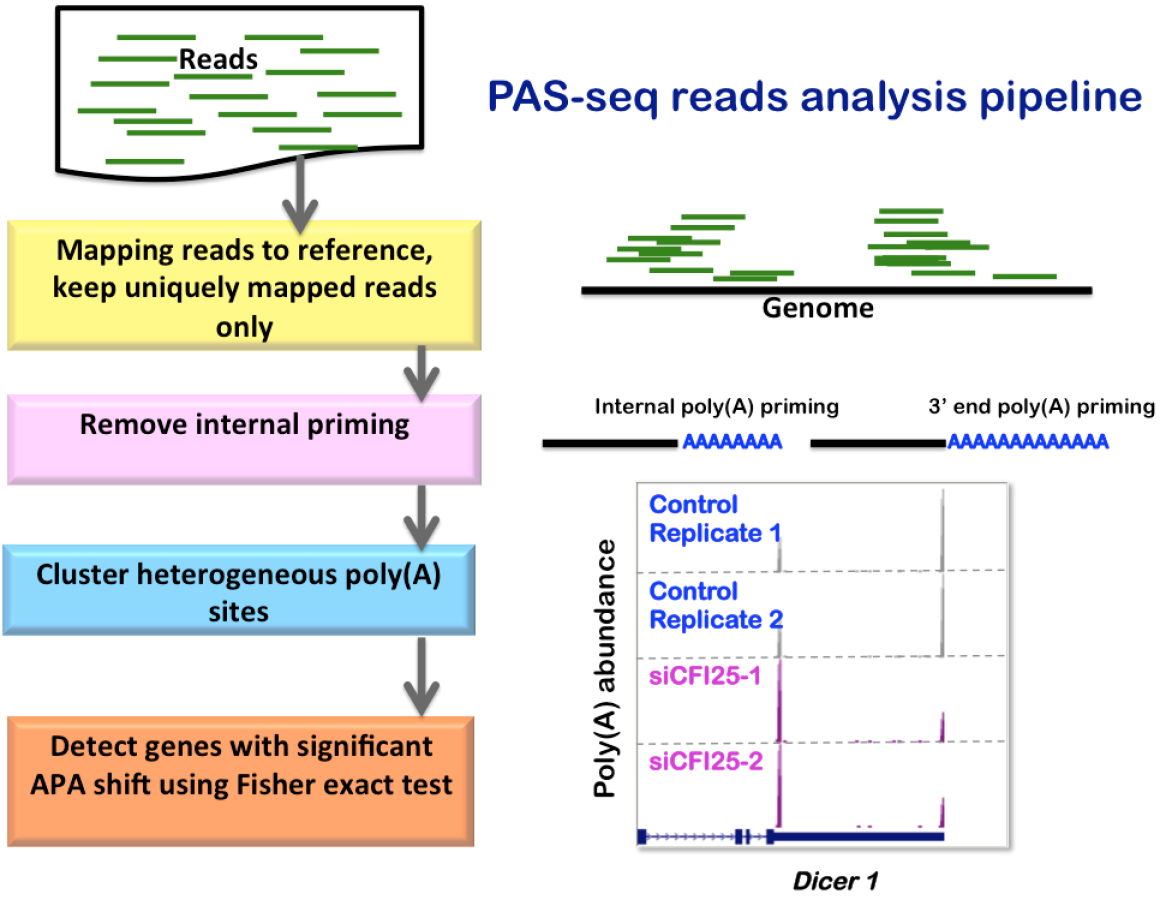


Figure 3.1: Bioinformatics pipeline for analyzing PAS-seq data

The very first step is to map reads back to reference genome. We use Bowtie [69] to map reads, and only uniquely mappable reads were retained for downstream analysis. According to the pro-

tocol of PAS-seq, the most 3' end position of mapped reads are putative poly(A) cleavage sites. Reads mapped to the same genome position and have the same barcode are considered to be PCR duplicates, are thus merged.

Secondly, PAS-seq protocol is efficient at deeply capturing APA events, but introduces a considerable amount of A-rich sequences because of oligo (dT) internal priming. So we need to remove reads with internal priming by checking if there are 6 continuous As or 7 As out of 10nt right after the 3' end of the reads.

Thirdly, because of the heterogeneity in poly(A) cleavage sites as well as sequencing errors, the exact poly(A) coordinates from the same transcript often have several nucleotide difference[93, 112]. In addition, downstream analysis requires a consensus polyA coordinate across different samples. Therefore, it is important to cluster nearby cleavage sites, and more importantly, to determine a representative position for each poly(A) cluster. To reduce the bias and get a consensus coordinate, we pool poly(A) sites from all samples together. For each cluster, we iteratively merge cleavage sites whose distance is within 40nt on the same chromosome, and assign the median location of all available cleavage sites as the representative poly(A) site, which is usually the peak position of this cluster. The abundance of a poly(A) sites is the total number of uniquely mapped reads landing inside cluster within each sample, and further normalized by sequencing depth (counts per million).

Lastly, pairwise comparisons of APA profiles between two biological samples/conditions are carried out, for example, APA profiles between control cell line and the same cell line with a certain poly(A) factor knock-down. For each poly(A) site, we conduct a Fisher exact test to compare the ratio of the abundance of this poly(A) site to the total abundance of remaining poly(A) sites on the same gene between two biological conditions. The derived P-value is then adjusted by multiple testing correction algorithm-Benjamini-Hochberg method. For each gene, we pick top two poly(A) sites with smallest P-values, and the poly(A) site closer to 5' end is called proximal site, and the other one is called distal site. Finally, genes with adjusted proximal and distal P-values less than a

given cutoff value are identified as genes with significant APA switches.

3.3 Summary

Genes showed statistically significant changes after a poly(A) factor is knocked down are believed to be regulated by this poly(A) factor. This pipeline helps biological researchers identify APA regulators, and therefore provides possible insights about the underlying mechanism of polyadenylation regulation.

Using only unique mappable reads helps remove false positive poly(A) sites, but loses a lot of information since only about 50% reads were retained. One way to augment the bioinformatics pipeline might keep reads mapped to multiple locations, and give each mapping location reasonable weights instead of throwing those information away.

Chapter 4

Tissue-specific alternative polyadenylation

4.1 Introduction

Polyadenylation(Poly(A)) occurs at the 3' end of the pre-mRNA molecule, it typically involves a cleavage of its 3' end and then the addition of a long string of adenine residues to form a poly(A) tail [27, 132, 21]. Poly(A) tails are essential structural and functional elements of eukaryotic mRNAs, significantly impacting many aspects of mRNA metabolism. For example, as the poly(A) tails are synthesized, they bind to multiple copies of poly(A) binding proteins, which helps regulate mRNA stability, localization and translation [27, 132, 21, 86]. The recognition of the poly(A) site requires the interaction of many cis-elements with core transcription factors. In mammals, the core transcription factors include poly(A) polymerase (also known as PAP) and four multisubunit protein complexes. The CPSF (cleavage and polyadenylation specificity factor) recognizes the AAUAAA hexamer, CstF (cleavage stimulatory factor) binds to U- or GU-rich downstream element (DSE), and CFIm (cleavage factor Im) binds to U-rich upstream element (USE), these factors together form the core mRNA 3' processing complex. This core complex then recruits CFIIm (cleavage factor IIm) and PAP (poly(A) polymerase) to stabilize the initial interaction and

trigger the cleavage and polyadenylation [27, 132, 21, 104].

Recent genome-wide poly(A) analyses have indicated that alternative polyadenylation (APA), a mechanism in which the same gene has multiple 3' ends due to the recognition of multiple distinct cleavage and poly(A) sites, is pervasive in eukaryotes [127, 112, 82]. In human, $\sim 70\%$ of human genes have multiple potential poly(A) sites [33]. As a result, a larger portion of RNA transcripts have different 3' untranslated regions (UTRs) or encode different proteins. APA not only expands the proteomic and functional diversity, but also plays an important role in gene regulation regarding mRNA metabolism [104, 36]. According to the different locations of poly(A) sites, Shi [104] proposed two types of alternative polyadenylation, i.e. (i) same exon APA (SE-APA), in which alternative poly(A) sites are in the same terminal exon, (ii) different exon APA (DE-APA), in which alternative poly(A) sites are located in different exons, as shown in Figure 4.1.

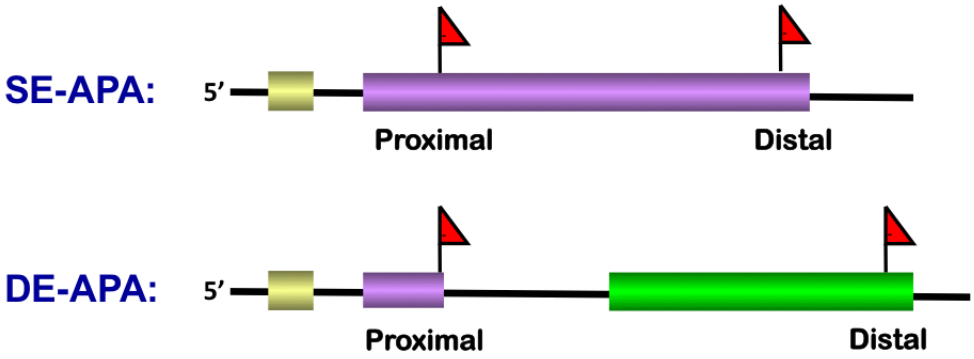


Figure 4.1: Two types of alternative polyadenylation

Many studies have revealed that polyadenylation and APA regulation defects would cause or contribute to the development of a wide spectrum of human disease [41, 82, 55, 87, 32], highlighting the importance of this process. The SE-APA isoforms differ in their 3' untranslated regions (3' UTRs). 3' UTR contains many regulatory elements for post-transcriptional and translational control, largely influences transcript transport, localization, stability, and translation. In general, longer 3' UTR has greater potential for containing regulatory elements, like microRNA target elements, results in lower translation activity [12, 82]. A well-characterized example would be BDNF (brain-derived neurotrophic factor) gene, which plays an important role in the development of the brain.

Two mRNA isoforms of BDNF only differ in 3' UTR. The short 3' UTR mRNA is restricted to cell soma, whereas the longer 3' UTR mRNA is preferentially localized in dendrites. In a mouse mutant that the long 3' UTR was truncated, the dendrites of hippocampal neurons showed impaired long potentiation, and these mutants got obesity due to impaired response to leptin [8, 76]. On the other hand, the DE-APA isoforms use alternative poly(A) sites located in distinct exons, and thus encode distinct proteins. For example, the IgM gene encodes two kind of proteins, the secreted or the membrane-bound forms of IgM. In the activated B cells, the secreted IgM isoform is highly expressed by using the proximal poly(A) site; while in the resting B cell, the membrane-bound IgM corresponding to the distal poly(A) site is preferentially expressed [98]. Furthermore, recent studies discovered the core 3' polyadenylation factors and other APA regulators have different expression level in different tissues or at different development stages [58]. APA could be broadly regulated under different cell conditions, many APA events display tissue-specific alternative 3' UTR patterns [118, 1]. For example, a general shift to shorter APA isoform is observed during reprogramming of normal cells into stem cells [67]. It has been reported that proliferating cells tend to use more proximal sites [100], while mammalian neurons tend to have longer 3' UTRs by using more distal poly(A) sites [103]. Therefore, it is important to understand the underlying polyadenylation mechanism for tissue-specific APA regulation. Although several computational approaches have achieved considerable success in predicting poly(A) sites with canonical features, unfortunately, computational framework studying tissue-specific APA regulation is rarely investigated, thus tissue-specific APA regulation remains poorly understood.

Due to recent technological breakthroughs, high-throughput sequencing technologies, such as PAS-seq and direct RNA sequencing, allow genome-wide mapping of poly(A) sites and detection of APA events in high resolution. A comprehensive analysis of APA in different cell conditions would greatly broaden our understand of tissue-specific APA regulation.

4.2 Related Work

In most early computational studies for poly(A) sites prediction in human or other species, researchers focused on known cis-elements around cleavage sites, like canonical poly(A) signal AAUAAA, U/GU-rich downstream sequence elements and U-rich upstream sequence elements. Cheng et al. [25] built recognition model on support vector machine (SVM). Position weight matrices (PWM) for 15 candidate cis-elements surrounding cleavage sites were used as input features [54]. Studies of poly(A) signal (PAS) motifs upstream of cleavage sites revealed 12 variants of this hexamer[13, 112], i.e. AAUAAA, AUUAAA, AGUAAA, UAUAAA, CAUAAA, GAUAAA, AAUAUA, AAUACA, AAUAGA, ACUAAA, AAGAAA and AAUGAA. A method called POLYAR divided human poly(A) sites into three groups (PAS-strong group for poly(A) sites with A(A/U)UAAA, PAS-weak group for poly(A) sites with the other 10 PAS motifs,and PAS-less group for poly(A) sites don't have any of 12 motifs), and adopted a linear discriminant model for prediction [5]. Dragon Poly(A) spotter [63] extracted additional features from thermodynamic, physico-chemical properties of bi-nucleotides, and tested them on artificial neural network and random forest models. Chang et al. also used SVM to predict human poly(A) sites. They incorporated secondary structure information, and demonstrated its association with polyadenylation via different RNA secondary structure search programs [23]. Ji et al. [56] built a generalized hidden Markov model (GHMM) to predict poly(A) sites in plants. More recently, Xie et al. [126] proposed to integrate hidden Markov model (HMM) and SVM for the prediction of poly(A) motifs.

One big problem of theses models is that they were trained on expressed sequence tags (EST) data. EST sequence is generated from a single sequencing run without verification, their quality is usually not comparable to the quality of high-throughput sequencing platform, like PAS-seq. Recently, Hafez et al. [47] tried to infer the regulatory mechanisms of polyadenylation in multiple cell types. They collected poly(A) sites in liver, kidney and brain tissues using paired-end sequencing, and developed predictive model using SVM with weighted degree string kernel with shifts (also known as WD kernel) to explore the differential usage between constitutive and tissue-

specific poly(A) sites. However, their model showed moderate classification performance between tissue-specific and constitutive poly(A) sites. Another problem in their model is that they only considered genes with single poly(A) site as constitutive sites. In reality, this is only a small portion of constitutive poly(A) sites, since only about 30% of genes in human have single poly(A) site. Another non-negligible portion of constitutive poly(A) sites are from genes with multiple poly(A) sites. It is generally believed that tissue-specific APA regulation mechanism in genes with single 3' UTR and genes with multiple 3' UTRs are different. Genes with only one poly(A) site change their mRNA abundance levels to show tissue-specific manner, while genes with multiple poly(A) sites mostly change 3' UTR isoform ratios to achieve tissue specificity [75]. In our study, we only considered genes that have multiple poly(A) sites and were expressed in multiple tissues. We define constitutive poly(A) sites are from genes with multiple poly(A) sites and with similar 3' UTR usage profiles across tissues.

4.3 Methods

4.3.1 Identify tissue-specific PAS

To infer a poly(A) code that distinguishes tissue-specific and constitutive mRNA alternative polyadenylation, the very first step is to obtain tissue-specific and constitutive poly(A) sites. The truly tissue-specific mRNA transcripts that express only in a single tissue are rare, most poly(A) sites are shared across all tissues, although some expressed at low levels. The tissue-specificity is largely determined by the variation of poly(A) site usage percentages among all tissues. A good metric for characterizing the variation of poly(A) site usage is the Shannon entropy, which has been broadly used in measuring tissue specificity of gene expression data [101]. To extend Shannon entropy for estimating tissue specificity of a poly(A) site, we calculate the relative read count percentage in a

gene as the expression level of each poly(A) site.

$$x_{g,p}^{t} = \frac{N_{g,p}^{t}}{\sum_{p \in g} N_{g,p}^{t}} \tag{4.1}$$

Tissue specificity of a poly(A) site measures the uniformity of usage over all tissues. We assume a poly(A) site has the same usage percentage in all tissues is tissue-nonspecific, or constitutive. We do not claim such poly(A) site is not regulated, only that it is regulated in a way that is independent on tissue conditions. A poly(A) site shows significant non-uniform pattern is more likely to exhibit tissue-dependent regulation. For a poly(A) site, we put its usage percentage of N tissues into a vector, $x = (x_1, x_2, ..., x_N)$, x_t denotes poly(A) site usage for tissue *t*. The entropy of this poly(A) site can then be calculated as

$$H = -\sum_{t=1}^{N} p_t \log_2(p_t), \tag{4.2}$$

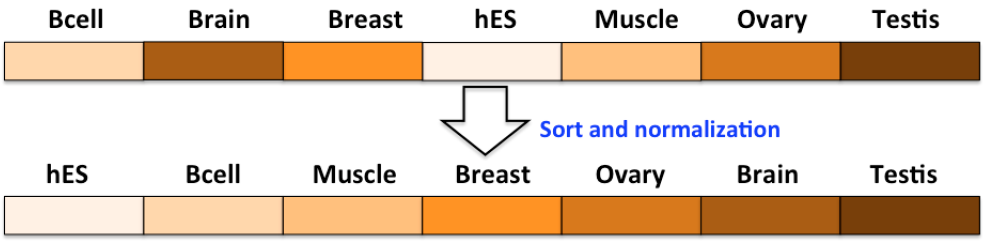
where $p_t = \frac{x_t}{\sum_{t=1}^{N} x_t}$ is the relative usage of x_t for tissue *t*. Entropy *H* measures the degree of overall tissue specificity, it ranges from zero to $\log_2(N)$, *N* is the number of tissues. Entropy values close to zero represent poly(A) sites are specific in a single tissue, and increase when the poly(A) sites are more broadly used in different tissues, or when the relative contribution from different tissues become closer. However, equation (4.2) only identifies poly(A) sites that are highly used in one or a few tissues, but fail to detect poly(A) sites that are depleted in one or a few tissues. In order to equally identify highly used, depleted, and mixed-pattern of tissue specific poly(A) sites, we introduce one-step tukey biweight T_{bw} to adjust the original usage vector, $x'_t = |x_t - T_{bw}|$. We then calculate a new entropy value for x'_t, using the same formula as for x_t. The one-step tukey biweight provides a robust weighted mean that able to resist 50% of outliers [43].

A big disadvantage of entropy measure is that it only measures the degree of overall tissue specificity of a poly(A) site, but does not indicate which tissue the poly(A) site is specific to. To identify the tissue-specific pattern, we apply an outlier detection method called ROKU proposed by Kadota

et al. [62]. ROKU method is based on Akaike's information criterion (AIC) [61], which is an information criterion used to identify an optimal model from a class of competing models by comparing the trade-off between the goodness of the model and the complexity of the model [4]. AIC based approach fits our tissue-specific pattern detection problem well, it can handle various situations, including single outlier, several lowest or highest outliers, two-sided outliers cases, and more importantly, these various cases are treated equally. It has been proved to be more robust than another popular Sprent's non-parametric method [60]. An illustration example of detecting tissue specific pattern is shown in Figure 4.2. We first normalize each poly(A) site vector $x = (x_1, x_2, ..., x_N)$ to zero mean and unit variance, and rank *N* tissues by the normalized values in the order of increasing magnitude. For each poly(A) site, we consider three types of tissue-specific patterns: 'up-type' that poly(A) site is highly used in a single or small number of tissues compared to the others, 'down-type' that poly(A) site is depleted in a single or a few tissues, and 'mix-type' that poly(A) site is highly used and depleted in some tissues. The total number of candidate combination is $(K+1)(K+2)/2 = 10$, *K* is the maximum number of outlier tissues. In our study, we set $K = 3$, so we consider 10 possible outlier tissue-pattern combinations. For each combination, we compute a statistic U [61],

$$U = \frac{1}{2}AIC = n \log \delta + \sqrt{2} \times s \times \frac{\log n!}{n} \tag{4.3}$$

where $N = n + s$, *s* and *n* denotes the number of outlier and non-outlier tissues, and δ denotes the standard deviation (SD) of scores assigned to *n* non-outlier tissues. The first term in the equation measures the variance of non-outliers, and the second term indicates model complexity increase as number of outlier candidates increase. An ideal tissue-specific pattern should have small non-outlier variance and the outlier candidates are kept in a small size. The combination with lowest U is assigned as the best tissue-specific pattern.



		outliers (high usage side)			
		none	Testis	Testis +Brain	Testis +Brain +Ovary
outliers (low usage side)	none	-24.4	**-24.7***	-23.9	- 22.4
	hES	-19.1	-20.4	-19.0	
	hES+Bcell	-15.6	-16.2		
	hES+Bcell +Muscle	-12.3			

Figure 4.2: Schema of tissue-specific pattern identification

4.3.2 Compedium of Putative Regulatory Features

A number of studies have demonstrated that information from relative short upstream and downstream sequence of the poly(A) sites can specify the true poly(A) signals to a great extent [3, 25, 5, 63]. More specifically, most of the main features of polyadenylation are located in the region from 100nt upstream to 100nt downstream [25, 47]. For most sequence features, we spanned a flanking regions of 100nt upstream and downstream of each poly(A) site, each sequence data is composed of 201 nucleotide bases, with the polyadenylation site right in the middle. The features in this 201nt region are split into four subregions approximately according to several previous studies of poly(A) signal characteristics in subregions [23, 112, 54], as shown in Figure 4.3.

We assembled a compendium of 658 RNA features, covering all major parameters known or with great potential to influence poly(A) regulation, including motifs of known polyadenylation regulator, unknown but potential motifs, evolutionary conservation level, secondary structure information, nucleosome positioning, features describing transcript structures, the complete list is available

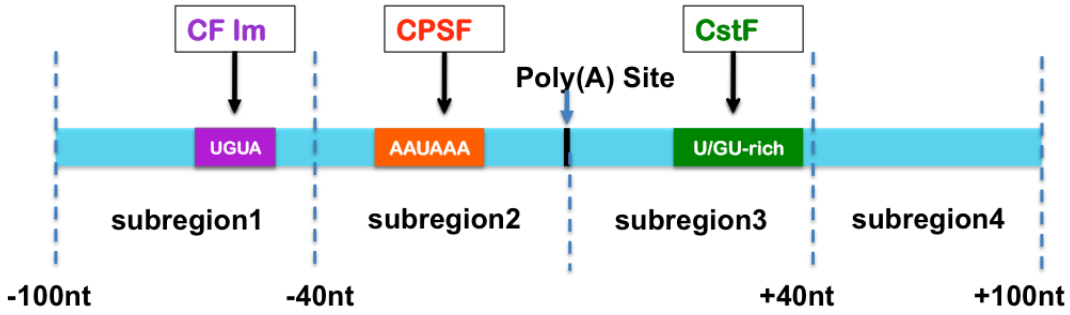


Figure 4.3: Schematic of four subregions of human poly(A) sites

in the supplementary material.

Motifs for known alternative polyadenylation regulators

Cleavage and polyadenylation mostly relies on the interaction of cis-elements with core multi-subunit protein complexes. One key cis-element that dictates cleavage is the consensus AAUAAA hexamer, is located about 10-30nt upstream of the cleavage site [63]. This hexamer is called polyadenylation signal (PAS), directly bind to CPSF30 and Wdr33 [22], two important subunits of CPSF complex. The canonical form AAUAAA of PAS is present in more than 50% human polyadenylation sites. There are a large portion of genes have one of its eleven single nucleotide variants (AUUAAA is the most common variant), they together are associated with $\sim 92\%$ human poly(A) sites [112].

In addition to PAS, a highly variable U- or GUrich downstream sequence element (DSE) is found within 30 nt downstream of the cleavage site in many mammalian PAS, and is recognized by the CstF64 or CstF64τ of the CstF complex. A number of auxiliary sequences have also been identified in some poly(A) sequences. For examples, some U-rich upstream sequence elements (USE), located upstream of the AAUAAA hexamer, are recognized by the Fip1 subunit of the CPSF complex. UGUA sequences, also typically found upstream of AAUAAA, are recognized by the CFIm complex. The cooperative binding of these core protein complex mostly determine

cleavage efficiency, or the intrinsic "strength" [116, 90].

Additionally poly(A) site recognition is modulated by other RNA binding proteins or regulatory factors. One such example is polyadenylation binding protein nuclear 1 (PABPN1), reduced function on PABPN1 results in a broad enhancement of polyadenylation at proximal sites [55]. Important polyadenylation factors we investigated includes CFI, PTB1, Nova1/2, hnRNPF/H, PCBP1/2, ESRP2, PABPN1, SFRS1, CstF64 and CstF64τ.

Evolutionary conservation level

Previous study showed the conservation level of poly(A) site was correlated to the cleavage efficiency. In general, proximal poly(A) sites are less conserved than distal poly(A) sites [9, 1]. Also, the conservation patterns in the upstream and downstream of cleavage site are very different. We include two type of conservation values, one is phastCons score based on a multiple alignment of 99 vertebrate genomes to the human genome [105], the other is phyloP (phylogenetic p-values) from the PHAST package (http://compgen.bscb.cornell.edu/phast/) for multiple alignments of 99 vertebrate genomes to the human genome [95]. Two conservation scores describe nucleotide conservation level from two aspects, phastCons score is calculated per position and represents probability of a nucleotide position to be part of a conserved sequence element; while phyloP score reflects individual alignment match, which only depends on a model of neutral evolution and does not take into account conservation at neighboring sites. We extract position-wise conservation in [-100,100]nt region from UCSC genome Browser and calculated the average conservation score in each subregion.

Local secondary structure

In addition to the RNA sequence pattern, RNA secondary structure plays a role in mRNA polyadenylation [123, 23]. We count the nucleotide frequency around the poly(A) sites on DRS data from

human HeLa cell line [128], and find the nucleotide frequency show an A-rich region from -25nt to -12nt and a U-rich region from -15 nt to -2 nt in the upstream, while the region from +1 to +5 nt downstream is A-rich, and the region from +5nt to +35nt is U-rich, as shown in Figure 4.4. Such symmetric structure implies potential secondary structures around poly(A) sites. A recent in vivo genome-wide profiling of RNA secondary structure confirmed this pattern [34].

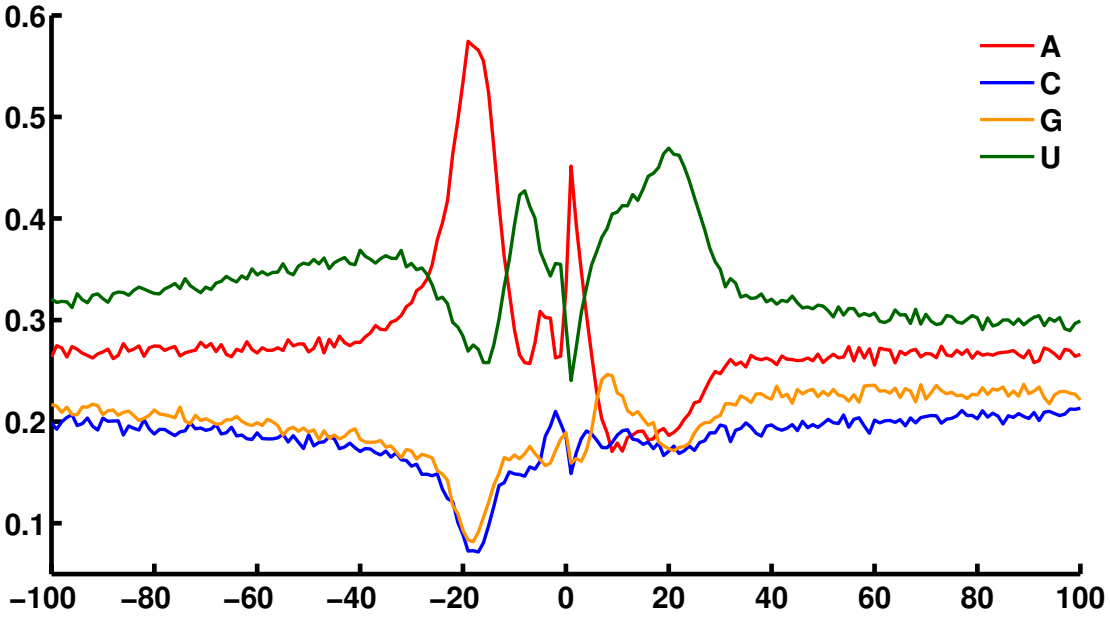


Figure 4.4: Nucleotide composition around poly(A) sites

RNA binding proteins recognize RNA targets in a sequence specific manner. The target binding sites are often located in single-stranded RNA region, therefore, RNA secondary structure information near the cleavage sites is likely to play an important role in defining the binding sites, and influence APA regulation. We use pentamer accessibility to characterize the single strandedness of a substring in a given RNA sequence, i.e. the expected fraction of bases in the pentamer that do not form base pairs (pentamer is a 5-mer substring of RNA sequence between position a and b):

$$EF_{a,b} = 1 - \frac{\sum_{i=a}^{b} \sum_{j=1}^{L} p_{i,j}}{b - a + 1} \tag{4.4}$$

where L is the length of the RNA sequence, and $P_{i,j}$ is the probability that base i and j are paired. The base pair probability $p_{i,j}$ is computed by combing all possible secondary structures generated by RNAfold [79]. We calculate all pentamers' accessibilities, the features are extracted by computing the average and maximum pentamer accessibility in each subregion.

Nucleosome positioning

The relation between nucleosome positioning and polyadenylation is rarely investigated, however, it has gained much research attention recently. Spies et al. [109] observed a strong nucleosome depletion around human poly(A) site and a nucleosome enrichment next to the poly(A) site in the downstream, suggesting the connection between nucleosome positioning and polyadenylation regulation. We therefore include features about nucleosome occupancy around the poly(A) sites. We use NuPoP (Nucleosome Positioning Prediction Engine) package [124] to compute nucleosome occupancy score based on sequence information in the region of [-300,300]nt. Features are defined as the average and maximum occupancy scores in the upstream 100nt and downstream 100nt of each poly(A) site.

Potential motifs

To discover potential new motifs that are likely to influence polyadenylation regulation, we perform a search of enriched motifs on a complete different data set, direct RNA-sequencing (DRS) dataset from human HeLa cell line [128]. We filter out low-quality poly(A) sites and used sequence in [-100, 100]nt region around poly(A) sites as positive data set. To construct a proper negative data set for background, instead of use random RNA sequence, for each poly(A) site, we randomly select 10 positions in the 3' UTR region between the transcript stop codon and the poly(A) site, but don't include the last 100 nucleotides. We count all 5-mer motifs in both upstream and downstream, and use simple linear models with the *L1* regularization term to do feature selection.

We select top 141 motifs that have greatest contributions in separating positive poly(A) sites from negative poly(A) sites. For each motif, in addition to the number of occurrence, we compute a conservation-weighted motif scores by incorporating the average phastCons conservation score at each occurrence.

Transcript sequence structure

Alternative polyadenylation (APA) is a widespread phenomenon, as mentioned earlier, about 70% of mammal genes have multiple 3' ends due to the presence of distinct poly(A) sites [33]. The underlying polyadenyalation mechanisms of different types of APA (SE-APA, DE-APA) are likely to be very different. Take the regulation rules control choice of one poly(A) site over another for example, SE-APA and DE-APA probably have different regulators that modulate the selection of their own optimal poly(A) sites: the selection of poly(A) sites in type SE-APA may be highly affected by the distance between alternative poly(A) sites, while the regulation in type DE-APA may be highly associated with alternative splicing regulation. We include features describing polyadenylation sequence structure into our model, including APA type, the distance between alternative poly(A) sites, the distance to closest upstream splice site and its splicing strength, distance to the stop codon as well as poly(A) site relative location in the gene (proximal, distal or intermediate poly(A) site), and we refer them to as "transcript sequence structure". We use Ensembl annotation of hg19 from UCSC track to extract those features.

4.3.3 Prediction Models

To infer a poly(A) code that can distinguish tissue-specific and constitutive mRNA alternative polyadenylation, we decide to formulation this poy(A) code question as a binary classification problem. Based on the tissue-specific and constitutive data we have collected, and 658 features we have assembled, we first construct a robust and predictive model to discriminate tissue-specific

poly(A) sites from constitutive poly(A) sites, and then identify features that make significant contributions to tissue-specific APA regulation, and finally infer a poly(A) code. In this study, we propose to use an adaptive ensemble learning algorithm to build a sequence of weak base estimator, which are then weightedly combined to a strong classifier so as to achieve a better performance. Boosting is a general ensemble method used to improve the accuracy of any given learning algorithm. A weak classifier with accuracy greater than random guess is created, and then new component classifiers are added to form an ensemble whose joint decision rule has greater accuracy in the training set. In such a case, we say the classification performance has been boosted. AdaBoost is a boosting algorithm, introduced in 1997 by Freund and Schapire [40], has successfully solved many classification problems. The main idea of AdaBoost is to fit a sequence of weak learners, such as small decision trees, on weighted training data. The predictions from all of weak learners are combined through a weighted majority vote, weak classifiers with lower classification errors usually have higher weights, to produce the final prediction.

In detail, AdaBoost is an iterative algorithm that combines many weak classifiers in a series of boosting iterations $t = 1, 2, ..., T$ to approximate the Bayes classifier $C^*(X)$. The algorithm maintains a set of weights over the training samples. Initially, weights of training samples are all set to $w_i = 1/N$, the first step is to simply train a decision tree classifier or other base classifier on the original data. For each iteration, the sample weights are individually modified, and the reweighted data will be reapplied to the base classifier. At a given step, training examples that were incorrectly predicted at the previous step have their weights increased, whereas the weights of samples are decreased for they were predicted correctly. Each base classifier is forced to concentrate on the examples that were mispredicted at the previous step. Typically, one may build hundreds or thousands classifiers this way, a score is assigned to each classifier. The final classifier is defined as the linear combination of classifiers at each step. The general procedure of AdaBoost algorithm can be described in Figure 4.5.

Algorithm 1 Adaboost Classifier learning procedure

Input: $(x_1, y_1), ..., (x_N, y_N)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

1. Start with $w_i = 1/N$ for $i = 1, ..., N$
2. Repeat for $t = 1, ..., T$:
   a) Train base classifier $F^{(t)}(X)$ (decision tree in our study) using weights w_i^t.
   b) Compute

$$err^{(t)} = \sum_{i=1}^{N} w_i^t \mathbb{I}(y_i \neq F^{(t)}(X_i)) / \sum_{i=1}^{N} w_i^t \tag{4.5}$$

   c) Compute

$$\alpha^{(t)} = \log \frac{1 - err^{(t)}}{err^{(t)}} \tag{4.6}$$

   d) Set

$$w_i^{t+1} = w_i^t \cdot \exp(\alpha^{(t)} \cdot \mathbb{I}(y_i \neq F^{(t)}(x_i))), \text{for } i = 1, 2, ..., N \tag{4.7}$$

   e) Renormalize w_i^{t+1} such that $\sum_i w_i^{t+1} = 1$
3. Output the final classifier

$$C(X) = \underset{k \in Y}{\operatorname{argmin}} \sum_{t=1}^{T} \alpha^{(t)} \cdot \mathbb{I}(F^{(t)}(X) = k) \tag{4.8}$$

Figure 4.5: Adaboost Classifier learning procedure

4.4 Result

4.4.1 Dataset for classification

The polyadenylation data used in this study covers seven human tissues, including naive B cells, brain, breast, embryonic stem (ES) cells, ovary, skeletal muscle and testis, is from a previous study [75]. We used the "cleaned alignment" version of their 3' seq data, in which only unique mappable reads were kept, and internal priming or spurious anti-sense reads were removed. To make sure read counts from different tissues are comparable, the expression of each peak was normalized by sequencing depth (counts per million) in each tissue. We mapped all poly(A) sites into known gene regions of the genome. In our study, we only considered genes that have multiple poly(A) sites, and are expressed in multiple tissues.

In order to control the number of false positive poly(A) sites, we removed poly(A) sites that either normalized read count in all tissues are less than 5, or the total read counts on gene level are less than 10. We also required the contribution percentage of each poly(A) sites on the gene level should be more than 5% in at least one tissue. We then computed tissue specificity, both original and adjusted entropy scores, on poly(A) sites passed all thresholds (see method more detail information). We set very restrict threshold to collect tissue-specific poly(A) sites (adjusted $H' < 1.8$ and $H < 2.2$) or constitutive poly(A) sites (adjusted $H' > 2.2$ and $H > 2.7$). Our restrict threshold led to 2276 tissue-specific poly(A) sites and 3903 constitutive poly(A) sites, which were fed into our classification model.

4.4.2 Classification performance

We collected 2276 samples in positive/tissue-specific data set, to keep a well-balanced data sets for classification, we randomly sampled 2276 negative samples from negative/constitutive data pool.

It is very important to find the optimal parameter sets that can accurately separate tissue-specific samples to constitutive samples, and keep good generalization performance at the same time. We therefore held out 552 samples for model testing, and used the reaming sample with 10-fold cross-validation to learn the optimal hyper parameters for each prediction model. We used both accuracy, and the area under the ROC curve (AUC) to estimate our classification performance. Accuracy is the proportion of true results, including both true positive and true negatives. AUC is defined by true positive rate (TPR) and false positive rate (FPR), which depicts relative trade-offs between true positive rate (sensitivity) and false positive rate (1-specificity).

We applied classification task on AdaBoost classifier and another two baseline models: logistic regression and linearSVM with regularization. We also compared to SVM model with WD-kernel proposed in a previous study on modeling tissue-specific alternative polyadenylation [47]. The SVM model extract sequences around poly(A) sites, and used WD-kernel to compute similarities between two sequences while taking positional information into account. In other words, they used sequence motifs with local shift as input for their SVM model. As shown in 4.6, the SVM model with WD-kernel has very moderate performance, the area under the ROC curve is only 0.62, while in our two linear models, whose input is our assembled RNA features, the area under the ROC curve are both greater than .80, outperform the SVM model with WD-kernel with a significant margin. As demonstrated in their own paper [47], SVM model with WD-kernel achieved excellent performance when comparing constitutive poly(A) sites with non-poly(A) sites background, but showed very moderate performance on the classification between tissue-specific and constitutive alternative poly(A) sites. This suggests using motif information with local shifts (such as input in SVM model with WD-kernel) might be efficient to identify real poly(A) sites from non-poly(A) background, but is definitely not sufficient to explain the difference between tissue-specific and constitutive alternative polyadenylation. Our AdaBoost classifier achieved test accuracy as high as 84.5%, the area under the ROC curve is 0.92. Our AdaBoost classification model significantly outperforms two linear models. This indicates features we assembled are highly informative for explain the differentiation between tissue-specific and constitutive poly(A) sites, and the relationship

between tissue-specific label and features are nonlinear.

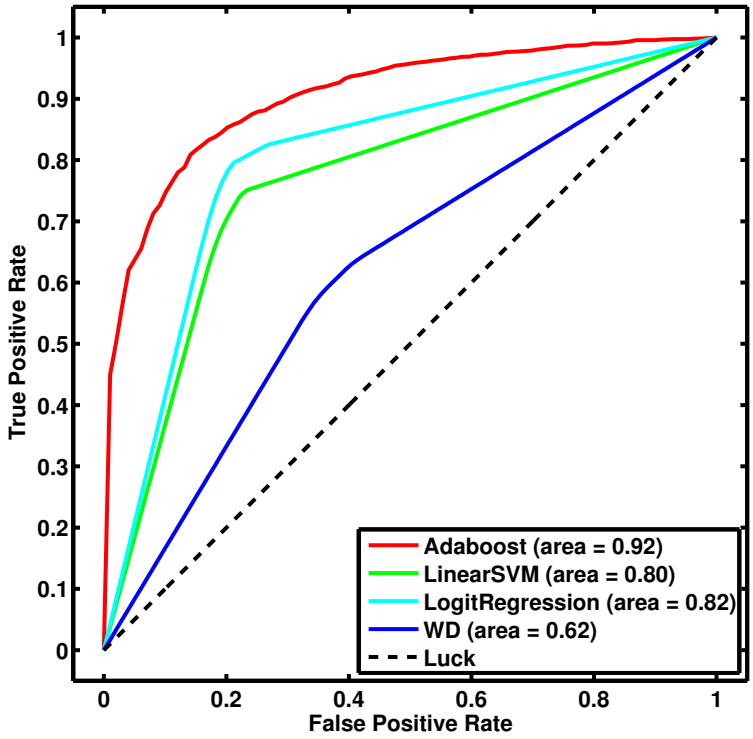


Figure 4.6: ROC curves of different classifiers

4.4.3 Robust polyadenylation code

We have demonstrated our model can predict tissue-specific poly(A) sites with high accuracy. We want to further answer two questions: 1) How important is each feature set with respect to accurately label test data set, 2) How does features, especially important features, influence or control APA selection and regulation. To answer the first question, we built classifiers on different feature set combinations, and compared their prediction capability, Figure 4.7.

We first investigate the prediction performance using known motif features only (including PAS hexamers and motifs of known polyadenylation regulators), which were used in most previous studies of poly(A) sites identification. This basic model turned out to have a very moderate ac-

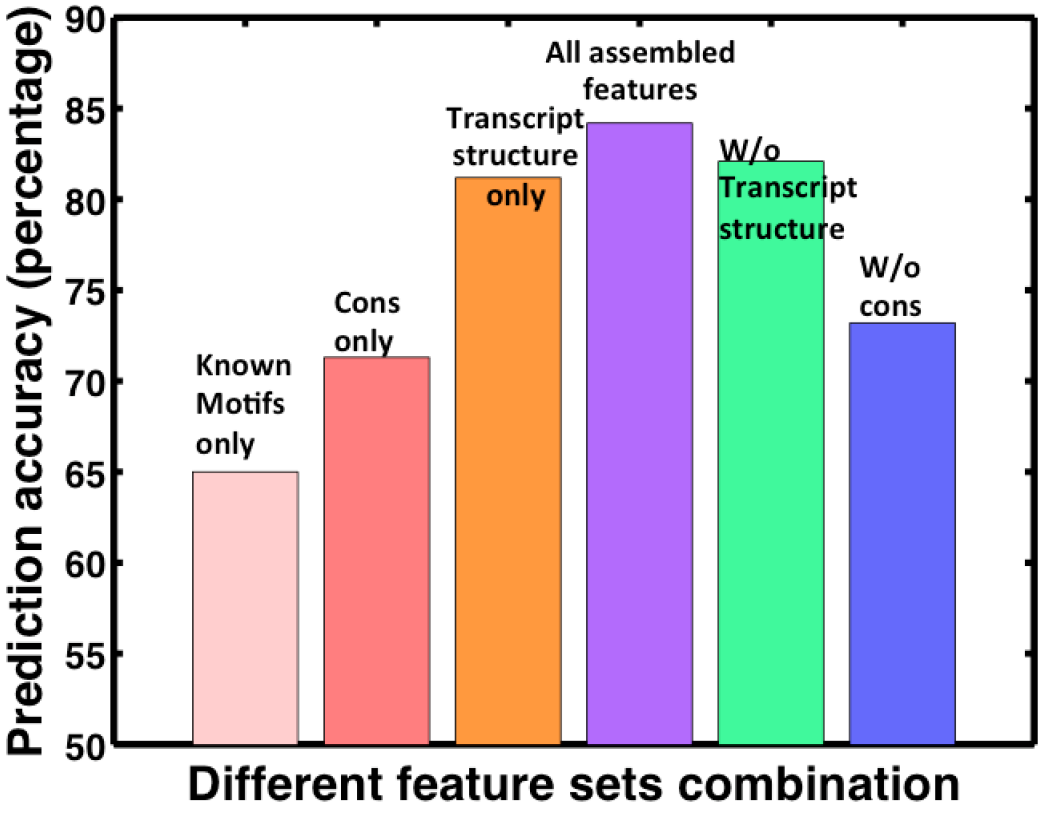


Figure 4.7: Prediction accuracy bar plot using different feature sets

curacy $\tilde{6}5.0\%$, implies known motif feature set is not enough to separate tissue-specific against constitutive poly(A) sites. When adding short 3-mer features or secondary structure features, the increase of prediction accuracy in both cases are less than 1.0%. Adding unknown but potential motifs we extracted increases the accuracy by 4.7%, while adding conservation features leads to a relative bigger accuracy improvement, 7.3%. Including transcript structure features to known motif feature set, surprisingly, the model achieves an accuracy more than 80%. Interestingly, AdaBoost classifier model, using transcript structure feature set as the only input can successfully separate about 81% of our samples, while using conservation level feature set itself can achieve 71.3% accuracy, both surpass the performance of known motif feature set. Classification with all assembled features gave the best performance (test accuracy 84.5%, AUC 92.0%). Detailed test accuracy using different feature sets are shown in Figure 4.8.

<table>
<tr><td rowspan="5">Known motif features only 65.0%</td><td>+ Short 3mers features</td><td>65.8%</td></tr>
<tr><td>+ Unknown motifs features</td><td>69.7%</td></tr>
<tr><td>+ Conservation level features</td><td>72.3%</td></tr>
<tr><td>+ Secondary structure features</td><td>65.2%</td></tr>
<tr><td>+ Transcript structure features</td><td>81.9%</td></tr>
<tr><td colspan="2">Transcript structure features only</td><td>81.2%</td></tr>
<tr><td colspan="2">Conservation level features only</td><td>71.3%</td></tr>
<tr><td rowspan="2">All assembled feature sets 84.5%</td><td>- Transcript structure features</td><td>73.2%</td></tr>
<tr><td>- Conservation level features</td><td>82.1%</td></tr>
</table>

Figure 4.8: Prediction accuracy table using different feature sets

According to the relative ability to accurately predict tissue-specific poly(A) sites, we concluded transcript structure features and conservation levels as the two most important feature sets. To test if the feature sets we collected are robust and could be applied to other data from different experiment conditions, we tried a new data set (we refer it to Xpad data). The new data set has APA

profile on five normal tissues from human breast, colon, kidney, liver and lung [78], and we derived the same conclusion when we applied our AdaBoost classifier to this new data set. In detail, the prediction accuracy using known motifs features only is 69.9%, adding transcript structure feature set leads to about 10% improvement, which is closer to the best performance with all assembled features 81.0%. Removing transcript structure features would significantly affect the performance, detail numbers are shown in Figure 4.9. Therefore, two data sets show consistent conclusion that the transcript structure feature set is very critical discriminating tissue-specific poly(A) sites from constitutive poly(A) sites, and the conservation level feature set is the second important feature set in both data sets.

Known motif features only **69.9%**	+ Short 3mers features	71.0%
	+ Unknown motifs features	74.2%
	+ Conservation level features	75.5%
	+ Secondary structure features	70.1%
	+ Transcript structure features	**79.4%**
Transcript structure features only		**79.3%**
Conservation level features only		**75.3%**
All assembled feature sets **81.0%**	- Transcript structure features	76.4%
	- Conservation level features	79.5%

Figure 4.9: Prediction accuracy using different feature sets on Xpad data set

4.4.4 In-depth analysis of transcript structure features

We have identified important feature set that distinguishes tissue-specific poly(A) sites from constitutive poly(A) sites. In order to infer detail poly(A) code and suggest possible mechanism for tissue-specific APA regulation, we conducted an in-depth analysis of transcript structure features.

For SE-APA, we found the distance between alternative poly(A) sites greatly determine the choice of poly(A) site. As shown in Figure 4.10, the distance between alternative poly(A) sites in tissue-specific samples are significantly longer than that in constitutive samples, no matter for tissue-specific proximal or tissue-specific distal sites. We also checked the distance between alternative poly(A) sites in target genes regulated by core poly(A) factors, as well as in non-target genes. In our previous study, we found most Fip1 regulated genes in mouse ES cells have longer distance between alternative poly(A) sites [67]. We further checked if this is the case for Fip1, and other polyadenylation factors in human. The majority of APA regulated target genes displayed a shift to downstream poly(A) sites after knocked down Fip1 or CstF64& CstF64τ, distance between alternative poly(A) sites in these target genes is significantly longer than that in not-target genes, the P-values are 0.0017 and 0.0011, respectively. For CFI68, although knocking down CFI68 causes most CFI68 regulated target genes shift to proximal site, the distance between alternative poly(A) sites are also significantly longer than distance in non-target genes, with P-value 1.02E-14.

Based on these two information, we suggest a possible APA regulation mechanism for SE-APA: alternative poly(A) sites far away from each other are more likely to be regulated, either in a tissue-specific manner, or regulated by specific regulatory factors.

On the other hand, for DE-APA, we found the distance from the intronic poly(A) site to its upstream 5' splice site, as well as the strength of the 5' splice site have great impact on tissue-specific APA regulation. The tissue-specific intronic poly(A) sites are farther from their upstream 5' splice sites, and their 5' splice sites are much stronger, Figure 4.12. We also suggest a possible hypothesis for APA regulation for intronic poly(A) sites in DE-APA: strong 5' splice site inhibits its downstream poly(A) sites, thus the intronic poly(A) site is generally skipped, with the help of external activator or promoter, this poly(A) site might be recognized and selected in some specific tissues; whereas weak 5' splice sites, the intronic poly(A) site is likely to be kept, and probably would be recognized and selected. Similarly, longer distance between an intronic poly(A) site and its upstream 5' splice site has greater potential for containing regulatory elements, thus more likely to be tissue-specific.

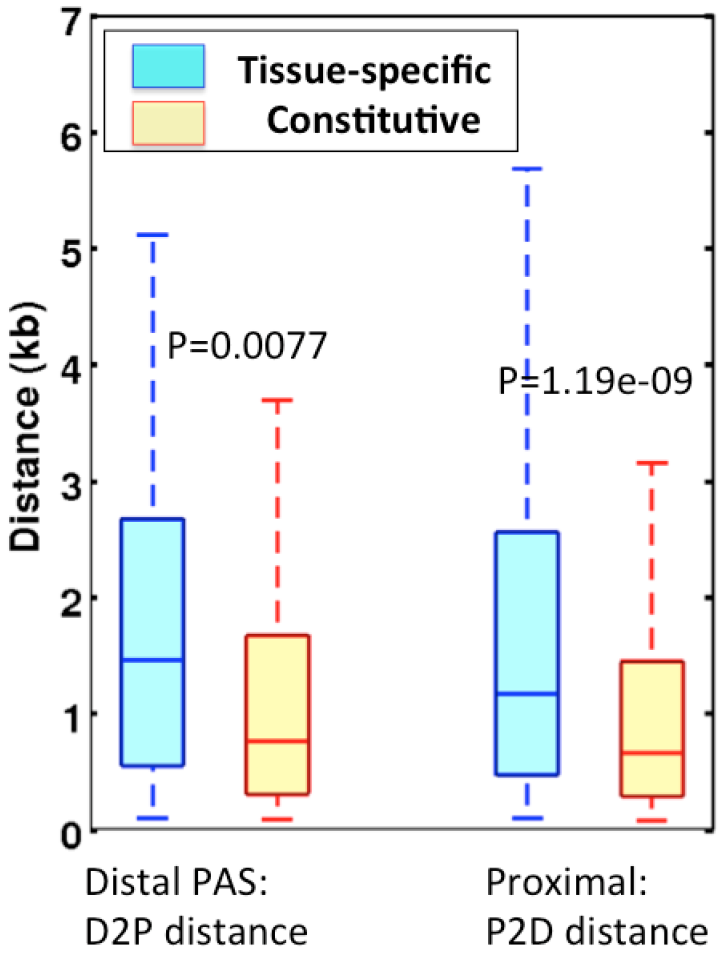


Figure 4.10: SE-APA: Distance between alternative poly(A) sites

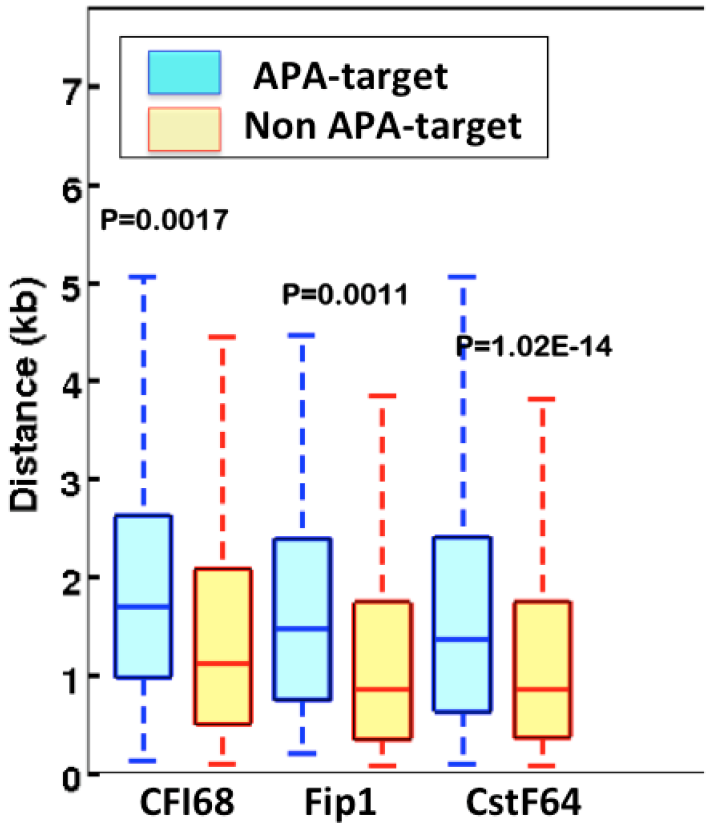


Figure 4.11: Distance between alternative poly(A) sites in APA-target genes and non-target genes

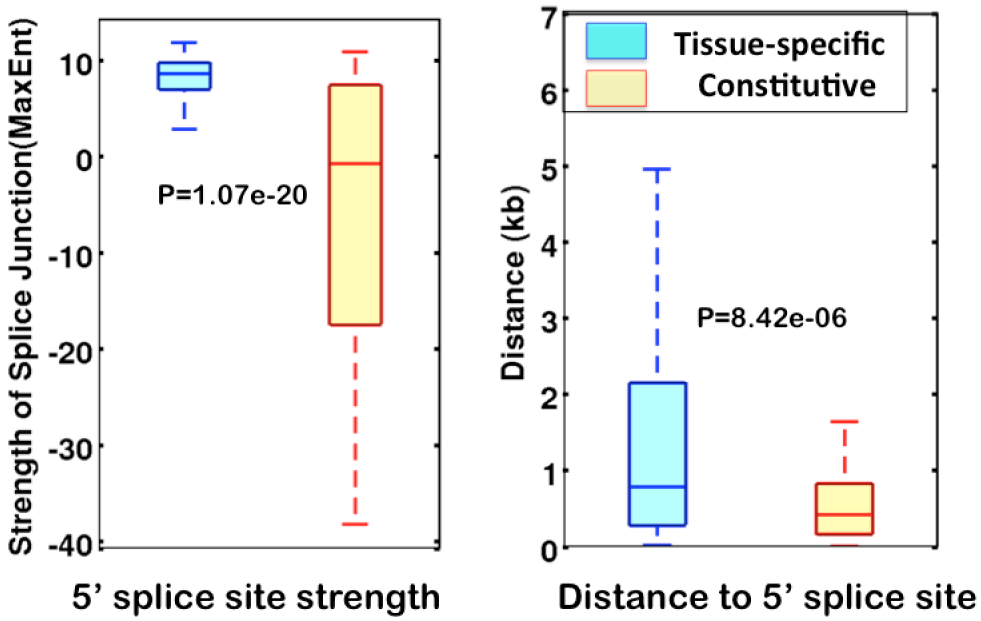


Figure 4.12: DE-APA: Distance between intronic poly(A) sites to 5' splice site and 5' splice site strength

4.4.5 Conservation level and other features

Several previous studies have reported that distal poly(A) sites are generally more conserved than proximal poly(A) sites. In our study, for the phastCon score distributions around poly(A) sites, distal poly(A) is consistently more conserved that proximal in the upstream, and the difference starts decreasing in the downstream, and after about 40nt, proximal poly(A) sites become more conserved, Figure 4.13, same pattern is found for phyloP score distribution. Interestingly, the conservation difference between tissue-specific and constitutive PASs displayed similar pattern 4.14. Is the similarity due to distal poly(A) sites are the majority in constitutive data set, or constitutive poly(A) sites themselves are more conserved? To figure out the reason behind such similarity, we extracted two types of APA genes, one has tissue-specific proximal site and constitutive distal sites, and the other has constitutive proximal site and tissue-specific distal site. We plotted the average of paired conservation level, and found tissue-specific poly(A) sites are less conserved no matter

in proximal or distal sites, Figure 4.15.

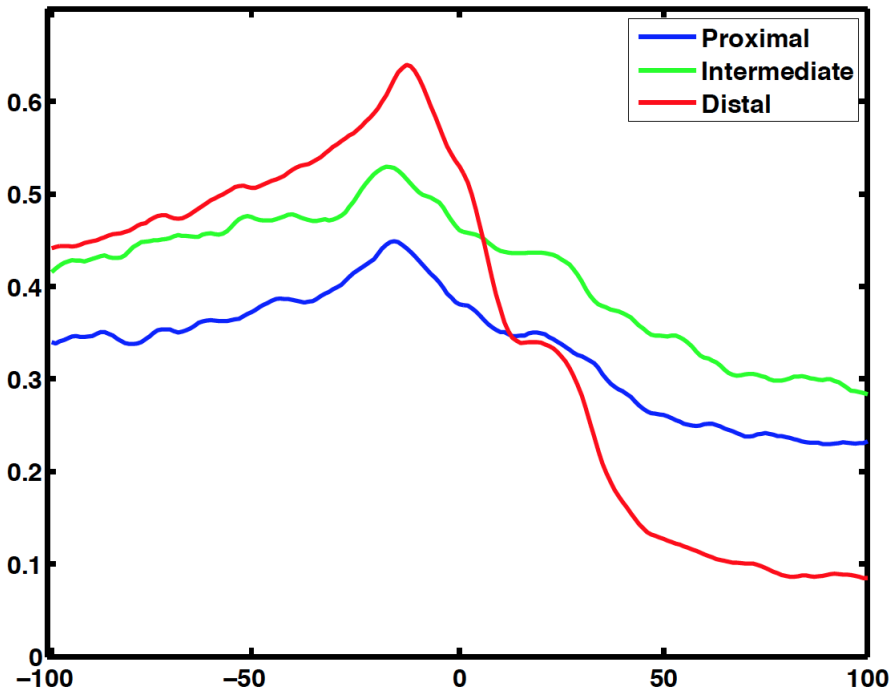


Figure 4.13: PhastCons conservation score distribution around proximal and distal poly(A) sites

We also compared the usage of AAUAAA hexamer and its variants in tissue-specific poly(A) sites to that in constitutive poly(A) sites, Figure 4.16, and found the canonical poly(A) signal hexamer AAUAAA is more often in constitutive poly(A) sites.

Based on relative position of poly(A) sites in a gene, we simply assigned poly(A) sites into three categories, proximal, intermediate and distal sites. We found the composition of those three types of poly(A) sites are very different in tissue-specific and constitutive data set. In general, the majority of constitutive poly(A) sites are distal sites, while there are only 8% distal sites in tissue-specific poly(A)s sites, Figure 4.17. For all distal sites we included, 91% of them are constitutive across seven tissues.

In summary, the poly(A) code can be partially deciphered from the rules of tissue-specific APA regulation. We established a poly(A) code that distinguishes tissue-specific and constitutive mRNA alternative polyadenylation. We assembled 658 RNA features, with many of them are novel, and

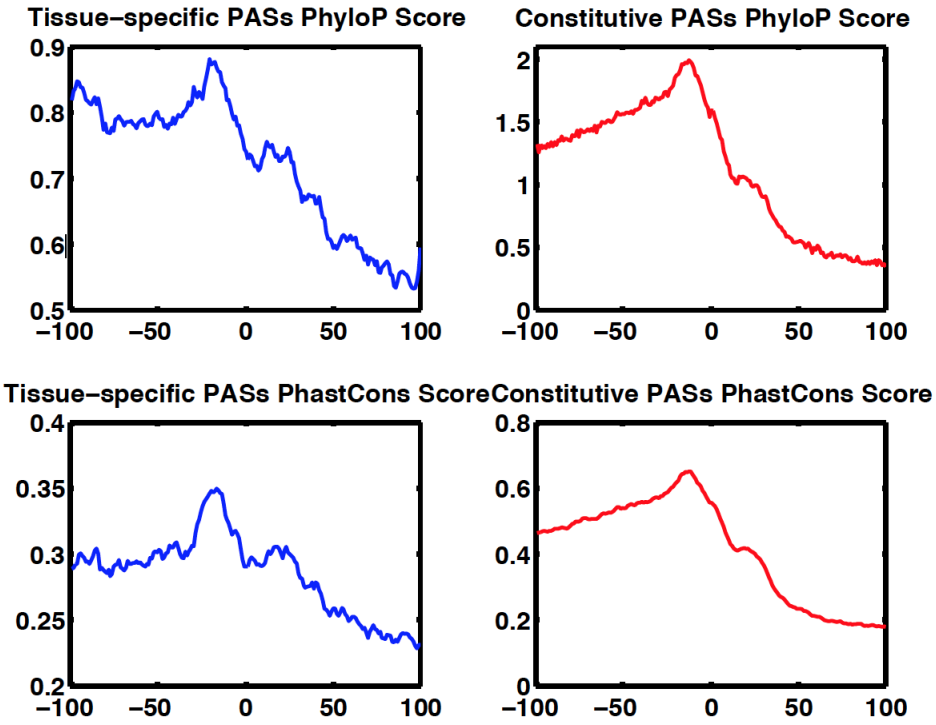


Figure 4.14: Conservation of tissue-specific and constitutive poly(A) sites

not covered in any of previous studies. The predictive model we constructed successfully discriminated tissue-specific poly(A) sites from constitutive poly(A) sites, with test accuracy 84.5% (auROC 0.92). More importantly, we identified highly informative features for tissue-specific APA regulation: distance between alternative poly(A) sites, distance to the upstream 5' splice site and 5' splice site strength. In detail, for SE-APA, the distance between alternative poly(A) sites is a key feature determines poly(A) sites to be tissue-specific or constitutive; for DE-APA, the distance from the intronic poly(A) site to its closest upstream 5' splice site as well as the strength of the 5' splice site greatly affect tissue-specific APA regulation. In addition, we found the evolutionary conservation level surrounding poly(A) sites is also very important for the APA regulation.

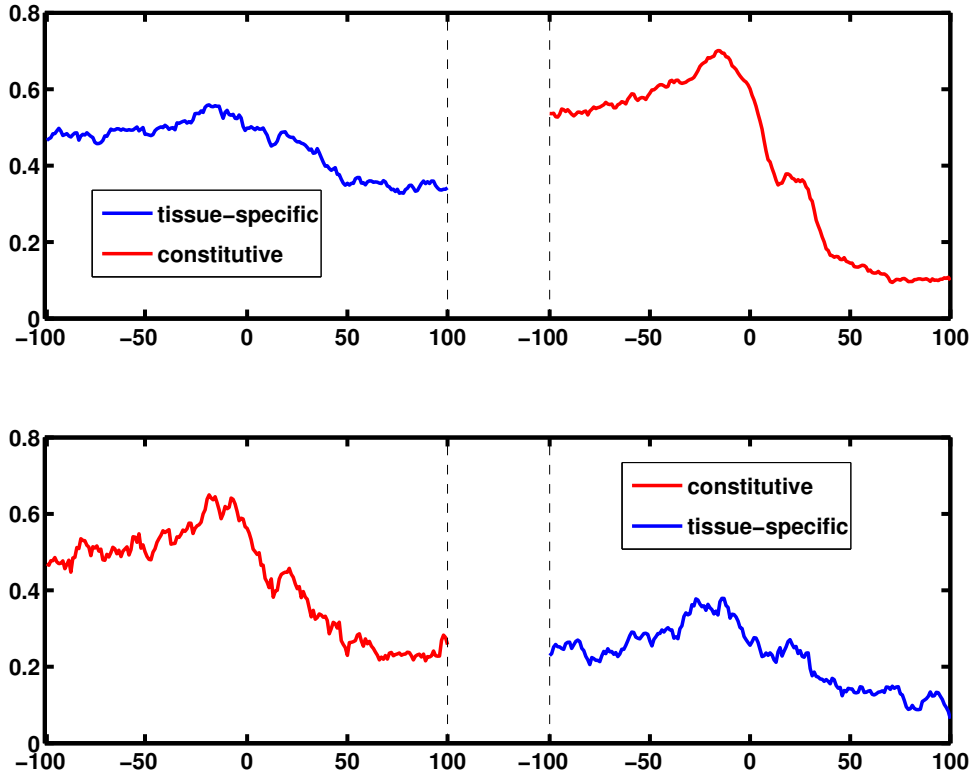


Figure 4.15: Conservation distribution in two types of proximal-distal paired APA genes

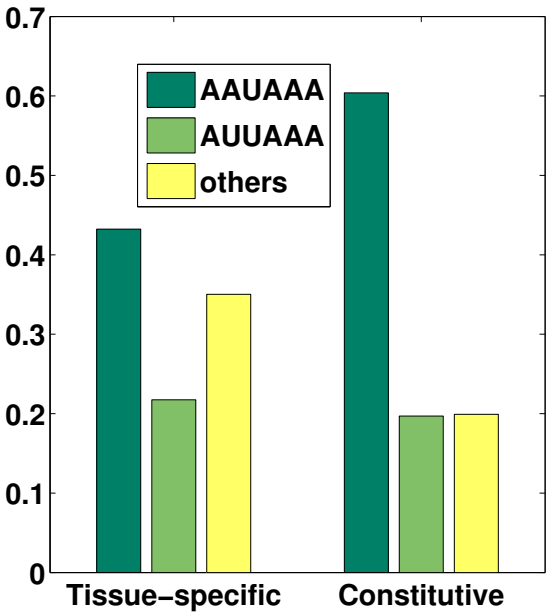


Figure 4.16: Percentage of samples using different poly(A) signal hexamers

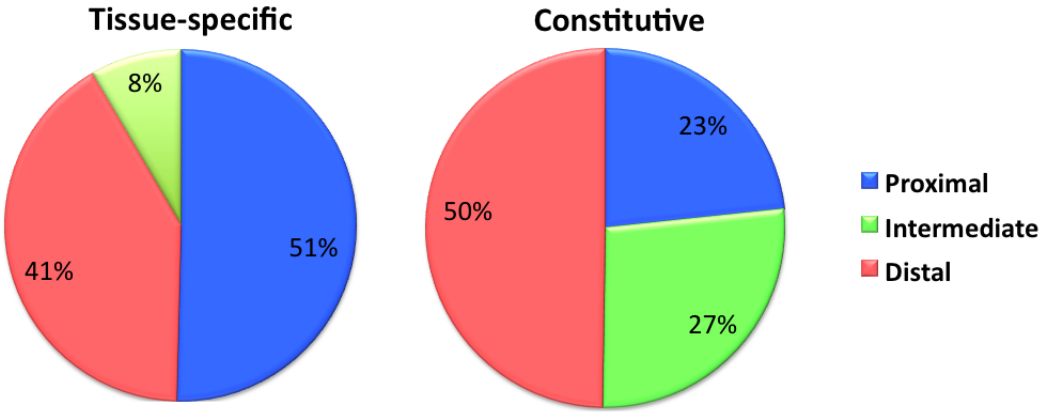


Figure 4.17: Composition percentage of proximal, intermediate, or distal among tissue-specific and constitutive poly(A) sites

4.5 Discussion

APA is an essential regulatory process that significantly impacting many aspects of mRNA metabolism. However, the rule that is responsible for recognizing and regulating APA is still poorly understood. Due to recent advancement in high-throughput sequencing, more and more quantitative methods for high-throughput sequencing of 3' ends of polyadenylated transcripts have been developed. Several of them have generated genome-wide alternative polyadenylation profiles for different tissues in different species [33, 75, 78]. However, the mechanism responsible for tissue-specific APA regulation is poorly understood. We developed an information-rich algorithm that can accurately distinguish constitutive and tissue-specific PAS in multiple recently published datasets and identified novel molecular features important for tissue-specific APA regulation.

Previously, people defined constitutive poly(A) sites as those from genes with single poly(A) site [47]. This kind of constitutive poly(A) site is always selected due to it is the only available poly(A) site in a gene. In general, there is a competition between multiple poly(A) sites in tissue-specific APA genes, improper or even non-optimal regulation on one poly(A) site would lead to another poly(A) site to be selected. Thus, the comparison between constitutive poly(A) sites from genes with single poly(A) site and tissue-specific APA genes is biased. More and more genome-wide analyses of tissue-specific APA events suggested APA genes change 3' UTR isoform ratios to achieve tissue specificity[75]. Therefore, we proposed to call poly(A) sites from genes with alternative poly(A) sites, and with similar usage percentage across tissues as constitutive poly(A) sites.

We extended existing Shannon entropy measuring to assess the tissue specificity for each poly(A) site, and applied an outlier detection method to identify the tissue-specific pattern for each tissue-specific poly(A) site. Our restrict criterion left us with a high-standard data set for the downstream investigation of APA regulation rules. Statistically analysis of our identified tissue-specific poly(A) sites showed consistent conclusion with published results. For example, the majority of constitutive

sites are from distal sites, while proximal poly(A) sites are more likely to be tissue-specific, this is consistent with known conclusion that distal poly(A) sites tend to have strong consensus features of canonical poly(A) signal, and stronger DSEs, and are generally well conserved.

We assembled 658 RNA features covering all major parameters known or with great potential to influence polyadenylation regulation. We built a successful predictive model to discriminate tissue-specific poly(A) sites against constitutive poly(A) sites. More important, we identified new important features for poly(A) code. Application of our predictive model to another data from different sequencing libraries [78], confirmed our discovery. Our methodology can be applied to another data from different sequencing libraries or different biological conditions for potential identification of new regulatory features, such as tutor tissues. Also, our poly(A) code might be used to predict the effects of diseases associated mutations near poly(A) sites.

Chapter 5

SNP-based Enrichment Analysis of GWAS

Recently we have witnessed a surge of interest in using genome-wide association studies (GWAS) to discover the genetic basis of complex diseases. Many genetic variations, mostly in the form of single nucleotide polymorphisms (SNPs), have been identified in a wide spectrum of diseases, including diabetes, cancer, and psychiatric diseases. A common theme arising from these studies is that the genetic variations discovered by GWAS can only explain a small fraction of the genetic risks associated with the complex diseases. New strategies and statistical approaches are needed to address this lack of explanation. One such approach is the pathway analysis, which considers the genetic variations underlying a biological pathway, rather than separately as in the traditional GWAS studies. A critical challenge in the pathway analysis is how to combine evidences of association over multiple SNPs within a gene and multiple genes within a pathway. Most current methods choose the most significant SNP from each gene as a representative, ignoring the joint action of multiple SNPs within a gene. This approach leads to preferential identification of genes with a greater number of SNPs.

We describe a SNP-based pathway enrichment method for GWAS studies. The method consists of the following two main steps: 1) for a given pathway, using an adaptive truncated product

statistic to identify all representative (potentially more than one) SNPs of each gene, calculating the average number of representative SNPs for the genes, then re-selecting the representative SNPs of genes in the pathway based on this number; and 2) ranking all selected SNPs by the significance of their statistical association with a trait of interest, and testing if the set of SNPs from a particular pathway is significantly enriched with high ranks using a weighted Kolmogorov-Smirnov test. We applied our method to two large genetically distinct GWAS data sets of schizophrenia, one from European-American (EA) and the other from African-American (AA). In the EA data set, we found 22 pathways with nominal P-value less than or equal to 0.001 and corresponding false discovery rate (FDR) less than 5%. In the AA data set, we found 11 pathways by controlling the same nominal P-value and FDR threshold. Interestingly, 8 of these pathways overlap with those found in the EA sample. We have implemented our method in a JAVA software package, called SNP Set Enrichment Analysis (SSEA), which contains a user-friendly interface and is freely available at http://cbcl.ics.uci.edu/SSEA.

The SNP-based pathway enrichment method described here offers a new alternative approach for analysing GWAS data. By applying it to schizophrenia GWAS studies, we show that our method is able to identify statistically significant pathways, and importantly, pathways that can be replicated in large genetically distinct samples.

5.1 Background

The power of genome-wide association studies (GWAS) to discover common genetic variants associated with complex traits has been empirically demonstrated[39, 19, 107, 131, 44, 48]. The single-SNP analysis tests genetic association on individual SNPs and identifies only the most significant SNPs because of the stringent statistical criteria necessary for minimizing false positive hits. The identified SNPs, however, represent only a small fraction of the genetic variants contributing to complex traits; the majority of the variations remain hidden within the statistical ”noise”[111, 83].

Genetic variants with small individual effect sizes but jointly significant genetic effects would be missed by single-SNP analysis. As a result, identified genetic variants only explain a small fraction of heritability for most studied traits[42].

It is increasingly recognized that pathway-based analysis, which considers cumulative association between the outcome and a group of SNPs or genes in a biological pathway, can greatly complement the single-SNP approach in understanding genetic determinants of common diseases as well as providing insight into the biological process of complex diseases[52, 49, 31, 119, 66, 94]. A pathway-based analysis by Baranzini et al[10] not only confirmed previously identified immunological pathways but also found that neural pathways might be responsible for multiple sclerosis. Joel Hirschhorn[49] pointed out that for many diseases, different risk loci are often clustered in a common pathway, so when a study highlights the role of one or a group of loci in a disease, it also provides important insights and predictive information on the role of other loci within the same biological group. He argued that the primary goal of genome-wide association studies should not be the prediction of individual risk loci but rather the discovery of biological pathways underlying polygenic diseases and traits. The genetic variants revealed in pathway-based analysis could be used to build predictive models for complex diseases, and provide insights on how multiple genetic variants jointly contribute to the etiology of complex human diseases.

One approach for pathway association analysis of GWAS is to extend the gene set enrichment analysis (GSEA) method, which has been successfully applied in gene expression data analysis [110]. However, a key difference between gene expression analysis and GWAS analysis is that each gene in GWAS is represented by many SNPs. The challenge is to determine the number as well as which SNPs are the best representatives for each gene.

Most of the current methods for pathway analysis of GWAS data are gene-based. Wang et al.[119] used the SNP with the strongest association to represent a gene. Choosing the smallest P-value for each gene might not be optimal in situations when the joint action of multiple SNPs within a gene explains more variance than the most significant SNP. For example, if a gene contains multiple

causal variants, it might not be identified by the smallest P-value method, which reduces the power of the subsequent pathway enrichment analysis. Moreover, this approach is likely to favour genes of large sizes, as genes with a larger number of SNPs have a higher chance of having significant SNPs, by chance alone. Consequently, the effects of genes with smaller numbers of SNPs would be underestimated. Holmans et al.[52] proposed ALIGATOR (Association LIst Go AnnoTatOR) method to study the significance of pathways. Although this method corrects variable gene sizes by simulations, it requires a pre-determined P-value cutoff for selecting significant SNPs and the evaluation of pathways is gene-based, not SNP-based. Yu et al.[130] used an adjusted P-value for each gene, and also treated gene as the basic unit for analysis. Since the gene-based approaches focused on testing significance at the gene-level, they may have low power to detect pathways containing only a few genes [20].

Recently, Holden et al.[50] proposed a SNP-based pathway analysis, which used all available SNPs to represent a gene. This approach is computationally intensive and might not be practical for genome-wide studies with millions of SNPs. ODushlaine et al. [91] developed a SNP ratio test (SRT) method which computed the ratio of the number of significant ($P < 0.05$) to the number of non-significant ($P >= 0.05$) SNPs for each pathway and used permutations to identify the significant pathways. When only one gene has significant SNPs, SRT method would reduce the pathway signal to gene signal. Because our method uses adaptive rank truncated product and permutations to determine the number of representative SNPs for each gene, and each gene includes at least one SNP, the pathway signal would be to some extent kept, then contributions from more genes would be emphasized in the pathway analysis. Besides, the SRT method treated all significant SNPs evenly, which might be suboptimal.

To address these limitations, we propose a new SNP-based pathway analysis method, called SNP Set Enrichment Analysis (SSEA), for GWAS studies. SSEA consists of two main steps: selecting representative SNPs for each gene, and performing pathway enrichment analysis using all selected SNPs. In the first step, we exploit an adaptive rank truncated product method with permutations

to choose the most significant subset of SNPs for each gene. The number of SNPs representing a gene is not predetermined, but data driven. Then for each pathway, we calculate the average number of representative SNPs for the genes within this pathway and re-select SNPs using this number. In the second step, we modify the existing GSEA algorithm [110] to conduct the pathway enrichment analysis using all selected SNPs. We rank all SNPs selected from the first step based on their strength of association with the trait, and then test whether the set of SNPs associated within a pathway is significantly enriched with high ranks using a weighted KolmogorovSmirnov test. Because this test is rank-based, SNPs with smaller P-values tend to contribute more in a pathway

5.2 Methods

5.2.1 Adaptive Rank Truncated Product of SNP Association

One difficulty in extending the pathway enrichment analysis of genes to SNPs is the many-to-one mapping from SNPs to genes. Generally, assigning the most significant SNP to a gene might miss other informative SNPs, while assigning too many SNPs to a gene might introduce noise and decrease statistical power. Both would introduce bias into the following pathway enrichment analysis. We select the best representative subgroup of SNPs for each gene in the following way.

For each SNP, a P-value is obtained by comparing the genotype frequencies between the cases and controls using the Pearsons chi-square test with two degrees of freedom. Extending the work of Yu et al. [130], we use an adaptive rank truncated product method. The L P-values of the L SNPs mapped to a gene are sorted from smallest to largest: $p_1 \leq p_2 \leq ... \leq p_L$, with p_l being the lth smallest P-value. We use $W_{(k)} = \prod_i^K p_i$ to combine the first K P-values, where K is the truncation point. Permuting the phenotypes and computing the statistic in permutated data allows us to assess the overall significance of the K SNPs. In the permutation procedure, we permute the phenotype values N times to obtain N permutated datasets. For the nth permutated dataset, we

denote the resulting P-values as $p_1^n, ..., p_L^n$, and calculate the corresponding $W_{(k)}^n$. Then the P-value for evaluating $W_{(k)}$ is calculated by $p(W_{(k)}) = \frac{\sum_{n=1}^{N} I(W_{(k)}^n \leq W_{(k)})}{N}$. To maximize the association of the subset of SNPs and the trait, all possible values of K are calculated and the one with the smallest P-value is chosen. The corresponding SNPs are used to represent the gene.

To avoid genes with larger number of SNPs dominating a pathway in the following SNP set enrichment analysis, and to let the contributions by more genes be emphasized in pathway analysis, we require genes in the same pathway have the same number of representative SNPs. Therefore, for each pathway, we calculate the average number of representative SNPs of genes and re-select SNPs using this number in the given pathway.

The computation needed for selecting representative SNPs for genes involves hundreds of permutations of thousands of subjects, recalculating the test statistic in each permutation based on about half a million SNPs, and testing on multiple values of the cutoff (i.e. threshold) point K. One way to limit the computational effort is to set the upper limit K_{upper} to 10 for the truncation point K. To further reduce the computational cost, we discard SNPs with large nominal P-values. On the other hand, if too few SNPs are selected, we might miss SNPs have low or moderate individual effects but jointly show a moderate or large effect. To seek a balance, we set a nominal threshold that is generous, say 0.05, i.e, only SNPs with P-values less than or equal to 0.05 will be selected. However, if none of the SNPs for a gene passes the threshold, the smallest SNP would be selected to avoid missing too many genes in pathway analysis. Both Kupper and P-value thresholds are are changeable in our software; other values can be used depending on the situation. In our experiment, we found that 10 as K_{upper} and 0.05 as the P-value threshold are useful choices.

5.2.2 SNP-based Pathway Enrichment Analysis

To conduct pathway analysis of SNP data from GWAS, we modified an existing gene set enrichment analysis (GSEA) algorithm [110]. The original GSEA algorithm ranks all genes by their

significance of differential expression and then looks for groups of biologically relevant genes that are enriched at either the top or bottom of the ranked list. To apply this idea to SNP data, we take the N selected representative SNPs across all the genes to form the SNP list, and compute the P-values for comparing genotype frequencies between cases and controls. To measure their strength of association, we define $r_i = \Phi^{-1}(1 - p_i), i = 1, ..., N$, where Φ^{-1} is the quantile function for the standard normal distribution. Let $r_1 \geq r_2 \geq ... \geq r_N$ be the sorted values from largest to smallest. A gene set sharing the same functional pathway is converted to a pathway consisting of SNPs. For a SNP-based pathway with N_H SNPs, we calculate a weighted Kolmogorov-Smirnov-like running sum [51] to measure the deviation of the pathway from a set of randomly picked SNPs in the genome:

$$ES(S) = \max_{1 \leq i \leq N} \{ \sum_{G_j \in S, j \leq i} \frac{|r_j|^p}{N_R} - \sum_{G_j \notin S, j \leq i} \frac{1}{N - N_H} \} \tag{5.1}$$

with $N_R = \sum_{G_j \in S} |r_j|^p$. Here p is a parameter that controls the weights to ensure SNPs with higher r values tend to contribute more in the pathway level. Following the original GSEA algorithm, we set $p = 1$.

5.2.3 Statistical Significance Evaluation

The enrichment score is expected to be high if most SNPs within a pathway are at the top of the list. We examine the statistical significance of a pathway by a permutation procedure. In each permutation, we permute the phenotype labels, recompute the P-values for SNPs and the corresponding enrichment score, denoted as perm_ES. Due to the size of large-scale genetic data, computational complexity would become extremely high when the number of permutations is very large. We used 1000 permutation-cycles to generate the permutated datasets. The nominal P-value is obtained by comparing the enrichment score for the observed phenotypes with scores computed

from permutated phenotypes.

$$Nom\_P = \frac{\#\ of(perm\_ES > obs\_ES)}{\#\ of\ permutations} \tag{5.2}$$

Adjustment for multiple testing is applied to control false positives. When many hypotheses are tested simultaneously, the probability that at least one type I error is committed is large. One common approach for accounting for multiple testing is to control the false discovery rate (FDR) [14]. The FDR is the expected proportion of falsely rejected hypotheses out of the rejected hypotheses. One can also control the family wise error rate (FWER), which is the probability of making one or more type I errors among the family of hypothesis tests. When the number of tests is large and some of the test hypotheses are in fact false, FWER is too conservative. Since multiple pathways might be involved in a complex trait, FDR, which controls the expected proportion of false discoveries, is more suited to identifying pathways relevant to a trait. To account for multiple testing in our pathway analyses, we used a robust method to estimate the false discovery rate proposed by Pounds and Cheng [96]. The q_value is the minimum FDR at which the test is called significant. For a given significant level α, the point estimate of $q\_value(\alpha)$ is defined as

$$q\text{-value}(\alpha) = min_{t \geq \alpha} FDR(t), \tag{5.3}$$

where $FDR(t)$ denotes an estimate of the proportion of tests when rejecting all null hypotheses with P-values less than or equal to the significance threshold t.

5.3 Results

To perform SNP-based pathway enrichment analysis of GWAS data, we developed a JAVA based software package called SNP Set Enrichment Analysis (SSEA) by extending the original GSEA code. SSEA consists of four procedures as outlined in Figure 5.1 : 1) calculating the P-value of the association of each SNP to a trait of interest, 2) selecting representative SNPs for each gene using an adaptive SNP combination method, calculating the average number of representative SNPs for genes in each pathway and reselecting SNPs for gene in each pathway, 3) ranking all selected SNPs by their P-values and testing if the SNPs from a pathway are enriched with high ranks, and 4) calculating the FDR of the discovered pathways. See Methods for details.

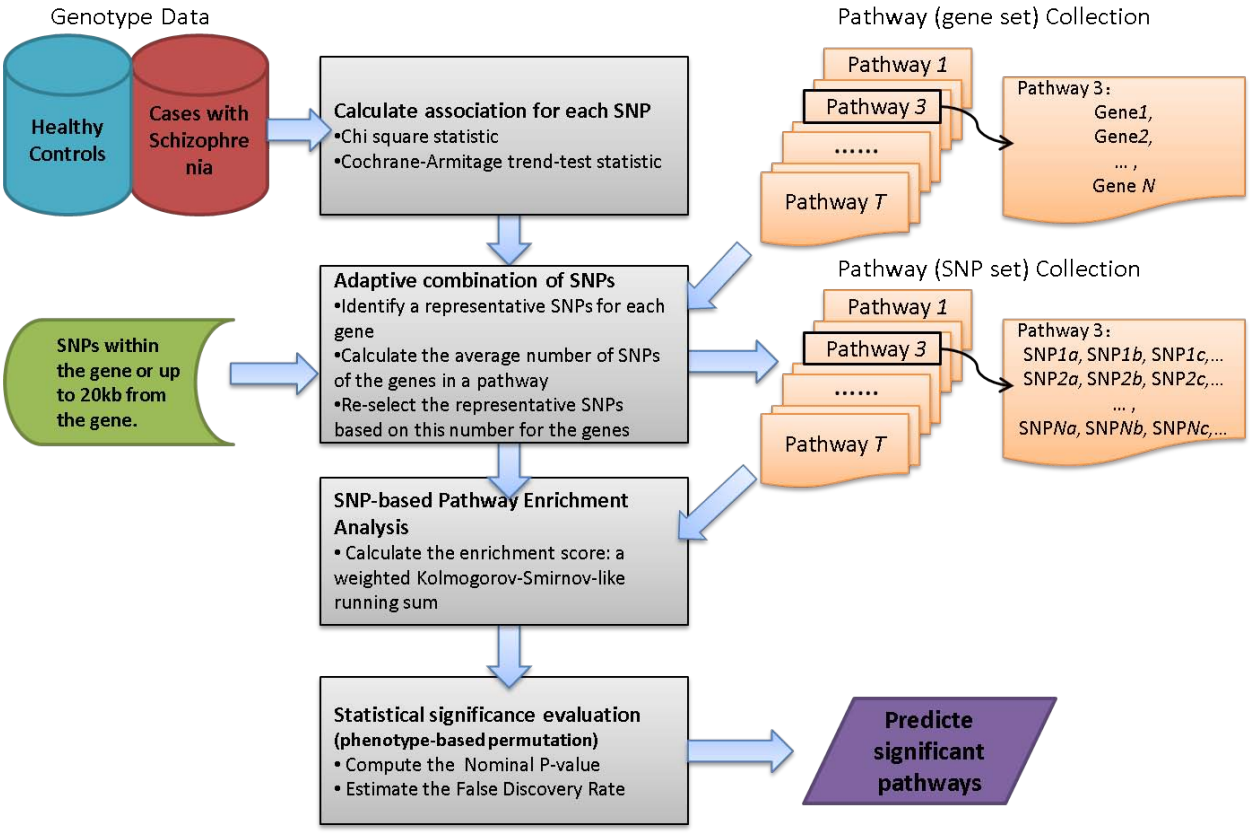


Figure 5.1: A diagram of procedures involved in SNP set enrichment analysis (SSEA)

We applied SSEA to two large genetically distinct GWAS data sets for schizophrenia from the Genetic Association Information Network (GAIN, `http://www.genome.gov/19518664`)

studies [81], available at the database of Genotype and Phenotype (dbGaP) [80]. The study version we reported here is phs000021.v2.p1 with general research use consent, which includes two samples; one is from the European American (EA) ancestry and the other one is from African American (AA) ancestry. Individuals in those two cohorts represent two genetically distinct populations [15, 113]. However, we should note that the two data sets were collected and quality controlled in a similar way, which might affect the independence of the two data sets. Both samples were genotyped by the Affymetrix SNP array 6.0. With GAIN quality-control criteria and after removing redundant subjects, the data sets included 1172 cases and 1378 controls in EA and 921 cases and 954 controls in AA. Since Linkage Disequilibrium (LD) is an important concern for selecting representative SNPs for each gene, we used Plink (`http://pngu.mgh.harvard.edu/~purcell/plink/`) to prune SNPs that are in strong LD (Plink uses 0.5 as the default pairwise R^2 threshold).

The final data used in our study consisted of 245,216 SNPs in EA and 482,914 SNPs in AA. The SNPs were assigned to genes on the basis of being located within the gene or up to 20kb from the gene. Most genes are associated with more than one SNP; we applied the adaptive rank truncated product of SNP association algorithm described in Method to selected representative SNPs for each gene. For pathways, we used 215 experimentally validated pathways from the KEGG database[64] (Release 55, accessed 12 September).

Application of SSEA to the two schizophrenia data sets resulted in the discovery of 22 pathways in the EA data set and 11 pathways in the AA data set with the nominal P-value less than or equal to 0.001. Using this P-value cutoff, the overall FDR is controlled within 5% for both data sets. The list of identified pathways from each sample is shown in the supplementary file, together with the related gene information.

Interestingly, the two data sets share 8 significant pathways; we used Monte Carlo simulation to assess the significance of sharing and found the P-value is less than 1.0E-6. To examine whether our method detects biologically relevant pathways or random combinations of genes, we permu-

tated genes and generated 215 random pathways for both EA and AA data sets; Our method only detected 6 significant pathways ($p <= 0.001$) in EA and 3 in AA, and none them is shared, indicating that the number of significant pathways detected by our methods is more than what expected by chance, and those significant pathways are likely to be biologically relevant. The list of the 8 replicated pathways are shown in Table 5.1, together with their nominal P-values, the gene set sizes and the SNP set sizes associated with each pathway, and the full list of 22 significant pathways in EA and 11 significant pathways in AA are shown in Table 5.2 and Table 5.3.

Table 5.1: Eight significant pathways (P$\leq$ 0.001) discovered in both European American ancestry and African American ancestry data sets of Schizophrenia

Pathways		European Ancestry (EA)			African Ancestry (AA)		
		NomP	GENE SIZE	SNP SIZE	NomP	GENE SIZE	SNP SIZE
HSA04720	Long-term potentiation	<0.001	67	135	0.001	67	66
HSA04270	Vascular smooth muscle contraction	<0.001	107	215	<0.001	107	105
HSA05412	Arrhythmogenic right ventricular cardiomyopathy (ARVC)	<0.001	72	144	<0.001	72	135
HSA04020	Calcium signaling pathway	<0.001	168	337	<0.001	174	165
HSA04360	Axon guidance	<0.001	122	245	<0.001	126	120
HSA04080	Neuroactive ligand-receptor interaction	<0.001	256	509	<0.001	266	248
HSA04510	Focal adhesion	<0.001	186	378	<0.001	191	186
HSA04730	Long-term depression	< 0.001	68	134	0.001	68	129

Table 5.2: Genes overlapping between eight significant pathways in EA data set

	HSA04360	HSA04730	HSA04720	HSA04080	HSA04020	HSA04270	HSA05412	HSA04510
HSA04360	126	7	11	0	7	7	5	19
HSA04730	7	68	24	7	18	40	0	10
HSA04720	11	24	67	9	40	32	1	14
HSA04080	0	7	9	256	59	10	0	0
HSA04020	7	18	40	59	168	43	7	10
HSA04270	7	40	32	10	43	107	4	18
HSA05412	5	0	1	0	7	4	72	27
HSA04510	19	10	14	0	10	18	27	186

Table 5.3: Genes overlapping between eight significant pathways in AA data set

	HSA04360	HSA04730	HSA04720	HSA04080	HSA04020	HSA04270	HSA05412	HSA04510
HSA04360	126	7	11	0	7	5	1	20
HSA04730	7	69	24	7	18	40	0	10
HSA04720	11	24	67	9	42	34	1	14
HSA04080	0	7	9	256	62	10	0	0
HSA04020	5	18	42	62	168	45	7	10
HSA04270	7	40	34	10	45	107	4	18
HSA05412	1	0	1	0	7	4	72	27
HSA04510	20	10	14	0	10	18	27	186

Schizophrenia [MIM 181500] is a complex brain disorder characterized by disturbances in mul-

tiple domains of brain function, including cognitive, emotional, and perceptual processes [70]. Evidence for schizophrenia as a neurodevelopment disorder began more than 30 years ago [121] and has been accepted commonly [85]. It is intriguing to note that the 8 pathways discovered by SSEA in both the EA and AA samples included 4 pathways important for neurodevelopment and neuronal functioning, such as axon guidance pathway, neuroactive ligand-receptor interaction pathway, long-term depression pathway and long-term potentiation pathway. Axon guidance pathway and neuroactive ligand-receptor interaction pathway are directly related to neuroplasticity and neuropathology, and thus are important to the genetic mechanism of schizophrenia [106]. Long-term depression pathway and long-term potentiation pathway were reported to be important for synaptic plasticity development and related to schizophrenia [28, 45]. Besides, axon guidance pathway, long-term depression pathway and long-term potentiation pathway were reported in a recent study where pathways were overrepresented by genes disrupted by copy number variants in schizophrenia cases [117]. Genes in the focal adhesion pathway are principally involved in the biological processes for synaptic transmission and cell adhesion [37]. In addition, arrhythmogenic right ventricular cardiomyopathy (ARVC) pathway is related to cardiovascular disease, which supports the previous study that patients with schizophrenia had higher rates of cardiovascular disease and mortality compared with the general population [16, 30]. We took a further investigation of gene intersections in the remaining two non-schizophrenia specific pathways. We found the calcium signaling pathway shared $40(24\%)$ genes with long-term potentiation pathway, while vascular smooth muscle contraction shared $40(37\%)$ genes with long-term depression pathway; the same genes implicated in different pathways might be a reason for their enrichment in our study.

As a comparison, we also applied another two methods, smallest P-value and PLINK set-based tests [97], to the two GAIN data sets. The smallest P-value method, which only used the SNP with the smallest P-value to represent a gene, detected 6 and 2 significant pathways in EA and AA data sets, respectively; only one was shared by both data sets. This showed our method could improve the power of detecting causal pathways by using multiple SNPs to represent a gene. For the set-based test method (with parameters –set-p 0.05, –set-r^2 0.5, the same as that in our method), 3

significant pathways were detected in EA, and 2 in AA, but none was shared between the two data sets. One reason for the loss of power of these two methods might be their favouring of pathways containing large numbers of genes and genes with large number of SNPs, as larger pathways are expected to show more significant genes or SNPs just by chance. We checked several potential factors that might affect pathway significance: pathway size, gene size, total bp content, and average content. We found these factors are uncorrelated with pathway significance 5.2, confirming that using multiple representative SNPs per gene and permutations are able to reduce the bias introduced by gene and pathway sizes.

Relaxing the nominal P-value cutoff to 0.01, with FDR q-value controlled within 10%, resulted in 40 significant pathways detection in EA data set, and 27 significant pathways in AA data set. Among them, 17 pathways are shared (Monte Carlo simulation P-value for sharing is less than 1.0E-6). The full list of 17 shared pathways is shown in Table 5.4.

Table 5.4: Eight significant pathways (P$\leq$ 0.01) discovered in both European American ancestry and African American ancestry data sets of Schizophrenia

Pathways		European Ancestry (EA)			African Ancestry (AA)		
		NomP	GENE SIZE	SNP SIZE	NomP	GENE SIZE	SNP SIZE
HSA04720	Long-term potentiation	0.001	67	66	<0.001	69	135
HSA04270	Vascular smooth muscle contraction	<0.001	107	105	<0.001	113	215
HSA05412	Arrhythmogenic right ventricular cardiomyopathy (ARVC)	<0.001	72	135	<0.001	74	144
HSA04020	Calcium signaling pathway	<0.001	168	165	<0.001	174	337
HSA04972	Pancreatic secretion	0.008	93	91	<0.001	94	184
HSA04360	Axon guidance	<0.001	122	120	<0.001	126	245
HSA04080	Neuroactive ligand-receptor interaction	<0.001	256	248	<0.001	266	509
HSA04510	Focal adhesion	<0.001	186	186	<0.001	191	378
HSA04730	Long-term depression	<0.001	68	129	0.001	68	134
HSA00330	Arginine and proline metabolism	0.001	47	47	0.002	52	102
HSA04970	Salivary secretion	0.003	80	151	0.002	86	166
HSA05146	Amoebiasis	0.009	100	99	0.003	103	199
HSA05414	Dilated cardiomyopathy	<0.001	88	86	0.005	90	173
HSA04070	Phosphatidylinositol signaling system	0.002	75	74	0.006	77	150
HSA04512	ECM-receptor interaction	0.001	81	81	0.007	82	161
HSA04260	Cardiac muscle contraction	0.009	63	61	0.009	67	128
HSA04540	Gap junction	0.003	80	80	0.009	85	165

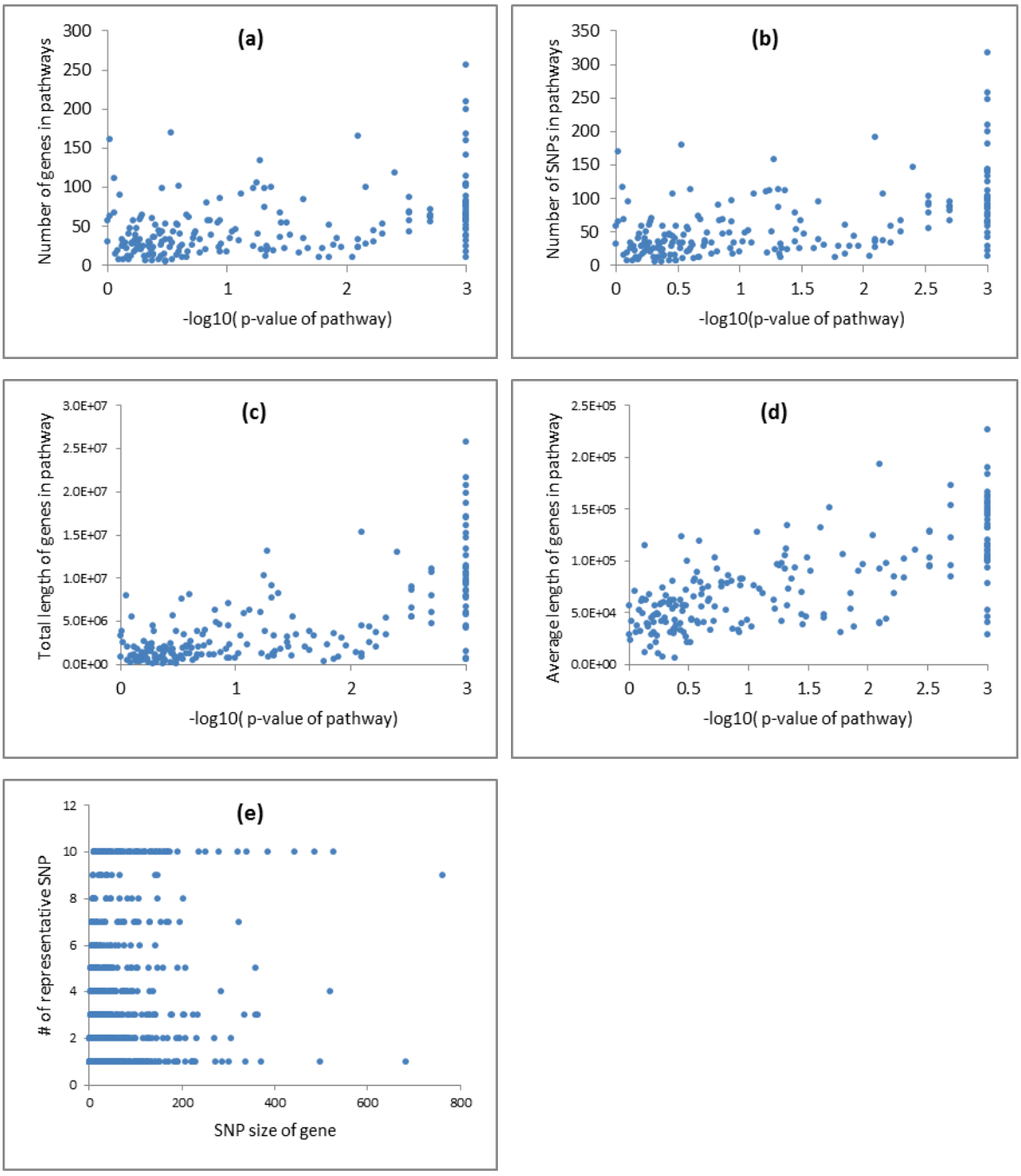


Figure 5.2: the significance of pathway(-log10(P-value)) in EA versus (a) the number of genes in pathways, (b) the number of significant SNPs in pathways, (c) Total length (bp) of genes in pathways, (d) average length (bp) of genes in pathways.(e)the number of representative SNPs selected verse the number of SNPs belongs to a gene.

5.4 Discussion

The traditional strategy for GWAS studies tests one SNP at a time. Although widely used, single-SNP GWAS analysis does not have adequate power to detect SNPs that have marginally weak, but jointly strong genetic effects. Jointly analyzing SNPs within the same biological pathway simultaneously complements the single-SNP analysis and can reveal new insights to the understanding of complex human traits. Our SNP set enrichment analysis operates on representative SNPs of genes and then combines the effects of SNPs within the same pathway by a weighted Kolmogorov-Smirnov running sum statistic test [51]. This strategy has the potential to increase the chance of identifying genetic variants that that individually have a modest risk.

Compared to gene set enrichment analysis, the SNP set enrichment analysis is a much larger scale and is more computationally challenging. Several pathway-based methods have recently been developed to analyse GWAS [119, 50, 65, 114, 53, 133]. In general, these methods can be classified into two categories, depending upon how representative SNPs for each gene are chosen: one selects the most significant SNP per gene, and the other selects all SNPs located within a gene [20]. Both approaches have limitations. Using all available SNPs per gene not only poses computational challenges, but also introduces significant amounts of noise into the analysis. Using the most significant SNP per gene might miss SNPs with moderate strength individually but strong effects jointly, and in addition it introduces biases of favouring large extensive pathways and genes with greater numbers of SNPs. The SSEA method we proposed uses an adaptive approach to choose SNPs in each gene, and can eliminate the limitations of other strategies.

It is also worthy to point out that the number of selected SNPs varies between genes. This is because we used permutations to decide both the number and the set of SNPs to represent each gene. The permutation of phenotypes and recalculation of statistical values for about half a million SNPs and thousands of subjects is computationally expensive. To seek a balance between the computational complexity and not losing too much information from SNPs, we set a nominal

significance threshold chose only SNPs with smaller P-value for pathway analysis. To further reduce computation, we recommend using an upper limit for the number of representative SNPs for each gene.

Our method has a critical assumption. In combing P-values of SNPs in a gene we assume that the P-values are independent, although in reality some SNPs in a gene are in linkage disequilibrium (LD). When comparing the results with and without removing SNPs in strong LD, we found there is no big difference between them. However, a future direction is to relax this assumption and develop a SNP selection method that explicitly takes the LD patterns into account rather than remove SNPs in LD. It is interesting to note that Peng et al. [94] also found that ignoring LD could actually lead to better results than methods with very conservative multiple testing corrections. The permutation test we consider might partially alleviate the effect due to LD.

A critical component for the success of the pathway-based analysis is the availability of a comprehensive collection of relevant gene sets related to the disease/genetic trait of interest. Current understanding of gene functions and pathways is still very limited. This is especially the case for neuropsychiatric diseases, as most of the gene sets currently available were generated based on experiments done on tumor cell lines. As a consequence, we have only limited knowledge regarding the pathways involved in brain development, and normal and pathological activities. In this regard, the pathways discovered by SSEA for schizophrenia are likely to be substantially incomplete. We expect the performance would improve as better and more comprehensive disease-related pathways become available. A future challenge is to curate pathways and gene sets in a disease specific way, possibly by taking advantage of the high-throughput functional genomics tools.

In summary, we have developed a new SNP-based method, called SNP Set Enrichment Analysis (SSEA), for pathway analysis of GWAS data. SSEA selects a multiple and varying number of SNPs to represent each gene using an adaptive truncated product statistic. The selected SNPs are then ranked and enrichment of pathways is tested using a weighted KolmogorovSmirnov test. We tested SSEA in two genetically distinct GWAS studies of schizophrenia with large samples, and

discovered 22 significant pathways in the European-American sample and 11 significant pathways in the African-American sample. Eight important pathways were found in both distinct samples providing support for our method. Our user-friendly JAVA package SSEA is public available, it consist of two step, step 1 will generate significant SNPs for each gene, then step 2 will find out enriched pathway, the interfaces of two steps as shown in 5.3 and 5.4.

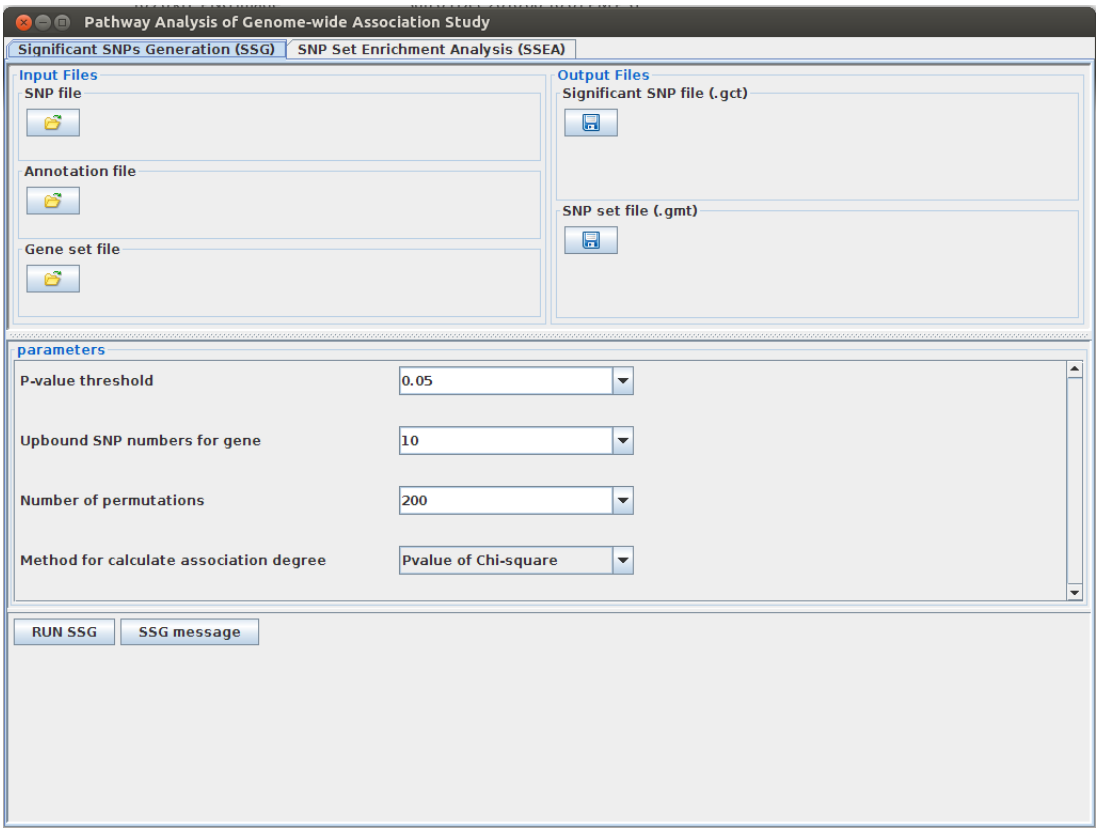


Figure 5.3: Interface of significant SNPs generation step.

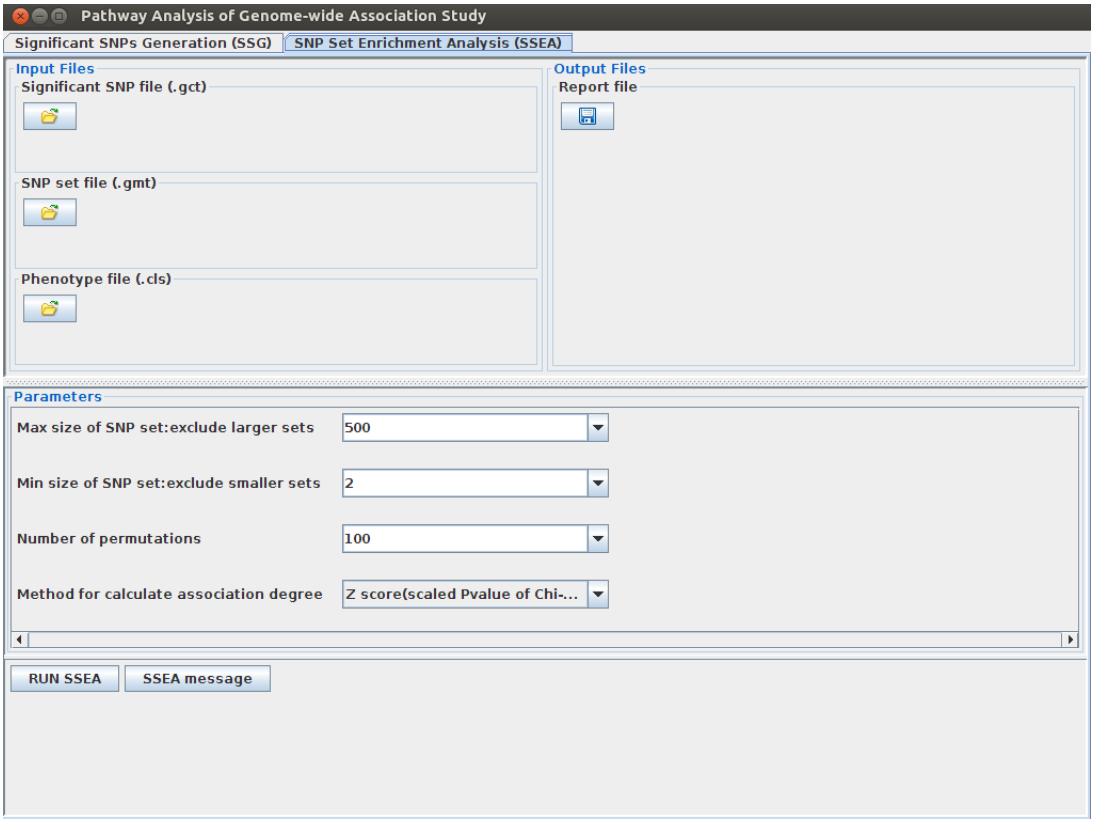


Figure 5.4: Interface of SNP set enrichment analysis step.

Chapter 6

Conclusion

With the advent of high-throughput technologies, such as microarray and next generation sequencing (NGS), we are now experiencing a data explosion in biomedical field. Comparing to 2005, when the 1st next generation sequencing system was introduced by lift Sciences, there are about 1 million folds improvement in the rate of DNA sequence generation. At the same time, the cost of to perform this high-throughput sequencing has dropped substantially. For instance, in January 2014, Illumina announced its new top-of-the-line system, the HiSeq X Ten Sequencing System, which is capable of sequencing complete human genomes at $1000 each, and has a throughput of 600 billion base pairs per day. These NGS technological advances allow high-throughput sequencing to become a practical tool for many areas of biology and medicine. Since large-scale data sets will become larger and more common in the future, most research in biomedical science is moving from a hypothesis-driven to a data-driven approach.

This high-throughput sequencing technology has significant impact on the detection, management and treatment of disease, and opens the door for personalized therapies. Meanwhile, it greatly expands our ability to study basic research, like genome structure, how genes are regulated and how cell and tissue differentiation occurs. The extraction of useful knowledge from this voluminous

data presents significant challenges related to data storage, processing, analysis and interpretation. More importantly, for computational biology people, it represents an unprecedented opportunity for uncovering the hidden pattern in high-throughput data, since it allows development of novel algorithms, new data analysis pipelines, as well as the creation of prediction models for biological applications. The availability of these new solutions to deal with the huge amount of data is very important for data annotation, integration, and finally for inferring knowledge and making it available to biomedical researchers.

In this thesis, we have presented four such computational and algorithmic solutions for high-throughput data. We first developed a fast and efficient sequence aligner to map high-throughput sequencing data to the reference sequence, which is an first step to transform high-throughput data into hypotheses and conclusions. Then we focused on providing computation solution for a specific problem we are interested in, the polyadenyltion code. We developed a bioinformatics pipeline to identify and profile genes with significantly APA switches between two biological conditions, so as to help determine genes regulated by specific polyadenylation factors. We further explored the polyadenylation code that distinguishes tissue-specific and constitutive mRNA alternative polyadenylation via a genome-wide modeling framework. Further more, we presented a robust and accurate statistical method for identifying disease related pathways for GWAS study. Deregulation of APA has been demonstrated in a variety of human diseases. This pathway detection method can be used here for detecting pathways involved in disease caused by APA, such as cancer.

Bibliography

[1] Global patterns of tissue-specific alternative polyadenylation in drosophila. *Cell Reports*, 1(3):277 – 289, 2012.

[2] A. Ahmadi, A. Behm, N. Honnalli, C. Li, L. Weng, and X. Xie. Hobbes: optimized gram-based methods for efficient read alignment. *Nucleic acids research*, 40(6):e41–e41, 2012.

[3] F. Ahmed, M. Kumar, and G. P. Raghava. Prediction of polyadenylation signals in human dna sequences using nucleotide frequencies. *In silico biology*, 9(3):135–148, 2009.

[4] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Selected Papers of Hirotugu Akaike*, pages 199–213. Springer, 1998.

[5] M. N. Akhtar, S. A. Bukhari, Z. Fazal, R. Qamar, and I. A. Shahmuradov. Polyar, a new computer program for prediction of poly (a) sites in human sequences. *BMC genomics*, 11(1):646, 2010.

[6] C. Alkan, J. M. Kidd, T. Marques-Bonet, G. Aksay, F. Antonacci, F. Hormozdiari, J. O. Kitzman, C. Baker, M. Malig, O. Mutlu, S. C. Sahinalp, R. A. Gibbs, and E. E. Eichler. Personalized copy-number and segmental duplication maps using next-generation sequencing. *Nature Genetics*, 41(10):1061–1067, 2009.

[7] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[8] J. J. An, K. Gharami, G.-Y. Liao, N. H. Woo, A. G. Lau, F. Vanevski, E. R. Torre, K. R. Jones, Y. Feng, B. Lu, et al. Distinct role of long 3 utr bdnf mrna in spine morphology and synaptic plasticity in hippocampal neurons. *Cell*, 134(1):175–187, 2008.

[9] T. Ara, F. Lopez, W. Ritchie, P. Benech, and D. Gautheret. Conservation of alternative polyadenylation patterns in mammalian genes. *BMC genomics*, 7(1):189, 2006.

[10] S. E. Baranzini, N. W. Galwey, J. Wang, P. Khankhanian, R. Lindberg, D. Pelletier, W. Wu, B. M. Uitdehaag, L. Kappos, C. H. Polman, et al. Pathway and network-based analysis of genome-wide association studies in multiple sclerosis. *Human molecular genetics*, 18(11):2078–2090, 2009.

[11] M. J. Bauer, A. J. Cox, and D. J. Evers. ELANDv2 - fast gapped read mapping for illumina reads. In *ISMB*. ISCB, 2010.

[12] F.-A. Bava, C. Eliscovich, P. G. Ferreira, B. Miñana, C. Ben-Dov, R. Guigó, J. Valcárcel, and R. Méndez. Cpeb1 coordinates alternative 3 [prime]-utr formation with translational regulation. *Nature*, 2013.

[13] E. Beaudoing, S. Freier, J. R. Wyatt, J.-M. Claverie, and D. Gautheret. Patterns of variant polyadenylation signal usage in human genes. *Genome research*, 10(7):1001–1010, 2000.

[14] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 289–300, 1995.

[15] A. M. Bowcock, J. R. Kidd, J. L. Mountain, J. Hebert, L. Carotenuto, K. K. Kidd, and L. Cavalli-Sforza. Drift, admixture, and selection in human evolution: a study with dna polymorphisms. *Proceedings of the National Academy of Sciences*, 88(3):839–843, 1991.

[16] M. Buda, M. T. Tsuang, and J. A. Fleming. Causes of death in dsm-iii schizophrenics and other psychotics (atypical group): a comparison with the general population. *Archives of general psychiatry*, 45(3):283–285, 1988.

[17] S. Burkhardt and J. Kärkkäinen. Better filtering with gapped q-grams. *Fundam. Inf.*, 56(1,2):51–70, 2002.

[18] M. Burrows and D. Wheeler. A block sorting lossless data compression algorithm. *Technical Report 124. Palo Alto, CA: Digital Equipment Corporation*, 25, 1994.

[19] P. R. Burton, D. G. Clayton, L. R. Cardon, N. Craddock, P. Deloukas, A. Duncanson, D. P. Kwiatkowski, M. I. McCarthy, W. H. Ouwehand, N. J. Samani, et al. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature*, 447(7145):661–678, 2007.

[20] R. M. Cantor, K. Lange, and J. S. Sinsheimer. Prioritizing gwas results: a review of statistical methods and recommendations for their application. *The American Journal of Human Genetics*, 86(1):6–22, 2010.

[21] S. Chan, E.-A. Choi, and Y. Shi. Pre-mrna 3-end processing complex assembly and function. *Wiley Interdisciplinary Reviews: RNA*, 2(3):321–335, 2011.

[22] S. L. Chan, I. Huppertz, C. Yao, L. Weng, J. J. Moresco, J. R. Yates, J. Ule, J. L. Manley, and Y. Shi. Cpsf30 and wdr33 directly bind to aauaaa in mammalian mrna 3 processing. *Genes & Development*, pages gad–250993, 2014.

[23] T.-H. Chang, L.-C. Wu, Y.-T. Chen, H.-D. Huang, B.-J. Liu, K.-F. Cheng, and J.-T. Horng. Characterization and prediction of mrna polyadenylation sites in human genes. *Medical & biological engineering & computing*, 49(4):463–472, 2011.

[24] S. Chaudhuri, V. Ganti, and R. Kaushik. A primitive operator for similarity joins in data cleaning. In *ICDE*, 2006.

[25] Y. Cheng, R. M. Miura, and B. Tian. Prediction of mrna polyadenylation sites by support vector machine. *Bioinformatics*, 22(19):2320–2325, 2006.

[26] D. Chung, P. Kuan, B. Li, R. Sanalkumar, K. Liang, E. Bresnick, C. Dewey, and S. Keleş. Discovering transcription factor binding sites in highly repetitive regions of genomes with multi-read analysis of chip-seq data. *PLoS Computational Biology*, 7(7):e1002111, 2011.

[27] D. F. Colgan and J. L. Manley. Mechanism and regulation of mrna polyadenylation. *Genes & development*, 11(21):2755–2766, 1997.

[28] G. L. Collingridge, S. Peineau, J. G. Howland, and Y. T. Wang. Long-term depression in the cns. *Nature Reviews Neuroscience*, 11(7):459–473, 2010.

[29] D. W. Collins and T. H. Jukes. Rates of transition and transversion in coding sequences since the human-rodent divergence. *Genomics*, 20(3):386 – 396, 1994.

[30] S. M. Curkendall, J. Mo, D. B. Glasser, S. M. Rose, and J. K. Jones. Cardiovascular disease in patients with schizophrenia in saskatchewan, canada. *The Journal of clinical psychiatry*, 65(5):715–720, 2004.

[31] R. K. Curtis, M. Orešič, and A. Vidal-Puig. Pathways to the analysis of microarray data. *Trends in biotechnology*, 23(8):429–435, 2005.

[32] E. de Klerk, A. Venema, S. Y. Anvar, J. J. Goeman, O. Hu, C. Trollet, G. Dickson, J. T. den Dunnen, S. M. van der Maarel, V. Raz, et al. Poly (a) binding protein nuclear 1 levels affect alternative polyadenylation. *Nucleic acids research*, page gks655, 2012.

[33] A. Derti, P. Garrett-Engele, K. D. MacIsaac, R. C. Stevens, S. Sriram, R. Chen, C. A. Rohl, J. M. Johnson, and T. Babak. A quantitative atlas of polyadenylation in five mammals. *Genome research*, 22(6):1173–1183, 2012.

[34] Y. Ding, Y. Tang, C. K. Kwok, Y. Zhang, P. C. Bevilacqua, and S. M. Assmann. In vivo genome-wide profiling of rna secondary structure reveals novel regulatory features. *Nature*, 505(7485):696–700, 2014.

[35] A. Döring, D. Weese, T. Rausch, and K. Reinert. Seqan an efficient, generic c++ library for sequence analysis. *BMC Bioinformatics*, 9(1):11, 2008.

[36] R. Elkon, A. P. Ugalde, and R. Agami. Alternative cleavage and polyadenylation: extent, regulation and function. *Nature Reviews Genetics*, 14(7):496–506, 2013.

[37] A. Ertel and A. Tozeren. Switch-like genes populate cell communication pathways and are enriched for extracellular proteins. *BMC genomics*, 9(1):3, 2008.

[38] P. Ferragina and G. Manzini. An experimental study of an opportunistic index. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 269–278. Society for Industrial and Applied Mathematics, 2001.

[39] N. B. Freimer and C. Sabatti. Human genetics: variants in common diseases. *Nature*, 445(7130):828–830, 2007.

[40] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[41] N. H. Gehring, U. Frede, G. Neu-Yilik, P. Hundsdoerfer, B. Vetter, M. W. Hentze, and A. E. Kulozik. Increased efficiency of mrna 3 end formation: a new genetic mechanism contributing to hereditary thrombophilia. *Nature genetics*, 28(4):389–392, 2001.

[42] D. B. Goldstein. Common genetic variation and human traits. *New England Journal of Medicine*, 360(17):1696, 2009.

[43] G. R. Grant, J. Liu, and C. J. Stoeckert. A practical false discovery rate approach to identifying patterns of differential expression in microarray data. *Bioinformatics*, 21(11):2684–2690, 2005.

[44] J. Gudmundsson, P. Sulem, D. F. Gudbjartsson, T. Blondal, A. Gylfason, B. A. Agnarsson, K. R. Benediktsdottir, D. N. Magnusdottir, G. Orlygsdottir, M. Jakobsdottir, et al. Genome-wide association and replication studies identify four variants associated with prostate cancer susceptibility. *Nature genetics*, 41(10):1122–1126, 2009.

[45] A. Guo, J. Sun, B. Riley, D. Thiselton, K. Kendler, and Z. Zhao. The dystrobrevin-binding protein 1 gene: features and networks. *Molecular psychiatry*, 14(1):18–29, 2008.

[46] F. Hach, F. Hormozdiari, C. Alkan, F. Hormozdiari, I. Birol, E. E. Eichler, and S. C. Sahinalp. mrsfast: a cache-oblivious algorithm for short-read mapping. *Nature Methods*, 7(8):576–577, 2010.

[47] D. Hafez, T. Ni, S. Mukherjee, J. Zhu, and U. Ohler. Genome-wide identification and predictive modeling of tissue-specific alternative polyadenylation. *Bioinformatics*, 29(13):i108–i116, 2013.

[48] C. A. Haiman, N. Patterson, M. L. Freedman, S. R. Myers, M. C. Pike, A. Waliszewska, J. Neubauer, A. Tandon, C. Schirmer, G. J. McDonald, et al. Multiple regions within 8q24 independently affect risk for prostate cancer. *Nature genetics*, 39(5):638–644, 2007.

[49] J. N. Hirschhorn. Genomewide association studies–illuminating biologic pathways. *New England Journal of Medicine*, 360(17):1699, 2009.

[50] M. Holden, S. Deng, L. Wojnowski, and B. Kulle. Gsea-snp: applying gene set enrichment analysis to snp data from genome-wide association studies. *Bioinformatics*, 24(23):2784–2785, 2008.

[51] M. Hollander, D. A. Wolfe, and E. Chicken. *Nonparametric statistical methods*, volume 751. John Wiley & Sons, 2013.

[52] P. Holmans, E. K. Green, J. S. Pahwa, M. A. Ferreira, S. M. Purcell, P. Sklar, M. J. Owen, M. C. O'Donovan, and N. Craddock. Gene ontology analysis of gwa study data sets provides insights into the biology of bipolar disorder. *The American Journal of Human Genetics*, 85(1):13–24, 2009.

[53] M.-G. Hong, Y. Pawitan, P. K. Magnusson, and J. A. Prince. Strategies and issues in the detection of pathway enrichment in genome-wide association studies. *Human genetics*, 126(2):289–301, 2009.

[54] J. Hu, C. S. Lutz, J. Wilusz, and B. Tian. Bioinformatic identification of candidate cis-regulatory elements involved in human mrna polyadenylation. *Rna*, 11(10):1485–1493, 2005.

[55] M. Jenal, R. Elkon, F. Loayza-Puch, G. van Haaften, U. Kühn, F. M. Menzies, J. A. Vrielink, A. J. Bos, J. Drost, K. Rooijers, et al. The poly (a)-binding protein nuclear 1 suppresses alternative cleavage and polyadenylation sites. *Cell*, 149(3):538–553, 2012.

[56] G. Ji, J. Zheng, Y. Shen, X. Wu, R. Jiang, Y. Lin, J. C. Loke, K. M. Davis, G. J. Reese, and Q. Q. Li. Predictive modeling of plant messenger rna polyadenylation sites. *BMC bioinformatics*, 8(1):43, 2007.

[57] Y. Ji, Y. Xu, Q. Zhang, K. Tsui, Y. Yuan, C. Norris Jr, S. Liang, and H. Liang. Bm-map: Bayesian mapping of multireads for next-generation sequencing data. *Biometrics*, 2011.

[58] Z. Ji and B. Tian. Reprogramming of 3 untranslated regions of mrnas by alternative polyadenylation in generation of pluripotent stem cells from different cell types. *PLoS One*, 4(12):e8419, 2009.

[59] P. Jokinen, J. Tarhio, and E. Ukkonen. A comparison of approximate string matching algorithms. *Softw. Pract. Exper.*, 26(12):1439–1458, 1996.

[60] K. Kadota, T. Konishi, and K. Shimizu. Evaluation of two outlier-detection-based methods for detecting tissue-selective genes from microarray data. *Gene regulation and systems biology*, 1:9, 2007.

[61] K. Kadota, S.-I. Nishimura, H. Bono, S. Nakamura, Y. Hayashizaki, Y. Okazaki, and K. Takahashi. Detection of genes with tissue-specific expression patterns using akaikes information criterion procedure. *Physiological genomics*, 12(3):251–259, 2003.

[62] K. Kadota, J. Ye, Y. Nakai, T. Terada, and K. Shimizu. Roku: a novel method for identification of tissue-specific genes. *BMC bioinformatics*, 7(1):294, 2006.

[63] M. Kalkatawi, F. Rangkuti, M. Schramm, B. R. Jankovic, A. Kamau, R. Chowdhary, J. A. Archer, and V. B. Bajic. Dragon polya spotter: predictor of poly (a) motifs within human genomic dna sequences. *Bioinformatics*, 28(1):127–129, 2012.

[64] M. Kanehisa and S. Goto. Kegg: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1):27–30, 2000.

[65] T. Kishi, M. Ikeda, T. Kitajima, Y. Yamanouchi, Y. Kinoshita, K. Kawashima, T. Okochi, T. Inada, N. Ozaki, and N. Iwata. Genetic association analysis of tagging snps in alpha4 and beta2 subunits of neuronal nicotinic acetylcholine receptor genes (chrna4 and chrnb2) with schizophrenia in the japanese population. *Journal of Neural Transmission*, 115(10):1457–1461, 2008.

[66] P. Kraft and S. Raychaudhuri. Complex diseases, complex genes: keeping pathways on the right track. *Epidemiology (Cambridge, Mass.)*, 20(4):508, 2009.

[67] B. Lackford, C. Yao, G. M. Charles, L. Weng, X. Zheng, E.-A. Choi, X. Xie, J. Wan, Y. Xing, J. M. Freudenberg, et al. Fip1 regulates mrna alternative polyadenylation to promote stem cell self-renewal. *The EMBO journal*, 33(8):878–889, 2014.

[68] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biology*, 10:r25, 2009.

[69] B. Langmead, C. Trapnell, M. Pop, S. L. Salzberg, et al. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biol*, 10(3):R25, 2009.

[70] D. A. Lewis and J. A. Lieberman. Catching up on schizophrenia: natural history and neurobiology. *Neuron*, 28(2):325–334, 2000.

[71] C. Li, B. Wang, and X. Yang. VGRAM: Improving performance of approximate queries on string collections using variable-length grams. In *VLDB*, 2007.

[72] H. Li and R. Durbin. Fast and accurate short read alignment with burrows-wheeler transform. *Bioinformatics*, 25:1754–1760, 2009.

[73] H. Li, J. Ruan, and R. Durbin. Mapping short dna sequencing reads and calling variants using mapping quality scores. *Genome research*, 18(11):1851, 2008.

[74] R. Li, C. Yu, Y. Li, T.-W. Lam, S.-M. Yiu, K. Kristiansen, and J. Wang. Soap2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25:1966–1967, 2009.

[75] S. Lianoglou, V. Garg, J. L. Yang, C. S. Leslie, and C. Mayr. Ubiquitously transcribed genes use alternative polyadenylation to achieve tissue-specific expression. *Genes & development*, 27(21):2380–2396, 2013.

[76] G.-Y. Liao, J. J. An, K. Gharami, E. G. Waterhouse, F. Vanevski, K. R. Jones, and B. Xu. Dendritically targeted bdnf mrna is essential for energy balance and response to leptin. *Nature medicine*, 18(4):564–571, 2012.

[77] H. Lin, Z. Zhang, M. Zhang, B. Ma, and M. Li. Zoom! zillions of oligos mapped. *Bioinformatics*, 24(21):2431, 2008.

[78] Y. Lin, Z. Li, F. Ozsolak, S. W. Kim, G. Arango-Argoty, T. T. Liu, S. A. Tenenbaum, T. Bailey, A. P. Monaghan, P. M. Milos, et al. An in-depth map of polyadenylation sites in cancer. *Nucleic acids research*, 40(17):8460–8471, 2012.

[79] R. Lorenz, S. H. Bernhart, C. H. Zu Siederdissen, H. Tafer, C. Flamm, P. F. Stadler, I. L. Hofacker, et al. Viennarna package 2.0. *Algorithms for Molecular Biology*, 6(1):26, 2011.

[80] M. D. Mailman, M. Feolo, Y. Jin, M. Kimura, K. Tryka, R. Bagoutdinov, L. Hao, A. Kiang, J. Paschall, L. Phan, et al. The ncbi dbgap database of genotypes and phenotypes. *Nature genetics*, 39(10):1181–1186, 2007.

[81] T. A. Manolio, L. L. Rodriguez, L. Brooks, G. Abecasis, D. Ballinger, M. Daly, P. Donnelly, S. V. Faraone, K. Frazer, S. Gabriel, et al. New models of collaboration in genome-wide association studies: the genetic association information network. *Nature genetics*, 39(9):1045–1051, 2007.

[82] C. Mayr and D. P. Bartel. Widespread shortening of 3 utrs by alternative cleavage and polyadenylation activates oncogenes in cancer cells. *Cell*, 138(4):673–684, 2009.

[83] M. I. McCarthy, G. R. Abecasis, L. R. Cardon, D. B. Goldstein, J. Little, J. P. Ioannidis, and J. N. Hirschhorn. Genome-wide association studies for complex traits: consensus, uncertainty and challenges. *Nature Reviews Genetics*, 9(5):356–369, 2008.

[84] G. Meyers. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *J. ACM*, 46:395–415, 1999.

[85] S. Miyamoto, A. S. LaMantia, G. E. Duncan, P. Sullivan, J. H. Gilmore, and J. A. Lieberman. Recent advances in the neurobiology of schizophrenia. *Molecular interventions*, 3(1):27, 2003.

[86] M. J. Moore and N. J. Proudfoot. Pre-mrna processing reaches back totranscription and ahead to translation. *Cell*, 136(4):688–700, 2009.

[87] A. R. Morris, A. Bos, B. Diosdado, K. Rooijers, R. Elkon, A. S. Bolijn, B. Carvalho, G. A. Meijer, and R. Agami. Alternative cleavage and polyadenylation during colorectal cancer development. *Clinical Cancer Research*, 18(19):5256–5266, 2012.

[88] D. Newkirk, J. Biesinger, A. Chon, K. Yokomori, and X. Xie. Arem: aligning short reads from chip-sequencing by expectation maximization. In *Research in Computational Molecular Biology*, pages 283–297. Springer, 2011.

[89] Z. Ning, A. J. Cox, and J. C. Mullikin. Ssaha: a fast search method for large dna databases. *Genome Res*, 11(10):1725–1729, 2001.

[90] N. M. Nunes, W. Li, B. Tian, and A. Furger. A functional human poly (a) site requires only a potent dse and an a-rich upstream sequence. *The EMBO journal*, 29(9):1523–1536, 2010.

[91] C. O'Dushlaine, E. Kenny, E. A. Heron, R. Segurado, M. Gill, D. W. Morris, and A. Corvin. The snp ratio test: pathway analysis of genome-wide association datasets. *Bioinformatics*, 25(20):2762–2763, 2009.

[92] F. Ozsolak, P. Kapranov, S. Foissac, S. W. Kim, E. Fishilevich, A. P. Monaghan, B. John, and P. M. Milos. Comprehensive polyadenylation site maps in yeast and human reveal pervasive alternative polyadenylation. *Cell*, 143(6):1018–1029, 2010.

[93] E. Pauws, A. Van Kampen, S. Van de Graaf, J. De Vijlder, and C. Ris-Stalpers. Heterogeneity in polyadenylation cleavage sites in mammalian mrna sequences: implications for sage analysis. *Nucleic acids research*, 29(8):1690–1694, 2001.

[94] G. Peng, L. Luo, H. Siu, Y. Zhu, P. Hu, S. Hong, J. Zhao, X. Zhou, J. D. Reveille, L. Jin, et al. Gene and pathway-based second-wave analysis of genome-wide association studies. *European Journal of Human Genetics*, 18(1):111–117, 2009.

[95] K. S. Pollard, M. J. Hubisz, K. R. Rosenbloom, and A. Siepel. Detection of nonneutral substitution rates on mammalian phylogenies. *Genome research*, 20(1):110–121, 2010.

[96] S. Pounds and C. Cheng. Robust estimation of the false discovery rate. *Bioinformatics*, 22(16):1979–1987, 2006.

[97] S. Purcell, B. Neale, K. Todd-Brown, L. Thomas, M. A. Ferreira, D. Bender, J. Maller, P. Sklar, P. I. De Bakker, M. J. Daly, et al. Plink: a tool set for whole-genome association and population-based linkage analyses. *The American Journal of Human Genetics*, 81(3):559–575, 2007.

[98] J. Rogers, P. Early, C. Carter, K. Calame, M. Bond, L. Hood, and R. Wall. Two mrnas with different 3 ends encode membrane-bound and secreted forms of immunoglobulin μ chain. *Cell*, 20(2):303–312, 1980.

[99] S. Rumble, P. Lacroute, A. Dalca, M. Fiume, A. Sidow, and M. Brudno. Shrimp: accurate mapping of short color-space reads. *PLoS computational biology*, 5(5):e1000386, 2009.

[100] R. Sandberg, J. R. Neilson, A. Sarma, P. A. Sharp, and C. B. Burge. Proliferating cells express mrnas with shortened 3'untranslated regions and fewer microrna target sites. *Science*, 320(5883):1643–1647, 2008.

[101] J. Schug, W.-P. Schuller, C. Kappen, J. M. Salbaum, M. Bucan, and C. J. Stoeckert. Promoter features related to tissue specificity as measured by shannon entropy. *Genome biology*, 6(4), 2005.

[102] A. X. L. K. Shen and E. Torng. Large scale hamming distance query processing. In *ICDE*, 2011.

[103] P. J. Shepard, E.-A. Choi, J. Lu, L. A. Flanagan, K. J. Hertel, and Y. Shi. Complex and dynamic landscape of rna polyadenylation revealed by pas-seq. *Rna*, 17(4):761–772, 2011.

[104] Y. Shi. Alternative polyadenylation: new insights from global analyses. *Rna*, 18(12):2105–2117, 2012.

[105] A. Siepel, G. Bejerano, J. S. Pedersen, A. S. Hinrichs, M. Hou, K. Rosenbloom, H. Clawson, J. Spieth, L. W. Hillier, S. Richards, et al. Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. *Genome research*, 15(8):1034–1050, 2005.

[106] M. K. Skinner, M. D. Anway, M. I. Savenkova, A. C. Gore, and D. Crews. Transgenerational epigenetic programming of the brain transcriptome and anxiety behavior. *PloS one*, 3(11):e3745, 2008.

[107] R. Sladek, G. Rocheleau, J. Rung, C. Dina, L. Shen, D. Serre, P. Boutin, D. Vincent, A. Belisle, S. Hadjadj, et al. A genome-wide association study identifies novel risk loci for type 2 diabetes. *Nature*, 445(7130):881–885, 2007.

[108] A. Smith, Z. Xuan, and M. Zhang. Using quality scores and longer reads improves accuracy of solexa read mapping. *BMC bioinformatics*, 9(1):128, 2008.

[109] N. Spies, C. B. Nielsen, R. A. Padgett, and C. B. Burge. Biased chromatin signatures around polyadenylation sites and exons. *Molecular cell*, 36(2):245–254, 2009.

[110] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[111] D. C. Thomas. The need for a systematic approach to complex pathways in molecular epidemiology. *Cancer Epidemiology Biomarkers & Prevention*, 14(3):557–559, 2005.

[112] B. Tian, J. Hu, H. Zhang, and C. S. Lutz. A large-scale analysis of mrna polyadenylation of human and mouse genes. *Nucleic acids research*, 33(1):201–212, 2005.

[113] S. A. Tishkoff, E. Dietzsch, W. Speed, A. J. Pakstis, J. R. Kidd, K. Cheung, B. Bonne-Tamir, A. S. Santachiara-Benerecetti, P. Moral, M. Krings, et al. Global patterns of linkage disequilibrium at the cd4 locus and modern human origins. *Science*, 271(5254):1380–1387, 1996.

[114] A. Torkamani, E. J. Topol, and N. J. Schork. Pathway analysis of seven common diseases assessed by genome-wide association. *Genomics*, 92(5):265–272, 2008.

[115] E. Ukkonen. Approximae string matching with q-grams and maximal matching. *Theor. Comut. Sci.*, 1:191–211, 1992.

[116] K. Venkataraman, K. M. Brown, and G. M. Gilmartin. Analysis of a noncanonical poly (a) site reveals a tripartite mechanism for vertebrate poly (a) site recognition. *Genes & development*, 19(11):1315–1327, 2005.

[117] T. Walsh, J. M. McClellan, S. E. McCarthy, A. M. Addington, S. B. Pierce, G. M. Cooper, A. S. Nord, M. Kusenda, D. Malhotra, A. Bhandari, et al. Rare structural variants disrupt multiple genes in neurodevelopmental pathways in schizophrenia. *science*, 320(5875):539–543, 2008.

[118] E. T. Wang, R. Sandberg, S. Luo, I. Khrebtukova, L. Zhang, C. Mayr, S. F. Kingsmore, G. P. Schroth, and C. B. Burge. Alternative isoform regulation in human tissue transcriptomes. *Nature*, 456(7221):470–476, 2008.

[119] K. Wang, M. Li, and M. Bucan. Pathway-based approaches for analysis of genomewide association studies. *The American Journal of Human Genetics*, 81(6):1278–1283, 2007.

[120] D. Weese, A.-K. Emde, T. Rausch, A. Döring, and K. Reinert. Razers-fast read mapping with sensitivity control. *Genome Research*, 19:1646–1654, 2009.

[121] D. R. Weinberger, E. Cannon-Spoor, S. G. Potkin, and R. J. Wyatt. Poor premorbid adjustment and ct scan abnormalities in chronic schizophrenia. *The American journal of psychiatry*, 1980.

[122] L. Weng, F. Macciardi, A. Subramanian, G. Guffanti, S. G. Potkin, Z. Yu, and X. Xie. Snp-based pathway enrichment analysis for genome-wide association studies. *BMC bioinformatics*, 12(1):99, 2011.

[123] C. Wu and J. C. Alwine. Secondary structure as a functional feature in the downstream region of mammalian polyadenylation signals. *Molecular and cellular biology*, 24(7):2789–2796, 2004.

[124] L. Xi, Y. Fondufe-Mittendorf, L. Xia, J. Flatow, J. Widom, and J.-P. Wang. Predicting nucleosome positioning using a duration hidden markov model. *BMC bioinformatics*, 11(1):346, 2010.

[125] C. Xiao, W. Wang, and X. Lin. Ed-join: An efficient algorithm for similarity joins with edit distance constraints. In *VLDB*, 2008.

[126] B. Xie, B. R. Jankovic, V. B. Bajic, L. Song, and X. Gao. Poly (a) motif prediction using spectral latent features from human dna sequences. *Bioinformatics*, 29(13):i316–i325, 2013.

[127] J. Yan and T. G. Marr. Computational analysis of 3-ends of ests shows four classes of alternative polyadenylation in human, mouse, and rat. *Genome research*, 15(3):369–375, 2005.

[128] C. Yao, J. Biesinger, J. Wan, L. Weng, Y. Xing, X. Xie, and Y. Shi. Transcriptome-wide analyses of cstf64–rna interactions in global regulation of mrna alternative polyadenylation. *Proceedings of the National Academy of Sciences*, 109(46):18773–18778, 2012.

[129] C. Yao, E.-A. Choi, L. Weng, X. Xie, J. Wan, Y. Xing, J. J. Moresco, P. G. Tu, J. R. Yates, and Y. Shi. Overlapping and distinct functions of cstf64 and cstf64τ in mammalian mrna 3 processing. *rna*, 19(12):1781–1790, 2013.

[130] K. Yu, Q. Li, A. W. Bergen, R. M. Pfeiffer, P. S. Rosenberg, N. Caporaso, P. Kraft, and N. Chatterjee. Pathway analysis by adaptive combination of p-values. *Genetic epidemiology*, 33(8):700–709, 2009.

[131] E. Zeggini, M. N. Weedon, C. M. Lindgren, T. M. Frayling, K. S. Elliott, H. Lango, N. J. Timpson, J. R. Perry, N. W. Rayner, R. M. Freathy, et al. Replication of genome-wide association signals in uk samples reveals risk loci for type 2 diabetes. *Science*, 316(5829):1336–1341, 2007.

[132] J. Zhao, L. Hyman, and C. Moore. Formation of mrna 3 ends in eukaryotes: mechanism, regulation, and interrelationships with other steps in mrna synthesis. *Microbiology and Molecular Biology Reviews*, 63(2):405–445, 1999.

[133] H. Zhong, X. Yang, L. M. Kaplan, C. Molony, and E. E. Schadt. Integrating pathway analysis and genetics of gene expression for genome-wide association studies. *The American Journal of Human Genetics*, 86(4):581–591, 2010.

Appendix A

A complete list of 658 RNA features

AcompleteListofRNAfeatures.csv

Index	Feature Name	Data Type	Feature Category
1	phylopcons_subregion1	real	Conservation level
2	phylopcons_subregion2	real	Conservation level
3	phylopcons_subregion3	real	Conservation level
4	phylopcons_subregion4	real	Conservation level
5	phast_cons_subregion1	real	Conservation level
6	phast_cons_subregion2	real	Conservation level
7	phast_cons_subregion3	real	Conservation level
8	phast_cons_subregion4	real	Conservation level
9	nucleosome_occp_up_mean	real	Nuclosome positioning
10	nucleosome_occp_up_max	real	Nuclosome positioning
11	nucleosome_occp_down_mean	real	Nuclosome positioning
12	nucleosome_occp_down_max	real	Nuclosome positioning
13	secstr_5mer_accessibility_mean_subregion1	real	Secondary structure
14	secstr_5mer_accessibility_max_subregion1	real	Secondary structure
15	secstr_5mer_accessibility_mean_subregion2	real	Secondary structure
16	secstr_5mer_accessibility_max_subregion2	real	Secondary structure
17	secstr_5mer_accessibility_mean_subregion3	real	Secondary structure
18	secstr_5mer_accessibility_max_subregion3	real	Secondary structure
19	secstr_5mer_accessibility_mean_subregion4	real	Secondary structure
20	secstr_5mer_accessibility_max_subregion4	real	Secondary structure
21	indicator_distal PAS	binary	Transcript structure
22	indicator_proximal PAS	binary	Transcript structure
23	indicator_middle PAS	binary	Transcript structure
24	sameexon_withLPAS	binary	Transcript structure
25	diffexon_withLPAS	binary	Transcript structure
26	sameexon_withRPAS	binary	Transcript structure
27	diffexon_withRPAS	binary	Transcript structure
28	Distance to previous poly(A) site	real	Transcript structure
29	Distance to next poly(A) site	real	Transcript structure
30	Distance to closest stop codon	real	Transcript structure
31	Diatance to closest to 3' splce site (if poly(A) in exon)	real	Transcript structure
32	Diatance to closest to 5' splce site (if poly(A) in intron)	real	Transcript structure
33	shortseq AAA frequency subregion1	integer	Short 3mer motif
34	shortseq AAC frequency subregion1	integer	Short 3mer motif
35	shortseq AAG frequency subregion1	integer	Short 3mer motif
36	shortseq AAT frequency subregion1	integer	Short 3mer motif
37	shortseq ACA frequency subregion1	integer	Short 3mer motif
38	shortseq ACC frequency subregion1	integer	Short 3mer motif
39	shortseq ACG frequency subregion1	integer	Short 3mer motif
40	shortseq ACT frequency subregion1	integer	Short 3mer motif
41	shortseq AGA frequency subregion1	integer	Short 3mer motif
42	shortseq AGC frequency subregion1	integer	Short 3mer motif
43	shortseq AGG frequency subregion1	integer	Short 3mer motif
44	shortseq AGT frequency subregion1	integer	Short 3mer motif
45	shortseq ATA frequency subregion1	integer	Short 3mer motif
46	shortseq ATC frequency subregion1	integer	Short 3mer motif
47	shortseq ATG frequency subregion1	integer	Short 3mer motif
48	shortseq ATT frequency subregion1	integer	Short 3mer motif
49	shortseq CAA frequency subregion1	integer	Short 3mer motif
50	shortseq CAC frequency subregion1	integer	Short 3mer motif
51	shortseq CAG frequency subregion1	integer	Short 3mer motif

52	shortseq CAT frequency subregion1	integer	Short 3mer motif
53	shortseq CCA frequency subregion1	integer	Short 3mer motif
54	shortseq CCC frequency subregion1	integer	Short 3mer motif
55	shortseq CCG frequency subregion1	integer	Short 3mer motif
56	shortseq CCT frequency subregion1	integer	Short 3mer motif
57	shortseq CGA frequency subregion1	integer	Short 3mer motif
58	shortseq CGC frequency subregion1	integer	Short 3mer motif
59	shortseq CGG frequency subregion1	integer	Short 3mer motif
60	shortseq CGT frequency subregion1	integer	Short 3mer motif
61	shortseq CTA frequency subregion1	integer	Short 3mer motif
62	shortseq CTC frequency subregion1	integer	Short 3mer motif
63	shortseq CTG frequency subregion1	integer	Short 3mer motif
64	shortseq CTT frequency subregion1	integer	Short 3mer motif
65	shortseq GAA frequency subregion1	integer	Short 3mer motif
66	shortseq GAC frequency subregion1	integer	Short 3mer motif
67	shortseq GAG frequency subregion1	integer	Short 3mer motif
68	shortseq GAT frequency subregion1	integer	Short 3mer motif
69	shortseq GCA frequency subregion1	integer	Short 3mer motif
70	shortseq GCC frequency subregion1	integer	Short 3mer motif
71	shortseq GCG frequency subregion1	integer	Short 3mer motif
72	shortseq GCT frequency subregion1	integer	Short 3mer motif
73	shortseq GGA frequency subregion1	integer	Short 3mer motif
74	shortseq GGC frequency subregion1	integer	Short 3mer motif
75	shortseq GGG frequency subregion1	integer	Short 3mer motif
76	shortseq GGT frequency subregion1	integer	Short 3mer motif
77	shortseq GTA frequency subregion1	integer	Short 3mer motif
78	shortseq GTC frequency subregion1	integer	Short 3mer motif
79	shortseq GTG frequency subregion1	integer	Short 3mer motif
80	shortseq GTT frequency subregion1	integer	Short 3mer motif
81	shortseq TAA frequency subregion1	integer	Short 3mer motif
82	shortseq TAC frequency subregion1	integer	Short 3mer motif
83	shortseq TAG frequency subregion1	integer	Short 3mer motif
84	shortseq TAT frequency subregion1	integer	Short 3mer motif
85	shortseq TCA frequency subregion1	integer	Short 3mer motif
86	shortseq TCC frequency subregion1	integer	Short 3mer motif
87	shortseq TCG frequency subregion1	integer	Short 3mer motif
88	shortseq TCT frequency subregion1	integer	Short 3mer motif
89	shortseq TGA frequency subregion1	integer	Short 3mer motif
90	shortseq TGC frequency subregion1	integer	Short 3mer motif
91	shortseq TGG frequency subregion1	integer	Short 3mer motif
92	shortseq TGT frequency subregion1	integer	Short 3mer motif
93	shortseq TTA frequency subregion1	integer	Short 3mer motif
94	shortseq TTC frequency subregion1	integer	Short 3mer motif
95	shortseq TTG frequency subregion1	integer	Short 3mer motif
96	shortseq TTT frequency subregion1	integer	Short 3mer motif
97	shortseq AAA frequency subregion2	integer	Short 3mer motif
98	shortseq AAC frequency subregion2	integer	Short 3mer motif
99	shortseq AAG frequency subregion2	integer	Short 3mer motif
100	shortseq AAT frequency subregion2	integer	Short 3mer motif
101	shortseq ACA frequency subregion2	integer	Short 3mer motif
102	shortseq ACC frequency subregion2	integer	Short 3mer motif
103	shortseq ACG frequency subregion2	integer	Short 3mer motif
104	shortseq ACT frequency subregion2	integer	Short 3mer motif

105	shortseq AGA frequency subregion2	integer	Short 3mer motif
106	shortseq AGC frequency subregion2	integer	Short 3mer motif
107	shortseq AGG frequency subregion2	integer	Short 3mer motif
108	shortseq AGT frequency subregion2	integer	Short 3mer motif
109	shortseq ATA frequency subregion2	integer	Short 3mer motif
110	shortseq ATC frequency subregion2	integer	Short 3mer motif
111	shortseq ATG frequency subregion2	integer	Short 3mer motif
112	shortseq ATT frequency subregion2	integer	Short 3mer motif
113	shortseq CAA frequency subregion2	integer	Short 3mer motif
114	shortseq CAC frequency subregion2	integer	Short 3mer motif
115	shortseq CAG frequency subregion2	integer	Short 3mer motif
116	shortseq CAT frequency subregion2	integer	Short 3mer motif
117	shortseq CCA frequency subregion2	integer	Short 3mer motif
118	shortseq CCC frequency subregion2	integer	Short 3mer motif
119	shortseq CCG frequency subregion2	integer	Short 3mer motif
120	shortseq CCT frequency subregion2	integer	Short 3mer motif
121	shortseq CGA frequency subregion2	integer	Short 3mer motif
122	shortseq CGC frequency subregion2	integer	Short 3mer motif
123	shortseq CGG frequency subregion2	integer	Short 3mer motif
124	shortseq CGT frequency subregion2	integer	Short 3mer motif
125	shortseq CTA frequency subregion2	integer	Short 3mer motif
126	shortseq CTC frequency subregion2	integer	Short 3mer motif
127	shortseq CTG frequency subregion2	integer	Short 3mer motif
128	shortseq CTT frequency subregion2	integer	Short 3mer motif
129	shortseq GAA frequency subregion2	integer	Short 3mer motif
130	shortseq GAC frequency subregion2	integer	Short 3mer motif
131	shortseq GAG frequency subregion2	integer	Short 3mer motif
132	shortseq GAT frequency subregion2	integer	Short 3mer motif
133	shortseq GCA frequency subregion2	integer	Short 3mer motif
134	shortseq GCC frequency subregion2	integer	Short 3mer motif
135	shortseq GCG frequency subregion2	integer	Short 3mer motif
136	shortseq GCT frequency subregion2	integer	Short 3mer motif
137	shortseq GGA frequency subregion2	integer	Short 3mer motif
138	shortseq GGC frequency subregion2	integer	Short 3mer motif
139	shortseq GGG frequency subregion2	integer	Short 3mer motif
140	shortseq GGT frequency subregion2	integer	Short 3mer motif
141	shortseq GTA frequency subregion2	integer	Short 3mer motif
142	shortseq GTC frequency subregion2	integer	Short 3mer motif
143	shortseq GTG frequency subregion2	integer	Short 3mer motif
144	shortseq GTT frequency subregion2	integer	Short 3mer motif
145	shortseq TAA frequency subregion2	integer	Short 3mer motif
146	shortseq TAC frequency subregion2	integer	Short 3mer motif
147	shortseq TAG frequency subregion2	integer	Short 3mer motif
148	shortseq TAT frequency subregion2	integer	Short 3mer motif
149	shortseq TCA frequency subregion2	integer	Short 3mer motif
150	shortseq TCC frequency subregion2	integer	Short 3mer motif
151	shortseq TCG frequency subregion2	integer	Short 3mer motif
152	shortseq TCT frequency subregion2	integer	Short 3mer motif
153	shortseq TGA frequency subregion2	integer	Short 3mer motif
154	shortseq TGC frequency subregion2	integer	Short 3mer motif
155	shortseq TGG frequency subregion2	integer	Short 3mer motif
156	shortseq TGT frequency subregion2	integer	Short 3mer motif
157	shortseq TTA frequency subregion2	integer	Short 3mer motif

158	shortseq TTC frequency subregion2	integer	Short 3mer motif
159	shortseq TTG frequency subregion2	integer	Short 3mer motif
160	shortseq TTT frequency subregion2	integer	Short 3mer motif
161	shortseq AAA frequency subregion3	integer	Short 3mer motif
162	shortseq AAC frequency subregion3	integer	Short 3mer motif
163	shortseq AAG frequency subregion3	integer	Short 3mer motif
164	shortseq AAT frequency subregion3	integer	Short 3mer motif
165	shortseq ACA frequency subregion3	integer	Short 3mer motif
166	shortseq ACC frequency subregion3	integer	Short 3mer motif
167	shortseq ACG frequency subregion3	integer	Short 3mer motif
168	shortseq ACT frequency subregion3	integer	Short 3mer motif
169	shortseq AGA frequency subregion3	integer	Short 3mer motif
170	shortseq AGC frequency subregion3	integer	Short 3mer motif
171	shortseq AGG frequency subregion3	integer	Short 3mer motif
172	shortseq AGT frequency subregion3	integer	Short 3mer motif
173	shortseq ATA frequency subregion3	integer	Short 3mer motif
174	shortseq ATC frequency subregion3	integer	Short 3mer motif
175	shortseq ATG frequency subregion3	integer	Short 3mer motif
176	shortseq ATT frequency subregion3	integer	Short 3mer motif
177	shortseq CAA frequency subregion3	integer	Short 3mer motif
178	shortseq CAC frequency subregion3	integer	Short 3mer motif
179	shortseq CAG frequency subregion3	integer	Short 3mer motif
180	shortseq CAT frequency subregion3	integer	Short 3mer motif
181	shortseq CCA frequency subregion3	integer	Short 3mer motif
182	shortseq CCC frequency subregion3	integer	Short 3mer motif
183	shortseq CCG frequency subregion3	integer	Short 3mer motif
184	shortseq CCT frequency subregion3	integer	Short 3mer motif
185	shortseq CGA frequency subregion3	integer	Short 3mer motif
186	shortseq CGC frequency subregion3	integer	Short 3mer motif
187	shortseq CGG frequency subregion3	integer	Short 3mer motif
188	shortseq CGT frequency subregion3	integer	Short 3mer motif
189	shortseq CTA frequency subregion3	integer	Short 3mer motif
190	shortseq CTC frequency subregion3	integer	Short 3mer motif
191	shortseq CTG frequency subregion3	integer	Short 3mer motif
192	shortseq CTT frequency subregion3	integer	Short 3mer motif
193	shortseq GAA frequency subregion3	integer	Short 3mer motif
194	shortseq GAC frequency subregion3	integer	Short 3mer motif
195	shortseq GAG frequency subregion3	integer	Short 3mer motif
196	shortseq GAT frequency subregion3	integer	Short 3mer motif
197	shortseq GCA frequency subregion3	integer	Short 3mer motif
198	shortseq GCC frequency subregion3	integer	Short 3mer motif
199	shortseq GCG frequency subregion3	integer	Short 3mer motif
200	shortseq GCT frequency subregion3	integer	Short 3mer motif
201	shortseq GGA frequency subregion3	integer	Short 3mer motif
202	shortseq GGC frequency subregion3	integer	Short 3mer motif
203	shortseq GGG frequency subregion3	integer	Short 3mer motif
204	shortseq GGT frequency subregion3	integer	Short 3mer motif
205	shortseq GTA frequency subregion3	integer	Short 3mer motif
206	shortseq GTC frequency subregion3	integer	Short 3mer motif
207	shortseq GTG frequency subregion3	integer	Short 3mer motif
208	shortseq GTT frequency subregion3	integer	Short 3mer motif
209	shortseq TAA frequency subregion3	integer	Short 3mer motif
210	shortseq TAC frequency subregion3	integer	Short 3mer motif

211	shortseq TAG frequency subregion3	integer	Short 3mer motif
212	shortseq TAT frequency subregion3	integer	Short 3mer motif
213	shortseq TCA frequency subregion3	integer	Short 3mer motif
214	shortseq TCC frequency subregion3	integer	Short 3mer motif
215	shortseq TCG frequency subregion3	integer	Short 3mer motif
216	shortseq TCT frequency subregion3	integer	Short 3mer motif
217	shortseq TGA frequency subregion3	integer	Short 3mer motif
218	shortseq TGC frequency subregion3	integer	Short 3mer motif
219	shortseq TGG frequency subregion3	integer	Short 3mer motif
220	shortseq TGT frequency subregion3	integer	Short 3mer motif
221	shortseq TTA frequency subregion3	integer	Short 3mer motif
222	shortseq TTC frequency subregion3	integer	Short 3mer motif
223	shortseq TTG frequency subregion3	integer	Short 3mer motif
224	shortseq TTT frequency subregion3	integer	Short 3mer motif
225	shortseq AAA frequency subregion4	integer	Short 3mer motif
226	shortseq AAC frequency subregion4	integer	Short 3mer motif
227	shortseq AAG frequency subregion4	integer	Short 3mer motif
228	shortseq AAT frequency subregion4	integer	Short 3mer motif
229	shortseq ACA frequency subregion4	integer	Short 3mer motif
230	shortseq ACC frequency subregion4	integer	Short 3mer motif
231	shortseq ACG frequency subregion4	integer	Short 3mer motif
232	shortseq ACT frequency subregion4	integer	Short 3mer motif
233	shortseq AGA frequency subregion4	integer	Short 3mer motif
234	shortseq AGC frequency subregion4	integer	Short 3mer motif
235	shortseq AGG frequency subregion4	integer	Short 3mer motif
236	shortseq AGT frequency subregion4	integer	Short 3mer motif
237	shortseq ATA frequency subregion4	integer	Short 3mer motif
238	shortseq ATC frequency subregion4	integer	Short 3mer motif
239	shortseq ATG frequency subregion4	integer	Short 3mer motif
240	shortseq ATT frequency subregion4	integer	Short 3mer motif
241	shortseq CAA frequency subregion4	integer	Short 3mer motif
242	shortseq CAC frequency subregion4	integer	Short 3mer motif
243	shortseq CAG frequency subregion4	integer	Short 3mer motif
244	shortseq CAT frequency subregion4	integer	Short 3mer motif
245	shortseq CCA frequency subregion4	integer	Short 3mer motif
246	shortseq CCC frequency subregion4	integer	Short 3mer motif
247	shortseq CCG frequency subregion4	integer	Short 3mer motif
248	shortseq CCT frequency subregion4	integer	Short 3mer motif
249	shortseq CGA frequency subregion4	integer	Short 3mer motif
250	shortseq CGC frequency subregion4	integer	Short 3mer motif
251	shortseq CGG frequency subregion4	integer	Short 3mer motif
252	shortseq CGT frequency subregion4	integer	Short 3mer motif
253	shortseq CTA frequency subregion4	integer	Short 3mer motif
254	shortseq CTC frequency subregion4	integer	Short 3mer motif
255	shortseq CTG frequency subregion4	integer	Short 3mer motif
256	shortseq CTT frequency subregion4	integer	Short 3mer motif
257	shortseq GAA frequency subregion4	integer	Short 3mer motif
258	shortseq GAC frequency subregion4	integer	Short 3mer motif
259	shortseq GAG frequency subregion4	integer	Short 3mer motif
260	shortseq GAT frequency subregion4	integer	Short 3mer motif
261	shortseq GCA frequency subregion4	integer	Short 3mer motif
262	shortseq GCC frequency subregion4	integer	Short 3mer motif
263	shortseq GCG frequency subregion4	integer	Short 3mer motif

264	shortseq GCT frequency subregion4	integer	Short 3mer motif
265	shortseq GGA frequency subregion4	integer	Short 3mer motif
266	shortseq GGC frequency subregion4	integer	Short 3mer motif
267	shortseq GGG frequency subregion4	integer	Short 3mer motif
268	shortseq GGT frequency subregion4	integer	Short 3mer motif
269	shortseq GTA frequency subregion4	integer	Short 3mer motif
270	shortseq GTC frequency subregion4	integer	Short 3mer motif
271	shortseq GTG frequency subregion4	integer	Short 3mer motif
272	shortseq GTT frequency subregion4	integer	Short 3mer motif
273	shortseq TAA frequency subregion4	integer	Short 3mer motif
274	shortseq TAC frequency subregion4	integer	Short 3mer motif
275	shortseq TAG frequency subregion4	integer	Short 3mer motif
276	shortseq TAT frequency subregion4	integer	Short 3mer motif
277	shortseq TCA frequency subregion4	integer	Short 3mer motif
278	shortseq TCC frequency subregion4	integer	Short 3mer motif
279	shortseq TCG frequency subregion4	integer	Short 3mer motif
280	shortseq TCT frequency subregion4	integer	Short 3mer motif
281	shortseq TGA frequency subregion4	integer	Short 3mer motif
282	shortseq TGC frequency subregion4	integer	Short 3mer motif
283	shortseq TGG frequency subregion4	integer	Short 3mer motif
284	shortseq TGT frequency subregion4	integer	Short 3mer motif
285	shortseq TTA frequency subregion4	integer	Short 3mer motif
286	shortseq TTC frequency subregion4	integer	Short 3mer motif
287	shortseq TTG frequency subregion4	integer	Short 3mer motif
288	shortseq TTT frequency subregion4	integer	Short 3mer motif
289	AATAAA frequency subregion1	integer	PAS signal & variants
290	ATTAAA frequency subregion1	integer	PAS signal & variants
291	AAGAAA frequency subregion1	integer	PAS signal & variants
292	AAAAAG frequency subregion1	integer	PAS signal & variants
293	AATACA frequency subregion1	integer	PAS signal & variants
294	TATAAA frequency subregion1	integer	PAS signal & variants
295	ACTAAA frequency subregion1	integer	PAS signal & variants
296	AGTAAA frequency subregion1	integer	PAS signal & variants
297	GATAAA frequency subregion1	integer	PAS signal & variants
298	AATATA frequency subregion1	integer	PAS signal & variants
299	CATAAA frequency subregion1	integer	PAS signal & variants
300	AATAGA frequency subregion1	integer	PAS signal & variants
301	AATAAA frequency subregion2	integer	PAS signal & variants
302	ATTAAA frequency subregion2	integer	PAS signal & variants
303	AAGAAA frequency subregion2	integer	PAS signal & variants
304	AAAAAG frequency subregion2	integer	PAS signal & variants
305	AATACA frequency subregion2	integer	PAS signal & variants
306	TATAAA frequency subregion2	integer	PAS signal & variants
307	ACTAAA frequency subregion2	integer	PAS signal & variants
308	AGTAAA frequency subregion2	integer	PAS signal & variants
309	GATAAA frequency subregion2	integer	PAS signal & variants
310	AATATA frequency subregion2	integer	PAS signal & variants
311	CATAAA frequency subregion2	integer	PAS signal & variants
312	AATAGA frequency subregion2	integer	PAS signal & variants
313	AATAAA subregion1 weighted	real	PAS signal & variants
314	ATTAAA subregion1 weighted	real	PAS signal & variants
315	AAGAAA subregion1 weighted	real	PAS signal & variants
316	AAAAAG subregion1 weighted	real	PAS signal & variants

317	AATACA subregion1 weighted	real	PAS signal & variants
318	TATAAA subregion1 weighted	real	PAS signal & variants
319	ACTAAA subregion1 weighted	real	PAS signal & variants
320	AGTAAA subregion1 weighted	real	PAS signal & variants
321	GATAAA subregion1 weighted	real	PAS signal & variants
322	AATATA subregion1 weighted	real	PAS signal & variants
323	CATAAA subregion1 weighted	real	PAS signal & variants
324	AATAGA subregion1 weighted	real	PAS signal & variants
325	AATAAA subregion2 weighted	real	PAS signal & variants
326	ATTAAA subregion2 weighted	real	PAS signal & variants
327	AAGAAA subregion2 weighted	real	PAS signal & variants
328	AAAAAG subregion2 weighted	real	PAS signal & variants
329	AATACA subregion2 weighted	real	PAS signal & variants
330	TATAAA subregion2 weighted	real	PAS signal & variants
331	ACTAAA subregion2 weighted	real	PAS signal & variants
332	AGTAAA subregion2 weighted	real	PAS signal & variants
333	GATAAA subregion2 weighted	real	PAS signal & variants
334	AATATA subregion2 weighted	real	PAS signal & variants
335	CATAAA subregion2 weighted	real	PAS signal & variants
336	AATAGA subregion2 weighted	real	PAS signal & variants
337	CFI subregion1	real	Known regulators
338	PTBP1 subregion1	real	Known regulators
339	NOVA1,NOVA2 subregion1	real	Known regulators
340	hnRNPF subregion1	real	Known regulators
341	PCBP1,PCBP2 subregion1	real	Known regulators
342	ESRP2 subregion1	real	Known regulators
343	PABPN1 subregion1	real	Known regulators
344	SFRS1 subregion1	real	Known regulators
345	CstF64 subregion1	real	Known regulators
346	U-rich subregion1	real	Known regulators
347	CFI subregion2	real	Known regulators
348	PTBP1 subregion2	real	Known regulators
349	NOVA1,NOVA2 subregion2	real	Known regulators
350	hnRNPF subregion2	real	Known regulators
351	PCBP1,PCBP2 subregion2	real	Known regulators
352	ESRP2 subregion2	real	Known regulators
353	PABPN1 subregion2	real	Known regulators
354	SFRS1 subregion2	real	Known regulators
355	CstF64 subregion2	real	Known regulators
356	U-rich subregion2	real	Known regulators
357	CFI subregion3	real	Known regulators
358	PTBP1 subregion3	real	Known regulators
359	NOVA1,NOVA2 subregion3	real	Known regulators
360	hnRNPF subregion3	real	Known regulators
361	PCBP1,PCBP2 subregion3	real	Known regulators
362	ESRP2 subregion3	real	Known regulators
363	PABPN1 subregion3	real	Known regulators
364	SFRS1 subregion3	real	Known regulators
365	CstF64 subregion3	real	Known regulators
366	U-rich subregion3	real	Known regulators
367	CFI subregion4	real	Known regulators
368	PTBP1 subregion4	real	Known regulators
369	NOVA1,NOVA2 subregion4	real	Known regulators

370	hnRNPF subregion4	real	Known regulators
371	PCBP1,PCBP2 subregion4	real	Known regulators
372	ESRP2 subregion4	real	Known regulators
373	PABPN1 subregion4	real	Known regulators
374	SFRS1 subregion4	real	Known regulators
375	CstF64 subregion4	real	Known regulators
376	U-rich subregion4	real	Known regulators
377	AAACG frequency up	integer	potential unknown motifs
378	AAAGC frequency up	integer	potential unknown motifs
379	AAAGT frequency up	integer	potential unknown motifs
380	AAATC frequency up	integer	potential unknown motifs
381	AAATG frequency up	integer	potential unknown motifs
382	AACCT frequency up	integer	potential unknown motifs
383	AAGTT frequency up	integer	potential unknown motifs
384	AATAA frequency up	integer	potential unknown motifs
385	AATGT frequency up	integer	potential unknown motifs
386	AGTAA frequency up	integer	potential unknown motifs
387	ATAAA frequency up	integer	potential unknown motifs
388	ATATG frequency up	integer	potential unknown motifs
389	ATGTA frequency up	integer	potential unknown motifs
390	ATGTG frequency up	integer	potential unknown motifs
391	ATGTT frequency up	integer	potential unknown motifs
392	ATTAA frequency up	integer	potential unknown motifs
393	ATTAT frequency up	integer	potential unknown motifs
394	ATTGT frequency up	integer	potential unknown motifs
395	CAAAC frequency up	integer	potential unknown motifs
396	CAATA frequency up	integer	potential unknown motifs
397	CATTA frequency up	integer	potential unknown motifs
398	CCAAT frequency up	integer	potential unknown motifs
399	CCCAC frequency up	integer	potential unknown motifs
400	CCCCA frequency up	integer	potential unknown motifs
401	CGTGA frequency up	integer	potential unknown motifs
402	CTGTG frequency up	integer	potential unknown motifs
403	CTGTT frequency up	integer	potential unknown motifs
404	CTTTG frequency up	integer	potential unknown motifs
405	GAAAT frequency up	integer	potential unknown motifs
406	GATTT frequency up	integer	potential unknown motifs
407	GCAAA frequency up	integer	potential unknown motifs
408	GCTGT frequency up	integer	potential unknown motifs
409	GTAAA frequency up	integer	potential unknown motifs
410	GTCAA frequency up	integer	potential unknown motifs
411	GTTCA frequency up	integer	potential unknown motifs
412	GTTGA frequency up	integer	potential unknown motifs
413	GTTTC frequency up	integer	potential unknown motifs
414	GTTTT frequency up	integer	potential unknown motifs
415	TAAAA frequency up	integer	potential unknown motifs
416	TAAAC frequency up	integer	potential unknown motifs
417	TAAAG frequency up	integer	potential unknown motifs
418	TAATA frequency up	integer	potential unknown motifs
419	TACAA frequency up	integer	potential unknown motifs
420	TACAT frequency up	integer	potential unknown motifs
421	TATTG frequency up	integer	potential unknown motifs
422	TCAAT frequency up	integer	potential unknown motifs

423	TCTAT frequency up	integer	potential unknown motifs
424	TCTGT frequency up	integer	potential unknown motifs
425	TGAAA frequency up	integer	potential unknown motifs
426	TGAAT frequency up	integer	potential unknown motifs
427	TGACT frequency up	integer	potential unknown motifs
428	TGCTT frequency up	integer	potential unknown motifs
429	TGGAA frequency up	integer	potential unknown motifs
430	TGTAA frequency up	integer	potential unknown motifs
431	TGTAC frequency up	integer	potential unknown motifs
432	TGTAT frequency up	integer	potential unknown motifs
433	TGTCA frequency up	integer	potential unknown motifs
434	TGTGA frequency up	integer	potential unknown motifs
435	TGTTA frequency up	integer	potential unknown motifs
436	TGTTT frequency up	integer	potential unknown motifs
437	TTAAA frequency up	integer	potential unknown motifs
438	TTATA frequency up	integer	potential unknown motifs
439	TTATG frequency up	integer	potential unknown motifs
440	TTCTA frequency up	integer	potential unknown motifs
441	TTCTG frequency up	integer	potential unknown motifs
442	TTGTA frequency up	integer	potential unknown motifs
443	TTGTC frequency up	integer	potential unknown motifs
444	TTGTG frequency up	integer	potential unknown motifs
445	TTGTT frequency up	integer	potential unknown motifs
446	TTTAC frequency up	integer	potential unknown motifs
447	TTTAT frequency up	integer	potential unknown motifs
448	TTTCA frequency up	integer	potential unknown motifs
449	TTTGT frequency up	integer	potential unknown motifs
450	AAATC frequency down	integer	potential unknown motifs
451	AAGGG frequency down	integer	potential unknown motifs
452	AAGGT frequency down	integer	potential unknown motifs
453	AATAG frequency down	integer	potential unknown motifs
454	AGGGG frequency down	integer	potential unknown motifs
455	AGTGG frequency down	integer	potential unknown motifs
456	ATCCC frequency down	integer	potential unknown motifs
457	ATGGG frequency down	integer	potential unknown motifs
458	ATGTG frequency down	integer	potential unknown motifs
459	CAAGG frequency down	integer	potential unknown motifs
460	CATTT frequency down	integer	potential unknown motifs
461	CCTGT frequency down	integer	potential unknown motifs
462	CGGGG frequency down	integer	potential unknown motifs
463	CTCAT frequency down	integer	potential unknown motifs
464	CTCTG frequency down	integer	potential unknown motifs
465	CTGGT frequency down	integer	potential unknown motifs
466	GAAGG frequency down	integer	potential unknown motifs
467	GAATC frequency down	integer	potential unknown motifs
468	GAATG frequency down	integer	potential unknown motifs
469	GACCC frequency down	integer	potential unknown motifs
470	GAGGG frequency down	integer	potential unknown motifs
471	GCGTC frequency down	integer	potential unknown motifs
472	GGAGT frequency down	integer	potential unknown motifs
473	GGCTC frequency down	integer	potential unknown motifs
474	GGGGA frequency down	integer	potential unknown motifs
475	GGTTG frequency down	integer	potential unknown motifs

476	GTAAG frequency down	integer	potential unknown motifs
477	GTCAC frequency down	integer	potential unknown motifs
478	GTCCA frequency down	integer	potential unknown motifs
479	GTCCT frequency down	integer	potential unknown motifs
480	GTCTC frequency down	integer	potential unknown motifs
481	GTCTG frequency down	integer	potential unknown motifs
482	GTCTT frequency down	integer	potential unknown motifs
483	GTGGT frequency down	integer	potential unknown motifs
484	GTGTC frequency down	integer	potential unknown motifs
485	GTGTT frequency down	integer	potential unknown motifs
486	GTTCT frequency down	integer	potential unknown motifs
487	GTTGT frequency down	integer	potential unknown motifs
488	TAAAT frequency down	integer	potential unknown motifs
489	TAGGC frequency down	integer	potential unknown motifs
490	TAGTC frequency down	integer	potential unknown motifs
491	TCATT frequency down	integer	potential unknown motifs
492	TCCTT frequency down	integer	potential unknown motifs
493	TCGTT frequency down	integer	potential unknown motifs
494	TCTTC frequency down	integer	potential unknown motifs
495	TCTTG frequency down	integer	potential unknown motifs
496	TCTTT frequency down	integer	potential unknown motifs
497	TGACT frequency down	integer	potential unknown motifs
498	TGAGT frequency down	integer	potential unknown motifs
499	TGATT frequency down	integer	potential unknown motifs
500	TGCGT frequency down	integer	potential unknown motifs
501	TGCTT frequency down	integer	potential unknown motifs
502	TGGCT frequency down	integer	potential unknown motifs
503	TGGGC frequency down	integer	potential unknown motifs
504	TGGGG frequency down	integer	potential unknown motifs
505	TGGGT frequency down	integer	potential unknown motifs
506	TGGTT frequency down	integer	potential unknown motifs
507	TGTCT frequency down	integer	potential unknown motifs
508	TGTGG frequency down	integer	potential unknown motifs
509	TGTGT frequency down	integer	potential unknown motifs
510	TTATT frequency down	integer	potential unknown motifs
511	TTCTC frequency down	integer	potential unknown motifs
512	TTCTT frequency down	integer	potential unknown motifs
513	TTGCG frequency down	integer	potential unknown motifs
514	TTGGG frequency down	integer	potential unknown motifs
515	TTGGT frequency down	integer	potential unknown motifs
516	TTTAT frequency down	integer	potential unknown motifs
517	TTTCT frequency down	integer	potential unknown motifs
518	AAACG up weighted	real	potential unknown motifs
519	AAAGC up weighted	real	potential unknown motifs
520	AAAGT up weighted	real	potential unknown motifs
521	AAATC up weighted	real	potential unknown motifs
522	AAATG up weighted	real	potential unknown motifs
523	AACCT up weighted	real	potential unknown motifs
524	AAGTT up weighted	real	potential unknown motifs
525	AATAA up weighted	real	potential unknown motifs
526	AATGT up weighted	real	potential unknown motifs
527	AGTAA up weighted	real	potential unknown motifs
528	ATAAA up weighted	real	potential unknown motifs

529	ATATG up weighted	real	potential unknown motifs
530	ATGTA up weighted	real	potential unknown motifs
531	ATGTG up weighted	real	potential unknown motifs
532	ATGTT up weighted	real	potential unknown motifs
533	ATTAA up weighted	real	potential unknown motifs
534	ATTAT up weighted	real	potential unknown motifs
535	ATTGT up weighted	real	potential unknown motifs
536	CAAAC up weighted	real	potential unknown motifs
537	CAATA up weighted	real	potential unknown motifs
538	CATTA up weighted	real	potential unknown motifs
539	CCAAT up weighted	real	potential unknown motifs
540	CCCAC up weighted	real	potential unknown motifs
541	CCCCA up weighted	real	potential unknown motifs
542	CGTGA up weighted	real	potential unknown motifs
543	CTGTG up weighted	real	potential unknown motifs
544	CTGTT up weighted	real	potential unknown motifs
545	CTTTG up weighted	real	potential unknown motifs
546	GAAAT up weighted	real	potential unknown motifs
547	GATTT up weighted	real	potential unknown motifs
548	GCAAA up weighted	real	potential unknown motifs
549	GCTGT up weighted	real	potential unknown motifs
550	GTAAA up weighted	real	potential unknown motifs
551	GTCAA up weighted	real	potential unknown motifs
552	GTTCA up weighted	real	potential unknown motifs
553	GTTGA up weighted	real	potential unknown motifs
554	GTTTC up weighted	real	potential unknown motifs
555	GTTTT up weighted	real	potential unknown motifs
556	TAAAA up weighted	real	potential unknown motifs
557	TAAAC up weighted	real	potential unknown motifs
558	TAAAG up weighted	real	potential unknown motifs
559	TAATA up weighted	real	potential unknown motifs
560	TACAA up weighted	real	potential unknown motifs
561	TACAT up weighted	real	potential unknown motifs
562	TATTG up weighted	real	potential unknown motifs
563	TCAAT up weighted	real	potential unknown motifs
564	TCTAT up weighted	real	potential unknown motifs
565	TCTGT up weighted	real	potential unknown motifs
566	TGAAA up weighted	real	potential unknown motifs
567	TGAAT up weighted	real	potential unknown motifs
568	TGACT up weighted	real	potential unknown motifs
569	TGCTT up weighted	real	potential unknown motifs
570	TGGAA up weighted	real	potential unknown motifs
571	TGTAA up weighted	real	potential unknown motifs
572	TGTAC up weighted	real	potential unknown motifs
573	TGTAT up weighted	real	potential unknown motifs
574	TGTCA up weighted	real	potential unknown motifs
575	TGTGA up weighted	real	potential unknown motifs
576	TGTTA up weighted	real	potential unknown motifs
577	TGTTT up weighted	real	potential unknown motifs
578	TTAAA up weighted	real	potential unknown motifs
579	TTATA up weighted	real	potential unknown motifs
580	TTATG up weighted	real	potential unknown motifs
581	TTCTA up weighted	real	potential unknown motifs

582	TTCTG up weighted	real	potential unknown motifs
583	TTGTA up weighted	real	potential unknown motifs
584	TTGTC up weighted	real	potential unknown motifs
585	TTGTG up weighted	real	potential unknown motifs
586	TTGTT up weighted	real	potential unknown motifs
587	TTTAC up weighted	real	potential unknown motifs
588	TTTAT up weighted	real	potential unknown motifs
589	TTTCA up weighted	real	potential unknown motifs
590	TTTGT up weighted	real	potential unknown motifs
591	AAATC down weighted	real	potential unknown motifs
592	AAGGG down weighted	real	potential unknown motifs
593	AAGGT down weighted	real	potential unknown motifs
594	AATAG down weighted	real	potential unknown motifs
595	AGGGG down weighted	real	potential unknown motifs
596	AGTGG down weighted	real	potential unknown motifs
597	ATCCC down weighted	real	potential unknown motifs
598	ATGGG down weighted	real	potential unknown motifs
599	ATGTG down weighted	real	potential unknown motifs
600	CAAGG down weighted	real	potential unknown motifs
601	CATTT down weighted	real	potential unknown motifs
602	CCTGT down weighted	real	potential unknown motifs
603	CGGGG down weighted	real	potential unknown motifs
604	CTCAT down weighted	real	potential unknown motifs
605	CTCTG down weighted	real	potential unknown motifs
606	CTGGT down weighted	real	potential unknown motifs
607	GAAGG down weighted	real	potential unknown motifs
608	GAATC down weighted	real	potential unknown motifs
609	GAATG down weighted	real	potential unknown motifs
610	GACCC down weighted	real	potential unknown motifs
611	GAGGG down weighted	real	potential unknown motifs
612	GCGTC down weighted	real	potential unknown motifs
613	GGAGT down weighted	real	potential unknown motifs
614	GGCTC down weighted	real	potential unknown motifs
615	GGGGA down weighted	real	potential unknown motifs
616	GGTTG down weighted	real	potential unknown motifs
617	GTAAG down weighted	real	potential unknown motifs
618	GTCAC down weighted	real	potential unknown motifs
619	GTCCA down weighted	real	potential unknown motifs
620	GTCCT down weighted	real	potential unknown motifs
621	GTCTC down weighted	real	potential unknown motifs
622	GTCTG down weighted	real	potential unknown motifs
623	GTCTT down weighted	real	potential unknown motifs
624	GTGGT down weighted	real	potential unknown motifs
625	GTGTC down weighted	real	potential unknown motifs
626	GTGTT down weighted	real	potential unknown motifs
627	GTTCT down weighted	real	potential unknown motifs
628	GTTGT down weighted	real	potential unknown motifs
629	TAAAT down weighted	real	potential unknown motifs
630	TAGGC down weighted	real	potential unknown motifs
631	TAGTC down weighted	real	potential unknown motifs
632	TCATT down weighted	real	potential unknown motifs
633	TCCTT down weighted	real	potential unknown motifs
634	TCGTT down weighted	real	potential unknown motifs

635	TCTTC down weighted	real	potential unknown motifs
636	TCTTG down weighted	real	potential unknown motifs
637	TCTTT down weighted	real	potential unknown motifs
638	TGACT down weighted	real	potential unknown motifs
639	TGAGT down weighted	real	potential unknown motifs
640	TGATT down weighted	real	potential unknown motifs
641	TGCGT down weighted	real	potential unknown motifs
642	TGCTT down weighted	real	potential unknown motifs
643	TGGCT down weighted	real	potential unknown motifs
644	TGGGC down weighted	real	potential unknown motifs
645	TGGGG down weighted	real	potential unknown motifs
646	TGGGT down weighted	real	potential unknown motifs
647	TGGTT down weighted	real	potential unknown motifs
648	TGTCT down weighted	real	potential unknown motifs
649	TGTGG down weighted	real	potential unknown motifs
650	TGTGT down weighted	real	potential unknown motifs
651	TTATT down weighted	real	potential unknown motifs
652	TTCTC down weighted	real	potential unknown motifs
653	TTCTT down weighted	real	potential unknown motifs
654	TTGCG down weighted	real	potential unknown motifs
655	TTGGG down weighted	real	potential unknown motifs
656	TTGGT down weighted	real	potential unknown motifs
657	TTTAT down weighted	real	potential unknown motifs
658	TTTCT down weighted	real	potential unknown motifs