An Analytical Method for Inferring Library Identity From Illumina NGS Read Groups

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Citable link</td>
<td><a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:33826059">http://nrs.harvard.edu/urn-3:HUL.InstRepos:33826059</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA">http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA</a></td>
</tr>
</tbody>
</table>
An Analytical Method for Inferring Library Identity from Illumina NGS Read Groups

Bradley R. Taylor

A Thesis in the Field of Biotechnology
for the degree of Master of Liberal Arts in Extension Studies

Harvard University
May 2017
Abstract

When working with genetic sequence data, it is important to know the individual, sample, and sequencing library from which the data derives. While multiple software packages exist that can detect when data has been swapped between samples, there are presently no methods to detect library mislabeling. Such mislabellings can negatively impact downstream analyses. Here, we present a tool for reconstructing library relationships from aligned sequence read groups. The basic approach relies on quantifying the similarity of read groups based on their distributions of duplicated insert molecules. Several similarity measures were considered. Library-identity decisions based on similarity-modeling resulted in >91% sensitivity for identifying pairs of read groups from the same library and >98% specificity for identifying pairs from different libraries. Further improvements were seen through unbiased clustering of read groups. Without analytical methods to detect library misidentification, there is no way to know how pervasive this problem is in sequencing data sets. The present tool addresses this unmet need, and provides researchers with unique insight into their data’s chain of evidence.
Dedication

For Emily, and for Peter. I love you more than I can say.
Acknowledgments

This project was made possible by numerous individuals, to whom I am extremely grateful. Thank you firstly to my Thesis Director, Yossi Farjoun. He originally proposed the topic, and graciously worked with me throughout the process to refine the final product. His advice, perspective, feedback, mathematical guidance, and patience have been invaluable. I have learned a lot from him, both directly and by example.

I owe a special thanks to Jon Bloom, whose preliminary work formed the basis for this project. In particular, he developed the concepts for Feature Selection (Section 2.1), as well as the vector and table representations referred to in Feature Extraction (Section 2.2). He also proposed the cosine and Jaccard similarity metrics and developed the Exact Pairs Affinity and All Pairs Affinity measures, including initial Python implementations (whereas I subsequently described these metrics as a product of maximum likelihood estimation, under guidance from my Thesis Director). It has been an honor to extend his work. Thank you to Kathleen Tibbetts, Associate Director of Data Sciences & Data Engineering Operations at the Broad Institute, both for the time to work on this project as well as for providing the computational resources. Thank you to Maura Costello, Justin Abreu, and Alyssa Macbeth of the Broad Institute Genomics Platform for assistance finding data and insight into library construction techniques. Thank you to Jay Carey for code discussions. I am grateful to the Genomics Platform generally for furnishing the data.

Thank you to my parents, Amy DeBrower and Robert Taylor. First, for raising me and loving me unfailingly, and more recently for babysitting regularly so I could work on this project. I could not have done this without them. Thank you to my son, Peter, for teaching me so much about life and for providing sufficient motivation for any task. Most of all, thank you to my wife, Emily Hickey. Her love, labor, patience, advice, support, and sacrifice are more than any person deserves. Thank you, Emily, for making my life
possible, and for making it fun.
Table of Contents

Dedication .............................................................. iv
Acknowledgments ...................................................... v
List of Tables ........................................................... ix
List of Figures .......................................................... x

1 Introduction ........................................................... 1

2 Research Methods ..................................................... 6
   2.1 Feature selection .................................................. 6
   2.2 Feature extraction ............................................... 7
   2.3 Feature improvement ............................................. 10
   2.4 Quantifying read group similarity ............................... 11
   2.5 Exact Pairs Affinity ............................................. 13
   2.6 Extension to Exact Pairs Affinity ............................... 16
   2.7 Modeling similarity values ..................................... 17
   2.8 Pairwise library identity decisions ............................. 18
   2.9 Evaluation criteria .............................................. 20
   2.10 Model selection ................................................ 20
   2.11 Expected false decision rates ................................ 21
   2.12 Clustering read groups ......................................... 21
   2.13 Data sets ...................................................... 25
   2.14 Design and technology choices ............................... 25
   2.15 Implementation ................................................ 27
3 Results................................................................................................. 29
  3.1 Similarity measure distributions .................................................. 30
  3.2 Assessment of feature improvement .............................................. 32
  3.3 Similarity measure comparison ...................................................... 37
  3.4 Performance evaluation ................................................................. 39
  3.5 Unbiased clustering ...................................................................... 42
  3.6 Run time performance .................................................................. 47
  3.7 Results summary ........................................................................... 51

4 Discussion.............................................................................................. 52
  4.1 Results of the present study ............................................................ 52
  4.2 Future directions ........................................................................... 57
  4.3 Summary ......................................................................................... 62

References.............................................................................................. 67
List of Tables

Table 2.1  Data sets summary ................................................. 26

Table 3.1  Results summary .................................................. 51
List of Figures

Figure 2.1 Feature representation ................................. 8
Figure 2.2 Illustration of expected false decision rates ............. 22
Figure 3.1 Visualization of pairwise similarities .................... 31
Figure 3.2 Qualitative effect of feature improvement ................. 34
Figure 3.3 Quantitative effect of feature improvement ............... 35
Figure 3.4 Similarity measure comparison ............................ 38
Figure 3.5 Exact Pairs Affinity sensitivity vs. specificity .......... 40
Figure 3.6 Exact Pairs Affinities for whole genome data sets ...... 41
Figure 3.7 Clustering vs. pairwise library assignment ............... 43
Figure 3.8 Visualization of unbiased clustering ..................... 45
Figure 3.9 Comparison of library determination methods ........... 46
Figure 3.10 Feature extraction time performance .................... 48
Figure 4.1 Relative duplicate set sizes .............................. 54
Chapter 1
Introduction

The cost-efficiency and scale enabled by second-generation DNA sequencing technologies has accelerated our understanding of basic human biology (Lander, 2011; Abecasis et al., 2010; Lek et al., 2016), human history (Lazaridis et al., 2016), the causes of Mendelian diseases (Bamshad et al., 2011), the genetic architecture of complex diseases (Fuchsberger et al., 2016), and the basis of cancer (Zheng et al., 2016).

Data quality is a critical factor in the success of these initiatives. Sequencing studies rely on extremely sensitive tools for read alignment (Li & Durbin, 2009) and variant detection (McKenna et al., 2010). Yet high-throughput sequencing technologies are subject to random errors and biases that impede subsequent analyses (Dohm et al., 2008; Ross et al., 2013).

The importance of quality control to sequencing experiments is attested by the many software tools designed to assess data quality. A significant number of these tools focus on intrinsic properties of the sequenced reads themselves, such as their base content, base quality scores, or the presence of adapter-derived bases (DeLuca et al., 2012; Katta et al., 2015; Okonechnikov et al., 2016; Yang et al., 2013; Cox et al., 2010; Hong et al., 2014; Pandey et al., 2016; Schmieder & Edwards, 2011; Babraham Bioinformatics, 2010). While summary metrics related to the sequence content of a data file are highly important, they provide no insight into the crucial subject of how the data was processed. In particular, they offer no information regarding the integrity of the metadata associated with the sequence reads, such as the individual, sample, and sequencing library to which they belong.

Maintaining correct metadata annotations is a critical aspect of sequence data integrity, particularly for large projects that require hundreds or thousands of samples.
There exist multiple tools which enable data-driven detection of sample-swaps or misla-
beled individuals (Bergmann et al., 2016; Jun et al., 2012; Cibulskis et al., 2011; Huang et al., 2013). These tools are often employed as standard quality control analyses by large sequencing data-generation centers to detect mishandling or contamination in the labora-
tory (Bergmann et al., 2016; Cibulskis et al., 2011). However, no methods exist to detect the mislabeling of data from the same individual. This is an issue in sequencing opera-
tions, as a single genetic sample may be sequenced multiple times.

Before a sample of genetic material can be sequenced using a particular technol-
ogy, such as the Illumina HiSeq, it must be subjected to a series of molecular modific-
tions to make it compatible with the physical and chemical demands of the platform. For example, the DNA or RNA may be fragmented into molecules several hundred bases long. These fragments may then be amplified by polymerase chain reaction (PCR) and adapter molecules may be attached to facilitate interaction with the sequencing machin-
ery, among other steps. Typically, only a portion of any given sample is used in this pro-
cess. A sample-portion that has been through such a process is referred to as a sequenc-
ing library. The process itself is known as library construction. Multiple libraries may be prepared from a single sample. Each library may be sequenced multiple times across different instruments to form numerous ‘read groups’.

Library identity is an important piece of information during the subsequent in-
formatic analysis of these read groups. The process of library construction can introduce artifacts into the final sequence data that must be detected and if possible removed. For example, DNA oxidation during the acoustic shearing that is part of some library prepara-
tion protocols can introduce pernicious artifacts in variant calling (Costello et al., 2013).

More routinely, the PCR amplification included in most library construction pro-
cedures is subject to various biases and can amplify individual base errors. Library anno-
tations play a significant role in identifying reads arising from PCR-duplicates. When a li-
brary is sequenced as multiple read groups, each read group may receive molecules dupli-
cated from a common template. Reads from these molecules should be marked as PCR-duplicates during analysis, even if they are in different read groups. Analytical methods for duplicate marking therefore only mark potential duplicates from different read groups if the groups’ library annotations match (DePristo et al., 2011). Otherwise, they are not marked. Erroneous library matches may cause independent reads to be marked as duplicates, while erroneous mismatches may cause duplicate reads to mistakenly remain unmarked.

Recently, laboratory operations personnel at our sequencing center uncovered a library assignment error during a validation experiment. New aliquots from an original sequencing library were mistakenly given new library IDs. Read groups belonging to the same sequencing library appeared to belong to different libraries. As a consequence of this annotation error, PCR duplicates from the true underlying library could not be identified when they appeared in different read groups. This resulted in artificially low apparent duplication rates (and better apparent data quality) for the affected read groups compared to other batches. While no mis-annotated data were released to collaborators, time and money were wasted searching for the source of this phantom variability. The misassignment of library labels was only discovered through detailed manual investigation of the processing history of each sample. This delayed the successful completion of the validation.

There is presently no data-driven approach to diagnose similar incidents. The tools cited previously for detecting sample-swaps all rely on individual genotypes, which are not expected to vary across libraries from the same individual. It is therefore an open question whether similar library mis-annotations have occurred without detection in the past. Library construction and sequencing involve numerous steps carried out over several days, and library identity must be accurately recorded and maintained through subsequent processing.

Further, there is notably no standard definition of what constitutes a library. An-
notations for read group and library are included in the SAM specification for sequence alignment data, but their semantic meanings are undefined (Li et al., 2009). Given the diversity of sequencing applications, library construction protocols, and sequencing centers, a single definition is impractical.

For example, consider multiplexed hybrid-selection library construction for Illumina sequencing (Fisher et al., 2011; Gnirke et al., 2009). Here, a DNA aliquot is fragmented and amplified via PCR. Amplified molecules are then ligated to standard sequencing adapters. Subsequently, a molecular-enrichment step known as hybrid-selection is performed. This step isolates the subset of ligated molecules corresponding to exons, the protein-coding portions of the genome. Multiple selections may be conducted from a single ligation-product. One sequencing center may consider each amplified, ligated product to be a single library, whereas another may define each separate selection as its own library. Discrepancies multiply when entirely different library construction protocols are considered.

These sorts of detailed procedural definitions are difficult to obtain when aggregating data, and tacit differences in the designation of libraries can hinder data harmonization efforts for reasons previously stated. A method is needed to detect errors or inconsistencies in library identity in order to prevent incorrect analyses.

Here, we present the development of a tool for determining directly from aligned sequence data whether different read groups derive from the same library. The basic approach examines patterns of sequence insert abundance within each read group. These patterns arise from the physical process of library construction and do not depend on the sequence genotype. Because it is a feature of library construction, read groups with more similar insert-patterns are considered more likely to be from the same library. Various similarity measures may be used to quantify the relatedness of read group pairs. The distribution of similarity scores can be modeled for known same-library and known different library read group pairs. From here, there are multiple ways of determining library iden-
tity for a given pair of read groups with unknown library annotations.

We evaluated and refined this general approach across a corpus of 249 clean control read groups, representing a diversity of sequencing applications and library construction techniques.

The final tool achieves greater than 91% sensitivity and 98% specificity when deciding whether two read groups belong to the same library, under recommended usage conditions. Performance was perfect for a number of data sets. As implemented, the tool will provide great utility as a spot-check when unexplained batch effects are observed in sequence data, as a leading indicator that more detailed investigation is necessary. Further refinements are possible that should improve performance and potentially enable use of this tool as a standard operational check in sequence data production.
The purpose of the proposed tool is to determine whether the library annotations attached to sequence data should be trusted. In other words, the tool must determine if a set of read groups bearing the same library annotation are likely to truly originate from the same library (and vice versa). To this end, we require some feature of the data that does not depend on sample identity or genotype, as these are not expected to vary across read groups from the same library. The properties of this feature must be learned over a set of data with trusted library annotations. These properties must then be used to discriminate collections of read groups that are same-library from others that are different-library. Each phase of this general approach is considered below.

2.1 Feature selection

A sequencing library, however constructed, is a population of physical DNA or RNA molecules derived from a sample of genetic material. These molecules represent the original material’s base content during the sequencing process. Such molecules may be referred to as inserts. The processes that generate inserts during library construction are typically stochastic. The set of specific inserts that are produced is non-deterministic, while the processing steps may be subject to certain biases that leave different regions of the reference genome represented non-uniformly.

For example, genomic DNA may be subjected to acoustic shearing, which cuts the DNA to produce inserts with a range of sizes. The DNA is not cut at specific sites within the reference, and different inserts will be produced from each reaction. Yet each possible insert is not equally likely, as some portions of the genome are more easily sheared.

Many library construction protocols also involve PCR. During PCR, each tem-
plate molecule is amplified according to its individual efficiency. Various physical properties of the templates, such as their length and GC content, will bias these efficiencies. The result is a non-uniform collection of amplification products.

Owing to such processes, the distribution of inserts is a characteristic feature of each individual library.

A read group is generated when a library is sampled and the sample is subjected to an independent sequencing readout. As samples of the same underlying population, read groups from the same library should contain a similar distribution of insert molecules. To determine whether two read groups belong to the same library, we may therefore examine their insert distributions and decide whether they are sufficiently similar. Measures for quantifying similarity and methods for generating decisions are discussed below. First, the insert distributions must be obtained and represented in a form that is useful for downstream calculation. This is the task of feature extraction.

2.2 Feature extraction

Given aligned sequence data, how can reads from the same insert molecule be identified, and distinguished from other inserts? This is the same question that must be answered when marking PCR duplicates, and the same methodology may be employed. For the purpose of determining molecular identity, an insert may be uniquely defined by the aligned positions of its unclipped 5’ ends within the reference genome. During Illumina sequencing, it is typical to sequence two reads for every insert, one from each end of the molecule, to form a mated read-pair (see Figure 2.1a; Lander et al. 2001). In this case, the relevant alignment positions are the start of one record and the end of its mate (taking orientations appropriately into account).

Using these criteria, we can sort reads into sets representing duplicates of the same unique insert. Figure 2.1a presents a hypothetical data set consistent of only five read pairs, denoted α through ε. Each read pair is depicted adjacent to the region of a hypo-
Figure 2.1. Feature representation. (a) Alignment illustration. One read is sequenced from each end of an insert molecule. Some intervening bases may not be sequenced (represented by the dashed interval). Read pairs are aligned against the reference genome. Pairs from group $A$ are shown in blue. Pairs from group $B$ are shown in red. The aligned positions of the unclipped 5’ ends of a read and its mate define a duplicate set. Here, read pairs are indexed $\alpha$ through $\epsilon$, with duplicate sets denoted similarly. Duplicate sets may span overlapping regions of the reference, as do $(\gamma_1, \delta_1)$ and $(\delta_2, \varepsilon_2)$. Strand information is not depicted in this illustration. (b) Vector representation (discussed in text). (c) Table representation (discussed in text).
theoretical reference chromosome spanned by its alignment. In most cases, each read from a pair is sorted into a separate duplicate set. See the HTSJDK software documentation for full details on duplicate set sorting (Samtools, 2009); our only modification has been to ignore library labels when assembling duplicate sets. Here, insert $\alpha$ is shown represented by duplicate two sets: $\alpha_1$ and $\alpha_2$.

One may represent the insert distribution for a particular read group as a vector of duplicate counts, as shown in Figure 2.1b. Each read group is encoded as a vector. Each unique duplicate set is assigned an index in the vector. The value at that index represents the number of reads from the group that are in the duplicate set. In Figure 2.1a, the duplicate set labeled $\beta_1$ contains three reads from read group $A$ in blue. The value of the Read Group $A$ vector at index $\beta_1$ is therefore “3”. Note that duplicate set $\gamma_1$ contains no read-pairs from read group $B$, so the value of the Read Group $B$ vector at index $\gamma_1$ is “0”. Given more read groups and many duplicate sets, we expect that a large fraction of duplicate sets will contain no reads for at least one group. We therefore expect these vectors to be quite sparse in practice.

The maximum possible dimensionality of a duplicate set vector is equal to the number of potential unique alignment-end positions in the reference genome. This is an incredibly large number for any realistic genome. For a given data set, we can restrict the dimensionality of our vectors to the number of unique duplicate sets containing reads from at least one read group. For our toy example in Figure 2.2, this was ten. In practice, we expect extremely high-dimensional vectors. Indeed, over 250 million duplicate sets were observed for thirty two hybrid-selected read groups from four libraries in our Exomes data set.

As we expect the duplicate set vectors to be extremely high-dimensional and quite sparse, a reduced representation is desirable for efficient storage and computation. Figure 2.1c illustrates one approach to data reduction. When comparing read groups, we are less interested in the particular identity of an insert than in how often we observe it across
different read groups. Duplicate set $\beta_1$ from Figure 2.1a contained three reads from group A and three from group B. We represent this pattern of read group occurrence as the tuple $(A, A, A, B, B, B)$. Only two duplicate sets in all of our data show this pattern: sets $\beta_1$ and $\beta_2$. We therefore construct a table containing the pattern tuple associated with the count, “2”. Four duplicate sets show the pattern $(A, B)$, indicating one read-pair from each group. These were $\alpha_1$, $\alpha_2$, $\delta_1$, and $\delta_2$. In the table, the count associated with the pattern $(A, B)$ is therefore “4”. The information from duplicate sets representing $\alpha$ and $\delta$ are thus collapsed into a single table entry. In practice many duplicate sets share a pattern of occurrence, and the number that can be merged into a single table entry is large.Collapsing individual duplicate sets in this way allowed for data reduction without sacrificing crucial patterning information.

2.3 Feature improvement

Two data cleaning steps were introduced to feature extraction with the aim of reducing noise in downstream analysis. First, problematic reads were filtered out prior to duplicate set assembly. All data were paired-end, and mated reads were filtered similarly to avoid creating aberrant fragment records. Mate pairs were filtered out if either read failed the Illumina base-calling “chastity filter”. Secondary reads from multiple-alignments and supplementary reads from chimeric-alignments were also filtered. Mates were also excluded if they were not mapped in a proper pair (for example, if they carried unexpected orientations due to unknown problems during sequencing or alignment).

Finally, mate pairs were excluded if one of the reads displayed low mapping quality. This was done primarily to exclude alignments localized to low-quality or repetitive portions of the genome. Such regions display extreme coverage peaks owing to poor intrinsic alignability and result in large duplicate sets. These large duplicate sets would bias a number of the similarity measures examined in this study. Mapping quality has proven to be a valid and convenient proxy for excluding alignments at these regions (Hogstrom,
The second step was to identify and remove optical duplicates. This is a special class of duplicate reads that is private to the Illumina sequencing platform. They occur when the sequencer misidentifies a single-molecule cluster as two separate clusters. As they arise during sequencing itself, downstream of library construction, these reads add noise to our feature measurement. Because they only occur within individual read groups, they have the effect of elevating within-group duplicate counts relative to between-group duplicate counts, deflating measures of similarity.

To facilitate optical duplicate identification, information regarding a read’s physical location during the sequencing run is embedded in the read name. If two duplicate reads are sufficiently close to one another, one read is marked as an optical duplicate. Mated reads carry the same physical location information and are treated consistently. Optical duplicate reads are excluded from consideration prior to determining the insert distribution.

The affect of these steps was evaluated in this study (see results in Section 3.2). Both of these steps may be optionally disabled in the final implementation.

2.4 Quantifying read group similarity

As noted in Section 2.2, a read group may be conceptually represented as a vector of insert counts. A number of common measures exist for numerically scoring the relatedness of two feature vectors. Several of these were evaluated during the course of developing this tool. Euclidean distance is a natural measure for comparing two vectors. However, it varies depending on the relative magnitude of the vectors (i.e. the read group sizes). We must account for the possibility of read groups with wildly different numbers of total reads. Small read groups are occasionally sequenced to add additional coverage to aggregated sequence data. A better measure is Cosine Similarity, defined for two vectors $a$ and
$b$ as:

$$\cos \theta = \frac{a \cdot b}{|a||b|}$$

This depends only on the angle between the vectors, not their magnitude. Unlike Euclidean distance, it ranges from 0 to 1, with higher values the closer two vectors are to being parallel. It is therefore attractive as a statistic. For these reasons, we avoided Euclidean distance in favor of Cosine Similarity.

Another commonly used measure is the Jaccard Similarity. This is also bounded $[0, 1]$, with higher values denoting greater similarity. It is defined as the ratio of the intersection of two sets to their union:

$$J(a, b) = \frac{|a \cap b|}{|a \cup b|}$$

Our data are vectors of duplicate counts, rather than sets. For calculating the Jaccard measure, we treated each read group’s duplicate count vector as a binary vector. A value of “1” at a particular index indicates that the corresponding duplicate set contained at least one read from this read group. For two read groups $A$ and $B$, the intersection $|A \cap B|$ therefore represents the number of duplicate sets seen at least once in both $A$ and $B$. The union $|A \cup B|$ represents the number of inserts seen in at least once in either $A$ or $B$.

Both Cosine and Jaccard similarity are intuitive and efficient to calculate, and both were evaluated as part of this project. However both measures suffer certain expected drawbacks. Neither measure is based on a null hypothesis regarding the relationship of read groups from a library. They carry no a priori expectation as to how high a value is sufficient to consider two read groups as being from the same library. We also suspected they might prove overly sensitive to non-library-based differences between read groups, such as differences in total sequencing depth or in the total genomic territory represented by the sequences.
Finally, given the expected sparsity of our feature vectors, we were concerned that the Jaccard measure would be low for all read groups regardless of their library relationships. This concern was somewhat born out in practice by the consistently low values observed for this measure (see Figure 3.1c). These concerns motivated the development of a set of custom similarity measures more specific to the problem at hand.

2.5 Exact Pairs Affinity

A more principled similarity measure was developed based on the null hypothesis that a given pair of read groups \( A \) and \( B \) truly belong to the same library. Under this assumption, read groups are generated from a library according to the sampling process described in section 2.1. Each read in a read group is labeled with group and library identifiers. During feature extraction, reads from both groups are pooled, while retaining their labels. Reads in the pool are sorted into duplicate sets representing each unique insert molecule from the original library. We model the distribution of read group labels within duplicate sets under the null hypothesis as follows:

Let the probability that any given read in the pool belongs to read group \( A \) be denoted \( \alpha \). Let the probability that it belongs to read group \( B \) be denoted \( \beta \). The probabilities \( \alpha \) and \( \beta \) are determined by the relative sizes of the read groups. We repeatedly draw a two-read duplicate set from the total pool of sets. For simplicity, larger duplicate sets are ignored. For each set, we examine the pattern of read group labels. There are three possible read group patterns: \((A,A)\), \((A,B)\), and \((B,B)\).

Each observed duplicate set can be categorized as matching one of these patterns. We define the pattern \((A,A)\) as category 1, the pattern \((A,B)\) as category 2, and the pattern \((B,B)\) as category 3. Let \( X \) be a set of random variables where \( X_i \) represents the total number of observations in category \( i \). Further, let \( p \) be a set of probabilities where \( p_i \) represents the probability that a given observation is of category \( i \), with \( \sum_{i=1}^{3} p_i = 1 \). We examine each size-two duplicate set from the population and record the observed counts.
$x_i$ for each category. Note that $\sum_{i=1}^{3} x_i = n$, the total number of size-two duplicate sets. The overall probability of observing a particular collection of counts $x$ given the set of probabilities $p$ is given by the multinomial distribution.

We recast our probabilities $p$ in terms of $\alpha$ and $\beta$, such that:

$$
\begin{align*}
p_1 &= \alpha^2 \\
p_2 &= 2\alpha\beta \\
p_3 &= \beta^2
\end{align*}
$$

Note the coefficient “2” in the case of $p_2$, which arises because the category $i = 2$ is satisfied by both patterns (A,B) and (B,A). From the multinomial distribution, the likelihood of a particular $\alpha$ and $\beta$ given a set of observed counts $x$ is therefore:

$$
L(\alpha, \beta \mid x) = \left( \frac{n!}{\prod_{i=1}^{3} x_i !} \right) \alpha^{2x_1} (2\alpha\beta)^{x_2} \beta^{2x_3}
$$

In reality, we are also interested in the alternate hypothesis that the two read groups truly belong to different libraries. In this case, we expect each read group to be more self-similar, and less similar to each other. We therefore expect fewer (A,B) and (B,A) observations than would be found due to chance under the same-library null model. To capture this, we introduce a term $\varepsilon$ to our expression for the probability $p_2$:

$$
p_2 = 2\varepsilon\alpha\beta
$$

This term should vary over $[0, 1]$ and represents the reduction in cross-read-group observations relative to the expectation under the same-library null. The likelihood of our model parameters given our data is therefore:

$$
L(\varepsilon, \alpha, \beta \mid x) = \left( \frac{n!}{\prod_{i=1}^{3} x_i !} \right) \alpha^{2x_1} (2\varepsilon\alpha\beta)^{x_2} \beta^{2x_3}
$$

(2.1)
We wish to find the values of \( \epsilon, \alpha, \) and \( \beta \) that maximize this likelihood given the observed counts \( x \). For mathematical convenience, we will maximize the logarithm of the likelihood function, since extrema will occur at the same values as for the original function. Importantly, our likelihood function is subject to the following linear equality constraint:

\[
1 = \sum_{i=1}^{3} p_i \tag{2.2a}
\]

in other words,

\[
1 = \alpha^2 + 2\epsilon\alpha\beta + \beta^2 \tag{2.2b}
\]

Maximization of a function along an equality constraint can be accomplished by the method of Lagrange Multipliers. Let \( f(\epsilon, \alpha, \beta) \) denote the likelihood function to be maximized, from Equation 2.1. Let \( c = g(\epsilon, \alpha, \beta) \) denote the equality constraint from Equation 2.2b, where \( c \) is the constant “1”. We introduce the Lagrange multiplier \( \lambda \) and define the Lagrange function as:

\[
\mathcal{L}(\epsilon, \alpha, \beta, \lambda) = \ln f(\epsilon, \alpha, \beta) - \lambda(g(\epsilon, \alpha, \beta) - c)
\]

\[
= \ln \left( \frac{n!}{\prod_{i=1}^{3} x_i!} \right) \alpha^{2x_1} (2\epsilon\alpha\beta)^{x_2} \beta^{2x_3} - \lambda(\alpha^2 + 2\epsilon\alpha\beta + \beta^2 - 1)
\]

Critical points for this function are found where the gradient is equal to 0:

\[
\nabla_{\epsilon, \alpha, \beta} \mathcal{L}(\epsilon, \alpha, \beta, \lambda) = 0
\]
This yields the following system of equations:

\[
0 = \frac{2x_1 + x_2 - 2\lambda \alpha (\alpha + \beta \epsilon)}{\alpha}
\]

\[
0 = \frac{x_2 + 2(x_3 - \lambda \beta (\alpha \epsilon + \beta))}{\beta}
\]

\[
0 = \frac{x_2}{\epsilon} - 2\lambda \alpha \beta
\]

Solving this system for \( \epsilon \) gives a maximum likelihood estimate of this parameter:

\[
\epsilon = \frac{x_2}{2\sqrt{x_1} \sqrt{x_3}}
\]

We take this quantity as a custom similarity measure which we call Exact Pairs Affinity. As noted, we expect this measure to vary over the range \([0, 1]\), with higher values implying greater similarity. In practice, values in excess of one were observed for some same-library read group pairs. This may simply be a consequence of the estimation process, though a more thorough investigation into particular cases producing these values is desirable. Here, values are capped at one for downstream analysis.

2.6 Extension to Exact Pairs Affinity

As defined, the Exact Pairs Affinity metric only considers duplicate sets of size two. In an effort to recover signal from larger duplicate sets, we developed several additional measures that extend the Exact Pairs concept. Consider a duplicate set containing four reads from read groups \( A \) and \( B \): \((A, A, A, B)\). From this set we may form a collection of all \( \binom{4}{2} = 6 \) pairwise combinations of reads. Exact Pairs Affinity was modified to examine each combination from a larger duplicate set as though it were an independent size-two duplicate set. We termed the resulting measure All Pairs Affinity.

We were concerned that a minority of extremely large duplicate sets would dominate the All Pairs Affinity signal due to quadratic growth in the number of combinations formed from a set as set size increases. We therefore produced a family of similarity mea-
sures that cap the maximum size of duplicate sets examined by the All Pairs process. We collectively name these “N”-limited Affinity, where N indicates the maximum size of duplicate sets considered. For this study we evaluated Ten, Five, Four, and Three-limited Affinity measures. Note that Exact Pairs Affinity can be considered an example of such a measure where N is two.

Finally, we explored a second means of recovering signal from duplicate sets larger than size two. Again, we form all pairwise combinations of reads from the set. Instead of examining every combination, we select one combination of reads among the possibilities as the read group pattern to count. The particular combination to be examined is chosen at random, in hopes that this selection will be unbiased across many insert sets. We have termed this similarity measure Random Pairs Affinity.

2.7 Modeling similarity values
Pairwise similarity values are useful for quantifying the relatedness of read groups. However, they are not directly interpretable in deciding whether two read groups belong to the same library. Similarity provides a measure, but no guidance as to when this measure is sufficiently extreme as to justify a ‘same-library’ decision. Without additional information, we cannot tell from the magnitude of a given similarity value how likely it is that such a value could be produced by different-library read groups, or by same-library read groups. Ideally, for any pair of read groups, we wish to quantify our relative confidence in their being truly same-library versus being truly different-library.

To obtain such a measure of confidence, we modeled the observed distribution of similarity values from ground truth data. First, we gathered a set of read groups which we were very confident carried correct library annotations (see Section 2.13). A similarity value was computed for each pair of read groups. Next, one distribution was fit to the observed similarity values for all true same-library pairs, and a second distribution was fit for all true different-library pairs.
The beta distribution was chosen for these models. Each of our similarity measures varies continuously over the range $[0, 1]$. In practice, we observed significant skew in similarity histograms, especially for different-library pairs whose values tended to cluster close to zero. Given these observations, the beta was an obvious choice of distribution.

A small set of faux similarities were added to both the same-library and different-library observations prior to fitting. This added slight additional probability to the tails of beta distributions across the full range of possible similarity values. Our limited training data may not fully represent the range of similarities observed for unseen read groups, and faux data was added to provide a small additional uniform uncertainty to account for this. The set of faux observations $f$ that were added was:

$$f = [0.2, 0.4, 0.6, 0.8]$$

Finally, for each new read group pair with untrusted library annotations, we calculated the similarity value and evaluated both distributions against this. We obtained the likelihood that the observed value arose from the same-library distribution as well as the likelihood that it arose from the different-library distribution. We then computed the log of the ratio of the same-library likelihood to the different-library likelihood.

Thus, if $(\alpha_s, \beta_s)$ represent the beta parameters for the same-library distribution, and $(\alpha_d, \beta_d)$ represent the parameters for the different-library distribution, and $B$ is the beta function, the same-library log likelihood ratio is given by:

$$LLR(x) = \ln \left( \frac{1}{B(\alpha_s, \beta_s)} x^{\alpha_s - 1} (1 - x)^{\beta_s - 1} \right) - \ln \left( \frac{1}{B(\alpha_d, \beta_d)} x^{\alpha_d - 1} (1 - x)^{\beta_d - 1} \right)$$

2.8 Pairwise library identity decisions

Log-likelihood ratios express our degree of confidence in read group relatedness, and are a primary output of the tool. However, we also wish to make a conclusive decision as to
whether to trust library labels. One approach is to consider each pair of read groups independently. One can then compare each pair’s log-likelihood ratio against some threshold. We will then call a pair ‘same-library’ when we observe confidence in excess of the threshold value. Any pair with confidence at or below the threshold will be called ‘different-library’.

Note that, because pairs are considered independently, this approach is not guaranteed to split the collection of read groups into a consistent partition of libraries. For example, consider three read groups A, B, and C. We may call the pairs (A, B) and (B, C) same-library, but call (A, C) different library. On the other hand, because the distributions and threshold are pre-trained over appropriate control data, this approach is immediately applicable to unseen data. It does not depend on the number or nature of read groups, and can be applied to even a single pair of read groups.

One could conceive of numerous methods for obtaining the threshold during training. For simplicity, we took the midpoint between the values for trusted same- and different-library pairs in the following sense. Let $LLR_s$ denote the set of all log-likelihood ratios for pairs of read groups in the training set carrying the same trusted library annotation. Let $LLR_d$ denotes the set of log-likelihoods for different-library pairs. The threshold may then be calculated as:

$$\text{threshold} = \frac{\min(LLR_s) + \max(LLR_d)}{2}$$

The resulting threshold has the potential to be negative in some cases. In this event, the threshold was set to zero so that a read group pair would have to be at least equally likely to be same-library as different-library in order to be called ‘same-library.’

We trained a separate log-likelihood threshold alongside each individual set of beta distributions we fit, and we evaluated the combination of the models and the threshold together.
2.9 Evaluation criteria

Effective model selection depends on having some means of comparing model performance. To this end, two evaluation criteria were defined. The key question is whether the algorithm generates the correct same/different-library decisions. We therefore focused on the sensitivity and specificity of the test for library identity. For each pair of read groups, we take a ‘same-library’ decision to be a positive call, and a ‘different-library’ decision to be a negative call. When evaluating the algorithm over a trusted test set whose true library annotations are known, pairwise sensitivity and specificity are given by:

\[
sensitivity = p(\text{same-library} | \text{truly same library})
\]

\[
specificity = p(\text{different-library} | \text{truly different-library})
\]

2.10 Model selection

In Chapter 3, both feature improvement and the choice of distance metric are assessed for their impact on our ability to fit good distributions. These choices are part of the form of the final model. To improve our estimates of a particular model’s sensitivity and specificity on unseen data, 5-fold cross-validation was performed (Bishop, 2015, p. 33; O’Neil & Schutt, 2014, p. 67). We split our data into a training set comprising 4/5 of the read groups and fit distributions and a threshold. We held the remaining 1/5 out as a test set, and used this to evaluate the performance of our distributions as described in section 2.9. We repeated this procedure five times, holding a different fifth out as the test set each time. When evaluating feature improvement, the five distributions were used directly. When comparing distance metrics, performance metrics were averaged across the five rounds.

Having selected the final form for our model, we then trained a final set of distributions and a threshold for each data set using all of the data. Performance estimates were
taken from the same data. Because training data was used in the evaluation, the reported sensitivity and specificity metrics are likely to be overestimates of the true performance on unseen data to some degree.

2.11 Expected false decision rates

To assess the impact of feature improvement on the inherent quality of our similarity distributions, we defined expected false decision rates based solely on these distributions’ properties. As described in sections 2.7 and 2.8, same-library and different-library distributions are used to transform a similarity value into a log-likelihood ratio, which is compared to a threshold value. If we know the similarity value corresponding to the likelihood threshold, we can quantify the proportion of each distribution that falls on either side.

For the same-library distribution, we evaluate the cumulative probability function to find the percentage of the total probability contained below the threshold. This gives us the fraction of values drawn from the same-library distribution that we expect will be called ‘different-library’. This is represented by the red-shaded region in Figure 2.2.

For the different-library distribution, we subtract the cumulative probability from one, to find the percentage of the total probability contained above the threshold. This is the fraction of different-library values that we expect will be called ‘same-library’. This is represented by the blue-shaded region in Figure 2.2.

In practice, the threshold is derived from log-likelihood-ratios observed in training data, and is not known as a similarity value. We therefore start with the threshold log-likelihood ratio and solve for the corresponding similarity numerically.

2.12 Clustering read groups

As an alternative to thresholds, we explored clustering as a means of partitioning read groups into inferred libraries. Several factors contributed to the choice of algorithm. Clus-
Figure 2.2. Illustration of expected false decision rates. The blue line depicts the probability density function (pdf) of a hypothetical distribution of similarity values for different-library read group pairs. The red line depicts a pdf of a hypothetical distribution of same-library values. The dotted line shows the similarity value of the log-likelihood-ratio threshold. It is shown here at the intersection of distributions, but this is not necessarily the case in practice. The red-shaded region represents the expected rate of false different-library calls (or one minus the expected sensitivity). The blue-shaded region represents the expected rate of false same-library calls (or one minus the expected specificity). Gaussian distributions are depicted for clarity, whereas beta distributions are used in practice.

Clustering must be unsupervised, as we do not trust the library annotations associated with our read groups. We do not know the true number of libraries \textit{a priori}, ruling out algorithms such as k-means which require one to pre-specify the number of clusters (Bishop, 2015, p. 424; O’Neil & Schutt, 2014, p. 82). Another common clustering technique involves trying to fit Gaussian mixture models to the data via the expectation-maximization algorithm (Bishop, 2015, p. 110, 430). However, in section 2.1 we described read groups as a sample from a discrete distribution of insert molecules constituting the library. As such, the Gaussian assumption is not a good one, and this technique is inappropriate. Other clustering methods could be considered in the future.

For this study, we chose to explore graph-based clustering. Here, we represent a data set as a fully-connected graph whose nodes are each read group. Each pairwise similarity value is the weight of the edge between two read group nodes in the graph.

We can represent this graph as a weighted adjacency matrix. For \( n \) read groups, we assemble an \((n \times n)\) matrix. The row and column indices of the matrix refer to the read groups, in the same order. Each element of the matrix therefore represents a pair of read groups (i.e. an edge in the graph), and it is assigned the pair’s similarity value. The
diagonal of the matrix is set to zero, as self-similarities are not calculated. If $s_{ij}$ is the similarity of read groups $i$ and $j$, the adjacency matrix $AM$ is then:

$$AM = \begin{bmatrix} 0 & \cdots & s_{ij} & \cdots & s_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{ij} & \cdots & 0 & \cdots & s_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{in} & \cdots & s_{nj} & \cdots & 0 \end{bmatrix}$$

Note that this matrix is symmetric, as our similarities $s_{ij}$ and $s_{ji}$ are equal for all $i$ and $j$.

Network research has provided a number of algorithms for partitioning such graphs into a set of communities. A community is roughly defined as a subset of the graph whose nodes are more strongly connected to each other than to nodes outside it (Newman & Girvan, 2004). No assumptions or conditions are imposed regarding the size or uniformity of a community relative to other communities.

To detect such communities, we employ the ‘Louvain’ algorithm proposed by Blondel et al. (2008). First, we must formalize our notion of connection density. To this end, we assign a modularity $Q$ to each possible community partition. This modularity measures the density of edges within communities as compared to edges between communities (Blondel et al., 2008; Waltman & van Eck, 2013). For a set of communities $c$ partitioned from a weighted graph:

$$Q = \frac{1}{2m} \sum_{ij} \left[ s_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

where:

- $c_i$ is the community to which node $i$ belongs

23
• The function \( \delta(c_i, c_j) \) indicates whether nodes \( i \) and \( j \) belong to the same community:

\[
\delta(c_i, c_j) = \begin{cases} 
1 & \text{if } c_i = c_j \\
0 & \text{if } c_i \neq c_j 
\end{cases}
\]

• \( s_{ij} \) is the weight of the edge between nodes \( i \) and \( j \)

• \( k_i \) is the degree of vertex \( i \), i.e. the sum of its weights: \( k_i = \sum_j s_{ij} \)

• \( m \) is the total weight of the edges in the graph: \( m = \frac{1}{2} \sum_{ij} s_{ij} \)

Taking modularity as a measure of community quality, the Louvain algorithm attempts to find the set of communities that has the highest modularity. This problem is known to be computationally NP-hard, and so the algorithm performs heuristic optimization. Initially, each vertex is taken as its own community. Each node is then sequentially reassigned to the community that results in the greatest modularity increase. This is repeated until node reassignment cannot increase modularity. Communities resulting from this process are then taken as individual nodes in a new graph-of-communities, by summing the edge weights of their constituents. The entire algorithm is then repeated starting from this new, coarser graph. Repetitions are halted when modularity can no longer be increased (Blondel et al., 2008).

To evaluate the success of clustering results, we calculate the same pairwise evaluation criteria as in section 2.9.

In addition to clustering raw similarity metrics, we evaluated clustering using likelihood ratios as edge weights. As a measure of same-library confidence, these values should better represent the distance between the read groups. In practice, we observed likelihood ratio values that were extremely high for same-library pairs. Likelihood ratios were capped at 1000 prior to clustering.
2.13 Data sets
To train distributions and thresholds and to evaluate the overall performance of the algorithm, several control data sets were obtained. These represented a diversity of data types and library construction (LC) protocols. Included were data from whole genomes (both PCR-plus and PCR-free LC), hybrid-selected exomes (for two different target sets), and RNA sequencing (two different LC protocols). See Table 2.1 for an overview of the data. For each data type, a combined data set was created from the union of read groups from different LC protocols. An overall combined data set was generated from all read groups except PCR-free genomes, which were not expected to perform well (see Results).

All DNA-based read groups were derived from aliquots of the NA12878 control sample, which has been put forward as a standard for use in informatics method development (Zook et al., 2014). RNA sequencing read groups were derived from the commonly used K562 commercial human cell line (Lozzio & Lozzio, 1979). These data were originally used as processing controls during routine sequencing operations. Chain of custody information was captured automatically during library construction by liquid handlers integrated into a custom Laboratory Information Management System (LIMS). Library and read group annotations were automatically assigned by the LIMS. As such, we have high confidence that these annotations are correct. All data had been previously aligned against the GRCh37 human reference genome build (Genome Reference Consortium, 2009). DNA read groups were aligned using the Burrows-Wheeler Aligner (Li & Durbin, 2009), while RNA sequencing read groups were aligned using TopHat (Trapnell et al., 2009).

2.14 Design and technology choices
We chose to develop the feature extraction utility as a new command line program within the Picard suite of sequence manipulation tools (The Broad Institute, 2009). The rea-
Table 2.1.
Data sets summary.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Short Name</th>
<th>Library Type</th>
<th>Sample</th>
<th>Read Groups</th>
<th>Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Illumina Content Exomes</td>
<td>ice_only</td>
<td>Hybrid Selection, Illumina Content</td>
<td>NA12878</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>Agilent Exomes</td>
<td>agilent_only</td>
<td>Hybrid Selection, Agilent</td>
<td>NA12878</td>
<td>42</td>
<td>5</td>
</tr>
<tr>
<td>Exomes (combined)</td>
<td>exomes</td>
<td>(union of above)</td>
<td>NA12878</td>
<td>74</td>
<td>9</td>
</tr>
<tr>
<td>RNA, Strand Agnostic</td>
<td>strand_agnostic</td>
<td>cDNA Shotgun Strand Agnostic</td>
<td>K562</td>
<td>48</td>
<td>3</td>
</tr>
<tr>
<td>RNA, Two Sense</td>
<td>two_sense</td>
<td>cDNA Shotgun Read Two Sense</td>
<td>K562</td>
<td>26</td>
<td>3</td>
</tr>
<tr>
<td>RNA (combined)</td>
<td>rna</td>
<td>(union of above)</td>
<td>K562</td>
<td>74</td>
<td>6</td>
</tr>
<tr>
<td>PCR-free Genomes</td>
<td>pcr_free</td>
<td>PCR-Free Human Whole Genome Shotgun</td>
<td>NA12878</td>
<td>48</td>
<td>2</td>
</tr>
<tr>
<td>PCR-plus Genomes</td>
<td>pcr_plus</td>
<td>Low Input Human Whole Genome Shotgun</td>
<td>NA12878</td>
<td>53</td>
<td>2</td>
</tr>
<tr>
<td>Genomes (combined)</td>
<td>genomes</td>
<td>(union of above)</td>
<td>NA12878</td>
<td>101</td>
<td>4</td>
</tr>
<tr>
<td>Combined</td>
<td>combined</td>
<td>(union of rna, exomes, pcr_plus)</td>
<td>NA12878</td>
<td>201</td>
<td>17</td>
</tr>
</tbody>
</table>

sons were several. Firstly, the speed and memory performance offered by Java cannot
be matched by higher level scripting languages. This is critical, as sorting aligned se-
quencing reads into duplicate sets is computationally intensive. Secondly, we wished to
take advantage of the easily extensible utility methods provided by the underlying HT-
SJDK library for manipulating sequence alignment data in the SAM format and for sorting
reads into duplicate sets (Li et al., 2009; Samtools, 2009). Picard builds on this library
and adds convenient functionality for specifying command line arguments and producing
parsable output metrics. Finally, Picard is a widely-used open-source product. Potential
inclusion in the main Picard repository would facilitate easy distribution of this method
to a large user base.

We chose to perform iterative development of the overall algorithm in Jupyter
Notebook (Kluyver et al., 2016). This included the performance evaluation, the cross-
validated comparison of similarity metrics, and the exploration of unbiased clustering.
The Jupyter platform facilitated interactive exploratory analyses that could be composed
into more repeatable units. The ability to display tabular and graphical output directly
adjacent to its generating code was invaluable.

Python was chosen as the language for developing analytical features. This was
motivated both by ease of programming as well as the ability to access key scientific
computing packages such as NumPy for linear algebra (van der Walt et al., 2011), SciPy
for statistics including fitting and evaluating beta distributions (Jones et al., 2001–), Net-
workX for graph manipulation and visualization (Hagberg et al., 2008), Python-Louvain for community detection (Aynaud, 2011), Pandas for data annotation and generating tables (McKinney, 2010), and Matplotlib for generating data graphics (Hunter, 2007).

2.15 Implementation

As noted above, feature extraction was implemented as a Picard command-line program in Java. Feature information was written out in JSON format to facilitate ingest into Python for subsequent analyses. The ability to cache feature data to JSON will be retained in order to update likelihood model parameters and as an output for debugging purposes.

We implemented a complete script in Python for parsing feature data, computing each similarity measure, fitting beta distributions, calculating likelihood ratios, deriving thresholds, performing cross-validation, clustering read groups, and evaluating results. The results presented in this study were generated using this script.

Having established the validity of our analytical approach, our chosen similarity measure was reimplemented in Java as part of a Picard command-line program to generate same/different-library decisions on novel data. Should this method garner sufficient community interest, the Java implementation will be submitted for inclusion in the main Picard source repository. It is under active development. Currently, the analysis portion exists as a separate program, though our intention is to merge this functionality with feature extraction. Users must presently supply their own distribution parameters. Going forward, we will supply the models obtained in this study as sensible defaults. These programs can be found on a fork of the Picard repository here:

https://github.com/bradtaylor/picard

Training-phase features such as distribution fitting will not ported to Java, as they represent an independent operating mode and we wish to keep the interface from becoming overly complex. Further, convenient library utilities to perform these functions either
do not exist in Java or are unlikely to be accepted into Picard. The Python analysis script therefore serves as an important component to the overall analysis tool, not only as a platform for piloting future refinements, but as a means of updating recommended parameters when run with new inputs.
Chapter 3
Results

To infer library relationships among read groups directly from sequence alignment data, we developed an approach based on quantifying the similarity between two read groups’ patterns of insert duplication. See Section 2.1 for details. In evaluating the performance of this approach, we examined three attributes that had notable impact on the quality of results. These were the choice of similarity measure, the data cleaning procedures employed during feature extraction, and the method by which similarity values are used to recover library information.

To train statistical models and assess the algorithm, a diverse collection of sequencing data was compiled. As described in Section 2.13, these data comprised two distinct library types for each of three different fundamental data types: DNA whole genomes, hybrid-selected exomes, and RNA sequence data. For each data type, a third set was formed by combining the two library types. Data were derived from control samples subjected to automated processing, and as such we have high confidence in their preexisting library annotations.

For each of these data sets, read groups were processed together to derive shared patterns of insert duplication. A similarity value was assigned to each pair of read groups. We used the read group’s existing library annotations as ground truth data to model these similarity values. Read group pairs were called ‘same-library’ if their annotations matched, and ‘different-library’ otherwise. Separate distributions were trained on same-library and different-library pairs.

For each unseen pair of read groups in the test data, we computed a similarity value and used the pre-trained distributions to calculate a log likelihood ratio. This ratio indicates how likely it is that the observed similarity value belongs to the same-library
population vs the different-library population (see Section 2.7). To make a definitive decision, we compared this ratio to a predetermined threshold trained alongside the distributions. Pairs displaying log-likelihood ratios in excess of the threshold were called ‘same-library’, while remaining pairs were called ‘different-library’.

3.1 Similarity measure distributions

Multiple similarity metrics were evaluated against every data set. Similarity values and their distributions were visualized. Figure 3.1 provides an example of four similarity measures (Exact Pairs Affinity, All Pairs Affinity, Jaccard Similarity, and Cosine Similarity) calculated over the Illumina Content Exomes data set.

In general, the Exact Pairs Affinity measure provided the largest degree of separation between similarity values assigned to same-library and different library pairs. All Pairs Affinity produced values nearer to one regardless of whether a pair was same-library or different-library, and there was poor separation between the values for these two populations. This generic inflation is understandable. From the definition given in Section 2.6, note that duplicate sets of size greater than two are broken out into pairwise combinations of reads. A single read from a large duplicate set therefore contributes more information to the overall metric than a read from a smaller duplicate set, due to its participating in more pairs. This problem quickly escalates with increasing duplicate set size, as the number of combinations grows quadratically. A small number of large duplicate sets may therefore overwhelm the information contributed by the remainder of the data.

This is especially troubling as Illumina sequencing data has been found to generate regions of extreme apparent duplication, likely due to issues aligning reads against certain difficult regions of the reference genome (Hogstrom, 2016). As they are partially a feature of the underlying reference genome, we expect such ‘reference duplicates’ to be common across different read groups. It may be these duplicates that are driving up All Pairs Affinity values for all read group pairs.
Figure 3.1. Visualization of pairwise similarities. Left: Heatmap of similarities. Read groups are plotted in rows and columns, sorted by expected library, such that each cell represents the similarity for a pair of groups. Darker cell shade indicates higher similarity. Color scale is provided adjacent to the matrix. Self-self pairs along the diagonal are set to zero. Colors are scaled separately per-panel. Right: Histogram of similarity values for known same-library and different-library pairs. Blue line: Probability density function (pdf) of the different-library beta distribution. Red line: pdf of the same-library beta distribution. (a) Exact Pairs Affinity distributions are well separated and the heatmap clearly indicates the expected library partition. (b) All Pairs Affinity distributions overlap significantly due to high different-library values. (c) Jaccard similarities are low for all pairs, but distributions are cleanly separated. (d) Cosine similarity distributions are poorly separated. Data are from the Illumina Content Exomes set prior to feature improvement.
This observation motivated the development of several metrics that employ the all-pairs concept but cap the maximum duplicate set size. In general, these measures behaved more akin to Exact Pairs Affinity, though some enrichment in different-library values could still be observed.

Cosine Similarity values for same- and different-library pairs also tended to be poorly separated, though it varied by data set whether the overall values were closer to zero or to one. The specific conditions that lead to high or low Cosine Similarity represent an interesting avenue for further exploration.

In contrast to All Pairs Affinity, the Jaccard Similarity measure typically displays a low ‘background’ of values for different-library pairs. However the values for same-library pairs are small relative to other measures.

3.2 Assessment of feature improvement

An effort was made to overcome some of the challenges observed in generating useful similarity values by improving the feature data. Section 2.3 details the two approaches used. First, various classes of problematic read pairs were filtered out. In particular, we aimed to minimize the impact of reads localized to the poorly-alignable genome regions referred to earlier. Second, optical duplicates were identified and removed. This is a special class of duplicates observed in Illumina sequencing data. They are generated downstream of library construction, during the actual sequencing process. Optical duplicates only occur within read groups. They shift the ratio of between-read-group duplicate pairs to within-read-group duplicate pairs towards the latter. We therefore expect their presence to depress similarity values for both same-library and different-library read group pairs.

Feature extraction was repeated for all data sets according to three conditions: First, without any feature improvement (“unfiltered”). Next, with problematic inserts filtered, but optical duplicates retained (“filtered”). Finally, with problematic inserts filtered and optical duplicates removed (“no-opticals”).
As an example to illustrate general trends, Figure 3.2 shows distributions for All Pairs Affinity values across these three conditions. The figure presents data from the Illumina Content Exomes data set. Other similarity measures and data sets gave similar results (data not shown). Broadly, the introduction of read pair filters leads to significant reduction in similarity values, particularly for different-library pairs. As a result, the distributions show reduced overlap, improving our ability to discriminate same-library from different-library pairs. Measures whose different-library similarities are already at or near zero (such as Exact Pairs Affinity) are not visibly affected by filtering. Removal of optical duplicates resulted in an upward shift in similarity values. Same-library values display a more pronounced shift, though both are affected. Visual overlap between the distributions remains small.

To achieve a more quantitative view of feature improvement, model selection was carried out as described in Section 2.10, based on evaluation criteria defined in Section 2.11. As noted above, we compared similarity-based likelihood ratios against a pre-trained threshold to produce a ‘same-library’ or ‘different-library’ decision for each pair of read groups.

Given a set of distribution parameters and a threshold, we can determine the percentage of decisions we expect to be wrong. To do so, we calculate the probability of drawing a value from the different-library distribution that is above the threshold, providing an expected rate of false same-library decisions (FSL rate). We likewise calculate the probability of drawing a value from the same-library distribution that is below the threshold, providing an expected rate of false different-library decisions (FDL rate). Details are provided in Section 2.11. Our goal for feature improvement is to achieve well-separated distributions that minimize both types of false call.

In order to more robustly assess the impact of feature improvement on our ability to model similarities, five distributions were trained for each data set over subsets the data, as part of 5-fold cross-validation. Expected FSL and FDL rates were calculated for
Figure 3.2. Qualitative effect of feature improvement. All Pairs Affinities were calculated from the Illumina Content Exomes data set. Similarity values are visualized as in Figure 3.1. (a) Values prior to feature improvement (from Figure 3.1b). (b) Values with problematic inserts filtered, but optical duplicates retained. (c) Values with problematic inserts filtered and optical duplicates removed. Insert filtering results in significant reduction of similarities and greatly increases the separation between the same-library and different-library populations. Removal of optical duplicates causes an upward shift in similarity values. Same-library values are shifted more than different-library values, though both are affected.
Figure 3.3. Quantitative effect of feature improvement. (a) Expected probability of false different-library decisions. (b) Expected probability of false same-library decisions. Data are displayed by dataset. Within a dataset, (left) is unfiltered data, (middle) is filtered data, (right) is filtered data with optical duplicates removed. Each boxplot is constructed from 5 values per similarity measure based on the models trained during cross-validation. All similarity measures are included. Individual points corresponding to the Exact Pairs Affinity models are overlaid in red. Feature improvement results in reduction of the expected rate of false same-library decisions for all datasets. Other results are discussed in the text.

Each distribution. (see Section 2.10; Bishop, 2015, p. 33; O’Neil & Schutt, 2014, p. 67).

Figure 3.3 shows changes to expected false decision rates when insert filtering and optical duplicate removal are applied. For the whole genomes data sets, filtering resulted in slight reductions in both the expected FSL and FDL rates. This affect was enhanced when optical duplicate removal was combined with filtering. Improvements to these data sets were important, as they displayed the highest expected error rates and were also the poorest-performing data sets in subsequent analyses (see Figures 3.4 and 3.5, and Table...
All remaining data sets likewise showed improvement in expected FSL rate after filtering. Expected FDL rates were either unaffected or reduced, with two notable exceptions. For the Agilent Exomes and Two Sense RNA data sets, increases were observed in the expected FDL rate. These were balanced by decreases in the FSL rate. This trade-off was deemed acceptable, especially given that expected error rates were lower for these data sets under all conditions, and they produced no false decisions during subsequent analyses.

The addition of optical duplicate removal to filtering led to further reductions in expected FSL rate for the Illumina Content and combined Exomes data sets, whereas the Agilent Exomes and RNA sequencing data sets appeared unaffected.

Figure 3.3 presents data aggregated across all similarity measures. Exact Pairs Affinity values are highlighted, as we observe in the following section that this measure gave the best overall performance. Exact Pairs results mostly tracked those of other data sets. For the Ice-only and combined Exomes data sets, we observed a slight increase in expected FSL rate after filtering, rather than a decrease. However, this was greatly outweighed by reduced FSL rate following optical duplicate removal.

Note that the extremely low magnitudes seen for some expected FSL and FDL rates cannot be taken as robust estimates of true false-decision rates in practice. These values were based on integrating the very small areas under the tails of beta distributions past a certain threshold similarity value. For the smallest FSL and FDL rates, this entailed integrating over a range of similarity values that were not represented by any actual observations from the training data.

Given that the affects of feature improvement were small in some cases, and mixed in others, some judgment must be exercised regarding whether to filter inserts and remove optical duplicates. The observed reduction in false same-library rates was consistent across data sets. The improvement to whole genome data sets in particular, whose
performance tended to be worse than either RNA or exomes, was deemed more important than the increased expected FDL rate for the Agilent Exomes and Two Sense RNA data sets.

With respect to optical duplicates, we note again that these are introduced during the sequencing process itself. As they occur after library creation, there is strong motivation to treat them as a source of potential bias.

In light of these results, the remaining analyses presented below include both filtering and optical duplicate removal. In the final implementation, both steps may be optionally disabled.

3.3 Similarity measure comparison

Having obtained high quality feature data, we compared our various similarity values in an effort to identify the best-performing measure. Whereas in the previous section we considered theoretical performance to assess the affect of data cleaning on our models, for the purpose of choosing a metric we examined empirical performance.

Taking the existing library annotations as ground truth, we can determine an empirical sensitivity and specificity for library identity decisions. We define sensitivity as the percentage of read group pairs correctly identified as same-library and specificity as the percentage of read group pairs correctly identified as different-library (see Section 2.9). In comparing models based on different similarity measures, 5-fold cross-validation was employed to improve the robustness of results. For each metric, on each data set, distributions and thresholds were trained on four-fifths of the data. Sensitivity and specificity were evaluated over the remaining one-fifth. This procedure was repeated a total of five times, and results were averaged. Cross-validated sensitivity and specificity results are depicted in Figure 3.4.

Exact Pairs Affinity gave the highest sensitivity and specificity when averaged across all data sets.
Figure 3.4. Similarity measure comparison. (a) Sensitivity and (b) specificity are displayed for all datasets by similarity measure. Both are defined as in Section 2.9. Exact Pairs Affinity yielded the highest mean sensitivity and specificity across data sets. Each point represents average performance from 5-fold cross-validation. Blue line displays the mean over all data sets for a given metric.
It was the most sensitive measure for all data sets except PCR-free, combined RNA, and the Combined data set. Sensitivity for the combined RNA data set remained high (94.3%). Only the Jaccard Similarity yielded a higher value (95.0%), and this measure performed worse on several other data sets.

The Combined data set included read groups from genomes, exomes, and RNA sequencing. Sensitivity was poor according to all similarity measures. Each “N”-limited Affinity was slightly more sensitive on this data set than Exact Pairs Affinity (maximum 63.9% versus 60.0% for Exact Pairs). Again, each of these measures performed worse than Exact Pairs for other data sets.

Interestingly, for PCR-free data, All Pairs Affinity was the most sensitive measure (100% sensitive versus 92.4% for Exact Pairs). PCR-free data might be expected to benefit from the inflation in similarity values observed with All Pairs Affinity. Given that our similarities are based on patterns of duplicate insert molecules, a lack of PCR-duplication should leave fewer such molecules to split across a library’s read groups. Indeed, it is more surprising that sufficient molecules are available to convey any same-library signal at all. This point is discussed in Section 4.1.3. It may be that there are fewer reads per duplicate set in PCR-free data, and that there is benefit in over-counting reads as pairwise combinations (see Section 2.6).

Exact Pairs Affinity yielded the highest specificity on all data sets except PCR-free, combined Genomes, and the Combined set. Specificity was still high for these datasets (98.3% for PCR-free, 99.2% for Genomes, 96.9% for Combined).

Based on these results, Exact Pairs Affinity was chosen as the similarity measure for further evaluation.

### 3.4 Performance evaluation

Having selected a distance metric, final distributions and thresholds were trained for each data set using all of the read groups. Figure 3.5 presents a combined view of sensitivity
Figure 3.5. Sensitivity vs. specificity for the Exact Pairs Affinity metric across all data sets. Several values are exactly 1.0 and are obscured beneath the ‘two_sense’ point. Dashed lines indicate 0.9.

and specificity for these final models. These results are also presented in Table 3.1.

Specificity was high, being >96.8% for all data sets. We observe >91.6% sensitivity for all data sets, with two exceptions. The first was the Combined set. This constitutes the superset of all read groups except those from PCR-free, which were not expected to perform well. As shown in Section 3.3, this data set received poor sensitivity from all similarity measures. There is apparently a major loss of ability to identify same-library pairs encountered when multiple data types are combined.

The other significant exception is the Genomes data set combining PCR-free and PCR-plus whole genomes. This displayed 74.1% sensitivity. This is similar to the cross-validation sensitivity during model selection in Section 3.3 (75.3%), which was the highest for all similarity measures. Specificity, by contrast, remained at 99.4% for the final Exact Pairs Affinity model. Evidently, mixing PCR-free and PCR-plus data significantly impedes the ability to detect same-library pairs.

As shown in Figure 3.6, PCR-free data yielded higher similarity values than did PCR-plus, particularly for different-library pairs. When PCR-free values were added to PCR-plus to form the combined Genomes set, the result was a positive shift in the different-library distribution and an elevated threshold. This led to more true same-library pairs being called different-library. It is interesting that PCR-free data yielded higher sim-
Figure 3.6. Exact Pairs Affinities for whole genome data sets. Similarity values are visualized as in Figure 3.1, with adjusted horizontal scale. Data sets shown are (a) PCR-plus, (b) PCR-free, (c) Genomes (combined). PCR-free read groups display higher Exact Pairs Affinity values than PCR-plus, for both same-library and different-library pairs. In the Genomes data, these two sub-populations may still be seen. The presence of elevated different-library values causes more same-library pairs to go undetected in this data set.

In general, failures were not distributed haphazardly across read groups. For PCR-free and PCR-plus data each, two specific read groups participate in most false decisions. For example, in Figure 3.6b, PCR-free read groups 5 and 29 display markedly lower same-library similarities than do other read groups. These read groups contribute the majority of false decisions for this data set. Further exploration of these particular read groups is taken up in the Discussion in Section 4.1.5.
For the Genomes data set, the presence of elevated different-library similarity values from PCR-free data skewed our model, such that many more PCR-plus read groups participated in false different-library decisions. In contrast to PCR-plus considered separately, many read groups in the Genomes data contributed only a small number of pairs to the total false decisions. These observations explain the reduced sensitivity associated with this data set. Given these results, we do not recommend that PCR-plus and PCR-free genomes be co-analyzed using this method.

Based on our observations for the Exome and RNA combined data sets, we do not see a substantial drop in performance when multiple library types are co-analyzed for these data type (at least for the library construction methods considered here). We therefore recommend that exome, RNA, PCR-free whole-genome, and PCR-plus whole-genome read groups be analyzed separately. We intend to include pre-trained distributions for each of these inputs in the final implementation. When run in the suggested manner, our pairwise method results in >91.6% sensitivity and >98.7% specificity across data sets (Table 3.1).

3.5 Unbiased clustering

In an effort to further improve performance, we investigated unbiased clustering as an alternate method for translating similarity values into library assignments. Our motivations were several. Firstly, by considering all read groups at once, the overall structure of similarity relationships within a dataset could contribute additional information beyond a single similarity value for each pair of read groups examined independently.

Secondly, considering each pair of read groups independently is not guaranteed to produce a consistent partition of the read groups into inferred libraries. For example, out of a set of three read groups $a, b, c$, the pairwise-thresholding method could call the pairs $(a, b)$ and $(a, c)$ same-library while calling the pair $(b, c)$ different-library. Clustering is attractive as it assigns each read group to a library cluster unambiguously, such that every
Figure 3.7. Clustering vs. pairwise library assignment. Cartoon representation of three read groups a, b, and c arranged in a two-dimensional feature space. One-dimensional distances between points represent $1 - \text{(some similarity metric)}$. The pairs (a,b) and (b,c) are 'closer' than the pairwise-threshold distance $T$, and are assigned same-library status by the pairwise method, as shown by the connecting edges. The pair (a,c) is 'further apart', and is assigned different-library status despite the relationship between the groups as a whole. However, all three read groups are captured in cluster C following some hypothetical clustering procedure.

Thirdly, our method based on log-likelihood ratios required us to pre-train distributions and thresholds on a trusted set of control data. It is possible such data may not be representative of every data set encountered in the future. For example, protocols may change or samples may be sequenced to very different coverages, potentially requiring our models to be refreshed. By contrast, our chosen clustering algorithm relies solely on the data within the test set.

The Louvain graph-based unbiased clustering algorithm was chosen for a variety of reasons discussed in Section 2.12. Two classes of inputs were considered. The first class was pairwise similarity values. For the second class, we clustered read groups based on their pairwise likelihood ratios. With this latter approach, we do not escape our dependence on pre-trained models. However, transforming raw similarities to likelihood ratios may render these values more informative, potentially increasing performance.

Figure 3.8 provides example clustering visualizations for both inputs. For anal-
ysis, each data set was treated as a fully-connected graph. The nodes of this graph comprised the read groups, while the edges were either pairwise similarity values or likelihood ratios. The Louvain algorithm partitioned this graph into clusters. Graphs were visualized by force-directed layout, with inferred clusters shown via node color. Both inputs resulted in good clusterings. Layouts displayed tighter point-clouds for likelihood ratio graphs. For data sets such as the Exomes set (Figure 3.8a), which combine multiple library types, sub-groupings are visible when the similarity graph is visualized. This positional information is not readily apparent for the likelihood ratio graph.

To facilitate a quantitative comparison of clustering methods against pairwise thresholding, we calculated the same pairwise sensitivity and specificity measures used previously. Figure 3.9 and Table 3.1 present these results.

Clustering similarity values offers significant improvement over the pairwise method (Figure 3.9a, middle). It was perfectly sensitive across all data sets, detecting all possible same-library pairs. It was also perfectly specific, or close to that, for most data sets (99.3% for the Agilent Exomes set and 99.8% for the combined Exomes set). Only the Combined data set showed lower specificity, and even for this troublesome data set clustering was 92.4% specific. As with the pairwise method, given the lessened performance we do not recommend co-analyzing exomes, genomes, and RNA sequencing data.

Clustering likelihood ratios produced mixed results. The method was more sensitive than the pairwise method for whole-genome data sets, but less sensitive for the Strand Agnostic and combined RNA data sets (Figure 3.9b, middle). Specificity remained over 90%, but was slightly worse than the pairwise thresholding method for the Genomes and Combined data sets, as well as PCR-free and combined Genomes.

The cases where the likelihood ratio clustering failed tended to involve outlier read groups which were more distant from their library-mates on average than were other read groups within the library. For these read groups the algorithm either erroneously separated them into a distinct cluster (as in the Strand-agnostic and combined RNA sets; Fig-
Figure 3.8. Visualization of unbiased clustering. Force-directed layouts of graphs based on similarity values (left) and likelihood ratios (right). Nodes are read groups, colored by their observed cluster from the Louvain community detection algorithm. Triangle nodes are those whose inferred cluster does not match their expected library. All graphs are fully-connected and edges are omitted for clarity. In general, likelihood ratio clusters appear tighter than those from similarity values. (a) Exomes (combined). The similarity graph shows two sub-populations corresponding to the different library construction methods. This is less clear in the likelihood ratio graph. For the similarity graph, the Louvain clustering fails to split the orange cluster into two distinct libraries. (b) Strand-agnostic RNA. For the likelihood ratio graph, the Louvain clustering incorrectly splits two read groups into a separate cluster. (c) PCR-free genomes. In both graphs, two points are somewhat distant from the two main groups. For the likelihood ratio graph, the Louvain clustering incorrectly assigned one of these points to the opposite library.
Figure 3.9. Comparison of library determination methods. (a) Three approaches to inferring library identity from similarity measures: pairwise log-likelihood ratio exceeds threshold (left), clustering of pairwise similarities (center), and clustering of likelihood ratios (right). Pairwise sensitivity and specificity are defined as in Figure 3.5. Performance for similarity clustering is almost perfect. (b) Direct comparison of sensitivity values between approaches. Similarity clustering is uniformly more sensitive than log-likelihood ratio thresholding (left) or likelihood ratio clustering (right). (c) Direct comparison of specificity values between approaches. (d) Data-set legend.
ure 3.8b) or assigning them to the neighboring library (as in the PCR-free and combined Genomes sets; Figure 3.8c). The single mis-assigned PCR-free read group was associated with markedly lower same-library similarity values than were other read groups from its library. This read group also performs poorly under the pairwise thresholding approach, where it participates in (23/23) possible false-different decisions (effectively splitting it off as a singleton). Possible reasons for the misclassification of this read group are considered in the Discussion. Similarity clustering handled all of these read groups appropriately.

In contrast, similarity value clustering failed on two small Agilent exomes libraries in the Agilent and combined Exomes sets (Figure 3.8a). These libraries were erroneously merged into a single cluster. Both Likelihood-ratio clustering and pairwise thresholding separated these libraries correctly. It is notable that this failure occurred when two of the libraries contained significantly fewer read groups than the remaining libraries.

Ultimately, more work will be needed to determine the requirements for successful clustering and better understand the situations in which clustering fails. This topic is taken up in Chapter 4. As such, the initial implementation of the Java command line program uses the pairwise approach. Given the evident performance improvements that are possible with similarity clustering, this work should proceed so that clustering may be incorporated into a future version of the final implementation.

3.6 Run time performance

While previous results indicate that our approach was highly successful at correctly inferring library identities, the cost in terms of total running time was high. The largest data set, the Combined, took over twenty-seven hours to process. Exomes and RNA data sets ran much faster, having many fewer reads. Yet run times were still on the order of hours.

The vast majority of this time was spent in feature extraction. Calculating simi-
Figure 3.10. Feature extraction time performance. (a) Total feature extraction time vs. number of reads. Points colored by data type. (b) Time sorting reads into duplicate sets vs. number of reads. Sorting consumes the majority of feature extraction time. Runs without optical duplicate removal are shown in magenta. Optical duplicate removal significantly slowed sorting. (c) Time iterating over duplicate sets vs. number of duplicate sets. Black line is least squares regression fit; $r^2 = 0.94$. (d) Ratio of sorting time with optical duplicate removal over sorting time without removal vs. number of reads. A y-value of “2” indicates that sorting was twice as slow with removal. Points are colored by value of the optical duplicate pixel distance parameter. The larger distance was associated with larger time increases.
larity values, cross-validated model fitting, evaluating distributions and clustering similarities totaled less than four minutes for the Combined data set. To facilitate model selection, feature information was cached to files following feature extraction. Figure 3.10 presents feature extraction run time information. Performance profiling and optimization were not undertaken as part of this study. To gain initial insight into time performance, wall-clock times for different components of the algorithm were taken from run logs. All feature extraction runs were executed single-threaded with 16G of memory on a compute-cluster server node running Linux.

Feature extraction consists of two distinct steps. In the first step, records are sorted into duplicate sets. This step consumes 65-89% of the total feature extraction time. Figure 3.10b displays the time required to perform the sort. This step makes use of Java’s `Arrays.sort` method, an implementation of the TimSort that is \( O(n \log n) \) (Oracle, 2016). Unfortunately, our method is not as fast as a full in-memory sort. We typically wish to sort more reads than may be held in main memory. Therefore, reads beyond a certain number (five hundred thousand by default) are written to temporary files after in-memory sorting. Reading and writing data to disk slows the algorithm. Multiple temporary files must also be polled intelligently when read to ensure correct sorting.

On high-memory systems, one could sacrifice additional memory for improved run time by raising the limit on reads allowed in main memory.

We may be able to improve run times by reducing the size of inputs to the method. Duplicate set sorting relies on mate pair information stored in each read. Each end of a read pair contributes redundant information regarding the distribution of inserts across read groups. Examining only one read from each mate pair could yield significant run time improvements with no compromise in quality.

Further optimization of the sorting step is more difficult. A reduced representation of the read is used in sorting, containing only the information necessary for duplicate identification. This data structure could possibly be further optimized to enable
more efficient comparisons, raise the number that can be held in memory, and decrease encode/decode time to disk.

The second step of feature extraction involves iterating over the duplicate sets to record read group patterning information, as described in Section 2.2. This step was linear according to the number of duplicate sets, and could take several hours (over six in our worst case). Unlike the sort, this step could be effectively parallelized. It is amenable to a map/reduce strategy, as the read group patterns of individual duplicate sets can be counted independently. Counts can be summed to form the table representation described in Section 2.2. A multi-threaded implementation based on Java parallel streams would provide CPU-level parallelism.

While filtering reads had no apparent impact on run times, optical duplicate removal did. As shown in Figures 3.10b and 3.10d, removing optical duplicates caused the time spent sorting reads to more than double for some data sets. Figure 3.10d shows the ratio of sorting times when optical duplicate removal was employed over sorting times when optical duplicate removal was not employed. There was not a clear association between this ratio and the total number of reads. However, some data sets required a higher value for the optical duplicate pixel distance parameter to facilitate optical duplicate detection on data from certain sequencing instruments. These data sets showed a markedly greater run time increase associated with optical duplicate removal, across a range of total read numbers. It is interesting that the response was less severe for the Genomes and Combined data sets, which had the highest numbers of reads. Further investigation may yield opportunities for optimization. Given the affect on total running time, users may wish to run feature extraction without optical duplicate removal. This feature can be optionally disabled.

As these observations highlight, though the analytical performance of our method was good, long run times remain a challenge which will require optimization.
Table 3.1.
Results summary. Percent sensitivity and percent specificity by method, for each data set. LR stands for “likelihood ratio”; LLR stands for “log-likelihood ratio”

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LLR Pairwise</td>
<td>LR Clustering</td>
</tr>
<tr>
<td>pcr_free</td>
<td>91.7</td>
<td>100.0</td>
</tr>
<tr>
<td>pcr_plus</td>
<td>95.3</td>
<td>100.0</td>
</tr>
<tr>
<td>genomes</td>
<td>74.1</td>
<td>100.0</td>
</tr>
<tr>
<td>combined</td>
<td>60.9</td>
<td>100.0</td>
</tr>
<tr>
<td>ice_only</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>agilent_only</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>exomes</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>two_sense</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>strand_agnostic</td>
<td>99.2</td>
<td>100.0</td>
</tr>
<tr>
<td>rna</td>
<td>96.8</td>
<td>100.0</td>
</tr>
</tbody>
</table>

3.7 Results summary

Table 3.1 presents a full summary of the final results discussed here. Given properly extracted input features and an appropriately selected similarity measure, the pairwise method shows >91% sensitivity and >98% specificity when used as recommended. Significant further performance improvements are possible through the application of a clustering approach. Our approach is therefore successfully able to recover library identity information from aligned sequencing data.
Chapter 4
Discussion

The method developed here is the first analytical tool able to detect mislabeled libraries in aligned sequence read groups. It therefore provides researchers with a novel capability when performing quality analysis.

Our results indicate that it is possible to achieve a sensitive and specific test for library identity based on patterns of insert duplication. Certain aspects of these results are considered in more detail below. Beyond the current study, a number of avenues remain available to better characterize this test and to improve its performance.

4.1 Results of the present study

Several of the results observed in Chapter 3 merit further consideration, due to their surprising nature or their implications for real-world applications.

4.1.1 Library and data type recommendations

Results from Section 3.4 indicated high analytical performance over individual data sets. Our approach also appears to generalize well when data from different library-construction techniques are co-analyzed. Results on the combined RNA and combined Exomes data were comparable with those given by their constituent data sets. Note that this observation applies to exomes targeting different portions of the reference genome. Our RNA and Exome models may therefore extend well to novel library types, but these should be evaluated individually. In contrast, PCR-free and PCR-plus genomes did not perform well when co-analyzed. We do not recommend simultaneous analysis of PCR-free and PCR-plus data.

We observed significant performance reductions when model fitting or cluster-
ing were conducted over a mixed set of exomes, RNA data, and whole-genomes. Data types spanning very different total genomic areas may produce differently scaled similarity values. This makes it hard to train a single threshold that successfully separates same-library from different-library pairs in all data types. We therefore do not recommend co-analyzing different data types.

4.1.2 Implications of run times
Perhaps the greatest obstacle to adopting this method as a routine quality check is the time required to run the program. Our training data sets contained more total reads than do typical research samples. Practical applications of this tool would likely require less time than the data sets analyzed here. Still, the tool required several hours to run for exome and RNA data and around a full day for whole-genome data.

Given these long run times, the tool may be most useful as an error-diagnostic rather than as a routine quality surveillance tool. It could be employed when a user already has concerns about their data, warranting investigation. Our method reports on a unique category of data mislabeling during lab processing, and could add important information to root cause analyses.

Opportunities for optimizing performance are discussed alongside results in Section 3.6. These efforts may yield significant time improvements. If further run time reductions are required, we may find that a smaller genomic territory such as one or more chromosomes could provide sufficient signal for the method. Reducing the input data in this way could save significant time spent sorting reads, provided the quality of the results remained acceptable.

4.1.3 PCR-free performance
One observation that ran counter to our expectations was the relatively good performance of our methods on PCR-free data. Without amplification, we would expect each insert
to be represented by a lone double-stranded molecule fragmented from the original sample. This should leave fewer duplicates reads to be split among read groups, reducing our ability to detect similarity.

Indeed, Figure 4.1 demonstrates that PCR-free data contains a lower proportion of duplicate sets of size two or greater relative to PCR-plus data. This is balanced by a higher proportion of single-read duplicate sets. It is interesting that there remain enough size-two duplicate sets in PCR-free data to provide usable Exact Pairs Affinity values. Yet our method was able to resolve library identities as observed in Chapter 3. One possibility is that the two reads in these duplicate sets represent the two strands of a single original double-stranded insert. We could examine the orientations of both set-members in an attempt to find support for this hypothesis.

With fewer size-two duplicate to contribute same-library signal, we would expect lower PCR-free similarity values than for PCR-plus. In this study, we observed the opposite (see Figure 3.6). What accounts for these higher similarities? In order to produce higher Exact Pairs Affinity values relative to PCR-plus, size-two duplicate sets from PCR-free data must contain a greater proportion of mixed-library reads, relative to same-

Figure 4.1. Relative duplicate set sizes. (a) Histogram of duplicate sets by size (up to size 5). Duplicate set counts are scaled relative to the total number of duplicate sets. PCR-free data contain a higher proportion of size-one duplicate sets and a lower proportion of size-two duplicate sets. (b) The same data in log scale. Duplicate sets of size greater than two are present in PCR-free data, though at slightly reduced fractions. Optical duplicates were removed prior to analysis.
library. That is, for two read groups $A$ and $B$, the ratio of $(A,B)$ duplicate sets relative to $(A,A)$ and $(B,B)$ must be higher for PCR-free than PCR-plus. As expected, 88.7% of size-two duplicate sets for PCR-free data were mixed-read-group, compared to 64.3% for PCR-plus.

We observed higher Exact Pairs Affinity for PCR-free in both same-library and different-library read group pairs. We therefore expect that a greater fraction of the size-two, mixed-read-group duplicate sets are somehow uninformative regarding library identity. What could be the source of such uninformative duplicate sets? One possibility is that our attempt to use mapping quality to exclude duplicates from poorly-resolved genomic locations was insufficient. At these locations, reads from repetitive or ambiguous sequences may be collapsed to a single aligned position, where they appear to be insert duplicates (Hogstrom, 2016). This could also account for the presence of PCR-free duplicate sets with size greater than two (Figure 4.1b), which is otherwise somewhat mysterious.

Ultimately, these explanations are speculative, and the performance of this method on PCR-free data remains an area for continued consideration.

4.1.4 Optical duplicate removal

Another somewhat surprising result was how little optical duplicate removal affected the separation of our distributions, as seen in Section 3.2. This class of duplicates is generated after library construction, during the actual sequencing run, and is therefore a source of noise. Optical duplicates are only detected within a particular read group. We expect that removing them would lower the number of same-read-group duplicate sets (types $(A,A)$ and $(B,B)$ from the example used above), thereby raising similarity values. This would apply to both same-library and different-library similarity values. This expectation was confirmed when similarity distributions were visualized, as in Figure 3.2.

It may be the case that the distributions of same-library and different-library simi-
larity values are shifted upward by a proportional amount, such that the amount of overlap between the two distributions is unaffected. This did not appear to be the case between Figures 3.2b and 3.2c. That said, given how well-separated the distributions are prior to optical duplicate removal, the affect of increased similarity values on distribution-overlap is difficult to inspect visually.

4.1.5 Exploration of false decisions

Though our results demonstrate that our approach successfully infers library relationships, we wish to better understand those cases in which it generates false decisions.

We have noted that false decisions tend to concentrate in certain read groups. Going forward, we should investigate fully whether any systemic factors cause some read groups to perform worse. Here, we have examined one particular set of read groups. In this instance, our method appears to detect a genuine unusual event during laboratory processing.

PCR-free was our lowest-performing library type, when analyzed separately. In Section 3.4, we noted two read groups from different libraries that each displayed markedly lower same-library similarity values relative to the rest of the set. These groups can be seen as outlier points in the clustering visualizations in Figure 3.8c, and as the white crosses at positions 5 and 29 in the heatmap from Figure 3.6b. These groups participated in the majority of the PCR-free pairwise thresholding false decisions, and one was incorrectly clustered by the likelihood ratio clustering approach.

The processing history of these read groups was noteworthy. Each of our PCR-free libraries was associated with three downstream denaturation events. These denaturations leaves the DNA single-stranded prior to sequencing. For each library, two of the denaturations were sequenced across multiple flowcells, yielding the majority of read groups in the data set. Our ability to identify libraries was not compromised by the presence of multiple denaturations or flowcells. In contrast, each problematic read group was
the sole result of a single denaturation. They were sequenced together on the same flow-cell and lane, apart from the other read groups in the set. This sequencing run was part of a development experiment, some of the details of which have been lost.

The problematic read groups still belong to the same library as unaffected read groups, according to our definition of a library. When clustered by similarity values, they were assigned the expected cluster. However, the observation that they were processed separately, with possible procedural differences, aligns with their apparent dissimilarity.

As noted in Chapter 1, the concept of a library does not have a single universal definition. Individual researchers must determine where in a chain of laboratory processing steps two divergent items should be given different library annotations. Ultimately, the purpose of reconstructing library annotations is to identify issues in laboratory processing. Individual laboratories may wish to differently calibrate the sensitivity of this method to laboratory perturbations. For some, these read groups should perhaps in truth have been given their own library IDs. The primary means by which researchers may tune the pairwise thresholding method is by supplying different data and truth labels during model fitting.

Even when library identification is successful, there are hints of substructure within our similarity relationships. See, for example, the heat-maps presented in Figures 3.1, 3.2, and 3.6. It would be interesting to ascertain the source of this apparent variability within same-library pairs. Perhaps they represent cryptic signals that read groups shared an Illumina flow cell or lane. Perhaps it would even be possible to capture these relationships at a finer-grained level than that of the library.

4.2 Future directions
While our efforts to detect signals of library identity from data were broadly successful, there are a number of potential improvements that could be made to further enhance performance.
4.2.1 Opportunities for improving clustering

Given the apparent performance benefits offered by clustering similarity values, further investigation should be conducted to establish this as a practical approach. The graph-based clustering algorithm employed here was selected primarily based on *a priori* expectations. Numerous other clustering algorithms exist, and a number of them could be compared directly to identify the most appropriate one.

If the Louvain algorithm should prove to be the best choice, there remain several possible enhancements to explore. The method accepts two parameters, which were left at their default settings in this study (Aynaud, 2011). The modularity optimization begins from a predefined initial partition. By default, this is the partition wherein each node belongs to its own community (Blondel et al., 2008). One could explore alternate initial partitions, including the partition formed by the read groups’ preexisting library annotations. There is also a ‘resolution’ parameter which affects the size of the final communities (Aynaud, 2011). This parameter was left at default but could be tuned.

Another possible extension of the Louvain algorithm is the multilevel-refinement enhancement proposed by Rotta & Noack (2011). This introduces a second modularity optimization step following each round of the Louvain iteration, and could improve clustering results.

A final potential improvement is spectral clustering via diffusion maps. Here, a non-linear dimensionality reduction step is introduced prior to clustering, in which similarity values are transformed into a ‘diffusion distance’ based on the eigenvectors of the Markov-normalized graph Laplacian (Nadler et al., 2008). The general aim is to map our read groups into a lower dimensional embedded space such that the structure within our data becomes more pronounced, enabling better clustering. The transformation itself is computationally inexpensive.

For any clustering algorithm, more insight must be gained into the specific condi-
tions that cause it to fail. In particular, we must characterize the limit at which we reach insufficient data. For the pairwise thresholding method, once the model has been trained, each unseen pair of read groups is considered independently. The method is applicable to even a single pair. Clustering, in contrast, would be unable to disambiguate whether a single pair of reads groups should be placed in the same cluster or different clusters. There is some lower limit to the number of read groups at which clustering performance becomes unacceptable. This limit may depend on other factors such as the number of true clusters in the data set, the relative sizes of true clusters, the presence of outlier points, etc. We must identify the relevant factors and determine the lower limit of data set size under different conditions.

One way to overcome the challenge of insufficient data may be to maintain a set of trusted read groups to cluster alongside the data of interest. These data would provide a benchmark of similarity values to guide the decisions made by the clustering algorithm. Given our results here, we would likely require separate mix-in sets for exomes, RNA, PCR-free, and PCR-plus genomes data at least.

When clustering, we must also consider how read groups are arranged. As shown in Figure 3.8a, our sole example of failure for similarity clustering occurred when the algorithm erroneously merged two libraries which were each much smaller than the other libraries in the data set. We should also systematically investigate performance when libraries of differing relative size are analyzed together. To do so, a larger data set could be decomposed into subsets, to ensure consistent data while different configurations are examined.

Our present results highlight the performance potential for a clustering-based approach and provide significant motivation to conduct the follow-up experiments described above. Ultimately, demonstrating the behavior of clustering over a wider variety of data sets will be necessary before this approach can be applied more broadly.
4.2.2 Opportunities for improving pairwise thresholding

As with clustering, various potential improvements to the pairwise thresholding algorithm could be explored. The threshold-determination formula described in Section 2.8 is conceptually simple, but other methods could be explored. For example, one could search the interval \([0, 1]\) to find the threshold value which maximizes some evaluation metric. We may ultimately wish to directly compare several threshold setting methods.

We could also compare additional distance metrics beyond those considered here. In doing so, we could continue our efforts to extend the theoretical basis of the Exact Pairs Affinity measure to better handle duplicate sets larger than size two. As noted in Section 2.6 and Chapter 3, the All Pairs combinatorial approach over-weights the information contributed by read groups from large duplicate sets. This led to the development of capped “N”-Pairs Affinity measures, as well as the Random Pairs Affinity. None of these measures performed consistently better in our evaluations than Exact Pairs.

As described in Section 2.5, Exact Pairs Affinity and related measures are based on a null model wherein two read groups were generated from a single library. Deviation from this model is handled via the introduction of a scale reduction factor \(\varepsilon\). One path toward a more intrinsically informative similarity metric is to develop a better conceptual model for the alternative hypothesis that two read groups were generated from different libraries. This would involve defining expected observations for our two feature vectors based on the physical processes that generate each read group. This situation is notably more complex than the null model, however. It remains to be seen whether a more helpful treatment can be arrived at.

When making pairwise decisions, we calculate a same-library likelihood and a different-library likelihood, and we compare them. It would aid interpretation to transform these likelihoods into true probabilities. Doing so would require assigning some prior probability to the same-library and different-library models. We do not have any
a priori expectation regarding whether an arbitrary pair of read groups should be same-library or different-library. The uniform prior $\beta(1, 1)$ may therefore be an appropriate choice. However, we might wish to minimize the risk of one type of false decision even at the expensive of the other type. For example, a false different-library decision may be more worrying to us, as this impedes our ability to mark duplicates across read groups. We may therefore wish to choose a prior that biases our posterior probabilities away from the more deleterious false decision.

4.2.3 General improvements

Finally, several experiments could be conducted that would enhance both the pairwise-threshold and clustering-based methods.

During feature extraction, additional read filters could be applied. Here, we excluded read pairs that failed vendor filters, were not aligned in a proper pair, or had low mapping quality. Secondary and supplementary alignments were also explicitly excluded. We considered the affect of all these filters simultaneously. It would be interesting to apply them individually and compare their affects.

In particular, we aimed to reduce the number of uninformative duplicates arising from poorly resolved portions of the reference genome, using low mapping quality as a proxy (Hogstrom, 2016). We speculate above that this filtering may be incomplete, and that these duplicates may be impacting PCR-free similarity values. We should directly examine alignments at known problematic regions before and after mapping-quality filtering. We should also attempt to directly exclude data from these regions by filtering reads aligning within pre-specified intervals. A base quality filter is unlikely to add significant value, as base qualities are not used directly to determine molecular identity.

Finally, an important question to address for both the pairwise and clustering approaches will be how they perform on low-quality data. This study used control data by necessity, and depth of sequencing coverage was fairly even within each sample type.
It will be informative to apply these methods to research samples displaying low coverage or other quality issues. We could simulate this to a degree. For example, we could analytically down-sample the reads within each read group to generate artificial reduced-coverage data, using the Picard software (The Broad Institute, 2009). This experiment should be carried out to demonstrate the method’s portability to reduced-coverage scenarios.

4.3 Summary

In summary, our results demonstrate the efficacy of an insert-duplication-based strategy for recovering library information from aligned sequence data. The outcome is a sensitive, specific implementation of a test for library identity based on pre-trained models of pairwise similarity measures. Future versions of this method could be improved by clustering read groups. This tool provides a novel capability to researchers seeking to verify the processing history of their data. It will also be of interest to sequencing centers looking to diagnose process errors and ensure reliable operations.


