

**Return on investment and library complexity
analysis for DNA sequencing**

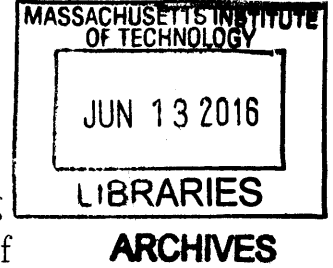
by
Larson J Hogstrom

Submitted to the Center for Computational Engineering
in partial fulfillment of the requirements for the degree of
Master of Science in Computation for Design and Optimization
at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

© Massachusetts Institute of Technology 2016. All rights reserved.



Signature redacted

Author

.....
Center for Computational Engineering
May 6, 2016

Signature redacted

Certified by..

.....
Yossi Farjoun
Computational Biologist, Broad Institute of MIT & Harvard
Thesis Supervisor

Signature redacted

Certified by..

.....
Bonnie Berger
Professor
Thesis Supervisor


Signature redacted

Certified by..

.....
Alan Edelman
Professor
Thesis Reader

Signature redacted

Accepted by


Nicolas Hadjiconstantinou
Professor
Co-Director, Computation for Design and Optimization

Return on investment and library complexity analysis for DNA sequencing

by

Larson J Hogstrom

Submitted to the Center for Computational Engineering
on May 6, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Computation for Design and Optimization

Abstract

Understanding the profiles of information acquisition during DNA sequencing experiments is critical to the design and implementation of large-scale studies in medical and population genetics. One known technical challenge and cost driver in next-generation sequencing data is the occurrence of non-independent observations that are created from sequencing artifacts and duplication events from polymerase chain reaction (PCR). The current study demonstrates improved return on investment (ROI) modeling strategies to better anticipate the impact of non-independent observations in multiple forms of next-generation sequencing data. Here, a physical modeling approach based on Pólya urn was evaluated using both multi-point estimation and duplicate set occupancy vectors. The results of this study can be used to reduce sequencing costs by improving aspects of experimental design including sample pooling strategies, top-up events, and termination of non-informative samples.

Thesis Supervisor: Yossi Farjoun

Title: Computational Biologist, Broad Institute of MIT & Harvard

Thesis Supervisor: Bonnie Berger

Title: Professor

Thesis Reader: Alan Edelman

Title: Professor

Co-Director, Computation for Design and Optimization: Nicolas Hadjiconstantinou

Title: Professor

Acknowledgments

This work was made possible through the guidance of my mentor, Yossi Farjoun, at the Broad Institute. His support and thoughtful feedback have been critical throughout the project. I also thank my wife Ju Yon Kim.

Contents

1	Introduction	13
1.1	Poisson-based models of PCR and library complexity	15
1.2	Motivations for an improved modeling strategy	16
2	Sequencer- and library-based sources of duplication	19
2.1	Complexity curves from read groups sets	19
2.2	Optical duplicate events	20
2.3	Multi-point parameterization and slope offset	23
2.4	Effect of optical duplicate pixel distance	24
2.5	Modeling insert features affecting duplication	26
3	Physical models of PCR duplication	31
3.1	Library sampling	31
3.1.1	Sampling without replacement from a library	31
3.1.2	Sampling with replacement from a library	32
3.2	Pólya urn modeling	33
3.3	PMF based on the expectation of binomial branching	37
3.4	Occupancy-based estimates	38
4	Parallel duplicate marking with Julia	41
5	Conclusion	47
A	Algorithms	51

List of Figures

1-1	Schematic of the library construction and sampling variables used often in this study. Each unique DNA insert is marked with a separate letter. The left side depicts a set of DNA inserts after DNA sheering. The right side depicts the library generated after sequence amplification.	15
2-1	Rates of optical (left) and non-optical (right) duplication as more read groups are added to a sequencing study. Each line represents one of 50 separate exome samples.	21
2-2	The graph on the left shows prediction for the library complexity curve using slope offset from equation 2.4 (red). Results are compared to observed data (black) as well as two Poisson-based models without slope offset. The graph on the right shows the least squares penalty for the parameterization of library size and slope offset.	23
2-3	Prediction for the library complexity curve using slope offset from equation 2.4 (red). Shown in the shaded regions are the relative contributions of the Poisson model, optical duplications, and slope offset to the final predictions.	24
2-4	Results for the slope-offset model where examined across many WGS samples.	25
2-5	The fraction of optical duplicates recorded (left) and the model error Poisson-based model.	26

2-6	Simulation of complexity curve prediction using the exponential mixture model Algorithm 2 (green) vs. global ROI prediction using the Li approach (blue).	29
2-7	Changes in duplication rate according to DNA insert length and GC content (left). The performance of the model for a single exome sample (right). In teal is the current Picard ROI approach. Plotted in blue and green are the cumulative Poisson approach and Poisson model according to GC-insert content.	29
3-1	Simulation showing the effect of various sampling methods (left) that assume sampling without replacement (hypergeometric distribution) sampling without replacement (binomial) and large sample size (exponential sampling). These three approaches are largely equivalent at the scale of library size and sampling observed in exome sequencing studies (right).	34
3-2	Error rates for the occupancy-based estimates are compared to multi-point Pólya estimates as well as Poisson-based models for exome samples. The graph on the right depicts results obtained from training the models on the three read groups. The left depicts a graph based only on one read group.	40
4-1	Overview of the processing and modeling strategy used by preseq. We identified read filtering using FLAG and read counting using POS as the bottleneck best suited for optimization and parallelization in Julia.	43
4-2	Schematic of the preseq.jl, a Julia wrapper that we implemented for complexity curve calculation from a BAM file input. Julia worker nodes perform the filtering and read count calculations in parallel, and the result is offloaded to the existing preseq.cpp code for the final complexity calculation.	44

4-3	Execution time of preseq.jl as a function of the number of parallel cores for three input BAM file sizes (upper). For comparison, the execution time of the serial preseq.cpp code is shown in parenthesis in the legend. (Breakdown of the execution time for the 135 Mb BAM file lower), demonstrating excellent performance improvement in read count calculation as the number of cores is increased, but significant overhead incurred from reading the BAM file to a plaintext SAM file in memory.	46
B-1	Simulation of read pairs drawn from a Poisson-based process (black). These results are plotted with the revised optical duplicate model (green), the current Picard ROI equation (red), and the consistent Picard equation (blue).	54
B-2	Results for a whole genome sequencing sample showing predictions with and without incorporating DNA insert length and GC content)	54
B-3	The fraction of optical duplicates recorded (left) and the model error Poisson-based model.	55
B-4	Changes in raw count, normalized count, and duplication rate according to DNA insert length (top). The bottom shows changes in the same count features according to GC content.	56
B-5	Observed counts for duplication sets from EstimateLibraryComplexity for representative WGS (left) and exome (right) samples. Observed results (red) are plotted with expected counts drawn from a zero-truncated Poisson distribution.	57
B-6	Simulation of Polya urn drawing events plotted with expectation from Equation 3.18	57
B-7	Simulation showing the expected distribution of counts after varying the library size (top) and number of Polya draws (bottom).	58
B-8	Result of binomial branching simulation compared to expectation as described in equation 3.18	59

B-9 Changes in observed occupancy as more read groups are added (red).
The blue curves depict expected occupancy. 59

Chapter 1

Introduction

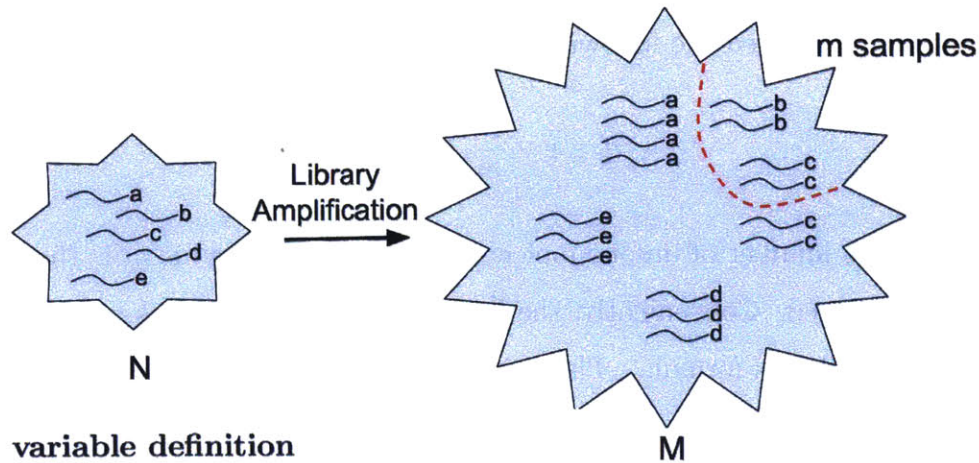
Strategies for high-throughput DNA sequencing were first developed in the 1990s and have continued to evolve, creating invaluable tools for population genetics, medical research, and experimental biology. Many of the most popular and cost-effective sequencing strategies available today have evolved from so-called "shotgun sequencing" technologies [1]. In such approaches, DNA is extracted from tissue or cell samples and then cleaved or sheered into small fragments called "inserts" or "templates" which are often only a few hundred base pairs. By reading these small segments of DNA, current sequencing technology enables the parallel reading of hundreds of millions of inserts. This strategy relies on algorithms for *de novo* genome assembly or mapping the small read segments to a reference genome.

Sequencing procedures for many technologies and sample types require DNA amplification steps before processing. Such amplification steps increase the total volume of DNA for reading either across the genome or, as in the case for exome sequencing, amplify the sequences of genomic regions of interest. The term *sequencing library* is commonly used to refer to a DNA sample which has been sheered, amplified and processed in preparation for sequencing. Polymerase chain reaction (PCR) is the most popular amplification procedure and aims to duplicate each DNA fragment many times during a series of heating and cooling reactions. The term *library complexity* is used in this thesis as a generic term to describe both the number of unique molecules that undergo an amplification protocol as well as the final concentration of

each molecule after library construction.

In practice, the true complexity of a library can be difficult to anticipate *a priori* due to variations in sample quality, PCR efficiency, and the physical properties of DNA inserts that affect the duplication rate. The library complexity of a given sample can have a substantial influence on the trajectory of a sequencing experiment. In a whole genome sequencing (WGS) experiment, for example, a DNA library would ideally contain many unique and overlapping DNA segments distributed evenly across the genome. Often, the genomic coverage (the average number of bases aligned to a given region) is quite variable for a given sample. Variations in genomic coverage are largely due to random sampling along regions of the starting DNA. Additionally, the amplified properties of a sequencing library can further impact observed coverage. In the case where a small number of DNA fragments have been duplicated at a very high rate, these can 'obscure' the occurrences of less-amplified fragments. Similarly, starting with low-quality or formaldehyde modified tissue may yield a smaller number of starting fragment before amplification.

Library complexity affects the rate of information acquisition during a sequencing study. During the earliest phase of sequencing a particular library, nearly every observed DNA insert is unique. At a later stage of the study, when more inserts have been read by the machine, new observations will have a higher probability of being repeats. For this reason, the overall fraction of duplicates increases as more DNA fragments are read by the sequencer. Duplicates are typically identified and excluded from variant calling and down-stream analysis because they represent non-independent observations of the underlying genomic information being recorded. The motivation for such exclusion is twofold. First, an error in PCR amplification might cause the appearance of a variant in a large number of observed reads even though this change in DNA is not present in the sample of interest. Second, amplification biases may skew estimates of the copy number or underlying genotype of a given sample.



N - number of unique seeds in the starting library
 k_i - number of the i th seed in the library
 M - total number of molecules in the amplified library. sum of all k_i
 m - number of samples observed from sequencing library

Figure 1-1: Schematic of the library construction and sampling variables used often in this study. Each unique DNA insert is marked with a separate letter. The left side depicts a set of DNA inserts after DNA sheering. The right side depicts the library generated after sequence amplification.

1.1 Poisson-based models of PCR and library complexity

Figure 1-1 is a pictorial representation of some of the library properties modeled in the current study. In this text, the total number of distinct DNA inserts after DNA sheering is N . The term "molecule" and "DNA insert" are use interchangeably. In PCR, these unique starting molecules undergo a series of amplification rounds to generate a library with M total molecules. In this larger library, the counts k_i can be expressed for each molecule i where $(i = 1, \dots, N)$. The distribution of k_i 's is unknown. The experimenter only observes a subset of DNA molecules in a sample of size m which have been recorded by the DNA sequencer.

A *library complexity curve* describes the change in the unique observed molecules to the total observed molecules throughout a sequencing experiment. In 2010, Li described a model to predict the library complexity curve for a given sample using a simple set of closed form equations [5]. This modeling strategy for sequence du-

plication assumes that counts of duplication events are drawn from a fixed Poisson distribution. Li showed the following estimation of duplication

$$d \simeq 1 - \frac{N}{m}(1 - e^{-m/N})$$

where N is the number of unique molecules in the library and m is the number of molecules observed. Conveniently, the Poisson parameter λ drops out of the during the derivation of this formula. This allows for estimation based only on the library total number of molecules observed at a given in time and the fraction of those molecules that are unique. Here the assumption that $m \rightarrow \infty$ is required as Li employs the common definition of exponential when modeling library sampling.

1.2 Motivations for an improved modeling strategy

This project is motivated by a desire to improve library complexity estimates for the Genome Analysis Tool Kit (GATK) best practices. The Genome Analysis Tool Kit is a suite of tools used for variant discovery in high-high throughput DNA sequencing. A companion to GATK is the Picard suite of pre-processing tools. Picard is designed to perform many supporting functions for variant analysis including manipulation of sequencing filetypes, reporting quality metrics, and various sorting procedures. Included in Picard is the "MarkDuplicates" tool which iterates through single- or paired-end sequencing data to mark duplicate reads according to the Sequence Alignment-Map (SAM) format specifications. The MarkDuplicates tool tracks different forms of read duplication and calculates a return-on-investment (ROI) metric to predict library complexity. This returned ROI estimate is expressed as a multiple of additional unique molecules that would be observed if same sample was sequenced further. The current study aims improve the ROI modeling strategy by improving the handling of sequencer-based duplicates and by improving the physical model of the PCR process.

A further goal of the current study is to explore approaches to physical modeling in the prediction of library complexity. Recent studies [2] [3] have demonstrated

predictive approaches based on extrapolating the shape of complexity curve at an early phase of a sequencing experiment. These approaches do not fully account for the physical processes through which duplication events arise. Recent discussions in statistical literature have drawn distinction between predictive and explanatory approaches in describing experimental phenomena [4]. Explanatory models are advantageous because they produce predictions by using components with an explicit physical interpretation. This approach has two main advantages. The first is that explanatory approach can help us to reason about the likely performance of predictions outside of the scale and regime of the data being used to parameterize the model. Second, we can have a better intuition about the performance or reliability of the predictions as sequencing technology changes (e.g. changes to sequencing approach, library preparation methods, hybrid selection, etc).

It should be stated that different forms of experimental data are often required to fully vet an explanatory model describing a physical processes. In DNA sequencing, examples of data sets that might build confidence in a physical model of library complexity could include: 1) sequencing samples with the library size known *a priori*, 2) samples with the same DNA extract, but various rounds of PCR sequencing 3) libraries without biological duplication events, or 4) sequencing of the same library with and without technologies impacted by optical duplication events. Such experiments are beyond the scope of the current study, but could be performed in future to bolster evidence for physical models explored here.

Interestingly, such controlled experiments are not always necessary for a predictive approach. As discussed by Shmueli, "Whereas for causal explanation experimental data are greatly preferred, subject to availability and resource constraints, in prediction sometimes observational data are preferable to 'overly clean' experimental data, if they better represent the realistic context of prediction in terms of the uncontrolled factors, the noise, the measured response, etc." [4]. This leads to interesting tradeoffs between physical and predictive modeling. In what circumstances, for instance, is it worth sacrificing some predictive power in order to utilize an explanatory model which is more physically interpretable and better generalizes to future sequencing

technology? This and related ideas are explored qualitatively in this text.

Chapter 2

Sequencer- and library-based sources of duplication

2.1 Complexity curves from read groups sets

In order to evaluate the complexity curve for a given input, sequence files were segmented and grouped into a series of files based on read group. Current Illumina sequencing technology relies on flow cell chips which contain billions of nano wells, each of which are designed to bind and amplify sequence from a single DNA template molecule. Flow cells are divided into 8 equally spaced lanes and data collected on a separate lane is called a read group. All data was read according to the Sequence Alignment/Map Format Specification (SAM and BAM files). Individual read groups were randomly selected and placed into an expanding set. During each addition of a random read group, a new file was generated using the updated set of read groups. For a starting input with n read groups, this resulted in n separate BAM files of increasing size. Read groups were then used as the basis to analyze complexity curves and return-on-investment calculations. This was done to minimize bias related to read location and pair orientation that can arise when randomly sampling records in BAM file. Because read groups occur on different lanes or flow cells of a sequencing study, they are taken here to represent a more independent measurements of a sequencing library.

Data:

n samples BAMs files

for $sample = 1$ to n **do**

 extract m read groups names from $sample$ BAM

for $j = 1$ to m **do**

samtools view -b -h $sample.bam$ -R $jth-read-group$ -h $chr21.bed$ -o
 $jth-read-group.bam$

$rgSet = \{1, \dots, j\}$

MarkDuplicates I=1st-read-group.bam, ... I= $jth-read-group.bam$

 OUTPUT= $rgSet.bam$ OPT_DUP_PIXEL_DISTANCE=2500

end

 remove 1st-read-group.bam, ... $jth-read-group.bam$

end

Output:

m read group set BAMs for each $samples$ BAM file

Algorithm 1: Read group set generation and Mark Duplicates

Once files were generated for the increasing read group sets, MarkDuplicates was run on each set file. The parameter for defining location-based duplication events was set to 2500 pixels which is the default parameter commonly used for the Illumina HiSeq X Ten System which was used to generate data analyzed in this study. Details regarding this distance parameter are discussed further in Section 2.4. Counts reported in the MarkDuplicates metrics file were used as the basis for complexity curves. These metrics included the number of read pairs observed for the read group set, the number of unique read pairs, and the count of optical duplicates. Algorithm 1 was proposed to organize and pre-processes read groups into a series of increasing sets in order to evaluate ROI methods.

2.2 Optical duplicate events

Separate sources of duplication may differentially impact a sample depending on the stage of a sequencing study. For this reason, the current study handles sequencer-based duplication, referred to in this text as 'optical duplication', separately from PCR duplication when modeling complexity curves. The term optical duplication comes from instances when a single cluster of molecules of the same DNA insert is counted two or more times at the same location on a flow cell. Optical duplication can

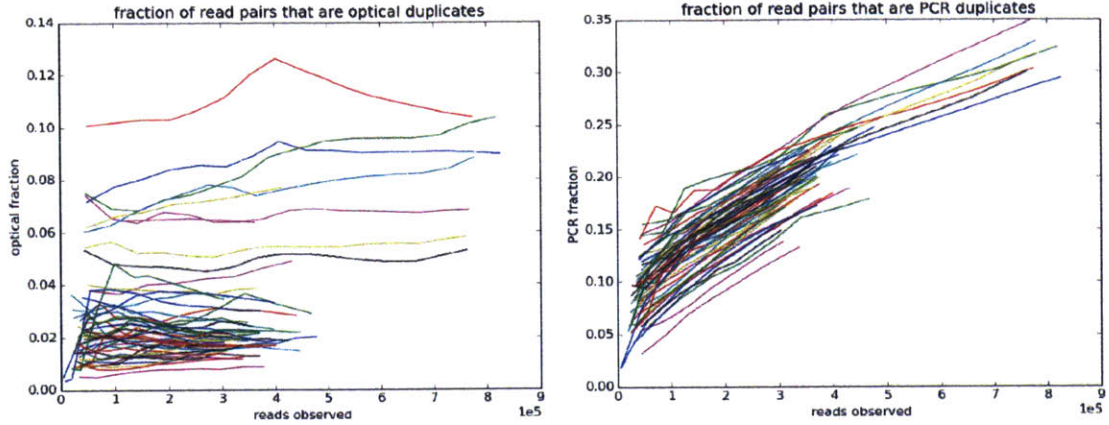


Figure 2-1: Rates of optical (left) and non-optical (right) duplication as more read groups are added to a sequencing study. Each line represents one of 50 separate exome samples.

also refer to ‘pad-hopping’ events where a single DNA-insert is copied and migrates to multiple adjacent wells on a patterned flow cell. Here, optical duplication events are identified as duplicates that occur within close proximity (as defined by a fixed pixel distance in the x and y direction) of the same sequencing tile. Figure 2.2 shows the rate of duplication for optical vs. non-optical duplicates as a sequencing study proceeds.

As the rate of optical duplication is largely constant throughout an experiment, they are modeled separately from estimates of PCR-based duplication where u_p is the fraction of unique molecules measured under the Poisson-based model. From Li 2010, the library size N_p under the Poisson model can be estimated as follows:

$$\frac{c}{N_p} = 1 - e^{-m_p/N_p} \quad (2.1)$$

Here, the optical duplicates d_o are subtracted from the total observed read counts $m_p = m_0 - d_o$ and c is the count of unique read pairs observed. Using the library size estimate, the fraction of unique molecules u_p can be calculated under the Poisson model for any sequencing multiple x :

$$u_p(x) \approx \frac{N_p}{xm_p} (1 - e^{-m_p x/N_p}) \quad (2.2)$$

The final fraction of unique read pairs u can then be expressed in terms of the the Poisson model above multiplied by a linear term representing the fraction of molecules that are non-optical duplicates:

$$u_t(x) = \left(\frac{m_0 - d_o}{m_0} \right) u_p(x) \quad (2.3)$$

Where m_0 and d_o are the total read count and optical duplication count at the extrapolation point respectively. This strategy is compared to the optical duplication handling that currently takes place in the Picard ROI estimation tool. This program subtracts off optical duplicates from the total reads observed when calculating the library size, but leaves in optical duplication events when applying the ROI equation. The result is that the Picard ROI tool systematically over estimates the fraction of unique molecules that are expected under the Poisson-based model. This is because subtracting off optical duplicates results in a larger estimated library size, but then uniqueness term expressed by $(1 - e^{-\frac{xm}{N_p}})$ is not lowered when optical duplicates contribute to the total count m that is modeled. Furthermore, the prediction is not valid at the extrapolation point. Figure B-1 depicts a simulation comparing the current Picard approach to that listed in equation 2.3. In this simulation, a complexity curve was generated by sampling items drawn from distribution of counts populated by Poisson(λ) (to simulate PCR duplication) with an additional linear source of repeats (to simulate optical duplication). Modeling the Poisson distribution without accounting for the linear component resulted in an overestimation of duplicates while the Picard ROI approach resulted in a systematic underestimation of duplications. Equation 2.3 best captured the complexity curve in the simulation. The Picard ROI method used as a comparison approach multiple times in this text and is often labeled "Picard_ROI_current".

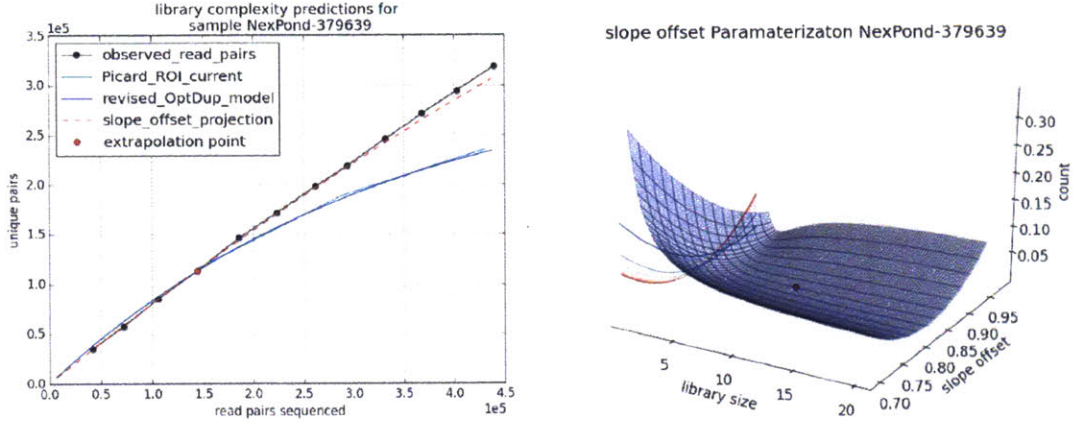


Figure 2-2: The graph on the left shows prediction for the library complexity curve using slope offset from equation 2.4 (red). Results are compared to observed data (black) as well as two Poisson-based models without slope offset. The graph on the right shows the least squares penalty for the parameterization of library size and slope offset.

2.3 Multi-point parameterization and slope offset

Alone, the Poisson-based model imposes that the first portion of a complexity curve will have a slope close to one. This does not allow for scenarios where the complexity curve is largely linear but offset from $y=x$ axis. This has been observed in many exome and WGS samples covered at depths common in production data at the Broad Institute. Such a scenario could be indicative of an unknown source of optical or sequencer-based duplication. To explore this possibility further, the Poisson model was modified to allow for a free linear term. A least squares objective was used to minimize the difference between the observed unique fraction and the uniqueness rate estimated from equation 2.2 modified with a new slope offset parameter s . The sum of the squared difference was calculated for read group sets 1 through M which make up the extrapolation points:

$$\sum_{i=1}^M \left(\frac{c_i}{m_i} - \frac{sN}{m_i} (1 - e^{-m_i/N}) \right)^2 \tag{2.4}$$

Figure 2-2 shows and prediction results for a single nexome sample using four out of twelve read group sets for extrapolation.

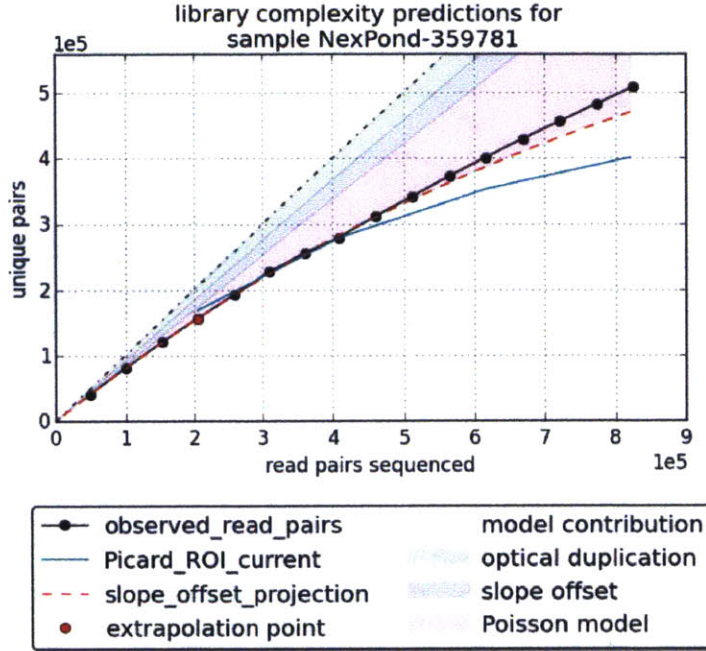


Figure 2-3: Prediction for the library complexity curve using slope offset from equation 2.4 (red). Shown in the shaded regions are the relative contributions of the Poisson model, optical duplications, and slope offset to the final predictions.

It is helpful to examine the relative contributions of the Poisson model, optical duplications, and slope offset to the final predictions. This can be described by the following equation for the total fraction of unique molecules u_t given x sequencing multiples:

$$u_t(x) = s \left(\frac{m_0 - d_o}{m_0} \right) u_p(x) \quad (2.5)$$

Where m_0 and d_o are the total read count and optical duplication count at the extrapolation point respectively. The relative contribution of the three model components can be visualized in Figure 2-4

2.4 Effect of optical duplicate pixel distance

To examine the effect of sequencer-based duplications, the distance threshold defining optical duplication events was manipulated. After grouping read pairs into duplicate sets, MarkDuplicates uses read names retrieve the coordinate location of clusters

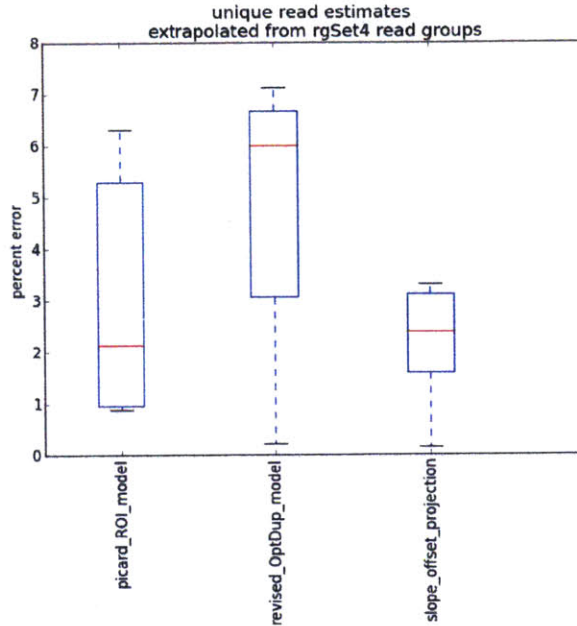


Figure 2-4: Results for the slope-offset model where examined across many WGS samples.

observed by the sequencer. If both the x and y distances are within the OPTICAL DUPLICATE PIXEL DISTANCE for any given pair in the set, a read is marked as an optical duplicate. This parameter, 'OPTICAL DUPLICATE PIXEL DISTANCE', was varied across a number of pixel distances. Figure 2-5 shows that the fraction of optical duplicates to total pairs observed increases as the parameter is increased in MarkDuplicates. The lowest distance threshold tested effectively only marks two read pairs as optical duplicates if are found in adjacent wells on a patterned flow cell. At the largest value tested, an arbitrarily high pixel distance, the threshold effectively represents the scenario where any two read pair duplicates are marked as optical if they appear on the same tile segment of the flow cell. A variety of values were tested in-between these two extremes. Currently, the default value for this parameter in the production of study-quality data is 2500 pixels for data collected on patterned flow cells.

Figure 2-5 shows the fraction of optical duplicates tails off around 4489 pixels. The right hand figure shows a reduction in error from the model Poisson-based model from equation 2.3. Although there appears to be a predictive benefit from setting

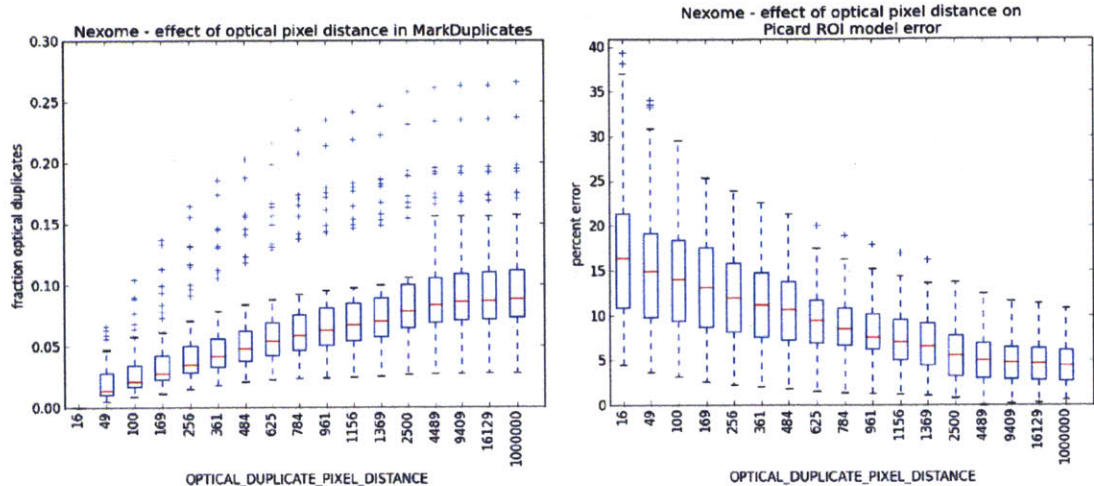


Figure 2-5: The fraction of optical duplicates recorded (left) and the model error Poisson-based model.

this parameter to the max, further analysis would be needed to assess the impact of false positive marking of optical duplicates. Interestingly, a more frequent marking of optical duplication on a tile reduces the contribution of the slope-offset parameter s in equation 2.4. Figure B-3 shows that as the pixel distance increases, the slope-offset moves closer to 1 in exome samples, suggesting that the impact of linear offset is largely related to optical duplication. The fact that the linear offset term does not reach 1, even at the highest distance threshold, could be taken as evidence for missing optical duplication.

2.5 Modeling insert features affecting duplication

One approach taken in this study was to incorporate features of DNA inserts when calculating duplication projections. In the following analysis, two features of genomic library inserts were chosen for modeling: the insert length and guanine-cytosine (GC) content of reads. It was hypothesized that these two features would be especially informative as they can influence PCR reaction efficiency. Both the length of a DNA insert as well as its GC content can effect melting temperature during the heating cooling cycles of PCR, leading to differences in molecular concentration in the resulting library. The empirical duplication count was examined for each feature

as well as in combination.

Reads were counted and binned according to their range of insert features. For the insert length and GC analysis, this resulted in a matrix $\mathbf{M} \in \mathbb{Z}^{m \times n}$ where m is the number of length bins and n is the number of evenly spaced GC bins. In the analysis shown, $m=50$ and $n=30$. The inferred insert size was based on location of paired end reads that were mapped to a reference genome using BWA alignment¹. The GC content of the insert was inferred using the reference sequence. First a cumulative GC count was calculated for each contig. An array was initialized matching the length of contig and each entry contained the cumulative count of G and C bases up to its corresponding index in the reference. For each read pair that mapped to the contig of interest, the number of GC bases in the insert was calculated with the cumulative GC array. The entry in the cumulative GC array corresponding to the starting mapped position of the read pair was subtracted from entry corresponding to the terminal mapped position of the read pair. This gave an estimated count of G and C bases in the DNA insert and this was normalized by the insert length to estimate the percent GC for each DNA insert. This calculation of insert GC relies on the assumption that the occurrence of single-nucleotide-polymorphism (SNP) or insertion-deletion events does not have a large impact on the overall GC content of a DNA insert. Unmapped reads were excluded from the analysis. Figure B-4 shows the observed differences in duplication rate that vary with GC content and insert size. Higher rates of duplication were observed with DNA inserts of length 200 b.p. and 10% GC. DNA inserts with these features, however, represent a small fraction of total read pairs examined.

¹In this approach the best alignment of a read is used.

Data:
Unique molecule counts: $\mathbf{C} \in \mathbb{Z}^{m \times n}$, m Length bins, n GC bins
Total molecule counts: $\mathbf{M} \in \mathbb{Z}^{m \times n}$, m Length bins, n GC bins
Optical duplication rate: d_r
initialization;
 $\mathbf{L} \leftarrow (m \times n)$ matrix to contain library size estimates
 $\mathbf{M}_p \leftarrow (m \times n)$ matrix $\mathbf{M} - d_r \mathbf{M}$
 $\mathbf{U} \leftarrow (k \times m \times n)$ matrix to contain unique molecules modeled
for $i = 1$ **to** m **do**
 for $j = 1$ **to** n **do**
 $c = \mathbf{C}[i,j]$
 $m_p = \mathbf{M}_p[i,j]$
 $\mathbf{L}[i,j] = \text{Solve for } N_p \text{ in } \frac{c}{N_p} = 1 - e^{-m_p/N_p}$
 for $x = 1$ **to** k **do**
 $\mathbf{U}[x,i,j] = \frac{N_p}{x m_p} (1 - e^{-m_p/N_p})$
 end
 end
end

Algorithm 2: ROI exponential mixture model

A mixture of exponential models was employed to incorporate the features extracted from the DNA inserts. For each entry in the insert length-GC count matrix, a separate exponential model was generated. Library size was estimated using the number of unique and total molecules in a given length-GC bin. A projection of unique molecules was calculated using Li’s method. The projections across all length-GC entries were combined proportional to the count of read pairs observed in a given bin. Additionally, the matrix of projections was masked for bins that contained less than 1% of the read pairs observed. For entries with counts below this threshold, the estimated fraction of unique molecules was set to the global estimate calculated with all read pairs combined. Pseudocode for the algorithm with the masking procedure procedure is listed in Appendix A.

This approach was first evaluated with a simulation. A matrix $P \in \mathbb{R}^{m \times n}$ was populated with positive, real values representing λ parameters for multiple different Poisson processes. Two additional matrices, M and C , were initialized to represent the total counts and unique counts respectively where each entry represents the m and c from equation 2.3. Matrix M was populated with random integers, but half

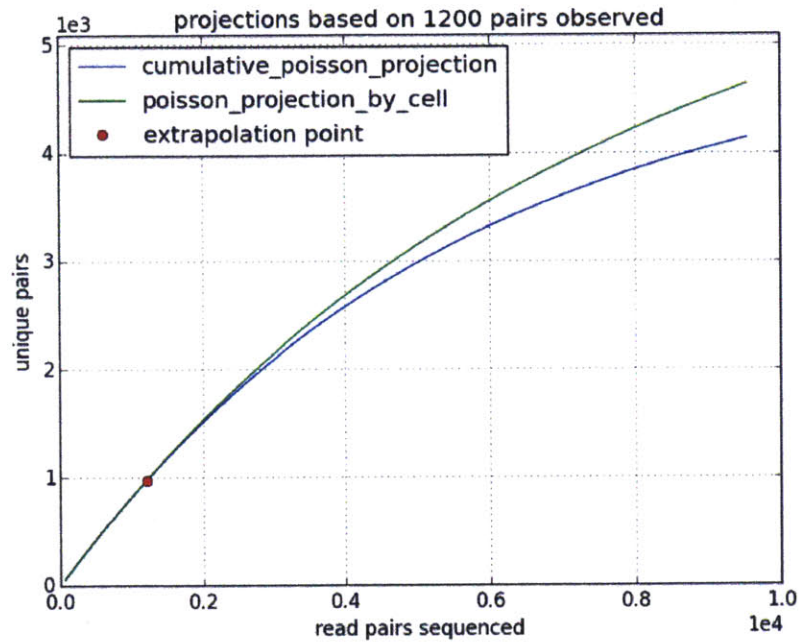


Figure 2-6: Simulation of complexity curve prediction using the exponential mixture model Algorithm 2 (green) vs. global ROI prediction using the Li approach (blue).

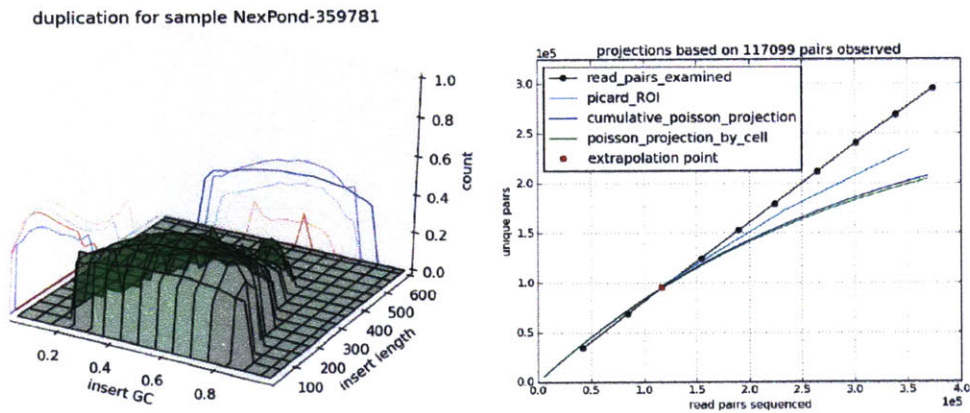


Figure 2-7: Changes in duplication rate according to DNA insert length and GC content (left). The performance of the model for a single exome sample (right). In teal is the current Picard ROI approach. Plotted in blue and green are the cumulative Poisson approach and Poisson model according to GC-insert content.

of the λ entries in P were inflated, creating a contrasts in the fraction of unique counts observed in C . For each entry of M , the library size was estimated along with the fraction of unique molecules using Algorithm 2. The ROI predictions from the cumulative counts (using the sum all entries of the M and C matrices) were compared to the estimates obtained by mixing exponential models entry wise as shown in Figure 2-6. The results of the simulation demonstrate how a model that handles reads with specific features and different duplication rates can give different results relative to an aggregated prediction with the same model.

In observed data, this approach showed little benefit in terms of explanatory power. Figure 2-7 shows results from the exponential mixture approach of Algorithm 2 applied to exome data. The results for all samples examined were nearly identical to the predictions obtained using the simple Poisson-model based on aggregated counts from all reads. It was concluded that these features do not have strong predictive power for return on investment calculations.

Chapter 3

Physical models of PCR duplication

3.1 Library sampling

Physical models of library complexity begin by first accounting for the sampling process that occurs in a sequencing study. During the processing of a sample in preparation for sequencing, DNA inserts are often size selected and then amplified as part of the library construction. To the experimenter, both the total number of DNA inserts and their full distribution of counts in the library is unknown. The experimenter only observes a sample of size m from the larger library with M total molecules. Figure 1-1 depicts how observed reads are only a subset of unique information from the total library. This section outlines three approaches to capturing this sampling process. The first is a sampling without replacement with a hypergeometric distribution. The second is the standard binomial approach to sampling without replacement. The third is another approach used in the Li method which makes an additional assumption around relative size of m and M .

3.1.1 Sampling without replacement from a library

One would expect the highest fidelity from an approach that models sampling without replacement. This is because a DNA insert that has been observed on the sequencer does not return back to the library for further recording. Given a single molecule with

k occurrences in a library of size M , the probability of c counts of the same molecule from m samples can be described by the hypergeometric distribution [3]

$$P_{k,c}^{(M,m)} = \binom{k}{c} \binom{M-k}{m-c} / \binom{M}{m} \quad (3.1)$$

This can be used in estimating the occurrence of a given seed during sampling. If c_i is the count of the i th seed from the library and observed in m samples, we can estimate the probability that a given molecule is observed at least once or $P(c_i \geq 1)$.

If k_i is known

$$P(c_i \geq 1) = P_{k_i, c_i \geq 1}^{(M,m)} = (1 - P_{k_i, c_i = 0}^{(M,m)}) \quad (3.2)$$

If k_i is not known, then this can be approximated by using a generic probability mass function (PMF)

$$P(c_i \geq 1) = \sum_{k=1}^{\infty} p_k (1 - P_{k, c_i = 0}^{(M,m)}) \quad (3.3)$$

Where p_k is the probability of a unique molecule having count k in the larger library. Equation 3.3 is only an approximation because as the sampling proceeds, the sampled values are not independent. This can yield nonzero covariance between sample values. The influence of this is minimized in scenarios where $m \ll M$.

3.1.2 Sampling with replacement from a library

A binomial distribution is often used to model sampling with replacement. Although the process of sequencing does not involve replacement, for large populations of DNA inserts, the covariance between any two draws from the library is approximately zero. The handling of binomial sampling is common and can follow analogous logic to equations 3.2 through 3.3. Alternatively, from Li [5], we can model sampling with replacement by evaluating the occurrence of a unique seed i in m samples from library of size M where

$$X_i = \begin{cases} 1, & \text{if seed } i \text{ occurs at least once} \\ 0, & \text{otherwise} \end{cases}$$

if we assume $m \ll M$ and the procedure follows sampling with replacement then

$$P(X_i = 1) = 1 - \left(1 - \frac{k_i}{M}\right)^m \simeq (1 - e^{-\frac{k_i m}{M}}) \quad (3.4)$$

This relies on the definition of the exponential and thus assumes that $m \rightarrow \infty$. From this it follows that the duplication rate is

$$d = 1 - \frac{\sum_{i=1}^N X_i}{m} \quad (3.5)$$

$$d \simeq 1 - \frac{N}{m} \sum_{k=0}^{\infty} (1 - e^{-\frac{km}{M}}) p_k \quad (3.6)$$

where p_k is the probability of count k in the sequence library before sampling.

The three sampling approaches (hypergeometric, binomial, and exponential) were simulated across values $k = 1 \dots 30$. Figure 3-1 (left) depicts a scenario where with a sample size of $m = 10^4$ and library size $N = 10^4$, the exponential sampling estimation separates, yielding different expected occupancies. In most sequencing experiments, however the sample and library size are much larger. Figure 3-1 (right) depicts the scenario where the sample and library size are larger than $m = 10^6$ and $N = 10^7$ respectively. For the sample sizes seen in typical high-throughput sequencing studies, the three approaches are concluded to be largely equivalent.

3.2 Pólya urn modeling

Although it is analytically convenient to assume that the count of molecules in the sequencing library is Poisson distributed, the author of the present study has not yet encountered direct evidence for this in the literature. Figure B-5 shows that there is weak evidence for the samples examined in the current study. Here, an effort is made to explore alternative models and distributions starting by first examining branching events that occur during PCR.

Given a set of starting seeds $i = 1 \dots N$, we can express the count of each unique seed A_i^j after j rounds of duplication. If every seed was duplicated once during each

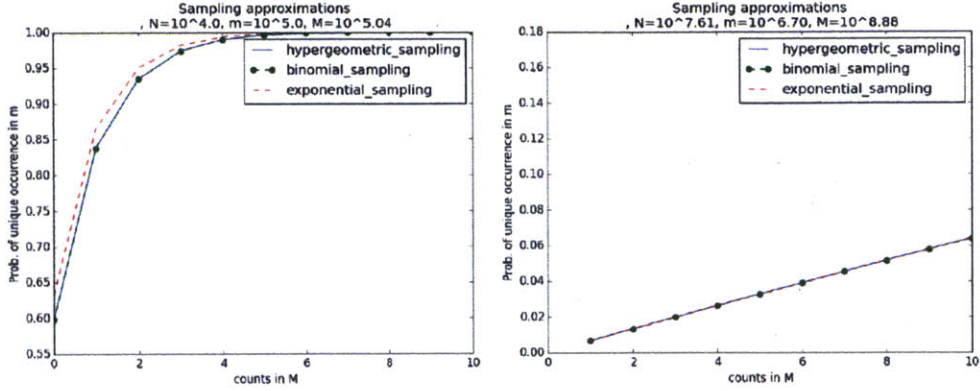


Figure 3-1: Simulation showing the effect of various sampling methods (left) that assume sampling without replacement (hypergeometric distribution) sampling without replacement (binomial) and large sample size (exponential sampling). These three approaches are largely equivalent at the scale of library size and sampling observed in exome sequencing studies (right).

round the count would be expressed as $A_i^j = 2A_i^{j-1}$. Thus a starting seed of count 1 would yield a final count of 2^k after k rounds of duplication.

In the case where there is a non-zero chance of seed i failing to duplicate during the j th round of PCR, one can express duplication in terms of a fixed probability α_i unique to the seed.

$$A_i^j = A_i^{j-1} + \text{Bin}(A_i^{j-1}, \alpha_i) \quad (3.7)$$

Formula 3.7 shows that during the j th duplication round, a series of A_i^{j-1} Bernoulli trials can be modeled, each with the probability of α_i of success. Similarly, if each α_i is drawn from a beta distribution, the counts for each seed as described by equation 3.7 are equivalent to

$$A_i^j = f\left(\sum_{k=1}^{j-1} A_i^k, \alpha, \beta\right) \quad (3.8)$$

where $f(k|n, \alpha, \beta)$ is the beta-binomial distribution with the shape parameters α and β controlling the duplication rate of all seeds. Since the exact sequence of $A_0, A_1 \dots A_j$ is unknown, the final counts of the library can be simulated as a long series of long

beta-binomial events. Here, the number of trials $n = \sum_{k=1}^{j-1} A_i^k$ is unknown and can be treated as a parameter of the data.

The beta-binomial process is often described in terms as draws from a Pólya urn model. During each draw, a molecule is selected from an existing library and placed back into the library along with c duplicate copies of the same molecule. Given a library with only two types of molecules, red and black, let a represent the starting count of red molecules and b represent the starting count of black molecules. Let X_i represent the outcome for a single trial i where $X_i = 1$ represents a draw and duplication of a red molecule and $X_i = 0$ otherwise. We wish to describe a sequence of such events with length n in the form $(x_1, x_2, \dots, x_n) \in \{0, 1\}^n$. It can be shown that for the $c = 1$ case, the finite dimensional distribution of such a sequence is equivalent to the a beta-Bernoulli process [6]. For the Bernoulli trial sequence \mathbf{X} with parameter p , the probability of a specific outcome is

$$P(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = \frac{a^{[k]} b^{[n-k]}}{(a+b)^{[n]}} \quad (3.9)$$

where $k = x_1 + \dots + x_n$ and $m^{[j]}$ represents the ascending power $m(m+1)(m+2)\dots(m+j-1)$. Similarly, the success of the first n trials can be represented as $Y_n = \sum_{i=1}^n X_i$ with the beta binomial probability mass function

$$P(Y_n = k) = \binom{n}{k} \frac{a^{[k]} b^{[n-k]}}{(a+b)^{[n]}} \quad (3.10)$$

Which can be equivalently represented using the beta distribution $B(\alpha, \beta)$

$$P(Y_n = k) = \binom{n}{k} \frac{B(k+a, n-k+b)}{B(a, b)} \quad (3.11)$$

If the Pólya urn process is applied to a single molecule, the model can be simplified further, with a starting library of size N , we can label $a = 1$ and $b = N - 1$ and the

PMF for k amplification events for a single seed becomes:

$$P(Y_n = k) = \binom{n}{k} \frac{(N-1)^{[n-k]}}{(N)^{[n]}} \quad (3.12)$$

and the probability that a given seed i will have a final count $k_i = k$ in a library becomes

$$P(k_i = k | n, N) = \binom{n}{k-1} \frac{(N-1)^{[n-k-1]}}{(N)^{[n]}} \quad (3.13)$$

Figure B-6 verifies equation 3.13 by plotting the predicted counts from equation 3.13 with a simulated sequencing library generated by a Pólya urn processes.

To relate this model to the observed data, the total number of molecules M in the library must be approximated

$$M = N \left[1 + \sum_{k=0}^{\infty} k \binom{n}{k} \frac{(N-1)^{[n-k]}}{(N)^{[n]}} \right] = N + n \quad (3.14)$$

Given this equation for M , along with the sampling approach described in equation refeq:ztpDup, the duplication rate is estimated as follows:

$$= 1 - \frac{N}{m} \sum_{k=1}^{\infty} (1 - e^{-\frac{km}{M}}) \binom{n}{k-1} \frac{(N-1)^{[n-k-1]}}{(N)^{[n]}} \quad (3.15)$$

The results of this Pólya approach appear in Figure 3-2. The Pólya urn modeled showed a median error rate of -0.8% when trained using three read groups from each of 48 different exome samples. This compares to an error rate of -1.1% observed in the Picard ROI approach. Importantly, there was a reduction in the variance of model error for the Pólya model with a standard deviation of 2.7% as compared to 5.4% with the Poisson-based model. To improve physical interoperability, future analytical work could relate the number of Pólya draws and library size to the number of rounds of PCR and reaction efficiency for amplification.

Coincidentally, the biased, underestimation of optical duplicates that was reported

from the Poisson-based model (as discussed in Section 2.2) actually helped to lower error rates for the Picard ROI approach. This handling could have been employed with the Pólya model as a heuristic, but was avoided due to its arbitrariness and lack of physical meaning.

3.3 PMF based on the expectation of binomial branching

The expectation of samples observed after n cycles of PCR has been derived based on branching [7]. Given seed i starting with M_0 molecules, its expected count in the sequencing library after n rounds of duplication:

$$E[k_i]_n = M_0(1 + p)^n \quad (3.16)$$

This was incorporated into an objective function to parameterize the unknown library size N and the estimated number of duplication rounds n :

$$\min \left(\frac{c_i}{x_i} - \frac{N}{m_i} \sum_{k=1}^{\infty} p(k|n, \alpha, \beta) (1 - P_{k,c=0}^{(M,m)}) \right)^2 \quad (3.17)$$

Where the probability mass function $p(k|n, \alpha, \beta)$ comes from

$$p(k|n, \alpha, \beta) = (1 + B(k|a, b))^n / \sum_{c=1}^{\infty} (1 + B(c|a, b))^n \quad (3.18)$$

Where M_0 is assumed to be 1 for molecules $1 \dots N$ and the parameters to the beta function determine the success of duplication for a given insert during each round of duplication and are assumed to result in a high PCR yield (e.g. $\alpha = 5$ and $\beta = 5$). The total count of molecules in the library is estimated using the mean probability of success from the beta distribution $M = N(1 + \frac{a}{a+b})^n$. This approach is not expected to capture the variance of count as distributed in the library since the model is only capturing the mean count across duplication rates sampled from the beta distribution.

Figure B-8 shows the results of a simulation of branching process. The left shows how equation 3.18 maps well onto a simulated branching event when the duplication rate is low (the ratio of $\alpha : \beta$ is low), while the estimation deviates when the ratio $\alpha : \beta$ grows larger.

3.4 Occupancy-based estimates

Given a sampling approach, one way to estimate library properties is based on the size of duplicate sets. Beyond the aggregate count of duplicates (as used in the Li method), the distribution of duplicate set sizes at a given point in a sequencing study can give additional information to predict the future changes in library complexity. The observed occupancy vector is defined as the number of DNA inserts with count k in a sequencing study. This idea has been used in recent studies [3] and is applied here to the Pólya urn modeling.

To enable predictions based on duplicate occupancy, a new method was added to the MarkDuplicates tool. In the MarkDuplicates tool a sorted, paired-end read list is first build, storing features of a SAM record relevant for duplicate marking. These include the read pairs indices as mapped onto the reference, pair orientation, read group information and xy coordinates on the sequencing tile. The collection of features for each record is then grouped into sets of paired-end reads that share same start and end reference indices. The MarkDuplicates tool was modified here to return back the size of these duplicate sets as the tool traverses all of the records in a SAM file. Optical duplication events are defined as those records in a duplicate set that occur on the same tile and within a fixed distance in the x or y direction. Optical duplicates must have the same strand orientation. The MarkDuplicates tool was also modified to return the set size of optical duplicates. After evaluating all records of the input data, the tool was modified to return a histogram of counts for duplicate set sizes for an input SAM or BAM file.

Once an occupancy vector is obtained from a sample, this data can be used to parameterize a model of sequence duplication. Let $O_m(k)$ be the count of duplicate

sets of size k after m DNA inserts have been sequenced from a library. Once parameterized, our model will estimate the function $O_M(k)$, the number of duplicate sets of size k in the larger sequencing library containing M total molecules. To estimate $O_M(k)$ we seek parameters for our model so as to minimize the distance between the observed and expected sampled occupancy vector. The expected occupancy vector is calculated by enumerating all possible cases in which a unique molecule with $O_M(k) = j$ and $O_m(k) = l$ where $j \geq l$. Each of such cases is weighted by multiplying the probability of count j in the large library by the probability of being observed l times in a sample of size m .

$$E[O_m(l)] = N \sum_{j=1}^{\infty} p(j|N, n) \text{bin}(l|m, \frac{j}{M})$$

The difference between this expectation and the observed counts is minimized with a least squares approach across all observed values of l . Algorithm 3 depicts using a least squares approach to estimate the parameters for the library size and the number of Pólya draws needed for the model.

Data:

n sample BAMs files

for $sample = 1$ to n **do**

 Estimate Library size N and number of polya draws n

$$\left| \min \left(O_m(k) - N \sum_{j=1}^{\infty} p(j|N, n) \text{bin}(k|m, \frac{j}{M}) \right)^2 \right.$$

 Calculate number of amplicons

$$M = \sum_{k=1}^{\infty} k O_M(k)$$

for $x = 1$ to $max\text{-multiple}$ **do**

$$\quad | \quad u(x) = 1 - \frac{1}{xm} \sum_{k=1}^{\infty} O_{xm}(k)(k-1)$$

end

end

Output:

$u(x)$ is the estimated fraction of unique molecules at x multiples of sequencing

Algorithm 3: Pólya urn projections based on occupancy vector

Error rates for this approach are depicted in Figure 3-2. At present, Pólya urn projection based on a single occupancy vector under performs relative to training the same model with multiple points along the complexity curve using Equation 3.15. Figure B-9. shows the observed occupancy for multiple read groups based on the

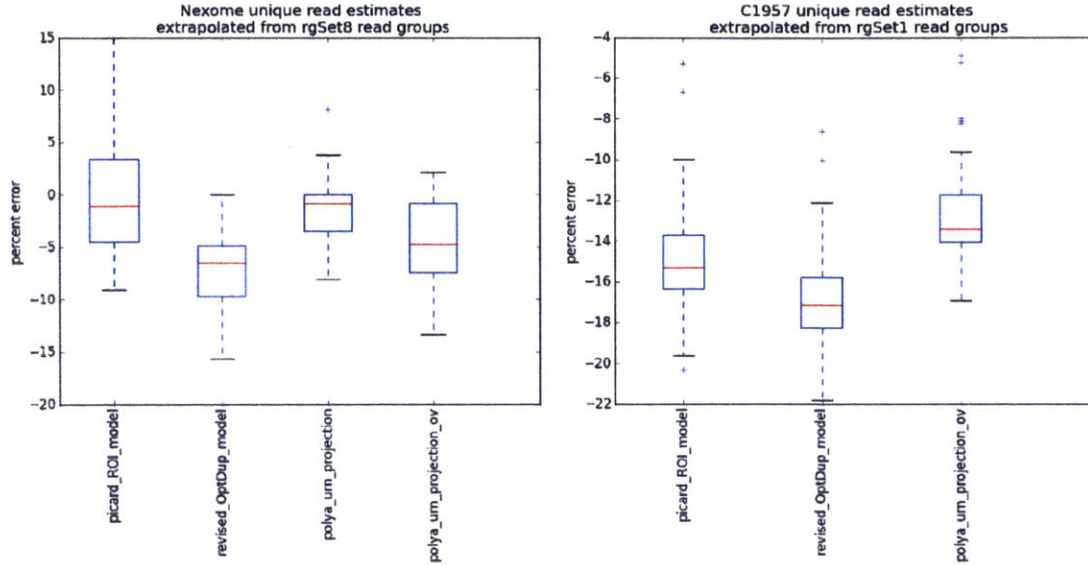


Figure 3-2: Error rates for the occupancy-based estimates are compared to multi-point Pólya estimates as well as Poisson-based models for exome samples. The graph on the right depicts results obtained from training the models on the three read groups. The left depicts a graph based only on one read group.

occupancy projections from the Pólya urn model. These are plotted with the projections fitted to the observed occupancy based on the Pólya urn model. Considerable error exists in this fit and future work will aim to address this issue by 1) modifying the objective function to penalize according to the log difference in entries of the observed minus expected occupancy and 2) improving the strategy for parameterization by trying a gradient decent or a log-grid search with local refinement.

Chapter 4

Parallel duplicate marking with Julia

Factors affecting computer language design and selection in computational biology have been discussed at length in recent years [8] [9]. Historically, the field's most frequently used and computationally demanding tools, including algorithms for DNA alignment and sequence similarity queries, have been implemented in low-level languages such as C, C++, or Fortran. The ability to compile such tools into highly efficient machine code has provided obvious performance benefits, but more recent scientific and computational demands in biology have transitioned, motivating many scientists to begin using high-level and dynamically typed computer languages. This transition has been driven, in part, by the flexibility offered by high-level languages in response to fast-paced changes in data collection and data structures that arise with new experimental techniques from molecular biology. Rapid development of new tools and data types have put a heavy emphasis on prototyping, ease of development cycles, and code maintainability [9]. For example, the Perl computer language gained popularity in bioinformatics during the 1990s for its syntactic brevity, handling of regular expressions, and support of both procedural and object-oriented programming. More recently, computational biologist have relied heavily on high-level languages including R and python to carry out cycles of analysis and processing. The community of researchers developing new bioinformatics capabilities in dynamic programming languages often place emphasis on functionality that increases the ease of data handling and statistical modeling. The Julia language, with strength in numeric computing,

is poised to build on the best parts of other high-level languages by extending performance and code interpretability when working with very large data sets observed in DNA sequencing studies. A few of Julia's most important design features include the use of multiple dispatch, just-in-time compilation, and metaprogramming.

New models to predict the impact of sequencing duplicates have recently been described in the literature [2] and in the current study, but the technical infrastructure to run these computations at scale are currently lacking. Julia's library of mathematical packages, its speed in numeric computing, and functionality to support operations performed in parallel are likely suited to many challenges in genomic processing and analysis. To test this hypothesis, Julia was used to implement an existing modeling strategy to predict a commonly observed sequencing artifact [2] in order to improve performance for return on investment calculations. Here, new Julia functionality is introduced in order to launch and manage duplicate marking events in sequencing files. A Julia wrapper was created to manage existing C code [2] for predicting return on investment using methods based on the Good Toulmin model. Because the approach of the Daley method is predictive in nature, rather than explicitly modeling the multiple physical sources of duplication, the results from this section are not formally compared to the modeling approaches from the work of the current study. Instead, we focus on the infrastructure for managing an alternative MarkDuplicates strategy. Results in this chapter represent exploratory work that was completed for the MIT course 15.337 during the fall semester of 2015 in collaboration with Mukarram Tahir and Andres Hasfura.

Benchmarking was performed using existing preseq C++ code [2] that marks duplicates and computes a complexity curve for a series of aligned data in BAM format. This was applied to 50 exome sequencing samples, allowing us to perform timing profiles and identify bottlenecks. Through a combination of profiling efforts, we mapped the procedure through which preseq generates the complexity curve for a given BAM file input. First, POS and FLAG entries are extracted for each read from the binary BAM file, which respectively correspond to the position of a read on the reference genome and an integer value from which read properties (such as whether it

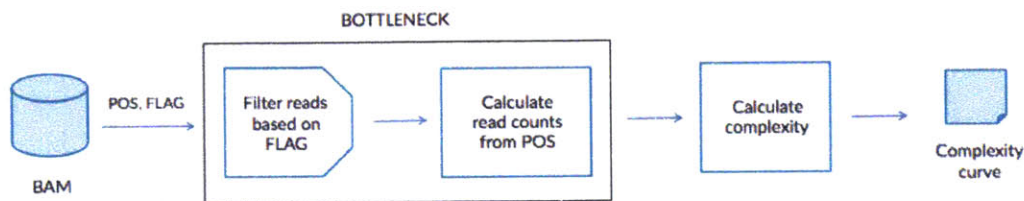


Figure 4-1: Overview of the processing and modeling strategy used by preseq. We identified read filtering using FLAG and read counting using POS as the bottleneck best suited for optimization and parallelization in Julia.

is a primary read or if it is mapped to the reference) can be deduced through bitwise operations. Preseq then filters the reads based on their FLAG values, and then proceeds to identify duplicates among these reads. Two reads are classified by preseq as duplicate if their position on the reference genome (signified by POS) are identical and a count of these records is maintained. These so-called read counts are then used for calculating the complexity curve, which is the output of the code. This procedure is summarized in the schematic shown in Figure 4-1. Of these steps, we determined that approximately 95% of all compute time was expended in loading, filtering, and counting duplicate reads in the BAM sequencing file. Given the near embarrassingly parallel nature of these tasks, we recognized the opportunity to re-implement them in Julia and attempt to achieve speedup through its parallel computing capabilities. Once the read counts are calculated, the C++ preseq code can be called with these pre-calculated counts (rather than the original BAM file) for the final complexity curve calculation.

The product of our efforts to parallelize the preseq.cpp code was a Julia wrapper preseq.jl, which handles all parts of the computation except the final calculation of complexity from read counts. The input to preseq.jl is a binary BAM file, but unlike preseq.cpp, this cannot be directly read for FLAG and POS values as there is currently no Julia library for interfacing with BAM alignment files. We therefore utilize an external executable known as samtools for converting the binary BAM file to a plaintext SAM file, which is then read into a DistributedArray object. When worker nodes are spawned in Julia, each node then has access to only a portion of the

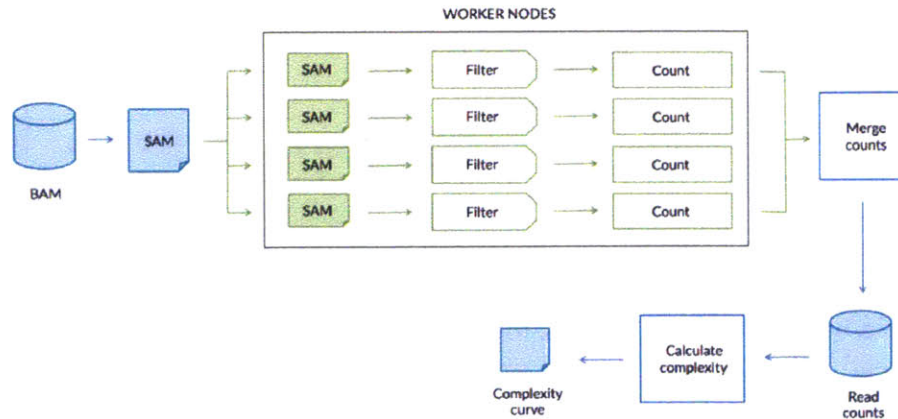


Figure 4-2: Schematic of the `preseq.jl`, a Julia wrapper that we implemented for complexity curve calculation from a BAM file input. Julia worker nodes perform the filtering and read count calculations in parallel, and the result is offloaded to the existing `preseq.cpp` code for the final complexity calculation.

SAM entries for further processing. These worker nodes proceed to extracting POS and FLAG fields from their given set of SAM entries, and then apply filters based on FLAG in a manner that is identical to the original C++ code. The filtered reads are then examined for duplicates, and a count of duplicates is maintained at each worker node. Given that a duplicate may be present across multiple worker nodes, the individual count arrays from the worker nodes are examined for overlap once they are returned to the master node. The final counts are then written to a temporary file on disk, and `preseq.cpp` is called with this count file (rather than the BAM file) for a much faster complexity curve calculation. The overall implementation is summarized in Figure 4-2.

From our earlier benchmarking of the serial `preseq.cpp` code, we determined execution times of approximately 5, 7, and 12 seconds respectively for 35 Mb, 67 Mb, and 135 Mb BAM files. For comparison, we ran our Julia wrapper `preseq.jl` on these three input file sizes, and varied the number of cores to determine reduction in execution time as the number of cores is increased. Figure 4-3 shows a plot of these execution times, and indicates a substantial speedup as the first few cores are added. A plateau in speedup is quickly reached, but the plateau varied according to file size. Given that production BAM files are of much larger sizes, we expect efficient consumption of a

large number of cores before such a point of diminishing returns is reached. It is concerning though, that the execution time at which these curves level off is still higher than the execution time of the serial `preseq.cpp` code for the corresponding input file size. To investigate this, we visualized the various components of the execution time for the 135 Mb input file, as shown in Figure 4-3 (lower). We notice immediately that filtering and counting reads, which were subject to parallelization, contribute to the majority of the speedup as the number of cores is increased. However, there appears to be a constant and substantial overhead associated with reading the BAM files in `preseq.jl` that is absent in the `preseq.cpp` code. Unlike the `preseq.cpp` code, which is able to directly access POS and FLAG fields for all entries using a C++ API that interfaces with BAM files, our `preseq.jl` code uses an external executable (`samtools`) to first convert the binary BAM file to a plaintext SAM file, and then extract POS and FLAG fields from each plaintext entry. This leads to a massive performance loss in our Julia wrapper (compared to the original serial code). This is not especially concerning, however, since a Julia library for interfacing with BAM files currently appears to be under development, and future iterations of this code could utilize this work to eliminate this bottleneck associated with reading BAM files.

Here, the use of the Julia language is demonstrated for extracting and analyzing frequency information from raw DNA sequencing data. Our results provide insight into Julia's suitability for genomic analysis including rapid prototyping, error analysis with C++ code, and an exploratory parallel algorithm for the identification of duplicate sequencing reads.

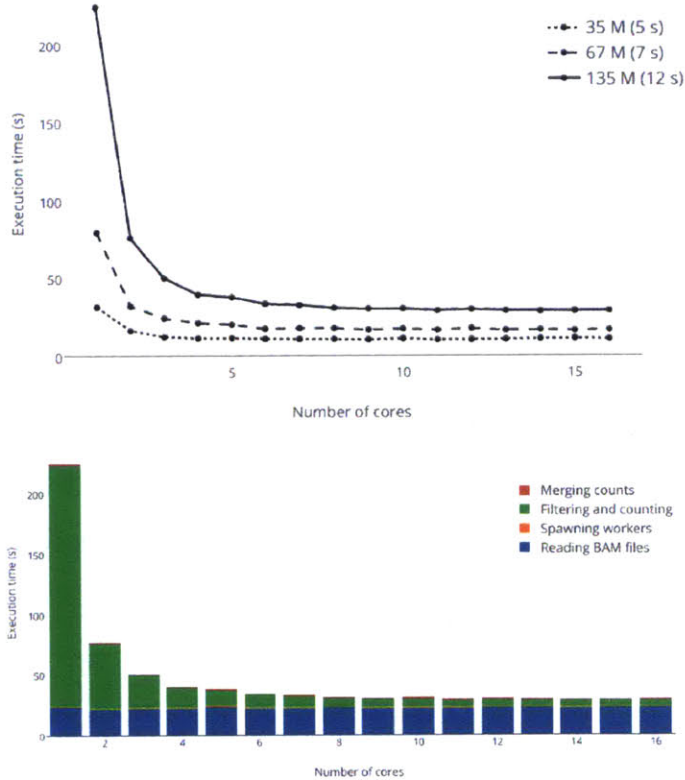


Figure 4-3: Execution time of `preseq.jl` as a function of the number of parallel cores for three input BAM file sizes (upper). For comparison, the execution time of the serial `preseq.cpp` code is shown in parenthesis in the legend. (Breakdown of the execution time for the 135 Mb BAM file lower), demonstrating excellent performance improvement in read count calculation as the number of cores is increased, but significant overhead incurred from reading the BAM file to a plaintext SAM file in memory.

Chapter 5

Conclusion

This study demonstrates modest improvements in library complexity prediction by physically modeling phenomena that contribute to duplication rates in next generation sequencing data. The approach of modeling PCR with a Pólya urn model and handling optical duplication events as a constant source of duplication reduced the overestimates of PCR duplication relative to predictions from a commonly used Poisson-based model. The Pólya urn modeled showed a median error rate of -0.8% when trained using three read groups from each of 48 different exome samples. This compares to an error rate of -1.1% observed in the Picard ROI approach. Importantly, there was a reduction in the variance of model error for this model with a standard deviation of 2.7% as compared to 5.4% with the Poisson-based model. As the prediction gains were modest in physical modeling, the improved interpretability of the Pólya urn model is highlighted as a benefit of the current approach. Such interpretability is important as sequencing technology continues to evolve along with new library construction techniques. Better understanding the factors that contribute to library complexity will play a role in identifying, predicting, and eventually reducing duplication events through through informed technology development.

In extending the work of the current study, future efforts may further compare predictive vs. explanatory models of library complexity. The slope-offset method reviewed in this study, as well as more sophisticated curve-fitting approaches performed in other studies, could play an important predictive role despite lacking model com-

ponents with a physical interpretation. Lastly, the results herein show the promise of using complexity curve estimates based on the observed occupancy vector of duplicate sets. This approach is advantageous because it allow for ROI predictions based on aggregated counts from a single read group. Error rates observed from this approach are not yet favorable with the naive implementation demonstrated in this work. A number of adjustments have been proposed to improve this approach, specifically around improving the objective function and strategies for parameter optimization.

Bibliography

- [1] E. LANDER, ET AL., *Initial sequencing and analysis of the human genome*, Nature 409, 860-921 (2001)
- [2] T. DALEY, A.D. SMITH, *Predicting the molecular complexity of sequencing libraries*, Nature Methods 10.4, 325-327 (2013)
- [3] C SCHRODER, S. RAHMANN, *Efficient duplicate rate estimation from subsamples of sequencing libraries*, Preprint (2010)
- [4] G. SHMUELI, *To explain or predict?*, Statistical Science 25:3, 289-310 (2010)
- [5] H. LI., *Mathematical Notes on SAMtools Algorithms*, (2010)
- [6] K. SIEGRIST., *Probability, Mathematical Statistics, Stochastic Processes*, <http://www.math.uah.edu/stat/index.html> (2010)
- [7] G STOLOVITZKY, G. CECCHI, *Efficiency of DNA replication in the polymerase chain reaction*, P.N.A.S (1996)
- [8] R. GENTLEMAN, ET AL., *Bioconductor: open software development for computational biology and bioinformatics*, Genome Biol. 5(10), R80 (2004)
- [9] M. NAIR, R. HANNES, R.H. GRIBBSKOV, *Perl in bioinformatics*, Modern Programming Paradigms in Biology Encyclopedia of Genetics, Genomics, Proteomics and Bioinformatics DOI: 10.1002/047001153X.g409321 (2005)
- [10] F. HOPPE, *Polya-like urns and the Ewens' sampling formula*, J. Mathematical Biology (1984)
- [11] N. LALAM, *Estimation of the reaction efficiency in polymerase chain reaction*, J. Theoretical Biology (2006)
- [12] F. SUN, *The polymerase chain reaction and branching processes*, J. Comp. Biology 2:63-86 (1995)

Appendix A

Algorithms

Data:

Unique molecule counts: $\mathbf{C} \in \mathbb{Z}^{m \times n}$, m Length bins, n GC bins

Total molecule counts: $\mathbf{M} \in \mathbb{Z}^{m \times n}$, m Length bins, n GC bins

Optical duplication rate: d_r

Result: expected fraction of unique molecules for after x sequencing multiples initialization;

matrix to contain library size estimates

$\mathbf{L} \leftarrow (m \times n)$ matrix of zeros

matrix to contain molecule counts modeled under Poisson distribution

$\mathbf{M}_p \leftarrow (m \times n)$ matrix $\mathbf{M} - d_r \mathbf{M}$

matrix to contain unique molecules modeled under Poisson distribution

$\mathbf{U} \leftarrow (k \times m \times n)$ matrix of zeros

```
for  $i = 1$  to  $m$  do
  for  $j = 1$  to  $n$  do
     $c = \mathbf{C}[i,j]$ 
     $m_p = \mathbf{M}_p[i,j]$ 
     $\mathbf{L}[i,j] = \text{Solve for } \mathbf{N}_p \text{ in } \frac{c}{N_p} = 1 - e^{-m_p/N_p}$ 
    for  $x = 1$  to  $k$  do
      if  $m_p \leq \text{minCount}$  then
         $\mathbf{U}[x,i,j] = \frac{N_p}{x m_p} (1 - e^{-m_p/N_p})$ 
      else
         $\mathbf{U}[x,i,j] = \text{global unique estimate}$ 
      end
    end
  end
end
end
```

Algorithm 4: ROI exponential mixture model

Appendix B

Figures

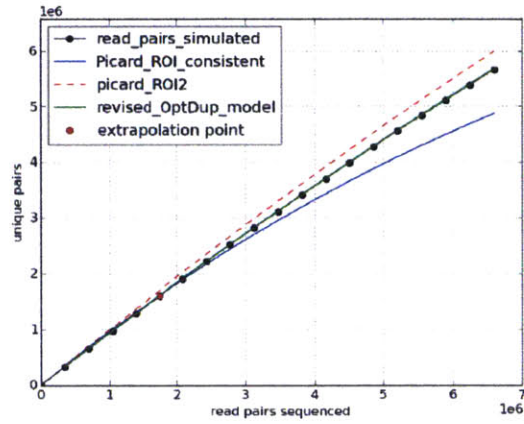


Figure B-1: Simulation of read pairs drawn from a Poisson-based process (black). These results are plotted with the revised optical duplicate model (green), the current Picard ROI equation (red), and the consistent Picard equation (blue).

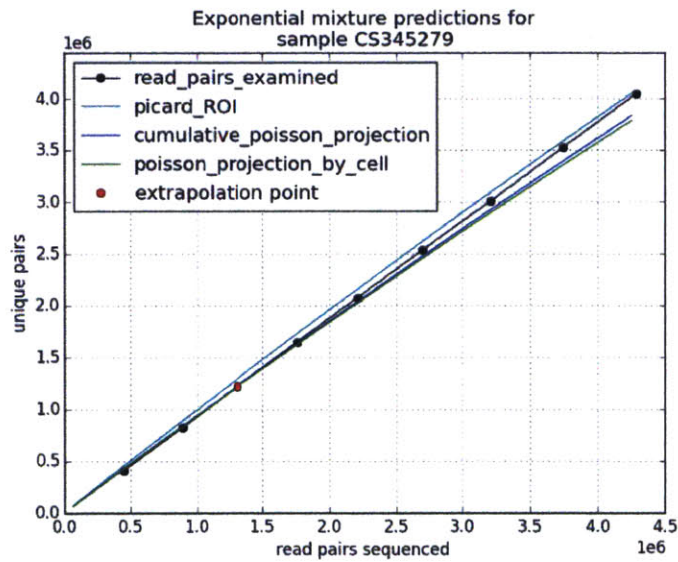


Figure B-2: Results for a whole genome sequencing sample showing predictions with and without incorporating DNA insert length and GC content)

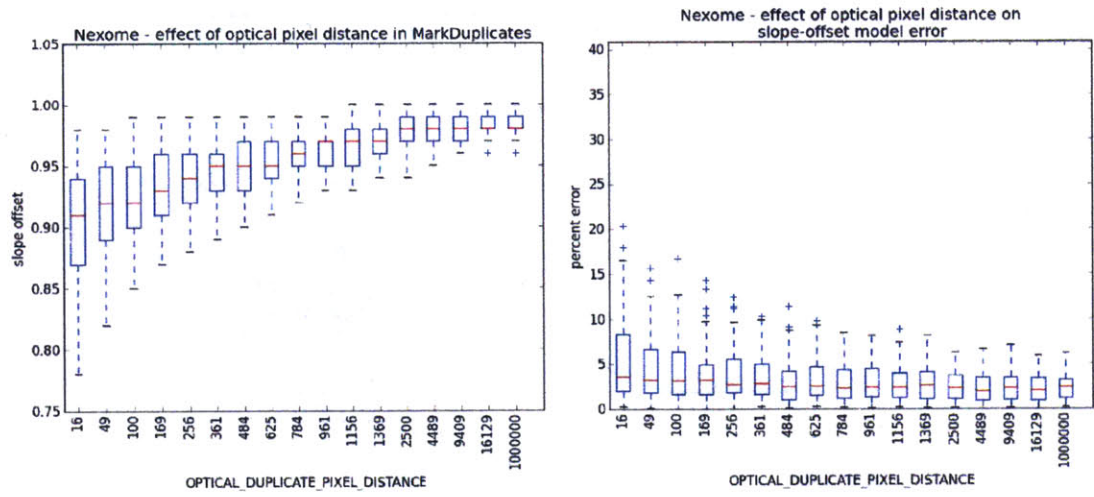


Figure B-3: The fraction of optical duplicates recorded (left) and the model error Poisson-based model.

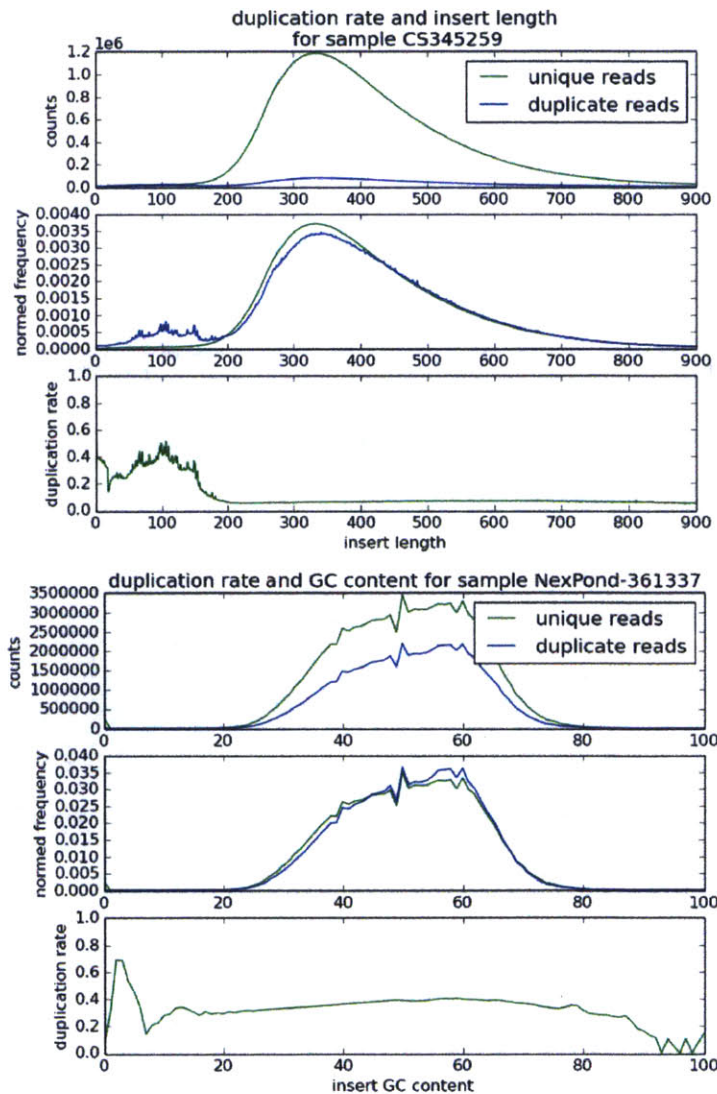


Figure B-4: Changes in raw count, normalized count, and duplication rate according to DNA insert length (top). The bottom shows changes in the same count features according to GC content.

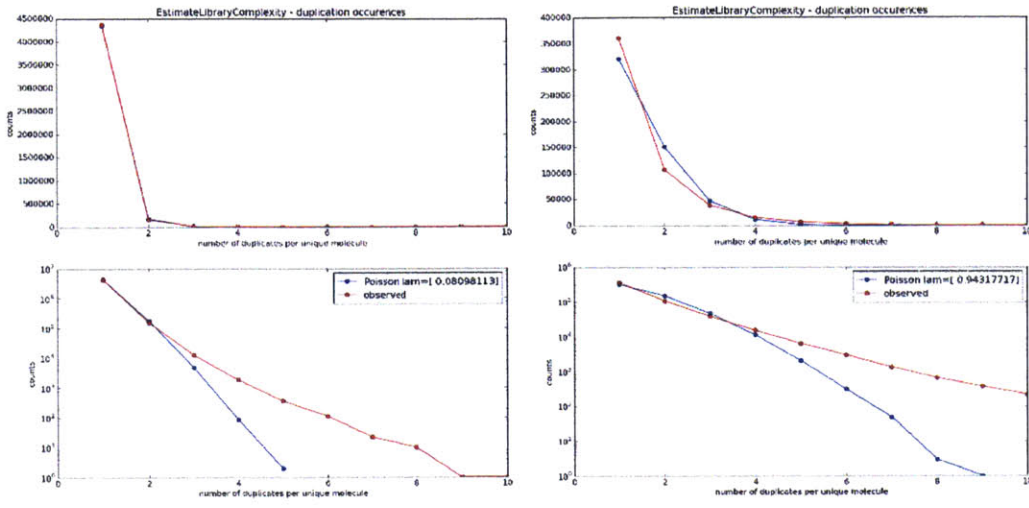


Figure B-5: Observed counts for duplication sets from EstimateLibraryComplexity for representative WGS (left) and exome (right) samples. Observed results (red) are plotted with expected counts drawn from a zero-truncated Poisson distribution.

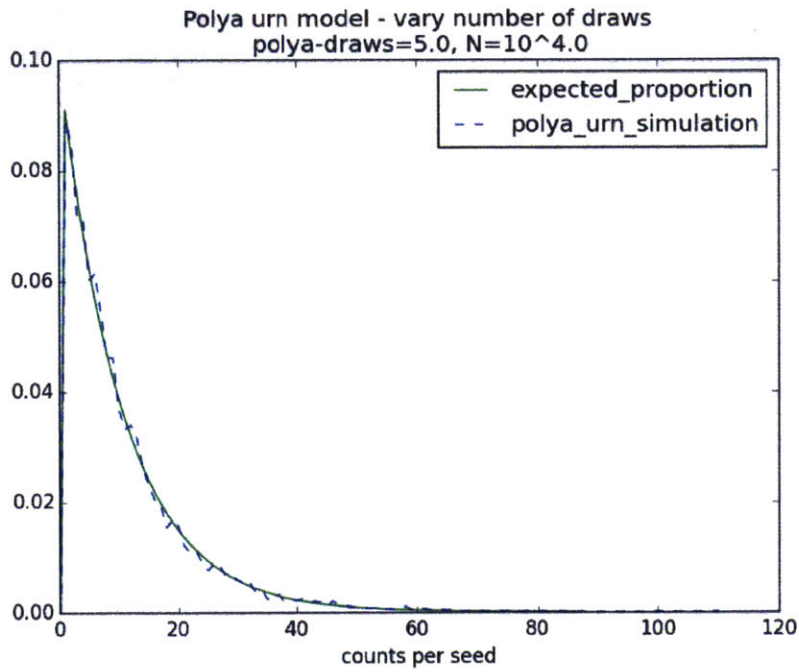


Figure B-6: Simulation of Polya urn drawing events plotted with expectation from Equation 3.18

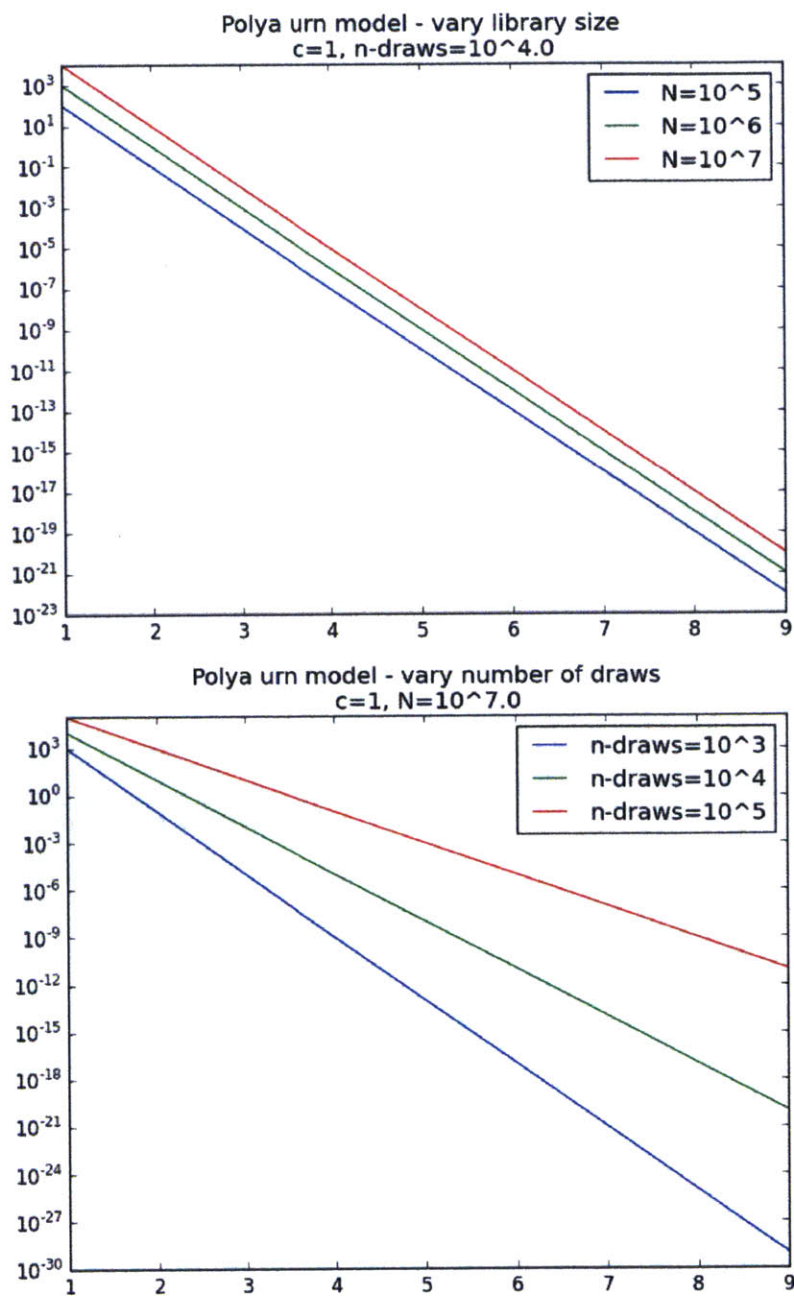


Figure B-7: Simulation showing the expected distribution of counts after varying the library size (top) and number of Polya draws (bottom).

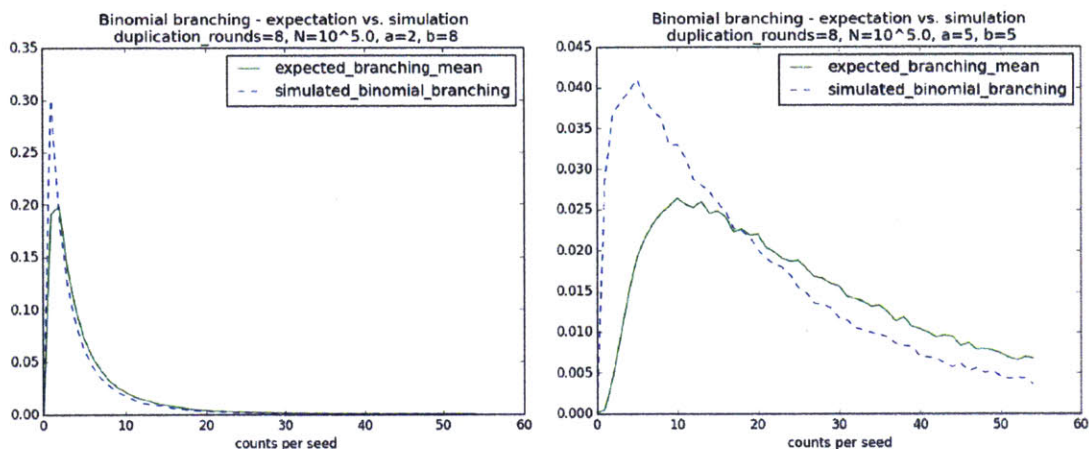


Figure B-8: Result of binomial branching simulation compared to expectation as described in equation 3.18

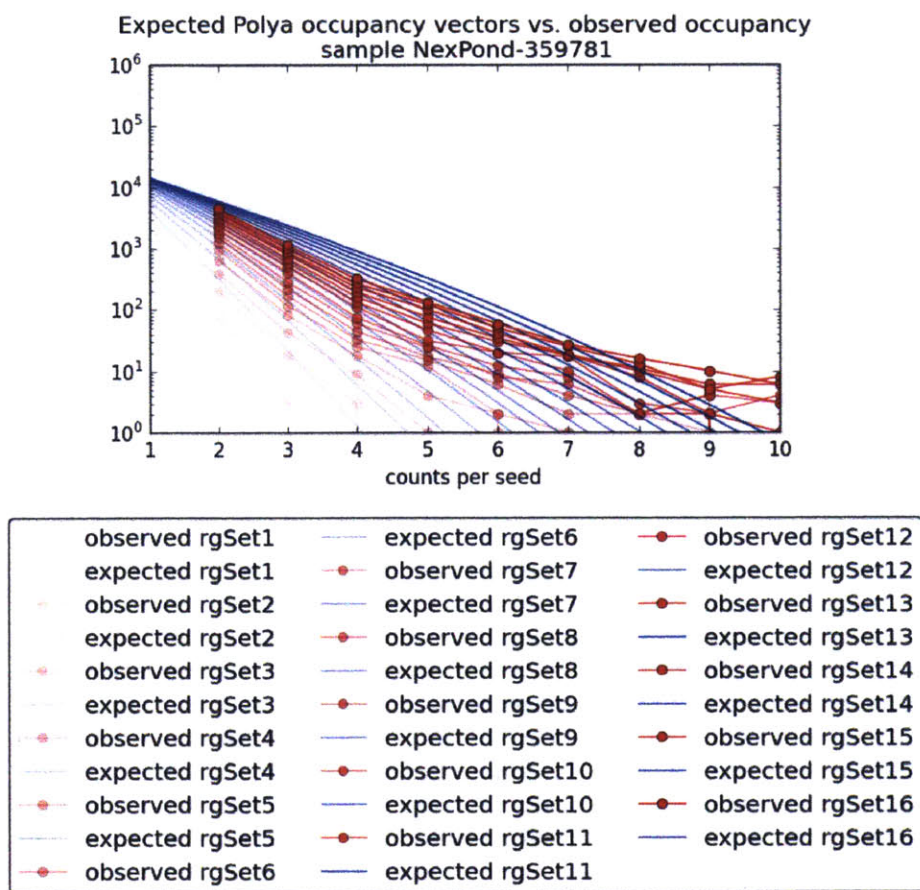


Figure B-9: Changes in observed occupancy as more read groups are added (red). The blue curves depict expected occupancy.