Statistical Models for Genome Assembly and Analysis

Atif Rahman

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-186
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-186.html

August 12, 2015

Statistical Models for Genome Assembly and Analysis

by

Atif Hasan Rahman

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

and the Designated Emphasis

in

Computational and Genomic Biology

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Lior Pachter, Chair
Professor Daniel Rokhsar
Professor Yun S. Song

Summer 2015

Statistical Models for Genome Assembly and Analysis

Abstract

Statistical Models for Genome Assembly and Analysis

by

Atif Hasan Rahman

Doctor of Philosophy in Computer Science
and the Designated Emphasis in
Computational and Genomic Biology

University of California, Berkeley

Professor Lior Pachter, Chair

Genome assembly is the process of merging fragments of DNA sequences produced by shotgun sequencing in order to reconstruct the original genome. It is complicated by repeated regions in genomes, sequencing errors, and experimental biases. Here we focus on our efforts to confront some of the challenges in genome assembly and analysis of genomes to find regions associated with phenotypes using statistical models.

Assembly algorithms have been extensively benchmarked using simulated data so that results can be compared to ground truth. However, in *de novo* assembly, only crude metrics such as contig number and size are typically used to evaluate assembly quality. We present CGAL, a novel likelihood-based approach to assembly assessment in the absence of a ground truth. We show that likelihood is more accurate than other metrics currently used for evaluating assemblies, and describe its application to the optimization and comparison of assembly algorithms.

We then extend this to develop a method for "scaffolding" i.e. linking contigs using read pairs based on optimizing assembly likelihood. It uses generative models to approximate whether joining contigs would result in an increase in assembly likelihood. The methods are grounded in a rigorous statistical model yet proper approximations make the implementation named SWALO efficient and applicable to practical datasets. We analyze SWALO on real and simulated datasets used previously to evaluate other scaffolding methods and find that it consistently outperforms all other scaffolders.

Finally, we focus on the problem of analyzing genomic data to associate regions in the genome to traits or diseases. We present an alignment free method for association studies that is based on counting k-mers in sequencing read, testing for associations directly between k-mers and the trait of interest, and local assembly of the statistically significant k-mers to identify sequence differences. Results with simulated data and an analysis of the 1000 genomes data provide a proof of principle for the approach. In a pairwise comparison of the Toscani in Italia (TSI) and the Yoruba in Ibadan, Nigeria (YRI) populations we find

that sequences identified by our method largely agree with results obtained using standard GWAS based on variant calling from mapped reads. However unlike standard GWAS, we find that our method identifies associations with structural variations and sites not present in the reference genome.

We also analyze the data from the Bengali from Bangladesh (BEB) population to explore possible genetic basis of high rate of mortality due to cardiovascular diseases (CVD) among South Asians and find significant differences in frequencies of a number of non-synonymous variants in genes linked to CVDs between BEB and TSI samples, including the site rs1042034, which has been associated with higher risk of CVDs previously and the nearby rs676210 in the *Apolipoprotein B (ApoB)* gene.

To my grandfather Dr. Arma Awwal and my friend Dr. Tashdid Rezwan Mugdho

They would have been among the first persons to consult on possible implications of our findings regarding cardiovascular diseases.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost I thank my advisor Lior Pachter whose knowledge and understanding of the many disciplines involved in computational biology is quite remarkable. I am grateful to him for encouraging me to learn genetics and statistics, providing me the freedom to explore many topics and to pursue my interests while offering guidance and advices throughout my years at Berkeley. I realized as years progressed that he shares many of the values I was raised with, albeit more staunchly.

I am also thankful to other members of my dissertation committee – Dan Rokhsar for sharing his insights on genome assembly and Yun Song for keeping an eye on my progress and providing ideas. I would like to thank Richard Karp and David Tse for their valuable feedback as members of the quals committee and Mike Eisen for contributing ideas and providing us computational resources. I would also like to thank Masud Hasan for introducing me to bioinformatics and computational biology and all other teachers I have had for providing me with an education, which was almost free of cost in Bangladesh, good enough to survive at UC Berkeley.

A wonderful aspect of being a student of Lior is having an amazing group of people with diverse backgrounds as labmates. I thank everyone I have overlapped with especially Meromit Singer for being there to answer queries on research, classes to take and academic requirements; Aaron Kleinman, Adam Roberts, Harold Pimentel, Nicolas Bray, Natth Bejraburnin for many helpful discussions; Harold for also managing our servers; Shannon Hateley, Lorian Schaeffer and Akshay Tambe for sharing their knowledge of biology and Shannon McCurdy for much needed encouragement during writing of this dissertation.

I also thank Ma'ayan Bresler, Caroline Uhler, Luqman Hodgkinson, Anand Bhaskar, Andrew Chan and other computational biology students for the exchange of ideas and discussions and Mosharaf Chowdhury with whom I navigated through the early days at Berkeley. I would also like to thank Brian McClendon and Xuan Quach who organized computational biology retreats and who along with La Shana Porlaris, Saheli Datta, Audrey Sillers and others helped me with administrative requirements.

Finally, I am grateful to my parents, my sisters and my brother-in-law for their continuous support, and encouraging me to pursue my interests. I also thank my friends from Bangladesh and from all over the world made through my Fulbright Fellowship and in the bay area, with whom I traveled, hung out, talked about Manchester United football matches, Bangladesh cricket matches, David Attenborough documentaries, or anything else.

Chapter 1

Introduction

In the last decade, research in biology has gone through a transformation with emergence of next-generation sequencing technologies. The low cost and high throughput nature of these technologies have led to development of various assays to explore many aspects of interest in biology. In this thesis we address the problem of assembling genomes which is essential for running most of these assays and how to find segments in genomes associated with phenotypes *i.e.* observable characteristics or traits.

The genome of an organism refers to all of its genetic material. It contains the information needed for development and functioning of an organism and is passed on from one generation to the next. It is packaged in one or multiple chromosomes and is encoded in DNA (deoxyribonucleic acid) or RNA (ribonucleic acid) in some viruses. DNA in turn can be thought of as a string of four types of nucleotides – adenine (A), cytosine (C), guanine (G) and thymine (T).

Genome includes protein coding genes - segments in DNA that are first transcribed into RNA and then translated into proteins which perform most of the cellular functions in organisms. Genomes also contain non-coding sequencing some of which help regulate when genes are expressed.

1.1 Genome assembly

Sequencing the genome of an organism *i.e.* determining the sequence of nucleotides that make up the genome is a prerequisite for performing various kinds of experiments to study an organism and is fundamental to understanding how different organisms and individuals relate to each other. However, genomes can be tens of thousands to billions of nucleotides long and sequencing instruments typically cannot determine the entire sequence.

Due to this there are two approaches to genome assembly. In *reference guided* or *assisted assembly*, the genome of a related organism is used to aid the genome assembly of the organism being sequenced. Whereas in *de novo* sequencing no such genome is available or used. In *de novo* sequencing a technique known as *shotgun sequencing* is used which leads to the computational problem of *genome assembly*.

Genome sequencing starts with many copies of the genome (Figure 1.1). Fragments are then generated by shearing these at random locations. This is usually followed by a size selection step to produce libraries of fragments with approximately a known size. Sequencing machines then generate the sequence of nucleotides from one end of the fragment known as *single-end reads* or simply *reads*. Most technologies can also determine sequences from both ends of the fragment. These are called *paired-end reads* or *mate-pair reads* when circularized DNA fragments are used. We use the term *read pair* to refer to either paired-end or mate-pair reads.

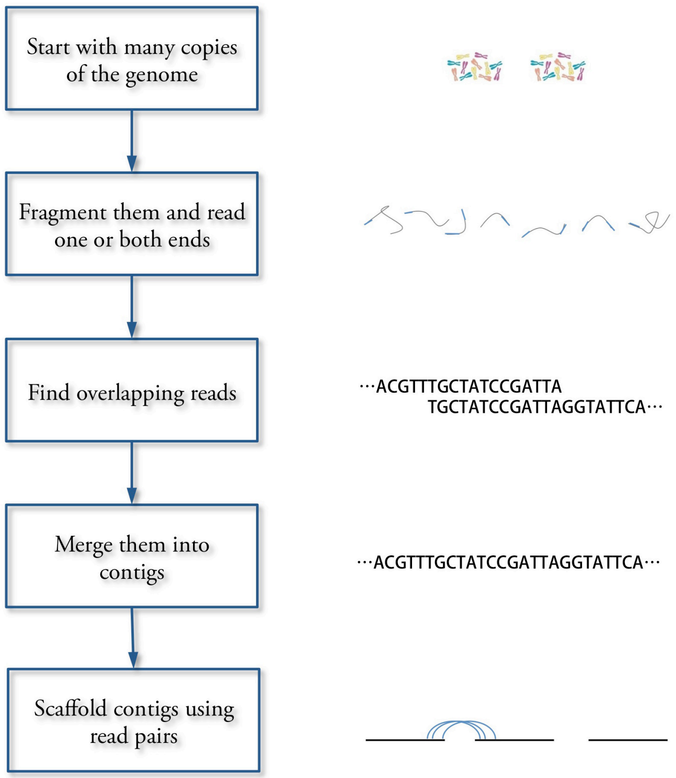

Start with many copies of the genome
Fragment them and read one or both ends
Find overlapping reads
···ACGTTTGCTATCCGATTA
TGCTATCCGATTAGGTATTCA···
Merge them into contigs
···ACGTTTGCTATCCGATTAGGTATTCA···
Scaffold contigs using read pairs

Figure 1.1: **Genome sequencing.** Genome sequencing starts with many copies of the genome which are sheared in random locations. One or both ends of these fragments are then read. Overlapping reads are assembled into contigs. Finally, contigs are oriented and linked together into scaffolds using read pairs.

The computational task of assembling or stitching together the overlapping reads generated is complicated by three issues – errors by sequencing instruments, non-uniform coverage of genome due to sequencing biases and most importantly presence of identical or near identical regions in the genome called *repeats*.

Under certain conditions depending on structure and lengths of repeats and read lengths, it may not be possible to reconstruct the genome uniquely [155]. The conditions are explained by Bresler *et al.* [11] and illustrated in Figure 1.2. If the genome contains repeats with three copies (shown in grey) or interleaved repeats (two sets of interleaving repeats shown in grey and black) with minimum length of L and read length is not greater than $L+1$ then the target genome cannot be determined uniquely.

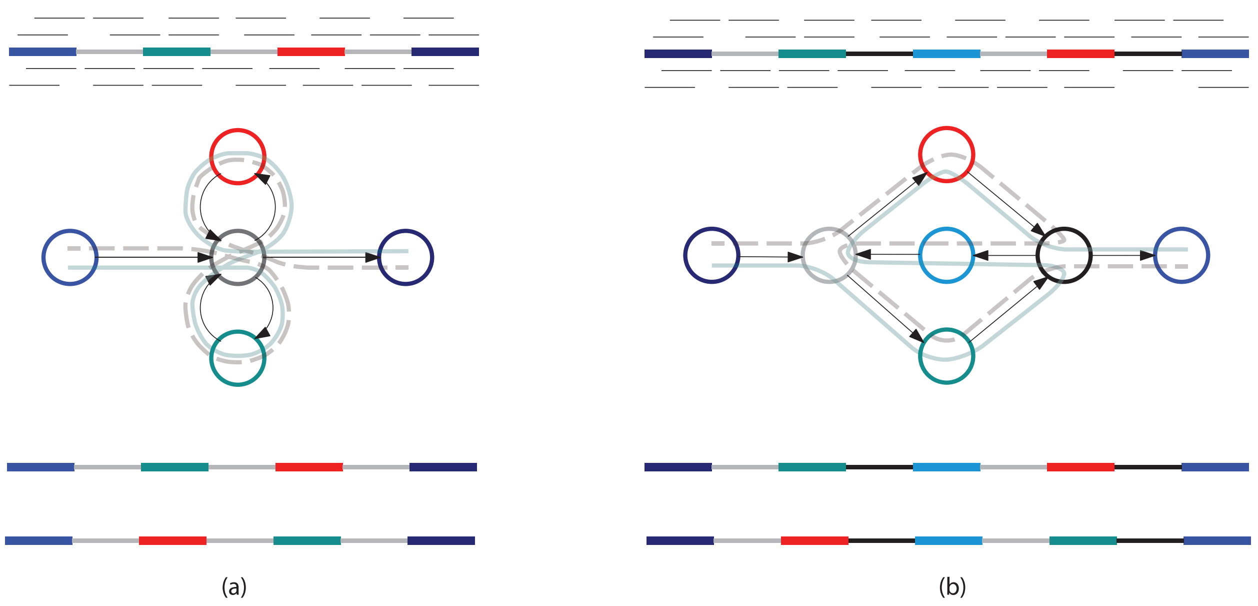


Figure 1.2: **Difficulties in genome assembly due to repeats.** Figure shows complications in assembly due to (a) triple repeats and (b) interleaved repeats. If there are no reads spanning any of the repeats, the reads can be equally well explained by both paths leading to two possible assemblies of the genome

The stochastic nature of the read generation process also complicates the assembly process. Lander and Waterman provided guidelines on the number of reads to be generated to ensure that the entire genome has been sampled with high probability assuming the start sites of reads are Poisson distributed [78]. In reality, all regions of the genome are not sampled uniformly due to sequencing biases [149] leading to unsampled regions and making estimation of number of copies of repeats difficult. Furthermore, sequencing errors complicates distinguishing between reads that actually overlap and reads that are from non-identical repeats.

Genome assembly has been shown to be NP-hard (computationally intractable) in a number of settings [100, 112] while Nagarajan and Pop explored how the interplay among complexity, read length, coverage and other parameters change the complexity of the problem [112]. Due to the theoretical hardness of the problem and practical issues, genome assembly is commonly done in two major steps.

In the first step overlapping reads are assembled into contiguous sequences commonly known as *contigs*. But most genomes contain repetitive regions longer than reads which cannot be resolved with single-end reads. Moreover, as explained earlier, some regions in the genome may not be read due to stochastic nature of fragment generation process or biases in the process.

In the second step, the contigs can be linked together into *scaffolds* if read pairs are available. This step of orienting and ordering contigs is known as *scaffolding*. Scaffolding increases the sizes of assembled sequences aiding downstream analysis. The scaffolding step may also estimate gaps between contigs the sequences within which can be determined in a gap-filling step.

1.2 Sequencing technologies

Although the general approach to genome assembly has largely remained unchanged, the technologies used for sequencing and the methods used for assembling the data have evolved over the years. Here we review major breakthroughs in sequencing technologies and changes in the nature of data generated. There are a number of review articles that provide more details on sequencing technologies [146, 93, 95, 102, 44, 121, 105, 94, 134].

The first known method for determining DNA sequences was developed by Ray Wu in the early 1970s [163, 164]. In 1977, Frederick Sanger developed the chain-termination method for sequencing [142]. This technique known as *Sanger sequencing* was faster and more efficient than earlier methods and laid the foundation for later sequencing methods. Gilbert and Maxam had also developed a method for sequencing and reported a sequence of 24 basepairs [98].

The basic Sanger method was later automated and refined to produce reads with lengths up to 2000bp and producing 96 reads per run [113, 95]. The human genome project [79, 157] was completed in 2003 using Sanger sequencing and despite the advances made in sequencing, it took more than a decade and cost around $1 billion [134].

Sequencing went through a revolution in the past decade due to the emergence of a set of technologies collectively known as *next generation sequencing* (NGS) technologies. These technologies speed up sequencing and increase the throughput by parallelizing the process while reducing the cost drastically at the same time.

The first next generation sequencing technology to emerge was the pyrosequencing method by 454 Life Sciences (now Roche) [96] in 2005 followed by the Illumina/Solexa sequencing in 2006 [52] and the Sequencing by Oligo Ligation Detection (SOLiD) by Applied Biosystems (now Life Technologies) in 2007 [156]. In 2010, Ion Torrent (now Life Technologies) announced the Ion semiconductor sequencing method [137].

Figure 1.3 shows how the sequencing cost has decreased over the years while Table 1.1 compares several characteristics of various technologies that are relevant to the computational

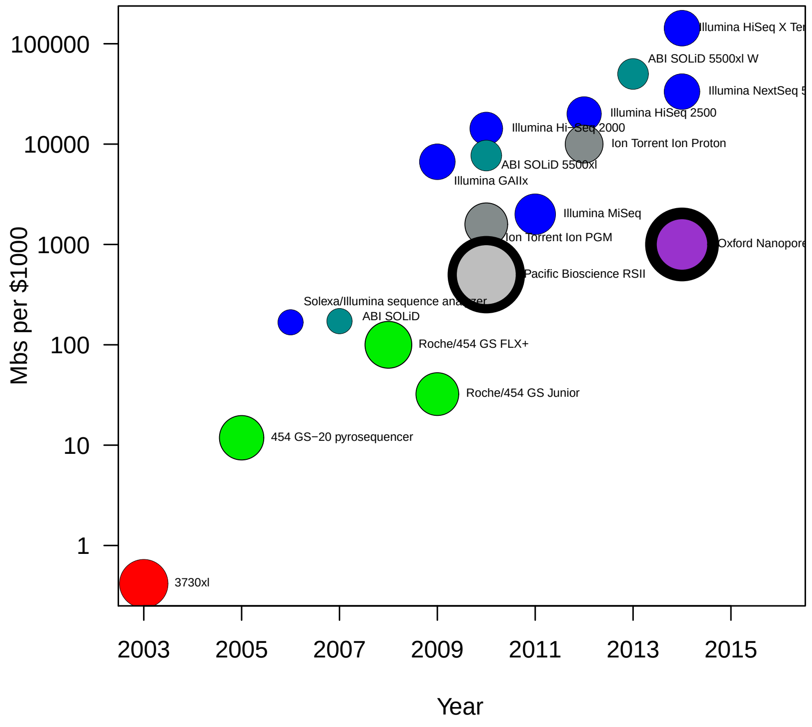


Figure 1.3: **Evolution of sequencing technologies.** Figure shows how the amount of sequences generated per $1000 has increased over the years. Circles are color coded by technology. The radii of the circles are proportional to log of read lengths and the border widths denote error rates.

task of assembly. The values used are approximate ones collected from [95, 130, 89, 90, 113] and other online sources.

The reduction in cost and increase in throughput brought by NGS technologies have spurred sequencing of genomes of many species across the tree of life [120] and led to sequencing of approximately 2500 individuals from 26 populations as part of the 1000 genomes project [4]. However, the shorter read lengths and quantity of data generated by NGS technologies have posed additional computational challenges.

More recently, a third generation of technologies that include the single molecule real time (SMRT) sequencing technology by Pacific Biosciences (PacBio) [37] and nanpore sequencing by Oxford Nanopore [10] have emerged. In addition, Moleculo technology and GemCode technology by 10X Genomics generate long range information from short reads using library preparation methods. But third generation technologies have not been widely adopted due to higher cost, higher sequencing error rate, lower throughput compared to NGS among other reasons and most of the sequencing is still being performed using NGS or a combination of

Table 1.1: **Comparison of several features of sequencing technologies relevant to genome assembly.**

Technology	Read length(bp)	Error rate	Reads per run	Time per run
Sanger	900 (Up to 2000)	0.1%	96	20 mins-3 hrs
Roche/454	400-700	2%	1 million	1 day
Illumina/Solexa	35-300	1%	Up to 6 billion	1-11 days
ABI/SOLiD	35-75	1%	Up to 1.4 billion	1-2 weeks
Ion Torrent	200-400	2%	Up to 80 million	2-4 hours
Pacific Biosciences	14000	13%	50,000	30 mins-4 hrs
Oxford Nanopore	6000 (Up to any)	18%	73,000	2 days

NGS and third generation sequencing.

1.3 Methods for genome assembly

The computational methods for genome assembly have had to adapt to evolving sequencing technologies. In this section, we review prior theoretical work and practical methods for contig generation, scaffolding and assessing genome assemblies. More detailed overview of approaches to genome assembly, methods based on these and other issues are available in [129, 128, 104, 113]

Contig assembly

In the early days, Sanger sequencing was used for generating reads and computer scientists formulated genome assembly as the shortest common superstring (SCS) problem – the problem of finding the shortest string that contains all reads as substrings. Although the problem is NP-hard, the greedy algorithm of iteratively joining reads with most overlap performs well in practice [41]. However, in reality genomes contain repetitive regions that are handled improperly by the SCS formulation [109, 71]. Early assemblers such as phrap [47] and TIGR [152] as well as a more recent one, VCAKE [66] were based on the greedy algorithm with heuristics used to detect repetitive regions.

Myers and Kececioglu introduced a graph theoretical formulation of the problem that gave rise to the overlap-layout-consensus (OLC) paradigm. In this formulation there is a vertex for each read and an edge between two vertices if the corresponding reads overlap. Then the assembly problem is related to finding a walk on the graph visiting all the vertices. The Celera assembler was based on this paradigm [107]. Later Myers proposed the *string graph* for genome assembly which is constructed by transitive reduction on the overlap graph and estimating the number of times each vertex is to be visited [108].

A different graph theoretic paradigm was proposed by Pevzner based on the *de Bruijn* graph [123, 124, 23]. In the context of genome assembly, de Bruijn graph contains a vertex for each k-mer (a continuous string of length k) present in the reads and there is a directed edge between two vertices if suffix of length $k-1$ of one is the prefix of length $k-1$ of the other. The assembly problem is then to find an Eulerian path *i.e.* a path that visits every edge exactly once. An implementation of this approach resulted in the EULER assembler [124].

With the emergence of NGS technologies, assembly algorithms were faced with new challenges due to large volume of data, short read lengths, and high error rates. To avoid finding overlaps between millions of reads, de Bruijn graph based approaches started to be more commonly used. A number of de Bruijn graph based assemblers, such as Velvet [167], ABySS [147], Euler-sr [15], Allpaths [14, 46], SOAPdenovo [85] have been developed to assemble NGS reads. SGA is however an overlap graph based assembler for NGS data which uses efficient string indexing data structures for overlap computation [148].

Although assembly algorithms adapted to tackle large volume of data, there have not been much effort to take advantage of quantity of data through a rigorous statistical model. Myers proposed a maximum likelihood reconstruction that is finding an assembly that maximizes the probability of observing the set of reads [109], and Medvedev and Brudno gave an algorithm for maximum likelihood genome assembly based on a bidirected network flow-based algorithm [99]. But there has not been a practical assembler of real NGS data based on a maximum likelihood approach.

Scaffolding

In all of the paradigms discussed above, if the reads are not long enough to resolve the repeat structure of the genome [155, 11] or if parts of the genome were not sequenced, the genome cannot be reconstructed uniquely based on single-end reads. Read pair or other sources of information such as genetic and optical maps [16, 114] are then used to orient and order the contigs. While there have been some efforts to integrate read pair information into contig assembly [101, 125, 13], most commonly scaffolding is done as a separate step after contigs have been generated using single-end reads.

Due to short reads generated by next-generation sequencing technologies, the scaffolding step has become of increased importance and as such a scaffolding module is built into most assemblers [167, 147, 14, 85, 17]. In addition, many stand alone scaffolders have been developed [76, 28, 7, 34, 42, 139, 48]. Typically scaffolders construct a graph with a vertex for each contig and edges representing read pairs linking contigs and attempt to maximize number of linking reads or minimize paired read violations.

Several formulations of this problem is known to be NP-hard [63, 73, 127] and scaffolding algorithms rely on heuristics. There are a number of scaffolding tools that are based on the greedy heuristic [7]. There are also other approaches – SOPRA uses statistical optimization [28], MIP employs mixed integer programs [139] while SCARPA and Opera are based on fixed parameter tractable algorithms [34, 42].

Evaluating genome assemblies

An important issue regardless of the sequencing technology used for generating the data is assessing the accuracy and completeness of an assembly [6, 113]. Assembly and scaffolding algorithms rely on heuristics due computational intractability of the problems [99, 112, 63, 73, 127] and complications caused by sequencing errors, experimental biases and the volume of data that must be processed. As a result assemblies produced by existing methods tend to differ from each other. In addition, assemblies generated by the same assembler often vary with parameter values used for assembly.

In recent years, there have been two major initiatives to evaluate assembly methods, the genome assembly gold-standard evaluation (GAGE) [141] and Assemblathon competitions [35, 9]. The results confirm the variability in the assemblies generated and suggest no method can be termed "the best", and thus highlight the need for methods to assess genome assemblies.

In *de novo* assembly, when there is no "ground truth", there has been a focus on contiguity due to advantages in downstream analysis and the correctness of the reconstructed sequences has been ignored typically. The most commonly used statistic is the *N50 scaffold or contig size* - which is the maximum contig (scaffold) length such that at least half the total assembly is contained in contigs (scaffolds) of length greater than or equal to that length. The numbers of contigs and scaffolds have also been used to assess assemblies.

The practice of assessing assemblies using N50 sizes has led some of the assemblers to aggressively stitch pieces together and omit hard to assemble regions at the expense of incorrect and incomplete assemblies. In 2005, even when genomes were assembled using Sanger sequencing data, Salzberg and Yorke found that there were mis-assemblies in most draft genomes they examined and questioned the emphasis on statistics based on size [140]. Alkan *et al.* reported that an assembly of the human genome using NGS reads is considerably shorter than the reference [1].

To address the problem of mis-assemblies, Phillippy *et al.* presented a software called amosvalidate [126] that identifies features that might arise due to mis-assemblies and uses them to detect suspicious regions; however, it does not have high specificity and has not been widely adopted. Narzisi *et al.* used these features and introduced feature response curves to rank assemblies [116, 159]. In feature response curves, total size of contigs with a mis-assembly feature threshold are plotted against different thresholds. This does not however produce a single value that can be optimized during assembly and does not incorporate amount of genome covered in the assembly.

A different approach to evaluate genome assemblies is to use experimental data such as optical map data, transcriptome data, chromosome organization data [114, 110, 170, 171]. Each has its advantages and disadvantages and the experiments are generally expensive and time consuming meaning thorough validations of assemblies using experimental approaches are performed rarely.

1.4 Genome analysis

Sequencing of the genome of a species or an individual is an important and often essential first step for subsequent analysis. Genomic analysis includes annotation of protein coding genes and other regulatory elements, understanding expression and regulation of genes, identification of variations in genomes and mapping their associations to phenotypes as well as comparison of genomes of multiple species and exploring their evolutionary relationships through phylogenetic analysis.

Although next generation sequencing technologies were initially intended for sequencing genomes, scientists have taken advantage of low cost and high throughput of NGS technologies especially Ilumina/Solexa sequencing and reduced many other experiments in biology to sequencing. These so called "*-Seq" assays probe diverse aspects such as protein-DNA binding (ChIP-Seq [67, 103]), RNA transcript abundances (RNA-Seq [106]), RNA structure (dsRNA-Seq [169], SHAPE-Seq [91, 5], DMS-Seq [138]), translation of RNA (ribosome profiling [64]), chromatin structure, methylation and other epigenetic features (DNAse-Seq [25], BS-Seq [88], Hi-C [86]). Many of the computational methods to analyze the data map reads to a reference genome requiring prior sequencing of the genome.

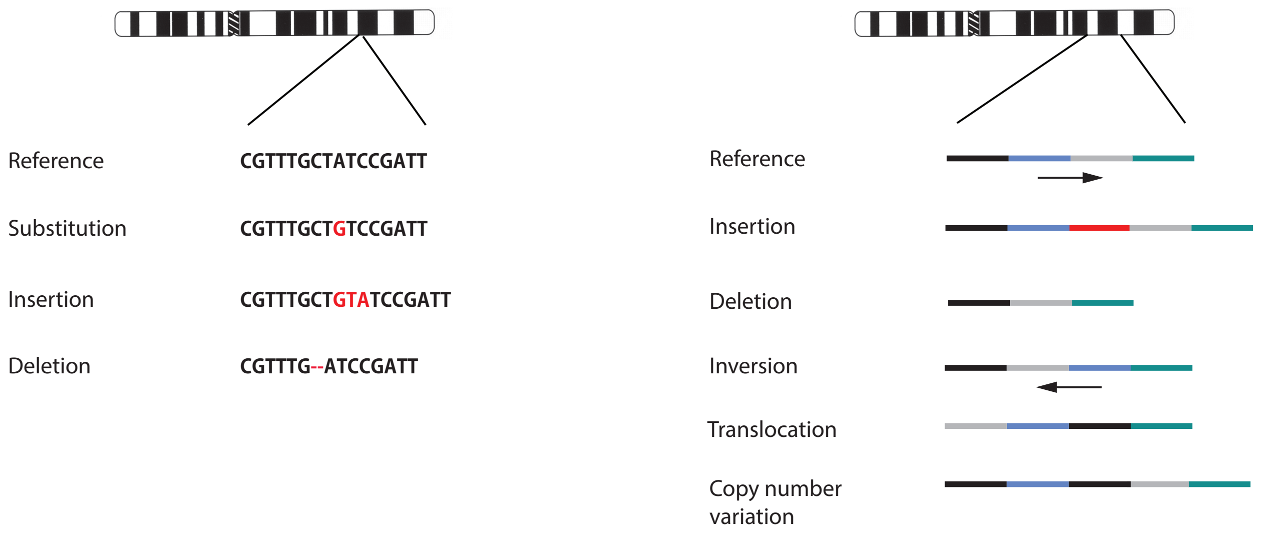
However, the difficulties in assembling genomes using NGS data motivated us to explore methods for genomic analysis that do not require a reference genome. Here we focus on a reference-free method for association mapping from NGS sequencing reads with only local assembly.

Association mapping

Although genomes of individuals of a species are quite similar, there are variations in the sequence of DNA across individuals. Genomic variations, illustrated in Figure 1.4, are broadly of two types – *sequence variations* and *structural variations*. Sequence variations include mutation at a single base from one base into another called substitutions, and insertion or deletion of a few bases, together commonly referred to as *indels*. In some cases, both variants or *alleles* resulting from a mutation persist in populations. These are known as *single-nucleotide polymorphisms (SNPs)*. On the other hand, structural variations are long-range variations in chromosomes including insertions, deletions, copy number variations (CNVs), inversions and translocations.

Some of these variations or genotypes result in changes in traits or phenotypes and can cause diseases through alteration of the structure of the proteins encoded, change in regulation of expression, or other mechanisms. Association mapping refers to associating regions in genome or variations to phenotypes. Although association does not imply causality, the associated regions may then be investigated for causal variants.

Association mapping is typically done in the form of a *genome-wide association study (GWAS)* illustrated in Figure 1.5. For categorical phenotypes firstly two groups of individuals are selected – individuals who have the trait or the disease of interest (cases) and who do



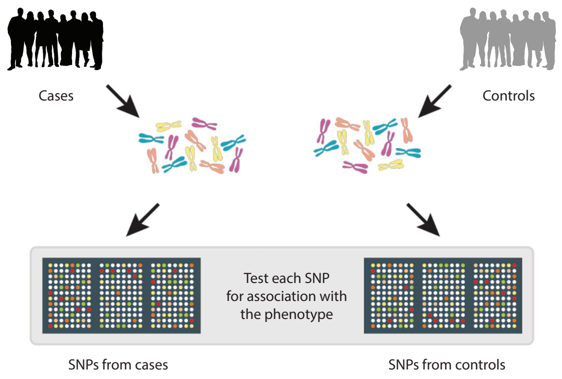
(a) Sequence variations (b) Structural variations

Figure 1.4: **Genomic variations.** Common types of (a) sequence variations and (b) structural variations. Sequence variations involve a single or a few nucleotides whereas structural variations affect large regions of genomes.

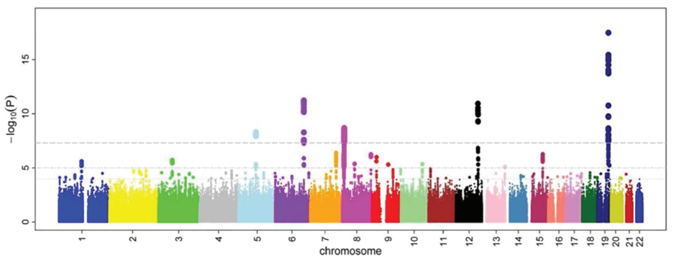
not (controls). Then a *SNP array* is used to determine the alleles present at a set of known SNP sites for all individuals in the study.

Then each SNP is tested for association with the phenotype by computing a P-value using a statistical test. *P-value* is the probability of observing an outcome as extreme as the one being observed if the *null hypothesis* is true. The null hypothesis in this context is the SNP is not associated with the phenotype whereas the *alternate hypothesis* is that it is associated with the phenotype. A popular approach to visualize the resulting P-values across the genome is to create a *Manhattan plot* where negative logarithm of P-values are plotted against genomic co-ordinates of SNPs. In a typical GWAS, P-values of millions of SNPs are computed. Therefore the P-value threshold for significance must be corrected for multiple testing and SNPs with P-values less than 5×10^{-8} are commonly considered significant for humans [56, 24]. Since there can be differences in allele frequencies across populations, correcting for population structure and other confounding factors such as age and sex is often needed for P-value computation.

GWAS may also involve analysis of quantitative phenotypic data. One approach is to regress the phenotype against principal components of the genotype data to account for population structure, as well as against other confounding factors. Then P-values are computed for each SNP typically using *chi-squared test* by including the SNP in the regression. Same approach can also be used for categorical phenotypes with the use of logistic regression.



(a)



(b)

Figure 1.5: **Genome wide association mapping.** Genome wide association mapping (GWAS) is performed by using a SNP array to determine the alleles at a set of SNP sites for some individuals with the phenotype (cases) and some who do not (controls). Each SNP is then tested for association with the phenotype and P-values are determined. P-values across the genome are often visualized using Manhattan plots (adapted from www.mpg.de and en.wikipedia.org).

Since 2005 thousands of GWA studies have been performed [160] mostly in human and association between many SNPs and phenotypes have been uncovered. However, GWAS design has a number of limitations. The construction of the SNP array requires knowledge of the reference genome and where the SNPs are located making it difficult to apply to most species other than human. Even in human if the disease is caused (risk elevated) by a structural variation or a rare variant not on the array, follow up is required to determine the causal variant even if association is detected which is often difficult. Moreover, if the disease is caused by a variant in a region not in the reference, reference based methods are unlikely to work. Many of these limitations can be overcome by using NGS data as sequencing gets cheaper but current approach to association mapping from sequencing reads relies on mapping reads to a reference genome and calling different kinds of variants. But calling structural variants is difficult and this approach is again unable to map associations in regions not in the reference.

1.5 Outline

In Part I of this dissertation, we present statistical models for genome assembly. In Chapter 2, the problem of evaluating genome assemblies is addressed. We present a generative model for sequencing and develop a method for computing likelihoods of genome assemblies implemented in CGAL (computing genome assembly likelihoods) which can be used for evaluating assemblies. Application to real and simulated datasets including the GAGE and Assemblathon 1 datasets reveal that likelihood is more accurate in assessing genome assemblies compared to contiguity based measures and reflects completeness of the assembly which is missed by other approaches.

In Chapter 3, we use the generative model for sequencing to develop a method for scaffolding. Similar generative models are used to estimate gaps between contigs and approximate the change in likelihood of the assembly if the contigs are joined with the estimated gap. The method is implemented in a tool called SWALO (scaffolding with assembly likelihood optimization). It is based on a rigorous statistical model yet we make approximations whenever necessary to make it efficient. We analyze datasets recently introduced by Hunt *et al.* to evaluate scaffolders and find that SWALO outperforms all other scaffolding tools.

Finally, in Part II of this dissertation, we focus on association mapping and present an alignment free method from sequencing reads. The method does not require a reference genome enabling association mapping in organisms with no or incomplete reference genome. It works by testing for association between each k-mer and performing local assemblies of the k-mers with significant association and the implementation is titled HAWK (hitting associations with k-mers). We also discuss some findings upon applying our method to the 1000 genomes project data.

Part I

Genome Assembly

Chapter 2

CGAL: computing genome assembly likelihoods

2.1 Background

Genome assembly is the process of merging fragments of DNA sequence produced by shotgun sequencing in order to reconstruct the original genome. The assembly problem is known to be NP-hard for a number of formulations [99, 100, 112] and is also complicated by many types of sequencing errors, experimental biases and the volume of data that must be processed. For these reasons, in addition to differences in underlying theory and algorithms, popular assembly methods employ many different heuristics and assemblies produced by existing methods differ substantially from each other [35, 141].

Paradoxically, the difficulties of sequence assembly have been compounded by sequencing advances in recent years collectively termed *next-generation sequencing* technologies. Next-generation sequencing technologies such as 454 pyrosequencing by Applied Sciences [96], Solexa/Illumina sequencing, the SOLiD technology from Applied Biosystems and the Helicos single-molecule sequencing [52] produce data of much greater volume at a much lower cost than traditional Sanger sequencing [142]. However, read lengths are considerably shorter and error rates are higher than those in Sanger sequencing. To allow de novo sequencing from short reads from next generation sequencing machines several assemblers have been developed such as Velvet [167], Euler-sr [15], ABySS [147], Edena [55], SSAKE [161], VCAKE [66], SHARCGS [32], Allpaths [14], SOAPdenovo [85], Celera WGA [107], the CLC bio assembler and others [35, 141]. A key problem that has arisen is to determine which assembler is "the best". In the past this has been done with the help of a number of measures such as N50 scaffold or contig lengths - which is the maximum contig (scaffold) length such that at least half the total length is contained in contigs (scaffolds) of length greater or equal that length. Although simulation studies show that simple metrics correlate with assembly quality, currently used metrics are crude and provide only condensed summaries of the result. They can therefore be very misleading [141, 158]. For example, the assembly consisting of simply gluing all reads end-to-end has a very large N50 length, but is obviously a poor assembly. Phillippy *et al.* presented a software called *amosvalidate* [126] that identifies mis-assembly features and suspicious regions but it does not have high specificity and has not been widely adopted. Narzisi *et al.* utilized feature-response curve [116] to rank assemblies based on features identified by amosvalidate. Studies such as [168, 87, 27, 1] have discussed these issues and produce interesting insights into assembler performance but do not provide an intrinsic direct measure of assembly quality. The recent Assemblathon 1 competition used 10 different metrics [35] that attempt to reveal more information than just N50 values, but most of them can only be computed when the genome that is being assembled is known, and are therefore not useful in practice on real data.

In this paper we present a computationally efficient approach for computing the likelihood of an assembly which provides a way to assess assemblies without a ground truth. Intuitively, the likelihood assessment evaluates the uniformity of coverage of the assembly, taking into account errors in the reads, the insert size distribution and the extent of unassembled data. Genome assembly by maximizing likelihood has been proposed previously by Myers [108] and Medvedev *et al.* [99] but their formulations are based on simplified models that omit evaluating and utilizing crucial parameters, especially sequencing error. To demonstrate the power of our approach for assembly quality evaluation we have implemented our methods in a program called CGAL that we evaluate by testing several assemblies from different programs with varying input parameters in setting where the desired target genome is known. For each assembly, we compute the likelihood using our tool and then compare our likelihood computation to standard measures such as N50 contig values, sequence similarity with the reference genome as well as values reported by amosvalidate. Although it is beyond the scope of this paper to compare all assemblers and explore all parameters, our results indicate that likelihood is meaningful and useful for evaluating assemblies.

2.2 Results

Our overall approach is simple: we describe a probabilistic generative model for sequencing that captures many aspects of sequencing experiments, and from which we can compute the likelihood of an assembly. This intuitive framework is, however, complicated by one major difficulty which is the problem we address in this paper: to compute the likelihood of an assembly it is necessary, in principle, to consider the possibility that a read was produced from every single location in the assembly. This results in an intractable computation, that we circumvent by approximating the likelihood via a reduction to a small set of "likely" sites from which each read originated (using a mapping of the reads to the assembly). This requires an examination of the quality of the approximation, and leads to yet another difficulty, which is how to compute the likelihood for reads that do not map to the assembly at all. These issues are addressed in this paper and their solution is what enables our program for likelihood computation to be efficient and practical.

We begin by describing the statistical model that forms the basis for our likelihood computation. We believe that our model incorporates many aspects of typical sequencing experiments, but it can be easily generalized to accommodate additional parameters if desired.

A generative model for sequencing

Let, $\mathcal{R} = \{r_1, r_2 \ldots r_N\}$ be a set of N paired end (or mate pair) reads generated from a genome, $\mathcal{G}$ (our model can, in principle be adapted to single end reads but we do not consider that here). We assume a fragment represented by two paired-end reads $r_i = (r_{i1}, r_{i2})$ is generated according to the following model:

- A fragment length, l_i is selected according to a distribution, F.
- A site for the $5'$ end of the fragment, s_i is selected according to a distribution, S.
- The ends of the fragment are read as r_{i1} and r_{i2} according to an error model, E which comprises mismatches as well as indels.

The generative model is illustrated in Figure 2.1.

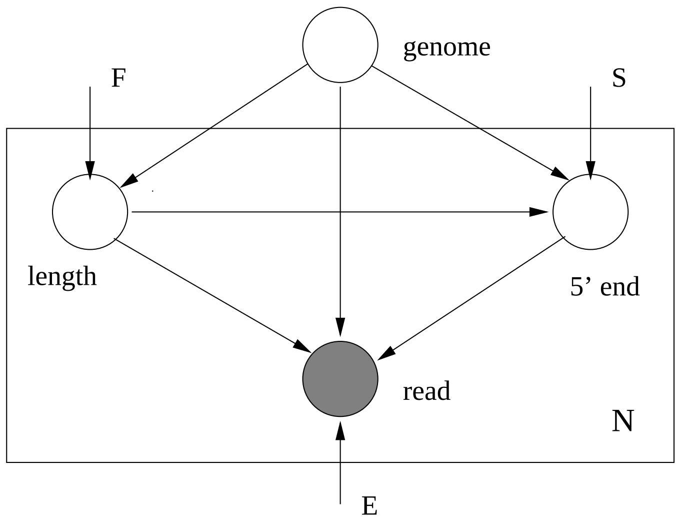


Figure 2.1: **A generative graphical model for sequencing.** N paired end reads are generated independently from a genome. Here, F denotes the distribution of fragment lengths, S is the distribution of start sites of reads and E stands for error parameters.

Computing likelihood

Computing the likelihood of an assembly means that the probability of the (observed) set of reads is computed with respect to a proposed assembly using the model described in the previous section. The probability of a sequence of length L generating a paired-end or mate pair read (termed read from now on) r_i is

$$p(r_i) = \sum_{l=1}^{L} p_F(l) \sum_{s=1}^{L-l+1} p_S(s) \sum_{e \in \mathcal{E}} p_E(r_i | a_s \dots a_{s+l-1})$$

where $a_s \dots a_{s+l-1}$ is the assembly subsequence starting at s of length l, $\mathcal{E}$ denotes all possible ways of obtaining r_i from $a_s \dots a_{s+l-1}$ and

$$\begin{aligned} p_F(l) &= \text{probability that the fragment length is } l, \\ p_S(s) &= \text{probability that the } 5' \text{ end of the fragment is at site } s, \\ p_E(r_i | a_s \dots a_{s+l-1}) &= \text{probability of obtaining } r_i \text{ from } a_s \dots a_{s+l-1} \\ & \text{with sequencing errors given by } e. \end{aligned}$$

Although in theory a read could have been generated from any site (assuming that every base could have been an error), in practice the probability decreases considerably with increasing number of disagreements between the source sequence and the read sequence. We therefore approximate the probability $p(r_i)$ by mapping the read to the assembly and ignoring mappings with large number of differences. If M_i is the number of such mappings of read r_i, the probability is given by

$$p(r_i) \approx \sum_{j=1}^{M_i} p_F(l_{i,j}) p_S(s_{i,j}) p_E(r_i | a_{i,j})$$

where $l_{i,j}$, $s_{i,j}$, $a_{i,j}$ and $e_{i,j}$ are fragment length, start site, assembly subsequence and errors corresponding to j-th mapping of i-th read respectively. The above equation generalizes to assemblies with more than one contig. Given an assembly $\mathcal{A}$ and a set of reads $\mathcal{R} = \{r_1, r_2 \dots r_N\}$, the *log likelihood* is given by

$$\begin{aligned} l(\mathcal{A}; \mathcal{R}) &= \log \prod_{i=1}^{N} p(r_i | \mathcal{A}) \\ &\approx \sum_{i=1}^{N} \log \sum_{j=1}^{M_i} p_F(l_{i,j}) p_S(s_{i,j}) p_E(r_i | a_{i,j}). \end{aligned}$$

In the above equation $M_i \geq 1$ for all reads r_i, and in Methods we explain how we obtain alignments for *all* reads and how to learn the needed distributions.

Validation with simulated data

To test our implementation we developed a simulator that generates reads according to error parameters provided and fragment lengths distributed according to a Gaussian distribution.

We generated 3 million 35bp paired end reads from a strain of *Escherichia coli* ([NCBI: NC_000913.2]) and an assembly of *Grosmannia clavigera* ([DDBJ/EMBL/GenBank: ACXQ00000000]) reported in [31]. Table 2.1 shows percentage difference in likelihood values computed using true parameters provided to the simulator and using parameters inferred by CGAL.

Table 2.1: **Percentage difference between the simulator and CGAL**

Genome	Length(bp)	% difference
E.coli	4.6M	0.074
G. clavigera	29.1M	0.0755

Performance of assemblers on *E. coli* reads

We assessed performance of four assemblers: Velvet, Euler-sr, ABySS and SOAPdenovo on an *Escherichia coli* dataset ([SRA:SRR 001665] and [SRA:SRR 001666]). We chose *E. coli* because its assembly is a true "gold standard" without questions about reliability or accuracy. We assembled the reads using the assemblers mentioned for different hash lengths (k-mer used for constructing *de Bruijn* graph [124]). Likelihood values for assemblies along with the likelihood value for the reference ([NCBI: U00096.2]) are shown in Figure 2.2(a).

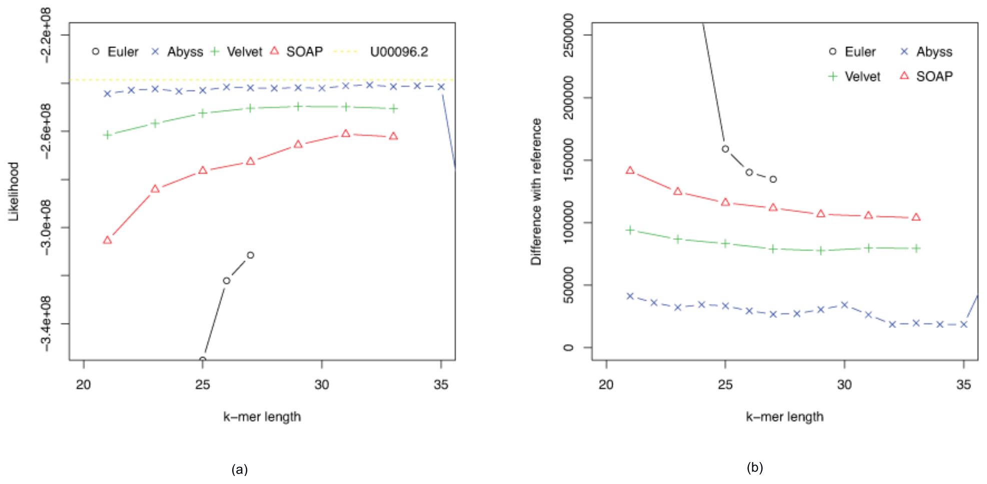


Figure 2.2: **(a) Hash length vs log likelihood for *E. coli*.** Log likelihoods of assemblies generated using different assemblers for varying k-mer lengths shown. The dotted line corresponds to the log likelihood of the reference. **(b) Hash length vs difference from reference for *E. coli*.** Differences between assemblies and the reference are shown where difference refers to numbers of bases in the reference not covered by the assembly or are different in the reference and the assembly.

For this dataset ABySS outperforms others when likelihood is used as the metric. We also aligned the assemblies to the reference with NUCmer [29] and Figure 2.2(b) shows differences from reference against hash lengths. The relations among likelihood, N50 length and similarity are illustrated in Figure 2.3 and Figure A.1. They suggest that likelihood values are better at capturing sequence similarity than other metrics commonly used for evaluating assemblies such as N50 scaffold or contig lengths. We also ran the amosvalidate pipeline to obtain numbers of mis-assembly of features and suspicious regions (Figure 2.4) and plotted the feature response curves (FRC) [116] of the assemblies (Figures A.3, A.3). FRC also ranks an ABySS assembly as the best one.

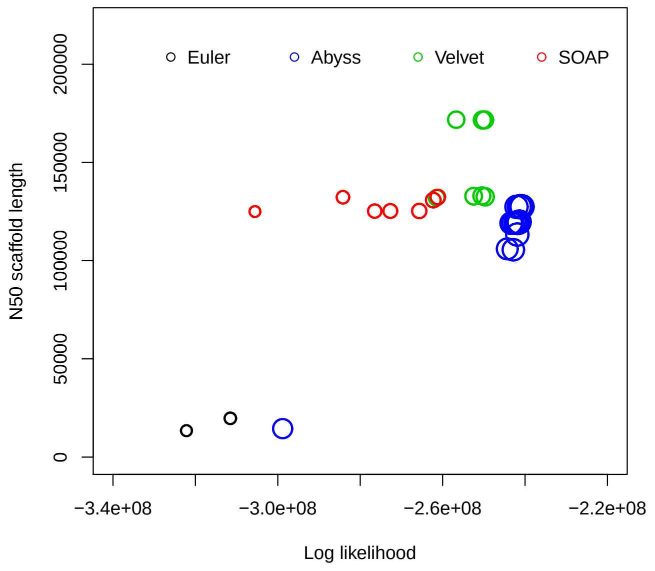


Figure 2.3: **Log likelihood vs N50 scaffold length for *E. coli.*** Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.9372048, (ii) log likelihood vs N50 scaffold length: 0.44011, (iii) N50 scaffold length vs similarity: 0.3216882.

Similar analysis was performed on a different *Escherichia coli* dataset downloaded from CLC bio [22]. It consists of approximately 2.6 million 35bp paired end Illumina reads (approximately 40X coverage) along with a reference genome ([NCBI: NC_010473.1]). We noticed that many of the assemblies have better likelihood than the reference. However, we assembled reads that could not be mapped to the reference and after running BLAST [3] we found another substrain of *Escherichia coli* strain K-12, MG1655 ([NCBI: NC_000913.2]) that has a better likelihood than all assemblies. We conjecture that the reads were generated from NC_000913.2. Likelihood values are shown in Figure 2.5 and relationships among likelihood, similarity and N50 values are illustrated in Figures A.5, A.6, A.7, A.8.

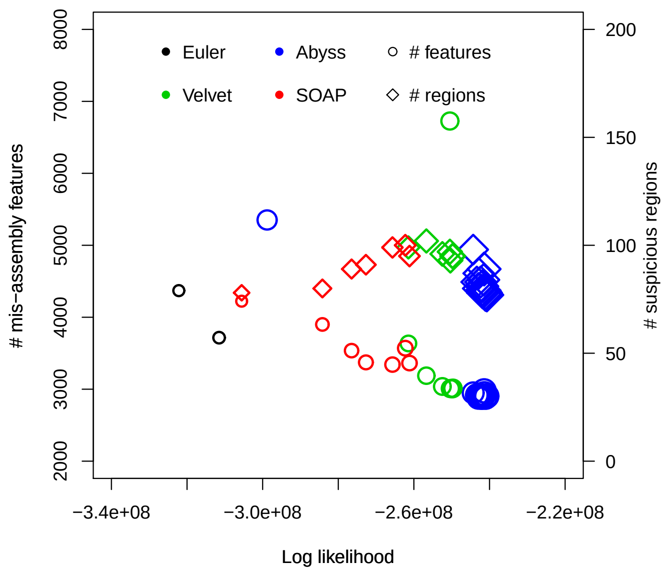


Figure 2.4: **Log likelihood vs numbers of mis-assembly features and suspicious regions for *E. coli*.** Numbers of mis-assembly features and suspicious regions reported by amosvalidate are shown against log likelihoods. Each symbol corresponds to an assembly generated using an assembler for some hash length and sizes of symbols correspond to similarity with reference. The R^2 values are (i) log likelihood vs # mis-assembly features: 0.8922, (ii) log likelihood vs # suspicious regions: 0.9039, (iii) similarity vs # mis-assembly features: 0.8211, (iv) similarity vs # suspicious regions: 0.7723.

Performance of assemblers on *G. clavigera* reads

To assess assemblies of a larger genome, we used the dataset generated for sequencing an ascomycete fungus, *Grosmannia clavigera* by DiGuistini *et al.* [31]. We ran Velvet, ABySS and SOAP on PE Illumina reads with fragment length mean of 200 bp [SRA:SRR 018008-11] and 700 bp [SRA:SRR 018012].

The likelihood values of the 200bp fragment reads for the assemblies are shown in Figure 2.6(a). It also shows likelihood values for assemblies [DDBJ/EMBL/GenBank: ACXQ00000000] and [DDBJ/EMBL/GenBank: ACYC00000000] reported in [31] which were generated using Sanger and 454 reads as well as Illumina reads. The numbers of mis-assembly features and suspicious regions identified by amosvalidate and the feature response curves (FRC) are shown in Figure A.12.

Figure 2.6(b) shows that the assembly with most sequence coverage is produced by ABySS. However, in this case ABySS assemblies are much longer compared to other assemblies and references (Tables A.9, A.10, A.11). This results in lower likelihoods compared to some assemblies by Velvet and SOAPdenovo. In FRC analysis, coverage is estimated using

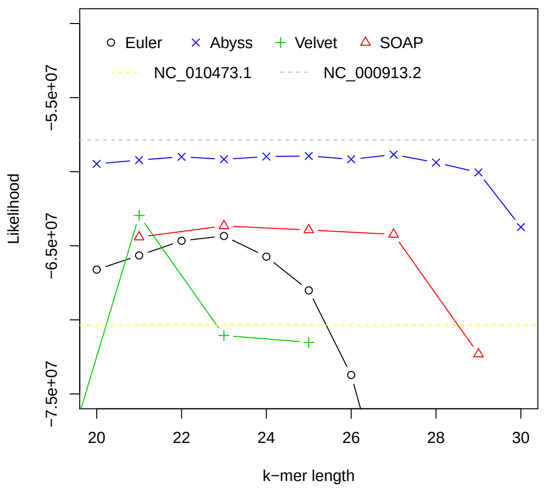


Figure 2.5: **Hash length vs log likelihood for *E. coli* data from CLC Bio.** Log likelihoods of assemblies generated using different assemblers for varying k-mer length are shown. The yellow dotted line corresponds to the log likelihood of the reference provided and the gray dotted line corresponds to the log likelihood of the strain we believe the reads were generated from.

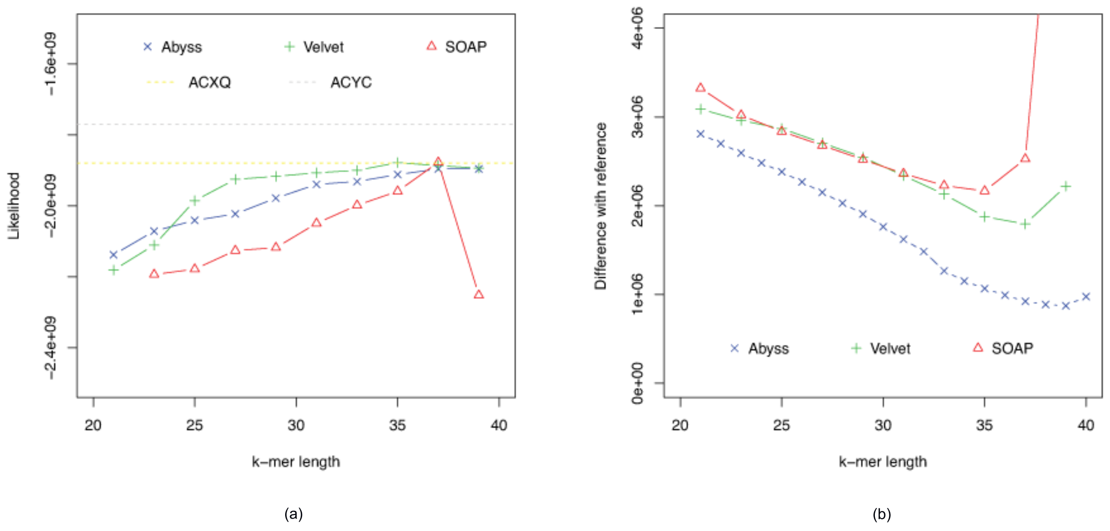


Figure 2.6: **(a) Hash length vs log likelihood for *G. clavigera.*** Log likelihoods of assemblies generated using different assemblers for varying k-mer lengths shown. The dotted lines correspond to log likelihood of the assemblies generated using Sanger, 454 as well as Illumina data. **(b) Hash length vs difference from reference for *G. clavigera.*** Differences between assemblies and the reference are shown where difference refers to numbers of bases in the reference not covered by the assembly or are different in the reference and the assembly.

assembly length and so it does not take into account the unassembled sequences and ranks ABySS assemblies above others. It is interesting that assemblies with the best likelihood and sequence similarity are generated for higher values of hash length than are optimal for producing high N50 values.

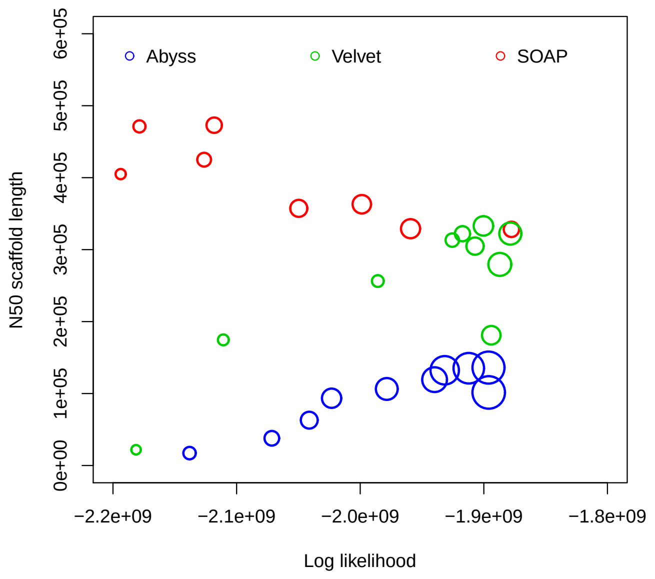


Figure 2.7: **Log likelihood vs N50 scaffold length for *G. clavigera*.** Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.4545793, (ii) log likelihood vs N50 scaffold length: 0.002397233, (iii) N50 scaffold length vs similarity: 0.006084032.

GAGE results

We computed likelihoods for the assemblies generated in the GAGE project [141]. Tables A.12, A.13, A.14 show likelihoods of Library 1 and number of reads mapped to assemblies by Bowtie 2 [80]. We found that likelihood values of Library 1 are dominated by coverage and contiguity does not affect these values greatly. However, contiguity have more effect on likelihoods of Library 2 with longer insert size (Tables A.12, A.13, A.14) as might be expected. Total likelihood along with coverage and N50 values are shown in Tables 2.2, 2.3, 2.4, 2.5. For human we computed Library 2 likelihoods for assemblies with best three likelihoods of Library 1. Likelihood values of Library 2 for bumble bee assemblies were not computed as only a small fraction of the reads could be mapped to the assemblies.

Table 2.2: **Likelihoods of GAGE assemblies of *S. aureus***

Assembler	Likelihood	# reads mapped	Coverage(%)	Scaffold N50 (kb)	Contig N50 (kb)
ABySS	-23.34×10^7	1236230	99.74 †	34	29.2
ALLPATHS-LG	-24.53×10^7	1220328	99.38	1092	96.7
Bambus2	-23.76×10^7	1200527	98.68	1084	50.2
MSR-CA	-25.85×10^7	1192001	98.70	2412	59.2
SGA	-26.61×10^7	1018936	98.09	208	4.0
SOAPdenovo	-23.55×10^7	1212384	99.62	332	288.2
Velvet	-23.28×10^7	1203907	99.21	762	48.4
Reference	-22.38×10^7	1268718	-	-	-

† Value reported in the GAGE paper is 98.63

Table 2.3: **Likelihoods of GAGE assemblies of *R. sphaeroides***

Assembler	Likelihood	# reads mapped	Coverage(%)	Scaffold N50 (kb)	Contig N50 (kb)
ABySS	-27.55×10^7	1199197	99.11†	9	5.9
ALLPATHS-LG	-26.61×10^7	1237938	99.53	3192	42.5
Bambus2	-32.56×10^7	1111596	95.07	2439	93.2
CABOG	-39.23×10^7	1022732	92.49	66	20.2
MSR-CA	-31.61×10^7	1155078	96.48	2976	22.1
SGA	-31.58×10^7	1031547	97.69	51	4.5
SOAPdenovo	-27.67×10^7	1212959	99.12	660	131.7
Velvet	-28.77×10^7	1176125	98.40	353	15.7
Reference	-25.99×10^7	1255750	-	-	-

† Value reported in the GAGE paper is 96.99

Table 2.4: **Likelihoods of GAGE assemblies of human chromosome 14**

Assembler	Likelihood	# reads mapped	Coverage(%)	Scaffold N50 (kb)	Contig N50 (kb)
ABySS	-23.44×10^8	22096466	82.22	2.1	2
ALLPATHS-LG	-22.77×10^8	23122569	97.24	81647	36.5
CABOG	-21.26×10^8	23433424	98.32	393	45.3
SOAPdenovo	*	*	98.17	455	14.7
Reference	-19.04×10^8	23978017	-	-	-

* Likelihood not computed as reads could not be mapped with Bowtie 2

Assemblathon 1 results

We also analyzed the assemblies submitted for Assemblathon 1 [35]. Likelihoods of library of insert size of mean 200bp for all assemblies are given in Table A.15 and Figure 2.8 shows

Table 2.5: **Likelihoods of GAGE assemblies of bumble bee, *B. impateins***

Assembler	Likelihood	# reads mapped	Scaffold N50 (kb)	Contig N50 (kb)
ABySS	-30.83×10^9	72629126	-	-
CABOG	-19.99×10^9	92844610	1125	23.5
MSR-CA	-22.84×10^9	78755756	1246	32.4
SOAPdenovo	*	*	1374	57.1

* Likelihood not computed as reads could not be mapped with Bowtie 2

the relationship between likelihood and coverage. Among these, we took the entries with the highest likelihood for top ten participants and computed likelihoods of libraries of insert sizes of means 3000bp and 10000bp. Table 2.6 shows total likelihoods of top ten participants along with their Assemblathon 1 rankings.

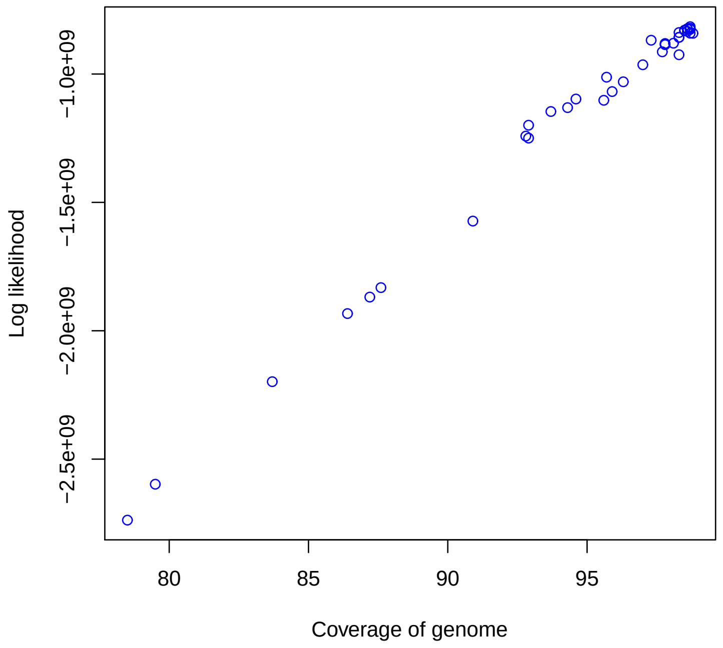


Figure 2.8: **Coverage vs Log likelihood for Assemblathon 1 entries.** Coverage is shown on the x-axis and log likelihood is shown on the y-axis. Each circle corresponds to an assembly. The R^2 value is 0.989972.

Table 2.6: **Likelihoods of Assemblathon 1 assemblies**

Assembler	Likelihood	#reads mapped	Assemblathon 1 rank
BGI 1	-20.17×10^8	42005212	2
CSHL 2	-20.19×10^8	41973576	5
BCCGSC 5	-20.23×10^8	41891758	7
IoBUGA 2	-20.49×10^8	41931526	9
RHUL 3	-20.69×10^8	41753084	10
DOEJGI 1	-20.73×10^8	41836210	4
WTSI-P 2	-20.81×10^8	41748504	11
Broad 1	-21.75×10^8	41778343	1
EBI 1	-21.83×10^8	41377165	8
WTSI-S 4	-30.81×10^8	37442672	3

2.3 Discussion

E. coli

We find that for both *E. coli* datasets assemblies with the best likelihoods are constructed by ABySS. They also have most similarity with references (assuming [NCBI: NC_000913.2] is the reference for the CLC bio dataset). The R^2 values (Figures 2.3, 2.4 and Figure A.1) reveal that likelihoods reflect sequence similarity better than contiguity statistics such as N50 values as well as numbers of mis-assembly features and suspicious regions identified by amosvalidate. Analysis of two different *E. coli* datasets also reveal that for assemblers like Velvet, SOAPdenovo higher likelihood values are achieved for different values of k-mer length used to construct the de Bruijn graph during assembly.

G. clavigera

For the *G. clavigera* dataset one of the Velvet assemblies has the highest likelihood. Although ABySS assemblies have more coverage, they have lower likelihood because of much longer total length. Despite this we see from R^2 values that likelihood values reflect sequence similarity better than N50 values (Figure 2.7 and Figures A.9, A.11) and numbers of mis-assembly features and suspicious regions reported by amosvalidate. This suggests that likelihood values are useful in simultaneously evaluating coverage and total assembly length.

GAGE

For the GAGE *S. aureus* dataset, we find that the assembly generated using Velvet has the best likelihood but likelihoods of a few other assemblies are close. For *R. sphaeroides*, the ALLPATHS-LG assembly has the best likelihood which is also the assembly with highest coverage and N50 scaffold length. The CABOG assembly of human chromosome 14 is the one with best likelihood. The CABOG assembly also has the highest coverage and N50 contig length among the assemblies. In all three cases, we find that the reference sequences have the highest likelihoods and the highest number of reads mapped to them by Bowtie 2. For the bumblebee data, the assembly using CABOG has best likelihood among the three (the likelihood of SOAPdenovo assembly could not be computed as reads could not be mapped to it using Bowtie 2).

Assemblathon 1

Figure 2.8 reveals that for the Assemblathon 1 dataset, likelihoods of small fragment library correlates well with coverages. Overall, we find that participants with the ten highest likelihoods were ranked within the top eleven by Assemblathon 1 organizers but there are differences between the two rankings. The entry with the highest likelihood is by Beijing Genomics Institute (BGI) which was ranked two in the original paper. The differences in rankings are primarily due to the emphasis on contiguity by Assemblathon1 organizers while our likelihood model implicitly places high importance on coverage. This brings up the issue that better contiguity statistics can be achieved by not reporting hard to assemble regions and these values may be misleading if they are not used in conjunction with an indicator of coverage.

Applications

Currently, assembly evaluation projects rely mostly on simulated data or data from genomes that have been sequenced previously [35, 141]. Having a tool that can assess quality of assembly without the need for a reference will allow researchers who work with real data from genomes that have not been sequenced before to assess the performance of different assemblers on their data, and to optimize parameters in the programs they are using.

Analysis of two different datasets from *E. coli* reveal that performance of some assemblers vary significantly depending on the k-mer chosen for constructing the de Bruijn graph. Moreover, the ‘optimal’ value depends on read length and sequence coverage. Likelihood values can therefore guide selection of parameter values.

Maximum likelihood genome assembly was introduced by Medvedev and Brudno [99] but they do not consider sequencing errors or paired end reads. A likelihood model taking into account these may be the next step towards genome assemblers for real data that try to maximize likelihood.

2.4 Conclusions

In this paper we presented a tool for computing the likelihood of an assembly. The result can be used as a metric for evaluating and comparing assemblies. In the past this has been done using many different criteria including N50 lengths, total sequence length, number of contigs. The likelihood model incorporates these directly or indirectly in addition to other important factors such as genome coverage and assembly accuracy and combines them into a single metric for evaluation.

We have also used our tool to assess performance of some assemblers on a few different datasets. Our results indicate that likelihood reflects sequence similarity which is missed by other metrics commonly used and is going to be a valuable tool for evaluating assemblies generated by different assemblers and for different values for input parameters.

2.5 Materials and methods

Mapping reads

The first step in computing the likelihood is mapping reads to the assembly. A number of tools are available for this such as Bowtie [81, 80], MAQ [83], BWA [82] and BFAST [57]. Our present implementation can use either BFAST or Bowtie 2 for mapping reads as they support mapping with indels and report multiple alignments in a way that gives all the required information without accessing the assembly sequence. But any tool that reports multiple alignments of reads and allows for insertions/deletions can be used with some minor modifications.

However, existing tools do not usually map all reads, and for the likelihood computation it is necessary to assign probabilities to reads that are unmapped. We found that mapping tools were unable to map a large fraction of reads in our experiments. One option is to assign probabilities to these reads assuming that they could have been generated from any site with number and types of errors not handled by the mapping tool. But it is then often the case that unmapped reads are deemed more probable than mapped ones, which we believe is anomalous. Furthermore, in our analyses we determined that the resulting probabilities were inaccurate (results not shown). Therefore, we chose to directly align the reads not mapped by BFAST or Bowtie 2 using an adaptation of the Smith-Waterman algorithm. For this we have adapted the striped implementation of Smith-Waterman algorithm by Farrar [38]. This step is time consuming, so we align only a random subset of reads with the number specified by the user and approximate probabilities using these.

Learning Distributions

To compute the likelihood from mapped reads, we need to learn the distribution of fragment lengths, their distribution across genome and error characteristics. Since they differ with library preparation methods and sequencing instruments, we have chosen to learn these from sequencing data generated in the experiment. We do this by mapping reads to the assembly and using reads that map uniquely. However, this can be easily extended to take into account all reads by using the EM algorithm at the expense of more iterations. We explain each distribution in more detail below.

- **Fragment length distribution**

  The distribution of fragment lengths depends on the method used for size selection and may not be approximated well by common distributions [135]. So, we use the empirical distribution.

- **Distribution of fragments along genome**

  In our implementation, we assume that fragments are distributed uniformly across the genome. We leave incorporating sequencing bias as future work

- **Error model**

  In the error model used at present, we have made the assumption that sequencing errors are independent of one another. We learn an error rate for each position in the read since error rates are known to be different across positions in reads. [33] We also learn separate error rates for each type of base and substitution types. Although errors are known to depend on sequence context [33] we have ignored them for the sake of simplicity.

  To account for varying indel rates across positions in reads, we learn an insertion rate and a deletion rate for each position in the read. Since short indels are more likely than longer ones we also count number of insertions and deletions by length.

Implementation

As mentioned earlier, we use BFAST or Bowtie 2 to map reads to assemblies. The parameters are set so that they report all alignments of a read found.

The remaining code for computing likelihood is written in C++ and it consists of three parts.

- **c**onvert: It converts the output generated by BFAST or Bowtie 2 to an internal format. It also separates reads with no end or one end mapped and reads with ends mapped to different scaffolds if needed. Separating this module also allow us to support other mapping tools by writing a conversion routine.

- **a**lign: To align the reads not mapped by the mapping tool, we have adapted the striped implementation of Smith-Waterman algorithm by Farrar [38]. As this step is time consuming, we align a random subset of reads with the number determined by the user. This step is multithreaded to speed up the process.

- **c**gal: This part learns the fragment length distribution and parameters for the error model using uniquely mapped reads and then uses these to compute the likelihood value.

Assembling genomes

To assemble reads, we varied k-mer length used to construct the de Bruijn graph to obtain different assemblies for each assembly tool. For other parameters default values or values suggested in manuals were used.

Data analysis

Likelihoods were computed by running CGAL with default parameters and aligning between 300 and 1000 randomly chosen reads not mapped the mapping tool used. The running time of CGAL was approximately 1/3 the time taken to map reads using Bowtie 2.

To compute the *difference* between an assembly and the reference we aligned the assembly to the reference using NUCmer [29] and the *difference* refers to the number of bases in reference that are either not covered by the assembly or different in reference and assembly. Contigs have been generated by splitting scaffolds at sites with 25 or more N's (character representing any base).

Chapter 3

SWALO: scaffolding with assembly likelihood optimization

3.1 Introduction

The emergence of next-generation sequencing technologies has led to development of various assays to probe many aspects of interest in molecular and cell biology due to their low cost and high throughput. However, there is still scope for improvement in genome assembly which is essential for running many of these assays. The high cost, low sequencing coverage and high error rates of single molecule real time (SMRT) and nanopore sequencing mean that most of the genomes are being assembled from next-generation sequencing data or a combination of the two. In this paper, we demonstrate that considerable improvement in genome assembly using next-generation sequencing can be achieved through application of statistical models for sequencing while joining contigs. The contigs themselves may be generated using any sequencing technology.

Genome assembly typically consists of two steps. The first step is to merge overlapping reads into contigs commonly done using de Bruijn or overlap graphs. In the second step which is known as "scaffolding", contigs are oriented and ordered using paired-end or mate-pair reads (we use the term read pair to refer to either). Scaffolding is a critical part of the genome assembly process and hence is built into most assemblers [167, 147, 14, 92, 148]. A number of stand-alone scaffolders such as Bambus2 [127, 76], GRASS [48], MIP [139], Opera [42], SCARPA [34], SOPRA [28], SSPACE [7] have also been developed. Most of the scaffolding algorithms rely on heuristics or user input to determine parameters such as minimum number read-pairs linking contigs to join them ignoring contig lengths, sequencing depth and sequencing errors. Recently, Hunt *et al.* evaluated scaffolding tools on real and simulated data and observed that although many of the scaffolders perform well on simulated datasets, they display inconsistent performance across real datasets and mapping tools [62]. Their results show that SGA, SOPRA and ABySS are conservative and make very few scaffolding errors while SOAPdenovo identified more joins at the expense of greater number of errors indicating the need for a better scaffolding method.

Here we present a scaffolding method called Swalo which is based on a generative model for sequencing [132] that incorporates sequencing errors and insert size distribution while making approximations where necessary to make it efficient and applicable in practice. We use this model to determine whether joining a pair of contigs would result in an improvement in likelihood of the assembly making our method largely free of user parameters. We are also able to accurately estimate gaps between contigs that will aid the gap-filling process. Moreover, the use of a probabilistic model enables us to resolve multi-mapped read pairs using the EM (expectation maximization) algorithm that are ignored in most scaffolders.

Analysis of the standard datasets used in [62] that include the GAGE [141] datasets reveal that Swalo outperforms all other scaffolders and show pareto-optimal performance. We find that application of Swalo lead to better contiguity of assemblies facilitating downstream analysis while making very few scaffolding errors at the same time.

3.2 Methods

Figure 3.1 illustrates the main steps of Swalo. In the first step, reads are aligned to contigs, insert size distribution and error parameters are learned using reads that map uniquely and likelihood of contigs is computed using a generative model. We then construct the bi-directed *scaffold graph* which contains a vertex for each contig and there is an edge between contigs if joining them would result in an increase in likelihood. It uses probabilistic models to estimate maximum likelihood gaps between contigs correcting for the issue that we may not observe inserts from the entire distribution due to gaps between contigs and their lengths. It then approximates whether joining contigs would result in an increase in genome assembly likelihood. We use the EM (expectation maximization) algorithm to resolve multi-mapped read pairs. Contigs are then joined if the increase in likelihood is significantly higher than all other conflicting joins as determined by a heuristic. We select multiple joins consistent with one another using the dynamic programming algorithm for the weighted interval scheduling problem. Each of these steps is described in more detail in following sections.

Learning distributions and computing likelihood

The first phase in Swalo is to compute likelihood of the contigs and learn parameters to be used in the scaffolding process. To compute the likelihood of an assembly we use the generative model and parameter estimation approach presented in Chapter 2 Section 2.2. We however make some modifications to the distributions discussed in Chapter 2 Section 2.5.

- **Insert size distribution:** We use the empirical distribution as normal distribution is not always a good approximation. The distribution is learned using contigs longer than a minimum length (provided by the user) so that the distribution is not biased. We use a smoothed version of the distribution using a window size of 25 as the number

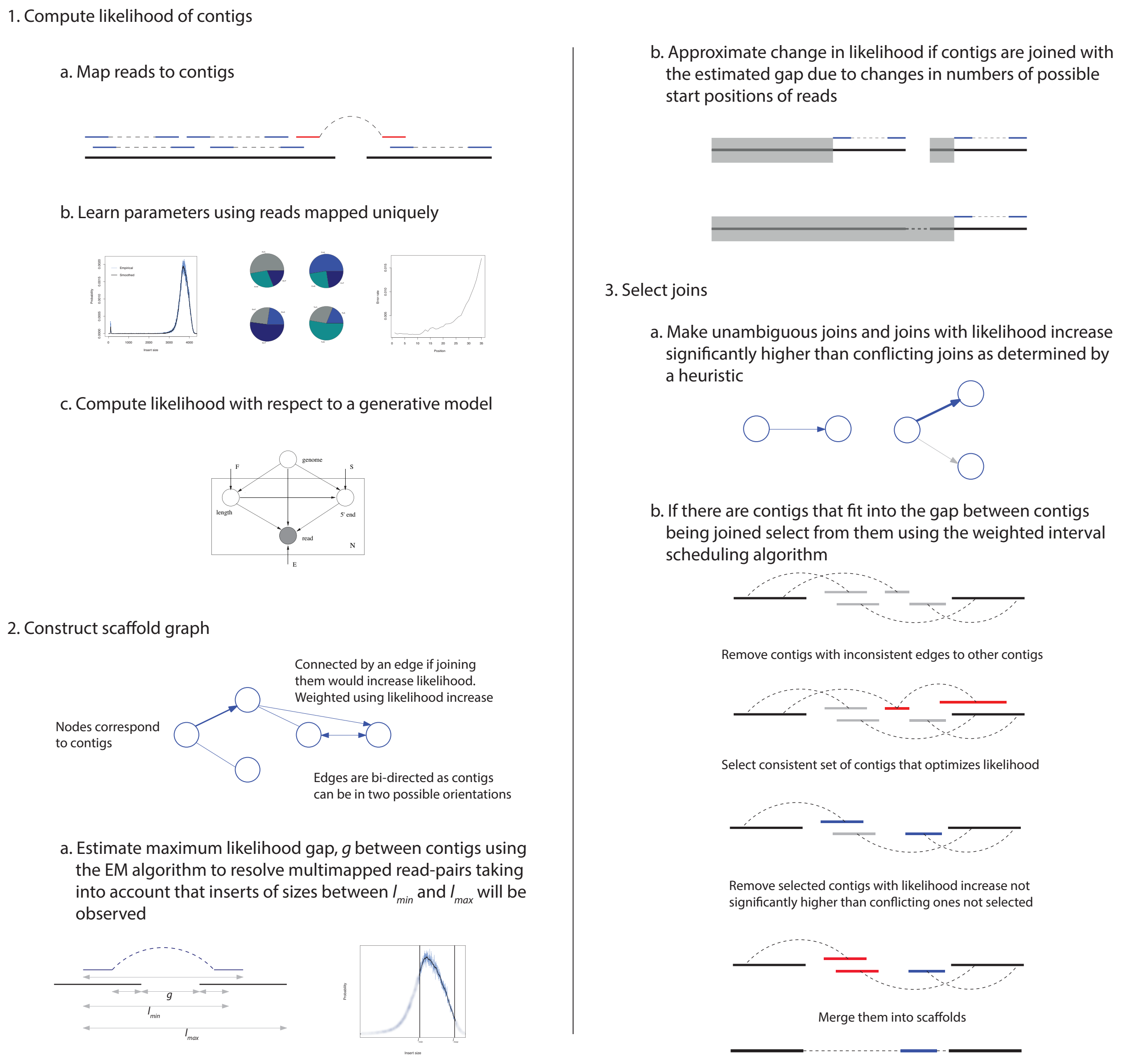

1. Compute likelihood of contigs
a. Map reads to contigs
b. Learn parameters using reads mapped uniquely
c. Compute likelihood with respect to a generative model
genome
F
S
length
5' end
read
N
E
2. Construct scaffold graph
Nodes correspond to contigs
Connected by an edge if joining them would increase likelihood. Weighted using likelihood increase
Edges are bi-directed as contigs can be in two possible orientations
a. Estimate maximum likelihood gap, g between contigs using the EM algorithm to resolve multimapped read-pairs taking into account that inserts of sizes between l_{min} and l_{max} will be observed
g
l_{min}
l_{max}
b. Approximate change in likelihood if contigs are joined with the estimated gap due to changes in numbers of possible start positions of reads
3. Select joins
a. Make unambiguous joins and joins with likelihood increase significantly higher than conflicting joins as determined by a heuristic
b. If there are contigs that fit into the gap between contigs being joined select from them using the weighted interval scheduling algorithm
Remove contigs with inconsistent edges to other contigs
Select consistent set of contigs that optimizes likelihood
Remove selected contigs with likelihood increase not significantly higher than conflicting ones not selected
Merge them into scaffolds

Figure 3.1: **Overview of SWALO.** In the first step, reads are aligned to contigs, uniquely mapped reads are used to learn insert size distribution and error parameters and likelihood of contigs is computed. In the next step, the scaffold graph is constructed by first estimating maximum likelihood gaps between contigs correcting for the issue that we may not observe inserts from the entire distribution due to gaps between contigs and their lengths and then approximating whether changes in number of possible start sites of reads (the regions shaded in grey) lead to an increase or decrease of assembly likelihood. Finally, we make the joins that are unambiguous or correspond to likelihood increase significantly higher than other conflicting joins. We use the dynamic programming algorithm for the weighted interval scheduling problem when there are multiple possible joins to select from.

of reads to learn the distribution from may not be very high for some datasets. We also compute mean, μ left-sided and right-sided standard deviations, σ_l and σ_r respectively and truncate the distribution at $\mu - 2.5 * \sigma_l$ and $\mu + 2.5 * \sigma_r$ to avoid linking contigs based on chimeric reads.

In situations where the number of read-pairs mapped to contigs long enough to learn distribution is very low (less than 100,000), we recommend switching to normal distribution and require the user to input a mean and a standard deviation.

- **Distribution of fragments:** We ignore sequencing bias such as GC content bias and assume all sites have same probability. The probability that an insert of length l starts at s is

$$\begin{aligned} p_S(s) &= \frac{1}{\tilde{T}(l)} \\ &= \frac{1}{\sum_{c \in \{contigs\}} (l_c - l + 1)} \end{aligned}$$

  where $\tilde{T}(l)$ is the total effective length *i.e.* number of possible start sites for insert size l and l_c is the length of contig c.

- **Error Model:** The error model described in [132] is used and learnt using reads that map uniquely to contigs.

The scaffold graph

The next phase of our algorithm is to construct the *scaffold graph*. The *scaffold graph* is a weighted bidirected graph [36] where there is a vertex for each contig. There is an edge between two contigs if joining them would result in an increase in likelihood of the assembly. The edges are bidirected as each contig in the pair may correspond to one of the two DNA strands resulting in four possible orientations of the pair of contigs. There may be more than one edge between two contigs provided the edges are of different types. The edge weights correspond to increase in likelihood achieved if the contigs were to be joined. We also have a gap estimate between two contigs associated with each edge. Computing the edge weights is done in following two steps.

- Estimate the gaps between pairs of contigs using maximum likelihood. Gaps may be negative if contigs overlap.

- Compute the change in assembly likelihood if contigs are linked with maximum likelihood gap estimates. This constitutes computing probability of linking reads as well as adjusting probabilities of all other reads.

Estimating gaps using maximum likelihood

The gaps between contigs are estimated using a generative model similar to the one used for computing likelihood of assembly. However, we modify the generative model to correct for the issue that we may not observe inserts from the entire distribution due to gap between contigs [17] and lengths of contigs.

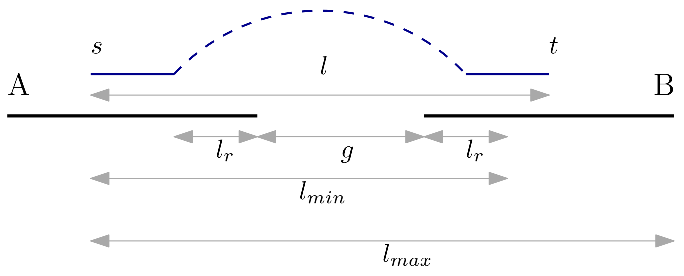


Figure 3.2: **Gap estimation.** Figure illustrates that inserts smaller than l_{min} and greater than l_{max} will not be observed due to lengths of contigs A and B and the gap g between them.

Consider contigs A and B separated by a gap g in Figure 3.2. If the $5'$ end of an insert is at s, then we will not observe inserts smaller than l_{min} and greater than l_{max} where l_r is the length of the second read of the pair. The probability of a read is then given by

$$p(r) \approx \sum p'_S(s)p'_F(l)p_E(r|a)$$

where p'_F and p'_S are the corrected insert size and start site distributions respectively given by

$$\begin{aligned} p'_F(l) &= \frac{p_F(l)}{\sum_{k=l_{min}}^{l_{max}} p_F(k)}, \\ p'_S(s) &= \frac{\text{p\{fragment starting at s and ending in B\}}}{\text{p\{fragment starting in A and ending in B\}}}. \end{aligned}$$

While p'_F can be efficiently pre-computed, computation of p'_S is time consuming. So we make the following approximation.

$$p'_S(s) \approx \frac{\text{p\{fragment starting at s and ending at t\}}}{\text{p\{fragment starting in A and ending at t\}}}.$$

We then find the gap, g that maximizes likelihood of linking reads, $\max_g \prod_{r\in\{linking\}} p(r)$ for every pair of contigs with read pairs linking them.

Resolving multi-mapped pairs using the EM algorithm

Application of a probabilistic model allows us to resolve read pairs with multiple mappings using the *expectation maximization* (EM) algorithm [30, 154] which are ignored in most other scaffolders. In the initialization step, we probabilistically assign read pairs to contig pairs using only the error probabilities and obtain gap estimates as discussed above. We then iterate the two steps of the EM algorithm.

- *E-step:* Determine the expected assignments of read pairs to contig pairs according to the generative model with current gap estimates.
- *M-step:* Find maximum likelihood estimates of gaps using the current assignments.

In early iterations when the gap estimates are inaccurate, we use relaxed cut-offs for insert size distribution and divide by two in each iteration to converge towards the cut-off points determined by standard deviations.

Computing edge weights

We then approximate the change in assembly likelihood if two contigs are joined with the MLE of gap. The likelihood of linking reads can be computed using the generative model for sequencing (Figure 2.1). However, we also need to adjust probabilities of all other reads since the effective length changes. Recall that the likelihood of an assembly is given by

$$l(\mathcal{A};\mathcal{R}) \approx \sum_{i=1}^{N} \log \sum_{j=1}^{M_i} p_F(l_{i,j}) \frac{1}{\tilde{T}(l_{i,j})} p_E(r_i|a_{i,j}).$$

Adjusting the probabilities using the above equation would require iterating over of all reads with more than one mapping. In order to make this step practical we make the following approximation.

$$\begin{aligned} l(\mathcal{A};\mathcal{R}) &\approx \sum_{i=1}^{N} \log \frac{1}{\tilde{T}(\hat{l}_i)} \sum_{j=1}^{M_i} p_F(l_{i,j}) p_E(r_i|a_{i,j}) \\ &= \sum_{i=1}^{N} \log \frac{1}{\tilde{T}(\hat{l}_i)} + \sum_{i=1}^{N} \log \sum_{j=1}^{M_i} p_F(l_{i,j}) p_E(r_i|a_{i,j}) \end{aligned}$$

where $\hat{l}_i$ is insert size corresponding to most probable mapping of read i. This allows us to count number of reads corresponding to a particular insert size and efficiently calculate the new likelihood. If $n_{\hat{l}}$ is number of reads with insert size $\hat{l}$, then

$$l_{new}(\mathcal{A};\mathcal{R}) \approx l_{old}(\mathcal{A};\mathcal{R}) - \sum_{\hat{l}} n_{\hat{l}} \left(\log \frac{1}{\tilde{T}_{old}(\hat{l})} - \log \frac{1}{\tilde{T}_{new}(\hat{l})} \right)$$

where $\tilde{T}_{new}(l) = \tilde{T}_{old}(l) + l - 1 + g$ is the new effective length if both contigs are sufficiently large compared to insert sizes and can be precomputed for various gap sizes. The adjustments for cases where contigs are small can also be precomputed.

The likelihood of linking reads and likelihood adjustments of all other reads are combined to estimate the edge weight. We retain the edge and assign it the computed weight if it is positive and delete the edge otherwise.

Selecting joins

Once the scaffold graph is constructed, one approach might be to select the set of edges that maximizes likelihood. Analogous problems have been proved to be NP-hard [63] but fixed parameter tractable algorithms are known [42]. However, this may lead to incorrect joins if there are repeats longer than the insert sizes. So, we use the following heuristic instead.

i Make unambiguous joins and compute the standard deviation, σ_L of the likelihoods of the joins made.

ii Sort other candidate joins in decreasing order of likelihood.

iii Join contigs if likelihood, l of the join is $\alpha(l) = \max(5e^{\lambda l}, 2)$ times greater or $2.5\sigma_L$ more than all other conflicting joins where $\lambda = -\frac{\ln(2/5)}{5\sigma_L}$.

For the remainder of this section we say two joins are inconsistent or in conflict with each other if none of the contigs with the estimated gap fits into the gap between the other two times a stretch factor of 1.1 and neither has increase in likelihood higher than the other determined by the rule above.

The intuition behind the heuristic is to learn a distribution of likelihoods of unambiguous joins and make a join if there is no inconsistent join within two and a half standard deviations of it. But this leads to missing joins with increase in likelihood less than $2.5\sigma_L$ if there are other conflicting joins. So, we add a factor determined by an exponential decay function, α with $\alpha(0) = 5$ and $\alpha(l) = 2$ for $l > 5\sigma_L$.

Following exceptions are however made based on practical observations.

- Similar to some other scaffolders [28, 42], we compute the mean sequencing depth per base of contigs and the right sided standard deviation and avoid joining any contig with sequencing depth two standard deviations more than the mean and more than 1.5 times the mean.

- Gap estimates based on only linking read pair are often inaccurate and lead to incorrect joins. So, we do not join contigs linked by a single read pair.

Selecting from multiple consistent joins

In many cases although there are multiple outgoing or incoming edges from or to a contig, one or more contigs fit into the gap between the contigs to be joined. In these cases more than one join can be made and we use the following algorithm.

i Find all contigs with edges from the two contigs currently being joined that fit into interval determined by the estimated gap times a stretch factor.

ii Remove contigs with conflicting edges to other contigs.

iii Select consistent set of contigs that optimizes likelihood using the dynamic programming algorithm for Weighted interval scheduling.

iv Remove selected contigs inconsistent with not selected ones.

In actual implementation there may be two possible positions for each contig within the interval. This requires solving the Discrete weighted interval scheduling problem. However, the problem is known to be NP-hard [115] and we find that the two possible positions almost always overlap. So, we run the Weighted interval scheduling algorithm and remove one in the rare case both were selected.

Edge propagation

When we merge two contigs, if there are edges to contigs with gap estimates large enough so that the contig being joined to along with the gap can fit in there, we propagate the edge from the contigs to the meta-node corresponding to the merged contigs.

Special cases

Under usual circumstances, Swalo takes as input a single parameter denoting the minimum size of contigs to be used to learn the insert size distribution. But in some cases discussed below, special modes are recommended.

Inadequate number of inserts to learn the distribution:

If the contigs are small compared to insert sizes or if the number of inserts mapping concordantly to contigs is low (less than 100,000), the insert size distribution cannot be learnt properly. So, we recommend that the user switches to Gaussian distribution and provide a mean and a standard deviation as inputs.

Insert size standard deviation very high:

If the insert size standard deviation is very high (greater than 1000), then there are large errors in gap estimates. So, we switch to a conservative mode where only uniquely mapped reads are used and multiple consistent joining step is skipped.

Multiple insert size libraries:

There are two approaches for multiple insert size libraries. The first is to build the scaffold graph separately and then combine the graphs before actually merging contigs. The other approach is to do the scaffolding hierarchically. We recommend the first approach unless some insert libraries have large standard deviations in which case we recommend scaffolding using all other libraries using the first approach and then scaffold using libraries with high standard deviation together using the conservative mode.

Implementation

The methods have been implemented in a tool called 'scaffolding with assembly likelihood optimization (Swalo)' using C/C++. The gap estimation phase (the M-step of the EM algorithm) is mutli-threaded for speed-up in computation.

3.3 Results

Datasets

To compare performance of Swalo with other scaffolders, we use the datasets used by Hunt *et al.* to evaluate scaffolding tools [62]. The datasets include four simulated datasets from *S. aureus* and six real datasets from *S. aureus*, *R. sphaeroides*, *P. falciparum* and human chromosome 14. Among these the *S. aureus*, *R. sphaeroides*, and human chromosome 14 datasets were also part of the GAGE project [141]. Table 3.1 provides a summary of the datasets used. For more details on the datasets and how the contigs were generated, please see [62]. We analyze results using the scripts provided in [62] and when applicable use the same parameter values for mapping and scaffolding.

Simulated data

Table 3.2 summarizes performance of scaffolding tools on simulated datasets. We find that Swalo makes no incorrect joins for any of the datasets. For 100kb contigs Swalo was able to make 100% of the correct joins using either library and all aligners. When the insert size library of 500 was used to scaffold 3kb contigs, Swalo made 99.0%, 99.3% and 99.0% correct joins using Bowtie 2, Bowtie -v 0 and Bowtie -v 3 respectively. We find that the only scaffolder that makes more than 99.3% correct joins is Opera at 99.8% when used in conjunction with BWA however at the cost of making 0.2% incorrect joins. For 3kb contigs and 3kb insert size library, Swalo was provided as input mean and standard deviation of insert size library as it would not be possible to learn the distribution accurately from the mapped reads. Swalo made 99.6%, 99.8% and 99.6% correct joins using Bowtie 2, Bowtie -v 0 and Bowtie -v 3 respectively. No other scaffolder made more than 99.6% correct joins. It is worth pointing out that Swalo was able to make more correct joins when used with

Table 3.1: **Summary of datasets used to analyze performance of Swalo**

Reference	Size (Mb)	Number of contigs	Type	Number of reads (millions)	Read length	Insert size
S. aureus	2.8	28	Simulated	0.76	76	505
		28	Simulated	0.76	76	2795
		941	Simulated	0.76	76	505
		941	Simulated	0.76	76	2995
S. aureus (GAGE)	2.9	168	Real	3.5	37	3385
R. sphaeroides (GAGE)	4.6	571	Real	2.1	101	3695
P. falciparum	23.3	9303	Real	52.5	76	645
		9303	Real	12.0	75	2705
Human chromosome 14 (GAGE)	88.2	19936	Real	22.7	101	2865
		19936	Real	2.4	57–82	34500

Bowtie -v 0 compared to Bowtie -v 3 and Bowtie 2 which may be due to reads not being mapped to some regions when Bowtie -v 3 and Bowtie 2 are used.

Real data

Performance of Swalo in comparison to other scaffolders for real datasets is illustrated in Figure 3.3. For the *S. aureus* dataset from GAGE, we find that Swalo made more correct joins than all other scaffolders while making 2, 2 and 3 incorrect joins using Bowtie 2, Bowtie -v 0 and Bowtie -v 3 respectively. However, upon closer inspection we find that 2 joins in each case that are labeled incorrect are in fact joins from the end to the start of circular chromosomes or plasmids and are actually correct.

Similarly for *R. sphaeroides* dataset more correct joins are made by Swalo than all other scaffolders when used in conjunction with Bowtie 2. Again we find that 3 joins that are marked as incorrect are joins linking ends of circular chromosomes or plasmids to the start. The sequencing error rate for this dataset is high compared to the *S. aureus* dataset. So, number of reads mapped is quite low [62] for Bowtie -v 3 and particularly for Bowtie -v 0 resulting in much less correct joins made by Swalo and other scaffolders when Bowtie is used compared to Bowtie 2.

The *P. falciparum* genome is known to be hard to assemble due its low GC content. We find that in this case although Swalo does not make more correct joins than all other scaffolders as in other cases, the number of correct joins made are only slightly less than

Table 3.2: **Comparison of performance of scaffolders on simulated datasets.** None of the scaffolders made any incorrect joins for 100kb contigs. Values for all scaffolders except Swalo are from [62].

Scaffolder	Aligner	100kb contigs		3kb contigs			
		500b lib	3kb lib	500b lib		3kb lib	
		%correct	%correct	%correct	%wrong	%correct	%wrong
ABySS	abyss-map	100.0	66.7	98.9	0.0	99.5	0.0
Bambus2	Bowtie 2	100.0	66.7	63.7	0.0	95.9	0.0
	BWA	48.2	48.2	59.6	0.0	96.2	0.0
MIP	Bowtie -v 0	100.0	96.3	98.9	0.0	98.4	0.0
	Bowtie -v 3	100.0	96.3	98.4	0.0	97.9	0.0
	Bowtie 2	100.0	29.6	96.3	0.5	98.7	0.5
	BWA	100.0	33.3	98.0	1.1	98.2	0.4
Opera	Bowtie	100.0	92.6	98.4	0.0	1.0	10.0
	BWA	100.0	92.6	99.8	0.2	1.2	80.0
SCARPA	Bowtie -v 0	100.0	96.3	98.9	0.0	95.0	0.0
	Bowtie -v 3	100.0	96.3	98.6	0.0	96.3	0.0
	Bowtie 2	85.2	96.3	96.8	0.0	76.3	0.7
	BWA	85.2	92.6	96.6	0.0	77.9	0.4
SGA	Bowtie 2	100.0	96.3	97.3	0.0	97.6	0.0
	BWA	100.0	92.6	99.0	0.0	96.2	0.0
SOAP2	SOAP2	96.3	96.3	98.6	0.0	99.5	0.0
SOPRA	Bowtie -v 0	100.0	96.3	98.3	0.0	98.2	0.0
	Bowtie -v 3	100.0	96.3	97.2	0.0	97.2	0.0
	Bowtie 2	74.1	100.0	91.5	0.5	85.6	0.5
	BWA	74.1	88.9	92.9	0.2	83.1	0.4
SSPACE	Bowtie -v 0	100.0	92.6	99.1	0.0	99.6	0.0
	Bowtie -v 3	100.0	92.6	98.7	0.0	99.3	0.0
SWALO	Bowtie 2	100.0	100.0	99.0	0.0	99.6	0.0
	Bowtie -v 0	100.0	100.0	99.3	0.0	99.8	0.0
	Bowtie -v 3	100.0	100.0	99.0	0.0	99.6	0.0

that of SOPRA, MIP and SCARPA while number of incorrect joins is less than or similar to what SOPRA made and much less than the numbers for SCARPA and MIP. We observe that many of the contigs have strings of consecutive ‘A’s or ‘T’s where very few reads are mapped to by aligners leading to poor gap estimates.

Finally, for the combined human chromosome 14 dataset, Swalo makes more correct joins than all other scaffolders except SOAP2 which makes many more incorrect joins compared to Swalo. Figure B.1 shows that the long jumping library is in fact a mixture inserts of two sizes. When the two libraries are mapped separately and used to estimate gaps before

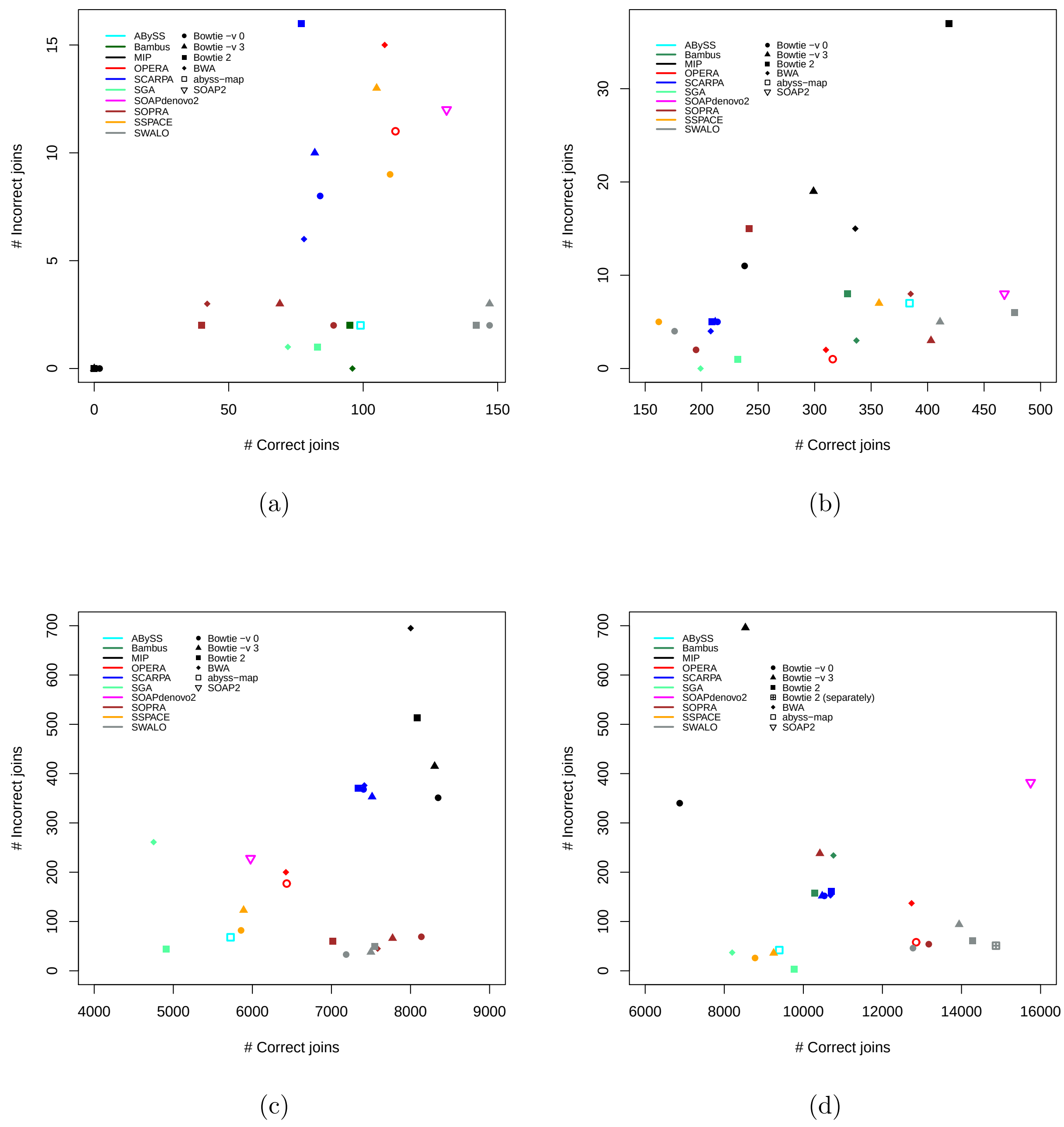


Figure 3.3: **Performance of scaffolders.** Scatter plots showing number of correct joins vs incorrect joins made by different scaffolders on (a) *S. aureus* data, (b) *R. sphaeroides* data, (c) *P. falciparum* combined short and long insert data, and (d) human chromosome 14 combined long insert and fosmid library data. Up to 2 and 3 joins in (a) and (b) respectively made by SWALO (and possibly other scaffolders) labelled incorrect are joins from end to start of circular chromosomes and are therefore correct. Values for all scaffolders except SWALO are from [62].

scaffolding, and the fosmid library is applied on the output, the results improve both in terms of increase in number of correct joins and decrease in number of incorrect joins.

Overall we find that Swalo outperforms all other scaffolders on real and simulated datasets. We observe that consistent results are achieved when it is used with Bowtie 2. However, when reads are largely error free results achieved using Bowtie -v 0 can be better possibly due to reads being mapped to more regions compared to Bowtie 2.

Although a comparison of running times is not appropriate since Swalo was run on a different machine to other scaffolding tools, but we would like to note that Swalo took from a few minutes for *S. aureus* datasets to less than two hours for combined human chromosome 14 dataset to run when used with Bowtie 2 (excluding the time required for mapping) using 32 cores. This makes it efficient and applicable to practical datasets.

3.4 Conclusions

We presented a scaffolding method, Swalo which is based on generative models to approximate whether joining contigs would lead to an increase in assembly likelihood. Experiments with real and simulated data used in [62] suggest Swalo outperform all other scaffolding tools. The method may further be improved by modifying the heuristic used to select among multiple candidate joins and by considering global properties of the scaffold graph. The improvement in scaffolding achieved by a practical method based on assembly likelihoods opens up the possibility that other problems related to assembly such as reference guided assembly, mis-assembly correction, copy number estimation may also be amenable to this approach.

Part II

Genome Analysis

Chapter 4

Association mapping from sequencing reads using k-mers

4.1 Introduction

Association mapping refers to linking of genotypes to phenotypes. Most often this is done using a genome-wide association study (GWAS) with single nucleotide polymorphisms (SNPs). Individuals are genotyped at a set of known SNP locations using a SNP array. Then each SNP is tested for statistically significant association with the phenotype. In recent years thousands of genome-wide association studies have been performed and regions associated with traits and diseases have been located.

However, this approach has a number of limitations. Firstly, designing SNP arrays requires knowledge about the genome of the organism and where the SNPs are located in the genome. This makes it hard to apply to study organisms other than human. Even the human reference genome is incomplete [2] and association mapping to regions not in the reference is difficult. Secondly, structural variations such as insertion-deletions (indels) and copy number variations are ignored in these studies. Despite the many GWA studies that have been performed a significant amount of heritability is yet to be explained. This is known as the "missing heritability" problem [172]. A hypothesis is some of the missing heritability is due to structural variations. Thirdly, the phenotype might be caused by rare variants which are not on the SNP chip. In last two cases, follow up work is required to find the causal variant even if association is detected in the GWAS.

Some of these limitations can be overcome by utilizing high throughput sequencing data. As sequencing gets cheaper association mapping using next generation sequencing may become feasible. The current approach to doing this is to map all the reads to a reference genome followed by variant calling. Then these variants can be tested for association. But this again requires a reference genome and it may induce biases in variant calling and regions not in the reference genome will not be included. Moreover, genotype calling is complicated when sequencing depth is low due to sequencing errors [117] and in repetitive regions. Methods have been proposed to do population genetics analyses that avoid the genotype calling step [40, 39] but these methods still require reads to be aligned to a reference genome. An alternate approach is simultaneous *de novo* assembly and genotyping using a tool such as Cortex [65] but this is not suited to large number of individuals. Furthermore, both these approaches are computationally very expensive.

In the past, alignment free methods have been developed for a number of problems including transcript abundance estimation [122], sequence comparison [150], phylogeny estimation [53], etc. Nordstrom *et al.* introduced a pipeline called needle in the k-stack (NIKS) for mutation identification by comparison of sequencing data from two strains using k-mers [119]. Here we present an alignment free method for association mapping. It is based on counting k-mers and identifying k-mers associated with the phenotype. The overlapping k-mers found are then assembled to obtain sequences corresponding to associated regions. Our method is applicable to association studies in organisms with no or incomplete reference genome. Even if a reference genome is available, this method avoids aligning and genotype calling thus allowing association mapping to many types of variants using the same pipeline and to regions not in the reference.

We have implemented our method in a software called 'hitting associations with k-mers' (HAWK). Experiments with simulated and real data demonstrate the promises of this approach. We leave taking into account confounding factors such as population structure as future work and apply our method to analyze sequencing data from three populations in the 1000 genomes project treating population identity as the trait of interest. Agreement with sites found using read alignment and genotype calling indicate that k-mer based association mapping will be applicable to studying disease associations.

4.2 Methods

Association mapping with k-mers.

We present a method for finding regions associated with a trait using sequencing reads without mapping reads to reference genomes. The workflow is illustrated in Fig 4.1. Given sequencing reads from case and control samples, we count k-mers appearing in each sample. We assume the counts are Poisson distributed and test k-mers for statistically significant association with case or control using likelihood ratio test for nested models (see Appendix C for details). The differences in k-mer counts may be due to single nucleotide polymorphisms (SNPs), insertion-deletions (indels) or copy number variations. The k-mers are then assembled to obtain sequences corresponding to each region.

Counting k-mers

The first step in our method for association mapping from sequencing reads using k-mers is to count k-mers in sequencing reads from all samples. To count k-mers we use the multi-

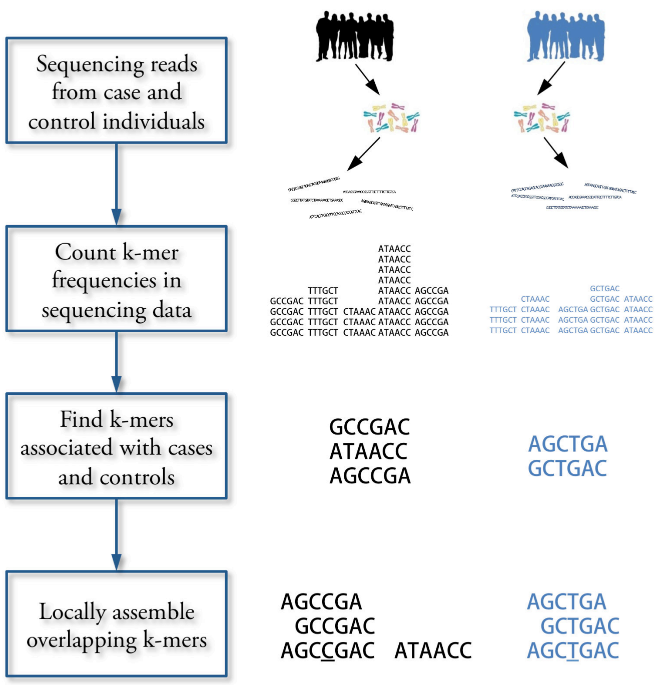

Sequencing reads from case and control individuals
Count k-mer frequencies in sequencing data
Find k-mers associated with cases and controls
Locally assemble overlapping k-mers
GCCGAC
ATAACC
AGCCGA
AGCTGA
GCTGAC
AGCCGA
GCCGAC
AGCCGAC ATAACC
AGCTGA
GCTGAC
AGCTGAC

Figure 4.1: **Workflow for association mapping using k-mers.** The HAWK pipeline starts with sequencing reads from two sets of samples. The first step is to count k-mers in reads from each sample. Then k-mers with significantly different counts in two sets are detected using likelihood ratio test. Finally, overlapping k-mers are assembled into sequences to get one or few sequences for each associated locus.

threaded hash based tool JELLYFISH developed by Marcais and Kingsford [97]. We use k-mers of length 31 and ignore k-mers that appear once in a sample for computational and memory efficiency as they likely are from sequencing errors.

Finding significant k-mers

Then for each k-mer we test whether that k-mer appears significantly more times in case or control datasets compared to the other using a likelihood ratio test for nested models. Suppose, a particular k-mer appears K_1 times in cases and K_2 times in controls, and N_1 and N_2 are the total number of k-mers in cases and controls respectively. The k-mer counts are assumed to be Poisson distributed with rates θ_1 and θ_2 in cases and controls. The null hypothesis is $H_0 : \theta_1 = \theta_2 = \theta$ and the alternate hypothesis is $H_1 : \theta_1 \neq \theta_2$. The likelihoods under the alternate and the null are given by (see Appendix C for details)

$$L(\theta_1, \theta_2) = \frac{e^{-\theta_1 N_1}(\theta_1 N_1)^{K_1}}{K_1!} \frac{e^{-\theta_2 N_2}(\theta_2 N_2)^{K_2}}{K_2!}$$

and

$$L(\theta) = \frac{e^{-\theta N_1}(\theta N_1)^{K_1}}{K_1!} \frac{e^{-\theta N_2}(\theta N_2)^{K_2}}{K_2!}.$$

Since the null model is a special case of the alternate model, $2 \ln \Lambda$ is approximately chi-squared distributed with one degree of freedom where Λ is the likelihood ratio. We get a p-value for each k-mer using the approximate χ^2 distribution of the likelihood ratio and perform Bonferroni corrections to account for multiple testing.

Merging k-mers

We then merge overlapping k-mers to get a sequence for each differential site using the assembler ABySS [147]. ABySS was used as the assemblies it generated were found to cover more of the sequences to be assembled compared to other assemblers [132]. We construct the de Bruijn graph using hash length of 25 and retain assembled sequences of length at least 49. It is also possible to merge k-mers and pair sequences from cases and controls using the NIKS pipeline [119]. However, we find that this is time consuming when we have many significant k-mers. Moreover, when number of cases and controls are not very high we do not have enough power to get both of sequences to be paired and as such pairing is not possible.

Implementation

Our method is implemented in a tool called ‘hitting associations with k-mers’ (HAWK) using C++. To speed up the computation we use a multi-threaded implementation. In addition, it is not possible to load all the k-mers into memory at the same time for large genomes. So, we sort the k-mers and load them into memory in batches. To make the sorting faster Jellyfish has been modified to output internal representation of k-mers instead of the k-mer strings. In future the sorting step may be avoided by utilizing the internal ordering of Jellyfish or other tools for k-mer counting.

Downstream analysis

The sequences can then be analyzed by aligning to a reference if one is available or by running BLAST [3] to check for hits to related organisms. The intersection results in this paper were obtained by mapping them to the human reference genome version GRCh37 using Bowtie2 [80] to be consistent with co-ordinates of genotypes called by 1000 genomes project. The breakdown analysis was performed by first mapping to the latest version of the reference, hg38 and then running BLAST on some of the ones that did not map. Specific loci of interest were checked by aligning them to RefSeq mRNAs using Bowtie 2 and on the UCSC Human Genome Browser [74] by running BLAT [72].

4.3 Results

Verification with simulated data.

The implementation was tested by simulating reads from genome of an *Escherichia coli* strain. We introduced different types mutations - single nucleotide changes, short indels (less than 10bp) and long indels (between 100bp and 1000bp) into the genome. Then `wgsim` of `SAMtools`[84] was used to first generate two sets of genomes by introducing more random mutations (both substitutions and indels) into the original and the modified genomes and then simulate reads with sequencing errors. The HAWK pipeline was then run on these two sets of sequencing reads. The fraction of mutations covered by resulting sequences are shown in Figure 4.2 for varying numbers of case and control samples and different types of mutations. The results are consistent with calculation of power to detect k-mers for varying total k-mer coverage (Figure C.1) with slightly lower values expected due to sequencing errors and conditions imposed during assembly.

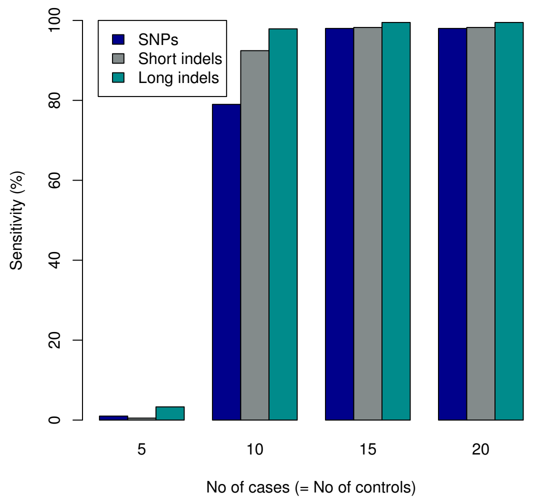


Figure 4.2: **Sensitivity with simulated *E. coli* data.** The figure shows sensitivity for varying number of case and control samples for different types of mutations. Sensitivity is defined as the percentage of differing nucleotides that are covered by a sequence. All of the sequences covered some location of mutation.

Verification with 1000 genomes data.

To analyze the performance of the method on real data we used sequencing reads from the 1000 genomes project [4]. The population identities were used as the phenotype of interest circumventing the need for correction of population structure. For verification, we used sequencing reads from 87 YRI individuals and 98 TSI individuals for which both sequencing reads and genotype calls were available at the time analysis was performed.

The analysis using k-mers resulted in 2,970,929 sequences associated with YRI samples and 1,865,285 sequences of significant association with TSI samples. We also performed similar analysis with genotype calls. VCFtools [26] was used to obtain number of individuals with 0, 1 and 2 copies of one of the alleles for each SNP site. Each site was then tested to check whether the allele frequencies are significantly different in two samples using likelihood ratio test for nested models for multinomial distribution (see Appendix C for details). We found that 2,658,964 out of the 39,706,715 sites had allele frequencies that are significantly different.

Figure 4.3 shows the extent of overlap among these discarding the sequences that did not map to the reference. We find that 80.3% (2,135,415 out of 2,658,964) of the significant sites overlapped with a sequence. We also find that approximately 95.2% of them overlapped with at least one k-mer.

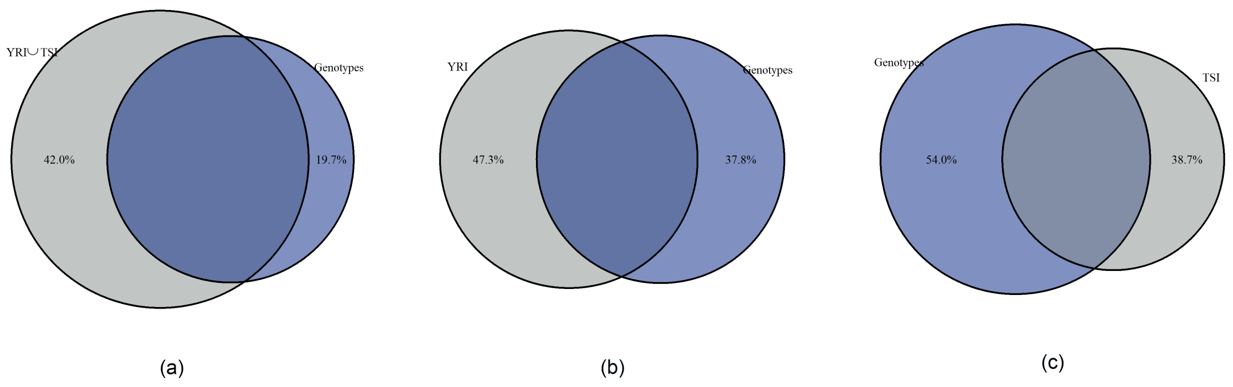


Figure 4.3: **Intersection analysis** Venn diagrams showing intersections among sequences obtained using Hawk and significant sites found by genotype calling. 'YRI' and 'TSI' denote sequences of significant association with YRI and TSI samples and 'Genotypes' denote the sites where allele frequencies are significantly different in two populations.

It was observed that around 42% of sites found using k-mers do not overlap with any sites found significant using genotype calling. While upto 20% of them correspond to regions for which we did not have genotype calls (chromosome Y, mitochondrial DNA and small contigs), repetitive regions where genotype calling is difficult and structural variations, many of the remaining sequences are possibly due to more power of the test based on counts than the one using only number of copies of an allele. We performed Monte Carlo simulations to determine powers of the two tests. Figure C.2 shows the fraction of trials that passed the p-value threshold after Bonferroni correction as the allele frequencies in cases were increased keeping the allele frequencies of control fixed at 0.

This is consistent with greater fraction of sequences in YRI (47.3%) not overlapping with sites obtained by genotyping compared to TSI (38.7%) as some low frequency variations in African populations were lost in other populations due to population bottleneck during the migration out of Africa. However, some false positives may result due to discrepancies in sequencing depth of the samples and sequencing biases. We provide scripts to lookup number of individuals with constituent k-mers and leave dealing with these confounding factors as well as population structure as future work.

Table 4.1 shows p-values of some of the well known sites of variation between African and European populations.

Table 4.1: **Known variants in YRI-TSI comparison.**

Gene	**SNP ID**	**Description**	**Allele**	**p-value**	**%YRI**	**%TSI**
ACKR1	rs2814778	Duffy antigen	C	9.72×10^{-114}	84.39%	1.78%
SLC24A5	rs1426654	Skin pigmentation	G	8.45×10^{-144}	87.39%	1.02%
SLC45A2	rs16891982	Skin/hair color	C	1.89×10^{-122}	92.18%	4.67%
G6PD	rs1050829	G6PD deficiency	C	1.53×10^{-29}	24.92%	1.02%
G6PD	rs1050828	G6PD deficiency	T	5.83×10^{-25}	18.32%	0.00%

Table shows p-values of sequences at some well known sites of variation between populations. The (%) values denote fraction of individuals in the sample with the allele present. The p-values and % values are averaged over k-mers constituting the associated sequences.

HAWK maps associations to different types of variants

HAWK enables mapping associations to different types of variants using the same pipeline. Figure 4.4 shows breakdown of types of variants found associated with YRI and TSI populations. The 'Multiple SNPs/Structural' entries correspond to sequences of length greater than 61 (the maximum length of a sequence due to a single SNP with k-mer size of 31). In additions to SNPs we find associations to sites with indels and structural variations. Furthermore, we find sequences that map to multiple regions in the genome indicating copy number of variations or sequence variation in repeated regions where genotype calling is known to be difficult.

We performed similar analysis on sequencing reads available from 87 BEB and 110 TSI individuals from the 1000 genomes project and obtained 529,287 and 462,122 sequences as-

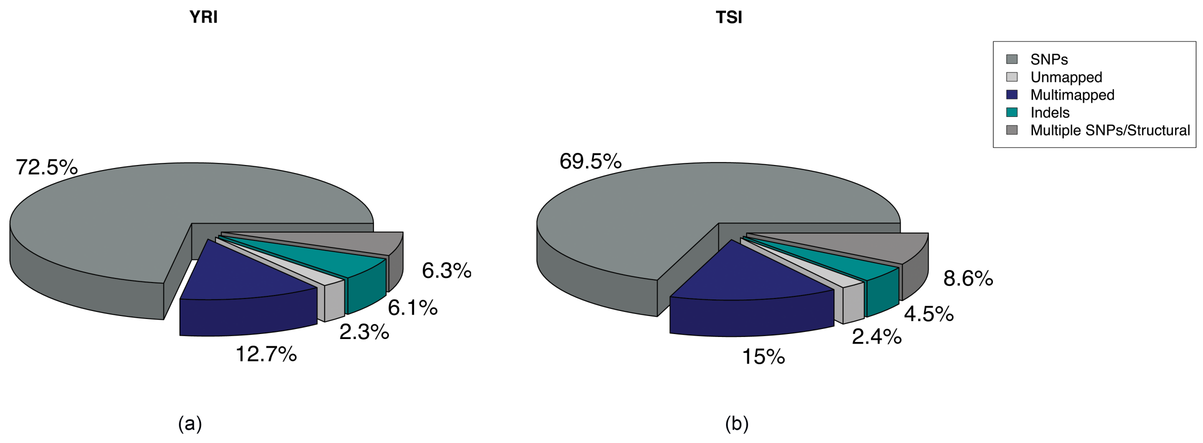


Figure 4.4: **Breakdown of types of variations in YRI-TSI comparison.** Pie charts showing breakdown of 2,970,929 and 1,865,285 sequences associated with YRI and TSI samples respectively. The 'Multiple SNPs/Structural' entries correspond to sequences of length greater than 61, the maximum length of a sequence due to a single SNP with k-mer size of 31 and 'SNPs' correspond to sequences of maximum length of 61.

sociated with BEB and TSI samples respectively, much fewer than the YRI-TSI comparison. Figure 4.5 shows breakdown of probable variant types corresponding to the sequences found associated with BEB and TSI samples.

Histograms of sequence lengths show (Figures C.3 and C.4) peaks at 61bp which is the maximum length corresponding to a single SNP for k-mer size of 31. We also see drops off after 98bp in all cases providing evidence for multinucleotide mutations (MNMs) reported in [51] since this is the maximum sequence length we can get when k-mers of size 31 are assembled with minimum overlap of 24.

HAWK reveals sequences not in the human reference genome.

As Hawk is an alignment free method for mapping associations, it is able to find associations in regions that are not in the human reference genome. The analysis resulted in 94,795 and 66,051 sequences of lengths up to 2,666bp and 12,467bp associated with YRI and TSI samples respectively that did not map to the human reference genome. Similarly BEB-TSI comparison yielded 19,584 and 18,508 sequences with maximum lengths of 1761bp and 2149bp associated with BEB and TSI respectively.

We found that few of the sequences associated with TSI samples some as long as 12kbp and 2kbp in comparisons against YRI and BEB respectively that mapped to the Epstein-Barr virus (EBV) genome, strain B95-8 [GenBank: V01555.2]. EBV strain B95-8 was used to transform B cells into lymphoblastoid cell lines (LCLs) in the 1000 Genomes Project and

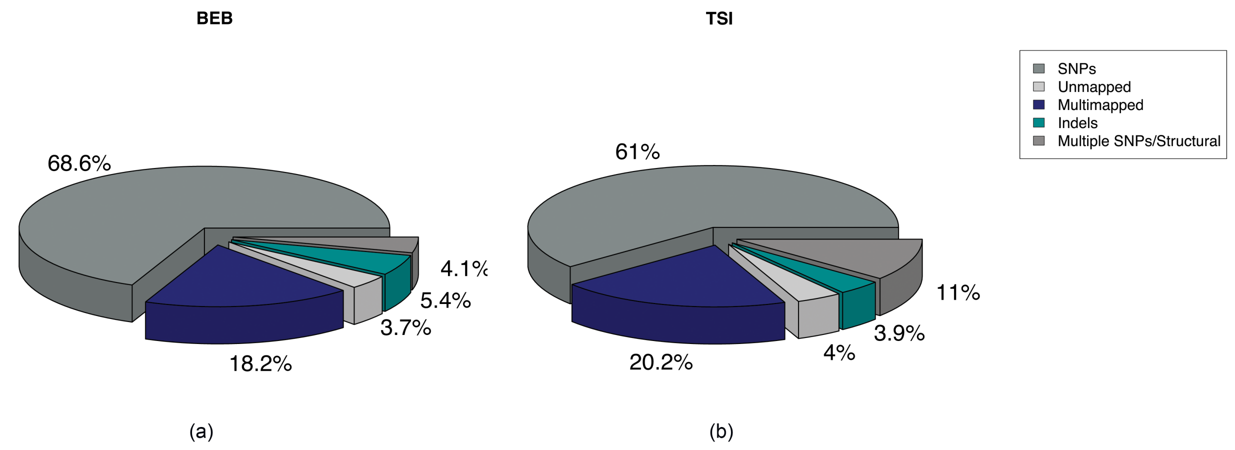


Figure 4.5: **Breakdown of types of variations in BEB-TSI comparison.** Pie charts showing breakdown of 529,287 and 462,122 sequences associated with BEB and TSI samples respectively. The 'Multiple SNPs/Structural' entries correspond to sequences of length greater than 61, the maximum length of a sequence due to a single SNP with k-mer size of 31 and 'SNPs' correspond to sequences of maximum length of 61.

is a known contaminant in the data [143].

Table 4.2 summarizes the sequences that could not be mapped to either the human reference genome or the Epstein-Barr virus genome using Bowtie2. Although an exhaustive analysis of all remaining sequences using BLAST is difficult, we find sequences associated with YRI that do not map to the human reference genome (hg38) with high score but upon running BLAST aligned to other sequences from human (for example to [GenBank: AC205876.2] and some other sequences reported by Kidd *et al.* [75]). We also find sequences with no significant BLAST hits to human genomic sequences some with hits to closely related species. Similarly, we find sequences associated with TSI aligning to human sequences such as [GenBank: AC217954.1] not in the reference. Although there much fewer long sequences obtained in the BEB-TSI comparison, we find sequences longer than 1kbp associated with each population with no BLAST hit.

Differential prevalence of variants in genes linked to CVDs in BEB-TSI comparison.

We noted that cardiovascular diseases (CVD) are a leading cause of mortality in Bangladesh [118] and age standardized death rates from CVDs in Bangladesh is higher compared to Italy [118, 162]. Moreover, South Asians have high rates of acute myocardial infarction (MI) or heart failure at younger ages compared to other populations and in several countries migrants from South Asia have higher death rates from coronary heart disease (CHD) at

Table 4.2: **Summary of sequences not in the human reference genome.**

Comparison	**Total no. sequences**	**No. sequences of length $\geq$ 1000bp**	**Total length in sequences of length $\geq$ 1000bp**	**No. sequences of length $\geq$ 200bp**	**Total length in sequences of length $\geq$ 200bp**
YRI (vs TSI)	94,795	41	59,956	478	225,426
TSI (vs YRI)	66,051	10	13,896	184	77,383
BEB (vs TSI)	19,584	3	3,835	75	33,954
TSI (vs BEB)	18,508	2	2,105	81	28,134

Table shows summary of sequences associated with different populations that did not map to the human reference genome (hg38) or to the Epstein-Barr virus genome.

younger ages compared with the local population [49, 68] and according to the INTERHEART Study, the mean age of MI among the poeple from Bangladesh is considerably lower than non-South Asians and the lowest among South Asians [166, 144]. This motivated us to explore probable underlying genetic causes.

The sequences of significant association with the BEB sample were aligned to RefSeq mRNAs and the ones mapping to genes linked to CVDs [70] were analyzed. Table 4.3 shows non-synonymous variants in such genes that are significantly more common in the BEB sample compared to the TSI sample. It is worth mentioning that the ‘C’ allele at the SNP site, rs1042034 in the gene *Apolipoprotein B (ApoB)* has been associated with increased levels of HDL cholesterol and decreased levels of Triglycerides [153] in individuals of European descent but individuals with the ‘CC’ genotype have been reported to have higher risk of CVDs in an analysis of the data from the Framingham Heart Study [77]. The SNP rs676210 has also been associated with a number of traits [111, 18]. Both alleles of higher prevalence in BEB at those sites have been found to be common in familial hypercholesterolemia patients in Taiwan [20]. On the other hand, prevalence of the risk allele, ‘T’ at rs3184504 in the gene *SH2B3* is higher in TSI samples compared to BEB samples.

We also observe a number of sites in the gene *Titin (TTN)* of differential allele frequencies in BEB and TSI samples (see Table C.1). However, *TTN* codes for the largest known protein and although truncating mutations in *TTN* are known to cause dilated cardiomyopathy [54, 151, 136], no such effect of other kinds of mutations are known.

4.4 Discussion

In this paper, we presented an alignment free method for association mapping from sequencing reads. It is based on finding k-mers that appear significantly more times in one set of

Table 4.3: **Variants in genes linked to cardiovascular diseases.**

Gene	SNP ID	Variant type	Allele	p-value	%BEB	%TSI
APOB	rs2302515	Missense	C	1.30×10^{-12}	29.29%	8.37%
APOB	rs676210	Missense	A	7.73×10^{-25}	72.93%	33.08%
APOB	rs1042034	Missense	C	2.28×10^{-23}	68.67%	31.91%
CYP11B2	rs4545	Missense	T	1.31×10^{-28}	31.33%	0.91%
CYP11B1	rs4534	Missense	T	9.36×10^{-36}	33.00%	0.91%
WNK4	rs2290041	Missense	T	1.53×10^{-14}	13.24%	0.47%
WNK4	rs55781437	Missense	T	1.30×10^{-12}	15.21%	0.91%
SLC12A3	rs2289113	Missense	T	7.40×10^{-13}	8.14%	0.00%
SCNN1A	rs10849447	Missense	C	8.67×10^{-12}	62.88%	39.92%
ABO	-	4bp (CTGT) deletion	-	1.17×10^{-13}	29.15%	10.55%
ABO	rs8176741	Missense	A	2.06×10^{-16}	27.70%	8.45%
SH2B3	rs3184504	Missense	C	8.22×10^{-23}	92.88%	63.87%
RAI1	rs3803763	Missense	C	1.32×10^{-12}	75.86%	51.17%
RAI1	rs11649804	Missense	A	1.95×10^{-19}	81.57%	52.79%

Variants in genes linked to cardiovascular diseases found to be significantly more common in BEB sample compared to TSI sample. The (%) values denote fraction of individuals in the sample with the allele present. The p-values and % values are averaged over k-mers constituting the associated sequences.

samples compared to the other and then locally assembling those k-mers. Since this method does not require a reference genome, it is applicable to association studies of organisms with no or incomplete reference genome. Even for human our method is advantageous as it can map associations in regions not in the reference or where variant calling is difficult.

We tested our method by applying it to data from the 1000 genomes project and comparing the results with the results obtained using the genotypes called by the project as well as using simulated data. We observe that more than 80% of the sites found using genotype calls are covered by some sequence obtained by our method while also mapping associations to regions not in the reference and in repetitive areas. Moreover, simulations suggest tests based on k-mer counts have more power than those based number of copies of an allele present at some site.

Breakdown analysis of the sequences found in pairwise comparison of YRI, TSI and BEB, TSI samples reveals that this approach allows mapping associations to SNPs, indels, structural and copy number variations through the same pipeline. In addition we find 2-4% of associated sequences are not present in the human reference genome some of which are longer than 1kbp. The YRI, TSI comparison yields almost 60kbp sequence associated with the YRI samples in sequences of length greater than 1kbp alone. This indicates populations around the world have regions in the genome not present in the reference emphasizing the importance of a reference free approach.

We explored variants in genes linked to cardiovascular diseases in the BEB, TSI comparison as South Asians are known to have a higher rate mortality from coronary heart diseases compared to many other populations. We find a number of non-synonymous mutations in those genes are more common in the BEB samples in comparison to the TSI ones underscoring the importance of association studies in diverse populations. The SNP rs1042034 in the gene *Apolipoprotein B (ApoB)* merits particular mention as the CC genotype at that site has been associated with higher risk of CVDs.

The results on simulated data and real data from the 1000 genomes project provide a proof of principle of this approach which has motivated us to extend this method to quantitative phenotypes and correct for population structure and other confounding factors and then apply it to association studies of disease phenotypes in humans and other organisms.

Chapter 5

Conclusion

In this dissertation, we showed that substantial improvements in genome assembly and analysis can be achieved through the use of statistical models. In Part I, we focused on two important problems related to genome assembly – evaluating genome assemblies and scaffolding contigs using read pairs. Genome assembly is one of the most widely studied problems in computational biology. To deal with the computational intractability of the problem primarily due to repeats, as well as practical issues such as sequencing errors, biases and volume of data generated by next-generation sequencing technologies, numerous methods have been developed based on many heuristics and approaches. The emphasis on contiguity based measures to evaluate assemblies and largely ignoring mis-assemblies and completeness of assembly mean many of the assembled genomes have errors in them and there have been calls for more principled ways to evaluate assemblies. The short lengths of next-generation sequencing reads have also complicated resolving repeat structure of genomes using only single end reads and made utilization of read pairs to scaffold contigs increasingly important.

We take advantage of the low cost and large volume of NGS data available and address these issues through use of statistical models. In Chapter 2, we present a generative model for sequencing that takes into account randomness in read generation process, insert size distribution and sequencing errors. We have developed a method called CGAL that learns distributions from the data and computes likelihood of an assembly of generating the given set of reads. Experiments with real and simulated data including GAGE and Assemblathon 1 datasets indicate that likelihood values more accurately assess assemblies compared to other approaches commonly used.

It has been suggested that scaling likelihood values by number of reads and genome size would allow comparison across genomes and different datasets using the scaled values. However, we observe that quality of reads generated varies substantially across datasets making comparison across datasets very difficult.

There have also been other efforts to evaluate assemblies using likelihood values [21, 43]. Likelihood values in addition to being useful for comparison of different assemblies generated by various assemblers and choosing input parameters to assemblers, provide an optimization criteria for genome assemblies. Assemblers based on maximum likelihood such as Piper [12], GAML [8] and one using a Bayesian approach [58] demonstrate that better assemblies can be achieved using statistical approaches although the methods presented do not scale to large genomes.

In Chapter 3, we presented a scaffolding method based on assembly likelihood. We use statistical models to approximates the change in likelihood if contigs are joined while employing necessary approximations to make it efficient. The method is implemented in a tool called SWALO. The performance of SWALO is compared with other scaffolders on datasets used in an evaluation paper and the results indicate that SWALO outperforms all other scaffolders and show pareto-optimal performance. The performance may be improved further by modifying the heuristic used to select from multiple candidate joins and by considering global properties of the scaffold graph.

The performance of a practical scaffolder based on assembly likelihood demonstrate that assemblies from next generation sequencing reads can be improved – Bresler arrived at a similar conclusion [12]. Statistical models may also be applied to other problem related to genome assembly – reference guided or assisted assembly [45, 145, 133], correction of misassemblies in addition to detection [126, 21, 61], copy number estimation [19, 165, 50], etc.

While we have made strides in improving genome assemblies, we also observe that some genomic analyses can be performed without requiring prior sequencing of a reference genome. In Part II of this dissertation, we presented a method for association mapping from sequencing reads that does not need a reference genome. The method is based on testing for associations between k-mers and the phenotype of interest and locally assembling the k-mers found significant. The implementation called HAWK is able to map associations to sequence and structural variation as well as to regions not in reference genomes.

Application of the method to analyze data from the Toscani in Italia (TSI), the Yoruba in Ibadan, Nigeria (YRI) and the Bengali from Bangladesh (BEB) populations from 1000 genomes project reveals sequences associated with each population that are not in the human reference genome. We also explored possible genetic basis of high rates of mortality due to cardiovascular diseases (CVD) among South Asians and find significant differences in allele frequencies between BEB and TSI samples at the sites rs1042034 and the nearby rs676210 in the *Apolipoprotein B (ApoB)* gene and a number of other non-synonymous variants in genes linked to CVDs. Individuals with the 'CC' genotype at rs1042034 have been reported to be at higher risks of CVDs at earlier ages and need to be followed up on.

HAWK promises to be an efficient, reference-free method for association mapping from sequencing reads that can map to associations to multiple kinds of variants without explicitly calling them. However, to apply this method to mapping associations to diseases, corrections of population structure and other confounding factors need to be incorporated and are the subjects of our future work.

Appendix A

Supplementary information for computing genome assembly likelihoods

The Assemblers

The following assemblers were used for assembling reads from *E. coli* and *G. clavigera*:

- ABySS-1.2.7
- Euler-sr.1.1.2
- SOAPdenovo-V1.05
- Velvet_1.1.04

Supplementary information for *E. coli* data

- Organism: *Escherichia coli*
- Total reads: 10408224
- Reads mapped using: Bowtie 2 - 2.0.0 - beta 6

Reference

- Identifier: [NCBI: U00096.2]
- Likelihood: -238572967.687551
- Number of reads mapped: 10305539

- Total length: 4639675 bases

Assemblies

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	775	-24.43×10^7	10239632	4716697	106014	106014	41232
22	577	-24.28×10^7	10252838	4655390	105565	105565	35940
23	466	-24.23×10^7	10260818	4668411	119098	113303	32206
24	423	-24.33×10^7	10258113	4662049	119103	119103	34380
25	385	-24.29×10^7	10260950	4647968	119103	119103	33392
26	296	-24.16×10^7	10267506	4776771	119289	119289	29411
27	290	-24.19×10^7	10270072	4655849	113303	113303	26690
28	268	-24.21×10^7	10269192	4653402	127397	127397	27186
29	251	-24.18×10^7	10267174	4655844	119060	119060	30500
30	243	-24.20×10^7	10262735	4648199	119061	119061	34214
31	200	-24.11×10^7	10273191	4660482	127380	127380	26299
32	183	-24.07×10^7	10278821	4720013	127449	127449	18566
33	191	-24.14×10^7	10274370	4715256	127449	127449	19629
34	184	-24.11×10^7	10275712	4677901	127449	127449	18549
35	187	-24.14×10^7	10271870	4678986	119691	114019	18551
36	1357	-29.88×10^7	9694443	4781873	14486	9005	58656

Table A.1: Details of ABySS assemblies of *E. coli*

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
23	8721	-79.29×10^7	5450331	4173530	941	941	596466
24	3915	-50.08×10^7	7963793	4546541	2675	2675	265291
25	1556	-35.52×10^7	9294984	4714101	6828	6828	159086
26	818	-32.21×10^7	9686807	4867555	13445	13445	140369
27	567	-31.15×10^7	9777746	4627451	19731	19731	134725

Table A.2: Details of Euler-sr assemblies of *E. coli*

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	103	-30.55×10^7	9769728	4536683	125011	42063	141257
23	108	-28.42×10^7	9894505	4547233	132296	47267	124586
25	126	-27.65×10^7	9947906	4553034	125245	58712	115939
27	142	-27.27×10^7	9989334	4555420	125292	63059	111662
29	133	-26.57×10^7	10033409	4556756	125296	60809	106641
31	131	-26.12×10^7	10055340	4561246	132417	59343	105305
33	137	-26.23×10^7	10057803	4558913	130824	63523	103877

Table A.3: Details of SOAPdenovo assemblies of *E. coli*

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	146	-26.14×10^7	10073984	4558369	132072	39884	94001
23	152	-25.67×10^7	10114372	4558963	171787	47058	86735
25	148	-25.25×10^7	10153953	4561882	132844	50967	83227
27	137	-25.04×10^7	10176741	4561981	171642	60809	78998
29	138	-24.97×10^7	10180739	4564097	132509	64114	77588
31	137	-24.98×10^7	10183271	4564728	171679	64138	79649
33	143	-25.05×10^7	10174263	4568906	132976	39335	79423

Table A.4: Details of Velvet assemblies of *E. coli*

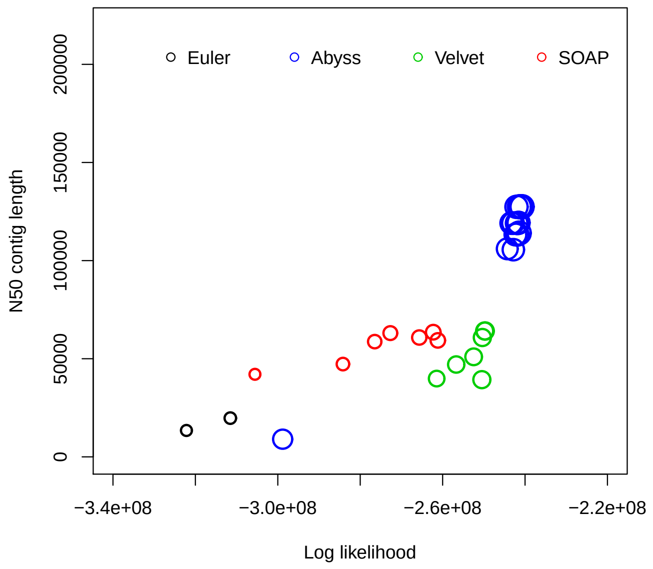


Figure A.1: Log likelihood vs N50 contig length for *E. coli*. Log likelihoods are shown on the x-axis and N50 contig lengths are shown on the y-axis. Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.9372048, (ii) log likelihood vs N50 contig length: 0.3316912, (iii) N50 contig length vs similarity: 0.4584904

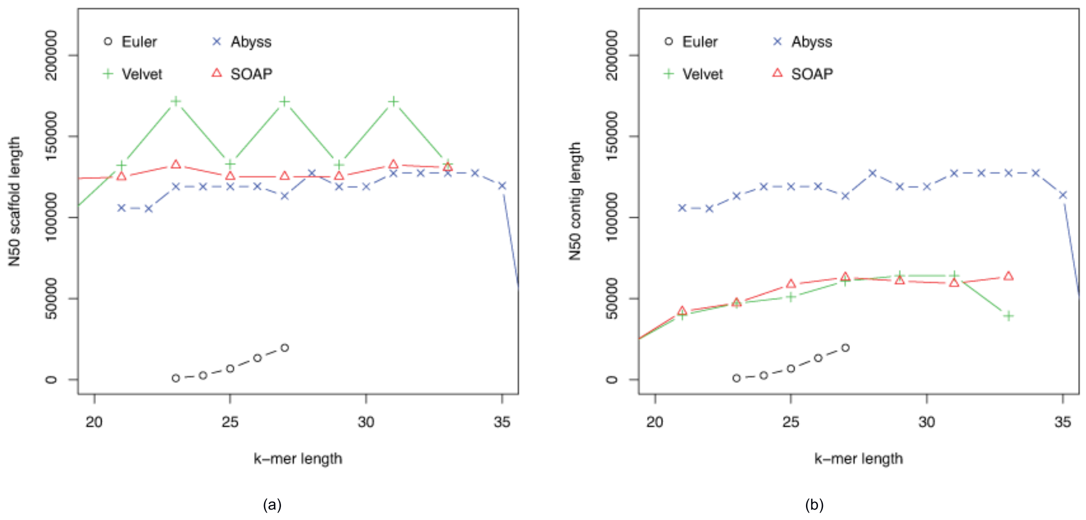
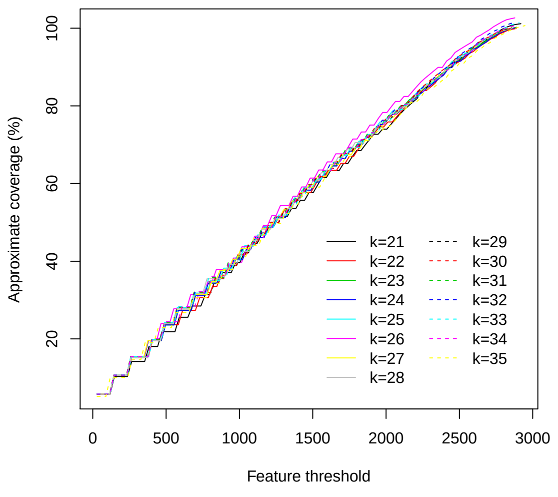
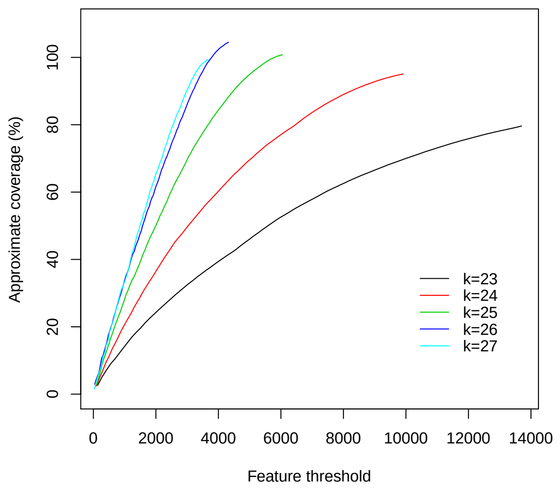


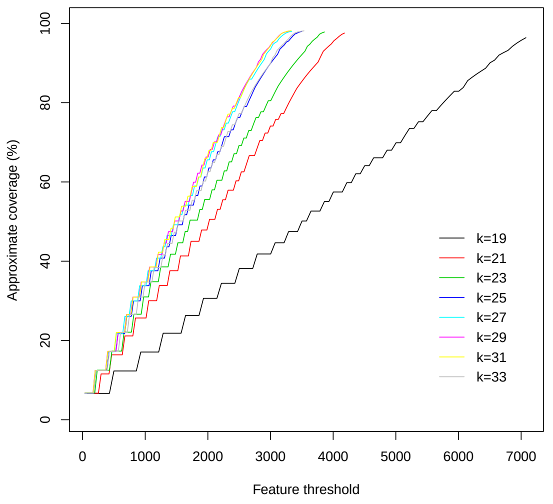
Figure A.2: (a) Hash length vs N50 scaffold length for *E. coli*, (b) Hash length vs N50 contig length for *E. coli*



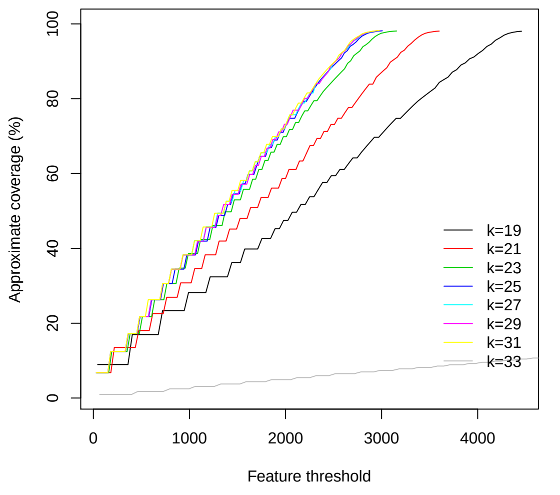
(a)



(b)



(c)



(d)

Figure A.3: Feature response curves for (a) ABySS, (b) Euler-sr, (c) SOAPdenovo, (d) Velvet assemblies of *E. coli*

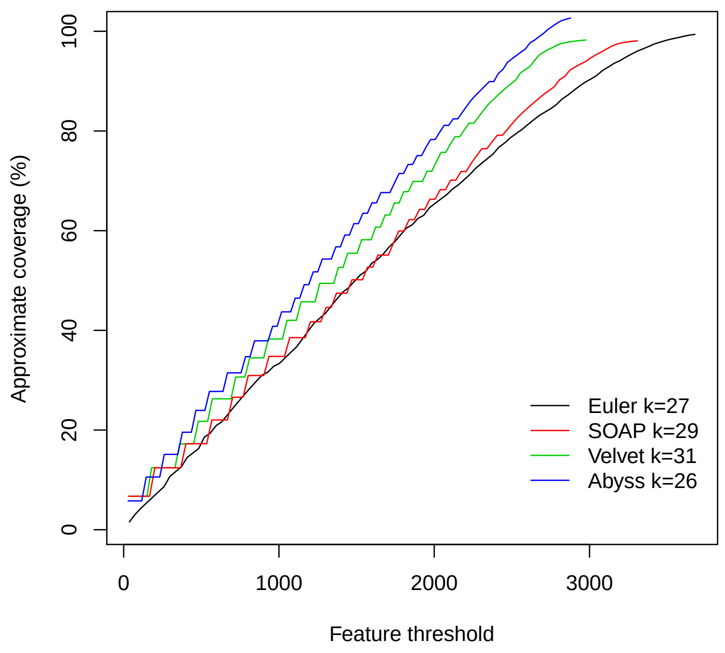


Figure A.4: Feature response curves for assemblies of *E. coli* by different assemblers

Supplementary information for ***E. coli*** data from CLC bio

- Organism: *Escherichia coli*
- Total reads: 2609307
- Reads mapped using: BFAST-0.6.5a

Reference - provided

- Identifier: [NCBI: NC_010473.1]
- Likelihood: -69204665.425366
- Number of reads mapped: 2523195
- Total length: 4686137 bases

Reference - conjectured

- Identifier: [NCBI: NC_000913.2]
- Likelihood: -57857459.428822
- Number of reads mapped: 2591992
- Total length: 4639675 bases

Assemblies

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	505	-59.21×10^6	2579014	4696557	82879	82879	40186
22	445	-58.99×10^6	2581729	4679494	95869	95869	34079
23	418	-59.16×10^6	2581745	4679963	92870	92870	35167
24	375	-58.98×10^6	2582234	4654094	89001	88061	34348
25	367	-58.94×10^6	2582400	4768360	88276	88276	34030
26	362	-59.16×10^6	2583036	4756869	78513	78513	31967
27	370	-58.85×10^6	2581930	4691177	80741	72160	36373
28	386	-59.38×10^6	2578904	4926998	59949	59185	34738
29	465	-60.05×10^6	2571176	4839950	42041	33858	40728
30	914	-63.74×10^6	2533423	4850165	16657	13990	47853

Table A.5: Details of ABySS assemblies of *E. coli* from CLC bio

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	620	-65.65×10^6	2522602	4692994	32728	30483	98983
22	569	-64.66×10^6	2528669	4691836	33655	29444	93766
23	527	-64.34×10^6	2527001	4598511	31478	27051	93579
24	560	-65.73×10^6	2517944	4608041	33556	22946	102399
25	667	-68.01×10^6	2494175	4561288	27187	14191	124413
26	830	-73.72×10^6	2431998	4545734	19849	8495	175362
27	1131	-84.04×10^6	2312331	4473173	12353	4780	280823

Table A.6: Details of Euler-sr assemblies of *E. coli* from CLC bio

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	180	-64.41×10^6	2530857	4538915	82523	53402	117331
23	176	-63.66×10^6	2536802	4541478	94647	57729	113126
25	179	-63.93×10^6	2538168	4544165	94659	58755	112015
27	175	-64.23×10^6	2527604	4539301	95159	46483	121115
29	200	-72.31×10^6	2438802	4488341	83775	12597	203884

Table A.7: Details of SOAPdenovo assemblies of *E. coli* from CLC bio

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
19	2091	-82.01×10^6	2352836	4532537	4112	4112	128441
21	198	-62.95×10^6	2542960	4542798	78546	53459	100298
23	896	-71.06×10^6	2471662	4532134	11286	11286	113920
25	1075	-71.52×10^6	2459554	4542157	8352	8386	111040

Table A.8: Details of Velvet assemblies of *E. coli* from CLC bio

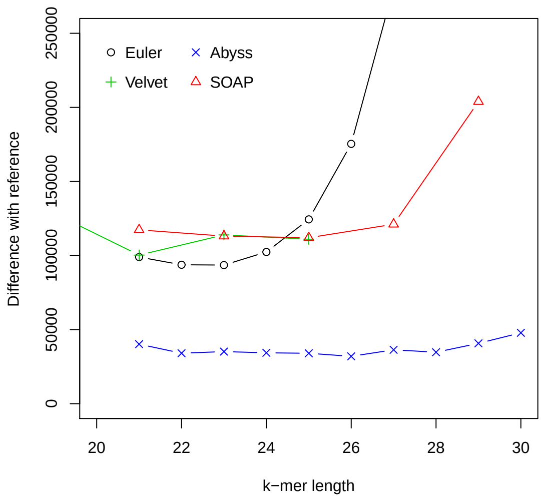


Figure A.5: Hash length vs difference from reference for CLC bio *E. coli* data. Differences between assemblies and the reference are shown on the y-axis where difference refers to numbers of bases in the reference not covered by the assembly or are different in the reference and the assembly.

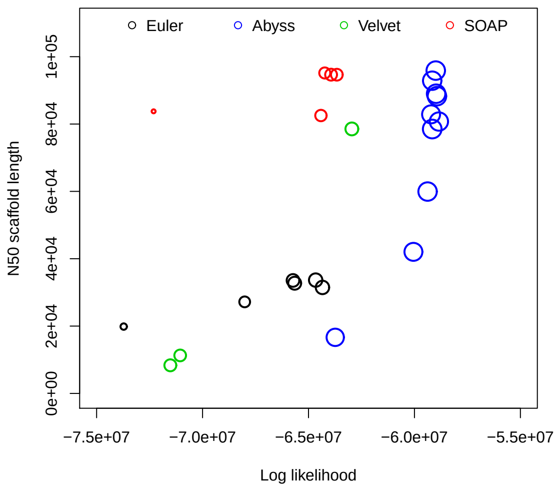


Figure A.6: Log likelihood vs N50 scaffold length for CLC bio *E. coli* data. Log likelihoods are shown on the x-axis and N50 scaffold lengths are shown on the y-axis. Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.7508, (ii) log likelihood vs N50 scaffold length: 0.4502, (iii) N50 scaffold length vs similarity: 0.1525

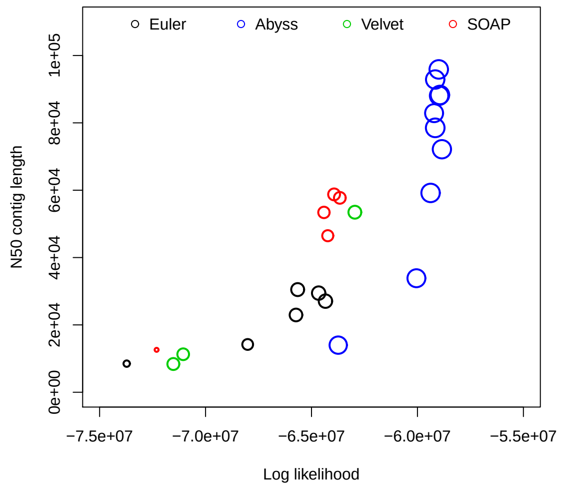


Figure A.7: Log likelihood vs N50 contig length for CLC bio *E. coli* data. Log likelihoods are shown on the x-axis and N50 contig lengths are shown on the y-axis. Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.7508, (ii) log likelihood vs N50 contig length: 0.6456, (iii) N50 contig length vs similarity: 0.4867

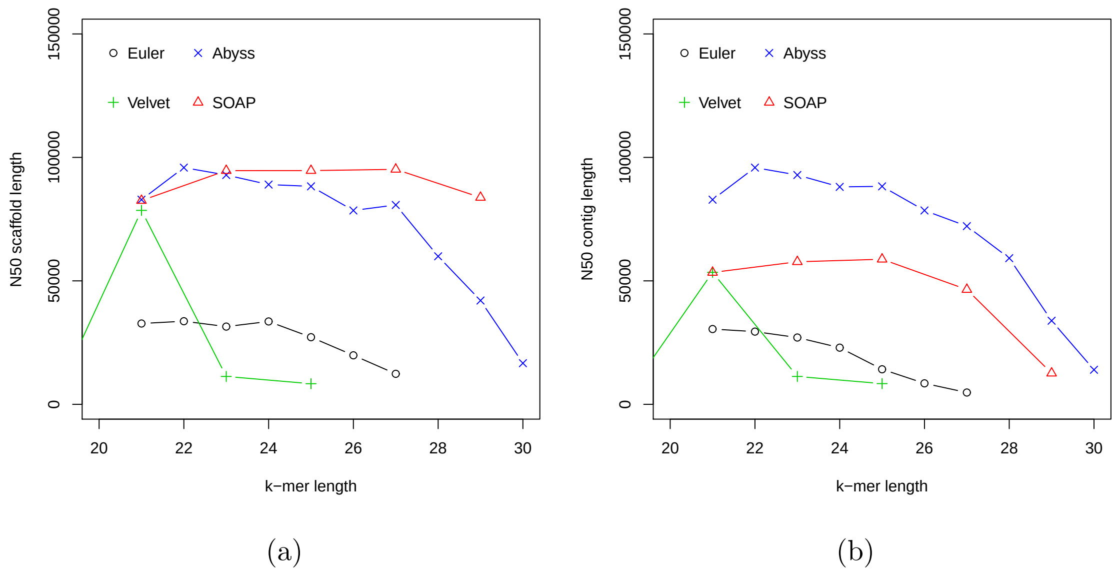


Figure A.8: (a) Hash length vs N50 scaffold length, (b) Hash length vs N50 contig length for CLC bio *E. coli* data

Supplementary information for *G. clavigera* data

- Organism: *Grosmannia clavigera*
- Total reads: 41034877
- Reads mapped using: BFAST-0.6.5a

References

- Identifier: [DDBJ/EMBL/GenBank: ACXQ00000000]
- Likelihood: -1879322697.243536
- Number of reads mapped: 35708831
- Total length: 29128742 bases

- Identifier: [DDBJ/EMBL/GenBank: ACYC00000000]
- Likelihood: -1770350912.163320
- Number of reads mapped: 36287407
- Total length: 29518845 bases

Assemblies

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	256124	-21.38×10^8	32979149	37142184	17220	16671	2809452
22	232501	-	-	36709685	27945	25559	2700584
23	210071	-20.71×10^8	33306434	36657119	37813	35026	2594137
24	192943	-	-	35750082	55105	49172	2482608
25	184177	-20.41×10^8	33472590	36531590	62982	58787	2382831
26	170997	-	-	37378311	77784	67029	2267368
27	159554	-20.23×10^8	33633432	35707870	93477	82886	2149608
28	149309	-	-	35786179	98522	92594	2028791
29	141391	-19.78×10^8	33809082	34799760	106367	92588	1906317
30	128886	-	-	34297703	112410	98015	1763278
31	118223	-19.40×10^8	34087970	34330058	119303	106618	1621203
32	107767	-	-	34689640	130674	105063	1484476
33	96232	-19.32×10^8	34385760	34434911	132294	113136	1266955
34	87415	-	-	35763495	135634	116906	1150820
35	77636	-19.12×10^8	34618713	34532607	135175	115835	1067088
36	69079	-	-	34687680	132362	104995	992028
37	61086	-18.96×10^8	34821797	34949964	136072	102040	922206
38	52639	-	-	34591823	118013	85684	886238
39	46557	-18.96×10^8	35034414	34312824	101526	63060	873592
40	42136	-	-	32248177	54066	28644	976543

Table A.9: Details of ABySS assemblies of *G. clavigera*. '-' indicates likelihood not computed

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
23	718	-21.94×10^8	33595856	28168246	404931	9519	3020692
25	854	-21.78×10^8	33889880	28382967	471227	16994	2837023
27	1077	-21.26×10^8	33906837	28633019	424982	24420	2679202
29	1319	-21.18×10^8	33943297	29022797	472893	28732	2520229
31	1672	-20.50×10^8	34161562	29315722	357348	31279	2362441
33	1959	-19.99×10^8	34537448	29386051	363057	26110	2227202
35	2319	-19.59×10^8	34962836	29343677	329057	15974	2166290
37	2737	-18.77×10^8	35088054	29073407	328162	4779	2527367
39	4411	-22.52×10^8	33276354	29618592	30833	440	7574440

Table A.10: Details of SOAPdenovo assemblies of *G. clavigera*

k-mer	#contigs	Log likelihood	Reads mapped	Length	N50 scaffold	N50 contig	Diff
21	4591	-21.81×10^8	33369710	26845917	21877	10000	3088905
23	1250	-21.11×10^8	34259414	26755488	174576	17299	2960501
25	1247	-19.86×10^8	34626195	26820775	256321	22175	2870534
27	1513	-19.26×10^8	34846513	26974390	313211	31692	2707470
29	1703	-19.17×10^8	34886953	27199402	322108	38029	2545990
31	1967	-19.07×10^8	34984676	27514303	304883	41579	2339277
33	2046	-19.00×10^8	35210381	27736636	332958	39448	2129331
35	2066	-18.79×10^8	35394715	28019566	322296	28230	1876746
37	2064	-18.87×10^8	35469983	28278498	279463	12808	1794095
39	2164	-18.94×10^8	35395912	28837034	181012	3210	2220725

Table A.11: Details of Velvet assemblies of *G. clavigera*

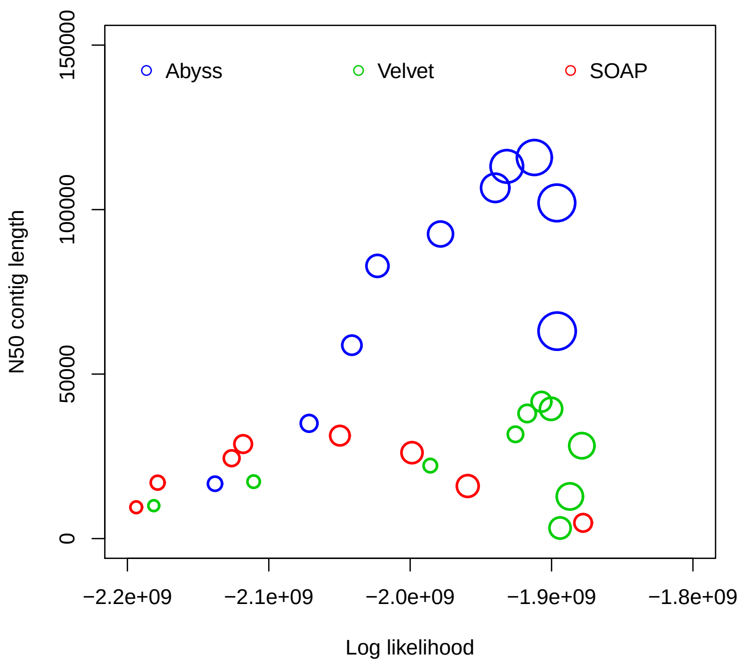
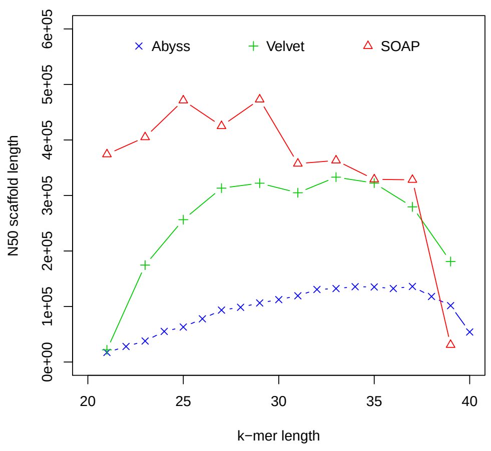
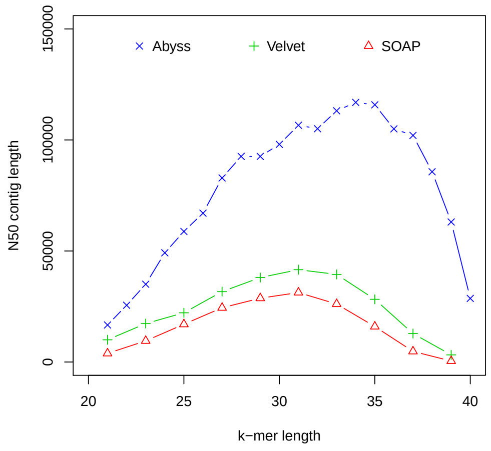


Figure A.9: Log likelihood vs N50 contig length for *G. clavigera*. Log likelihoods are shown on the x-axis and N50 contig lengths are shown on the y-axis. Each circle corresponds to an assembly generated using an assembler for some hash length and sizes of circles correspond to similarity with reference. The R^2 values are (i) log likelihood vs similarity: 0.4545793, (ii) log likelihood vs N50 contig length: 0.1582344, (iii) N50 contig length vs similarity: 0.3287432



(a)



(b)

Figure A.10: (a) Hash length vs N50 scaffold length, (b) Hash length vs N50 contig length for *G. clavigera* data

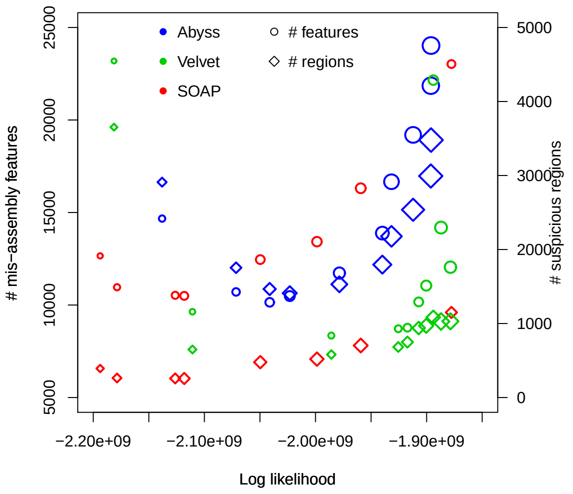
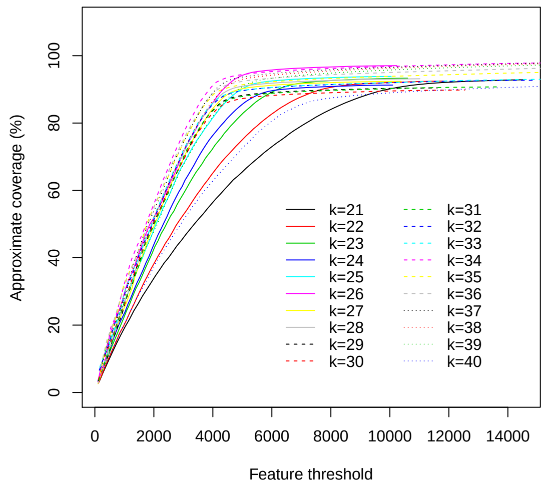
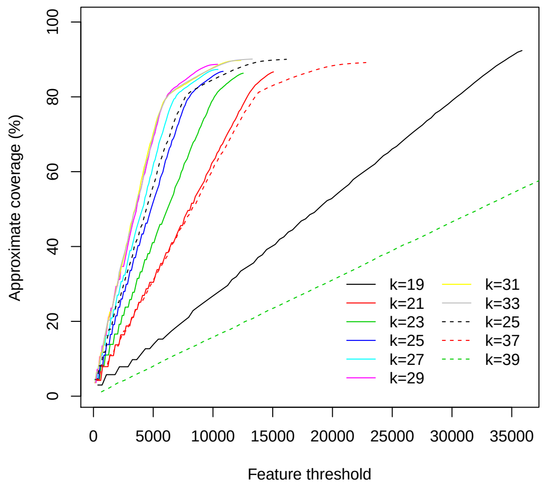


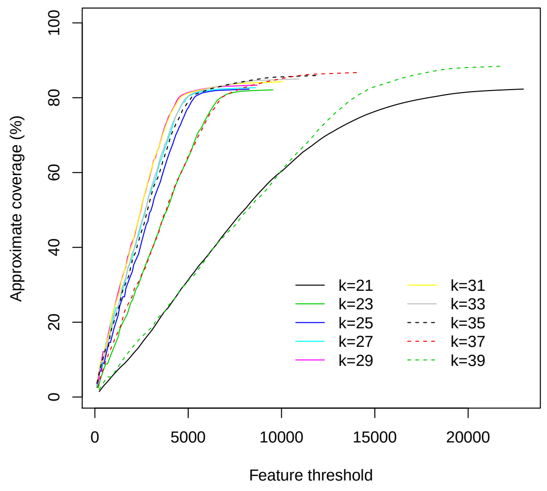
Figure A.11: Log likelihood vs numbers of mis-assembly features and suspicious regions for *G. clavigera*. Log likelihoods are shown on the x-axis and Numbers of mis-assembly features and suspicious regions reported by amosvalidate are shown on the y-axis. Each symbol corresponds to an assembly generated using an assembler for some hash length and sizes of symbols correspond to similarity with reference. The R^2 values are (i) log likelihood vs # mis-assembly features: 0.0726, (ii) log likelihood vs # suspicious regions: 0.0002, (iii) similarity vs # mis-assembly features: 0.2429, (iv) similarity vs # suspicious regions: 0.0588



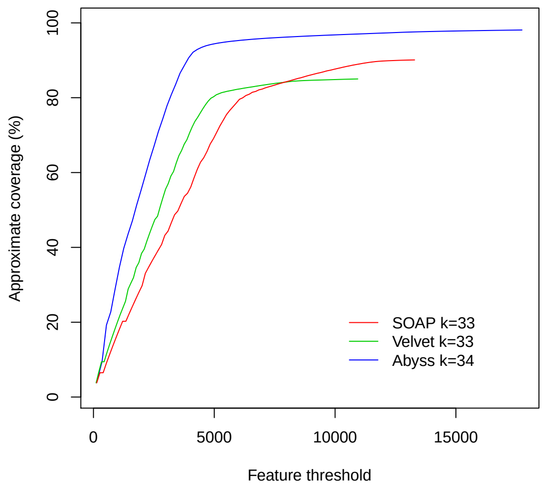
(a)



(b)



(c)



(d)

Figure A.12: Feature response curves for (a) ABySS, (b) SOAPdenovo, (c) Velvet, (d) all assemblies of *G. clavigera*

Supplementary information for GAGE assemblies

- Reads mapped using Bowtie 2 - 2.0.0 - beta 6

	Library 1		Library 2	
Assembler	Likelihood	# reads mapped	Likelihood	# reads mapped
ABySS	-63.67×10^6	530004	-16.97×10^7	706226
ALLPATHS-LG	-80.69×10^6	497645	-16.46×10^7	722683
Bambus2	-74.04×10^6	497912	-16.36×10^7	702615
MSR-CA	-92.54×10^6	475548	-16.59×10^7	716453
SGA	-79.25×10^6	483150	-18.68×10^7	535786
SOAPdenovo	-64.69×10^6	518963	-17.08×10^7	693421
Velvet	-65.18×10^6	515646	-16.76×10^7	688261
Reference	-63.55×10^6	530582	-16.03×10^7	738136

Table A.12: Likelihoods of GAGE assemblies of *S. aureus*

	Library 1		Library 2	
Assembler	Likelihood	# reads mapped	Likelihood	# reads mapped
ABySS	-11.18×10^7	830206	-16.37×10^7	368991
ALLPATHS-LG	-11.25×10^7	840079	-15.35×10^7	397859
Bambus2	-13.64×10^7	782413	-18.92×10^7	329183
CABOG	-16.17×10^7	735552	-23.06×10^7	287180
MSR-CA	-13.74×10^7	786563	-17.87×10^7	368515
SGA	-12.06×10^7	776980	-19.52×10^7	254567
SOAPdenovo	-11.23×10^7	832984	-16.44×10^7	379975
Velvet	-11.47×10^7	825280	-17.29×10^7	350845
Reference	-11.04×10^7	844992	-14.95×10^7	410758

Table A.13: Likelihoods of GAGE assemblies of *R. sphaeroides*

	Library 1		Library 2	
Assembler	Likelihood	# reads mapped	Likelihood	# reads mapped
ABySS	-82.28×10^7	17668025	-15.21×10^8	4428441
ALLPATHS-LG	-96.97×10^7	17409906	-13.11×10^8	5712663
Bambus2	-22.77×10^8	14162003	-	-
CABOG	-90.14×10^7	17579185	-12.25×10^8	5854239
MSR-CA	-10.44×10^8	16817275	-	-
SGA	-10.68×10^8	16065518	-	-
SOAPdenovo	*	*	-	-
Velvet	-13.07×10^8	14382698	-	-
Reference	-77.90×10^7	17938368	-11.25×10^8	6039649

Table A.14: Likelihoods of GAGE assemblies of human chromosome 14. * Likelihood not computed as reads could not be mapped with Bowtie 2

Supplementary information for Assemblathon 1 data

- Reads mapped using BFAST-0.7.0a
- Due to an issue with mapping first 21200000, 10550000, 10550000 reads from libraries of insert size 200, 3000, 10000 respectively were mapped

Assembler	Likelihood	#reads mapped	Coverage
ASTR 1	-15.73×10^8	19583345	90.9
WTSI-P 1	-84.10×10^7	21111310	98.7
WTSI-P 2	-83.38×10^7	21140835	98.6
EBI 1	-91.35×10^7	20903073	97.7
EBI 2	-12.50×10^8	19866346	92.9
WTSI-S 1	-88.44×10^7	21021575	97.8
WTSI-S 2	-88.12×10^7	21026735	97.8
WTSI-S 3	-88.56×10^7	21008685	97.8
WTSI-S 4	-83.84×10^7	21111281	98.3
CRACS 1	-10.30×10^8	20701940	96.3
CRACS 2	-10.68×10^8	20616519	95.9
CRACS 3	-11.02×10^8	20588439	95.6
BCCGSC 1	-82.38×10^7	21140990	98.6
BCCGSC 2	-82.36×10^7	21142978	98.6
BCCGSC 3	-82.25×10^7	21143989	98.7
BCCGSC 4	-82.23×10^7	21148816	98.7
BCCGSC 5	-82.20×10^7	21148400	98.7
DOEJGI 1	-86.84×10^7	21098465	97.3
IRISA 1	-11.46×10^8	20504763	93.7
IRISA 2	-12.42×10^8	20193993	92.8
IRISA 3	-11.99×10^8	20349895	92.9
IRISA 4	-10.12×10^8	20762461	95.7
IRISA 5	-10.97×10^8	20563254	94.6
CSHL 1	-18.69×10^8	18879773	87.2
CSHL 2	-83.01×10^7	21140814	98.5
DCSISU 1	-11.31×10^8	20525628	94.3
IoBUGA 1	-85.73×10^7	21106253	98.3
IoBUGA 2	-85.69×10^7	21103249	98.3
IoBUGA 3	-87.99×10^7	21057492	98.1
UCSF 1	-21.98×10^8	18216040	83.7
RHUL 1	-82.92×10^7	21138821	98.5
RHUL 2	-83.03×10^7	21140544	98.5
RHUL 3	-81.56×10^7	21142931	98.7
RHUL 4	-96.40×10^7	20868381	97.0
RHUL 5	-82.61×10^7	21148936	98.7
GACWT 1	-18.32×10^8	18734425	87.6
GACWT 2	-25.98×10^8	17023161	79.5
GACWT 3	-19.33×10^8	18455366	86.4
CIUoC	-27.38×10^8	17087357	78.5
BGI 1	-84.13×10^7	21108642	98.8
Broad 1	-92.49×10^7	21026955	98.3

Table A.15: Assemblathon 1 likelihoods

Appendix B

Supplementary information for scaffolding with assembly likelihood optimization

Table B.1: Parameters used to run aligners and SWALO

Dataset	Mapping parameters (Bowtie 2 / Bowtie)	Max insert size (Bowtie2 / Bowtieconvert)	Min contig length (SWALO)	Others (SWALO)
S. aureus (100kb contigs, short inserts)	-a	650	650	-
S. aureus (100kb contigs, long inserts)	-a	4000	4000	-
S. aureus (3kb contigs, short inserts)	-a	650	650	-
S. aureus (3kb contigs, long inserts)	-a	4000	4000	-d 3000 200 (all)
S. aureus (GAGE)	-a	6000	4400	-
R. sphaeroides (GAGE)	-a	6000	4400	-d 3700 400 (bowtie -v 0, -v 3)
P. falciparum (short inserts)	-k 5	1200	1200	-
P. falciparum (long inserts)	-k 5	6000	6000	-d 3045 750 (bowtie -v 0)
Human chromosome 14 (GAGE short jump)	-k 5	6000	3600	-
Human chromosome 14 (GAGE fosmid)	-k 5	50000	40000	-c -d 35000 2000 (all)

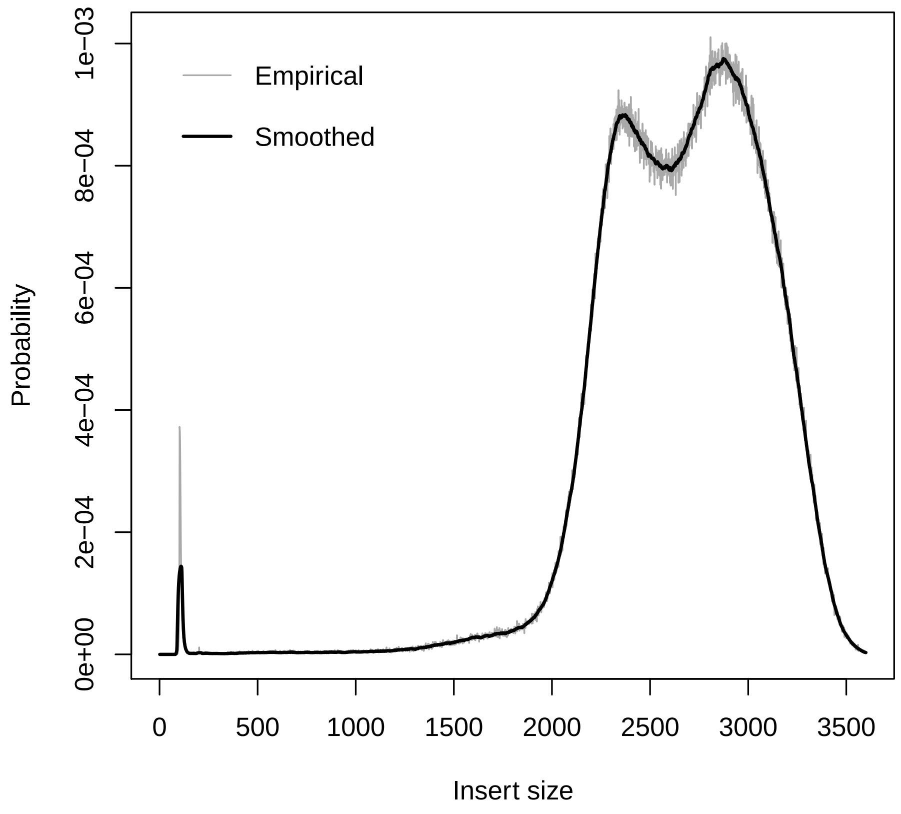
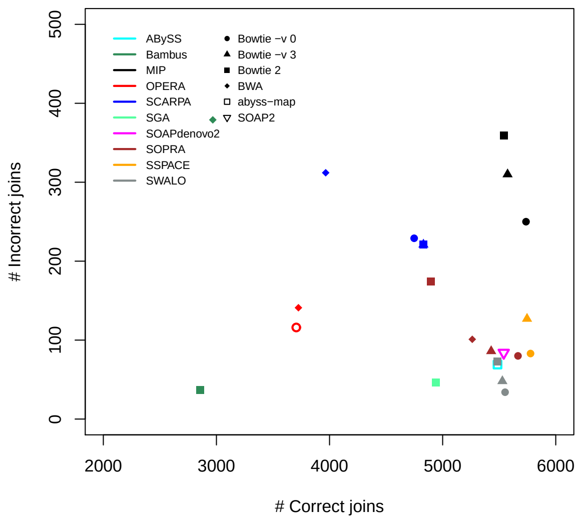
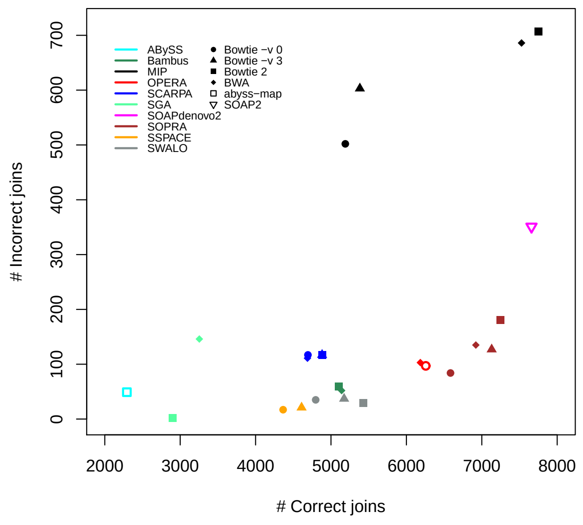


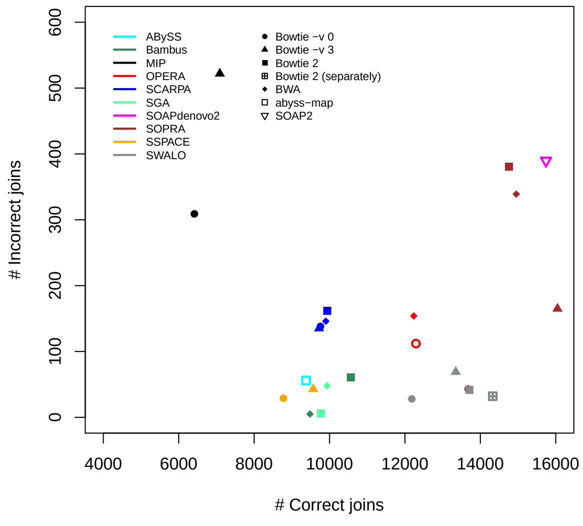
Figure B.1: **Insert size distribution for human chromosome 14 GAGE short jump library.**



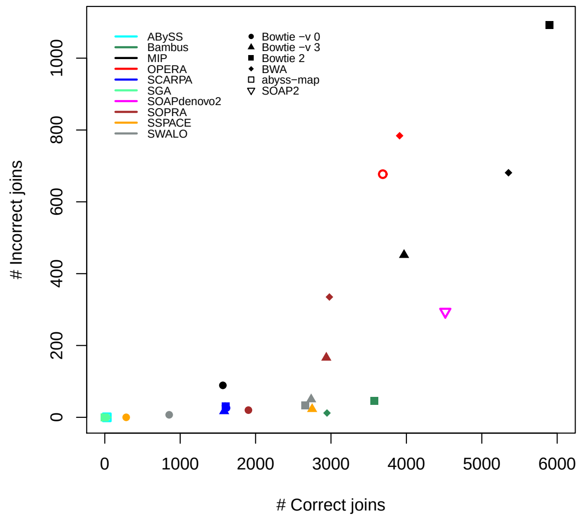
(a)



(b)



(c)



(d)

Figure B.2: **Performance of scaffolders.** Scatter plots showing number of correct joins vs incorrect joins made by different scaffolders on (a) *P. falciparum* short library, (b) *P. falciparum* short jump library, (c) human chromosome 14 short jump library, and (d) human chromosome 14 fosmid library. Values for all scaffolders except SWALO are from [62].

Appendix C

Supplementary information for association mapping from sequencing reads using k-mers

Finding differential k-mers in association studies

We consider the case where we have s_1 and s_2 samples from two populations. We observe a specific k-mer $k_{1,1}, \ldots, k_{1,s_1}$ and $k_{2,1}, \ldots, k_{2,s_2}$ times in the samples from two populations and total k-mer counts in the samples are given by $n_{1,1}, \ldots, n_{1,s_1}$ and $n_{2,1}, \ldots, n_{2,s_2}$. We assume that the k-mer counts are Poisson distributed with rate θ_1 and θ_2 in the two populations where the θ's can be interpreted as quantities proportional to the average number of times the k-mer appears in the two populations. The null hypothesis is $H_0 : \theta_1 = \theta_2 = \theta$ and the alternate hypothesis is $H_1 : \theta_1 \neq \theta_2$

We test the null using likelihood ratio test for nested models. The likelihood ratio is given by

$$\Lambda = \frac{\sup\{L(\theta_1, \theta_2)\}}{\sup\{L(\theta, \theta)\}}$$

where

$$L(\theta_1, \theta_2) = \prod_{i=1}^{s_1} \frac{e^{-\theta_1 n_{1,i}} (\theta_1 n_{1,i})^{k_{1,i}}}{k_{1,i}!} \prod_{i=1}^{s_2} \frac{e^{-\theta_2 n_{2,i}} (\theta_2 n_{2,i})^{k_{2,i}}}{k_{2,i}!}$$

and

$$L(\theta) = \prod_{i=1}^{s_1} \frac{e^{-\theta n_{1,i}} (\theta n_{1,i})^{k_{1,i}}}{k_{1,i}!} \prod_{i=1}^{s_2} \frac{e^{-\theta n_{2,i}} (\theta n_{2,i})^{k_{2,i}}}{k_{2,i}!}.$$

$L(\theta_1, \theta_2)$ is maximized at $\hat{\theta}_1 = \frac{\sum_{i=1}^{s_1} k_{1,i}}{\sum_{i=1}^{s_1} n_{1,i}}$, $\hat{\theta}_2 = \frac{\sum_{i=1}^{s_2} k_{2,i}}{\sum_{i=1}^{s_2} n_{2,i}}$ and $L(\theta)$ is maximized at $\hat{\theta} = \frac{\sum_{i=1}^{s_1} k_{1,i} + \sum_{i=1}^{s_2} k_{2,i}}{\sum_{i=1}^{s_1} n_{1,i} + \sum_{i=1}^{s_2} n_{2,i}}$. Therefore,

$$\Lambda = \frac{L(\hat{\theta}_1, \hat{\theta}_2)}{L(\hat{\theta}, \hat{\theta})}.$$

Since the null model is a special case of the alternate model, $2 \ln \Lambda$ is approximately chi-squared distributed with one degree of freedom [59, 60].

We note that the test statistic stays the same if the likelihood values are computed by pooling together the counts in samples from two populations *i.e.*

$$L(\theta_1, \theta_2) = \frac{e^{-\theta_1 N_1}(\theta_1 N_1)^{K_1}}{K_1!} \frac{e^{-\theta_2 N_2}(\theta_2 N_2)^{K_2}}{K_2!}$$

and

$$L(\theta) = \frac{e^{-\theta N_1}(\theta N_1)^{K_1}}{K_1!} \frac{e^{-\theta N_2}(\theta N_2)^{K_2}}{K_2!}$$

where $\sum_{i=1}^{s_1} k_{1,i} = K_1$, $\sum_{i=1}^{s_2} k_{2,i} = K_2$, $\sum_{i=1}^{s_1} n_{1,i} = N_1$, and $\sum_{i=1}^{s_2} n_{2,i} = N_2$.

For each k-mer in the data, we compute the statistic as described above and obtain a P-value using χ_1^2 distribution. The P-values are then corrected for multiple testing using Bonferroni correction.

Verification with simulated *E. coli* data

The simulation with *E. coli* genome (~4.6 million bp) was performed by first introducing 100 single base changes, 100 indels of random lengths less than 10bp and 100 indels of random lengths between 100 and 1000bp at random locations. Then different number of controls and cases were generated using `wgsim` of `SAMtools`[84] introducing more mutations with default parameters and 300000 paired end reads of length 70 (~5x k-mer coverage) were generated with sequencing error rate of 0.01.

The sensitivity and specificity analysis was done by aligning the sequences generated by HAWK using Bowtie 2 [80] and checking for overlap with mutation locations with in-house scripts. The power calculation is done by finding the minimum k-mer count to obtain a p-value less than significant threshold and then calculating the probability of observing at least that count for different coverages using R.

Testing for significant SNPs

Consider a site with two alleles. Let $n_{1,0}, n_{1,1}, n_{1,2}$ be the number of individuals with 0,1 and 2 copies of the minor allele respectively in the sample from population 1 and $n_{2,0}, n_{2,1}, n_{2,2}$ are the corresponding ones from population 2. Let p_1 and p_2 be the minor allele frequencies in the two populations and N_1 and N_2 be the number of samples. The null hypothesis is $H_0 : p_1 = p_2 = p$ and the alternate hypothesis is $H_0 : p_1 \neq p_2$.

We test the null using likelihood ratio test for nested models. The likelihood ratio is given by

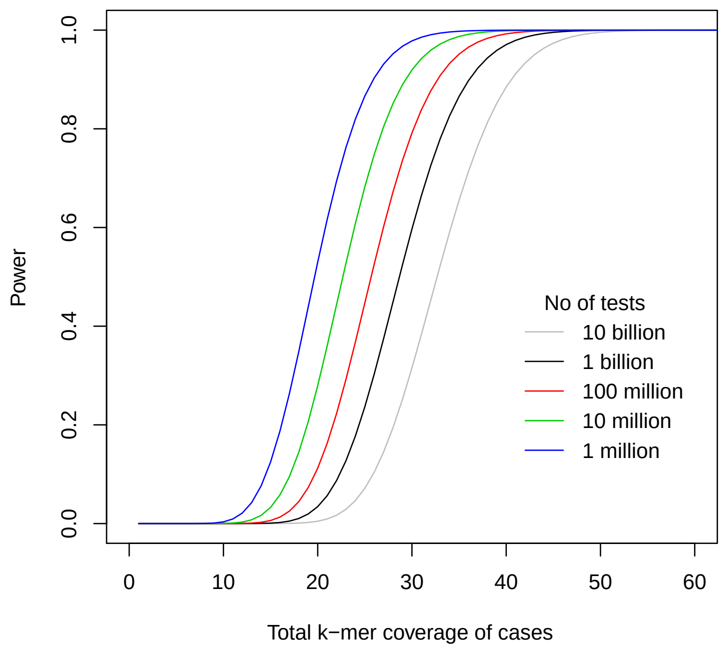


Figure C.1: **Power for different k-mer coverage.** The figure shows power to detect a k-mer present in all case samples and no control sample against total k-mer coverage of cases using Bonferroni correction for different number of total tests for p-value=0.05.

$$\Lambda = \frac{\sup\{L(p_1, p_2)\}}{\sup\{L(p, p)\}}$$

where under random mating

$$L(p_1, p_2) = \binom{N_1}{n_{1,0}, n_{1,1}, n_{1,2}} (1-p_1)^{2n_{1,0}} (2p_1(1-p_1))^{n_{1,1}} p_1^{2n_{1,2}} \\ \binom{N_2}{n_{2,0}, n_{2,1}, n_{2,2}} (1-p_2)^{2n_{2,0}} (2p_2(1-p_2))^{n_{2,1}} p_2^{2n_{2,2}}$$

and

$$L(p) = \binom{N_1}{n_{1,0}, n_{1,1}, n_{1,2}} (1-p)^{2n_{1,0}} (2p(1-p))^{n_{1,1}} p^{2n_{1,2}} \\ \binom{N_2}{n_{2,0}, n_{2,1}, n_{2,2}} (1-p)^{2n_{2,0}} (2p(1-p))^{n_{2,1}} p^{2n_{2,2}}.$$

$L(p_1, p_2)$ is maximized at $\hat{p_1} = \frac{n_{1,1}+2n_{1,2}}{2N_1}$, $\hat{p_2} = \frac{n_{2,1}+2n_{2,2}}{2N_2}$ and $L(p)$ is maximized at $\hat{p} = \frac{n_{1,1}+2n_{1,2}+n_{2,1}+2n_{2,2}}{2N_1+2N_2}$. Therefore,

$$\Lambda = \frac{L(\hat{p_1}, \hat{p_2})}{L(\hat{p}, \hat{p})}.$$

Since the null model is a special case of the alternate model, $2 \ln \Lambda$ is approximately chi-squared distributed with one degree of freedom.

For each SNP site in the data, we compute the statistic as described above and obtain a p-value using χ^2_1 distribution. The p-values are then corrected for multiple testing using Bonferroni correction.

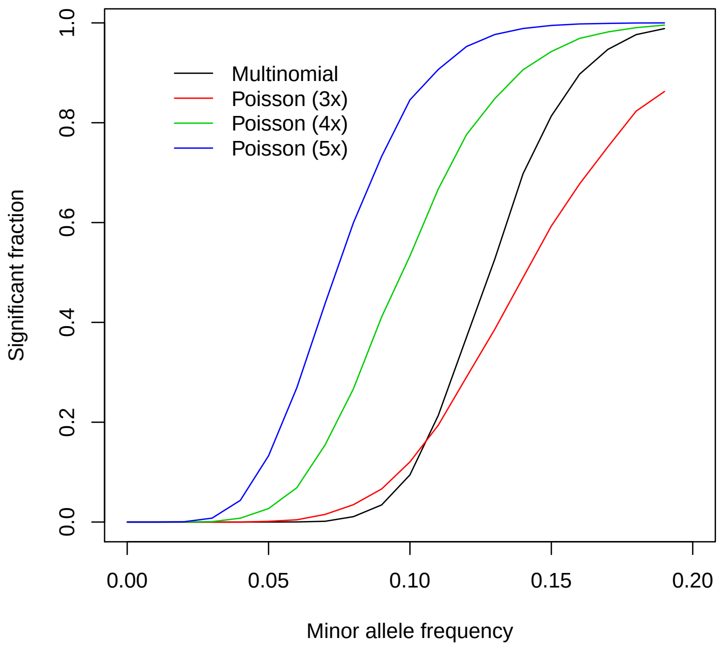


Figure C.2: **Comparison of powers.** Figure shows fraction of runs found significant by tests against minor allele frequency of one of samples (with the other one fixed at 0) after Bonferroni correction for total number of k-mers tested (for test based on the Poisson distribution) and total number of genotypes tested (for test based on the multinomial distribution). The curve labeled multinomial and Poisson correspond to likelihood ratio test using multinomial distribution and Poisson distributions with different k-mer coverage.

Intersection analysis using 1000 genomes data

For the intersection analysis we used data from 87 YRI individuals and 98 TSI individuals for which both sequencing reads and genotype calls were available. The HAWK pipeline was run using a k-mer size of 31. The assembly was using ABySS [147]. The sequences were aligned to the human reference genome version GRCh37 using Bowtie2. Each of the genotypes called by the 1000 genomes project was then tested for association with populations using the approach described in the previous section. The extent of intersection of the loci found using two methods was then determined using BEDtools [131].

Breakdown analysis

The types of variants corresponding to sequences found using the HAWK pipeline were estimated by mapping them to the human reference genome version hg38 using Bowtie2 and using following properties.

Unmapped: The sequences that were not mapped to the reference using Bowtie2.

Multimapped: The sequences with multiple mappings. These may be due to copy number variations or sequence variation in repetitive regions.

Indels: The sequences that mapped to the reference with one or more indels.

Multiple SNPs/Structural: The maximum length of a sequence due to a single SNP with 31-mer is 61. Sequences with length greater than 61 were assigned this label.

SNPs: All other sequences.

Probing for variants of interest

We searched for well known variants and other variants of potential biological interest by aligning sequences to RefSeq mRNAs using Bowtie2 and then looking up gene names from RefSeq mRNA identifiers using the UCSC Table Browser [69]. Variants of interest were then further explored by running BLAT [72] on the UCSC Human Genome Browser [74].

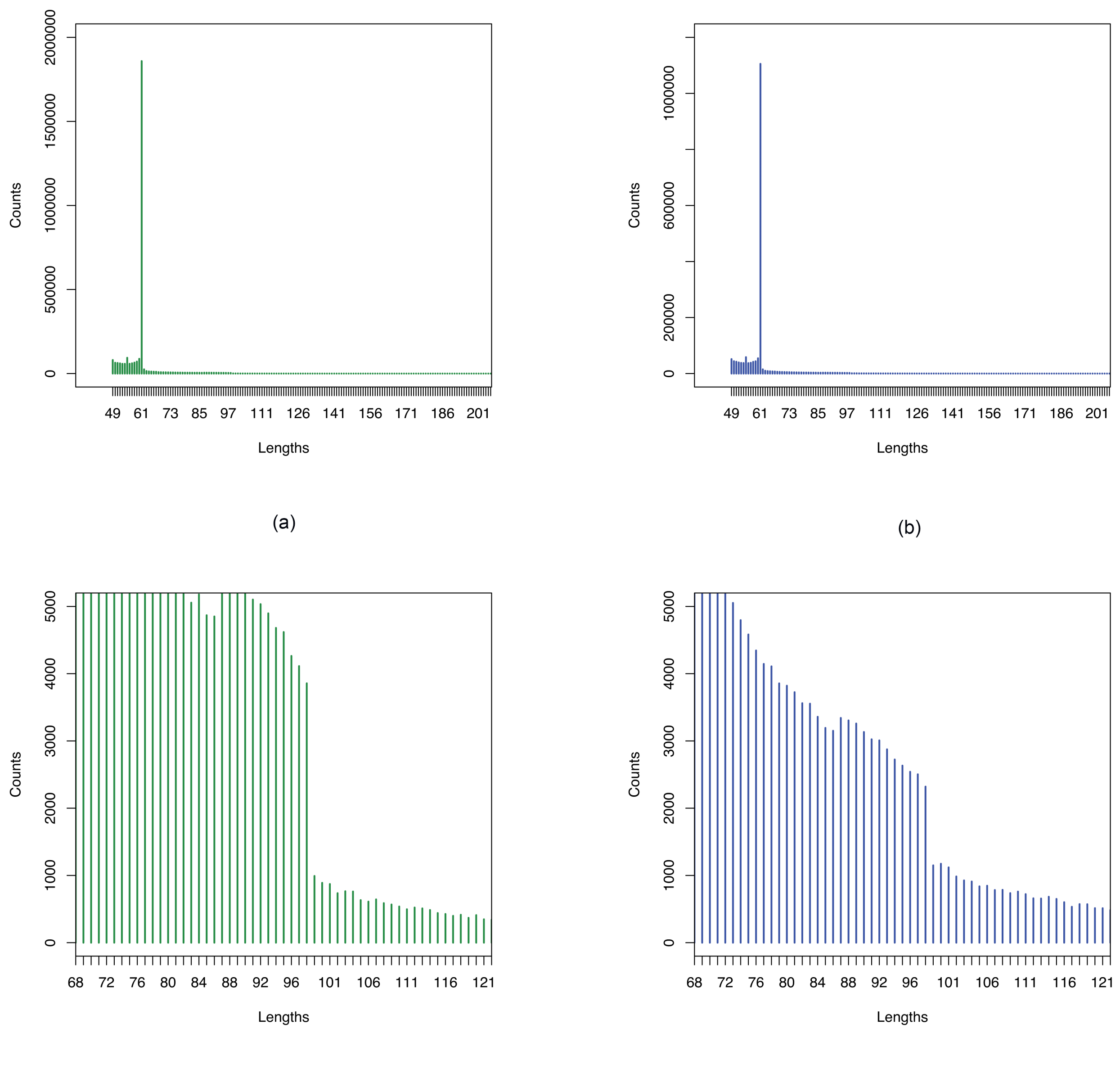


Figure C.3: **Histograms of sequence lengths in YRI-TSI comparison.** Figures show sections of histograms of lengths of sequences associated with (a),(c) YRI and (b),(d) TSI in comparison of YRI and TSI samples. Figures (a), (b) show peaks at 61, the maximum length corresponding to a single SNP with k-mer size of 31. Figures (c), (d) show drop off after 98 which is the maximum length corresponding to two close-by SNPs as 31-mers were assembled using a minimum overlap of 24.

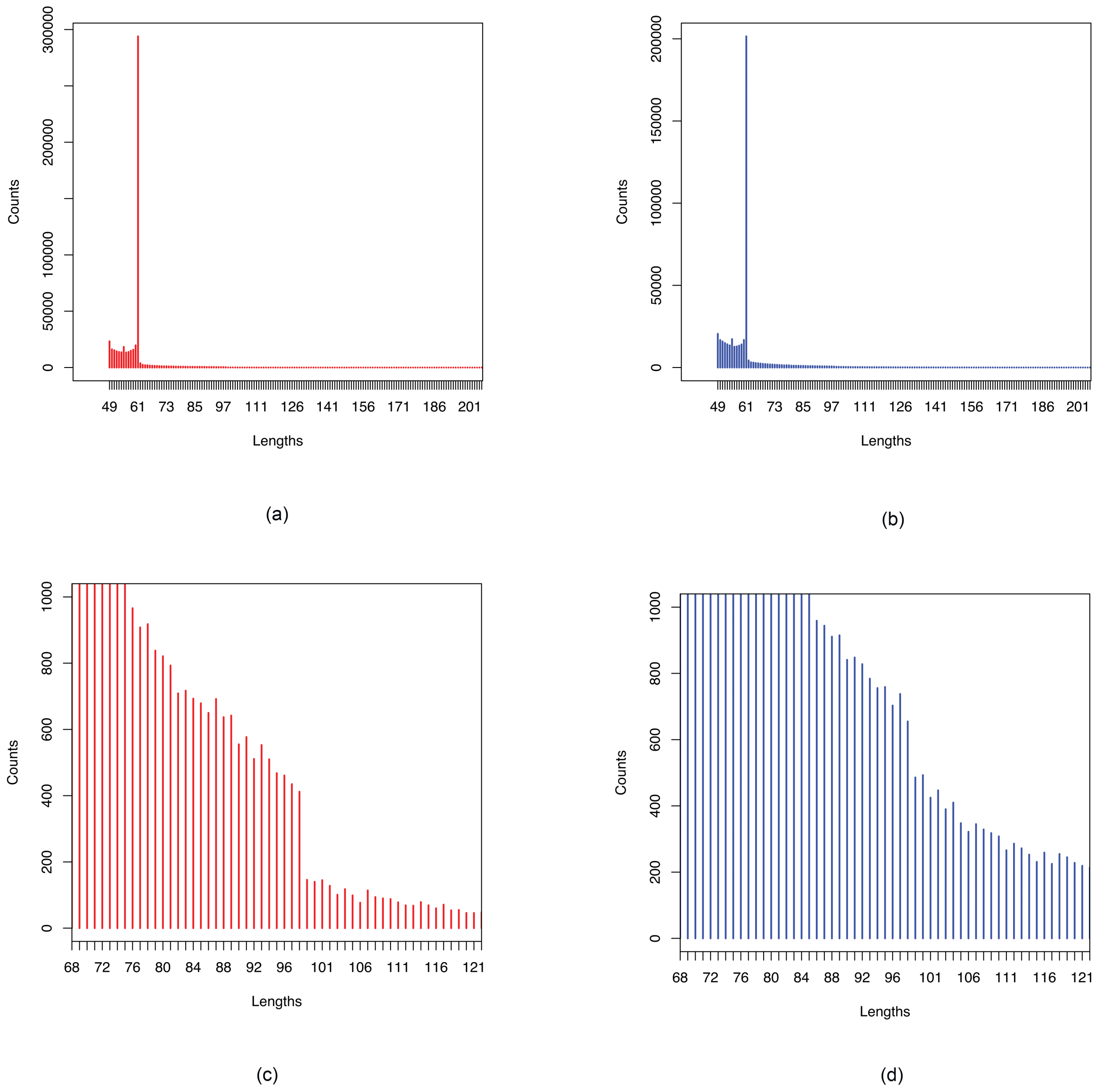


Figure C.4: **Histograms of sequence lengths in BEB-TSI comparison.** Figures show sections of histograms of lengths of sequences associated with (a),(c) BEB and (b),(d) TSI in comparison of BEB and TSI samples. Figures (a), (b) show peaks at 61, the maximum length corresponding to a single SNP with k-mer size of 31. Figures (c), (d) show drop off after 98 which is the maximum length corresponding to two close-by SNPs as 31-mers were assembled using a minimum overlap of 24.

Table C.1: **Variants in *Titin* of differential prevalence in BEB-TSI comparison.**

Gene	SNP ID	Variant type	Allele	p-value	%BEB	%TSI
TTN	rs9808377	Missense	G	1.70×10^{-15}	66.44%	41.91%
TTN	rs62621236	Missense	G	2.33×10^{-16}	27.70%	5.72%
TTN	rs2291311	Missense	C	1.06×10^{-11}	25.77%	7.77%
TTN	rs16866425	Missense	C	8.19×10^{-12}	21.73%	2.73%
TTN	rs4894048	Missense	T	2.00×10^{-23}	22.65%	2.26%
TTN	rs13398235	Intron/missense	A	2.04×10^{-13}	41.00%	17.40%
TTN	rs11888217	Intron/missense	T	4.18×10^{-13}	27.25%	4.55%
TTN	rs10164753	Missense	T	3.69×10^{-13}	28.48%	6.19%
TTN	rs10497520	Missense	T	1.66×10^{-23}	54.76%	18.86%
TTN	rs2627037	Missense	A	6.99×10^{-13}	25.06%	4.72%
TTN	rs1001238	Missense	C	1.66×10^{-17}	64.66%	38.21%
TTN	rs3731746	Missense	A	1.26×10^{-14}	50.72%	30.21%
TTN	rs17355446	Intron/missense	A	3.31×10^{-11}	15.44%	1.11%
TTN	rs2042996	Missense	A	1.03×10^{-17}	71.41%	35.87%
TTN	rs747122	Missense	T	1.59×10^{-11}	28.57%	7.24%
TTN	rs1560221*	Synonymous	G	1.11×10^{-22}	70.71%	34.66%
TTN	rs16866406	Missense	A	2.17×10^{-12}	35.60%	17.51%
TTN	rs4894028	Missense	T	2.58×10^{-13}	27.54%	6.89%
TTN	-	Insertion	T	1.09×10^{-12}	34.11%	8.59%
TTN	rs3829747	Missense	T	4.72×10^{-12}	37.55%	20.30%
TTN	rs2291310	Missense	C	2.18×10^{-20}	36.63%	8.04%
TTN	rs2042995	Intron/Missense	C	7.83×10^{-12}	56.32%	31.64%
TTN	rs3829746	Missense	C	5.30×10^{-29}	75.94%	37.60%
TTN	rs744426	Missense	A	1.36×10^{-13}	37.15%	18.92%

Variants in *Titin*, a gene linked to cardiovascular diseases, that were found to be significantly more common in BEB samples compared to TSI samples.The (%) values denote fraction of individuals in the sample with the allele present. The p-values and % values are averaged over k-mers constituting the associated sequences.
* A stop gained SNP exists at the same position.

Bibliography

[1] Can Alkan, Saba Sajjadian, and Evan E Eichler. “Limitations of next-generation genome sequence assembly”. In: *Nature Methods* 8 (2011), pp. 61–65. DOI: http://dx.doi.org/10.1038/nmeth.1527.

[2] Nicolas Altemose et al. “Genomic Characterization of Large Heterochromatic Gaps in the Human Genome Assembly”. In: *PLoS Computational Biology* 10.5 (May 2014), e1003628. DOI: 10.1371/journal.pcbi.1003628.

[3] S. F. Altschul et al. “Basic local alignment search tool.” In: *Journal of Molecular Biology* 215 (Oct. 1990), pp. 403–410. DOI: 10.1006/jmbi.1990.9999.

[4] “An integrated map of genetic variation from 1,092 human genomes”. In: *Nature* 491.7422 (Oct. 2012), pp. 56–65. DOI: 10.1038/nature11632.

[5] Sharon Aviran et al. “Modeling and automation of sequencing-based characterization of RNA structure”. In: *Proceedings of the National Academy of Sciences* 108.27 (2011), pp. 11069–11074.

[6] Monya Baker. “De novo genome assembly: what every biologist should know”. In: *Nature Methods* 9.4 (2012), pp. 333–337.

[7] M Boetzer et al. “Scaffolding pre-assembled contigs using SSPACE”. In: *Bioinformatics* 27 (2011), pp. 578–579. DOI: 10.1093/bioinformatics/btq683.

[8] Vladimír Boža, Broňa Brejová, and Tomáš Vinař. “GAML: genome assembly by maximum likelihood”. In: *Algorithms for Molecular Biology* 10.1 (2015), p. 18.

[9] KR Bradnam et al. “Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species”. In: *Gigascience* 2 (2013), p. 10. DOI: 10.1186/2047-217X-2-10.

[10] Daniel Branton et al. “The potential and challenges of nanopore sequencing”. In: *Nature Biotechnology* 26.10 (2008), pp. 1146–1153.

[11] Guy Bresler, Ma’ayan Bresler, and David Tse. “Optimal assembly for high throughput shotgun sequencing”. In: *BMC Bioinformatics* 14.Suppl 5 (2013), S18.

[12] Maayan Bresler. “Statistical Methods for Genome Assembly”. PhD thesis. University of California, Berkeley, 2014.

[13] Ma'ayan Bresler et al. "Telescoper: de novo assembly of highly repetitive regions". In: *Bioinformatics* 28.18 (2012), pp. i311–i317.

[14] Jonathan Butler et al. "ALLPATHS: De novo assembly of whole-genome shotgun microreads". In: *Genome Research* 18 (2008), pp. 810–820. DOI: `10.1101/gr.7337908`.

[15] Mark J Chaisson and Pavel A Pevzner. "Short read fragment assembly of bacterial genomes". In: *Genome Research* 18.2 (2008), pp. 324–330.

[16] Jarrod A Chapman et al. "A whole-genome shotgun approach for assembling and anchoring the hexaploid bread wheat genome". In: *Genome Biology* 16.1 (2015), p. 26.

[17] Jarrod A. Chapman et al. "Meraculous: *De Novo* Genome Assembly with Short Paired-End Reads". In: *PLoS ONE* 6.8 (Aug. 2011), e23501. DOI: `10.1371/journal.pone.0023501`.

[18] Daniel I. Chasman et al. "Forty-Three Loci Associated with Plasma Lipoprotein Size, Concentration, and Cholesterol Content in Genome-Wide Analysis". In: *PLoS Genetics* 11 (2009). DOI: `10.1371/journal.pgen.1000730`.

[19] Derek Y Chiang et al. "High-resolution mapping of copy-number alterations with massively parallel sequencing". In: *Nature Methods* 6.1 (2009), pp. 99–103.

[20] Kuan-Rau Chiou and Min-Ji Charng. "Common mutations of familial hypercholesterolemia patients in Taiwan: Characteristics and implications of migrations from southeast China". In: *Gene* 498.1 (2012-04-25), 100(7).

[21] Scott Clark et al. "ALE: a generic assembly likelihood evaluation framework for assessing the accuracy of genome and metagenome assemblies". In: *Bioinformatics* (2013), bts723.

[22] *CLC bio: NGS example data*. `http://www.clcbio.com/index.php?id=1290`.

[23] Phillip EC Compeau, Pavel A Pevzner, and Glenn Tesler. "How to apply de Bruijn graphs to genome assembly". In: *Nature Biotechnology* 29.11 (2011), pp. 987–991.

[24] International HapMap Consortium et al. "A haplotype map of the human genome". In: *Nature* 437.7063 (2005), pp. 1299–1320.

[25] Gregory E Crawford et al. "Genome-wide mapping of DNase hypersensitive sites using massively parallel signature sequencing (MPSS)". In: *Genome Research* 16.1 (2006), pp. 123–131.

[26] Petr Danecek et al. "The variant call format and VCFtools". In: *Bioinformatics* 27.15 (2011), pp. 2156–2158. DOI: `10.1093/bioinformatics/btr330`.

[27] Aaron E. Darling et al. "Mauve Assembly Metrics". In: *Bioinformatics* 27 (2011), pp. 2756–2757. DOI: `10.1093/bioinformatics/btr451`.

[28] A Dayarian, TP Michael, and AM Sengupta. "SOPRA: scaffolding algorithm for paired reads via statistical optimization". In: *BMC Bioinformatics* 11 (2010), p. 345. DOI: `10.1186/1471-2105-11-345`.

[29] Arthur L. Delcher et al. “Fast algorithms for large-scale genome alignment and comparison.” In: *Nucleic acids research* 30 (June 2002), pp. 2478–2483. DOI: 10.1093/nar/30.11.2478.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin. “Maximum Likelihood from Incomplete Data via the EM Algorithm”. English. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 1–38.

[31] Scott DiGuistini et al. “De novo genome sequence assembly of a filamentous fungus using Sanger, 454 and Illumina sequence data”. In: *Genome Biology* 10 (2009), R94. DOI: 10.1186/gb-2009-10-9-r94.

[32] Juliane C. Dohm et al. “SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing”. In: *Genome Research* 17 (2007), pp. 1697–1706. DOI: 10.1101/gr.6435207.

[33] Juliane C. Dohm et al. “Substantial biases in ultra-short read data sets from high-throughput DNA sequencing”. In: *Nucleic Acids Research* 36 (2008), e105. DOI: 10.1093/nar/gkn425.

[34] N Donmez and M Brudno. “SCARPA: scaffolding reads with practical algorithms”. In: *Bioinformatics* 29 (2013), pp. 428–434. DOI: 10.1093/bioinformatics/bts716.

[35] Dent A. Earl et al. “Assemblathon 1: A competitive assessment of de novo short read assembly methods”. In: *Genome Research* (2011). DOI: 10.1101/gr.126599.111.

[36] Jack Edmonds and Ellis L. Johnson. “Matching: A Well-Solved Class of Integer Linear Programs”. In: *Combinatorial Optimization - Eureka, You Shrink!* Ed. by Michael Jünger, Gerhard Reinelt, and Giovanni Rinaldi. Vol. 2570. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, pp. 27–30. DOI: 10.1007/3-540-36478-1_3.

[37] John Eid et al. “Real-time DNA sequencing from single polymerase molecules”. In: *Science* 323.5910 (2009), pp. 133–138.

[38] Michael Farrar. “Striped Smith-Waterman speeds database searches six times over other SIMD implementations”. In: *Bioinformatics* 23 (2007), pp. 156–161. DOI: 10.1093/bioinformatics/btl582.

[39] Matteo Fumagalli et al. “ngsTools: methods for population genetics analyses from Next-Generation Sequencing data”. In: *Bioinformatics* 30.10 (2014), pp. 1486–1487. DOI: 10.1093/bioinformatics/btu041.

[40] Matteo Fumagalli et al. “Quantifying Population Genetic Differentiation from Next-Generation Sequencing Data”. In: *Genetics* 195.3 (2013), pp. 979–992. DOI: 10.1534/genetics.113.154740.

[41] John Gallant, David Maier, and James Astorer. “On finding minimal length superstrings”. In: *Journal of Computer and System Sciences* 20.1 (1980), pp. 50–58.

[42] S Gao, W-K Sung, and N Nagarajan. “Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences”. In: *Journal of Computational Biology* 18 (2011), pp. 1681–1691. DOI: 10.1089/cmb.2011.0170.

[43] Mohammadreza Ghodsi et al. “De novo likelihood-based measures for comparing genome assemblies”. In: *BMC Research Notes* 6.1 (2013), p. 334.

[44] Travis C Glenn. “Field guide to next-generation DNA sequencers”. In: *Molecular Ecology Resources* 11.5 (2011), pp. 759–769.

[45] Sante Gnerre et al. “Assisted assembly: how to improve a de novo genome assembly by using related species”. In: *Genome Biology* 10.8 (2009), R88.

[46] Sante Gnerre et al. “High-quality draft assemblies of mammalian genomes from massively parallel sequence data”. In: *Proceedings of the National Academy of Sciences* 108.4 (2011), pp. 1513–1518.

[47] P Green. *PHRAP documentation.* June 1994. URL: http://www.phrap.org.

[48] A Gritsenko et al. “GRASS: a generic algorithm for scaffolding next-generation sequencing assemblies”. In: *Bioinformatics* 28 (2012), pp. 1429–1437. DOI: 10.1093/bioinformatics/bts175.

[49] Milan Gupta, Narendra Singh, and Subodh Verma. “South Asians and Cardiovascular Risk: What Clinicians Should Know”. In: *Circulation* 113.25 (2006), e924–e929. DOI: 10.1161/CIRCULATIONAHA.105.583815.

[50] Gavin Ha et al. “Integrative analysis of genome-wide loss of heterozygosity and monoallelic expression at nucleotide resolution reveals disrupted pathways in triple-negative breast cancer”. In: *Genome Research* 22.10 (2012), pp. 1995–2007.

[51] Kelley Harris and Rasmus Nielsen. “Error-prone polymerase activity causes multinucleotide mutations in humans”. In: *Genome Research* 24.9 (2014), pp. 1445–1454. DOI: 10.1101/gr.170696.113.

[52] Timothy D. Harris et al. “Single-Molecule DNA Sequencing of a Viral Genome”. In: *Science* 320 (2008), pp. 106–109. DOI: 10.1126/science.1150427.

[53] Bernhard Haubold. “Alignment-free phylogenetics and population genetics”. In: *Briefings in Bioinformatics* 15.3 (2014), pp. 407–418. DOI: 10.1093/bib/bbt083.

[54] Daniel S Herman et al. “Truncations of titin causing dilated cardiomyopathy.” In: *The New England journal of medicine* 366.7 (2012), pp. 619–28. DOI: 10.1056/NEJMoa1110186.

[55] David Hernandez et al. “De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer.” In: *Genome Research* 18 (2008), pp. 802–809. DOI: 10.1101/gr.072033.107.

[56] Clive J Hoggart et al. “Genome-wide significance for dense SNP and resequencing data”. In: *Genetic Epidemiology* 32.2 (2008), pp. 179–185.

[57] Nils Homer, Barry Merriman, and Stanley F. Nelson. "BFAST: An Alignment Tool for Large Scale Genome Resequencing". In: *PLoS ONE* 4 (2009), e7767+. DOI: `10.1371/journal.pone.0007767`.

[58] M Howison et al. "Bayesian Genome Assembly and Assessment by Markov Chain Monte Carlo Sampling". In: *PLoS ONE* 9.6 (2014).

[59] John P. Huelsenbeck and Keith A. Crandall. "Phylogeny estimation and hypothesis testing using maximum likelihood". In: *Annual Review of Ecology and Systematics* 28.1 (1997), pp. 437–466. DOI: `10.1146/annurev.ecolsys.28.1.437`.

[60] John P. Huelsenbeck, David M. Hillis, and Rasmus Nielsen. "A Likelihood-Ratio Test of Monophyly". In: *Systematic Biology* 45.4 (1996), pp. 546–558. DOI: `10.1093/sysbio/45.4.546`.

[61] M Hunt et al. "REAPR: a universal tool for genome assembly evaluation". In: *Genome Biology* 14 (2013), R47. DOI: `10.1186/gb-2013-14-5-r47`.

[62] Martin Hunt et al. "A comprehensive evaluation of assembly scaffolding tools". In: *Genome Biology* 15.3 (2014), R42. DOI: `10.1186/gb-2014-15-3-r42`.

[63] Daniel H. Huson, Knut Reinert, and Eugene W. Myers. "The Greedy Path-merging Algorithm for Contig Scaffolding". In: *J. ACM* 49.5 (Sept. 2002), pp. 603–615. DOI: `10.1145/585265.585267`.

[64] Nicholas T Ingolia et al. "Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling". In: *Science* 324.5924 (2009), pp. 218–223.

[65] Zamin Iqbal et al. "De novo assembly and genotyping of variants using colored de Bruijn graphs". In: *Nature Genetics* 44.2 (Feb. 2012), pp. 226–232. DOI: `10.1038/ng.1028`.

[66] William R. Jeck et al. "Extending assembly of short DNA sequences to handle error". In: *Bioinformatics* 23 (2007), pp. 2942–2944. DOI: `10.1093/bioinformatics/btm451`.

[67] David S Johnson et al. "Genome-wide mapping of in vivo protein-DNA interactions". In: *Science* 316.5830 (2007), pp. 1497–1502.

[68] P Joshi et al. "RIsk factors for early myocardial infarction in south asians compared with individuals in other countries". In: *Journal of the American Medical Association* 297.3 (2007), pp. 286–294. DOI: `10.1001/jama.297.3.286`.

[69] Donna Karolchik et al. "The UCSC Table Browser data retrieval tool". In: *Nucleic Acids Research* 32.suppl 1 (2004), pp. D493–D496. DOI: `10.1093/nar/gkh103`.

[70] Sekar Kathiresan and Deepak Srivastava. "Genetics of Human Cardiovascular Disease". In: *Cell* 148.6 (2012), pp. 1242 –1257. DOI: `http://dx.doi.org/10.1016/j.cell.2012.03.001`.

[71] John D Kececioglu and Eugene W Myers. "Combinatorial algorithms for DNA sequence assembly". In: *Algorithmica* 13.1-2 (1995), pp. 7–51.

[72] W. James Kent. “BLAT – The BLAST-Like Alignment Tool”. In: *Genome Research* 12.4 (2002), pp. 656–664. DOI: 10.1101/gr.229202.

[73] W James Kent and David Haussler. “Assembly of the working draft of the human genome with GigAssembler”. In: *Genome Research* 11.9 (2001), pp. 1541–1548.

[74] W. James Kent et al. “The Human Genome Browser at UCSC”. In: *Genome Research* 12.6 (2002), pp. 996–1006. DOI: 10.1101/gr.229102.

[75] Jeffrey M. Kidd et al. “Characterization of missing human genome sequences and copy-number polymorphic insertions.” In: *Nature Methods* 7.5 (May 2010), pp. 365–371.

[76] S Koren, TJ Treangen, and M Pop. “Bambus 2: scaffolding metagenomes”. In: *Bioinformatics* 27 (2011), pp. 2964–2971. DOI: 10.1093/bioinformatics/btr520.

[77] Alexander M. Kulminski et al. “The role of lipid-related genes, aging-related processes, and environment in healthspan”. In: *Aging Cell* 12.2 (2013), pp. 237–246. DOI: 10.1111/acel.12046.

[78] Eric S Lander and Michael S Waterman. “Genomic mapping by fingerprinting random clones: a mathematical analysis”. In: *Genomics* 2.3 (1988), pp. 231–239.

[79] Eric S Lander et al. “Initial sequencing and analysis of the human genome”. In: *Nature* 409.6822 (2001), pp. 860–921.

[80] Ben Langmead and Steven L. Salzberg. “Fast gapped-read alignment with Bowtie 2”. In: *Nature Methods* 9 (Mar. 2012), pp. 357–359. DOI: 10.1038/nmeth.1923.

[81] Ben Langmead et al. “Ultrafast and memory-efficient alignment of short DNA sequences to the human genome”. In: *Genome Biology* 10 (2009), R25+. DOI: 10.1186/gb-2009-10-3-r25.

[82] Heng Li and Richard Durbin. “Fast and accurate short read alignment with Burrows-Wheeler transform.” In: *Bioinformatics (Oxford, England)* 25 (2009), pp. 1754–1760. DOI: 10.1093/bioinformatics/btp324.

[83] Heng Li, Jue Ruan, and Richard Durbin. “Mapping short DNA sequencing reads and calling variants using mapping quality scores”. In: *Genome Research* 18 (2008), pp. 1851–1858. DOI: 10.1101/gr.078212.108.

[84] Heng Li et al. “The Sequence Alignment/Map format and SAMtools”. In: *Bioinformatics* 25.16 (2009), pp. 2078–2079. DOI: 10.1093/bioinformatics/btp352.

[85] Ruiqiang Li et al. “De novo assembly of human genomes with massively parallel short read sequencing”. In: *Genome Research* 20 (2010), pp. 265–272. DOI: 10.1101/gr.097261.109.

[86] Erez Lieberman-Aiden et al. “Comprehensive mapping of long-range interactions reveals folding principles of the human genome”. In: *Science* 326.5950 (2009), pp. 289–293.

[87] Yong Lin et al. “Comparative Studies of de novo Assembly Tools for Next-generation Sequencing Technologies”. In: *Bioinformatics* 27.15 (2011), pp. 2031–2037. DOI: `10.1093/bioinformatics/btr319`.

[88] Ryan Lister et al. “Human DNA methylomes at base resolution show widespread epigenomic differences”. In: *Nature* 462.7271 (2009), pp. 315–322.

[89] Lin Liu et al. “Comparison of next-generation sequencing systems”. In: *BioMed Research International* 2012 (2012).

[90] Nicholas J Loman et al. “Performance comparison of benchtop high-throughput sequencing platforms”. In: *Nature Biotechnology* 30.5 (2012), pp. 434–439.

[91] Julius B Lucks et al. “Multiplexed RNA structure characterization with selective 2’-hydroxyl acylation analyzed by primer extension sequencing (SHAPE-Seq)”. In: *Proceedings of the National Academy of Sciences* 108.27 (2011), pp. 11063–11068.

[92] R Luo et al. “SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler”. In: *Gigascience* 1 (2012), p. 18. DOI: `10.1186/2047-217X-1-18`.

[93] Elaine R Mardis. “Next-generation DNA sequencing methods”. In: *Annu. Rev. Genomics Hum. Genet.* 9 (2008), pp. 387–402.

[94] Elaine R Mardis. “Next-generation sequencing platforms”. In: *Annual Review of Analytical Chemistry* 6 (2013), pp. 287–303.

[95] Elaine R Mardis. “The impact of next-generation sequencing technology on genetics”. In: *Trends in genetics* 24.3 (2008), pp. 133–141.

[96] M Margulies et al. “Genome Sequencing in Microfabricated High-Density Picolitre Reactors”. In: *Nature* 437 (2005), pp. 376–380. DOI: `10.1038/nature03959`.

[97] Guillaume Marais and Carl Kingsford. “A fast, lock-free approach for efficient parallel counting of occurrences of k-mers”. In: *Bioinformatics* 27.6 (2011), pp. 764–770. DOI: `10.1093/bioinformatics/btr011`.

[98] Allan M Maxam and Walter Gilbert. “A new method for sequencing DNA”. In: *Proceedings of the National Academy of Sciences* 74.2 (1977), pp. 560–564.

[99] Paul Medvedev and Michael Brudno. “Maximum likelihood genome assembly.” In: *Journal of Computational Biology* 16 (2009), pp. 1101–1116. DOI: `10.1089/cmb.2009.0047`.

[100] Paul Medvedev et al. “Computability of Models for Sequence Assembly”. In: *Algorithms in Bioinformatics*. Ed. by Raffaele Giancarlo and Sridhar Hannenhalli. Vol. 4645. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2007, pp. 289–301.

[101] Paul Medvedev et al. “Paired de bruijn graphs: a novel approach for incorporating mate pair information into genome assemblers”. In: *Journal of Computational Biology* 18.11 (2011), pp. 1625–1634.

[102] Michael L Metzker. “Sequencing technologies – the next generation”. In: *Nature Reviews Genetics* 11.1 (2010), pp. 31–46.

[103] Tarjei S Mikkelsen et al. “Genome-wide maps of chromatin state in pluripotent and lineage-committed cells”. In: *Nature* 448.7153 (2007), pp. 553–560.

[104] Jason R Miller, Sergey Koren, and Granger Sutton. “Assembly algorithms for next-generation sequencing data”. In: *Genomics* 95.6 (2010), pp. 315–327.

[105] Marcos Morey et al. “A glimpse into past, present, and future DNA sequencing”. In: *Molecular Genetics and Metabolism* 110.1 (2013), pp. 3–24.

[106] Ali Mortazavi et al. “Mapping and quantifying mammalian transcriptomes by RNA-Seq”. In: *Nature Methods* 5.7 (2008), pp. 621–628.

[107] E W Myers et al. “A Whole-Genome Assembly of Drosophila”. In: *Science* 287 (2000), pp. 2196–2204. DOI: `10.1126/science.287.5461.2196`.

[108] Eugene W. Myers. “The fragment assembly string graph”. In: *Bioinformatics* 21 (2005), pp. 79–85. DOI: `http://dx.doi.org/10.1093/bioinformatics/bti1114`.

[109] Eugene W Myers. “Toward simplifying and accurately formulating fragment assembly”. In: *Journal of Computational Biology* 2.2 (1995), pp. 275–290.

[110] Eugene W Myers et al. “A whole-genome assembly of Drosophila”. In: *Science* 287.5461 (2000), pp. 2196–2204.

[111] Kari-Matti Mkel et al. “Genome-Wide Association Study Pinpoints a New Functional ApoB Variant Influencing Oxidized LDL Levels but Not Cardiovascular Events: AtheroRemo Consortium”. In: *Circulation: Cardiovascular Genetics* (2012). DOI: `10.1161/CIRCGENETICS.112.964965`.

[112] Niranjan Nagarajan and Mihai Pop. “Parametric complexity of sequence assembly: theory and applications to next generation sequencing.” In: *Journal of Computational Biology* 16 (2009), pp. 897–908. DOI: `10.1089/cmb.2009.0005`.

[113] Niranjan Nagarajan and Mihai Pop. “Sequence assembly demystified”. In: *Nature Reviews Genetics* 14.3 (2013), pp. 157–167.

[114] Niranjan Nagarajan, Timothy D Read, and Mihai Pop. “Scaffolding and validation of bacterial genome assemblies using optical restriction maps”. In: *Bioinformatics* 24.10 (2008), pp. 1229–1235.

[115] Katsuto Nakajima, S. Louis Hakimi, and Jan Karel Lenstra. “Complexity results for scheduling tasks in fixed intervals on two types of machines”. In: *SIAM Journal on Computing* 11.3 (1982), pp. 512–520.

[116] Giuseppe Narzisi and Bud Mishra. “Comparing De Novo Genome Assembly: The Long and Short of It”. In: *PLoS ONE* 6 (Apr. 2011), e19175. DOI: `10.1371/journal.pone.0019175`.

[117] Rasmus Nielsen et al. "SNP Calling, Genotype Calling, and Sample Allele Frequency Estimation from New-Generation Sequencing Data". In: *PLoS ONE* 7.7 (July 2012), e37558. DOI: `10.1371/journal.pone.0037558`.

[118] *Noncommunicable diseases country profiles 2011.* World Health Organization, 2011.

[119] Karl J. V. Nordstrom et al. "Mutation identification by direct comparison of whole-genome sequencing data from mutant and wild-type individuals using k-mers". In: *Nature Biotechnology* 31.4 (Mar. 2013), pp. 325–330. DOI: `10.1038/nbt.2515`.

[120] I Pagani et al. "The Genomes OnLine Database (GOLD) v. 4: status of genomic and metagenomic projects and their associated metadata". In: *Nucleic Acids Research* 40 (2012), pp. D571–D579. DOI: `10.1093/nar/gkr1100`.

[121] Chandra Shekhar Pareek, Rafal Smoczynski, and Andrzej Tretyn. "Sequencing technologies and genome sequencing". In: *Journal of Applied Genetics* 52.4 (2011), pp. 413–435.

[122] Rob Patro, Stephen M. Mount, and Carl Kingsford. "Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms". In: *Nature Biotechnology* 32.5 (May 2014), pp. 462–464. DOI: `10.1038/nbt.2862`.

[123] Pavel A Pevzner and Haixu Tang. "Fragment assembly with double-barreled data". In: *Bioinformatics* 17.suppl 1 (2001), S225–S233.

[124] Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. "An Eulerian path approach to DNA fragment assembly". In: *Proceedings of the National Academy of Sciences* 98 (2001), pp. 9748–9753. DOI: `10.1073/pnas.171285098`.

[125] Son K Pham et al. "Pathset graphs: a novel approach for comprehensive utilization of paired reads in genome assembly". In: *Journal of Computational Biology* 20.4 (2013), pp. 359–371.

[126] Adam Phillippy, Michael Schatz, and Mihai Pop. "Genome assembly forensics: finding the elusive mis-assembly". In: *Genome Biology* 9 (2008), R55. DOI: `10.1186/gb-2008-9-3-r55`.

[127] M Pop, DS Kosack, and SL Salzberg. "Hierarchical scaffolding with Bambus". In: *Genome Research* 14 (2004), pp. 149–159.

[128] Mihai Pop. "Genome assembly reborn: recent computational challenges". In: *Briefings in Bioinformatics* (2009), bbp026.

[129] Mihai Pop and Steven L Salzberg. "Bioinformatics challenges of new sequencing technology". In: *Trends in Genetics* 24.3 (2008), pp. 142–149.

[130] Michael A Quail et al. "A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers". In: *BMC Genomics* 13.1 (2012), p. 341.

[131] Aaron R. Quinlan and Ira M. Hall. “BEDTools: a flexible suite of utilities for comparing genomic features”. In: *Bioinformatics* 26.6 (2010), pp. 841–842. DOI: `10.1093/bioinformatics/btq033`.

[132] Atif Rahman and Lior Pachter. “CGAL: computing genome assembly likelihoods”. In: *Genome Biology* 14.1 (2013), R8. DOI: `10.1186/gb-2013-14-1-r8`.

[133] Tobias Rausch et al. “A consistency-based consensus algorithm for de novo and reference-guided sequence assembly of short reads”. In: *Bioinformatics* 25.9 (2009), pp. 1118–1124.

[134] Jason A Reuter, Damek V Spacek, and Michael P Snyder. “High-Throughput Sequencing Technologies”. In: *Molecular Cell* 58.4 (2015), pp. 586–597.

[135] Adam Roberts et al. “Improving RNA-Seq expression estimates by correcting for fragment bias”. In: *Genome Biology* 12 (2011), R22. DOI: `10.1186/gb-2011-12-3-r22`.

[136] Angharad M. Roberts et al. “Integrated allelic, transcriptional, and phenomic dissection of the cardiac effects of titin truncations in health and disease”. In: *Science Translational Medicine* 7.270 (2015), 270ra6. DOI: `10.1126/scitranslmed.3010134`.

[137] Jonathan M Rothberg et al. “An integrated semiconductor device enabling non-optical genome sequencing”. In: *Nature* 475.7356 (2011), pp. 348–352.

[138] Silvi Rouskin et al. “Genome-wide probing of RNA structure reveals active unfolding of mRNA structures in vivo”. In: *Nature* 505.7485 (2014), pp. 701–705.

[139] L Salmela et al. “Fast scaffolding with small independent mixed integer programs”. In: *Bioinformatics* 27 (2011), pp. 3259–3265. DOI: `10.1093/bioinformatics/btr562`.

[140] Steven L Salzberg and James A Yorke. “Beware of mis-assembled genomes”. In: *Bioinformatics* 21.24 (2005), pp. 4320–4321.

[141] Steven L. Salzberg et al. “GAGE: A critical evaluation of genome assemblies and assembly algorithms”. In: *Genome Research* (2011). DOI: `10.1101/gr.131383.111`.

[142] F. Sanger, S. Nicklen, and A. R. Coulson. “DNA sequencing with chain-terminating inhibitors”. In: *Proceedings of the National Academy of Sciences* 74 (1977), pp. 5463–5467.

[143] Gabriel Santpere et al. “Genome-wide analysis of wild-type Epstein-Barr virus genomes derived from healthy individuals of the 1000 Genomes Project”. In: *Genome Biology and Evolution* (2014). DOI: `10.1093/gbe/evu054`.

[144] Nazmus Saquib et al. “Cardiovascular diseases and Type 2 Diabetes in Bangladesh: A systematic review and meta-analysis of studies between 1995 and 2010”. In: *BMC Public Health* 12.1 (2012), p. 434. DOI: `10.1186/1471-2458-12-434`.

[145] Korbinian Schneeberger et al. “Reference-guided assembly of four diverse Arabidopsis thaliana genomes”. In: *Proceedings of the National Academy of Sciences* 108.25 (2011), pp. 10249–10254.

[146] Jay Shendure and Hanlee Ji. “Next-generation DNA sequencing”. In: *Nature Biotechnology* 26.10 (2008), pp. 1135–1145.

[147] Jared T. Simpson et al. “ABySS: A parallel assembler for short read sequence data”. In: *Genome Research* 19 (2009), pp. 1117–1123. DOI: `10.1101/gr.089532.108`.

[148] JT Simpson and R Durbin. “Efficient de novo assembly of large genomes using compressed data structures”. In: *Genome Research* 22 (2012), pp. 549–556. DOI: `10.1101/gr.126953.111`.

[149] Douglas Smith et al. “Rapid whole-genome mutational profiling using next-generation sequencing technologies”. In: *Genome Research* 18 (2008), pp. 1638–1642. DOI: `10.1101/gr.077776.108`.

[150] Kai Song et al. “New developments of alignment-free sequence comparison: measures, statistics and next-generation sequencing”. In: *Briefings in Bioinformatics* 15.3 (2014), pp. 343–353. DOI: `10.1093/bib/bbt067`.

[151] Karin Y. van Spaendonck-Zwarts et al. “Titin gene mutations are common in families with both peripartum cardiomyopathy and dilated cardiomyopathy”. In: *European Heart Journal* (2014). DOI: `10.1093/eurheartj/ehu050`.

[152] Granger G Sutton et al. “TIGR Assembler: A new tool for assembling large shotgun sequencing projects”. In: *Genome Science and Technology* 1.1 (1995), pp. 9–19.

[153] Tanya M. Teslovich et al. “Biological, clinical and population relevance of 95 loci for blood lipids”. In: *Nature* 7307 (2010), 707713. DOI: `10.1038/nature09270`.

[154] Cole Trapnell et al. “Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation”. In: *Nature Biotechnology* 28.5 (2010), pp. 511–515.

[155] Esko Ukkonen. “Approximate string-matching with q-grams and maximal matches”. In: *Theoretical Computer Science* 92.1 (1992), pp. 191–211.

[156] Anton Valouev et al. “A high-resolution, nucleosome position map of C. elegans reveals a lack of universal sequence-dictated positioning”. In: *Genome Research* 18.7 (2008), pp. 1051–1063.

[157] J Craig Venter et al. “The sequence of the human genome”. In: *Science* 291.5507 (2001), pp. 1304–1351.

[158] Francesco Vezzi, Giuseppe Narzisi, and Bud Mishra. “Feature-by-Feature Evaluating *De Novo* Sequence Assembly”. In: *PLoS ONE* 7 (Feb. 2012), e31002. DOI: `10.1371/journal.pone.0031002`.

[159] Francesco Vezzi, Giuseppe Narzisi, and Bud Mishra. “Reevaluating Assembly Evaluations with Feature Response Curves: GAGE and Assemblathons”. In: *PLoS ONE* 7 (2012), p. 52210.

[160] Peter M Visscher et al. “Five years of GWAS discovery”. In: *The American Journal of Human Genetics* 90.1 (2012), pp. 7–24.

[161] Rene L. Warren et al. “Assembling millions of short DNA sequences using SSAKE”. In: *Bioinformatics* 23 (2007), pp. 500–501. DOI: 10.1093/bioinformatics/btl629.

[162] *World Life Expectancy*. http://www.worldlifeexpectancy.com/cause-of-death/coronary-heart-disease/by-country/.

[163] Ray Wu. “Nucleotide sequence analysis of DNA”. In: *Nature* 236.68 (1972), pp. 198–200.

[164] Ray Wu and Ellen Taylor. “Nucleotide sequence analysis of DNA: II. Complete nucleotide sequence of the cohesive ends of bacteriophage λ DNA”. In: *Journal of Molecular Biology* 57.3 (1971), pp. 491–511.

[165] Ruibin Xi et al. “Copy number variation detection in whole-genome sequencing data using the Bayesian information criterion”. In: *Proceedings of the National Academy of Sciences* 108.46 (2011), E1128–E1136.

[166] Salim Yusuf et al. “Effect of potentially modifiable risk factors associated with myocardial infarction in 52 countries (the {INTERHEART} study): case-control study”. In: *The Lancet* 364.9438 (2004), pp. 937 –952. DOI: http://dx.doi.org/10.1016/S0140-6736(04)17018-9.

[167] Daniel R. Zerbino and Ewan Birney. “Velvet: Algorithms for de novo short read assembly using de Bruijn graphs”. In: *Genome Research* 18 (2008), pp. 821–829. DOI: 10.1101/gr.074492.107.

[168] Wenyu Zhang et al. “A Practical Comparison of De Novo Genome Assembly Software Tools for Next-Generation Sequencing Technologies”. In: *PLoS ONE* 6 (Mar. 2011), e17915. DOI: 10.1371/journal.pone.0017915.

[169] Qi Zheng et al. “Genome-wide double-stranded RNA sequencing reveals the functional significance of base-paired RNAs in Arabidopsis”. In: *PLoS Genetics* 6.9 (2010), e1001141.

[170] Shiguo Zhou et al. “A whole-genome shotgun optical map of Yersinia pestis strain KIM”. In: *Applied and Environmental Microbiology* 68.12 (2002), pp. 6321–6331.

[171] Aleksey V Zimin et al. “A whole-genome assembly of the domestic cow, *Bos taurus*”. In: *Genome Biology* 10.4 (2009), R42.

[172] Or Zuk et al. “The mystery of missing heritability: Genetic interactions create phantom heritability”. In: *Proceedings of the National Academy of Sciences* 109.4 (2012), pp. 1193–1198. DOI: 10.1073/pnas.1119675109.