CORRECTING REFERENCE BIAS IN HIGH-THROUGHPUT SEQUENCING ANALYSIS

Shunping Huang

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2015

Approved by:

Wei Wang

Leonard McMillan

Vladimir Jojic

Fernando Pardo-Manuel de Villena

Wei Sun

ABSTRACT

SHUNPING HUANG: Correcting Reference Bias in High-throughput Sequencing Analysis
(Under the direction of Wei Wang)

Mapping reads to a reference sequence is a common step when analyzing high throughput sequencing data. The choice of reference is critical because its effect on quantitative sequence analysis is non-negligible. Recent studies suggest aligning to a single standard reference sequence, as is common practice, can lead to an underlying bias depending the genetic distances of the target sequences from the reference. To avoid this bias researchers have resorted to using modified reference sequences. Even with this improvement, various limitations and problems remain unsolved, which include reduced mapping ratios, shifts in read mappings, and the selection of which variants to include to remove biases.

To address these issues, I proposed novel and generic pipelines that integrate the genomic variations from known or suspected founders into reference sequences and then perform read alignment. Experiments show that my pipelines can align more reads with much lower reference bias than the traditional pipeline where reads are mapped against the standard reference sequence. They can be applied to a wide range of organisms, including inbreds, F1s, and outbreds, and various high throughput sequencing approaches, such as RNAseq, DNAseq, ChiPseq, etc.

Dedicated to my parents and my wife.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor Dr. Wei Wang for her support and guidance during my Ph.D. journey. She has encouraged my study and provided me lots of constructive suggestions on research and career development. I am also very thankful to Dr. Leonard McMillan for inspiring me with new thoughts and ideas in every discussion we had. I thank Dr. Fernando Pardo Manuel de Villena for the collaboration oppotunity and his insightful advices on my research projects. I am also grateful to Dr. Vladimir Jojic and Dr. Wei Sun for their invaluable feedbacks on my dissertation.

Many of my research projects were joint efforts with other collaborators, so my appreciation extends to them, the members in COMPGEN group and CEGS group. I thank Dr. Xiang Zhang for his helps during my first few years of research. I also thank Dr. Yi Liu, Zhaojun Zhang, Andrew Parker Morgan, Weibo Wang, Wei Cheng, James Matt Holt, Chia-Yu Kao, Zhaoxi Zhang, Jack Wang, and many other UNC graduate students for supporting my work in these years.

I was funded by SAS Institute in the last year of my Ph.D. research. I am so thankful for their internship opportunity. I would like to thank my former manager Taiyeong Lee for the trust and support. I also thank my colleages in SAS for such a great working environment.

At last, I am so grateful to my parents and my wife. Every time I feel tired and frustrated, it is their love that brings power and courage back to me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DNA	Deoxyribonucleic acid
RNA	Ribonucleic acid
mRNA	Messenger RNA
rRNA	Ribosomal RNA
tRNA	Transfer RNA
cDNA	Complementary DNA
DNA-seq	DNA Sequencing
RNA-seq	RNA Sequencing
SNP	Single Nucleotide Polymorphism
CNV	Copy Number Variation
RIL	Recombinant Inbred Line
MLE	Maximum Likelihood Estimate
MAP	Maximum a posteriori
STD	Standard Deviation
MAD	Median Absolute Deviation
ROC	Receiver Operating Characteristic
NGS	Next Generation Sequencing

CHAPTER 1: INTRODUCTION

It is well known that the biological information of all organisms and many viruses is stored in deoxyribonucleic acid (DNA) composed of nucletides. This information plays an essential role in the suvival, development, and reproduction of an organism. Since 1970s, researchers have been using all kinds of sequencing approaches to determine the order of nucleotides and unwind the secret of DNA. While in the early days sequencing was prohibitively time-consuming, labor intensive, and expensive, its cost has been dramatically brought down by the recent advance of high-throughput sequencing technology, making it universally accessible to average labs. Nowadays, sequencers generate huge volumnes of raw reads, which then should be further analyzed.

There are two common strategies for processing raw reads: alignment and assembly. Given that the procedure of assembly is usually memory- and time-consuming, many researchers are inclined to use alignment approaches. One of the prerequisites of alignment is the reference sequence, to which reads are mapped. While the standard reference sequence of an organism is commonly used in alignment, it is not always guaranteed that the target organisms match exactly the standard reference. Ignoring such difference may result in bias during read alignment, namely reference bias.

In this chapter, I first introduce some basic biology, which should be sufficient for average readers to understand the rest of my dissertation. Then I present the alignment procedure, reference sequence, and genomic variants. I also survey some related work on reference bias.

1.1 Basic Biology

Deoxyribonucleic acid (DNA) is a molecule that contains the genetic code of an organism. Most DNA molecules have a double-stranded structure, with each strand composed of repeating units called **nucleotides**. In general, there are four different types of nucleotides: guanine (G), adenine (A), thymine (T), or cytosine (C). Each nucleotide is binded to one another in the same strand, and paired with another nucleotide in the other strand under the base pairing rule: A-T and C-G.

DNA wrapped with proteins is organized into thread-like structures called **chromosomes**. There are two types of chromosomes based on whether they control somatic characters or sex characters. The former type is called **autosomes**, and the latter type **sex chromosomes**. The entire set of chromosomes, which contains all genetic information of an organism, is called the **genome** of this organism. The number of chromosomes in a genome is different for different **plodies** (i.e. the number of sets of chromosomes in a cell) and species. For example, among diploid organisms, where chromosomes are in pairs, humans have 23 pairs of chromosomes, including 22 pairs of autosomes and 1 pair of sex chromosomes; mice have 20 pairs of chromosomes, where 19 of them are autosomes and the remaining pair are sex chromosomes (Figure 1.1).

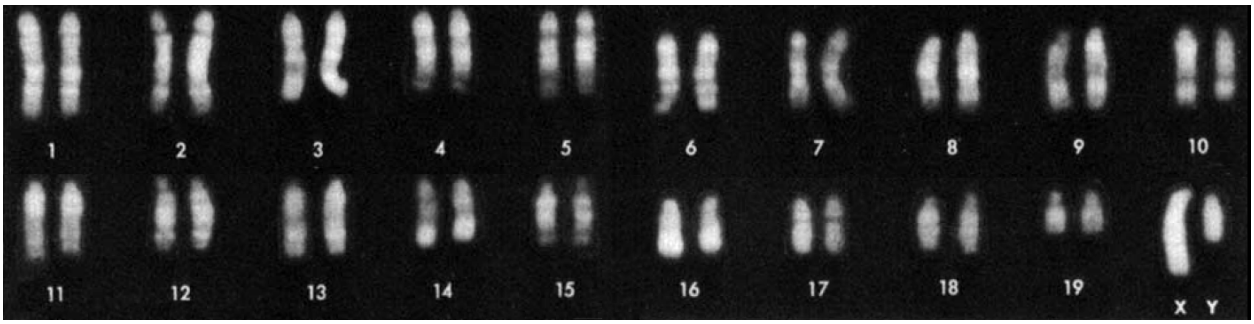


Figure 1.1: Mouse Karyotype (Committe on Standardized Genetic Nomeclature for Mice, 1972). The mouse genome has 19 pairs of autosomes and one pair of sex chromosomes.

DNA controls the organism's cells and body through a multi-step process called **protein synthesis**. However, not all of the genomic sequence is functional in this process. The region that is functional and corresponding to a unit of inheritance is called a **gene**, and the region that does not encode protein sequence is called **noncoding DNA**. Protein synthesis consists of two major steps, **transcription** and **translation** (Figure 1.2), and involves three types of **ribonucleic acid (RNA) – messenger RNA (mRNA)**, **ribosomal RNA (rRNA)**, and **transfer RNA (tRNA)**.

Transcription With the help of certain enzymes, the double-stranded DNA is unwound and one of the strands is transcribed into mRNA. During the transcription, the **pre-mRNA** molecule (a type of primary **transcript**) is generated to be complementary to the DNA strand from which it is transcribed, except that thymine (T) is replaced with uracil (U). Then the pre-mRNA is modified by **RNA splicing**, during which some RNA sequences are discarded while others remain (Figure 1.3). The final product after splicing is called **mature mRNA** or simply mRNA. The sequences remaining in the mRNA are called **exons**, while

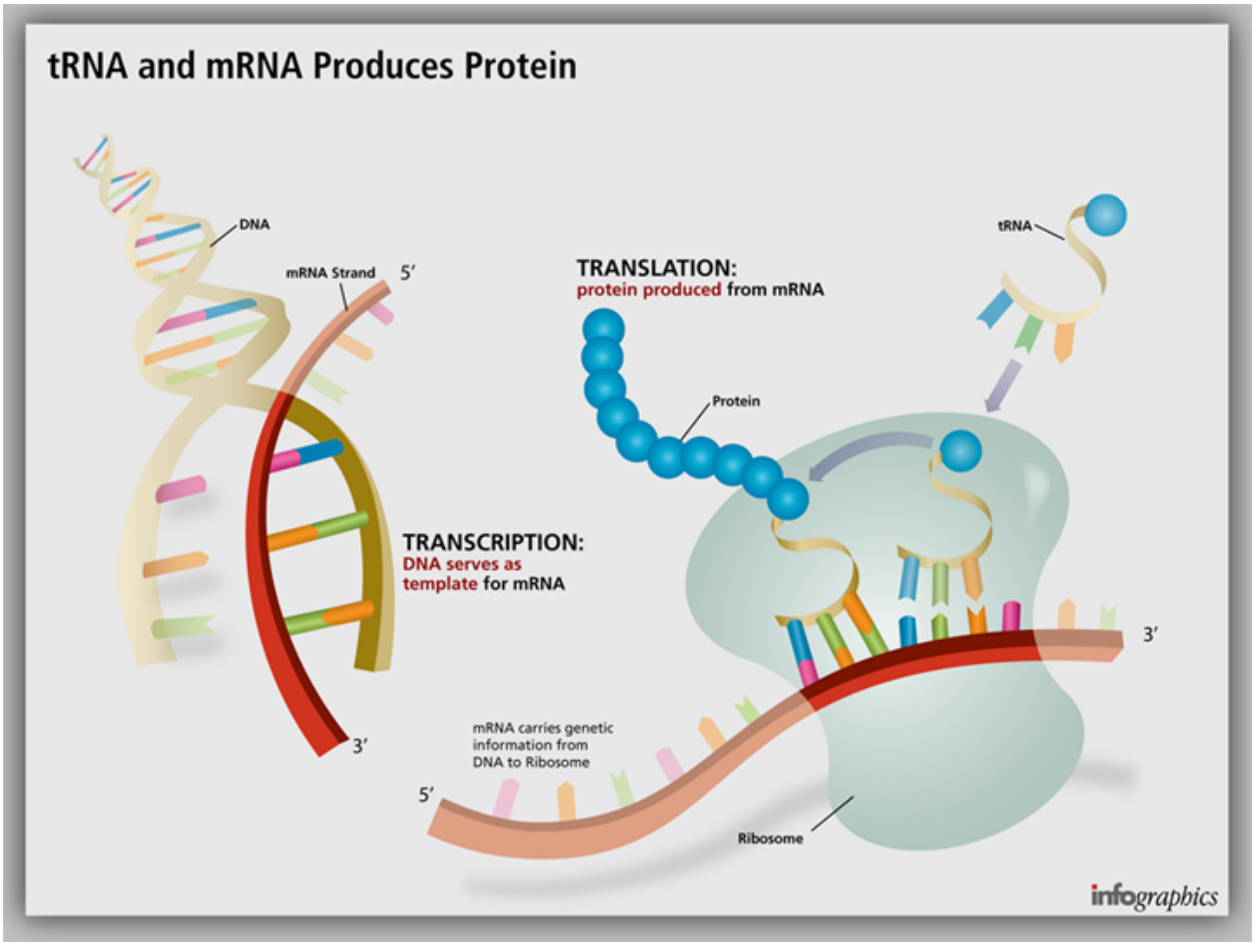


Figure 1.2: DNA Transcription and Translation are the two major steps in protein synthesis. The genetic code of DNA is first transcribed to mRNA and then translated into amino acids. (`http://www.ignyc.com/our-work/portfolio/transcription-translation/`)

the discarded ones are called **introns**. For a given gene, there can be different ways of splicing, also called **alternative splicing**, which will result in different mRNA sequences being produced.

Translation After the transcription is complete, the **ribosome**, to which rRNA is attached, is the place where the nucleotide sequence in mRNAs is translated into proteins. Every three consecutive nucleotides, also called a **codon**, represent an amino acid or a start/stop signal of translation. As the ribosome moves along the mRNA sequence, tRNA molecules are responsible for carrying the amino acids that match the codon to ribosome. After ribosome decodes the whole mRNA sequence into a long chain of amino acids, a protein (or polypetide) is produced.

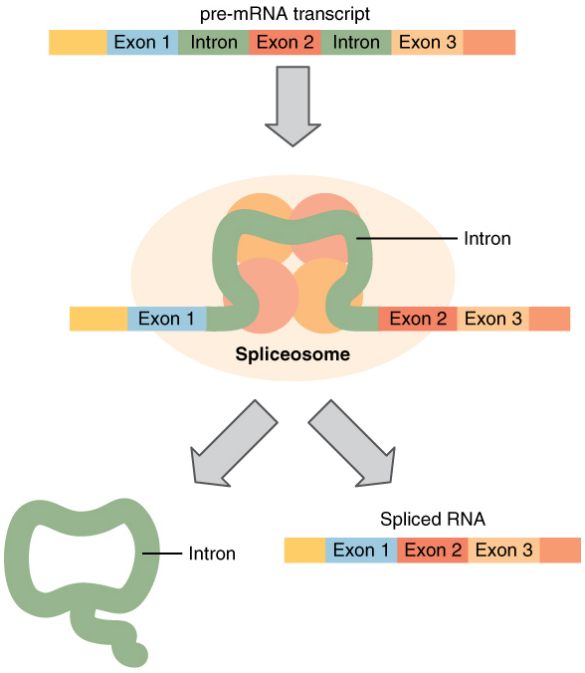


Figure 1.3: RNA Splicing. The exons are remained in the mature mRNA, while the introns are removed. (`http://en.wikipedia.org/wiki/RNA_splicing`)

1.2 High-throughput Sequencing

Sequencing technologies always have a great impact on the research of sequence analysis. In the early decades, sequencing was a time-consuming and money-consuming process, since it was hardly automated and could only determine a few sequence each time. Recent development of high-throughput sequencing technologies has allowed sequencing processes to be run in parallel. This has dramatically lowered down the cost of DNA sequencing, making it accessible to labs of different sizes in the research community. So far, high-throughput sequencing technologies have been widely used in the studies of whole genome sequencing (DNA-seq), transcriptome profiling (RNA-seq), DNA-protein interactions (ChIP-seq), etc.

Take a look at RNA sequencing on the Illumina Platform (Figure 1.4). After the mRNA molecules are extracted and purified, they are fragmented into small pieces. The fragmented mRNAs then go through a reverse transcription step in which complementary DNAs (cDNAs) are generated. The cDNA insert size of the library is around 100-400bp. Finally, massively parallel sequencing of cDNAs takes place in the sequencers, and millions of raw reads are generated. Such high volumes of read data have posted a great challenge in sequence analysis.

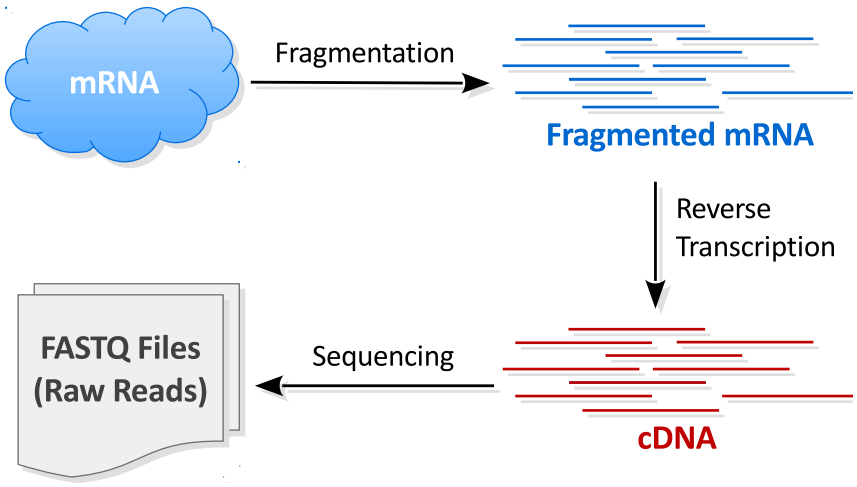


Figure 1.4: RNA-seq Procedure. The mRNA molecules from a sample is extracted, purified, and reversely transcribed to cDNAs. Then cDNAs are sequenced by the sequencer machine to produce reads.

1.3 Alignment

Read alignment (or simply **alignment**) is the process of figuring out the localtion of each read in a **reference sequence**, which is a very long sequence served as a template. After reads are obtained from the sequencer, alignment is the preferred way of read processing, since it requires less time and memory than assembly.

An **aligner** is a piece of code/tool that performs the alignment procedure. For each read in a given set, an aligner will try to find its location in the reference sequence. If such a location can be found, it is called a **mapping** of this read. A read mapping is not guaranteed to match the exact sequence of the read. Some aligners can allow certain amounts of mismatches or gaps. In the end, there are three different outcomes while aligning a read: (a) only one location is found, which is also called a **unique mapping**; (b) more than one location is found, i.e. **multiple mappings**; (c) no location is found, namely **no mapping**.

1.3.1 Reference Sequence

A common prerequisite of nearly all sequence analysis pipelines is to align read fragments to a high-quality reference genome sequence, regardless of the underlying organism's ploidy. This reference sequence/genome is usually called the **standard reference sequence/genome** of this organism.

Typically this standard reference sequence is of a genetically close organism with the same karyotype and genomic arrangement as the target organism. Moreover, a large amount of annotation effort has generally been applied to the reference genome. In particular, the placements and extents of genes and exons (Flicek

et al., 2012), functional elements (The ENCODE Project Consortium et al., 2011), and genetic variants (The 1000 Genomes Project Consortium, 2010; Keane et al., 2011) are given in coordinates relative to a reference genome.

1.3.2 Genomic Variation

Although the reference sequence for one species is high-quality, it cannot represent the sequence of every samples from this species. The difference between the standard reference and the target organism is called **genomic variation**.

Genomic variation can be separted into different types based on the size and the effect on the genome.

Single Nucleotide Polymorphisms (SNPs) A SNP is a point mutation which causes a substitution of a single nucleotide with another. There are two categories of point mutation: transitions and transversions (Figure 1.5). Transitions occur about ten times more often than transversions.

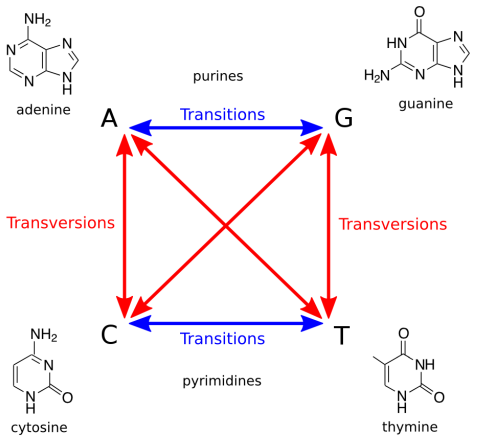


Figure 1.5: Transition and Transversion. Transition represents the conversion between A and G (purine to purine) or C and T (pyrimidine to pyrimidine), while transversion represents the other combinations. (`http://en.wikipedia.org/wiki/Transition_(genetics)`)

Insertions or Deletions (Indels) An indel represents an insertion or deletion of bases in the sequence (Figure 1.6).

Large-scale Structure Variations There are several types of structure variations. A **copy number variation (CNV)** corresponds to a large region of sequence being duplicated or deleted on the same chromosome. The duplication cases are also called copy number gains, while the deletion cases are copy number losses.

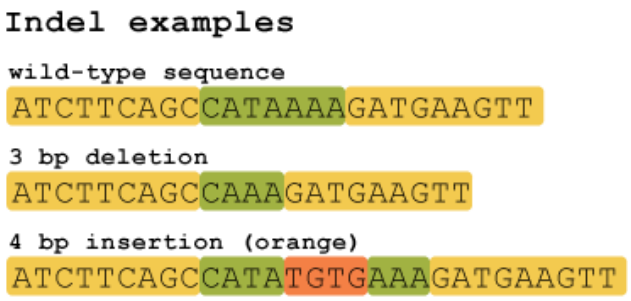


Figure 1.6: Indel examples (http://blog.hackbrightacademy.com/2013/07/indel-finder-how-the-python-version-of-this-program-works/)

A **translocation** occurs when a large chromosome region is exchanged with another large chromosome region. An **inversion** happens when one chromosome breaks in two different locations, and the middle part of sequence is reversed and re-attached to the chromosome. The examples of different types of structure variation are shown in Figure 1.7.

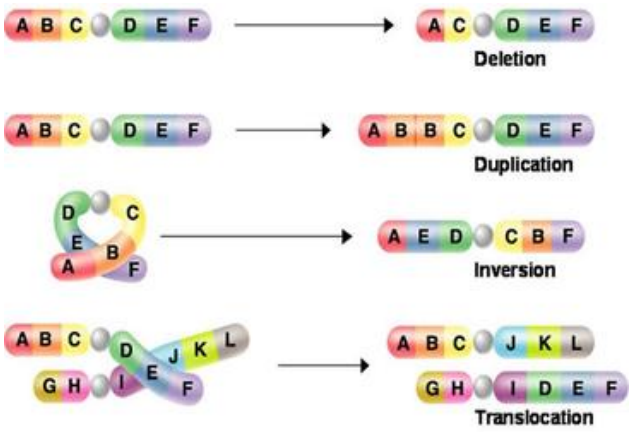


Figure 1.7: Examples of Structural Variations. (http://hodnett-ap.wikispaces.com/Chapter+23+The+Evolution+of+Populations/)

1.4 Related Work on Reference Bias

A multitude of new quantitative sequence analysis methods are based upon an initial alignment to some reference sequence. By *quantitative* I mean methods where depth of coverage and the consistency of reads at a given genomic position factor into some measure of interest. In the case of RNA-seq analysis, this includes estimation of relative or absolute transcript abundance (Marioni et al., 2008; Trapnell et al., 2010), assessing parent-of-origin effects (Gregg et al., 2010; DeVeale et al., 2012), and estimating RNA-editing rates (Picardi

et al., 2010; Peng et al., 2012). Whereas ascertaining copy-number and calling genomic variants are common DNA-seq quantitative analysis examples.

While the standard reference genome is widely used in read alignment, the quality of the alignment decreases as measured by the numbers of unmapped and misaligned fragments, as the genetic distance between the reference and target genomes increases. Also, aligning reads to a reference genome can introduce local alignment biases, (i.e., regions that better match the reference sequence tend toward higher coverage than regions with variations (Degner et al., 2009; McDaniell et al., 2010)) which largely confounds downstream quantitative analyses.

Many researchers have addressed this issue by incorporating variants, to various extents, into a *pseudo*-reference genome sequence. Incorporating only SNPs is commonplace and straightforward (Satya et al., 2012), since it does not change the coordinates of the constructed genome sequence. Alternatively, Degner *et al.* (Degner et al., 2009) masked every known polymorphic location in the reference genome by introducing a third allele, thus increasing the genetic distance between the target and the reference in an unbiased fashion. However, this approach reduces the total number of reads aligned because the added masked alleles always introduce mismatches, which all aligners attempt to minimize. In fact, unmapped reads result when the best mapping considered has excessive mismatches. In RNA-seq experiments, this reduction in mapped reads leads to underestimation of expression level of genes with variations(Turro et al., 2011).

There have also been some attempts to utilize other variants besides SNPs. Rivas-Astroza *et al.* (Rivas-Astroza et al., 2011) developed a software (perEditor) to build a personal genome with different variant types, but they only focussed on genome construction without resolving the coordinate inconsistency after read alignment.

The problem is getting even worse if the target organism is non-inbred. When parental haplotypes differ in their similarity to the reference sequence a significant alignment bias can result. Gregg *et al.* (Gregg et al., 2010) aligned F1 hybrids from reciprocal crosses between the isogenic mouse strains CAST/EiJ and C57BL/6J to the NCBI37/mm9 mouse genome and transcriptome to study parent-of-origin effects. However, this approach favors reads with reference alleles, and it is worth noting that the mouse reference genome is largely based on C57BL/6J.

Several attempts have been made to create a sample-specific reference genome or transcriptome for F1 alignments. Keane *et al.* (Keane et al., 2011) aligned reads from F1 cross of C57BL/6J $\times$ DBA/2J to a

C57BL/6J-based reference genome and an approximate DBA/2J genome where known DBA/2J variants, primarily SNPs, were substituted into the reference. Turro *et al.* (Turro et al., 2011) proposed a hybrid pipeline that first aligned reads to a reference genome in order to call SNPs, and then re-aligned the same reads to a customized transcriptome with the discovered SNPs incorporated. Since single-base substitutions do not change genome coordinates, it is straightforward to embed SNPs. However, this method cannot be easily generalized to other frame-shifting variants such as small indels, inversions and CNVs to which a sequence aligner is more sensitive. Rozowsky *et al.* (Rozowsky et al., 2011) proposed AlleleSeq for constructing a modified diploid genome by inserting SNPs and indels into the reference genome and using this diploid genome as the reference during alignment to avoid errors caused by reference bias. While AlleleSeq is similar to one of my proposed pipelines, it is limited to diploid organisms. Moreover, as shown below, it analyzes differential expression at variant positions, which will become more difficult as the density of variants increases.

After reads are mapped, the relative read counts in specific regions of the sequence are often used to quantify abundance within a genomic region. In DNA sequencing (DNA-seq), local read counts are used to estimate copy-number gain or loss (Yoon et al., 2009; Magi et al., 2012). In RNA sequencing (RNA-seq), local read counts are used to quantify gene expression levels and to identify the isoforms expressed (Richard et al., 2010; Turro et al., 2011). In diploid organisms, researchers have been interested in assessing the differential expression levels between parental haplotypes, (i.e., parent-of-origin or allele effects). In a typical analysis of differential expression, the read coverage at each known variant position are partitioned by allele and then used to estimate the imbalance (Gregg et al., 2010; Keane et al., 2011; Rozowsky et al., 2011). Statistical corrections to the read counts are required when the density of local variations allows multiple variants to fall in the same read, or read-pair. Thus, in regions with dense genomic variations, the quantitative use of read counts is complicated both by the inability to align, and by the difficulty of establishing the independence of each variant observation.

1.5 Contributions

1.5.1 Thesis Statement

My thesis statement is as follows: combining knowledge of genomic variants into reference sequence for alignment can largely reduce reference bias. The pipelines I proposed can align more reads with much lower

reference bias than the traditional pipeline where reads are mapped against the standard reference sequence. They can be applied to a wide range of organisms, including inbreds, F1s, and outbreds. Performance is measured by the ratio of mapped or uniquely mapped reads as well as the distribution of reads among different founders.

1.5.2 Organization

The remainder of my dissertation is organized as follows:

- **Chapter 2** presents a general variation format between genomes, which plays an important role in the pipelines I proposed.
- **Chapter 3** presents the new alignment pipelines for different types of organisms to correct reference bias.
- **Chapter 4** presents a probabilistic model to handle remaining bias by common trend estimation and outlier detection.
- **Chapter 5** concludes the thesis and discusses some ideas of future work.

CHAPTER 2: MOD FORMAT

In this chapter, I propose a general framework for mapping coordinates back and forth between genomes. It employs a new format, namely MOD format, to represent known variants between genomes. This format greatly facilitates genome construction and position mapping, and is the key component in the pipelines for correcting reference bias in Chapter 3.

2.1 Design of the MOD format

The MOD format is composed of instructions that transform one genome sequence into another. It is essentially an edit transcript relating two strings (Gusfield, 1997), and it provides a basis for quantifying the similarity of two sequences. A MOD file is not necessarily unique, nor is it claimed to be minimal. The genome before transformation is called the **source** and the one after is called the **destination**. Each MOD file is directional, i.e. always from the source to the destination.

A MOD file consists of two parts (Figure 2.1a): a header and a body. The header includes the metadata of the transformation, such as, the version of the MOD format, the source, the destination, and so forth. The body holds the instructions, each of which has its affected position and arguments. Positions are all stored in the source coordinate system, and the bases before and after modification are included in the arguments.

There are three basic types of instructions defined in the MOD format: **s-**, **d-**, and **i-instructions**. They describe single-base substitutions, single-base deletions, and insertions, respectively. All instructions are **atomic**, in that they reference no more than one position from the source. It is obvious that both s-instructions and d-instructions are atomic. For i-instructions, I merely add new sequence after an anchor position in the source without altering any base; thus they are also atomic.

One way to generate a MOD file is to convert common variant calls into instructions. For example, SNPs and genomic insertions can be directly changed to s-instructions and i-instructions, respectively. Genomic deletions need to be broken up into single-base deletions before converting to d-instructions (Figure 2.1a and 2.1b). Notice that the position information in adjacent d-instructions is redundant. However, the design

choice of keeping all instructions atomic facilitates later MOD-file manipulations, whose advantages are considered to outweigh this slight redundancy. Moreover, the additional space overhead is recovered when MOD files are compressed.

Complex structure variants, such as tandem duplications, inversions, and translocations, can be described by the current set of instructions. For example, a tandem duplication is represented by repeating an i-instruction at the same location, while inversions (or translocations) are implemented by a series of d-instructions at the source sequence position and a corresponding i-instruction of the inverted (or transferred) sequence at its new position. It is recommended to annotate such coupled sets of instructions using comments following the instructions.

New MOD files can also be derived from other MOD files by various properties of the format. This will be discussed in Section 2.2 .

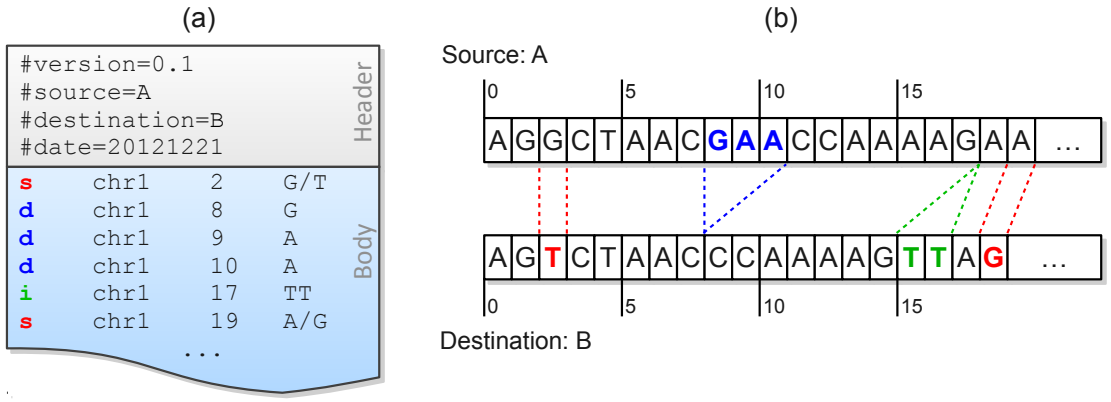


Figure 2.1: A MOD file example (a) and the corresponding sequences of the source and the destination (b). There are two SNPs between these sequences, and they are represented as two s-instructions at source positions 2 and 19. A three-base deletion (from source positions 8 to 10) is observed, and it is broken up into three d-instructions. The insertion after position 17 is directly added to the MOD file without any conversion due to its atomicity.

2.2 Properties of the MOD format

2.2.1 Invertibility

A MOD file specifies all changes from the source genome to the destination genome; this includes bases both before and after each change. Therefore, one can exchange the source and the destination by inverting the instructions in the file (Figure 2.2).

For example, each s-instruction specifies a position and a nucleotide from the source and its replacement nucleotide in the destination, so it can be inverted by merely swapping the two nucleotides. The d-instructions contain bases that they remove from the source, so inverting them will result in inserting these deleted bases back to the destination, i.e. i-instructions. Moreover, since d-instructions are restricted to be one-based but i-instructions are not, adjacent i-instructions can be combined into one as an optimization. Similarly, i-instructions are broken up into multiple d-instructions While they are inverted.

The position of each instruction must also be modified when inverting a MOD file. Positions in the source coordinate system are changed into destination coordinates in the output as described in Section 2.3.2 .

2.2.2 Concatenability

The MOD files with the same source genome can be concatenated. In other words, one can combine a prefix sequence generated from one MOD file with a suffix sequence from another MOD file without messing up the coordinates or missing any variants on the segment boundaries, as long as the two MOD files have the same source (Figure 2.2). Concatenation is used to construct new genomes for hybrid organisms (e.g., F2s and backcrosses) to account for recombinations.

Concatenability results from the use of atomic instructions in a single source coordinate system. Given a genomic region, every instruction in a MOD file will be either inside or outside the region; there is no case where an instruction crosses a region boundary. Therefore, unlike variant calls that require special care in the boundary cases, MOD files can be safely cropped.

If the cropped regions from different MOD files are disjoint, their instructions can simply be stacked together; otherwise, it is possible that some positions may be affected by more than one instruction. When multiple instructions refer to the same genomic coordinate, several rules are used as the tiebreaker. Generally speaking, if instructions of different types are performed on the same location, d-instructions take precedence over s-instructions. If instructions have the same type, only the last one will be used. Notice that there is no preference between i-instructions and other instruction types, because insertions have no footprint in the source, and therefore will not contradict other instructions. If two or more i-instructions specify the same position, they are added in order.

2.2.3 Composability

Given two MOD files containing instructions to transform genome A to genome B and genome B to genome C, respectively, one can construct a MOD file transforming genome A to genome C (Figure 2.2). This property is called composability.

Let $\mathbf{P}(A \mapsto B)$ and $\mathbf{Q}(B \mapsto C)$ represent the two MOD files to be composed, and $\mathbf{R}(A \mapsto C)$ be the resulting MOD file. The procedure of composition is briefly described as follows. First, we invert $\mathbf{P}$ to obtain $\mathbf{P}'$, which contains instructions mapping from genome B to genome A. Second, we compute the intersection of $\mathbf{P}'$ and $\mathbf{Q}$, i.e., $\mathbf{P}' \cap \mathbf{Q}$. These shared instructions indicate that A and C are identical at the corresponding positions, because same changes should be made to change B to A and B to C. Third, we remove the intersection from $\mathbf{P}'$ and $\mathbf{Q}$ obtaining $\bar{\mathbf{P}}'$ and $\bar{\mathbf{Q}}$ separately. These two MOD files are the actual difference between A and C. Finally, we map the instruction positions in $\bar{\mathbf{Q}}$ from genome B coordinates to genome A coordinates (described in Section 2.3.2), and combine the result with the inversion of $\bar{\mathbf{P}}'$. This gives the expected MOD file $\mathbf{R}$.

2.2.4 Other properties

In addition to these three properties, the MOD format has other virtues. For example, it can be easily converted from the VCF format (Danecek et al., 2011), which is commonly used to store variant calls. Also, the MOD files can be compressed by bgzip (Li et al., 2009) and indexed by tabix (Li, 2011), so that the file sizes are reduced and they can be efficiently queried.

It is also convenient to edit a MOD file to incorporate new variants or to mask obsolete ones. Since all positions are in the same coordinate system for each MOD file, there is no need to worry about adjusting positions when adding or removing variants.

2.3 Use of the MOD format

2.3.1 Genome Construction

MOD formatted files provide a generative procedure for transforming a source sequence to a destination; thus, they are ideally suited for constructing an in silico target genomic sequence from a given reference. This generated genome is called a **pseudogenome**.

One can easily construct MOD files for entire catalogs of common inbred strains using readily available variant calls. The property of concatenability makes it convenient to create the pseudogenomes for arbitrary crosses between inbred strains and recombinant inbred lines (RILs)(Silver et al., 1995). The genomes of RILs are a mosaic of two or more founder genomes. Once the haplotype structure of a RIL is inferred (Liu et al., 2010; Fu et al., 2012), one can concatenate the regions of instructions from founder MOD files to form a new MOD file for the RIL, which can then be utilized for alignments.

When using MOD files it is often assumed that there is a common source genome, or reference, but this restriction is unnecessary. The MOD format can be used to map between any two genomes or genome versions, allowing the source sequence to be transformed to any destination sequence.

2.3.2 Position Mapping

The MOD format also provides the capability to map coordinates or intervals from the source to the destination, and vice versa. This is done by scanning a MOD file and accumulating the number of shifted bases affected by d-instructions and i-instructions. For every pair of corresponding regions in the two genomes, I record a pair of offsets. Given a position in the source, I first look up in the source offsets to find out in which region it falls, and then compute its destination position.

The invertibility of the MOD files guarantees that one can map positions back and forth between the source and the destination. The composability also extends the mapping ability. For example, given two MOD files, from the reference to two non-reference strains, one can invert one and compose them to get a third. With the help of this MOD file, one can map positions between the two non-reference strains.

Position mapping can be applied to genome annotations, which is usually presented in the reference coordinate system, to get a new target-specific annotation. Position mapping can also be applied to genome alignment results, so one can remap the alignments from one genome to another.

2.3.3 Application in Mouse Genome

Here I used the mouse as the model organism, but the MOD format and the tools I proposed are also applicable to other organisms.

I used three wild-derived inbred mouse strains in my experiments: CAST/EiJ, PWK/PhJ, and WSB/EiJ, all of which are highly diverged from the *Mus musculus* reference genome derived largely from C57BL/6J.

Strain	s-instructions	d-instructions	i-instructions
CAST	17,674,364	4,834,899	4,206,776
PWK	17,202,935	4,715,249	3,457,436
WSB	6,045,875	2,026,461	1,579,714

Table 2.1: Statistics of MOD files for CAST/EiJ, PWK/PhJ and WSB/EiJ. The counts are in units of base-pairs. For s-instructions and d-instructions, they are just the numbers of instructions, respectively. For i-instructions, the counts are derived from adding up the number of bases in each inserted sequence.

The SNP and indel variants for these strains were downloaded from the Wellcome Trust Sanger Institute (Keane et al., 2011), while the mouse reference genome data is from NCBI MGSCv37.

To generate MOD files for the three target strains, I first extracted SNPs and indels from the VCF files (downloaded from `ftp://ftp-mouse.sanger.ac.uk/REL-1105/`). Only high-confidence SNPs and indels for the 19 autosomes and X were incorporated into the MOD files. Variants on Y and mitochondria (M) were extracted from other sources (`http://cgd.jax.org/datasets/popgen/diversityarray/yang2009.shtml`). The MOD files used in this paper can be found at `http://www.csbio.unc.edu/CCstatus/index.py?run=Pseudo`. I summarized the statistics for each MOD file in Table 2.1.

The total number of bases involved in all instructions of a MOD file can be used as an estimation of genomic distance between a strain and the reference. Figure 2.6 shows such distance for the three strains studied. The CAST strain is the most distant genetically from the reference and the WSB strain is the genetically closest to the reference.

To illustrate the density of the genomic variants (currently described in the MOD files) of the three strains and their potential impact to read alignment, I divided the pseudogenome into 100bp windows and counted the number of bases that are modified by any instruction in each window. In about 11.72 % of windows three or more bases are affected by CAST/EiJ variants, while the percentages are 11.22% and 3.80% for PWK/PhJ and WSB/EiJ, respectively. In general, high-variant windows are uniformly distributed along the genome, suggesting that, if I use the reference genome for read alignment, the alignment quality may be substantially compromised over the entire genome. I show the distribution of the counts for the three strains in Figure 2.3, 2.4, and 2.5.

Strain	Exons	Transcripts	Genes
CAST	5.98%	78.04%	62.37%
PWK	5.87%	76.68%	61.42%
WSB	3.01%	65.76%	52.75%

Table 2.2: Percentage of exons, transcripts, and genes that have different lengths in the pseudogenomes.

I also constructed strain-specific gene annotations for the CAST, PWK and WSB pseudogenomes, and investigated how many exons, transcripts, and genes were changed after variants were incorporated in the reference.

The gene annotation of mm9 was from Ensembl (Flicek et al., 2012). There are, in total, 688,311 exons, 97,251 transcripts, and 37,620 genes in the latest release (release 67).

To accomplish this, I developed a tool, *modmap*, for mapping positions and intervals from source to the destination. It takes a MOD file and an annotation file as input. It first builds a position mapping between genomes internally and then changes the annotation file's position columns from source coordinates to destination coordinates. In the current setting, the source is the reference genome, and the destination is the pseudogenome.

After the strain-specific annotation was obtained, I compared it with the original annotation in terms of the start positions and the ranges of exons, transcripts, and genes.

Not surprisingly, because of the integrated indels and structure variants, the start positions of almost all exons, transcripts and genes (over 99%) are shifted in the pseudogenome annotation. In addition, about 6% of exons, 78% of transcripts and 62% of genes have different lengths in the CAST pseudogenome (Table 2.2). There is a strong correlation between the number of changes in a strain and the genetic distance of the strain from the reference (Figure 2.6).

It is worth noting that the pseudogenome annotations could still be inaccurate. In fact, Gan *et al.* (Gan et al., 2011) recently raised the issue that simply mapping gene annotations back and forth may lead to incorrect annotations. Nevertheless, MOD-file derived pseudogenomes can serve as a first-cut approximations to facilitate initial reannotations while also supporting efficient remapping back to reference. As a more accurate picture of the actual genomic structure develops it can easily be incorporated into the MOD-file.

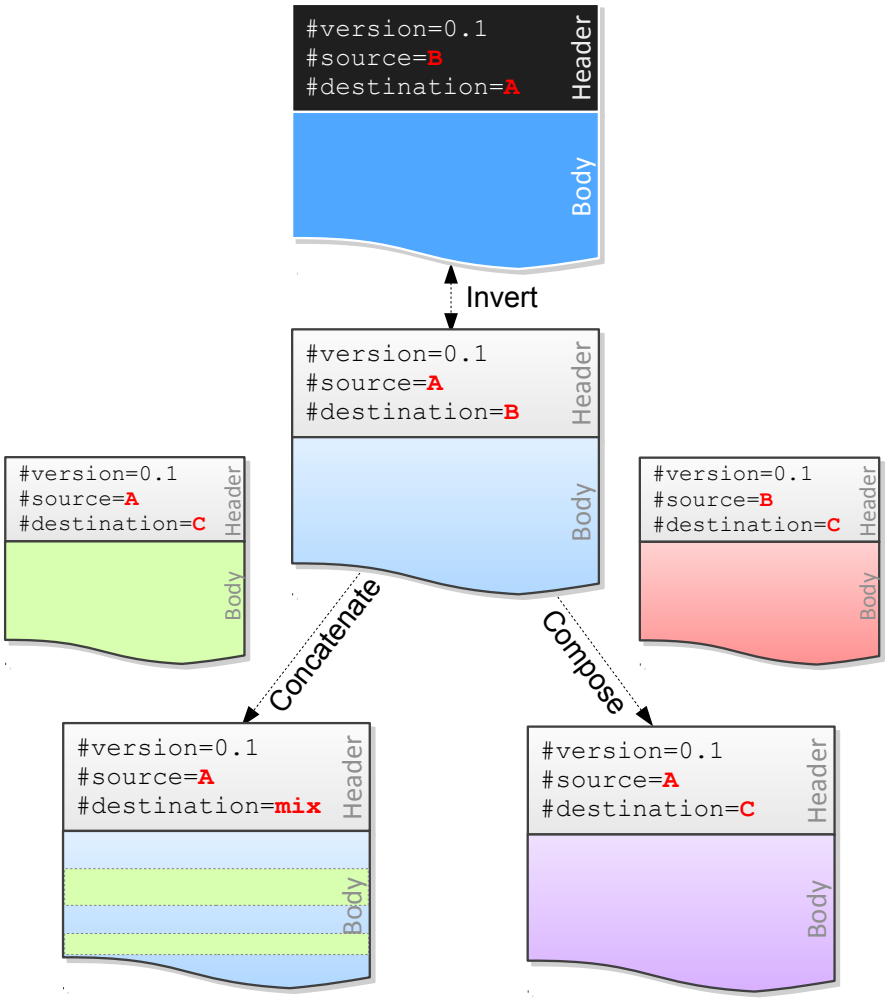


Figure 2.2: The three main properties of the MOD format enable a wide range of operations on MOD files. The original MOD file is in the middle, with A as the source and B as the destination. Inversion only involves one MOD file, and it will exchange the source and the destination (top part). Concatenation and composition, however, need two or more MOD files. Concatenating two MOD files with the same source (A in the figure) will generate a mixed destination (bottom left part). Composing two MOD files, from A to B and B to C, can be considered as transferring from A to C through B (bottom right part).

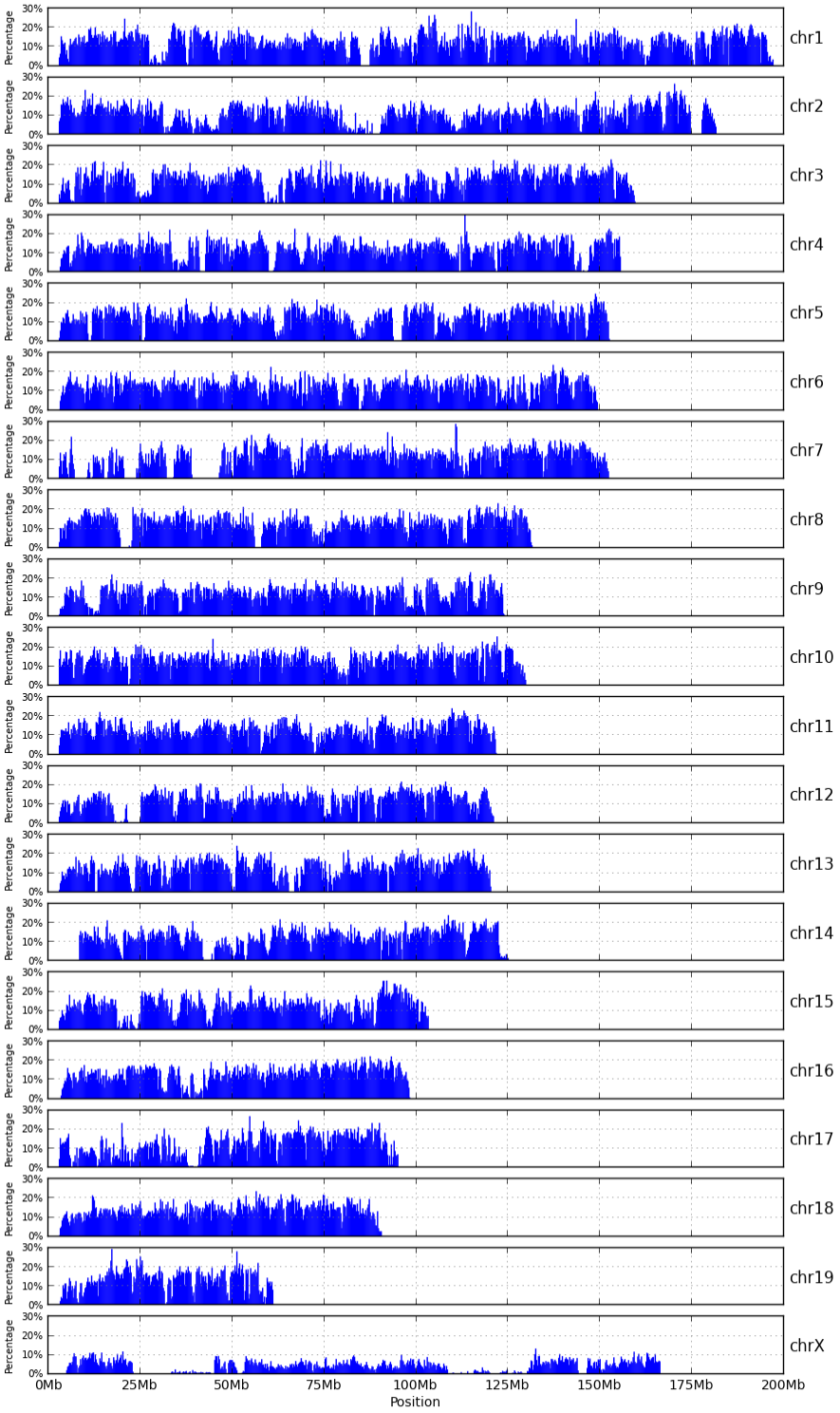


Figure 2.3: The bar charts of high-variant intervals of CAST pseudogenome. Each bar represents the percentage of 100bp windows within a 500Kb region that contain 3 or more sequence variations relative to the reference strain. Such regions present problems for most sequence aligners.

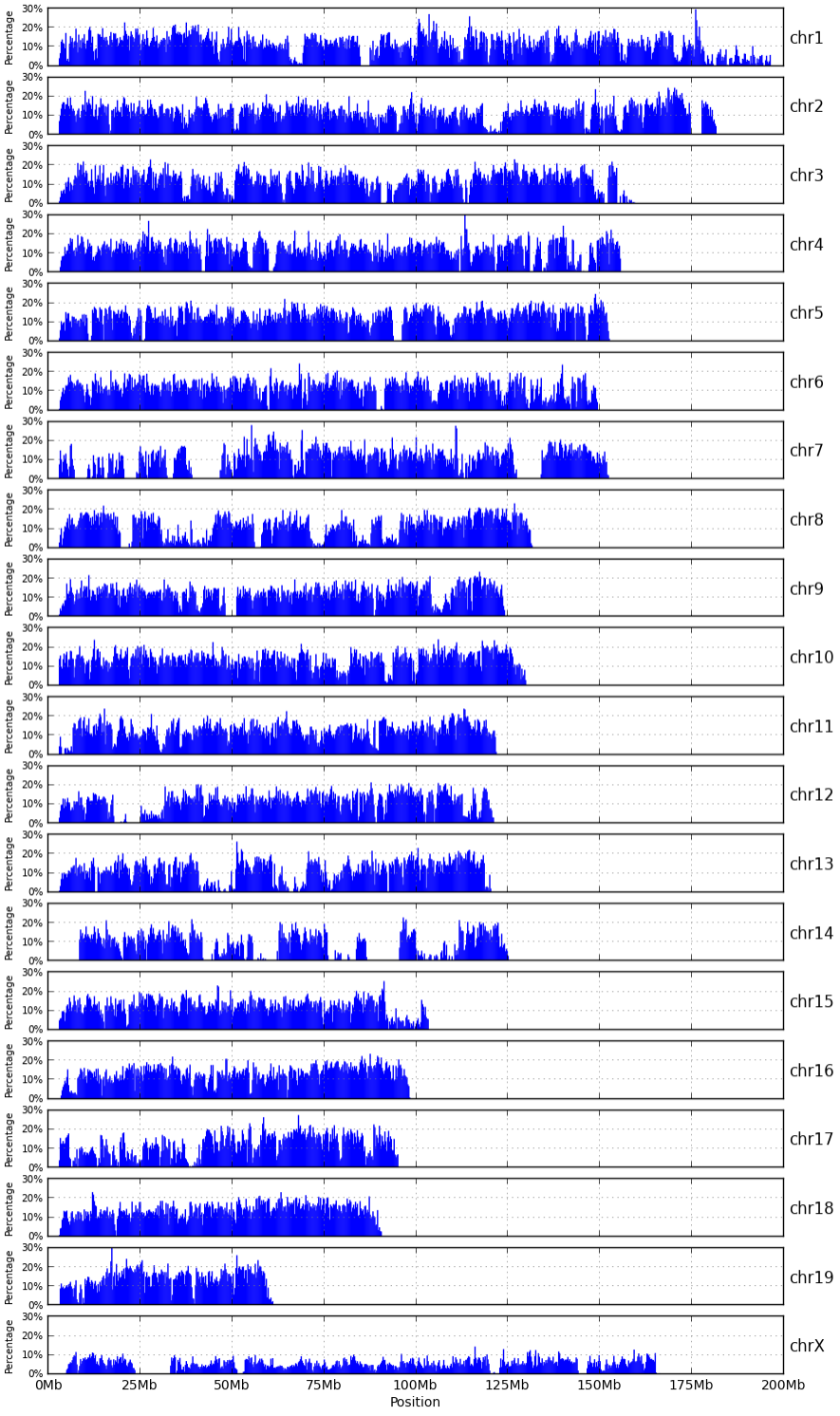


Figure 2.4: The bar charts of high-variant intervals of PWK pseudogenome.

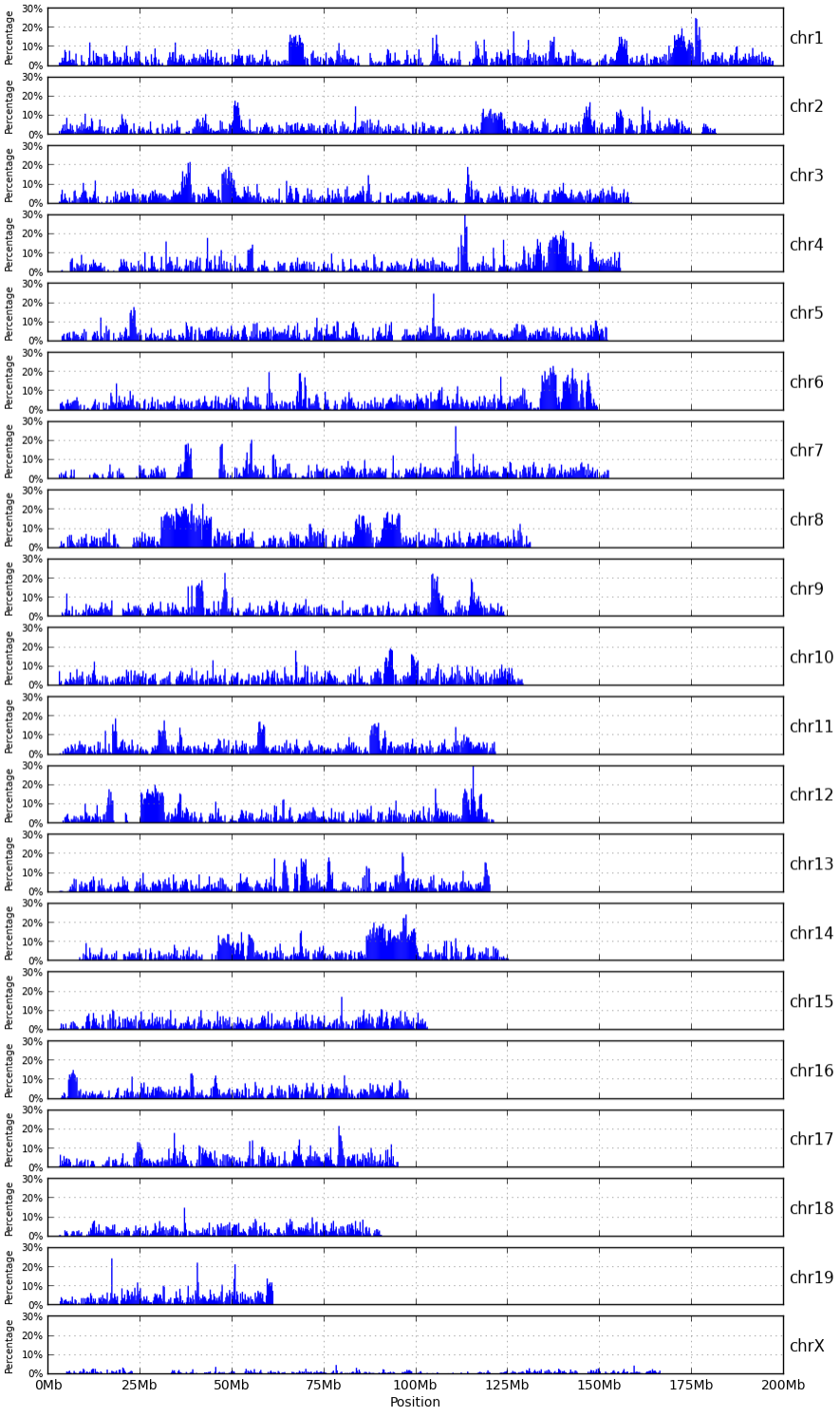


Figure 2.5: The bar charts of high-variant intervals of WSB pseudogenome.

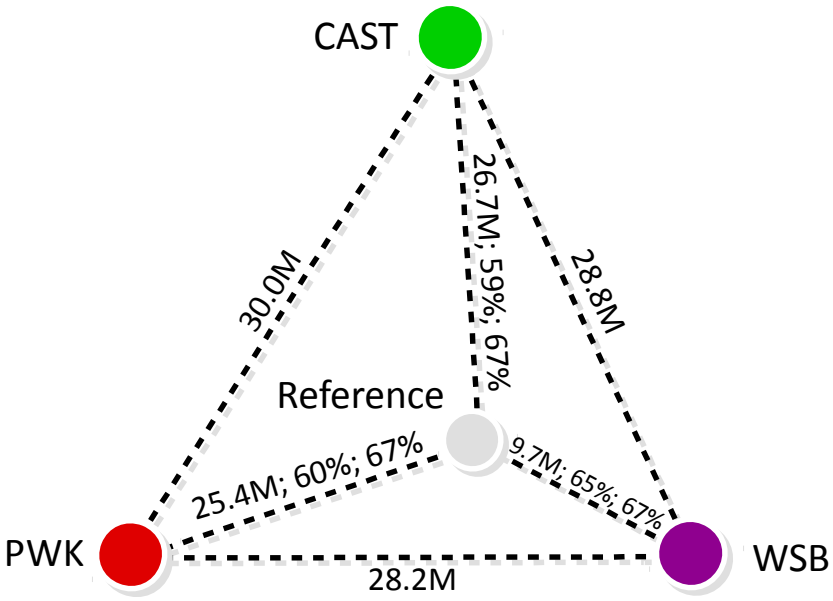


Figure 2.6: The estimated genetic distances between the reference and each of the three strains tested. The first number on edge label is the total number of MOD file instructions necessary to transform the reference into the target strain's pseudogenome. This number is proportional to the edit distance between the two genome sequences. The second number represents a typical percentage of reads from the target that can be uniquely aligned using the reference sequence. Notice that these numbers are inversely proportional to the strain's genetic distance from the reference, and they illustrate the so-called reference bias. The third number is a typical percentage of those reads that uniquely align to a MOD-file generated pseudogenome.

CHAPTER 3: NOVEL ALIGNMENT PIPELINES

In this chapter, I describe my pipelines for correcting reference bias on inbred and multi-parental sequencing data. For comparison purposes, I also consider a traditional analysis pipeline that employs a single reference genome and attempts to achieve similar annotations. In all fairness, this single-reference pipeline is only an approximation to the front-ends of other published methods. I have deliberately attempted to separate the annotation phase of sequence analysis from subsequent analyses. Assessments of the differential expression levels due to parent, allele, or slice variants are considered downstream uses of the annotations. I contrast my approaches with the representative reference-based pipeline and highlight their major differences.

3.1 Single Reference Pipeline

In traditional reference-based alignment pipelines, short-reads from high throughput sequencers are first mapped to a standard reference genome or sequence(Figure 3.1) and genetic variation is considered afterward. There are significant advantages in using the standard reference. In addition to supplying a common coordinate system for comparison between target genomes, reference coordinates anchor nearly all of the genome's functional annotations, such as gene/exon locations, transcription factor binding sites, and notations of common variants. When all samples are aligned to this reference, genomic comparisons are significantly simplified. However, the mappability to the reference genome is reduced if a sample has a large number of variations from the reference. This results in either a reduction in the number of reads mapped and/or an increase in mapping errors. If the number of errors exceeds the aligner's tolerance, the read will be simply dropped from the output and its information will be lost. In short, a sample that is genetically distant from the reference will typically align fewer reads and with reduced confidence than a sample that is closer to the reference.

If the reads did not come from an inbred organism, there is an extra step of annotating them after they are mapped. Specifically, the known allelic variations between the parental genomes are used to assign its origin. Consider a diallel cross of two inbred mouse strains as an example. If a mapping shares three SNPs

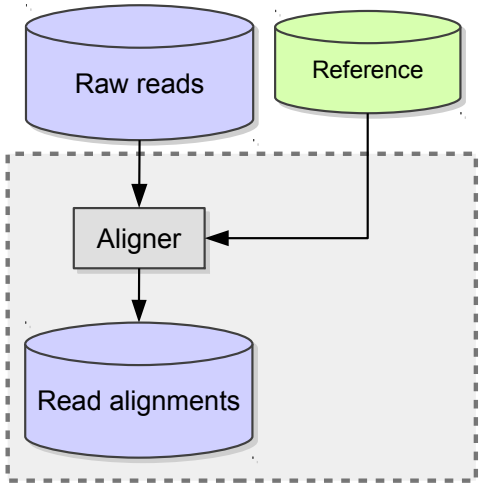


Figure 3.1: The single reference pipeline is the traditional way of read alignment. The standard reference sequence is directly used by the aligners for read alignment.

with the maternal strain but only one with the paternal one, the single-reference pipeline assumes that the mapping is from the maternal side. If such counts are the same, suggesting an equal chance of coming from either one, then the strain origin of this mapping cannot be determined.

3.2 Alignment Pipeline for Inbreds

Aligning read fragments to a reference genome or transcriptome sequence is subject to reference bias. Starting from inbred organisms, which is both simple and commonly used in experiments, I propose the following new alignment pipeline to correct reference bias.

Executing the instructions of a MOD file for an inbred organism, incorporates variants into the reference genome sequence to obtain a pseudogenome. Reads can then be mapped to the pseudogenome using an alignment tool such as Tophat (Trapnell et al., 2009) or Bowtie (Langmead et al., 2009; Langmead and Salzberg, 2012). The aligned read file (typically a BAM file) can then be remapped back to the reference genome's coordinates for analysis using the same MOD file. I developed a tool for this purpose called *Lapels*, which remaps the positions of the read fragment alignments, modifies their associated CIGAR string, and annotates the variants seen in each fragment (Figure 3.2) .

The reason for remapping alignments back to reference coordinates is twofold. First, there are abundant resources specified in the reference coordinate system, including databases of genetic variants, gene/exon annotations, and catalogs of other functional genomic elements. Second, many studies involve multiple

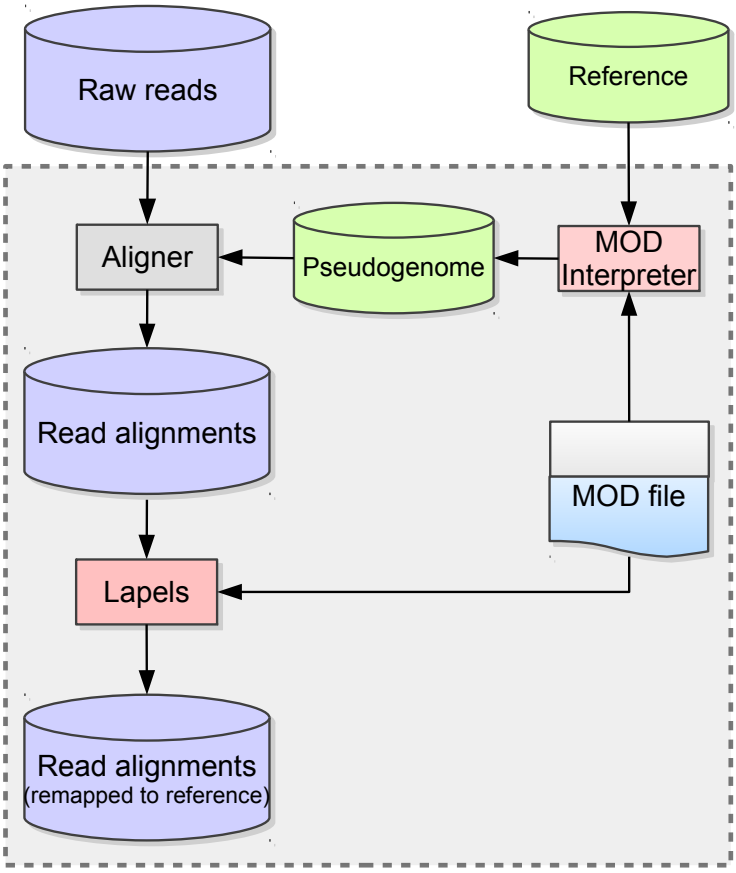


Figure 3.2: Inbred Alignment Pipeline. Instead of using the standard reference for alignment, the inbred pipeline takes the pseudogenome, which is constructed from the MOD file and the standard reference, as the reference and performs read alignment. After read alignments, *Lapels* will convert the positions of each reads from pseudogenome coordinates back to standard coordinates.

strains. It is convenient to have a common coordinate system so that comparisons between strains will become feasible.

3.2.1 Pseudogenome Construction

Given a MOD file, it is easy to transform one source sequence to another destination sequence. In my experiments MOD files were mainly derived from known genetic variants (SNPs, insertions, and deletions) of several wild-derived mice. Therefore, pseudogenomes for these mice were generated based on the mouse standard reference sequence and MOD files.

In the pipeline shown in Figure 3.2, the *MOD Interpreter* executes each instruction in a MOD file for an inbred and incorporates variants into the reference genome sequence to obtain a pseudogenome. The detailed procedure is shown below. The generated pseudogenome is then used in the alignment process instead of the

```
1: function BUILDSEQ(chrom, chromLen, srcSeq)
2:     instructionSet ← LoadInstructions(chrom)
3:     srcPos ← 0
4:     dstSeq ←''
5:     for all inst ∈ instructionSet do
6:         insPos ← inst.position
7:         if srcPos < insPos then
8:             dstSeq += srcSeq[srcPos : insPos]
9:             srcPos ← insPos
10:        end if
11:        Execute Instruction s on Position insPos
12:    end for
13:    if srcPos < chromLen then
14:        dstSeq += srcSeq[srcPos : chromLen]
15:    end if
16:    return dstSeq
17: end function
```

reference genome, so the end result of alignment is a BAM file (Li et al., 2009) using the coordinate system of the pseudogenome.

3.2.2 Pseudogenome Alignment and Annotation

Separate alignments create problems when comparing samples. If I tried to compare a CAST/EiJ inbred to a PWK/PhJ inbred using only the pseudogenome alignments, there would be two different genomic coordinate systems in play.

To alleviate this issue, I use the same set of known allelic differences incorporated into the pseudogenome to translate all of the mapped reads back to the reference coordinate system. This involves going through each mapped read and adjusting the mapping position, cigar string, and edit distance to match the reference genome instead of the pseudogenome. This step is done by a program called *Lapels* in my pipeline.

After that, each mapping is annotated with a series of tags to preserve information from the original pseudogenome alignment. In order to assess the mapping quality, each remapped read retains the cigar string and edit distance from the original pseudogenome mapping as tags. These tags allow us to calculate the original quality scores and preserve information regarding the differences between the reference and pseudogenome mappings for that read.

```
function REMAPREADS(chrom)
    readSet ← LoadReads(chrom)
    instructionSet ← LoadInstructions(chrom)
    regionMap ← BuildMapping(instructionSet)
    for all r ∈ readSet do
        Look up r.position in regionMap and get delta
        r.position += delta
        Save and adjust r.cigar, tag NM (edit distance)
    end for
    return readSet
end function
function BUILDMAPPING(instructionSet)
    srcPos ← 0
    delta ← 0
    regionMap ← {}
    for all inst ∈ instructionSet do
        Insert (srcPos → delta) into regionMap
        if inst.type is s-instruction then
            srcPos+ = inst.length
            continue
        else if inst.type is i-instruction then
            delta + = inst.length
        else if inst.type is d-instruction then
            srcPos+ = s.length
            delta − = inst.length
        end if
    end for
    Insert (srcPos → delta) into regionMap
    return regionMap
end function
```

3.2.3 Result on RNA-seq read alignment of inbred mice

The sequenced mouse samples were derived from the aforementioned wild-derived mouse strains. Over 1.2G reads were sequenced on the Illumina HiSeq 2000 platform of mRNA from the brain tissue extracted from 12 samples (4 samples per strain) using paired-end reads with 100 bp (2x100). The number of reads per sample is shown in Table 3.1 .

Table 3.1: Number of reads for 12 samples from 3 inbred strains.

Strain	Sample 1	Sample 2	Sample 3	Sample 4	Total
CAST	88,520,554	78,903,440	78,976,480	135,080,364	381,480,838
PWK	140,642,004	96,388,598	96,735,248	132,859,376	466,625,226
WSB	130,888,744	123,814,138	66,178,922	92,044,920	412,926,724

I used TopHat (v.1.4.0), with default parameter settings, to map reads to pseudogenomes derived from MOD files and the NCBI MGSCv37 *Mus musculus* reference genome.

After aligning, the read fragments from the resulting BAM files were remapped back to the reference genome and tagged with the number of observed variants (i.e., the number of variants incorporated in the pseudogenome and observed in the read). Based on the number of alignments in the resulting BAM file, we categorized each read into one of the three classes: unmapping (0), unique mapping (1), and multiple mapping (>1).

I also aligned the same reads to the standard reference genome and compared them to the reads mapped to the pseudogenome. The percentage of reads by category and the average percentages of biological replicates are shown in Table 3.2 .

Observe that more reads map uniquely to the pseudogenome than to the reference (shaded cells). The percentage increase is 7.46% for CAST, 6.76% for PWK, and 2.38% for WSB.

On one hand, the percentage of reads that uniquely map to the reference increases as we move from CAST (59.35%) to PWK (60.43%) to WSB (65.02%). This again accurately reflects the genetic distance of each strain from the reference (Figure 2.6) . The more different that a strain is from the reference, the fewer of its reads are mapped to reference genome, thus illustrating a reference bias.

In contrast, the percentage of reads that uniquely mapped to the pseudogenome is consistent among the three strains (around 67%). Thus, with pseudogenomes, the different strains are brought to comparable levels of mappability, which implies that the reference bias has been largely diminished. In all samples around 30%

Table 3.2: Average percentage of RNA-seq reads from CAST, PWK, and WSB samples mapped to the pseudogenome and the reference. (Tophat 1.4.0, Bowtie 0.12.8)

Alignment		**Reference**			
		Unique	Multiple	Unmapped	Total
CAST Pseudogenome	Unique	58.91%	0.10%	7.80%	66.81%
	Multiple	0.11%	2.31%	0.01%	2.43%
	Unmapped	0.33%	0.01%	30.42%	30.76%
	Total	59.35%	2.42%	38.23%	100.00%
PWK Pseudogenome	Unique	60.01%	0.09%	7.09%	67.19%
	Multiple	0.14%	2.37%	0.01%	2.52%
	Unmapped	0.28%	0.01%	30.00%	30.29%
	Total	60.43%	2.47%	37.10%	100.00%
WSB Pseudogenome	Unique	64.83%	0.04%	2.53%	67.40%
	Multiple	0.06%	2.47%	$< 0.01\%$	2.54%
	Unmapped	0.13%	$< 0.01\%$	29.93%	30.06%
	Total	65.02%	2.52%	32.46%	100.00%

of reads remain unmapped, I believe this residual represents current limitations of the aligner in combination with the remaining inaccuracies in the pseudogenome sequence.

Notice that the largest percentage increase is due to reads that were mapped uniquely to the pseudognome but unmapped to the reference (cells with pink shading). This suggests that my pipeline has rescued many reads that are discarded in the traditional method, especially when the strain is distant from the reference.

In order to understand how the embedded variants affect the alignment result, I investigated what percentage of reads have observed variants in the each categories and summarized the result in Table 3.3. Notice that categories involving unmapped reads in pseudogenomes are not included in the table, because such reads have no alignment and, thus, no observed variants of the strains.

Most of the reads that were mapped uniquely to the pseudogenome but unmapped in the reference alignments have variants: 88.74% for CAST, 88.08% for PWK, and 84.25% for WSB. Similarly, around 78% of the reads that were unmapped in the reference alignment, but mapped to multiple positions in the pseudogenome, contained a strain-specific variant. Such unmapped-to-multi-mapped reads, however, account for less than 0.01% of all aligned reads. Another group (0.05-0.1%) of the reads mapped to multiple positions in the reference alignment became unique mapping in the pseudogenome alignment owing to the added variants. In short, by incorporating the variants in the reference, I have provided the reads a better genome sequence to map to; such reads were neither thrown away nor misplaced by the aligner in the new pipeline.

Table 3.3: Average percentage of RNA-seq reads with observed variants from CAST, PWK, and WSB samples in each category.

Alignment		**Reference**			
		Unique	Multiple	Unmapped	Total
CAST Pseudogenome	Unique	21.50%	64.59%	88.74%	29.58%
	Multiple	74.99%	2.74%	78.80%	6.10%
PWK Pseudogenome	Unique	21.68%	61.38%	88.08%	28.85%
	Multiple	76.46%	4.01%	77.48%	8.22%
WSB Pseudogenome	Unique	7.79%	56.51%	84.25%	10.71%
	Multiple	75.96%	2.42%	78.37%	4.40%

Table 3.4: Comparison of mapping positions and CIGAR strings for reads uniquely mapped to both genomes.

Strain	Unequal Start	Equal Start but Unequal CIGAR	Equal Start and CIGAR
CAST	0.32% (82.43%)	0.41% (97.70%)	99.26% (20.98%)
PWK	0.32% (79.78%)	0.39% (97.41%)	99.29% (21.20%)
WSB	0.12% (79.03%)	0.20% (97.62%)	99.68% (7.53%)

Of those reads that mapped uniquely in both the reference and pseudogenomes, only a small fraction of them contain the strain-specific variants. This is to be expected since there are plenty of conserved regions in the genome. If reads originate from one of these regions, it is possible that they will be mapped uniquely to both genomes without any observed variant. Moreover, both alignments of this kind of reads should have the same mapping position, because I remapped the pseudogenome-aligned reads back to the reference coordinate system. In order to verify my hypothesis, I looked further into this category of reads by comparing the two alignments in terms of their positions and CIGAR strings. As shown in Table 3.4 , over 99% of reads in this category have the same mapping position and CIGAR string in both alignments, which justifies my previous assumption. I also observed that a large portion of remaining reads contain variants, the percentage of which is shown in parenthesis. Lastly, I found that for such reads fewer mismatches were seen in the pseudognome alignments than in the reference alignments, suggesting a better alignment result is achieved by using the pseudogenome.

As it is pointed out that the old version of Tophat/Bowtie does not tolerate gaps, I repeated the whole analysis pipeline using the recent releases of Tophat 2.0.6 and Bowtie 2.0.5 with default parameters.

Table 3.5: A comparison between Tophat 1.4.0 (with Bowtie 0.12.8) and Tophat 2.0.6 (with Bowtie 2.0.5)

Alignment	Aligner	**Tophat 1.4.0**			**Tophat 2.0.6**		
	Strain	CAST	PWK	WSB	CAST	PWK	WSB
Pseudogenome	mapping	69.24%	69.71%	69.94%	72.49%	72.44%	72.82%
	unique mapping	66.81%	67.19%	67.40%	69.41%	6927%	69.96%
	multiple mapping	2.43%	2.52%	2.54%	3.08%	3.17%	2.85%
	no mapping	30.76%	30.29%	30.06%	27.51%	27.56%	27.18%
Standard Reference	mapping	61.77%	62.89%	67.54%	67.20%	68.03%	69.32%
	unique mapping	59.35%	60.43%	65.02%	64.17%	64.96%	66.24%
	multiple mapping	2.42%	2.47%	2.52%	3.03%	3.07%	3.08%
	no mapping	38.23%	37.11%	32.46%	32.80%	31.97%	30.68%

Although I was able to map more reads using the newer version, the overall result is still consistent: pseudogenome alignments recover more reads and these alignments exhibit a reduced reference bias (Table 3.5). This suggests that different aligners can all benefit from my approach. My claims are that any aligner can significantly benefit from a reference sequence that better reflects the given sample, and that improving an aligner's tolerance to errors and gaps is no panacea.

3.2.4 Result on DNA-seq read alignment of inbred mice

My inbred alignment pipeline can also be used to align DNA-seq short reads. In this experiment, the DNA-seq data set was provided by the Wellcome Trust Sanger Institute (`ftp://ftp-mouse.sanger.ac.uk/current_bams/`), in which mouse strains were sequenced with Illumina HiSeq platform with over 40-fold coverage. Reads are 100bp paired-end.

I extracted the raw reads from the BAM file of CAST/EiJ and obtained 646,514,920 reads in total. Then I realigned them to both the standard reference genome and the CAST pseudogenome using Bowtie (v.2.0.5). Default parameter settings were used, and the aligner only reported the best alignment for each read. The comparison of read alignments to the pseudogenome and the reference genome is shown in Table 3.6 .

Most reads (over 95%) aligned to both genomes, which suggests that the aligner, in general, compensates for differences in genomic sequences. However, the pseudogenome, recovers about 0.70% more reads. Notice that this number is much smaller than the recovery rate of the RNA-seq experiments described in Section 3.2.3 (i.e., 7.48%). I speculate that the reason for this is twofold. First, without splice junctions, DNA-seq reads are intrinsically easier to align than RNA-seq reads. The 100bp DNA-seq reads have a higher

Table 3.6: Percentage of DNA-seq reads from CAST samples mapped to the pseudogenome and the reference.

Alignment		**Reference**		
		Mapped	Unmapped	Total
CAST Pseudogenome	Mapped	95.74%	0.74%	96.48%
	Unmapped	0.04%	3.47%	3.51%
	Total	95.78%	4.21%	100.00%

Table 3.7: Breakdown percentage of reads that mapped in both genomes. NM_1 and NM_2 are the edit distances from read alignments to the CAST pseudogenome and the reference, respectively.

	$NM_1 < NM_2$	$NM_1 = NM_2 = 0$	$NM_1 = NM_2 > 0$	$NM_1 > NM_2$	Total
Same Pos.	36.19%	30.70%	24.01%	0.10%	91.00%
Same Chrom. Diff. Pos.	1.78%	0.28%	0.59%	0.17%	2.81%
Diff. Chrom.	0.41%	0.56%	0.62%	0.35%	1.94%

mappability than the short fragments that are separated by exon junctions as in RNA-seq reads. Second, the tolerance of mismatches and gaps is different between Bowtie and Tophat. With default settings, as used in the experiments, Bowtie allows more mismatches than Tophat.

I further investigated the 95% of reads by grouping them based on alignment positions and edit distances. For each read, I compared the alignments to the pseudogenome with those to the reference genome. Alignments are considered to occupy the same position only if their starting positions and CIGAR strings are identical. The edit distance is represented by the NM tag of each alignment. I show the results in Table 3.7 . On one hand, 91% of the reads aligned to exactly the same position. While over 54% of the reads have no mismatches or an equal edit distance in the two alignments, about 36% of the reads aligned to the reference genome have a larger edit distance than the same read's alignment to the pseudogenome. It is worth mentioning that this percentage may vary for different aligner parameter settings and may also be influenced by noise. On the other hand, around 4.7% of the reads have different alignment locations. Since I have incorporated known variants into the pseudogenome, it is expected that the alignment positions in the pseudogenome are more accurate than those in the standard reference.

Although my pipeline did not directly rescue as many reads in the DNA-seq data as it did in RNA-seq data, it greatly increased the robustness and accuracy of the alignment results.

3.3 Multi-Alignment Pipeline for F1s

In this part, I move my experiments from inbred to another type of organisms, F1s, and present my pipeline for annotating multi-parental sequencing data. For the purpose of discussion, it is assumed that the data set being analyzed is RNA-seq from a two-founder diallel cross. The diallel produces samples from crossing two isogenic parental genomes. My pipeline is not limited to analyzing diallels, nor is it limited to RNA-seq analysis. The new pipeline consists of multiple alignments that incorporate all known genetic variants into a genomic model followed by annotation and merging.

One of the flaws with the single reference pipeline, applied to an entirely homozygous inbred sample, is the fact that the alignment process does not take into account known sequence differences between the given inbred and the reference. Instead, this is typically handled later during analysis or as a post-alignment annotation (as described in Section 3.1). One of the major differences in my new pipeline is to take advantage of known allelic differences as early in the pipeline as possible.

3.3.1 Pseudogenome Alignment and Annotation

Alignments are more difficult to assess for multi-parental crosses and diploid organisms in general because there are multiple sets of allelic differences to influence the alignment. In a diallel sample, there are two sets of allelic differences to take into account. I addressed this problem by constructing pseudogenomes for all contributing founder genomes, performing separate alignments of the full data set to each pseudogenome, and remapping them back to a reference genome while annotating differences as shown in Figure 3.3.

3.3.2 Merging - Comparing Alignments

Next, I considered all annotated alignments as input and merged them into a single output by choosing the best mapping for each read. This was performed by *Suspenders* in my pipeline (Holt et al., 2013). Suspenders sorts the BAM files by read name and systematically compares each alternative mapping. For each read, it extracts all mappings from each Lapels-annotated BAM file before comparing the results. As explained earlier, information from the original pseudogenome alignment is preserved by the annotation step.

The first step in the merge process is to identify the origin of each mapping. To do this, Suspenders first identifies **identical mappings** based on the mapping start position in the reference genome (chromosome and coordinate), its cigar string in the reference genome, its pairing (either paired or unpaired) flag, its

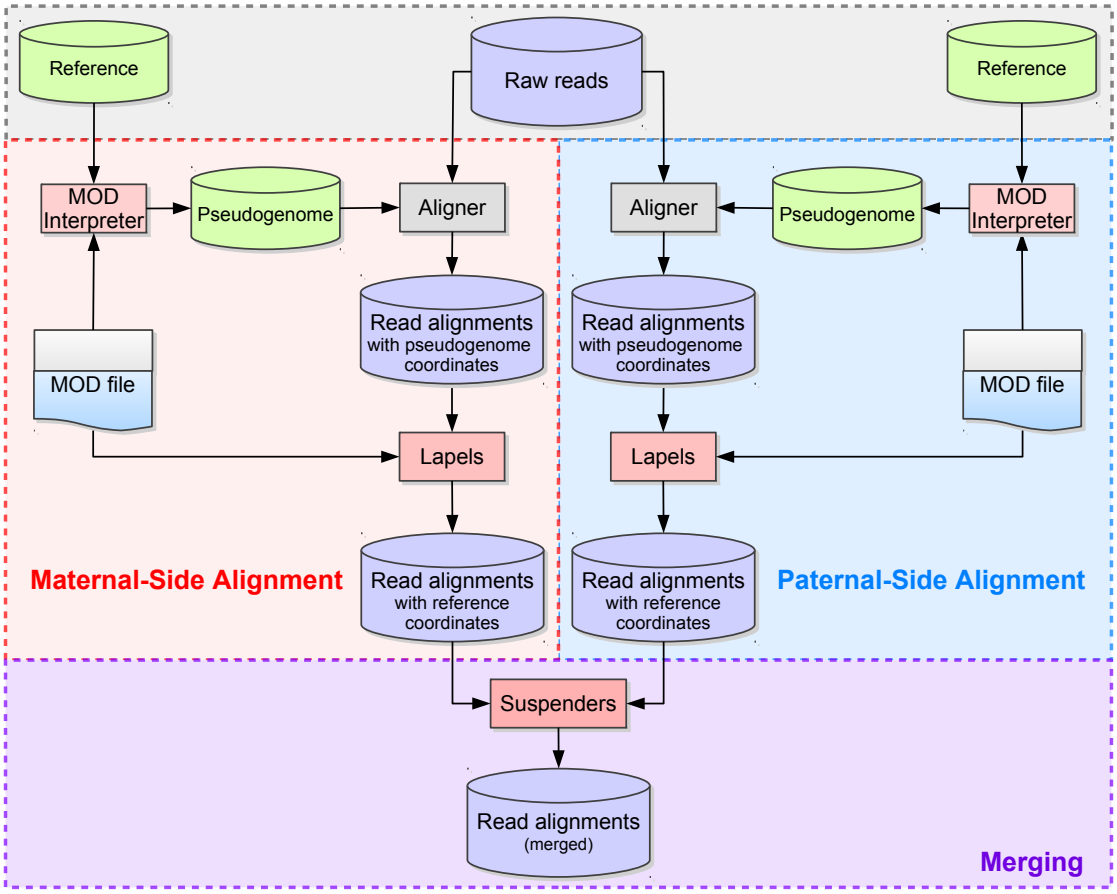


Figure 3.3: Multi-Alignment Pipeline for a diallel cross. Here it is assumed that the organisms being considered are diploid. The first step is to create two pseudogenomes using the list of known allelic differences, align the same reads to both pseudogenomes, and convert the mappings back to the reference coordinate system. Next I merge the two alignments and keep only the best mappings to either pseudogenome in the final merged file.

fragment end flag (either first or second), and its quality score (e.g., the Bowtie end-to-end score (Langmead, 2013)) from the pseudogenome mappings. The start position and cigar string assure that the two mappings in question cover the same genomic interval in the reference coordinate system. If the pairing and fragment end flags are also the same, it indicates that the read fragment was mapped in the same way in the separate pseudogenome alignments. The final criterion of comparing quality scores implicitly takes allelic differences into account and is discussed in greater detail later. If a read's mappings to two or more pseudogenomes are identical, Suspenders merges the mappings into one logical unit and tags it with a bit vector to identify the origin. For example, for a diallel sample, it would tag each mapping with a 2-bit flag set indicating its origin (01: first parent, 10: second parent, 11: either parent). Read mappings that uniquely map to a single pseudogenome are tagged according to their source (i.e. by setting a single bit).

For paired-end reads of a fragment, parental origins are "linked". The main idea is that if one read from a fragment can be assigned to a parent unambiguously, it is inferred that its mate also came from that same parent, even if the mate contains no informative variants. To illustrate the significance of linking, in the example single-reference pipeline (Section 3.1), each read is treated independently during the annotation step even if it is mapped as a paired-end read, thus requiring the presence of an informative allele in the read to assign its origin. This approximates the common process of examining alignments only at informative variant positions, while retaining the ability to detect if two variants fall on the same read. In my new pipeline, if two read mappings are properly paired, then they are treated as a single unit throughout the entire merging process. This means that 1) the mappings of two properly paired reads will be marked with the same parental origin, 2) the quality score will be calculated once based on the mismatches and indels from the paired mapping, and 3) the merge always prefers a paired-end mapping to one or two single-end mappings from the pair.

3.3.3 Merging - Filter Pipeline

Entering into the filtering section of the pipeline, one has a set of possible mappings for a given fragment where each mapping is marked with an origin flag. By sending the set through a series of filters, one removes mappings from the set until only one possible mapping remains for each fragment. Prior to filtering, Suspenders checks to see if any of the possible mappings are mated paired ends. If so, it immediately removes all unpaired mappings from consideration since a paired mapping is preferred over an unpaired one. If there are no paired-end mappings, the mappings are grouped depending upon whether they are the first or second read from the fragment, and a mapping set from each read is independently sent through the Suspenders filters. Note that the two unpaired ends of the same fragment may be filtered in different ways because they are handled separately.

The next step is to send the mapping sets through a series of three filters (shown in Figure 3.4): Unique, Quality, and Random. If a mapping is output by a filter, additional annotations will be added to indicate the chosen mapping as coming from that filter. The Unique filter identifies the reads whose mapping sets contain a single mapping and outputs these mappings. These include all reads having a unique mapping to only a single pseudogenome, and reads mapping to multiple pseudogenomes with single identical mapping as defined previously. In short, the Unique filter outputs simple cases where a read (or two paired reads) has one unique mapping across all pseudogenomes. In Figure 3.4, reads 1, 2, and 3 each have a single unique positional and score mapping, so they are all output by the Unique filter.

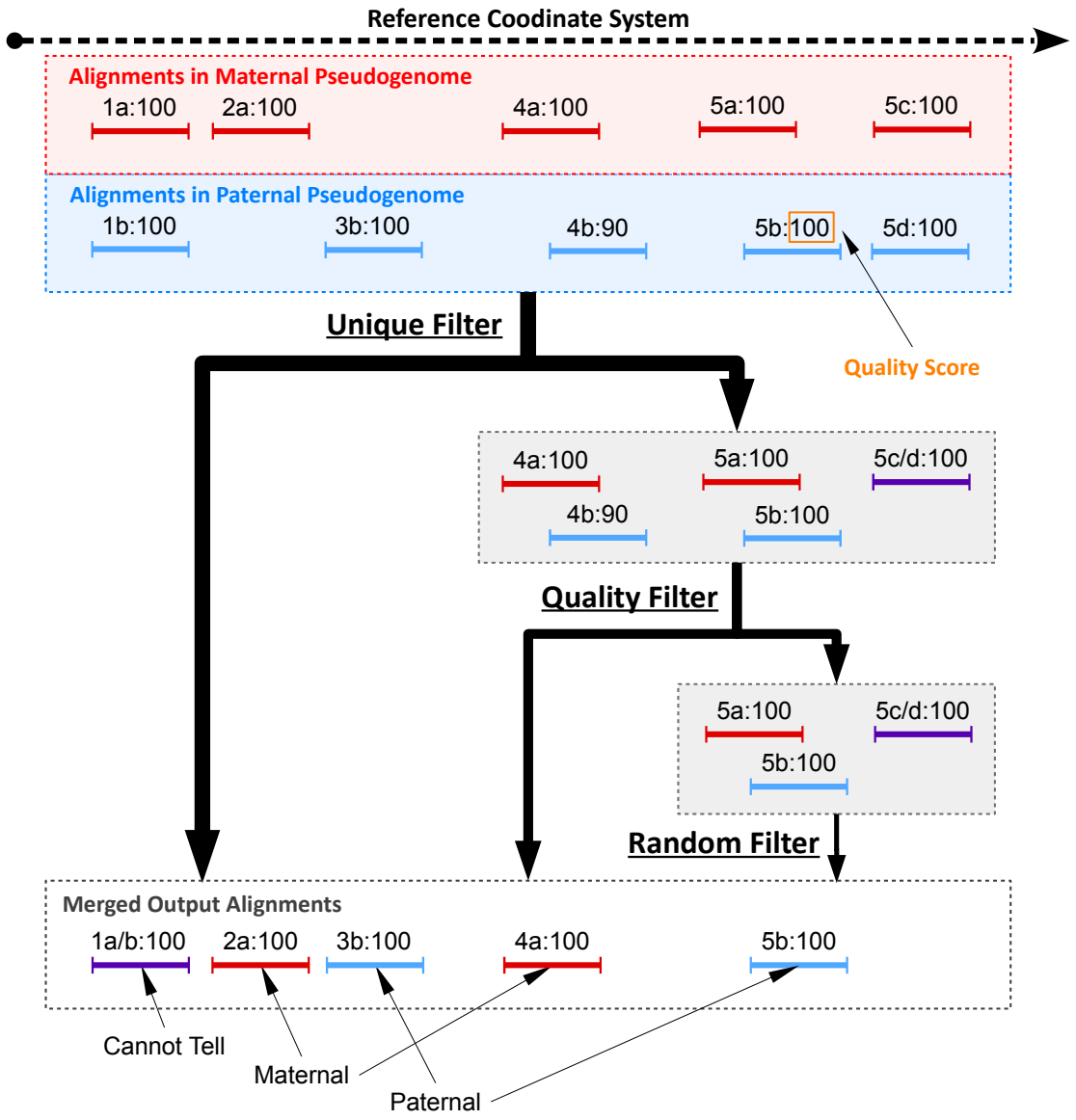


Figure 3.4: Sample filter path for mapping sets of five reads in a diallel cross labeled such that "4b:90" is mapping "b" of read 4 with a score of 90. Prior to filtering, 1a and 1b are combined into 1a/b because they have the same position and score in both mappings. Additionally, 5c and 5d are also combined. The mapping sets of reads 1, 2, and 3 are all output by the Unique filter since there is a single positional mapping for each. The mapping sets of reads 4 and 5 have multiple mappings, so they are diverted to the Quality filter by the Unique filter. The Quality filter outputs 4a since only one mapping of read 4 has the best score (100 compared to 90). The mapping set of read 5 has three mappings with identical scores and is therefore diverted to the Random filter which chooses one mapping arbitrarily. Since there are three in the set, each one has a 33.3% chance of being chosen, with 5b being the arbitrary choice in this example.

As mentioned earlier, the score comparison is where this pipeline implicitly takes into account allelic variations in the sample. An aligner typically uses a quality score to quantify the mapping quality which

is a function of the number of mismatches, insertions, and deletions. Only the mapping(s) with the best score are output. For example, TopHat uses the Bowtie scoring scheme (Langmead, 2013) when reporting possible mappings(Langmead and Salzberg, 2012; Langmead et al., 2009; Trapnell et al., 2009). Assume that a read aligns to multiple pseudogenomes that straddles an informative variant caused by a SNP. The mappings to pseudogenomes with the matching variant will have fewer mismatches than that to genomes with the alternate allele. Since sequencing errors are attributes of the read, they contribute mismatches equally to all pseudogenome mappings. In places with no informative alleles, an aligner will report mappings to all genomes with identical number of mismatches. Additionally, if there are multiple variants under a read's mapping, the read may be mapped to multiple positions in the genome, but usually only the best mappings are reported. The Quality filter attempts to simulate this behavior by keeping only the best mappings and their corresponding references. Prior to the filters, identical mappings (with same coordinates and scores) were combined into a single unit. However, the mappings were not combined if their scores were different. The Unique filter treats them as two different mappings from distinct origins and passes them to the Quality filter.

Read fragments with multiple mappings (possibly to the same position) are passed to the Quality filter. For each read fragment, I regenerated the original scores of the pseudogenome mappings from the stored cigar strings and edit distances saved during the annotation phase when remapping back to the reference. If only one mapping has the best score, then that mapping is output by the Quality filter. In Figure 3.4, the mapping set of read 4 has two different mappings (one from maternal and one from paternal). The maternal mapping has a higher quality score, so it is output by the Quality filter.

However, if multiple mappings of a mapping set have the same best score, then those mappings are passed to the Random filter as a last resort. The Random filter will choose only one from the set at random to keep in the merged result. Note that each chosen mapping is tagged with the filter it came from, so the option to remove all mappings from the Random filter can be performed in downstream analysis. In Figure 3.4, the mapping set of read 5 has three possible mappings with identical scores. Each mapping has a 33.3% chance of being chosen for the final output. After each read fragment has been processed using the filter pipeline, the final result is a single merged file containing at most one mapping for each single-end read and at most one paired mapping or two unpaired mappings for each paired-end read. Additionally, each read mapping is tagged to identify its pseudogenome origin and the filter that output it during the merge process.

Strain	CAST × PWK	PWK × CAST
Sample1	115,936,064	119,926,340
Sample2	87,988,306	90,706,788
Sample3	142,479,432	170,423,066
Sample4	137,698,560	92,829,168
Sample5	137,953,398	93,801,072
Total	622,055,760	567,686,434

Table 3.8: Total number of reads for 10 samples from two F1 hybrid crosses.

3.3.4 Result on RNA-seq read alignment of F1 mice

I evaluated my pipeline using a full diallel cross of CAST/EiJ and PWK/PhJ. I use the notation of CAST × PWK to represent the cross whose maternal and paternal parents are CAST/EiJ and PWK/PhJ, respectively. Likewise, the reciprocal cross is denoted by PWK × CAST.

The mRNA were first extracted from brain tissues of 10 female samples (5 for each cross). Then Illumina HiSeq 2000 platform was used to sequence the transcribed cDNA and obtained around 1.2G paired-end reads with 100bp (2x100). The number of reads per sample is shown in Table 3.8.

I followed the similar steps shown in Section 2.3.3 to generate MOD files for both CAST and PWK. Then I ran several tools with their default parameter settings in both pipelines. RNA-seq reads were aligned against the genomes by Tophat (v2.0.5). In the multi-alignment pipeline, I used Lapels (v1.0.4) to map each read coordinate back to the reference, and Suspenders (v0.2) to merge and tag mappings. Only one mapping per read was reported in the final output.

3.3.4.1 Comparison of Mapping Ratio

I examined the fraction of mapped reads from alignments to pseudogenomes, and compared it to the fraction of the same reads when mapped to the standard reference genome. Note that this is an imperfect comparison, since I considered only whether a read maps without considering the accuracy or quality of the mapping. The mapping ratios are shown in Figure 3.5.

Observe that more reads are mapped to each parental pseudogenome than to the reference. The percentage gain is about 3% for both pseudogenomes in the two crosses. A similar increase can be seen in the percentages of uniquely mapped reads. This suggests that by integrating the variations of parental strains into the reference, I have obtained two better genomes for the reads to align to.

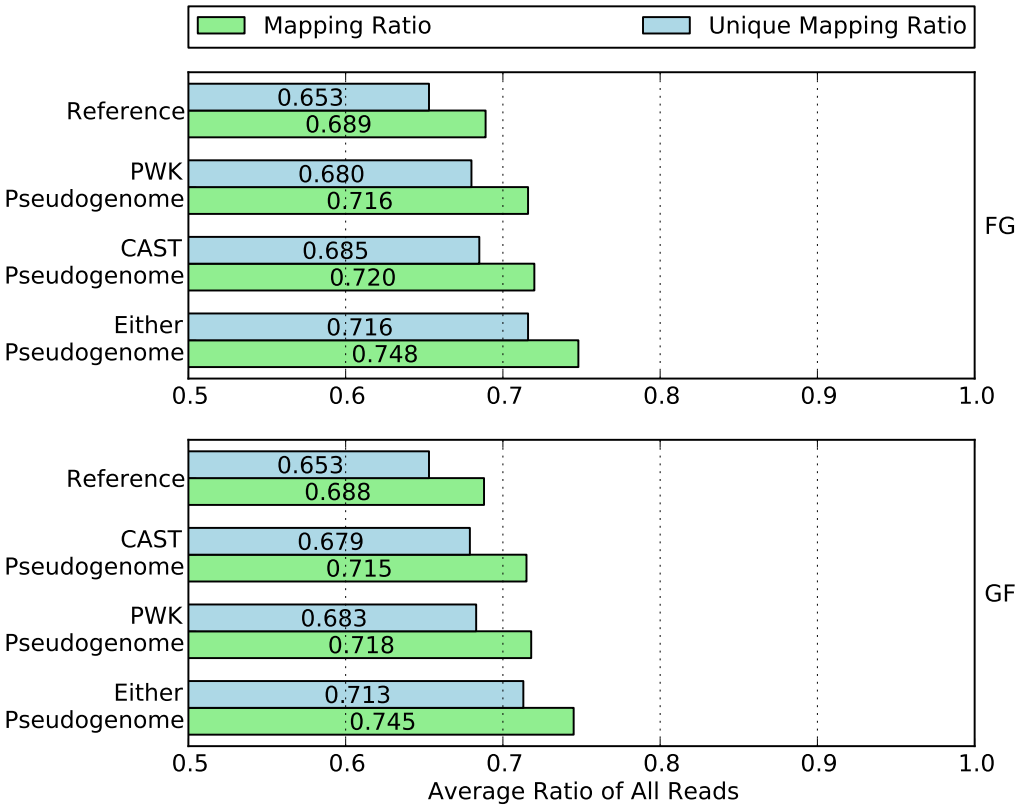


Figure 3.5: Mapping ratio and unique mapping ratio of reads to the reference genome, two pseudogenomes, and either pseudogenome. Using a single pseudogenome provides a gain of approximately 3% over the reference genome and using both almost doubles the gain to 6%.

If one consider whether reads mapped to either or both pseudogenomes, the combined recovery rate gain almost doubles to around 6%. To take advantage of this gain, I used a merge process to combine the two sets of alignments in the following section.

3.3.4.2 Comparison of Parental Origin Labeling

After reads are mapped, my new pipeline and single-reference pipeline next attempt to label the pseudogenome origin of every read where possible. This is a crucial step for downstream analyses which leverage the labels to determine differential gene expression between the parental strains.

Since common aligners can allow a small amount of mismatches during alignment, reads containing SNPs may still be mapped to the reference. After alignment, the single reference pipeline can check every SNPs positions in the reads and count the numbers of maternal alleles and paternal alleles. Then the ratio of the two allele counts is used to determine the parental origin of each reads. If no maternal and paternal

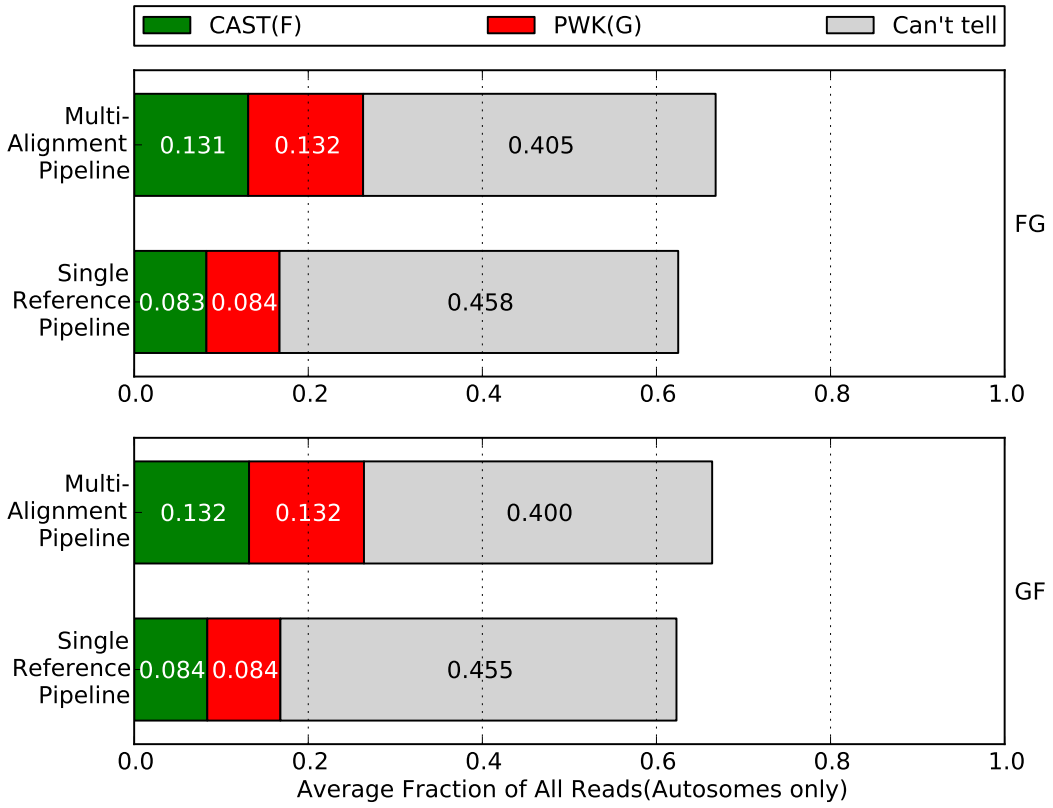


Figure 3.6: Percentage of parental origin labels in the single reference pipeline compared to the multi-alignment pipeline. The single reference labels were generated after alignment by checking the alignment for strain specific alleles. Note that I removed non-autosome mappings, multi-mapped reads, and reads filtered by the Random filter. In general, each strain category gains about 5% in the multi-alignment pipeline which can be broken down into about 2% from reads that did not map and about 3% which were "Can't tell" in the single reference pipeline.

alleles are observed or both counts are the same, it will be classified into the "can't tell" category. Otherwise, I choose to label the mapping according to which the allele count is greater.

In my proposed pipeline, the label for each mapping is determined during the merging stage after considering the mappings to multiple pseudogenomes. This process takes place in three filtering stages whose details were discussed in Section 3.3.3.

I compared the performance of both pipelines in labeling read origins. To reduce bias between two crosses, I only used reads that mapped to the autosomes. The biases were introduced by mitochondrial RNA expression, which is entirely of maternal origin, and a skewing of the X-inactivation ratios in heterozygotes, which prefers genes expressed from the CAST/EiJ chromosome(Chadwick et al., 2006). Furthermore, reads with multiple mappings were discarded by the single-reference pipeline, which is a common strategy

		Single Reference Pipeline				
		CAST	PWK	Cannot Tell	Others	Total
Multi-Alignment Pipeline	CAST	7.95%	0.12%	2.89%	2.13%	13.09%
	PWK	0.12%	8.03%	2.91%	2.10%	13.16%
	Cannot Tell	0.03%	0.03%	39.08%	1.33%	40.47%
	Others	0.18%	0.19%	0.88%	32.03%	33.28%
	Total	8.28%	8.37%	45.76%	37.59%	100.00%

Table 3.9: Parental origin of reads comparison for the two pipelines from CAST $\times$ PWK samples. Note that non-autosome mappings, multi-mapped reads, and Random filtered reads are in the "Others" category. The diagonal represents reads labeled with the same origin in both pipelines. 5.8% of reads were "Cannot Tell" in the single reference pipeline but labeled as either CAST or PWK in the multi-alignment pipeline. Additionally, 4.23% were in the "Other" category for the single reference, but labeled as either CAST or PWK in the multi-alignment pipeline. These two values are compared to the .06% and .37% of reads marked as CAST or PWK in the single reference pipeline, but labeled as "Cannot Tell" or "Other" in the multi-alignment pipeline. The net result is an increase of about 10% for the number of reads the multi-alignment pipeline could assign a parent of origin over the single reference pipeline. Similar results for the reciprocal cross are shown in Table 3.10.

		Single Reference Pipeline				
		CAST	PWK	Cannot Tell	Others	Total
Multi-Alignment Pipeline	CAST	8.09%	0.12%	2.96%	2.05%	13.22%
	PWK	0.12%	8.08%	2.95%	2.01%	13.16%
	Cannot Tell	0.03%	0.03%	38.73%	1.25%	40.04%
	Others	0.17%	0.18%	0.87%	32.26%	33.48%
	Total	8.41%	8.41%	45.51%	37.57%	100.00%

Table 3.10: Parental origin of reads from PWK $\times$ CAST samples. Note that non-autosome mappings, multi-mapped reads, and Random filtered reads are in the "Others" category.

described by many researchers (Degner et al., 2009; Keane et al., 2011; Cumbie et al., 2011; Missirian et al., 2012). To make the comparison more fair, I ignored the mappings output by the Random filter in my pipeline. Any mappings output by the Random filter were treated as unmapped reads in the figures and tables. The percentage of reads in each category is shown Figure 3.6.

While only (4.3%) more reads are processed in my method than in the traditional one, there is higher percentage of reads assigned to a unique parent of CAST or PWK. Specifically, approximately 5% of reads are gained for each parental category, while the reads in the "can't tell" class are reduced by over 5%.

To better understand the results, I investigated the relation between categories in the two pipelines. The results of the two reciprocal crosses are shown in Tables 3.9 and 3.10, respectively.

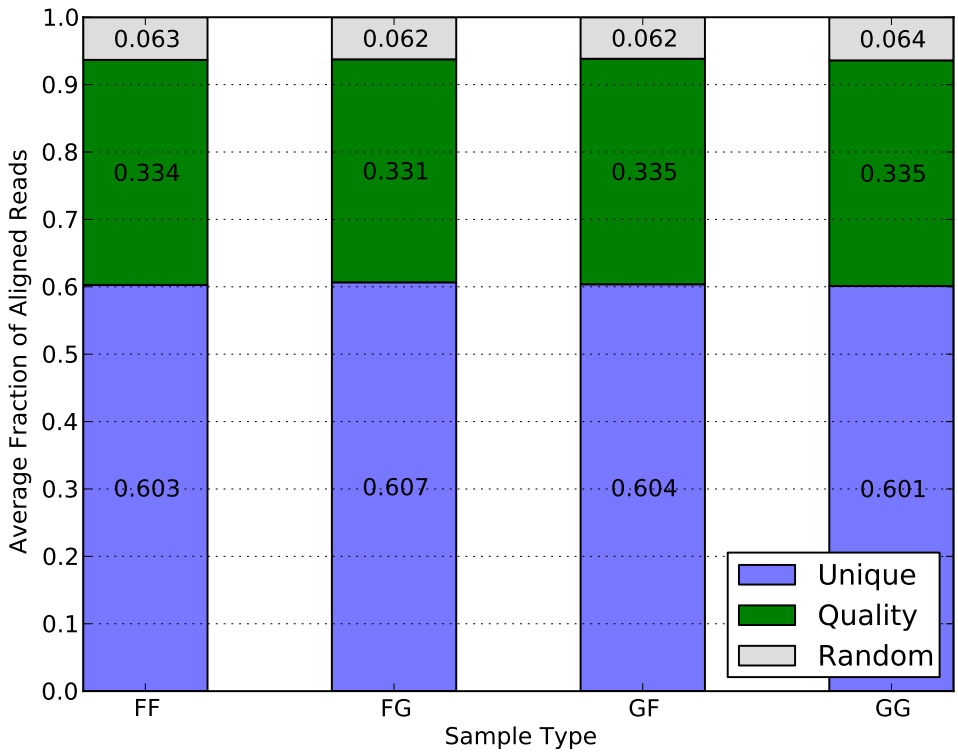


Figure 3.7: Average filter distribution of mapped reads for diallel samples in the multi-alignment pipeline. F and G denote CAST and PWK respectively. Since each category is approximately equal, it suggests that there is no inherent bias to a strain caused by the filters.

On one hand, a large portion of reads in the CAST category of the single-reference pipeline were assigned to the same category in the multi-alignment pipeline. The percentage is around 96% for both crosses. The same percentage can be seen in the PWK category as well. This reflects that most reads with non-trivial labels in the traditional single-reference method are covered in the corresponding categories in my approach.

On the other hand, the previous 5% increase in CAST and PWK categories of my method can be attributed to the (2%) reads that cannot be aligned to the standard reference and the (3%) reads whose parental origin cannot be determined using the traditional method. This is to be expected since my approach utilizes a merged set of alignments and leverages more information, such as quality score and linking, to decide the origin labels.

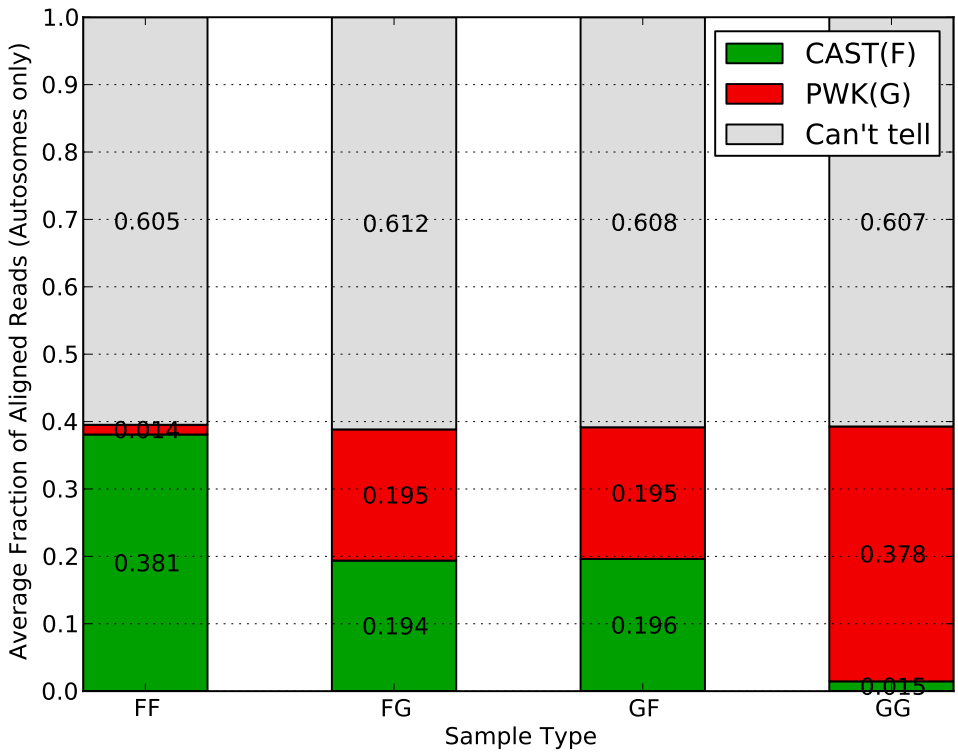


Figure 3.8: Average parent-of-origin distribution of autosome mapped reads for multiple sample types in the multi-alignment pipeline. In all sample types, approximately 60% are can't tell with the remaining 40% divided into parent categories. For inbreds, the vast majority (38%) are from the associated parent category with only 2% having a better mapping to the other pseudogenome. For the F1 hybrids, the categories are roughly equal which is expected for the autosomes.

3.3.4.3 Performance of Merging

In order to evaluate the accuracy and consistency of the merging procedure, I applied the same multi-alignment pipeline to inbred samples of CAST and PWK, pretending that they were F1 hybrids crosses (i.e. I performed alignments to both pseudogenomes, annotated reads and remapped them back to the reference, and merged the results). These inbred strains can be considered as the negative controls.

Figure 3.7 shows the percentage of mapped reads that are output by each of the three filters. Around 60% of reads are merged in the Unique filter step, suggesting they have either unique mappings in one of the pseudogenomes or identical mappings in both of them. Another 33% reads have multiple pseudogenome mappings with one mapping better than the rest, so they are filtered by quality score. The remaining 6% have multiple mappings with identical quality scores, so one was randomly chosen to be reported in the final

output. The consistency of filtering percentages in different strains, including the hybrid crosses and inbreds, suggests that the filters in the merging process do not bias the result.

In Figure 3.8, I show the percentage of mapped reads in each of the parental origin categories. To avoid the bias caused by the X chromosome and mitochondria, only reads mapping to autosomes were considered in this analysis. Around 60% of reads fall into the "Can't tell" class, and this percentage is consistent in F1 hybrids (the second and the third) as well as negative controls (the first and the last). For the residual 40%, we can see the ratio of CAST reads to PWK reads is 1:1 for the F1 hybrids, which is expected because reads are equally likely to come from either parent in autosomes. The ratio for inbred strains, however, is quite different. In fact, the majority of the 40% are classified to the corresponding inbred strain, while only 1.4%-1.5% are mislabeled. This error rate is likely caused by sequencing noise or unannotated parental alleles.

CHAPTER 4: HANDLING REMAINING BIAS

Although the alignment pipelines discussed in Chapter 3 can correct the majority of reference bias, the variation of read depth after alignment is observable along the genome. This variation can be very common due to random noise or some context-dependent distribution, such as GC content, gene expression levels in RNAseq, etc. However, it can also be a sign of remaining bias that we cannot correct during alignment. In this chapter, I investigate this problem based on the alignment of multiple samples derived from the same set of founder organisms.

I attribute the remaining bias to two causes. First, the pseudogenome sequences are imperfect. Wrongly annotated variants (mainly SNPs and indels) may be integrated in the pseudeogenomes, and a certain amount of structure variations can be missing. In this case, the variation of read depth is consistent among the majority of samples. That is to say, we are able to see an increase/descrease pattern of read depth shared among samples. Second, there are genomic events (e.g. point mutations, copy number variations) that affect the local read depth of some samples. This kind of bias is only seen in a sample or a very small subset of samples.

To analyze the remaining bias, I proposed a probabilistic model for common trend estimation and outlier detection in read depths of multiple samples. Discovering the underlying common trend enables us to locate the wrong annotations or missing variants, while finding outliers helps us to identify sample-specific variations.

4.1 Preliminary

In this chapter, it is assumed that reads from different samples are aligned to the same reference sequence(s). The region of interest is divided into equal-width bins and the number of reads is counted for each bin and sample. To avoid counting reads multiple times, a read is assigned to a bin only if the center of this read falls in the bin boundaries.

Assuming the number of samples is S, and the number of bins in each sample is N, let $\mathbf{X}_j = (\mathbf{x}_{1j}, \mathbf{x}_{2j}, \ldots, \mathbf{x}_{Nj})^T$ be the random vector of read counts for sample j $(j = 1, 2, \ldots, S)$, and $\mathbf{Y}_j = (\mathbf{y}_{1j}, \mathbf{y}_{2j}, \ldots, \mathbf{y}_{Nj})^T$ be the random vector of scaled read counts corresponding to $\mathbf{X}_j$. Denote the scale factor between $\mathbf{X}_j$ and $\mathbf{Y}_j$ by β_j, i.e.

$$\mathbf{Y}_j = \beta_j \mathbf{X}_j. \tag{4.1}$$

Notice that β_j is a sample-wise scalar parameter and is introduced to balance the different read coverage of samples.

For simplicity, I assume the only disturbance factor is an additive Gaussian noise. Notice that counts are usually modeled with a Poisson distribution or a negative binomial distribution in many literatures. If the counts are not small, we can use a Gaussian distribution to approximate. Below is the model for $\mathbf{Y}_j$:

$$\mathbf{Y}_j = \boldsymbol{\mu} + \boldsymbol{\epsilon}_j \tag{4.2}$$

where $\boldsymbol{\epsilon}_j = (\epsilon_{1j}, \epsilon_{2j}, \ldots, \epsilon_{Nj})^T$ is a vector of random noise with ϵ_{ij} $(1 \leq i \leq N)$ i.i.d. and Gaussian with mean 0 and variance σ^2, and $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_N)^T$ is the expected value of $\mathbf{Y}_j$. The parameter vector $\boldsymbol{\mu}$ is shared among samples, and thus it is also called the **common trend**.

If every sample has the same read coverage, then $\beta_1 = \beta_2 = ... = \beta_S = 1$ in Equation (4.1) and the Maximum Likelihood Estimate(MLE) of $\boldsymbol{\mu}$ is given by the average of observed read counts across samples: $\hat{\boldsymbol{\mu}} = \frac{1}{S} \sum_{j=1}^{S} \mathbf{X}_j$.

Despite the straightforward solution, the above assumption is not often seen in the real-world experiment setting. First, read coverage varies among different samples. Such variation may be introduced during sample preparation and/or sequencing. Second, in addition to random noise, the read counts may also be disturbed by other factors. For example, a sample-specific copy number gain (or loss) in some region may increase (or descrease) the local read depth; a point mutation occured in one sample may lead to more (or less) reads aligned to that local region. Such unexpected variation of read depth in isolated samples is considered as an **outlier** event. While random noise exists in every region across all samples, outliers are only seen at few locations of few samples.

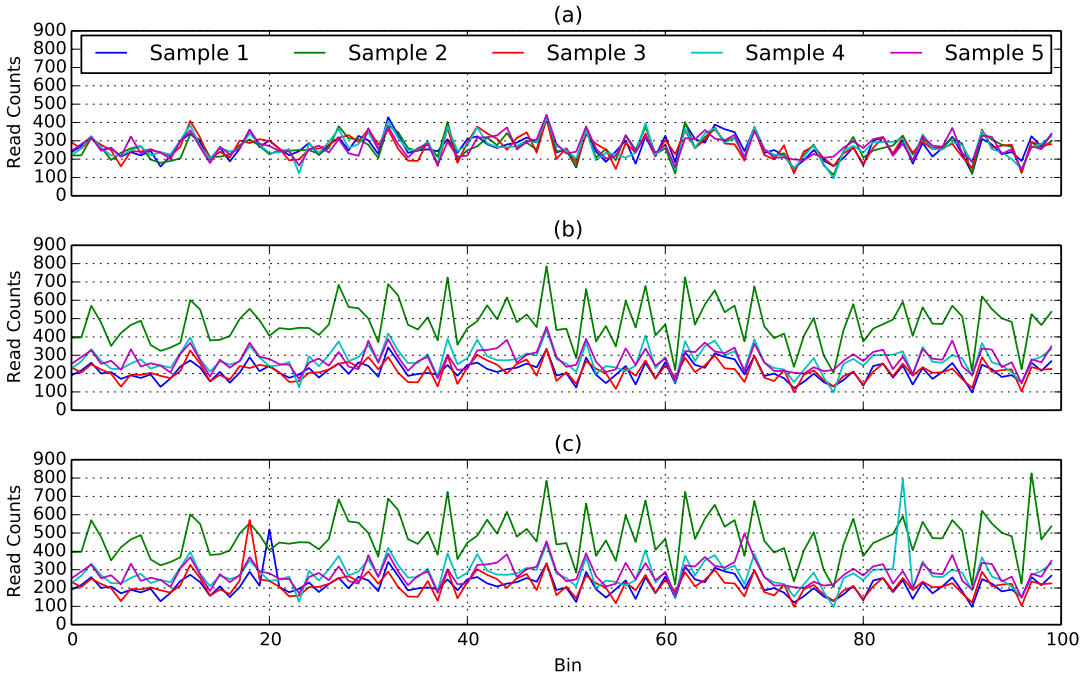


Figure 4.1: An example of three synthetic data sets under different assumptions: (a)the same read coverage across samples, (b)different read coverages for different samples, and (c)different read coverages for different samples with outliers.

To better illustrate common trend and outliers in read counts, Figure 4.1 shows three synthetic data sets. At first, samples have the same read coverage and there is no outlier. In such case, the common trend, i.e. the consistent pattern of read counts, is clear and easy to find. The second data set is similar to the first one, except that the read coverage varies among samples. In the third data set, I randomly added outliers in the read counts, which makes it hard to determine the shared pattern and isolated points in the data. In the remainder of this chapter, the data sets in my experiments are comparable to the third one, and I used my proposed model to simultaneously estimate the common trend and identify outliers among multiple samples with different read coverages.

4.2 Probabilistic Model for Common Trend and Outliers

To take outliers into account, I introduced term $\mathbf{\Gamma}_j$ to the model in Equation (4.2). $\mathbf{\Gamma}_j = (\gamma_{1j}, \gamma_{2j}, \ldots, \gamma_{Nj})^T$ is a parameter vector that captures unexpected increase or decrease of read counts for sample j.

Notation	Meaning
S	the number of samples
N	the number of bins in a sample
$\mathbf{X}_j$	the random vector of read counts from sample j
$\mathbf{Y}_j$	the random vector of scaled read counts from sample j
$\mathbf{x}_{ij}$	the read count of bin i from sample j
$\mathbf{y}_{ij}$	the scaled read count of bin i from sample j
$\boldsymbol{\mu}$	the expected value of all $\mathbf{Y_j}$ $(1 \leq j \leq S)$
β_j	the parameter of scale factor for sample j
$\boldsymbol{\Gamma}_j$	the parameter vector of outlier indicator for sample j
$\boldsymbol{\gamma}_{ij}$	the parameter of outlier indicator of bin i for sample j
σ	the standard deviation of the random noise
λ	the regulation penalty of ℓ_1-norm

Table 4.1: Notation Table

I proposed the following probablistic model:

$$\mathbf{Y}_j|\boldsymbol{\mu}, \boldsymbol{\Gamma}_j \sim \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{\Gamma}_j, \sigma^2\mathbf{I}_N). \tag{4.3}$$

Plugging Equation (4.1) into Equation (4.3), I could get

$$\mathbf{X}_j|\boldsymbol{\mu}, \beta_j, \boldsymbol{\Gamma}_j, \sim \mathcal{N}\left(\frac{\boldsymbol{\mu} + \boldsymbol{\Gamma}_j}{\beta_j}, \left(\frac{\sigma}{\beta_j}\right)^2 \mathbf{I}_N\right). \tag{4.4}$$

A summary of notations can be found in Table 4.1

4.2.1 Objective Function

Let $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_S)^T$ and $\boldsymbol{\Gamma} = (\boldsymbol{\Gamma}_1, \boldsymbol{\Gamma}_2, \ldots, \boldsymbol{\Gamma}_S)$ represent the vector of scale factors and the matrix of the outlier indicators, respectively. The probability function is given by

$$p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\beta}, \boldsymbol{\Gamma}) = \prod_{j=1}^{S} p(\mathbf{X}_j|\boldsymbol{\mu}, \beta_j, \boldsymbol{\Gamma}_j) \tag{4.5}$$

Finding the MLE of $\boldsymbol{\mu}$, $\boldsymbol{\beta}$, $\boldsymbol{\Gamma}$ in Equation (4.5) is equivalent to minimizing the following objective function:

$$\min_{\boldsymbol{\mu}, \boldsymbol{\beta}, \boldsymbol{\Gamma}} \quad \frac{1}{2}\sum_{j=1}^{S} \|\beta_j\mathbf{X}_j - \boldsymbol{\mu} - \boldsymbol{\Gamma}_j\|^2.$$

To achieve sparsity on outliers, I further added a ℓ_1 penalty on each $\mathbf{\Gamma}_j$ $(1 \leq j \leq S)$, and thus yielded the penalized objective funtion as follows:

$$\min_{\boldsymbol{\mu},\boldsymbol{\beta},\mathbf{\Gamma}} \quad \frac{1}{2}\sum_{j=1}^{S} \|\beta_j \mathbf{X}_j - \boldsymbol{\mu} - \mathbf{\Gamma}_j\|^2 + \lambda \sum_{j=1}^{S} \|\mathbf{\Gamma}_j\|_1 \tag{4.6}$$

where λ is the regulation penalty of $\mathbf{\Gamma}$, $\|\cdot\|$ is the ℓ_2-norm, and $\|\cdot\|_1$ is the ℓ_1-norm.

In order to circumvent the trivial solution in Equation (4.6), in which $\boldsymbol{\beta}$ is set to $\mathbf{0}$, I put a restriction on $\boldsymbol{\beta}$:

$$\boldsymbol{\beta}^T \mathbf{1}_S = S, \tag{4.7}$$

where $\mathbf{1}_S$ is a column vector of length S with all entries 1. This restriction leads to a constrained optimization problem.

With Lagrange multipliers and partial differentiating, it can be minimized by the following solution:

$$\hat{\boldsymbol{\mu}} = \frac{1}{S}\sum_{j=1}^{S}(\beta_j \mathbf{X}_j - \mathbf{\Gamma}_j), \tag{4.8}$$

$$\hat{\beta}_j = \left(a_j + \frac{S - \sum_{k=1}^{S} a_k b_k}{\sum_{k=1}^{S} b_k}\right) b_j \qquad 1 \leq j \leq S \tag{4.9}$$

$$\hat{\boldsymbol{\gamma}}_{ij} = \text{sign}(\beta_j \mathbf{x}_{ij} - \mu_i)(|\beta_j \mathbf{x}_{ij} - \mu_i| - \lambda)_+, \qquad 1 \leq i \leq N,\ 1 \leq j \leq S \tag{4.10}$$

where $a_j = \mathbf{X}_j^T(\mu + \boldsymbol{\gamma}_j)$, $b_j = \frac{1}{\mathbf{X}_j^T \mathbf{X}_j}$, sign$(\cdot)$ is the sign function, and $(\cdot)_+ = \max(\cdot, 0)$, which returns the positive part of the input.

Notice that the solution of $\boldsymbol{\mu}$ depends on $\boldsymbol{\beta}$ and $\mathbf{\Gamma}$, and vice versa. To achieve the optimization, the three variables should be updated iteratively in rounds. In each round the value of each variable is calculated while the other two variables are fixed with values from the last round. This whole process will repeat until the convergence of the objective function.

4.2.2 Polishing

The model training in Equation (4.6) and (4.7) was based on all data points. Given the existence of outliers, the common trend estimation was inevitably biased. In order to get a more accurate estimation, I considered

the previous training as a selection process for normal data points and further polished the result by re-training the model only with "clean" data.

The normal data point indicator h_{ij} is defined as

$$h_{ij} = \begin{cases} 1 & \text{if} \quad \boldsymbol{\gamma}_{ij} = 0 \\ 0 & \text{if} \quad \boldsymbol{\gamma}_{ij} > 0 \end{cases}.$$

A modified objective function (with no $\boldsymbol{\Gamma}$ related terms) was then used to refine the common trend estimation:

$$\begin{aligned} \min_{\boldsymbol{\mu},\boldsymbol{\beta}} \quad & \frac{1}{2}\sum_{j=1}^{S} \|\text{diag}(\mathbf{h}_j) \cdot (\beta_j \mathbf{X}_j - \boldsymbol{\mu})\|^2 \\ \text{subject to} \quad & \boldsymbol{\beta}^T \mathbf{1}_S = S, \end{aligned}$$

where $\text{diag}(\mathbf{h}_j)$ is a N-by-N diagonal matrix with elements $\mathbf{h_j} = (h_{1j}, h_{2j}, \ldots, h_{Nj})^T$ on the diagonal. Due to the space limit, the solution is omitted here, as it is similar to Equation (4.8) and (4.9).

4.2.3 Tuning λ

To determine the best regularization parameter λ for the model, I used Bayesian Information Criteria (BIC). In general, a search space for λ, i.e. [0, λ_{max}], was specified, and M points were picked uniformly in this range to form a set of candidates.

For every candidate λ, I computed BIC and chose the λ that minimizes the score. The BIC score I used is defined as:

$$BIC = -2l(\hat{\theta}) + klog(S). \tag{4.11}$$

In Equation (4.11), $l(\cdot)$ is the likelihood function, $\hat{\theta}$ is the parameter estimate, and $k = S + N + card(\{h_{ij} = 1 | 1 \leq i \leq N, 1 \leq j \leq S\})$, where $card(\cdot)$ is the cardinality of a set.

4.2.4 Ranking Outliers

Deciding whether a point is an outlier is always subjective: different people may want to find different types of outliers. For sequencing analysis, researchers are more interested in the relative scale factor between the observed and estimated read counts than the absolute magnitude of increase or decrease. To better capture this type of outliers, I defined the ranking score for each outlier as follows:

$$RRscore(\mathbf{y}_{ij}, \mu_i) = \frac{(\mathbf{y}_{ij} - \mu_i)^2}{(\mathbf{y}_{ij})(\mu_i)}, \tag{4.12}$$

where $\mathbf{y}_{ij}$ is the scaled read count of bin i for sample j, and μ_i is the expected value of $\mathbf{y}_{ij}$. For brievity, $RRscore(\mathbf{y}_{ij}, \mu_i)$ is also written as $RRscore_{ij}$.

Property 1. Let $\phi > 1$ be the scale factor between a and b ($a > 0, b > 0$), then $RRscore(a, b) \in [0, +\infty)$ and is asymptotically linear to ϕ, i.e. $RRscore(a, b) \in \Theta(\phi)$.

Proof. Given $a > 0$ and $b > 0$, it is easy to see $RRscore(a, b) > 0$. Without loss of generality, assume $a = \phi b$. Then $RRscore(a, b) = \frac{(\phi b - b)^2}{\phi b^2} = \frac{(\phi - 1)^2}{\phi} = \phi + \frac{1}{\phi} - 2 \in \Theta(\phi)$. Similarly, it can be proved to be true when $b = \phi a$. Notice that $RRscore(a, b) = 0$ if and only if $\phi = 1$, i.e. $a = b$. □

One can categorize outliers into two classes based on whether it is greater or smaller than the estimated read count. The former type is called the **gain outlier**, while the latter the **loss outlier**.

Figure 4.2 shows these two types of outliers. The estimated read counts at the outlier Bin 2 are the same (i.e. 150) in Figure 4.2a and Figure 4.2b. The read count for the gain outlier is 300, while the read count for the loss outlier is 75. Although the two outliers have different distances from the estimate, they have the same $RRscore$. This unsymmetrical property in $RRscore$ is shown and proved as follows.

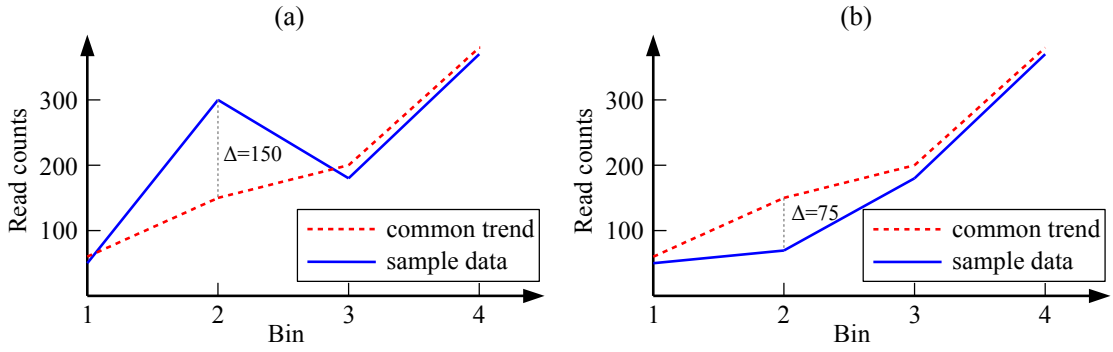


Figure 4.2: Examples of outliers in local read depth. In both figures, Bin 2 is considered as a significant outlier. The left figure shows a gain outlier, while the right figure suggests a loss outlier. Notice that the RRscores of these two outliers are the same, i.e. 0.5 .

Property 2. If $RRscore(a, b) = RRscore(a, c)$ and $b > a > c$, then $b - a > a - c$.

Proof. Let $\phi_1 = b/a > 1$ and $\phi_2 = a/c > 1$. We have $b - a = (\phi_1 - 1)a$ and $a - c = (\frac{\phi_2 - 1}{\phi_2})a$. From $RRscore(a, b) = RRscore(a, c)$, it can be proved that $\phi_1 = \phi_2$. Therefore, $b - a = (\phi_1 - 1)a > (\frac{\phi_1 - 1}{\phi_1})a = (\frac{\phi_2 - 1}{\phi_2})a = a - c$. □

4.3 Results

In this section, I evaluate the performance of two tasks, common trend estimation and outlier detection, with simulation data and a real world mouse DNAseq data set.

In the first task, I compared my method with two other approaches: one based on mean and the other based on median. In the MEAN approach, which is commonly used for read count normalization, the scale factor β is considered to be **inversely** proportional to the mean read counts of a sample. Then once can calculate the common trend $\boldsymbol{\mu}$ by taking the mean of scaled read counts across samples for every bin. In the MEDIAN approach, median is used to replace the role of mean to achieve more robust estimation. Notice that in order to get a unique solution with these two approaches, I imposed the same restriction on $\boldsymbol{\beta}$ as in Equation (4.7). The corresponding formulas for $\boldsymbol{\beta}$ and $\boldsymbol{\mu}$ are listed as follows.

$$\hat{\beta}_j^{\text{MEAN}} = \frac{S}{\text{mean}_i(\mathbf{x}_{ij}) \sum_{k=1}^{S} \frac{1}{\text{mean}_i(\mathbf{x}_{ik})}} \tag{4.13}$$

$$\hat{\mu}_i^{\text{MEAN}} = \text{mean}_j \left(\hat{\beta}_j^{\text{MEAN}} x_{ij} \right) \tag{4.14}$$

$$\hat{\beta}_j^{\text{MEDIAN}} = \frac{S}{\text{median}_i(\mathbf{x}_{ij}) \sum_{k=1}^{S} \frac{1}{\text{median}_i(\mathbf{x}_{ik})}} \tag{4.15}$$

$$\hat{\mu}_i^{\text{MEDIAN}} = \text{median}_j \left(\hat{\beta}_j^{\text{MEDIAN}} \mathbf{x}_{ij} \right) \tag{4.16}$$

In the second task, I used the previous result and contrasted my outlier ranking score with the other two scores based on standard deviation(STD) and median absolute deviation(MAD), respectively. These scores are widely used in outlier detection and are defined as

$$STDscore_{ij} = \frac{|\hat{\beta}_j^{\text{MEAN}} \mathbf{x}_{ij} - \hat{\mu}_i^{\text{MEAN}}|}{\text{std}_j \left(\hat{\beta}_j^{\text{MEAN}} \mathbf{x}_{ij} \right)} \tag{4.17}$$

$$MADscore_{ij} = \frac{|\hat{\beta}_j^{\text{MEDIAN}} \mathbf{x}_{ij} - \hat{\mu}_i^{\text{MEDIAN}}|}{\text{mad}_j \left(\hat{\beta}_j^{\text{MEDIAN}} \mathbf{x}_{ij} \right)} \tag{4.18}$$

where $\text{mad}_i(a_i) = \text{median}_i \left(|a_i - \text{median}_j(a_j)| \right)$.

4.3.1 Simulation Experiments

The synthetic data were generated as follows. First, a common trend $\boldsymbol{\mu}$ of length N ($N = 800$) was constructed with random numbers drawn from a negative binomial distribution $\mathcal{NB}(r, p)$ ($r = 30, p = 0.1$). The mean of the common trend is $\frac{(1-p)r}{p} = 270$.

For each sample, $\rho\%$ of bins were randomly chosen to be outliers, of which $\omega\%$ were gain outliers and $(100 - \omega)\%$ were loss outliers. I generated the scaled read counts $\mathbf{y}_{ij}$ ($1 \leq i \leq N,\ 1 \leq j \leq S$) by adding random noise and outliers into the common trend:

$$\mathbf{y}_{ij} = \begin{cases} \phi\mu_i + \epsilon_{ij} & \text{gain outliers} \\ \frac{1}{\phi}\mu_i + \epsilon_{ij} & \text{loss outliers} \\ \mu_i + \epsilon_{ij} & \text{normal bins} \end{cases}, \tag{4.19}$$

where ϕ is a random number drawn from a uniform distribution $\mathcal{U}(\phi_{min}, \phi_{max})$ ($1 < \phi_{min} < \phi_{max}$), and $\epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$.

Finally, the observed read count $\mathbf{x}_{ij}$ was given by $\mathbf{y}_{ij}/\beta_j$, where β_j ($j = 1, 2, \ldots, S$) were randomly chosen and roughly on the same order of magnitude. Notice that Equation (4.7) was also imposed on β_j so that the underlying common trend was unique for each simulation.

To show the effect of different parameters on the performance, I chose σ, ρ, ω, ϕ_{min}, ϕ_{max} in the following sets: $\sigma \in \{30, 40, 50\}$, $\rho \in \{0.5, 1, 2\}$, $\omega \in \{0, 25, 50, 75, 100\}$, and $(\phi_{min}, \phi_{max}) \in \{(1.5, 3.0), (3.0, 6.0)\}$. However, by default, the parameter settings were set as $\sigma = 40$, $\rho = 1$, $\omega = 50$, $(\phi_{min}, \phi_{max}) = (1.5, 3.0)$. I repeated the experiment 30 times for each parameter setting, with each time a newly simulated data set.

4.3.1.1 Common Trend Estimation

I compared the accuracy of common trend estimation among MEAN, MEDIAN and my method using two measures. The first one is $\Delta\mu$, which shows the distance between the estimated trend $\hat{\boldsymbol{\mu}}$ and the true trend $\boldsymbol{\mu}$. It is defined as

$$\Delta\mu = \|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|. \tag{4.20}$$

The second meausre is the residual sum of square (RSS) on normal points and is defined as

$$RSS_{normal} = \sum_{i=1}^{N} \sum_{j=1}^{S} g_{ij} (\hat{\beta}_j \mathbf{x}_{ij} - \hat{\mu}_i)^2, \tag{4.21}$$

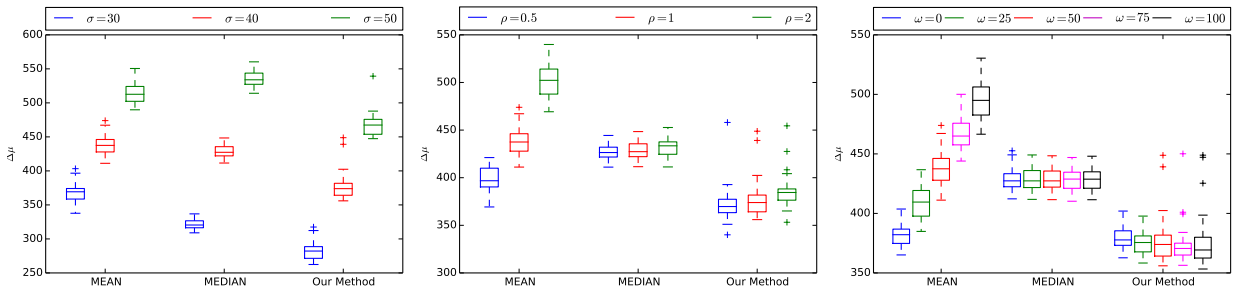
where $g_{ij} = 0$ if bin i of sample j is an outlier bin and otherwise it is 1. RSS measures how well a method can see through the outliers and learn the common trend on normal data.

Figure 4.3 shows the box plots of $\Delta\mu$ while $\phi_{min} = 1.5$, $\phi_{max} = 3.0$. Observe that for every parameter setting my method has the lowest mean of $\Delta\mu$, which suggests that it always outperforms the two alternative methods. Also as the standard deviation of noise σ increases, the increase of $\Delta\mu$ is similar for all methods, because in essence it is changing the signal-to-noise ratio that has an equal effect on every method. However, the change of ρ (the percentage of outlier bins) and ω (the percentage of gain outliers) has a larger effect on MEAN than on MEDIAN and my method. For one thing, the increase of ρ directly leads to more outliers in the data. For the other, the change of ω has some underlying effect on the magnitude of outliers. As it was shown that the distance between an outlier and the expected value is not symmetric among gain outliers and loss outliers, the increase of the gain outlier ratio actually enlarges the average outlier magnitude indirectly. Similar trends of σ, ρ, and ω can be seen in Figure 4.4 where I doubled the outlier magnitude by setting $\phi_{min} = 3.0$, $\phi_{max} = 6.0$. Therefore, whereas MEAN is vulnerable to outliers, my method is as robust as MEDIAN, and the estimate from my method is closer to the true common trend than other methods.

Table 4.2 and 4.3 summarize the result for RSS_{normal}. Similar to $\Delta\mu$, it can be easily seen that RSS_{normal} of MEAN is sensitive to the change of ρ and ω, while the measure from MEDIAN and my method is rather stable. Furthermore, the RSS_{normal} of my method is always 2%-7% smaller than MEAN, and 3%-5% smaller than MEDIAN. As RSS_{normal} does not include any outlier residuals, it actually suggests that the common trend estimated by my method fits the normal data with least error.

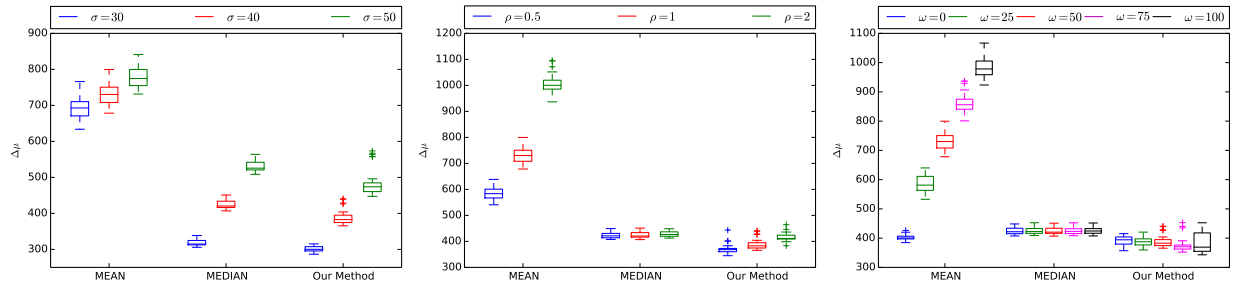
4.3.1.2 Outlier Detection

I also compared $RRscore$ with $STDscore$ and $MADscore$ in terms of outlier detection performance. These scores are considered as quantitative measures of outlierness for each bin, and I further assumed that scores across different bins are comparable. I calculated three scores for each bin and sample, and hen pooled them together regardless of their origin bin or sample. Let $\mathcal{Z}$ represent the set of scores. Threshold-based binary classifiers were trained on $\mathcal{Z}$ to separate normal points from outliers. The classifier $f : \mathcal{Z} \rightarrow \{0, 1\}$ is defined



(a) Box plot for different σ (b) Box plot for different ρ (c) Box plot for different ω

Figure 4.3: The box plots of $\Delta\mu$ for different σ, ρ, and ω when $\phi_{min} = 1.5$, $\phi_{max} = 3.0$. In each figure, one parameter is varied while the others are set to the default values. The red boxes represent the default parameter settings, i.e. $\sigma = 40$, $\rho = 1$, and $\omega = 50$.



(a) Box plot for different σ (b) Box plot for different ρ (c) Box plot for different ω

Figure 4.4: The box plots of $\Delta\mu$ for different σ, ρ, and ω when $\phi_{min} = 3.0$, $\phi_{max} = 6.0$.

σ	ρ	ω	MEAN	MEDIAN	My Method
30	1	50	6.971e+06 $\pm$ 2.73e+05	6.714e+06 $\pm$ 2.04e+05	6.463e+06 $\pm$ 2.36e+05
40	1	50	1.196e+07 $\pm$ 4.07e+05	1.193e+07 $\pm$ 3.54e+05	1.148e+07 $\pm$ 4.86e+05
50	1	50	1.838e+07 $\pm$ 5.88e+05	1.863e+07 $\pm$ 5.70e+05	1.786e+07 $\pm$ 5.74e+05
40	0.5	50	1.174e+07 $\pm$ 3.61e+05	1.200e+07 $\pm$ 3.61e+05	1.152e+07 $\pm$ 4.46e+05
40	1	50	1.196e+07 $\pm$ 4.07e+05	1.193e+07 $\pm$ 3.54e+05	1.148e+07 $\pm$ 4.86e+05
40	2	50	1.238e+07 $\pm$ 4.04e+05	1.182e+07 $\pm$ 3.45e+05	1.140e+07 $\pm$ 3.30e+05
40	1	0	1.156e+07 $\pm$ 3.21e+05	1.194e+07 $\pm$ 3.59e+05	1.148e+07 $\pm$ 3.41e+05
40	1	25	1.175e+07 $\pm$ 3.57e+05	1.193e+07 $\pm$ 3.55e+05	1.145e+07 $\pm$ 3.38e+05
40	1	50	1.196e+07 $\pm$ 4.07e+05	1.193e+07 $\pm$ 3.54e+05	1.148e+07 $\pm$ 4.86e+05
40	1	75	1.220e+07 $\pm$ 4.39e+05	1.193e+07 $\pm$ 3.51e+05	1.146e+07 $\pm$ 4.27e+05
40	1	100	1.244e+07 $\pm$ 4.52e+05	1.193e+07 $\pm$ 3.51e+05	1.149e+07 $\pm$ 4.79e+05

Table 4.2: The table for RSS_{normal} (with 95% confidence interval) when σ, ρ, and ω are varied, respectively. ($\phi_{min} = 1.5$, $\phi_{max} = 3.0$)

σ	ρ	ω	MEAN	MEDIAN	My Method
30	1	50	1.013e+07 $\pm$ 8.16e+05	6.733e+06 $\pm$ 2.65e+05	6.589e+06 $\pm$ 2.49e+05
40	1	50	1.513e+07 $\pm$ 9.00e+05	1.196e+07 $\pm$ 4.62e+05	1.162e+07 $\pm$ 5.14e+05
50	1	50	2.156e+07 $\pm$ 1.05e+06	1.868e+07 $\pm$ 7.20e+05	1.813e+07 $\pm$ 9.39e+05
40	0.5	50	1.339e+07 $\pm$ 6.47e+05	1.200e+07 $\pm$ 4.36e+05	1.153e+07 $\pm$ 4.64e+05
40	1	50	1.513e+07 $\pm$ 9.00e+05	1.196e+07 $\pm$ 4.62e+05	1.162e+07 $\pm$ 5.14e+05
40	2	50	1.906e+07 $\pm$ 1.07e+06	1.185e+07 $\pm$ 4.25e+05	1.164e+07 $\pm$ 4.74e+05
40	1	0	1.177e+07 $\pm$ 4.45e+05	1.197e+07 $\pm$ 4.59e+05	1.161e+07 $\pm$ 4.93e+05
40	1	25	1.336e+07 $\pm$ 6.57e+05	1.196e+07 $\pm$ 4.59e+05	1.160e+07 $\pm$ 4.95e+05
40	1	50	1.513e+07 $\pm$ 9.00e+05	1.196e+07 $\pm$ 4.62e+05	1.162e+07 $\pm$ 5.14e+05
40	1	75	1.694e+07 $\pm$ 9.95e+05	1.196e+07 $\pm$ 4.65e+05	1.154e+07 $\pm$ 5.51e+05
40	1	100	1.888e+07 $\pm$ 1.19e+06	1.196e+07 $\pm$ 4.65e+05	1.160e+07 $\pm$ 6.31e+05

Table 4.3: The table for RSS_{normal} (with 95% confidence interval) when σ, ρ, and ω are varied, respectively. ($\phi_{min} = 3.0$, $\phi_{max} = 6.0$)

as

$$f(z) = \begin{cases} 0 & z < t \\ 1 & z >= t \end{cases}, \tag{4.22}$$

where $z \in \mathcal{Z}$, t is a predefined thresdhold, 0 represents the normal class, and 1 represents the outlier class. Changing t can result in different classifiers, and thus different true positive rates and false positive rates.

I computed the average ROC curves and their 95% confidence intervals over 30 simulated data sets using the vertical averaging method (Fawcett, 2006). Figure 4.5 and Figure 4.6 summarize the ROC curves under different parameter settings. Observe that my method outperforms MEAN+STD and MEDIAN+MAD in all ROC plots. In detail, σ has a similar effect on outlier detection and common trend estimation (as shown in Section 4.3.1.1). After all, random noise may not only disturb the common trend, but also make outliers indiscernible from normal data points. The change of ρ affects MEAN+STD, but hardly MEDIAN+MAD and my method. ϕ_{min}, ϕ_{max}, and ω, all of which are related to the magnitude of outliers, have the main impact on the accuracy. Comparing Figure 4.5 with Figure 4.6, one can see that as ϕ_{min}, ϕ_{max} increase, all methods become more accurate. Despite the fact that the performance of MEAN+STD and MEDIAN+MAD deteriorates as ω decreases (i.e. the number of loss outliers increases), my method remains consistent, which suggests that it has higher accuracy in finding loss outliers than other methods.

4.3.2 Experiment on DNAseq Data

I used DNAseq data from 10 samples of mouse strain FVB/NJ $\times$ (WSB/EiJ $\times$ PWK/PhJ). These samples were deeply sequenced with Illumina Next Generation Sequencing (NGS) platforms with different coverages. In total, there are 1.8 billion pair-end reads in the data, and the length of each end is 100bp.

In each sample, one copy of chromosomes was entirely from FVB, while the other was a mosaic derived from WSB and PWK. The haplotype structure is different in different samples: in a given region, some samples may only have FVB and WSB sequences, while the others have FVB and PWK sequences (as shown in Figure 4.7). Therefore, instead of conducting a genome-wide experiment on all samples, I chose a local region and a subset of samples that have the same haplotype structure in that region for evaluation.

To achieve this, the DNAseq reads were aligned to three founder genomes with the multi-alignment pipeline in Section 3.3, so that reference bias in read alignment was reduced (Huang et al., 2013) (Holt et al., 2013). Second, with the help of MegaMuga genotyping arrays, the recombination breakpoints for each sample could be determined (Fu et al., 2012). The whole genome was then divided into regions according to the union of all breakpoints, and thus in each region the samples would fall into two classes, i.e. FVB/PWK and FVB/WSB, based on what sequences they have. Third, every region was further broken up into bins of size 25kbp and the number of reads per bin and sample was counted. The final read counts were used for the analysis of common trend and outliers.

Figure 4.8 shows the estimated common trend in a 13.5Mbp region on Chromosome 2 (from 73.8Mbp to 87.3Mbp on NCBI Build 37 coordinate) with samples in class FVB/PWK. The Euclidean distances between two of the three $\hat{\boldsymbol{\mu}}$ are shown as follows:

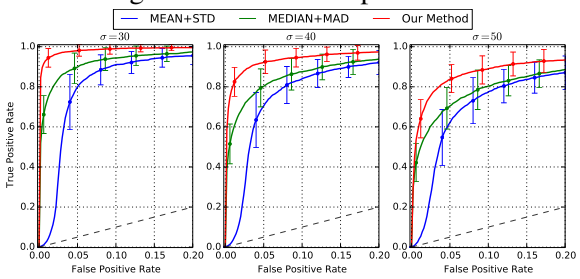
$$\begin{aligned}
\|\hat{\boldsymbol{\mu}}^{\text{MEAN}} - \hat{\boldsymbol{\mu}}^{\text{MEDIAN}}\| &= 2433.39, \\
\|\hat{\boldsymbol{\mu}}^{\text{MEAN}} - \hat{\boldsymbol{\mu}}^{\text{My Method}}\| &= 2449.84, \\
\|\hat{\boldsymbol{\mu}}^{\text{MEDIAN}} - \hat{\boldsymbol{\mu}}^{\text{My Method}}\| &= 214.00.
\end{aligned}$$

The most significant difference between MEAN and the other two methods occurs at around 77.8Mbp, where a high peak is seen in the MEAN estimate. This is due to the outlier bins in that region, which makes the MEAN method locally biased. Also, MEDIAN approach and my approach look alike, suggesting they both have robust estimation.

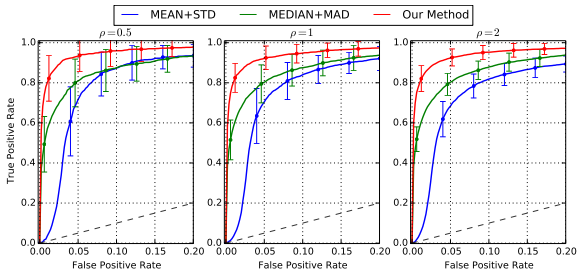
The Venn diagram of the numer of outliers claimed by different methods is shown in Figure 4.9. Here I set the threshold t in Equation (4.22) to be the $20th$ (Figure 4.9a) and $40th$ (Figure 4.9b) largest scores in the score set. In other words, I considered the data points with top 20 and top 40 scores as outliers. It can be seen from the figure that the outliers discovered by MEAN+STD have no intersection with the other two methods. This is due to both inaccurate estimate of the common trend and the standard deviation being compromised by outliers.

Although MEDIAN+MAD and my method have some shared outliers, their difference in outlier ranking is observable. Under the top 20 setting, a close-up of the region between 77Mbp and 78Mbp is shown Figure 4.10a. Each color line represent the scaled read counts for a sample, and the green circles indicate outliers. While most of the outliers between 77.7Mbp and 77.9Mbp are found by MEDIAN+MAD and my method, there are two outlier points around 77.8Mbp that were missed by MEDIAN+MAD. Also, MEDIAN+MAD claimed another outlier near 77.4Mbp that does not seem to be significant at all. Under the top 40 setting, let's zoom in the region between 80Mbp and 81Mbp (Figure 4.10b). Both methods could find the outlier near 80.3Mbp, but two obvious outliers around 80.8Mbp were missed by MEDIAN+MAD. Furthermore, the other outliers labelled by MEDIAN+MAD are quite ambigous, such as the ones around 80.6Mbp or 81.0Mbp. Therefore, the above results suggest that the outliers claimed my method are more significant and accurate than the other two approaches in a sequence analysis sense.

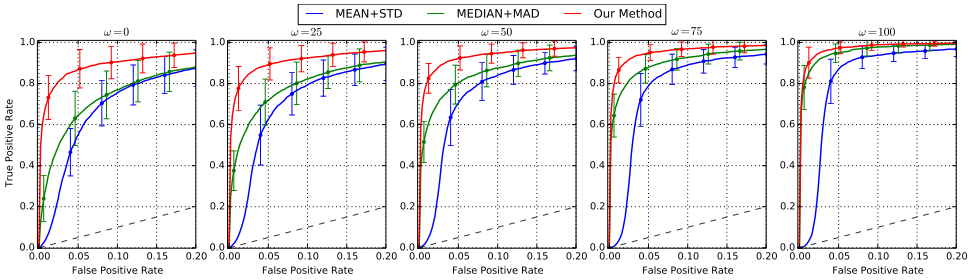
Figure 4.5: The ROC plots for different σ, ρ, and ω when $\phi_{min} = 1.5$, $\phi_{max} = 3.0$.



(a) ROC plot for different σ

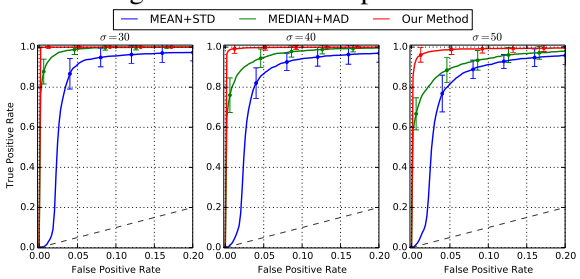


(b) ROC plot for different ρ

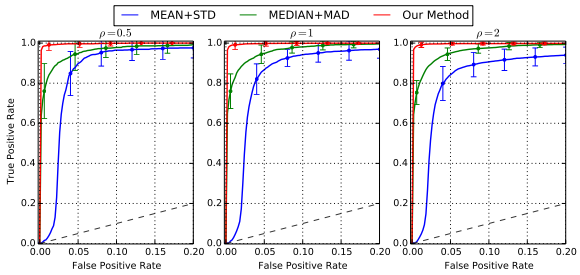


(c) ROC plot for different ω

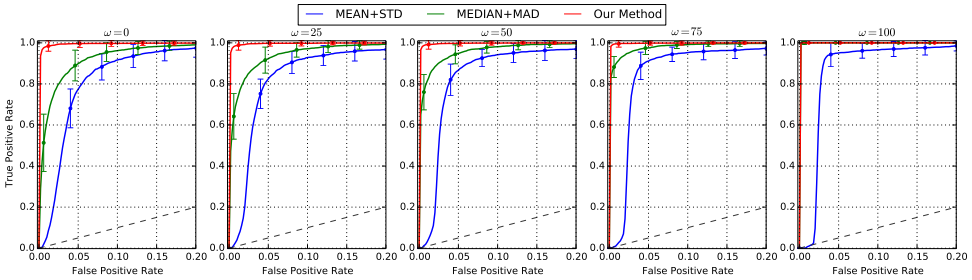
Figure 4.6: The ROC plots for different σ, ρ, and ω when $\phi_{min} = 3.0$, $\phi_{max} = 6.0$.



(a) ROC plot for different σ



(b) ROC plot for different ρ



(c) ROC plot for different ω

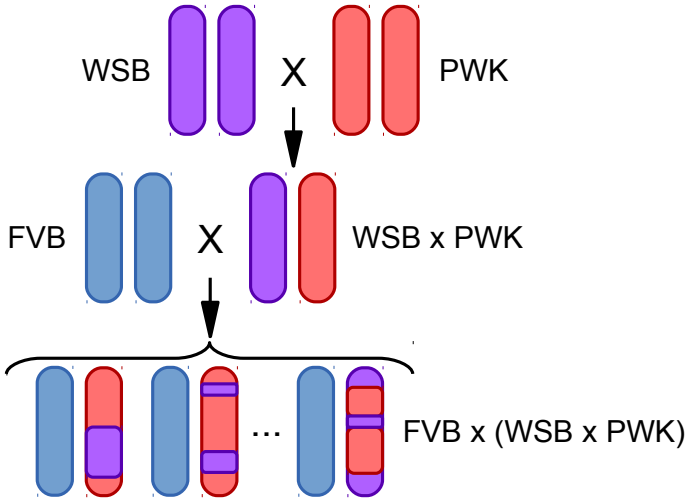


Figure 4.7: An illuatration of the FVBx(WSBxPWK) strain

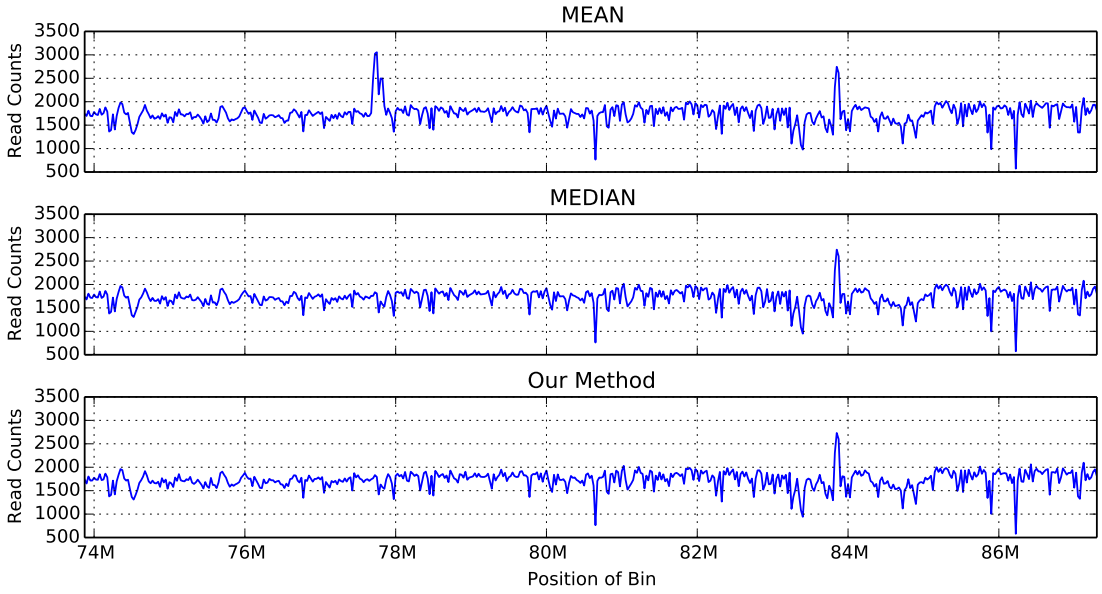


Figure 4.8: Estimated common trends for FVB/PWK samples in chr2:73.8Mbp-87.3Mbp. While MEDIAN and my method have nearly identical estimates, MEAN method shows a peak around 77.8Mbp, which is due to some outliers.

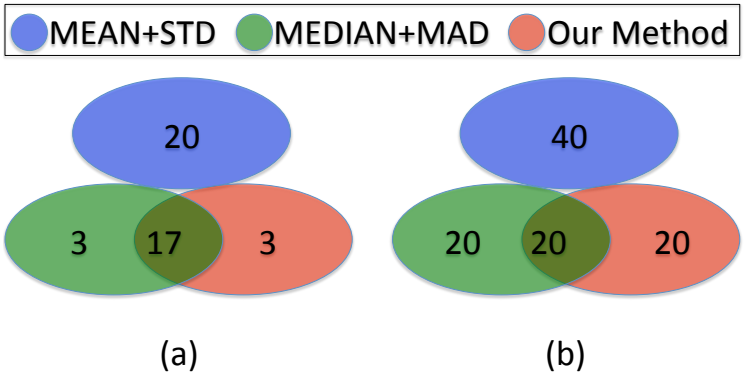
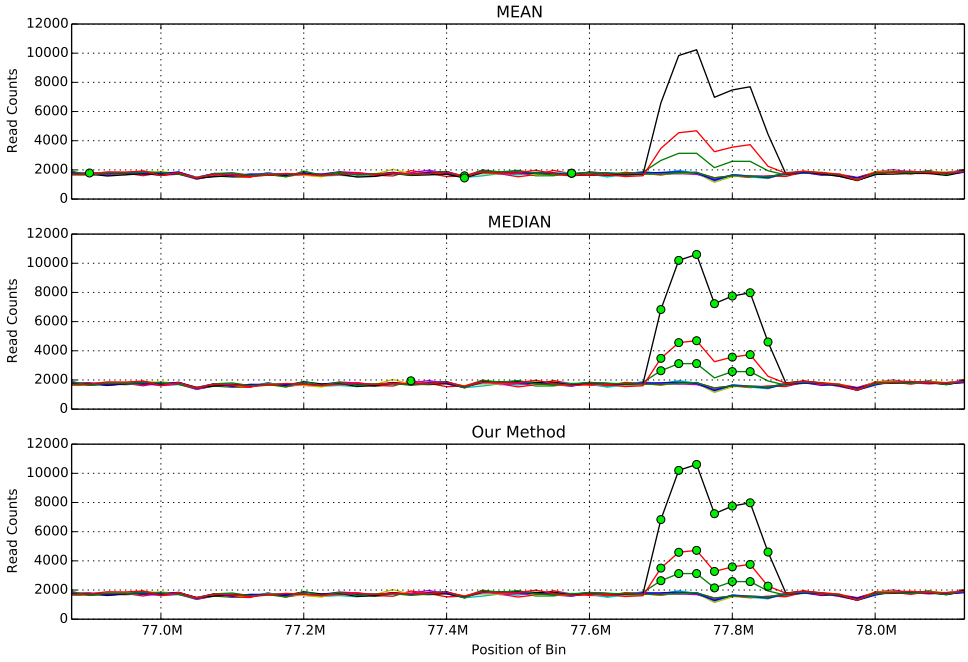
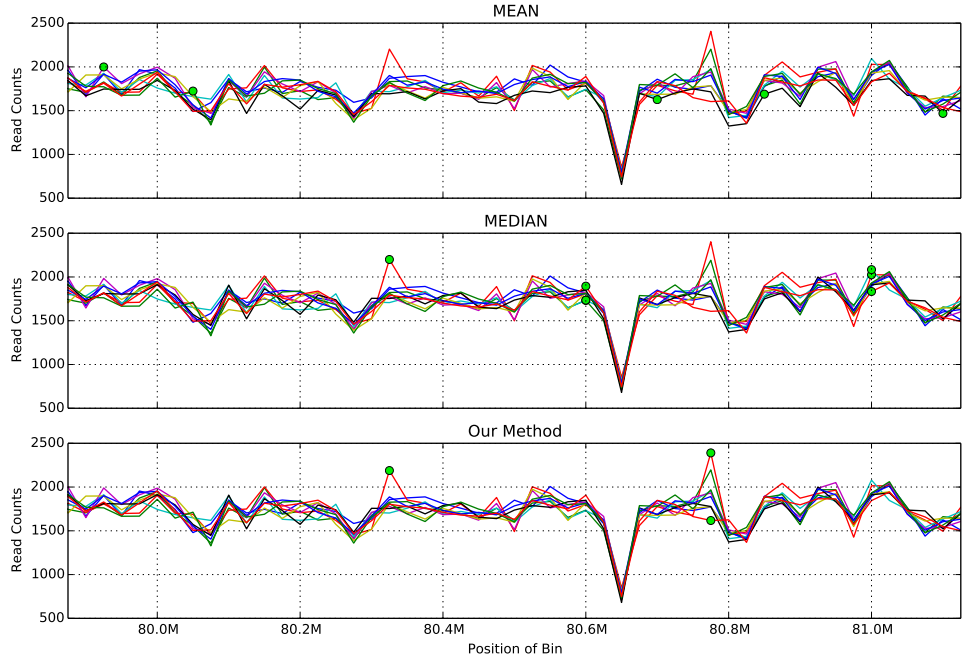


Figure 4.9: Venn diagram showing the number of outliers found by each method and their relationship to other methods: (a)Top 20 outliers, (b)Top 40 outliers.



(a) Scaled read counts and outliers (top 20)



(b) Scaled read counts and outliers (top 40)

Figure 4.10: Scaled read counts and outliers. Read counts from different bins are visualized with line plots, while outliers are annotated with green circles.

CHAPTER 5: SUMMARY AND FUTURE DIRECTIONS

5.1 Summary

While it is common practice among the research community to align reads against the standard reference sequence, the difference between target organisms and the reference may lead to reference bias in read alignment. Such bias will confound downstream analysis where read depth is used as a quantitative measure.

In this dissertation, I have showed that better reference sequences (i.e. pseudogenomes) can be constructed by incorporating genomic variations from founders into the standard reference sequence. New alignment pipelines based on these pseudogenomes were proposed and experiments suggested that they can largely reduce reference bias in the output alignment. Given that some biases are remained, I proposed a probablistic model to analyze and locate them. All of these methods are applicable to a wide range of organisms and various sequencing methods.

5.2 Future Directions

There are several promising directions for future work. First, the MOD format I proposed can handle structure variation in theory, but how to represent them efficiently in practise remains a challenging problem. In addition, whereas the MOD files used in my experiments were basically from VCF files, it will be very useful to derive MOD files by comparing two genome sequences. This topic is related to the whole genome alignment (WGA) problem, and it is interesting for futher exploration.

Second, while I chose to use a diallel experiment to evaluate my new pipelines, it will be interesting to apply them to other multi-parental crosses. For example, my multi-alignment pipeline can be directly applied to Recombinant Inbred Lines (RILs) (Silver et al., 1995) and back-crosses. For a multi-parental cross with N distinct inbred founders, one would generate N pseudogenomes and perform N separate alignments. These alignments can then be merged using N BAM files. In this scenario, each mapping that is saved to the

output will have an N-bit flag set indicating which files the read was found in. This allows for cases where a mapping's origin is shared/ambiguous between multiple founders.

Third, additional filters can be incorporated into the multi-alignment pipeline to better determine the origin of mappings. In my experiment, I only used the Unique and Quality filters as informative filters. This resulted in approximately 5% of the mapped reads being handled by the Random filter. Adding an additional filters before the Random filter will help to reduce the amount of random choices made in the final output. One possible filter is a Pileup filter based on choosing among otherwise equal mappings the single mapping that has the most surrounding mappings supporting it. To do this, One can first find all mapping sets that can be filtered by the Unique or Quality filters and use their chosen mappings to compute the read coverage at each base in the reference genome. Then, any mapping sets that couldn't be resolved using Unique or Quality would compare the pileup coverage of each potential mapping in the set and choose the mapping with the highest coverage. This will be particularly useful for reducing the number of reads that map to pseudogenes in RNA-seq. In cases where the pileups are not significantly different, more computation or simply using the Random filter may be necessary.

Last but not least, it is promising to further investigate and correct the remaining bias. After discovering the possible bias locations, we can classify them into different categories in terms of the causes. For example, we may be able to find out where are the wrongly annotated variants in a pseudogenome. Then, the next steps will be to correct them and rerun the alignment. It would also be interesting to see how we can repeat this aligning-correcting procedure to improve accuracy of pseudogenome and alignment.

BIBLIOGRAPHY

Chadwick, L. H., Pertz, L. M., Broman, K. W., Bartolomei, M. S., and Willard, H. F. (2006). Genetic control of x chromosome inactivation in mice: definition of the xce candidate interval. *Genetics*, 173(4):2103–2110.

Committe on Standardized Genetic Nomeclature for Mice (1972). Standard karyotype of the mouse, mus musculus. *Journal of Heredity*, 63(2):69–72.

Cumbie, J. S., Kimbrel, J. A., Di, Y., Schafer, D. W., Wilhelm, L. J., Fox, S. E., Sullivan, C. M., Curzon, A. D., Carrington, J. C., Mockler, T. C., et al. (2011). Gene-counter: a computational pipeline for the analysis of rna-seq data for gene expression differences. *PLoS One*, 6(10):e25279.

Danecek, P., Auton, A., Abecasis, G., Albers, C., Banks, E., DePristo, M., Handsaker, R., Lunter, G., Marth, G., Sherry, S., et al. (2011). The variant call format and vcftools. *Bioinformatics*, 27(15):2156–2158.

Degner, J., Marioni, J., Pai, A., Pickrell, J., Nkadori, E., Gilad, Y., and Pritchard, J. (2009). Effect of read-mapping biases on detecting allele-specific expression from rna-sequencing data. *Bioinformatics*, 25(24):3207–3212.

DeVeale, B., van der Kooy, D., and Babak, T. (2012). Critical evaluation of imprinted gene expression by rna–seq: A new perspective. *PLoS Genetics*, 8(3):e1002600.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874.

Flicek, P., Amode, M. R., Barrell, D., Beal, K., Brent, S., Carvalho-Silva, D., Clapham, P., Coates, G., Fairley, S., Fitzgerald, S., et al. (2012). Ensembl 2012. *Nucleic Acids Research*, 40(D1):D84–D90.

Fu, C.-P., Welsh, C. E., de Villena, F. P.-M., and McMillan, L. (2012). Inferring ancestry in admixed populations using microarray probe intensities. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, BCB '12, pages 105–112, New York, NY, USA. ACM.

Gan, X., Stegle, O., Behr, J., Steffen, J. G., Drewe, P., Hildebrand, K. L., Lyngsoe, R., Schultheiss, S. J., Osborne, E. J., Sreedharan, V. T., et al. (2011). Multiple reference genomes and transcriptomes for arabidopsis thaliana. *Nature*, 477(7365):419–423.

Gregg, C., Zhang, J., Weissbourd, B., Luo, S., Schroth, G. P., Haig, D., and Dulac, C. (2010). High-resolution analysis of parent-of-origin allelic expression in the mouse brain. *Science*, 329(5992):643–648.

Gusfield, D. (1997). *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press.

Holt, J., Huang, S., McMillan, L., and Wang, W. (2013). Read annotation pipeline for high-throughput sequencing data. In *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, page 605. ACM.

Huang, S., Kao, C.-Y., McMillan, L., and Wang, W. (2013). Transforming genomes using mod files with applications. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*. ACM.

Keane, T., Goodstadt, L., Danecek, P., White, M., Wong, K., Yalcin, B., Heger, A., Agam, A., Slater, G., Goodson, M., et al. (2011). Mouse genomic variation and its effect on phenotypes and gene regulation. *Nature*, 477(7364):289–294.

Langmead, B. (2013). Bowtie2 manual. `http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml`. Accessed: 2013-05-14.

Langmead, B. and Salzberg, S. L. (2012). Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359.

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

Li, H. (2011). Tabix: fast retrieval of sequence features from generic tab-delimited files. *Bioinformatics*, 27(5):718–719.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., et al. (2009). The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079.

Liu, E., Zhang, Q., McMillan, L., de Villena, F., and Wang, W. (2010). Efficient genome ancestry inference in complex pedigrees with inbreeding. *Bioinformatics*, 26(12):i199–i207.

Magi, A., Tattini, L., Pippucci, T., Torricelli, F., and Benelli, M. (2012). Read count approach for dna copy number variants detection. *Bioinformatics*, 28(4):470–478.

Marioni, J., Mason, C., Mane, S., Stephens, M., and Gilad, Y. (2008). Rna-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome research*, 18(9):1509–1517.

McDaniell, R., Lee, B.-K., Song, L., Liu, Z., Boyle, A. P., Erdos, M. R., Scott, L. J., Morken, M. A., Kucera, K. S., Battenhouse, A., et al. (2010). Heritable individual-specific and allele-specific chromatin signatures in humans. *Science*, 328(5975):235–239.

Missirian, V., Henry, I., Comai, L., and Filkov, V. (2012). Pope: pipeline of parentally-biased expression. In *Bioinformatics Research and Applications*, pages 177–188. Springer.

Peng, Z., Cheng, Y., Tan, B., Kang, L., Tian, Z., Zhu, Y., Zhang, W., Liang, Y., Hu, X., Tan, X., et al. (2012). Comprehensive analysis of rna-seq data reveals extensive rna editing in a human transcriptome. *Nature biotechnology*, 30(3):253–260.

Picardi, E., Horner, D., Chiara, M., Schiavon, R., Valle, G., and Pesole, G. (2010). Large-scale detection and analysis of rna editing in grape mtdna by rna deep-sequencing. *Nucleic acids research*, 38(14):4755–4767.

Richard, H., Schulz, M. H., Sultan, M., Nürnberger, A., Schrinner, S., Balzereit, D., Dagand, E., Rasche, A., Lehrach, H., Vingron, M., et al. (2010). Prediction of alternative isoforms from exon expression levels in rna-seq experiments. *Nucleic acids research*, 38(10):e112–e112.

Rivas-Astroza, M., Xie, D., Cao, X., and Zhong, S. (2011). Mapping personal functional data to personal genomes. *Bioinformatics*, 27(24):3427–3429.

Rozowsky, J., Abyzov, A., Wang, J., Alves, P., Raha, D., Harmanci, A., Leng, J., Bjornson, R., Kong, Y., Kitabayashi, N., et al. (2011). Alleleseq: analysis of allele-specific expression and binding in a network framework. *Molecular systems biology*, 7(1).

Satya, R., Zavaljevski, N., and Reifman, J. (2012). A new strategy to reduce allelic bias in rna-seq readmapping. *Nucleic Acids Research*, 40(16):e127–e127.

Silver, L. et al. (1995). *Mouse genetics: concepts and applications.* Oxford University Press.

The 1000 Genomes Project Consortium (2010). A map of human genome variation from population scale sequencing. *Nature*, 467(7319):1061–1073.

The ENCODE Project Consortium et al. (2011). A user's guide to the encyclopedia of dna elements (encode). *PLoS Biol*, 9(4):e1001046.

Trapnell, C., Pachter, L., and Salzberg, S. (2009). Tophat: discovering splice junctions with rna-seq. *Bioinformatics*, 25(9):1105–1111.

Trapnell, C., Williams, B., Pertea, G., Mortazavi, A., Kwan, G., Van Baren, M., Salzberg, S., Wold, B., and Pachter, L. (2010). Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature biotechnology*, 28(5):511–515.

Turro, E., Su, S.-Y., Gonçalves, Â., Coin, L., Richardson, S., Lewin, A., et al. (2011). Haplotype and isoform specific expression estimation using multi-mapping rna-seq reads. *Genome Biol*, 12(2):R13.

Yoon, S., Xuan, Z., Makarov, V., Ye, K., and Sebat, J. (2009). Sensitive and accurate detection of copy number variants using read depth of coverage. *Genome research*, 19(9):1586–1592.