YOUR TITLE GOES HERE

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Scott Clark

August 2012

YOUR TITLE GOES HERE

Scott Clark, Ph.D.

Cornell University 2012

Your abstract goes here. Make sure it sits inside the brackets. If not, your biosketch page may not be roman numeral iii, as required by the graduate school.

BIOGRAPHICAL SKETCH

Scott Clark grew up in Tigard, Oregon and graduated from Central Catholic High School in Portland, Oregon in 2004. He recieved Bachelor of Science degrees in Mathematics, Computational Physics and Physics (Magna Cum Laude) from Oregon State University in 2008.

In 2008 Scott was awarded a Department of Energy Compuational Science Graduate Fellowship (CSGF) supporting his doctoral work at Cornell University Center for Applied Mathematics (CAM). He plans to move to San Francisco after graduation and has accepted a position at Yelp Inc.

This document is dedicated to my family.

Your constant, unconditional love and support at every stage of my education made this possible.

ACKNOWLEDGEMENTS

I have had the priveledge of working with many amazing educators and researchers throughout my academic career. They have all helped sculpt me into the student and person I am today. At Central Catholic High School Steve Workman, Paul Wallulis and Phil Collins taught me the value of hard work and opened my eyes to the world of math and science. My REU advisors, Prof. Daniel Cox of University of California Davis and Prof. Steven Tomsovic of Washington State University. My undergraduate advisors at Oregon State, Malgo P and Rubin Landau. My practicum supervisors: Dr. Nick Hengartner and Dr. Joel Berendzen of Los Alamos National Labratory; and Dr. Zhong Wang and Rob Egan of the Joint Genome Institute of Lawerence Berkeley National Labratory. My committee at Cornell: Prof. Steve Strogatz and Prof. Bart Selman. And, of course, my advisor Prof. Peter Frazier whose unwavering patience, support and encouragement made this possible.

And my family and friends...

My research was supported by a Department of Energy Computational Science Graduate Fellowship, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-FG02-97ER25308. I was also supported by a Startup and Production Allocation Award from the National Energy Research Scientific Computing Center (NERSC), which is supportedby the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Part I

ALE: Assembly Likelihood Evaluation

CHAPTER 1

ALE INTRODUCTION

Recent advances in metagenomics have enabled the study of the microbes directly from complex communities implicated in environment and health Tringe, 2005. DNA extracted from microbial communities can be investigated either by 16S rDNA sequencing to explore their phylogenetic diversity Costello 2009;, or by whole metagenome shotgun sequencing to understand their full genetic constituents Venter 2004; Tringe 2005;Qin 2010;Yooseph 2010; Hess 2011. With the unprecedented amount of data obtained from next generation high-throughput sequencing (NGS), individual genomes can be directly assembled from a mixture of thousands of genomes, without the need to isolate single genomes through laboratory cultivation or single cell sorting Hess, 2011. The ability to assemble a metagenome to resolve the genomes of individual species, or at least the abundant ones from a complex community, is crucial to understanding their community structure and dynamics and to exploring interspecies interactions, thereby providing deeper insights into the function of the community.

Assembly of individual genomes from metagenomics datasets generated by NGS poses significant informatics challenges. Many of these challenges, including short read length, noisy data and large data volume, are also faced in the assembly of single large genomes from short reads (reviewed in Lin 2011; Li 2010). Beyond those faced in assembling single genomes, there are also several challenges unique to metagenome assembly. First, unlike in single genome assembly where the sequence depth for the target genome is expected to be uniform, the sequencing depth of genomes in a metagenome usually vary greatly. Second, most genome assemblers have difficulties in resolving repetitive regions within a single genome, and this problem is exacerbated in metagenome assembly because conserved genomic regions and lateral gene transfer have greatly increased the portion of the repetitive genomic regions. Finally, although there are quite a few single genome assemblers such as Velvet Zerbino and Birney 2008, ABySS Simpson et al 2009, soapDenovo Li 2010 and All-Path-LG Gnerre et al 2010 that are capable of assembling large genomes, there is no metagenome specific assembler yet. Instead, assemblers designed for single genomes are being applied to metagenome data without significant modifications Qin 2010; Hess 2011. The impact of using an assembler developed to assemble single genome has not yet been systematically evaluated for metagenome assembly, especially in how well it addresses challenges like variable sequencing depth and closely related species that are unique to metagenomes. Quantitative measurement of the quality of a metagenome assembly, as well as the ability to compare the results of different assemblers from the same data set, are so far impossible. Many current studies either use only the overall size of assembly (N50), which ignores accuracy, or maps the resulting assembly onto some known reference genomes obtained independently to estimate the accuracy Hess 2011, but such references are not available in most cases. In this work we focus on the accuracy of the proposed genome using only the proposed assembly and the reads used to create it. This allows us to pinpoint localized errors instead getting bogged down trying to quantify the completeness of an entire metagenome.

Quality measurements of a single genome assembly have been derived by using a basic statistical framework integrating several metrics Lin 2011; . These metrics in general include the completeness (the total assembled bases), the contiguity (N50), and the accuracy (how well the assembly is supported by the reads and mate pairs) Li 2010. The first two often improve as the sequencing depth increases, and the accuracy at the base level is also correlated with sequence depth. However, accuracy at the contig/scaffold level is often independent of sequence depth, as chimeras can form by mis-joins and repeats regions can be misplaced and miscounted, which requires additional assessments. Such assessments can be derived from the original reads that formed the assembly, for example, by mapping the reads back to the proposed assembly Zhong, Rob: REF? Maybe bowtie?. Good assemblies require mate pairs to map to the same loci, insert length to be normally distributed around the mean, and most reads to map onto the assembly Zhong, Rob: REF? Or common knowledge?. Additionally they require that when reads are mapped onto the assembly their coverage, or depth, must be consistent with the Poisson distribution, which follows from a random shearing process Lander 1988. Unfortunately these requirements may not be directly applied to evaluate metagenome assembly quality. For example, single genome assemblies require all read pairs to be oriented consistently by the assumption a single genome, a constraint not followed when dealing with metagenome assemblies because read pairs can map to similar regions on different genomes.

In this work we aim to provide a comprehensive integrated framework for evaluating the quality of single genome and metagenome assemblies. In related work, Phillipy 2008 also proposed a method for evaluating the quality of a whole single genome, but the method proposed is a pipeline of conceptually separate techniques for evaluating the different aspects of genome quality. Choi 2008 also combined evidence from several conceptually separate measures of genome quality to identify mis-assemblies. In the previous literature, those works that use a single statistical framework tend to focus on only a single aspect of genome quality: Zimin 2008 introduced a metric (CE statistic) for finding gaps in an assembly by comparing to the genome of a related organism; Meader 2010 developed a statistical method for estimating the rate of erroneous insertions and deletions present in an assembly by comparison to an assembly of a closely related species; Olson 2009 uses mate pair information and a reference genome from a similar organism to identify assembly errors and structural variation; Kelly 2010 introduced a method that detects false segmental duplication using mate pair and coverage information. All of these previous approaches contrast with the current work, which is an integrated method for validating several aspects of genome and metagenome assembly quality simultaneously based on a single statistical model. Like the current work, Laserson 2011 also used a single statistical model to assign a likelihood score to assemblies, focusing specifically on the metagenomic case, but the statistical model used ignores the role of mate pairs in assessing assembly quality, which are becoming more prevalent with NGS technologies such as Illumina and PacBio strobe reads..

In this work, we measure the overall quality of an assembly in a mathematically rigorous way, using a probabilistic model for the way that reads are generated from a genome. Using Bayesian statistics, we give explicit expressions for the probability that an assembly is correct, and computational methods based on these expressions. These mathematical methods avoid the pitfalls of summary statistics like N50 score, which only capture one dimension of assembly quality. The methods provided may be used in a number of ways. First, they are useful as a tool for comparing different assemblies of the same genome or metagenome, and can be used to show how much more likely one assembly is than another of being correct. Two assemblies with roughly equal likelihood of correctness can be considered as having roughly equal quality, while if one assembly has a likelihood that is much lower, then we may safely discard it as inferior. Second, when considering a single assembly in isolation, these methods can also be used to give an absolute score that indicates the quality of this assembly, and how this quality compares to the quality of other assemblies. Third, these methods can be used to examine a single assembly and identify errors and their location, which is particularly useful when finishing the assembly.

CHAPTER 2

ALE METHODS

2.1 The ALE Score and the likelihood of an assembly

The ALE framework is founded upon a statistical model that describes how reads are generated from an assembly. Given a proposed assembly (a set of scaffolds or contigs), S, and a set of reads R, this probabilistic model gives the likelihood of observing this set of reads if the proposed assembly were correct. We write this likelihood, $P(R|S)$, and its calculation includes information about read quality, agreement between the mapped reads and the proposed assembly, mate pair orientation, insert length (for paired end reads), and sequencing depth. This statistical model also provides a Bayesian prior probability distribution $P(S)$ describing how likely an assembly S would be, if we were unable to observe any read information. This prior probability is computed using the k-mer distribution of the assembly.

The ALE score is computed from these two values, and it is proportional to the probability that the assembly S is correct. We write this probability as $P(S|R)$. Bayes rule tells us that this probability is

$$P(S|R) = \frac{P(S|R)\,P(S)}{Z} \tag{2.1}$$

where Z is a proportionality constant that ensures that P(S—R) is a probability distribution. We see P(S—R) as a statistical measure of the overall quality of an assembly S. As is typical in large-scale applications of Bayesian statistics, it is computationally intractable to compute the constant Z exactly. The ALE score is computed by replacing the constant Z with an approximation described in the

Methods section.

Although the ALE score can be reported as a standalone value, and understood as an approximation to P(R—S), it is most useful for comparing two different assemblies of the same genome, using the same set of reads to evaluate them. As shown in the Methods section, the assembly with the higher ALE score is then also the one with the larger probability of being correct. Moreover, we show that the difference between two assemblies ALE scores describes their relative probabilities of correctness. If one assemblys ALE score is larger than the others, with a difference of x between their ALE scores, then the assembly with the larger ALE score is more likely to be correct by a multiplicative factor of ex. Below, we refer to the ALE score more precisely as the total ALE score, to differentiate it from the sub-scores (described below) used to construct it.

Figure 1 shows the pipeline used to compute the total ALE score. Given a set of reads and a proposed assembly, ALE first takes as input the alignments of the reads onto the assembly in the form of a SAM or BAM file (Li and Durbin 2009), which can be produced by a third-party alignment algorithm such as bowtie (Langmead et al. 2009) or bwa (Li et al. 2009). ALE then determines the probabilistic placement of each read and a corresponding placement sub-score for each mapped base, which describes how well the read agrees with the assembly. In the case of paired end reads, ALE also calculates an insert sub-score for all mapped bases of the assembly from the read pair, which describes how well the distance between the mapped reads matches the distribution of lengths that we would expect from the library. ALE also calculates a depth sub-score, which describes the quality of the sequencing depth accounting for the GC bias prevalent in some NGS techniques. The placement, insert and depth scores together determine P(R—S). Independently, with only the assembly and not the reads, ALE calculates the k-mer sub-score and P(S). Each sub-score is calculated for each scaffold or contig within an assembly independently, allowing for variations commonly found in metagenomes. The four sub-scores are then combined to form the total ALE score. The constituent calculations in this pipeline are described in the Methods section.

In addition, these four sub-scores are reported by ALE as a function of position within the assembly, and can be visualized with the included plotting package or imported in table form to another package such as the Integrative Genomics Viewer (IGV) (Nicol et al. 2009), or the UCSC genome browser (Kent et al. 2002). When used in this way, these sub-scores can be used to locate specific errors in an assembly.

2.2 Probabilistic ingredients of the total ALE score

We can combine the two probabilities, $P(R|S)$ and $P(S)$, to provide an expression for the probability of the assembly given the reads, $P(S|R)$. While this combined expression is too computationally expensive to compute exactly, ALE provides a summary measure of quality called the total ALE score that is proportional to $P(S|R)$ and can be used compare assemblies.

Below, we first describe how $P(R|S)$ and $P(S)$ are computed from a set of reads R and a given assembly S. We then describe how they are combined to compute the ALE score, and how the ALE score can be used to compare the qualities of different assemblies. We first provide an overview of how $P(R|S)$ and $P(S)$ are defined and computed, beginning with $P(R|S)$ and then discussing

$P(S)$.

The statistical model from which $P(R|S)$ is calculated supposes that each paired read is generated at random from the genome or collection of genomes according to the following process. First, the distance between the two paired read ends, and their orientation, is chosen at random from a distribution that is specific to the library used to generate them. Second, the locations of the mate pairs on that genome are chosen at random, potentially with a consistent GC bias. Third and finally, the content of each of the two paired ends are generated by taking the genomes true base pairs at the chosen locations and then copying these base pairs into the reported read, with a given probability of error, insertion, or deletion for each base pair given by the sequencers quality score.

The likelihood $P(R|S)$ that results from this process can be factored into three components

$$P(R|S) = P_{placement}(R|S)P_{insert}(R|S)P_{depth}(R|S) \tag{2.2}$$

The first component, $P_{placment}(R|S)$ describes how well the reads contents match the assembly at the locations to which they are mapped. The second component, $P_{insert}(R|S)$, describes how well the distances and orientations between each paired read match the distances and orientations that we would expect from the library. The third component $P_{depth}(R|S)$ describes how well the depth at each location agrees with the depth that we would expect given the GC content at that location. Contributions to these three quantities, as a function of position in the assembly, are used to produce the placement, insert and depth sub-scores.

Together with the likelihood of the reads R given the assembly S, $P(R|S)$, the

ALE framework also depends upon a Bayesian prior probability distribution over assemblies, written $P(S)$. $P(S)$ describes how likely we would believe the assembly S to be, if we did not have any read information. In this prior probability distribution, we encode the belief that within a single genome, each k-mer has a unique k-mer frequency. This is the frequency of any set of k base pairs appearing in order in the genome. This defines a 4^k dimensional vector that is conserved across a genome and can help discover when different genomes have been mistakenly combined in a metagenome setting (Teeling et al. 2004; Woyke et al. 2006). Because P(S) is determined by k-mer frequencies, we use the notation $P_{kmer}(S)$ rather than the more generic $P(S)$ when referring to this probability distribution. Contributions to $P_{kmer}(S)$ as a function of position in the genome is referred to as the k-mer sub-score.

2.2.1 Placement sub-score

$P_{placement}(R|S)$ quantifies the likelihood of observing a read r_i, or set of reads R, given an assembly S. It includes information about how the read maps onto the assembly, the quality score of each base and orientation.

We assume that every paired read is independent of all other pairs of reads, which allows us to write $P_{placement}(R|S)$ as

$$P_{placement}(R|S) = \prod_{r_i \in R} P_{placement}(r_i|S), \tag{2.3}$$

where $P_{placement}(r_i|S)$ describes how well the contents of a single read r_i match the assembly at the locations to which they are mapped, as well as how well the distance and orientation between that reads paired ends match the distance and orientation that we expect from the library. We assume independence of these distributions, allowing us to write this as

$$P_{placement}\left(r_i|S\right) = P_{matches}\left(r_i|S\right) P_{orientation}\left(r_i|S\right). \tag{2.4}$$

$P_{matches}\left(r_i|S\right)$ measures how well the read matches the section of the assembly to which it maps. Making the assumption that each base j of the read is correctly called by the sequencer independently with a probability equal to the bases quality score Q_j, we can write this as

$$P_{matches}\left(r_i|S\right) = \prod_{base_j \in r_i} P\left(base_j|S\right) \tag{2.5}$$

where

$$P\left(base_j|S\right) = Q_j \tag{2.6}$$

when the base j correctly matches the assembly and

$$P\left(base_j|S\right) = (1 - Q_j)/4 \tag{2.7}$$

when it does not. This expression follows from our modeling assumption that all 4 possible errors that the sequencer could have reported at that base (three different substitutions and deletion) are equally likely when the read does not match the sequence. This symmetry requires each of the four possible reported errors (a base not equal to the assembly or a deletion) to have equal probability. An insertion, which does not have a corresponding base in the assembly, is modeled similarly, with the 4 in the denominator representing the uniform likelihood of observing any of the 4 possible bases on the assembly at that position. The product across all bases in a read is then equal to the total probability of observing that particular read at the given location in the assembly.

If the assembly has an unknown base at the location (denoted by an N) then we set

$$P\left(base_j|S\right) = 1/4 \tag{2.8}$$

modeling the fact that there is no information about the correct base at that location with a uniform distribution over all 4 possible bases. If an ambiguity code is reported by the sequencer then the above expression is modified to account for a distribution over the possible bases encoded by the corresponding code.

Each read is only allowed to be placed at a single position in the assembly. If the aligner placed a particular read at more than one position we choose a single position at random, weighted by $P_{placement}(r_j|S)$ score for each proposed position of the read on the assembly. This allows for repeat regions to be properly represented with the correct number of reads in expectation.

The orientation likelihood, $P_{orientation}(r_i|S)$, is calculated by first counting the number of times that each orientation occurs in each library from the mapping information. The probability that a particular read from a particular library has a particular orientation is then modeled as that orientations empirical frequency in the library (this can be overridden with user-specified values for the probabilities). The likelihood $P_{orientation}(r_i|S)$ is then the empirical frequency of the observed orientation of the read r_i in the library from which r_i belongs.

After combining these two independent probabilities we are left with the total placement score $P_{placement}(R|S)$ for a given read. Below, we use this when calculating the probability that an assembly is correct given the reads, as well as the overall total ALE score. We also use it to calculate per-base placement scores at particular positions in the assembly. The placement sub-score at a particular position is given as the geometric mean of $P(r_j|S)$ of all r_j covering that specific position,

$$\left[\prod_{R} P(R|S)\right]^{1/N} \tag{2.9}$$

where the product is over all reads R covering the given position, and N is the number of such reads.

2.2.2 Insert sub-score

The insert likelihood, $P_{insert}(r_i|S)$, is determined by first observing all insert lengths from all mappings of all reads and calculating the population mean, μ, and standard deviation, σ^2 of these lengths (the mean and standard deviation can also be set by the user, if they are known). This step only needs to be done once. Once completed, we calculate the insert likelihood for each read r_i by assuming that the corresponding observed insert length Li is distributed normally with this mean and variance, so that

$$P_{insert}(r_i|S) = Normal\left(L_i; \mu, \sigma^2\right) \tag{2.10}$$

In this expression, the insert length L_i is computed from the read r_i and its mapping to the assembly S. Similar to the placement score we can calculate the geometric mean of insert scores at a given position to come up with the insert sub-score. This can be useful for determining areas of constriction or expansion within a proposed assembly.

2.2.3 Depth sub-score

$P_{coverage}(R|S)$ describes how well the depth at each location agrees with the depth that we would expect given the GC content at that location (which is ideally Poisson-distributed (Lander and Waterman 1988)).

For each read, the GC content is the proportion of bases that are either G or C. Modern sequencers and library preparation techniques can bias GC-rich areas of a genome (Aird et al. 2011) This bias affects the observed depth of reads mapping onto specific areas of an assembly. To correct for this bias we first calculate for each of the following 100 ranges of GC content over the average read length, 0% to 1%, 1% to 2%, . . ., 99% to 100%, the average observed depth for positions in each contig in the assembly with a GC content in this range. Call $\mu_{depth(X_i)}$ the observed average depth of all reads with a GC content falling in the same range as the GC content percentage X_i. We set the minimum expected depth to be 10, discounting regions of exceptionally low average depth.

We model the depths to be Poisson distributed about a mean drawn from a Gamma distribution centered at the expected depth for that position given its GC content. This models the dependence of the expected depth on more than just the GC content at that position, such as the presence of hard stops, and the GC content at nearby positions. It results in an infinite mixture of Poissons that is equivalent to a Negative Binomial distribution. For simplicity and computational convenience, we make an independence assumption when computing this component. This causes the expected coverage at a location to depend only upon the GC content at that position, and not the GC content at nearby positions.

Then, at any given position the depth sub-score is

$$\begin{aligned} &P_{depth}\left(d_j|S, X_i\right) \\ &= Poisson\left(d_j; Y_i]\right), Y_i \sim Gamma(\max(10, \mu_{depth(X_i)}), 1) \\ &= NegBinom(d_j; \max(10, \mu_{depth(X_i)}), 1/2) \end{aligned} \tag{2.11}$$

where the depth is d_i and where the GC content percentage X_i is averaged across all reads that map (in the placement step) to that position.

2.2.4 k-mer sub-score

$P_{kmer}(S) \propto P(S)$, the k-mer sub-score, describes the likelihood of the assembly S, in the absence of any read information. Within this prior probability distribution, we encode the belief that within a single genome, each k-mer (a permutation of k base pairs, where k is a fixed user defined number initially set to 4) has a unique k-mer frequency. This is the frequency with which the k-mer appears in the genome. The 4^k dimensional vector giving this frequency for each k-mer is conserved across a genome and can help determine if two different genomes have been mistakenly combined (Teeling et al. 2004; Woyke et al. 2006). Let K be the set of all possible unique k-mers, so $|K| = 4^k$, and for each i in K let n_i be the number of times this k-mer appears in a contig in the assembly. Then, the frequency f_i of a particular k-mer i within a contig is

$$f_i = \frac{n_i}{\sum_{j \in K} n_j} \tag{2.12}$$

The k-mer score is the product of this frequency over each k-mer appearing in each contig of the assembly S, which can be written as

$$P_{kmer}(S) = \prod_{i \in K} f_i^{n_i} \tag{2.13}$$

This is equivalent to assuming each k-mer in the assembly is drawn independently with identical distributions from a multinomial distribution with probabilities empirically estimated from the assembly.

The k-mer sub-score of a base at any given position in the assembly is the (geometric) average of $P_{kmer}(S)$ of all k-mers that cover that position. In calculating this average, the very first base in the genome only has one contributing k-mer, the second has two, up to k contributing k-mers after $k - 1$ bases.

2.3 Approximating Z

Bayes rule tells us that the probability that the assembly S is correct is

$$P(S|R) = \frac{P(R|S)P(S)}{Z} \tag{2.14}$$

where Z is a proportionality constant that ensures that $P(S|R)$ is a probability distribution, where Z is found by summing over all possible assemblies S',

$$Z = \sum_{S'} P(R|S')P(S') \tag{2.15}$$

Z cannot be explicitly computed because the space of all possible assemblies is far too large (4^L where L is the length of the assembly).

Instead we compute below an approximation $\hat{Z}$ to Z. This provides an approximation to $P(S|R)$,

$$P(S|R) \approx \frac{P(R|S)P(S)}{\hat{Z}} \tag{2.16}$$

We can compare two assemblies generated from the same library of reads without calculating Z, the denominator in our Bayesian likelihood framework, because it cancels when taking the ratio of the likelihoods of the two assemblies. To determine a total ALE score for a single assembly, however, we must calculate or approximate Z. Our goal in approximating Z is to use a quantity that does not depend on the assembly S (or only depends weakly through some empirically estimated quantities included as parameters in the overarching statistical model), and is approximately of the same order of magnitude as the exact value of Z. In this section, we refer to our approximate Z as , and define it as a product of terms,

$$\hat{Z} = \hat{Z}_{placement}\hat{Z}_{insert}\hat{Z}_{depth}\hat{Z}_{kmer}. \tag{2.17}$$

We define each term in this product separately.

2.3.1 Aprroximating $Z_{placement}$

$\hat{Z}_{placement}$ is defined as

$$\hat{Z}_{placement} = \prod_{r \in R} \hat{Z}_{placement}(r|S). \tag{2.18}$$

In this expression R is the set of reads actually observed, r is one read in this set of reads, and $\hat{Z}_{placement}(r|S)$ is defined as

$$\begin{aligned} \hat{Z}_{placement}(r|S) &= \mathbb{E}\left[P_{placement}(r')|S\right] \\ &= \textstyle\sum_{r' \in R'} \left[P_{placement}(r'|S)\right]^2 \\ &= \textstyle\sum_{matches} \sum_{orientations} \left(P_{matches}(r'|S) P_{orientations}(r'|S)\right)^2 \end{aligned} \tag{2.19}$$

In this expression R' is the set of all possible reads of the length given by r, and we sum over all possible matches and orientations, which is analogous to summing over all possible reads. Although S appears in this expression, its value does not depend on S because of permutation symmetry. This symmetry allows us to calculate this expression analytically, without enumerating over R'. In addition to its lack of dependence of S and its ease of computation, this choice for $P_{placement}(r)$ is motivated by the belief that this quantity scales roughly like

$$P_{placement}(r) = \sum_{S'} P_{placement}(r|S')P(S'), \tag{2.20}$$

which is a quantity identical in form to Z, but restricted to the placement probability of a particular read.

2.3.2 Aprroximating Z_{insert}

We define

$$\begin{aligned} \hat{Z}_{placement}(r|S) &= \mathbb{E}\left[P_{insert}(r')|S\right] \\ &= \textstyle\sum_{r' \in R'} \left[P_{insert}(r'|S)\right]^2 \\ &= \textstyle\sum_{insert} \left(P_{insert}(r'|S)\right)^2 \end{aligned} \tag{2.21}$$

Where again, in this expression R' is the set of all possible reads of the length given by r, and we sum over all possible insert lengths.

2.3.3 Aprroximating Z_{depth}

We define $\hat{Z}_{depth}$ as,

$$\hat{Z}_{depth} = \prod_{base_i \in S} \hat{Z}_{depth}(X_i|S) \tag{2.22}$$

where

$$\hat{Z}_{depth}(X_i|S) = \mathbb{E}\left[P_{depth}(X_i|S)|S\right] = \sum_{d=0}^{\infty} \left(P_{depth}(d|X_i, S)\right)^2, \tag{2.23}$$

where P_{depth} is defined as before. Although S appears in this expression, its value does not depend on S. We can calculate this expression analytically using a hyper-geometric function, see implementation.

2.3.4 Aprroximating Z_{kmer}

We define $\hat{Z}_{kmer}$ as the expected kmer score,

$$\hat{Z}_{kmer} = \mathbb{E}\left[P_{kmer}(S)\right] = \left(\sum_{i \in K} f_i^2\right)^N \tag{2.24}$$

where f_i, K and n_i are defined as before. This method finds the expected k-mer score for a uniform random k-mer and applies that score N times. Although $\hat{Z}_{kmer}$ depends on S through the empirically determined f_i, which may be undesirable, this dependence follows naturally from our statistical model because the fi are considered parameters, which are estimated from data and then treated as known by the model.

The preceding calculations allow us to approximate Z and find an absolute likelihood score for a given assembly without comparing it to another with the same library and alignment.

2.4 Relationship of the difference of total ALE scores to probability of correctness

Here we derive an expression described in the Results section for the difference of two total ALE scores in terms of the probability that an assembly is correct. Suppose we have two assemblies, S_1 and S_2. Call A_1 the total ALE score of the first assembly, and A_2 the total ALE score of the second assembly both generated from the same set of reads R. The difference of these scores is then

$$\begin{aligned}
&A_1 - A_2 \\
&= \log\left(P\left(R|S_1\right)\right) + \log\left(P\left(S_1\right)\right) - \log\left(\hat{Z}\right) + \log\left(P\left(R|S_2\right)\right) + \log\left(P\left(S_2\right)\right) + \log\left(\hat{Z}\right) \\
&\approx \log\left(P\left(R|S_1\right)\right) + \log\left(P\left(S_1\right)\right) - \log\left(Z\right) + \log\left(P\left(R|S_2\right)\right) + \log\left(P\left(S_2\right)\right) + \log\left(Z\right) \\
&= \log\left(\frac{P(R|S_1)P(S_1)}{P(R|S_2)P(S_2)}\right)
\end{aligned} \tag{2.25}$$

2.5 Thresholding the total ALE score

In the plotting program distributed with ALE we use a thresholding algorithm to highlight potential areas of poor assembly quality using the per-base ALE scores. We do this by averaging scores within windows, allowing for the discovery of large errors in the assembly while smoothing out the noise. By the Central Limit Theorem, when we average many independent and identically distributed random variables the result is approximately normally distributed. This allows us to create a threshold for which to delineate good scores from bad and pinpoint problematic regions. This is represented by a solid black line in the figures labeled 5σ. This line is calculated by assuming that the individual scores at each position in the assembly are drawn from a mixture of two normal distributions: one for high accuracy and another for low accuracy. We use maximum likelihood to determine the mean and variance of the two underlying distributions. The threshold is set as five standard deviations from the mean of the high accuracy distribution. This allows us to readily find areas of inaccuracy that are unlikely to be drawn from an accurate region. Five standard deviations corresponds to 1 false positive in 2 million positions if the joint normal distribution assumptions hold. The number of standard deviations at which the black line is drawn can be set from the command line by the user.

A black bar is drawn on the plot if the likelihood falls below the threshold at a significant fraction of the positions in any contiguous region with a given length (this fraction and length are user defined, and are initially set to 0.01% and 1000bp respectively) see figure XXX. These red bars correspond to regions of potential inaccuracy in the assembly that should be examined further. The plotter outputs these regions in a tab delineated text file for easy input into genome viewing software programs like IGV.

2.6 Influence of alignment input

ALE takes as input a proposed assembly and a SAM/BAM (Li 2009) file of alignments of reads onto this proposed assembly. This allows ALE to calculate the probability of observing the assembly given the reads. ALE assumes that this mapping will include, if not all possible mappings, at least the ”best” mapping for each read in the library (if such a mapping exists). For assemblies with many repeat regions (¿100) or libraries with large insert sizes, this can be difficult to obtain due to the bias introduced using default parameters of standard aligners. While an extensive review of alignment packages and their optimization is beyond the scope of this paper a review can be found in (Li and Homer 2010). If an assembly has many repeats and the aligner bias causes the reporting of reads only mapping to a fraction of possible regions, then ALE will see the unmapped regions as having 0 depth (no supporting reads) which will result in artificially low depth sub-scores. The robustness of ALE will still allow for comparison between assemblies with similar biases, but should be taken into account if the input to ALE is biased for only certain assemblies. To avoid this bias some mappers must be explicitly forced to search for all possible placements (-a in bowtie).

In summary, ALE determines the likelihood of an assembly given the reads and an accurate, unbiased alignment of those reads onto the assembly, without which the model assumptions are violated. These preconditions are usually met except for certain pathological genomes, and even in these cases can be readily corrected for by changing the parameters of the aligner used to make ALEs input.

CHAPTER 3

ALE RESULTS

CHAPTER 4

ALE IMPLEMENTATION

Part II

EPI: Expected Parallel Improvement

CHAPTER 5

EPI INTRODUCTION

CHAPTER 6

EPI METHODS

CHAPTER 7

EPI RESULTS

CHAPTER 8

EPI IMPLEMENTATION

Part III

Velvetrope

CHAPTER 9

VELVETROPE INTRODUCTION

CHAPTER 10

VELVETROPE METHODS

CHAPTER 11

VELVETROPE RESULTS

CHAPTER 12

VELVETROPE IMPLEMENTATION

APPENDIX A

CHAPTER 1 OF APPENDIX

Appendix chapter 1 text goes here