Pan-genomic analysis of clonal bacterial samples using nanopore reads and genome graphs

Rachel Colquhoun

Wolfson College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity Term 2019

Acknowledgements

First and foremost, I would like to express gratitude to my supervisors Zamin Iqbal and Derrick Crook for their support, enthusiasm and guidance. Thank you Zam for your eternal optimism - you were usually proved right!

I would like to thank all those at the Wellcome Trust Centre for Human Genetics and at the European Bioinformatics Institute who made my time at each place so enjoyable: Phelim and Sorina for the great conversations in Oxford in the early days, Robyn, Martin and Michael whose coding input and advice has been invaluable, and all the past and present members of the Iqbal group for valuable encouragement and feedback along the way. I am also very grateful to those in the Modernising Medical Microbiology group in Oxford whose biological insights benefited my work so much, and who did all the hard work generating microbial sequence data for me to use: especially Lou, Nicole, Hang and Sophie.

I thank the Wellcome Trust and the EBI for their generous financial support.

Finally, I would like to thank my family. In particular my parents and my husband Fergus who enabled me to be where I needed to be and who believed I would get there in the end.

Abstract

Bacterial genetic variation originates through multiple mechanisms, including mutations during replication, movement of mobile elements, and various forms of recombination. As a result, genomes can be highly divergent with only a small fraction of genes core to all and a large pangenome of genes which have been identified in one or more sequenced samples.

In this context, the ability to accurately detect genetic variation throughout the pangenome and compare many genomes remains a difficult problem. Here we present a novel pangenome reference graph structure, which represents the known genetic variation within a species as a collection of 'floating' graphs. Each of these represents some homologous region such as a cluster of genes. By approximating a sequenced genome as a mosaic of genomes from the reference panel, this design forms the basis for a systematic framework in which to analyse diverse sets of samples where a single reference would be inappropriate.

Applying this method to *E. coli*, we demonstrate how it enables us to describe genetic variation at both a coarse (gene presence) and a fine (SNP/indel) level. We demonstrate how this enables us to successfully compare divergent genomes within a species, gaining dramatically higher sensitivity to SNP variation than single reference-genome approaches. We go on to demonstrate how this method enables us to investigate global genetic variation in *K. pneumoniae*, and to describe the spectrum of allele frequencies in accessory genes.

The method works for either long Nanopore or short Illumina reads, and we hope will provide the basis for addressing many questions in diverse datasets.

Contents

List of Figures

Chapter 1

Introduction

The comparison of collections of bacterial genomes allows us to answer questions about bacterial diversity, evolution and adaptation. Over recent years, the advancement of DNA sequencing technologies has hugely improved the cost and speed of sequencing whole genomes. As a result, there are now increasingly large and diverse datasets to be investigated.

Many variant calling methods however, have been designed with human genomes (and short read sequence data) in mind, where levels of diversity, presence/absence variation and rearrangement are comparatively low. It has become clear that both gene content and genome structure can be highly variable within a bacterial species and these methods can therefore be limiting.

Consider the typical paradigm of describing variation within a set of genomes by describing how each differs from a common reference genome. The cartoon in Figure 1.1 highlights the effect that the choice of a reference genome has on the proportion of nucleotide variants within a dataset that can be described using perfect reads. If a gene is in the reference we assume that we are able to call SNPs in that gene, with the exception of clustered SNPs (numbered 11-15 and 17-22): typical read mappers struggle to map reads when the sequence becomes too divergent. Variation between other genomes in genes which are not in the reference are invisible without further analysis. As a result, only 34-56% of SNP variants in the cartoon can be described with respect to any given genome.

Whilst there do exist reference free methods for sequence comparison including multiple sequence alignment, these can struggle with complex coordinate systems to describe variants, particularly as they scale to many genomes.

In recent years there have been a number of population reference approaches developed which incorporate known variation into a reference graph and are able to improve sensitivity of variant calls near structural variation as a result. These have

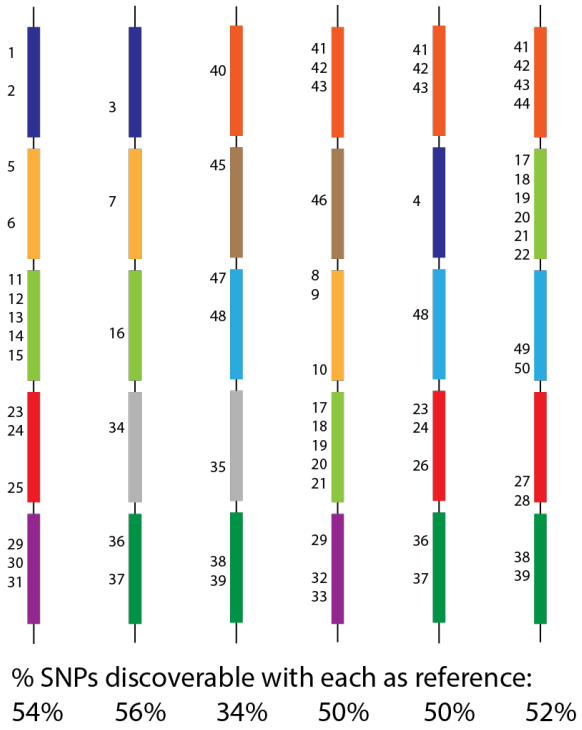


Figure 1.1: *The challenge of describing variation between diverse genomes using a single reference.* The cartoon depicts 6 bacterial 'genomes', with genes represented by coloured blocks. Numbers label 50 segregating SNPs. We calculate the percentage of these SNPs which can be detected using each of the 6 'genomes' as reference by mapping perfect reads from the remaining 5 to this reference.

mostly focused on human genomes and do not allow for structural variation at the scale of gene presence, absence and reordering and so are poorly suited to complex bacterial families. Instead, to compare such bacterial genomes, we are forced to simplify the problem and consider the fine scale variation such as SNPs only in the region of the genome shared by all samples, and coarse presence or absence information for the remainder of the genome. Fine scale variation in the accessory genome is generally inaccessible.

In addition, most of these methods only work for short read data, such as Illumina sequence data. For new long read sequencing technologies such as Oxford Nanopore Technologies, the high error rate (and concerns about systematic biases) have meant that attempts to enable variant calling have been limited in scope.

This work extends prior population reference graph ideas to the pangenome, and introduces a hierarchical framework in which all variation between sets of bacterial genomes can be described. This enables variation calling in the accessory genome for the first time and thereby facilitates high resolution comparisons of diverse sets

of genomes. Techniques are compatible with both Illumina and Nanopore sequence data and outperform the only published Nanopore variant caller.

In Chapter 3 we introduce our novel pangenome reference graph (PanRG) structure which represents known variation as a collection of 'floating' graphs, diffusing the complexity of genome rearrangement. Based on the known mechanisms introducing genetic diversity, we treat genomes as mosaics of genes, and genes as mosaics of reference haplotypes. In Chapter 4 we introduce methods for mapping long or short reads to the PanRG and inferring a closest reference mosaic for local sequences found in each sample. Chapter 5 introduces methods to genotype a single sample, and Chapter 6 expands methods to genotyping of collections of samples. In doing so, we provide the first (to our knowledge) method for systematically choosing a best reference sequence for a dataset under study. We evaluate this with simulations and then with an empirical dataset of 4 *E. coli* genomes selected from across the *E. coli* phylogroups. With this small dataset we demonstrate starkly the effect shown above in Figure 1.1 - reference based methods have much lower recall than this approach. The impatient reader is invited to look at Figure 6.8. We go on to describe an extension of this work by another PhD student to enable *de novo* discovery of variation.

Finally, in Chapter 7 we apply our methods in two contexts where analysis was previously hard or impossible. First to a diverse set of genomes from surveillance sequencing following a carbapenem resistant *E. coli* outbreak in a hospital, and second to evaluate a model for the site frequency spectrum of dispensable genes.

A note on originality

All the work for this thesis is my own unless otherwise stated. The pronoun 'we' is used throughout as a personal stylistic preference.

A note on reproducibility

Most analyses for this thesis were performed using Nextflow pipelines [Di Tommaso et al., 2017]. Software was installed in a number of Singularity containers [Kurtzer et al., 2017], and individual processes within the Nextflow pipeline ran directly within singularity images. As a result, when the analyses are reproduced on different computers or compute clusters they can continue to make use of the same software environment and should generate the same results.

In practice, analyses were completed in a progressive fashion, with small improvements and bug fixes to software throughout. When there was no cause for concern,

old analyses were not reproduced with the latest singularity image due to limitations of time and resources, so small variations in results are to be expected.

Nextflow pipelines and commands used are available at [https://github.com/rmcolq/DPhil_analysis]. Singularity images used for this analysis are automatically downloaded by the Nextflow pipeline, and are available at [http://singularity-hub.org/collections/1297] and [http://singularity-hub.org/collections/1285].

Chapter 2

Background

There is extensive genetic diversity within many bacterial species. Much of this diversity arises as a result of inheritance mechanisms which are not present in eukaryotic genomes. As a result, tools which are designed to describe variation between a set of eukaryotic genomes are often unsuited to describe the variation between bacterial genomes.

In this section we will briefly outline the main mechanisms of these modes of inheritance and describe their effect on the diversity of bacterial pangenomes. We will then summarize some of the existing methods for comparison of bacterial genomes and their limitations when considering horizontally acquired variation. We will discuss how variant calling tools can be evaluated. Finally we will outline some of the specifics of the work undertaken in this thesis.

2.1 Inheritance in bacteria

Mechanisms of inheritance

Bacterial genomes rapidly evolve as a result of both *vertical* and *horizontal* modes of inheritance. Point mutations may occur during replication, indels may occur due to strand slippage and both point mutations and indels arise as a result of DNA damage and repair [Lovett, 2004]. These variants are passed *vertically* from parent to daughter cells during replication. In addition, DNA can be transferred between bacterial cells by three main mechanisms:

natural transformation : uptake of cell free DNA by a recipient cell

transduction : a recipient cell is infected by a bacteriophage containing donor material

conjugation : the transfer of mobile genetic elements (such as plasmids or transposons) between two adjacently located bacteria through direct contact with pili. [Furuya and Lowy, 2006]

In most of these cases, the new DNA is not able to self replicate, and must be incorporated into the genome by recombination to persist over generations. Figure 2.1 depicts several modes by which this can occur. *Homologous* recombination replaces a

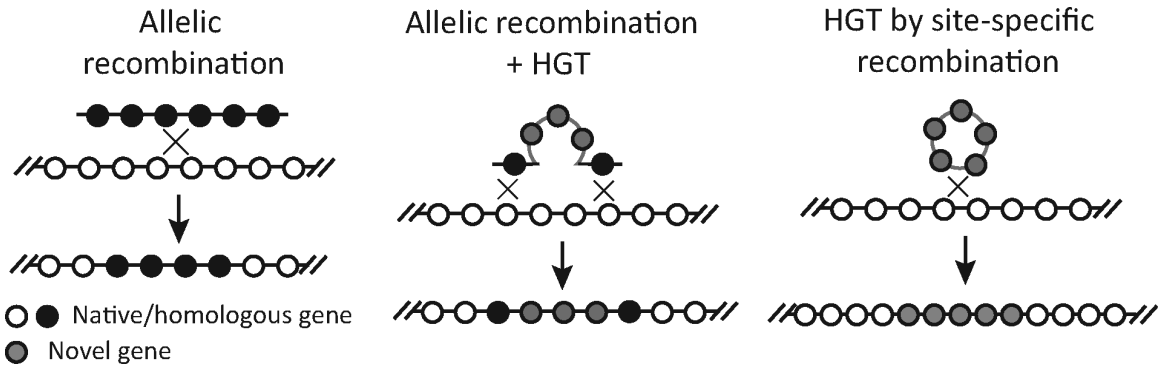


Figure 2.1: *Modes of recombination incorporating DNA into the genome.* Source: [Rocha, 2018]. Reprinted from E. P. Rocha, 'Neutral theory, microbial practice: Challenges in bacterial population genetics', Molecular Biology and Evolution, (2018), 35(6):1338−1347, by permission of Oxford University Press.

fragment of one genome with very similar sequence from another genome. When this results in the conversion from one allele to another, it is called *allelic recombination* [Rocha, 2018]. For homologous recombination to take place, the new DNA must contain regions of between 25 and 200 bp of sequence which are highly similar to the recipient genome [Thomas and Nielsen, 2005].

If no such regions of sequence homology exist, non-homologous recombination can integrate material between short regions of as little as 5 to 12 bp shared sequence [Thomas and Nielsen, 2005] with specific recombinases [Rocha, 2018]. When novel genes are acquired as a result of homologous or site-specific recombination, this is called *horizontal gene transfer*. For many species, these horizontally acquired genes are inserted at a small number of hotspots between core genes [Oliveira et al., 2017].

As a result, locally we expect a newly sequenced version of a gene to look like a mosaic of those previously seen due to allelic recombination. Globally, we expect to see different and reordered genetic loci between genomes as a result of horizontal gene transfer.

The relative rates of horizontal and vertical inheritance

For many species, the amount of horizontally introduced variation is not insignificant. The relative rates of nucleotide substitution due to allelic recombination and point mutation give an indication of the relative impact of each on sequence diversification [Didelot and Maiden, 2010]. For *Escherichia coli* this rate has been estimated at 1.02, with some recombination hotspots having a higher rate [Didelot et al., 2012]. For *Klebsiella pneumoniae* this rate is estimated to be 0.3 and for *Salmonella enterica* 30.2 [Vos and Didelot, 2009]. A new preprint [Sakoparnig et al., 2019] has demonstrated that the local genealogy for a collection of bacterial genomes changes many thousands of times along the core genome alignment, and in this sense recombination dominates genome evolution. For species like these then, recombination cannot be ignored when making inferences about variation within sets of genomes.

2.2 The pangenome

Horizontal gene transfer between and within bacterial populations gives rise to gene content variability between genomes of the same species. The result is that *core* genes are present in almost every genome of that species, whilst others are only found in some and are described as *accessory* or *dispensable* genes. The full collection of genetic material seen in genomes of a species is called the *pangenome*.

The degree of variability and the overall size of the pangenome varies greatly between species and a species is described as having a *closed* pangenome if it is of limited size, and *open* if is more extensive. Within an open pangenome it is typical for gene orthologues to be either very common and present in nearly every genome, or very rare. The resulting gene orthologue frequency distribution has a characteristic U-shape as shown in Figure 2.2.

Gene repertoire relatedness may be poorly correlated with phylogenetic distance as estimated from SNPs in the core genome as shown in Figure 2.3. A pair of genomes which are only distantly related based on a core genome phylogeny can have more similar gene repertoire than a pair of closely related genomes. Further, it is not uncommon for pairs of genomes from different species to have more orthologs than pairs of genomes from the same species as demonstrated for *Acinetobacter* spp in [Touchon et al., 2014].

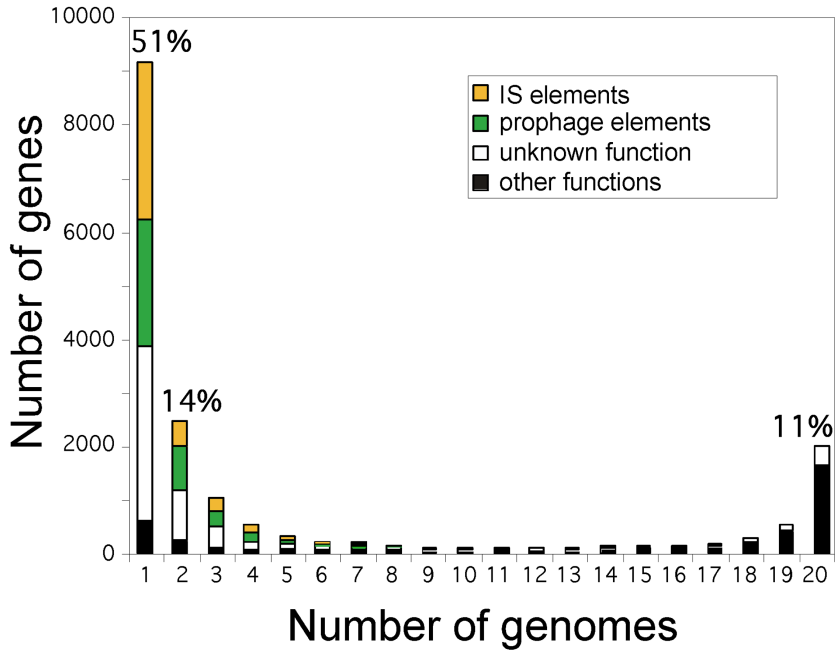


Figure 2.2: *Frequency of genes within 20 analyzed E. coli genomes.* 51% of pangenome genes were present in a single genome, while only 11% of the pangenome was made up of core genes found in all 20 genomes. Source: [Touchon et al., 2009]. Reprinted from Touchon *et al.*, 'Organised genome dynamics in the *Escherichia coli* species results in highly diverse adaptive paths', PLoS Genetics (2009), 5(1):e1000344 under CC BY.

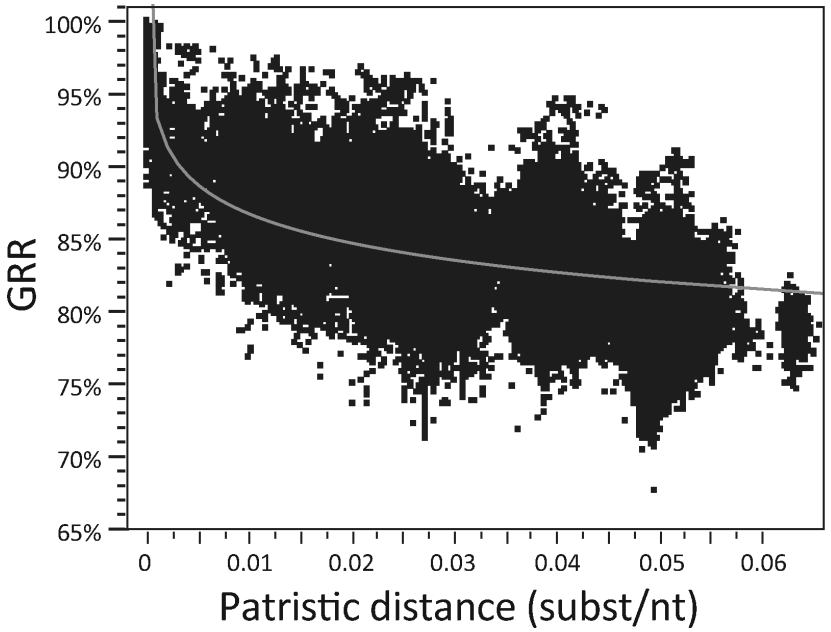


Figure 2.3: *Gene repertoire and phylogenetic relatedness in E. coli.* GRR is defined as the number of gene families present in both genomes divided by the number of gene families in the genome with fewer families. Source: [Rocha, 2018]. Reprinted from E. P. Rocha, 'Neutral theory, microbial practice: Challenges in bacterial population genetics', Molecular Biology and Evolution (2018), 35(6):1338−1347, by permission of Oxford University Press.

2.2.1 The size of the pangenome

Figure 2.4 shows the relative size of the core and accessory genome for 11 current bacterial resistance threats. For many of these, recombination events occur as often as mutations. In addition, many of these species have a relatively small core genome proportion.

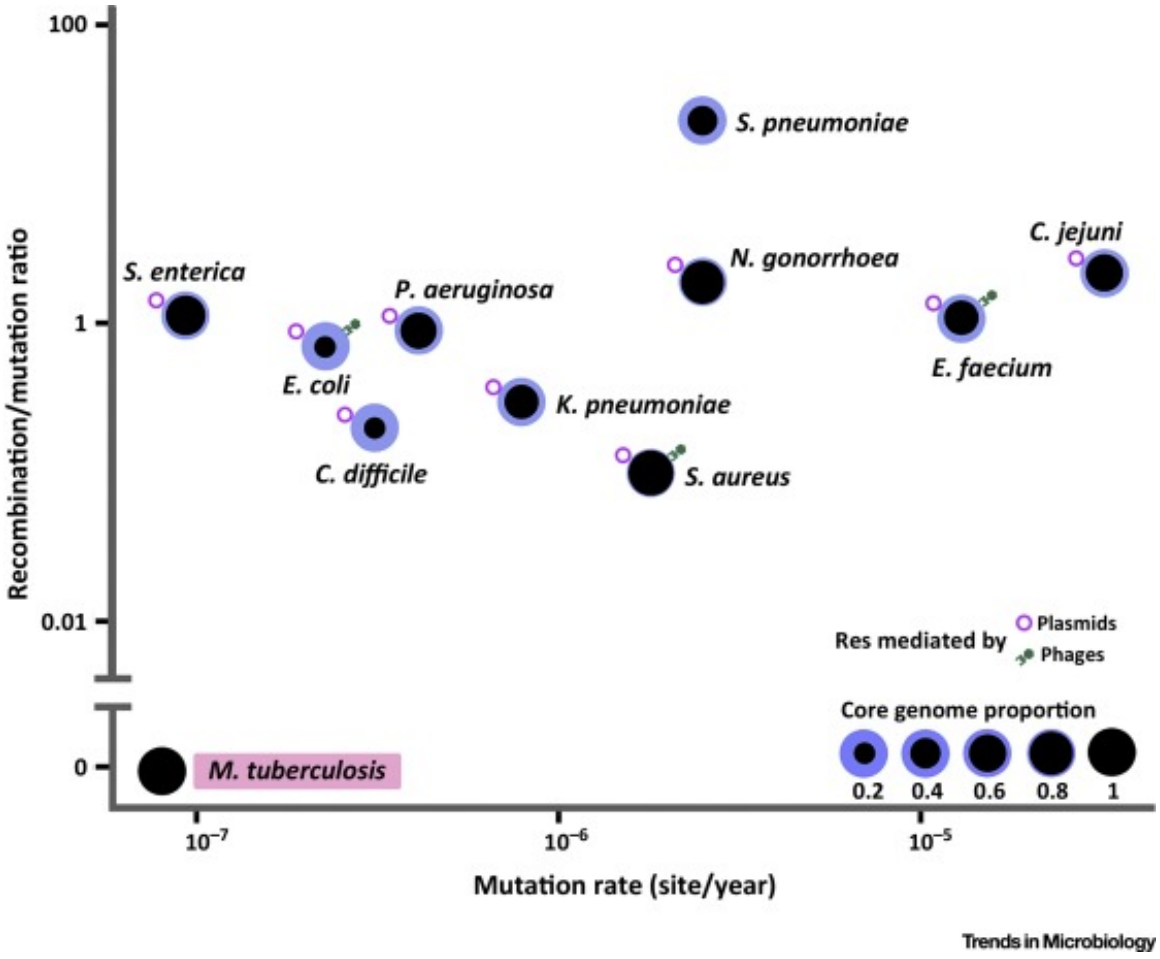


Figure 2.4: *Basic genomic features of bacterial resistance threats.* The core genome proportion is the proportion of a typical individual genome which is common to all genomes of that species. The mutation rate is estimated from whole-genome phylogenetic analyses of clinical isolates. Presence of plasmid- or phage-mediated resistance (Res) is indicated. Source: [Eldholm and Balloux, 2016]. Reprinted from Trends in Microbiology, 24(8), V. Eldholm and F. Balloux, 'Antimicrobial Resistance in *Mycobacterium tuberculosis*: The Odd One Out', 637−648, Copyright (2016), with permission from Elsevier.

The existence and size of the pangenome reflects the diversity of environmental niches in which these species can exist. Transfer of genetic material using plasmids and phages allows material to be exchanged with other strains or species, enabling adaptation to different conditions. [Croll and McDonald, 2012].

2.2.2 Describing bacterial ancestry with trees

Phylogenetic analyses of bacterial datasets are frequently used both to investigate the relationships between isolates within that particular dataset, as well as to investigate the nature of evolution within that species generally. Where recombination means that a tree is inappropriate, the standard representation is the Ancestral Recombination Graph (ARG) introduced by [Griffiths, 1981] and [Hudson, 1983].

Vertically inherited variation between collections of genomes is inherently well suited to being described by a phylogenetic tree. For many species of bacteria however, the mechanisms of homologous and non-homologous recombination also act with significant evolutionary force. When recombination is ignored, this can lead to an overestimation of the number of mutations along branches [Schierup and Hein, 2000].

There have been a number of efforts to account for recombination within phylogenies. Methods introduced by [Didelot and Falush, 2007] and [Didelot and Wilson, 2015] define the subset of the genome that has not undergone recombination (the Clonal Frame) for each branch in the genealogy. They stop short of attempting to identify the origin of regions of homologous recombination due to its complexity. [Vaughan et al., 2017] goes one step further and implements a Bayesian method to infer the population history from genetic sequence data and reconstruct the ARG, although this method is limited by its ability to scale.

2.2.3 The biological significance of accessory variation

Comparison of the core genome in a collection of samples captures how those samples are related through vertical inheritance mechanisms. This can be used to identify which bacterial samples are part of clonal outbreak and which are not [den Bakker et al., 2014], [Jackson et al., 2016], as well as to identify a transmission history and the putative source [Köser et al., 2012], [Snitkin et al., 2012], [Roetzer et al., 2013]. When used as part of routine surveillance it can help identify putative outbreaks and enable effective resolution methods to be rapidly enforced [den Bakker et al., 2014], [Pightling et al., 2018], [Durand et al., 2018].

However there are many biological problems of interest which involve the accessory part of the genome. We provide examples of a few below.

AMR and virulence genes

The widespread use of antibiotics in healthcare and agriculture over recent decades has led to a rise in the frequency of antibiotic resistance, and this is now one of the biggest threats to global health [WHO, 2014]. Clinical isolates which are resistant to most drugs have been seen for species including *Mycobacterium tuberculosis, Enterococcus faecium, Staphylococcus aureus, Klebsiella pneumoniae, Neisseria gonorrhoeae, Acinetobacter baumannii* and *Pseudomonas aeruginosa* [Nathan and Cars, 2014].

It is becoming increasingly important to understand what factors most significantly contribute to this rise in resistance and what the biological mechanisms of resistance are. Genes encoding for antimicrobial resistance and virulence are usually found in the accessory genome, often gained by horizontal transfer of mobile genetic elements.

The presence or absence of whole genes alone is not sufficient to predict which classes of antibiotic a bacteria is susceptible to. For example, the TEM-1 β-lactamase gene (found in many gram-negative species) has evolved to have more than 90 distinct descendent genes, some of which confer resistance to a much wider range of β-lactam antibiotics than the ancestral TEM-1 β-lactamase [Hall, 2004].

In [Holt et al., 2015], they demonstrated that hypervirulent clones of *Klebsiella pneumoniae* are significantly associated with invasive community-acquired disease, whilst antimicrobial resistance genes are common among human carriage isolates and hospital-acquired infections, which generally lack the genes associated with invasive disease. They called for surveillance to track extensively drug resistant hypervirulent clones as they emerge. Early this year a preprint identified South and Southeast Asia as high risk regions for emerging antimicrobial resistant and hypervirulent *K. pneumoniae* clones [Wyres et al., 2019] and a strain of ST23 *K. pneumoniae* isolated from a patient in China was reported by [Shen et al., 2019] to contain a rare hybrid plasmid with virulence genes and $bla_{CTX-M-24}$.

Mobile elements can cause outbreaks

An investigation into a non-clonal multispecies outbreak of bla_{KPC} at in a single institution found evidence of bla_{KPC} being spread at multiple genetic levels, resulting in a high level of diversity in bla_{KPC}-positive *Enterobacteriaceae* isolates [Sheppard et al., 2016].

Understanding asymptomatic enteropathogenic *E. coli*

Strains of *E. coli* are often classified based on the presence of absence of different factors in the accessory genome. Enteropathogenic *E. coli* (EPEC) for example are characterized by their able to form lesions on the surfaces of intestinal epithelial cells due to a chromosomal pathogenicity island called the locus of enterocyte effacement, and absence of a phage carrying Shiga toxin. They are further classed as 'typical' or 'atypical' based on the presence or absence of an adherence factor plasmid (pEAF). These strains are a common cause outbreaks of diarrhoea in children, although they can be carried asymptomatically and it remains unclear what differentiates these cases [Croxen et al., 2013].

The accessory genome may drive host adaptation

Many species of bacteria are able to inhabit multiple ecological niches, including the environment and multiple host species. For example, *S. aureus* is a major cause of infection in both humans and livestock. A recent paper [Richardson et al., 2018] found evidence of numerous ancient and recent host switching events, with humans acting as a major hub. In addition, they found accessory genes in mobile elements which were responsible for adaptation to a host after a switch event.

2.3 DNA Sequencing technologies

In 1995, the genome for *Haemophilus influenzae* became the first published bacterial whole genome [Fleischmann et al., 1995]. Both this and the human genome, finished in draft form in 2001 [Lander et al., 2001], [Venter et al., 2001], used shotgun Sanger sequencing. Since then, there have been several waves of innovation in DNA sequencing technologies.

The introduction of Roche/454 pyrosequencing in 2005 and Illumina/Solexa sequencing in 2006 marked the beginning of High Throughput Sequencing (HTS) or Next Generation Sequencing (NGS) technologies. These enabled whole human genomes to be sequenced in a matter of days at a cost of less than $1000 (compared with an estimated 0.5 billion dollars for the first human genome [Reuter et al., 2015]), although they typically produced read lengths ranging from 35 to 400 bp, much shorter than the reads of up to 800bp used for the first human genome. In recent years, Illumina has come to dominate the NGS market. Reads are highly accurate with overall error rates $< 1\%$ and typically range in length from 100-400bps. Assemblies from these reads tend therefore to be accurate ($> 99.99\%$) but fragmented [Powers et al., 2013].

A third generation of sequencing technologies took off in 2009 with the advent of SMRT sequencing by Pacific Biosciences [Eid et al., 2009], followed by the Oxford Nanopore Technologies MinION in 2014. These technologies are able to sequence significantly longer reads of upwards of 10,000bps enabling the structure within complex or repetitive regions of the genome to be resolved in a way that was not possible with short read technologies [Koren et al., 2012]. However, they suffer from much higher error rates of 5-15% including systematic bias [Watson and Warr, 2019].

Oxford Nanopore Technologies in particular have received a lot of attention in recent years. The sequencing machine itself is cheap and portable and can be plugged into the USB port of a high-performance laptop, with data output in real time. As a result, it has potential to be used in locations where there is little existing infrastructure, and the technology has already been used for surveillance in West Africa during the Ebola outbreak [Quick et al., 2016], in the Americas for phylogenetic analysis of the Zika outbreak e.g. [Faria et al., 2017] and even on the International Space Station [Castro-Wallace et al., 2017]. However, it takes more than 168 CPU hours of assembling and polishing to achieve a draft genome with 99.8% accuracy [Koren et al., 2017] (falling recently to 91 CPU hours polishing time to get 99.96% accuracy [`http://simpsonlab.github.io/2017/06/30/nanopolish-v0.7.0/`]). In just the last couple of months, at least 5 new assemblers and polishing tools have been announced or released, so these figures are set to improve.

Due to the high error rate, much of the software development around Nanopore sequencing to date has focused on mapping (e.g. `GraphMap` [Sović et al., 2016], `Minimap2` [Li, 2018]), assembly (e.g. `Canu` [Koren et al., 2017], `Unicycler` [Wick et al., 2017b], `Flye` [Kolmogorov et al., 2019], `Ra` [`https://github.com/rvaser/ra`], `Wtdbg2` [Ruan and Li, 2019], `SHASTA` [`https://github.com/chanzuckerberg/shasta`]) and pre- or post- assembly polishing (e.g. `Racon` [Vaser et al., 2017], `Pilon` [Walker et al., 2014], `Nanopolish` [Loman et al., 2015] and `Medaka` [`https://nanoporetech.github.io/medaka/index.html`]) rather than variant calling.

More details about these respective sequencing technologies are described in a number of good reviews including [Schatz et al., 2010], [Reuter et al., 2015] and [Goodwin et al., 2016]. In this thesis, we focus on methods for variant calling which can be applied to Nanopore reads as well as reads from the current market leader Illumina.

2.4 How do we compare bacterial genomes?

2.4.1 Typing methods

In many modern settings both sequence typing and variant calling methods are routinely used to build a picture about relatedness between bacterial isolates. Whilst sequence typing is not the focus of this thesis, it is a fundamental tool in pathogen outbreak surveillance [Pérez-Losada et al., 2017] and has some parallels with the methods we have developed.

2.4.1.1 MLST

Multi-locus Sequence Typing (MLST), was introduced as a method to classify both pathogenic and non-pathogenic bacterial species so that results could be compared between different laboratories [Maiden et al., 1998]. The original method involves classifying each genome as a known 'type' based on the nucleotide sequences of internal fragments of a small number (typically less than 10) of housekeeping genes. As whole genome sequence data has become more widely available, schemes have emerged based on larger sets of genes. These include core genome MLST (cgMLST), based on the set of core genes shared by a group of related strains (generally a few hundred genes), and whole genome MLST (wgMLST) which relies on a set of thousands of genes [Maiden et al., 2013]. The classification of new genomes into known categories is a highly useful tool, but the limitation of such methods is that they can have lower sensitivity to discriminate between closely related strains [Jolley and Maiden, 2010] [Le and Diep, 2013]. These methods typically quantify how closely related types are based on the number of genes which differ in sequence (or presence) between them rather than the number of nucleotide differences. As a result, they are not well suited to elucidate transmission events.

Like wgMLST, the approach taken in this thesis also focuses on a gene-by-gene approach to genome comparison, and uses a database of known variation, but with the ultimate aim of describing nucleotide level variation instead of classification.

2.4.1.2 Species, Strain and AMR typing

In addition to sequence typing, there exist a number of other classification tools for both Illumina and Nanopore sequence data, allowing a high level comparison of sample genomes. For Illumina sequence data, examples include, `Kraken` [Wood and Salzberg, 2014], `MetaPhlAn` [Segata et al., 2012] and `StrainPhlAn` [Truong et al., 2017] to allow species or strain identification, and `ResFinder` [Zankari et al., 2012] or

`Mykrobe` [Bradley et al., 2015] to identify resistance genes. For Nanopore sequence data, `Krocus` [Page and Keane, 2018] performs 7 gene MLST and both `RASE` [Brinda et al., 2018] and `Mykrobe` perform resistance typing.

2.4.2 Variant calling

To infer the joint ancestry of a population we need to be able to describe the nucleotide level variation present. The process of identifying segregating genetic variation in a population is known as *variant discovery*. This often involves identifying at which positions in a reference sequence there exist polymorphisms in the population. For a prepared panel of variants, the process of *genotyping* or *genotype calling* involves using statistics to determine which alleles in the panel are supported by the sequence data for each individual. Variants in these panels are described with respect to a single reference sequence and panels often restrict to short variants such as single nucleotide polymorphisms (SNPs). The combination of variant discovery and genotyping is often called *variant calling*, or *SNP calling* when it restricts to SNP variants.

Variant calling methods broadly fall into two categories. The first category, mapping based methods, identify candidate variants by aligning reads to a reference assembly and identifying positions where reads show a consistent difference to the reference [Li et al., 2009]. An extension of this are methods which instead map reads to a graph rather than an assembly. The second category are *de novo* assembly based methods which identify candidate variants as differences between aligned regions of sample assemblies [Delcher et al., 1999], [Li, 2015] or between *de Bruijn* assembly graphs [Iqbal et al., 2012].

2.4.2.1 Mapping based methods

The first step for mapping based methods is to find the location of each read relative to the reference assembly. Read alignment tools include fast and memory efficient algorithms based on the Burrows Wheeler Transform (e.g. `BWA` [Li, 2013]) and hash-based algorithms (e.g. `Novoalign` [`novocraft.com/main/index.php`], `Stampy` [Lunter and Goodson, 2011]).

After mapping, candidate variants must be found using Bayesian, likelihood or machine learning methods, based on the number of high quality base calls which disagree with the reference, and mapping quality scores [Olson et al., 2015]. Additionally, some methods such as `Platypus` [Rimmer et al., 2014] use local assembly to add candidate variants from more divergent regions. These variants can then be

filtered based on known types of systematic errors to reduce the number of false positive variant calls (at a sensitivity cost), and individual samples are genotyped at these variant sites.

For Illumina sequence data, variant callers include `Samtools` [Li et al., 2009], GATK `UnifiedGenotyper` and `HaplotypeCaller` [Depristo et al., 2011, McKenna et al., 2010], `Platypus` [Rimmer et al., 2014] and `Freebayes` [Garrison and Marth, 2012]. There have been several systematic comparisons of combinations of these read aligners and variant callers on human genomes [Cornish and Guda, 2015, Hwang et al., 2015] and *Listeria monocytogenes* [Pightling et al., 2015].

All of these methods handle isolated SNPs, but vary in performance when SNPs become more clustered. Whilst some of these methods handle small indels or repeated sequences better than others, in general all alignment-based methods struggle to handle structural variation or genomic duplication and perform poorly in regions where there is more diversity between the reference and the sequenced genome [Alkan et al., 2011], [Nielsen et al., 2011].

For microbial genomes, a number of very similar SNP-calling pipelines have been developed combining these mapping and genotyping steps. These include `Snippy` [`https://github.com/tseemann/snippy`], `SNVPhyl` [Petkau et al., 2017] and `LyveSET` [Katz et al., 2017]. The latter two have been designed for infectious disease and foodborne outbreaks respectively. Both `Snippy` and `SNVPhyl` act as a wrapper for `BWA-MEM`, `Samtools` and `Freebayes`, whilst `LyveSET` uses `VarScan` [Koboldt et al., 2009] rather than `Freebayes` for variant calling. Given these similarities and the popularity and ease of use of the `Snippy` pipeline, we compare variant calling by `pandora` with Illumina data to those of `Snippy`. We happily note that a recent systematic comparison of bacterial SNP calling pipelines by [Bush et al., 2019] found that `Snippy` was consistently the highest-performing pipeline when considering a full range of divergent genomes.

For Nanopore sequence data however, there exist very few options for variant calling. Mapping using `Minimap2` [Li, 2018] could in theory be combined with a variant caller such as `Samtools`. However the noise, length and low quality of the reads (in particular the high frequency of indels in reads) provide a challenge when generating a pileup and in many cases there will be no fundamentally correct alignment. These methods are therefore likely to be very sensitive to details of the mapping algorithm chosen. One of the first published variant calling pipelines used `MarginAlign` and `MarginCaller` [Jain et al., 2015], although only the alignment method has been applied elsewhere. Since then, the only published variant caller is `Nanopolish` [Loman

et al., 2015], which was originally designed as a tool for polishing draft assemblies using signal-level data. The method involves modifying a 'draft' genome assembly with substitutions, insertions and deletions from the reads and evaluating if these modifications improve the probability of observing the signal-level data generated during sequencing. It has since been reworked to allow variant calling as well as detection of base modifications such as methylation. A second variant calling tool `Clairvoyante` for Nanopore data has recently been described in a preprint [Luo et al., 2018]. It allows training of a neural network model and subsequent SNP and indel calling.

Both of these approaches have been designed with variant calling in the human genome in mind and they have been demonstrated to achieve high precision and recall statistics on the Genome In A Bottle datasets (99.38% precision and 96.62% recall for `Nanopolish` and 99.06% precision and 95.19% recall for `Clairvoyante` [see `https://nanoporetech.github.io/medaka/snp.html`]). Our focus is on diverse microbial pangenomes, and since we lack adequate material to train a neural network for these, we focus on comparison with `Nanopolish` whenever signal-level data is available. In many cases only the extracted reads are made available, and in this scenario there currently exists no suitable Nanopore variant caller for comparison.

2.4.2.2 Assembly based methods

It is possible to call variants directly between pairs of aligned whole genome assemblies. Software such as `MUMmer` [Kurtz et al., 2004], or `Mugsy` [Angiuoli and Salzberg, 2011] are able to infer SNPs, indels and structural variants, but cannot assign confidence scores to these variant calls due to the uniform 1x coverage. Alternatively, *de novo* assembly and variant calling can both be performed together with `Cortex` [Iqbal et al., 2012]. This method is better suited to discovering indels and more complex or structural variation than mapping based methods [Iqbal et al., 2012]. However, it can have lower sensitivity to detect SNPs, particularly in repetitive regions [Didelot et al., 2016].

2.4.2.3 Graph based methods

There are two main limitations of single reference mapping based methods: firstly, mapping to a single reference has been demonstrated to introduce a reference bias during variant calling [Bertels et al., 2014], [Pightling et al., 2014] and secondly, they have poor sensitivity near regions of structural variation [Alkan et al., 2011]. Population reference graphs have already been used with success to alleviate these issues by encoding known variation (including large-scale structural variation such

as inversions and duplications) from multiple references into a graph and performing read alignment with respect to this graph.

The first published implementation of a population reference graph method was [Dilthey et al., 2015], which constructed a reference graph for the human MHC region. To do this they created a multiple sequence alignment of 8 haplotype sequences and collapsed shared sequence together. The resulting graph was augmented with SNPs from phase 1 of the 1000 Genomes Project. For a set of (short) reads, a hidden markov model was used to infer the most likely paths through the graph based on k-mer matches. Standard variant calling techniques were then applied between the reads and this individualized reference. This paper represented a proof of concept rather than an easily reproducible method. The ideas implemented in this thesis have many parallels with the broad umbrella of this method.

A similar method at the whole genome scale was implemented in `Graphtyper` [Eggertsson et al., 2017], which constructed a variation graph for each 50kb section of the human genome from a single reference and a panel of variants. The resulting directed acyclic graph is indexed with short overlapping k-mers and graph alignment is performed with seed and extend methods. In a complementary approach, [Garrison et al., 2018] developed `vg`, a toolkit including an efficient method for mapping reads onto arbitrary variation graphs using generalized compressed suffix arrays. In addition, [Maciuca et al., 2016] developed a method for encoding variation in a BWT to enable mapping and genotyping in complex regions of the malaria parasite *P. falciparum* genome.

However all these methods to date have been designed to use only short read (Illumina) sequence data as input. They do not handle long read Nanopore or PacBio sequence data, and the reliance of each of these methods on seed and extend methods makes them unsuitable to handle long noisy reads. In addition, each has been designed to handle the types of structural variation seen in complex regions of the human (or plasmodium) genome, rather than the extensive genome reordering seen in bacterial pangenomes.

2.4.3 How can we evaluate variant callers

Performance Metrics

There are a variety of metrics which are commonly used to evaluate the performance of variant calling pipelines. Given a set of genotyped variant calls for a haploid

individual from an algorithm and truth set of variant calls, all variants are typically classified into one of 4 categories:

True Positives (TP): Variants called by the variant caller with the same variant allele called as in the truth.

False Positives (FP): Variants called by a variant caller with an incorrect variant allele called by comparison with the truth, or which does not exist in the truth.

False negatives (FN): Variants called with a variant allele in the truth set that were not called or were called incorrectly by the variant caller (these could be incorrect in the truth panel).

True negatives (TN): Variants called as the reference allele in truth set and either called with the reference allele or not called in the variant caller set.

Based on the size of each of these categories, we can evaluate metrics as described in Figure 2.5.

Metric	Calculation	Interpretation (Ideal)	Alternative names
Accuracy	$\frac{TP+TN}{TP+FP+TN+FN}$	Ratio of correct calls to total calls and variants (1)	
Specificity	$\frac{TN}{TN+FP}$	Non-variants not called as variants relative to the total non-variants (1)	
Sensitivity	$\frac{TP}{TP+FN}$	True variants called relative to all variants (1)	Recall, true positive rate (TPR), positive call rate
Precision	$\frac{TP}{TP+FP}$	True variants called relative to total calls (1)	Positive predictive value (PPV)
False positive rate	$\frac{FP}{TN+FP}$	Non-variants called relative to the total non-variants (0)	

Figure 2.5: *Performance Metrics.* Source: [Olson et al., 2015]. Reprinted from Olson *et al.*, 'Best practices for evaluating single nucleotide variant calling methods for microbial genomics', Frontiers in Genetics (2015) under CC BY.

For many variant calling pipelines, calls are only output where there is evidence that the sample or query genome differs from a reference genome, i.e. sites where the sample has the reference allele are not typically output. When we move away from this paradigm to instead describe how a sample compares to a reference graph, the 'reference path' through the graph is arbitrary and all sites where a call has been made, whether reference or a variant allele is called, need to be evaluated. In this situation we include reference calls and update the definitions of categories as follows:

True Positives (TP): Alleles called by the variant caller which agree with the truth. Alternatively, alleles from the truth call set which are called correctly by the variant caller

False Positives (FP): Alleles called by a variant caller which are incorrect by comparison with the truth

False negatives (FN): Alleles called in the truth set which are not called or are called incorrectly in the variant caller set.

In this context we do not define True Negatives. Based on these definitions, we will use measures of precision and recall to evaluate variant calling in this thesis.

Choice of dataset

To compare the performance of multiple variant callers, each needs to be evaluated on the same dataset for which there needs to be some sort of a truth. For human variant calling pipelines there have been a number of works using the Genome In a Bottle dataset which has 'gold standard reference variant calls' to use as a truth [Cornish and Guda, 2015], [Highnam et al., 2015] and [Hwang et al., 2015]. However, bacterial genetic variation is different and tools designed to call variants in the context of smaller, circular genomes with a higher degree of structural variation are better evaluated on bacterial datasets. Perhaps the simplest way to get such a dataset is to use a reference genome and a matched dataset of reads from this genome. By introducing simulated variation into the reference genome and detecting variants with respect to this mutated truth, we know exactly what variants we expect to find.

In this thesis we will evaluate how well a variant-calling method allows us to compare two or more samples and the effect that choice of reference genome has on the subsequent variant calls. For each sample we have a high quality 'truth' assembly and matched read datasets, but we do not have an empirically correct and complete panel of known variants representing all differences between the genome sequences. We can use the true genome assembly for each sample to categorize variant calls as false positives or false negatives. To evaluate the recall we use a panel of confident variants, which may represent only a fraction of the true variants and from this estimate the true positives and false negatives in the context of this panel.

Choice of reference genome

The choice of reference genome has been shown to significantly affect the detection of SNPs and other variants [Pightling et al., 2014]. For reads from regions which are either not in the reference genome or are more genetically distinct from the reference, poor mapping and alignment of these reads to the reference sequence may cause artefacts and higher error rates or prohibit discovery of variation in these regions, creating a reference bias.

When this is used to construct a phlylogeny for a number of samples, this can result in both inaccurate tree topologies and branch lengths. These phylogenetic errors can be mitigated by merging alignments generated using multiple references from different taxa as in `REALPHY` [Bertels et al., 2014]. Alternatively, multiple references can be used as part of the SNP discovery process as in [Shen et al., 2009], however the approach used in this work does not scale to large genomes or high throughput reads and certainly not noisy long reads.

The Variant Calling Format

The Variant Calling Format (VCF) was developed for the 1000 Genomes Project as a means to store DNA polymorphism data such as SNPs, insertions, deletions and structural variants [Danecek et al., 2011], [Altshuler et al., 2012] and has since become ubiquitous in bioinformatics. Each row corresponds to a variant with respect to the reference genome, with the first 9 columns providing details about this variant. Each subsequent column corresponds to a sample, and stores information about the evidence for each allele in that sample, and a the called genotype.

In this format, the same variation can be described in several ways. For example a SNP could be described as a longer variant by inclusion of flanking sequence in both the reference and alternative allele, or the position of an allele representing an extra copy of some repeated segment can be varied, or consecutive variants can be merged to form a single variant with a number of longer alleles.

There are many tools which have been developed to evaluate the performance of variant callers for human genomes. Some of these can be applied also to microbial variant callers. Examples include the `SMaSH` benchmarking toolkit [Talwalkar et al., 2014] which estimates the precision and recall of mappers and variant callers including calculation of uncertainty due to imperfect benchmark call sets. GATK has tools to combine and compare variant and genotype calls [McKenna et al., 2010], [Depristo et al., 2011]. The `vcflib` library [`https://github.com/vcflib/vcflib`] can regularize variants, compare several vcf files and generate a ROC curve. The `USeq VcfComparator` tool [`http://useq.sourceforge.net/`] can also generate ROC curves and restrict analysis to variants inside a region specified by a bed file. However all of these require a truth call set.

A recently developed alternative from within the Iqbal group is provided by `minos` [`https://github.com/iqbal-lab-org/minos`]. A subroutine of this was designed specifically for microbial VCFs to evaluate the precision of calls when no truth call set exists but a truth assembly is available instead. The initial software was written

(a) VCF example

Header

```
##fileformat=VCFv4.1
##fileDate=20110413
##source=VCFtools
##reference=file:///refs/human_NCBI36.fasta
##contig=<ID=1,length=249250621,md5=1b22b98cdeb4a9304cb5d48026a85128,species="Homo Sapiens">
##contig=<ID=X,length=155270560,md5=7e0e2e580297b7764e31dbc80c2540dd,species="Homo Sapiens">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##ALT=<ID=DEL,Description="Deletion">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant">
```

```
#CHROM POS ID     REF  ALT    QUAL FILTER  INFO                 FORMAT     SAMPLE1  SAMPLE2
```

Body

```
1        1  .     ACG  A,AT    40  PASS    .                    GT:DP      1/1:13   2/2:29
1        2  .     C    T,CT    .   PASS    H2;AA=T              GT         0|1      2/2
1        5  rs12  A    G       67  PASS    .                    GT:DP      1|0:16   2/2:20
X      100  .     T    <DEL>   .   PASS    SVTYPE=DEL;END=299   GT:GQ:DP   1:12:.   0/0:20:36
```

(b) SNP

Alignment

```
1234
ACGT
ATGT
 ^
```

VCF representation

```
POS REF ALT
2   C   T
```

(c) Insertion

```
12345
AC-GT
ACTGT
  ^
```

```
POS REF ALT
2   C   CT
```

(d) Deletion

```
1234
ACGT
A--T
 ^^
```

```
POS REF ALT
1   ACG A
```

(e) Replacement

```
1234
ACGT
A-TT
 ^^
```

```
POS REF ALT
1   ACG AT
```

(f) Large structural variant

Alignment

```
   100       110       120       290       300
    .         .         .         .         .
ACGTACGTACGTACGTACGTACGTACGT[...]ACGTACGTACGTAC
ACGT------------------------[...]----------GTAC
```

VCF representation

```
POS REF  ALT    INFO
100  T   <DEL>  SVTYPE=DEL;END=299
```

(g) Resolving ambiguity

Alignment

```
1234567890
TTTCCCTCTA
CTTACCT--A
^  ^   ^^
```

Possible representation

```
POS  REF         ALT
1    TTTCCCTCT   CTTACCTA
```

Possible representation

```
POS REF  ALT
1   T    C
4   C    A
7   TCT  T
```

Recommended VCF representation

```
POS   REF   ALT
1     T     C
4     C     A
5     CCT   C
```

Figure 2.6: "*Example of valid VCF.* The header lines ##fileformat and #CHROM are mandatory, the rest is optional but strongly recommended. Each line of the body describes variants present in the sampled population at one genomic position or region. All alternate alleles are listed in the ALT column and referenced from the genotype fields as 1-based indexes to this list; the reference haplotype is designated as 0. ... The first data line shows an example of a deletion (present in SAMPLE1) and a replacement of two bases by another base (SAMPLE2); the second line shows a SNP and an insertion; the third a SNP; the fourth a large structural variant described by the annotation in the INFO column, the coordinate is that of the base before the variant." Source: Figure and caption from [Danecek et al., 2011]. Reprinted from Danecek *et al.*, 'The variant call format and VCFtools', Bioinformatics (2011), 27(15):2156−2158, by permission of Oxford University Press.

and developed by Martin Hunt. I added features which allow it to be compatible with graph-based VCFs. In addition, I added methods to estimate the recall for variants which differentiate a pair of samples based on a panel of SNP differences between their assemblies, and to evaluate the recall of variant callsets with respect to a simulated reference genome by comparison with the VCF of simulated variants which were introduced.

2.4.3.1 Evaluating precision with `minos`

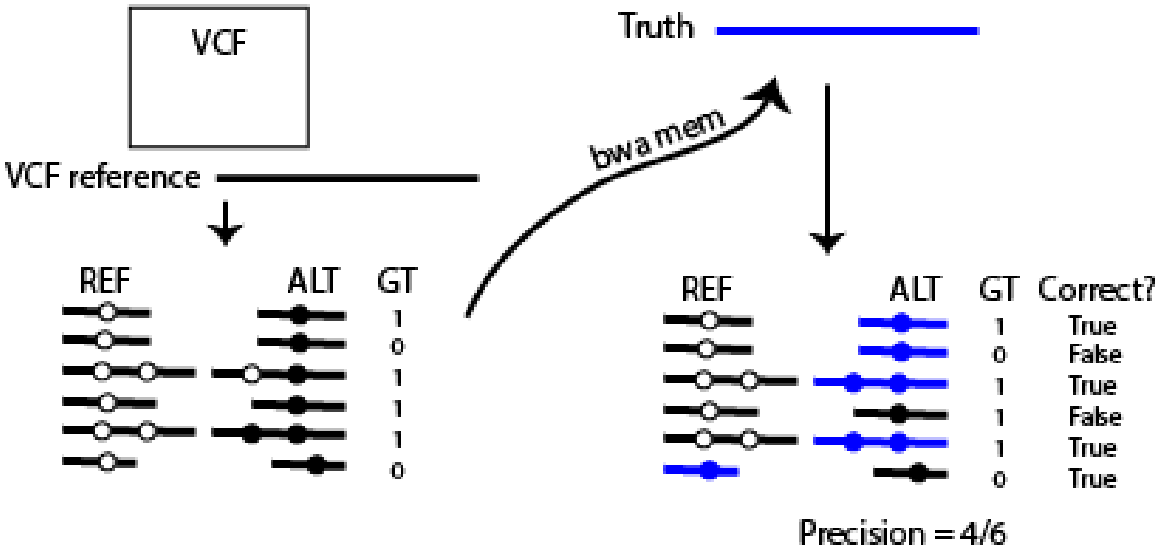


Figure 2.7: Workflow to evaluate precision from a VCF.

We evaluated the precision of a set of variant calls using a truth assembly. Each VCF allele and flanking sequence was mapped to the truth assembly with `bwa mem` as in Figure 2.7. This method is agnostic to possible differences in the way variation can be presented in a VCF, and allowed us to verify both that the called allele has a match in the truth assembly, and that it was a better match than the alternative allele. We used a flank size $f = 31$ to try and ensure only a single mapping location was found, and allow mismatches only in these flanks. We merged alleles which were located within a flanking distance to form a single longer allele for evaluation. This process allowed us to classify of each call in the VCF as a *true positive* if it mapped successfully to the truth, and a *false positive* if it did not map, or if an alternative allele mapped better. These results were summarized in tables of true positive calls and false positive calls stratified by genotype confidence.

2.4.3.2 Evaluating recall with `minos` using a truth callset

In Section 5.4.2.2 we will compare the recall of variant calling pipelines on a read dataset with a truth assembly by mutating the truth assembly using a VCF of calls. With each pipeline we then evaluate recall of these introduced variants.

By construction, we have a truth panel of variants we expect to see represented in the VCFs. To evaluate the fraction of these simulated variants which were called, we use `bwa aln` to align the reference allele from each simulated call together with its flanking sequence in the truth, to a panel of all VCF alleles and their flanking sequences in the VCF reference. If a match is found with a called allele we have a *true positive* and otherwise a *false negative*. We used a flank size $f = 9$ and merged alleles which were located within a flanking distance to form a single longer allele for evaluation. The flank size was chosen to be large enough that we expect few incorrect alignments (empirically for one *E. coli* sample, we measured that 99.2% of 19-mers were unique), and small enough that `bwa aln` was able to provide an exact alignment even after some alleles were merged.

2.4.3.3 Evaluating recall of variants segregating between two samples with `minos`

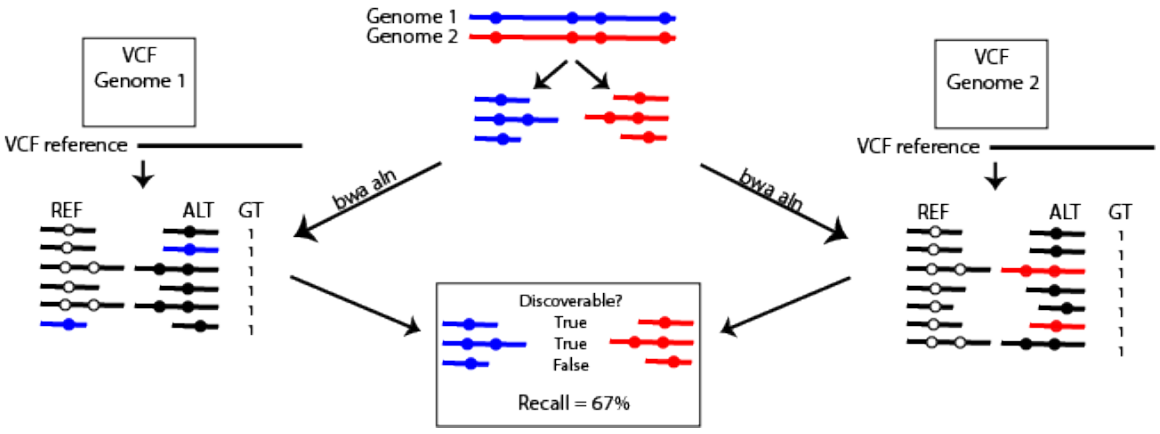


Figure 2.8: Workflow to evaluate pairwise SNP recall.

When reference-based methods call variants between samples, they do so by calling variants with respect to a single reference from reads or assemblies of each sample. The differences between the samples are then inferred as differences between these call sets.

We evaluated recall as the fraction of true differences between samples which have been described. However, instead of a truth describing what the true

differences are between the samples, we use truth assemblies for each sample. Figure 2.8 illustrates the method to estimate the pairwise SNP recall using these assemblies.

The truth assemblies for a pair of samples were compared using the `dnadiff` utility from `MUMMER3` [Kurtz et al., 2004] to establish a panel of confident SNP differences. We evaluated the fraction of these SNPs which could be discovered from the VCFs generated by each method by identifying which SNP alleles were contained in the VCF callsets.

To do this we used `bwa aln` [Li and Durbin, 2010] to map the SNP allele for each sample, together with its flanking sequence in the sample truth assembly, to a panel of all VCF alleles and their flanking sequences in that sample. This allowed us to identify variants regardless of their presentation within the VCF. If a match was found with a called allele for either sample, we said the pairwise SNP difference could be discovered. We used a flank size $f = 9$ and merged alleles which were located within a flanking distance of each other to form a single longer allele for evaluation. We chose this flank size to be sufficiently large to be confident that a match with a called allele and flank was unlikely to have occurred by chance - this would require at least a 19bp repeat elsewhere in the genome at which location we would have had to also identify a variant site. We required the flank be small enough that merged alleles could still be aligned successfully.

For each method, the total number of pairwise SNP differences which could be discovered divided by the total number of (merged) SNP differences identified with `dnadiff` was used as a measure of recall. We evaluated this recall stratified by genotype confidence of calls.

2.4.3.4 Limitations of `minos` evaluations

There are a small number of known issues when evaluating VCFs with `minos` which we outline here.

When neither the reference allele nor the alternative allele map back to the truth assembly, the variant call is classified as a false positive. There are other reasons why alleles might not map, such as sequence in the flanking region of the site being too divergent from the truth.

When the position of a variant is close to the beginning or end of the VCF reference 'chromosome' on which it lies, the flank added is truncated, i.e. only a small flank is added. Most of the time this does not cause issues, but occasionally the resulting sequence is too short for `bwa` to map. This results in additional false positive calls. In some cases we therefore exclude these regions from precision evaluations.

Merging VCF records within a flanking distance is often not possible in regions of the graph where more complex variation is represented by nested region alleles. We will return to this in a later section after defining the the graph structure. For now, the approach taken is that if there is a single unambiguous merge, that merge is performed, and otherwise the records are left as defined. The motivation behind merging alleles is to improve the accuracy of sequence in the flanking sequence so that the overall query probe is more likely to map successfully to the truth assembly. The result of this is that `pandora` VCFs (the only ones containing nested variation) are penalised in precision evaluations.

Chapter 3

A Pangenome Population Reference Graph

3.1 Motivation

For reference-based variant calling pipelines, some form of mapping to a reference sequence is a fundamental initial step [Olson et al., 2015]. As a result, for species of bacteria containing a high degree of variation, the distance of the reference from a sample directly affects the sensitivity of resulting calls [Pightling et al., 2014]. When comparing multiple samples by describing how each differs from a single reference, this problem is compounded as the reference sequence which is closest to one sample may not be closely related to another.

For long read sequence data, the challenges of mapping to a single reference sequence are exacerbated: wheras for short read data several reads are required to cover a gene, for long read data each read covers several genetic features. As a result, the structure and sequence similarity of a longer stretch of the genome needs to be conserved between the sample and reference for successful mapping, or else the subsequent analysis must be able to handle chimeric mappings.

Historically (for short read data) the problem has been simplified by considering only very closely related sets of bacterial samples, or by considering only the core part of the pangenome within a set of samples. However this makes it very difficult to study the many interesting features in the non-core genome as described in Background section 2.2.3. Table 3.1 outlines some recent studies in collections of bacteria and the methods used to investigate them whilst handling diversity between samples in the studied dataset.

Table 3.1: Workarounds in previous studies.

Study and Species	Problem	Solution
[Chewapreecha et al., 2014] *S. pneumoniae*	Match recombination hotspots in 7 clusters of samples, each analysed using different close reference genome	Line up figures and squint (Figure 3 in that paper, pers. comm. N Croucher).
[Croucher et al., 2013] *S. pneumoniae*	Comparison of 616 asymptomatically carried *S. pneumoniae* to investigate changing antigenic profile after vaccine introduction	Sequences were clustered based on core orthologous genes, and each of 15 clusters was further analysed with a cluster specific reference.
[Croucher et al., 2014] *S. pneumoniae*	Investigate the population structure and drivers of bacterial diversification including movement of phage and intragenomic rearrangements	Gene presence, plus multiple rounds of custom reference construction and remapping, for IS elements, loci of interest.
[Kallonen et al., 2017] *E. coli*	11 year survey of (diverse) invasive *E. coli*	Phylogenetic trees based on core SNPs with recombinant regions and phages masked out, AMR gene presence/absence and typing.
[Stoesser et al., 2014] *K. pneumoniae*	Investigation of community versus hospital associated transmission for endemic NDM-producing *K. pneumoniae*	The first infected neonatal case was PacBio sequenced and assembled. This assembly was used as a reference for the remaining outbreak samples.
[Gorrie et al., 2017] *K. pneumoniae*	Investigate whether colonisation with *K. pneumoniae* at time of admission was a risk factor for infection in the ICU	A maximum likelihood phylogenetic tree was constructed from an alignment of core SNPs, and this was used to identify lineages. Within lineages, assembly and read mapping allowed detailed pairwise SNP comparisons (calling with respect to a single reference not sufficient).

Graph reference genomes describing variation seen in a population have already been used with success to enable more effective mapping of short reads in complex regions of the human genome (MHC) [Iqbal et al., 2012], [Dilthey et al., 2015], in the wider human genome [Garrison et al., 2018] and in the malaria parasite *P. falciparum* [Maciuca et al., 2016]. However, these methods have not been designed for use with Nanopore sequence data, neither are they compatible with the degree of structural rearrangement that can occur within a bacterial species like *E. coli*.

In this section we describe a novel modular pangenome reference graph structure and describe the advantages of such a graph in enabling detection of genetic features of interest which can occur in different genetic contexts. In the next chapters we will describe methods to quasi-map to this pangenome reference graph and to use it as a substrate for variation analysis between diverse genomes.

3.2 Definitions

Let $s = (a_0, ..., a_{n-1})$ be a DNA sequence with length n on the alphabet $\Sigma = \{A, C, G, T\}$. We define $s_{i,j} = (a_i, a_{i+1}, ..., a_{j-1})$ to be the substring (or subsequence) of s induced by the interval $[i, j)$ and $s_i^k = s_{i,i+k}$ to be the k-mer or k-length substring starting at position i in s. For any letter $a \in \Sigma$ let $\bar{a}$ be the Watson-Crick reverse complement and define $\bar{s} = (\bar{a_n}, ..., \bar{a_1})$ to be the reverse complement sequence of s.

A *sequence graph* is a directed graph in which the vertices represent DNA sequences and the edges connect the end of one sequence to the start of another to represent adjacency in a genome. Traversing the graph yields permissible concatenations of sequences. We say such a graph is *acyclic* if it contains no cycles (every possible graph traversal includes each node at most once).

We define a *Pangenome Reference Graph (PanRG)* as an unordered collection of acyclic sequence graphs (termed *local graphs*), each of which represents a locus, such as a gene, mobile element, or intergenic region.

The disjointedness within the PanRG allows for detection of loci of interest within different genomic contexts. The requirement for the local graphs to be acyclic allows for sequence inference within a local graph using methods such as Hidden Markov Models.

In other contexts outside this thesis the terms *pangenome reference graph* and *population reference graph* are used interchangeably to describe a graphical representation of a number (greater than 1) of genomes of a species. There is an implicit

assumption in these contexts that the pangenome reference graph is a single connected graph. While this definition still holds for our pangenome reference graph, the implicit assumption does not.

3.3 Construction

We decided to construct the PanRG of a species so that each local graph represents a locus of interest, such as a gene, mobile element or intergenic region. In implementations for this thesis, PanRGs include local graphs for genes and intergenic regions, thereby encapsulating the vast majority of the genome generally, but excluding phage and other mobile elements.

There exist a number of pre-existing tools which are able to identify clusters of orthologous sequences from collections of genome assemblies, including `Roary` [Page et al., 2015], `PanX` [Ding et al., 2017] and for intergenic regions connecting orthologous sequences `Piggy` [Thorpe et al., 2018]. The creators of both `PanX` and `Piggy` have made available multiple sequence alignments (MSA) of these clusters for a number of species.

We construct each local graph from a multiple sequence alignment of known alleles at a locus. We use a recursive clustering algorithm as depicted in Figure 3.1 and described in Algorithm 1. The output is guaranteed to be a (directed) acyclic sequence graph including hierarchical nesting of genetic variation while meeting a 'balanced parenthesis' criterion. Each path through the graph from source to sink represents a possible sequence for the locus as a mosaic of input sequences.

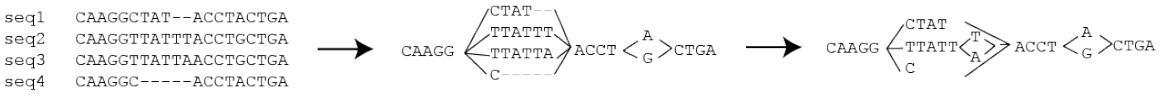


Figure 3.1: *Toy example of local graph construction from MSA*: We use $min\_match\_length = 4$ and show the local graph corresponding to the MSA on the left after 1 and 2 iterations.

3.3.1 Partitioning a Multiple Sequence Alignment

Let A be the set of aligned sequences from a multiple sequence alignment, each of length n. Let $s(a)$ be the DNA associated with aligned sequence $a \in A$, obtained by removing all '-' (or non-AGCT) symbols.

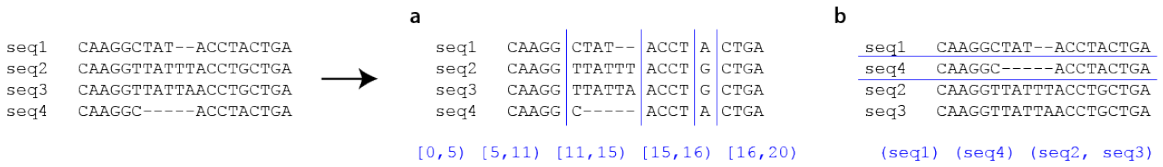


Figure 3.2: *Partitions of a Multiple Sequence Alignment.* In (a) we give an example of a vertical partition defined by a partition of the interval $[0, n)$ and in (b) a horizontal partition defined by a partition of the set of sequence ids.

We can partition alignment A in 2 ways as shown in Figure 3.2: by partitioning the interval $[0, n)$ or by partitioning the set of unique sequence identifiers in A. Each of these partitions yields a number of sub-alignments. We partition in both of these ways using an iterative process described in Algorithm 1 to construct a nested variation graph.

3.3.1.1 Partitioning into slices

Define $\text{slice}_A(i, j) = \{a_{i,j} : a \in A\}$. We will say that interval $[i, j)$ is a *match* interval if

$$j - i \geq min\_match\_length$$

and there is a single non-trivial sequence in the slice. i.e.

$$\|\{s(a) \ : \ a \in \text{slice}_A(i, j) \ \text{ and } \ s(a) \neq \text{‘’}\}\| = 1$$

and otherwise call it a *non-match* interval.

3.3.1.2 Partitioning with k-means clustering

K-means clustering is an algorithm which is widely used to partition data into a fixed number of clusters, K, based on similarity [MacQueen, 1967].

In the general case, if we have observations $\{\mathbf{x_1}, ..., \mathbf{x_n}\}$ where $\mathbf{x_i} \in \mathbf{R}^d$, then K-means clustering partitions these observations into K clusters $\mathcal{C} = \{C_1, ..., C_K\}$ so as to minimize the within-cluster sum of squares:

$$\arg\min_{\mathcal{C}} \sum_{j=1}^{K} \sum_{\mathbf{x} \in C_j} \|\mathbf{x} - \mu_j\|^2$$

where μ_j, the mean of cluster j, is defined to be

$$\mu_j = \frac{1}{\|C_j\|} \sum_{\mathbf{x} \in C_j} \mathbf{x}.$$

To apply this to sequences, we first transform them into vectors. For $a \in A$ we transform sequence $s(a)$ into vector $\mathbf{x_a} = \{x_a^1, ..., x_a^m\}$ where x_a^i is the count of a specific substring of length *min_match_length* in string $s(a)$.

We use python package `scikit-learn` (`https://scikit-learn.org/stable/modules/clustering.html`) to perform K-means clustering during graph construction.

ALGORITHM 1: Construct Local Graph from MSA

Input : A – a multiple sequence alignment
Output: G – a sequence graph

$\mathcal{A} = \{\mathsf{A}\}$
while *iteration* $<$ *max_iterations* **do**
  $\mathcal{A}' = \emptyset$
  foreach $A' \in \mathcal{A}$ **do**
    $\mathcal{B} = (\mathcal{B}^+, \mathcal{B}^-) = \texttt{partition}_v(\mathsf{A'})$ /* Partition into match (+) */ /* and non-match (-) intervals */
    foreach $B \in \mathcal{B}^+$ **do**
      Replace match interval B with sequence node
    foreach $B \in \mathcal{B}^-$ **do**
      $\mathcal{A}' = \mathcal{A}' \cup \texttt{partition}_h(\mathsf{B})$ /* Partition non-match interval B */ /* with K-means clustering */
  $\mathcal{A} = \mathcal{A}'$
Replace each sub-alignment $\mathsf{A'} \in \mathcal{A}$ with a node for each sequence in $\{s(a) : a \in \mathsf{A'}\}$

The resulting sequence graph has been constrained to contain nested *bubbles*. As described above, match intervals are defined with a minimum match length, by default 7. Having such a restriction makes biological sense as for longer variants introduced by recombination we expect some shared sequence in the flanking regions. It also provides a control on the complexity of the graph by reducing the number of times we define the start of a variant region. The result is that variation which lies close together in a sequence is clustered.

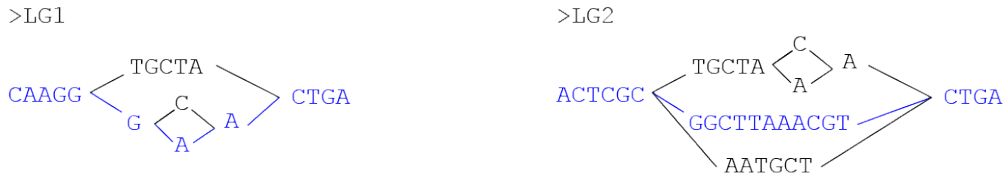
Note that alternative alignments for the same region may well result in slightly different local graphs, but this should not affect downstream analyses. There does not exist a canonical definition of what constitutes an optimal sequence graph, and we do not attempt to claim ours is optimal.

3.4 Describing variation with respect to a graph

One of the most common formats for representing the differences between a number of genomes is the VCF file format, made popular by the 1000 Genomes Project [Danecek et al., 2011], [Altshuler et al., 2012]. An example can be found in Figure 2.6 in Section 2.4.3. In this format, sequence variation is stored in *records*, along with information about the type of sequence variant. A column for each sample provides information about the allele seen in that sample, often along with details about the support from the data for each allele.

In designing methods for this work, we were keen to enable downstream analyses to be compatible with existing tools, and so decided that the primary output should be in the VCF format.

Each local graph is expected to have a unique identifier. A single path through the local graph is chosen as the reference path and all variation is described relative to this path as demonstrated in Figure 3.3. We include multiple records to describe the nested levels of variation. In this way we can represent variation with respect to a reference graph, rather than a single reference sequence.



#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT
LG1	6	.	GAA	TGCTA	.	.	.	.
LG1	7	.	A	C	.	.	.	.
LG2	7	.	GGCTTAAACGT	AATGCT,TGCTACA,TGCTAAA	.	.	.	.

Figure 3.3: *VCF describing graph variation for 2 local graphs.* For two example local graphs we describe variation relative to the path in blue, the 'reference' path. The local graph identifier is used in the CHROM field and multiple records are added to describe nested variation.

We provide options to allow variation to be output with respect to user input reference sequences (if they exist in the graph), and otherwise choose a suitable path from the graph (see sections 5.2.2 and 6.2.2).

3.5 *E. coli*

3.5.1 *E. coli* epitomizes the difficulties of comparison

The *E. coli* species is widely present both in the environment and in the human gastrointestinal tract, typically colonising it within hours of birth. It also has many disease pathotypes, causing enteric diseases such as diarrhoea and dysentery, and extra-intestinal infections such as urinary tract infections and meningitis [Kaper et al., 2004]. In addition the increasing prevalence of antimicrobial resistance in *E. coli* and related species is considered a public health threat [WHO, 2014].

The *E. coli* pangenome was first observed by [Welch et al., 2002] when they compared 3 complete genomes and discovered that only 39.2% of the total set of predicted proteins were present in all three strains. As more complete genomes have been sequenced and compared, pangenomes have been observed for many other species [McInerney et al., 2017]. As larger numbers of genomes have been compared, it has become apparent that some pangenomes are *open* in that every new sequenced genome reveals new genes not seen in the previous genomes [Collins and Higgs, 2012]. *E. coli* has one of the larger pangenomes, estimated in a study of 2000 genomes by [Land et al., 2015] to contain 3100 core gene families and 89,000 different gene families in total. There is known to be sharing of genes between species of the *Enterobacteriaceae* family [Sheppard et al., 2016].

As a result, it is both one of of the most relevant and widely studied and sequenced species of bacteria, and yet one of the most problematic for the types of comparison we have described above. By demonstrating the applicability of `Pandora` to *E. coli*, we can conclude that it will continue to work in less problematic species.

3.5.2 Selection of sequences

The problem of identifying orthologous genes or clustering intergenic sequences is challenging. A number of tools have been developed for this purpose as we outline below.

Gene curation

There are a number of tools for clustering orthologous genes, designed to handle different degrees of divergence between input reference genomes. `OrthoMCL` [Li et al., 2003] and `OrthoFinder` [Emms and Kelly, 2015] are well suited to identifying orthologues across species. In contrast, `Roary` [Page et al., 2015] was designed to cluster

very large numbers of similar genomes, and can handle different degrees of diversity by modification of an identity cutoff parameter. A more recent tool `PanX` [Ding et al., 2017] was designed as part of a project to allow interactive exploration of species pangenomes for any species. It uses a traditional initial approach, identifying groups of homologous genes by similarity search of protein sequences using `DIAMOND` [Buchfink et al., 2014] and clustering by `MCL` [Enright et al., 2002]. Unusually, it then uses phylogenetic post-processing to define the final orthologous groups based on the tree structure, and so scales identity cutoffs relative to the core genome diversity. It appears to perform as well as `Roary` in scenarios where `Roary` is well suited, but is also applicable to scenarios where there is more diversity.

Multiple sequence alignments for genes curated with `PanX` were made available by [Ding et al., 2017] at `http://pangenome.de/`. These were downloaded on 03-05-2018 and were curated from approximately 300 RefSeq assemblies.

3.5.2.1 Intergenic region curation

Intergenic regions form 10-15% of a bacterial genome and have been shown to be subject to purifying selection in the core genome even after excluding known regulatory elements [Thorpe et al., 2018], [Molina and Van Nimwegen, 2008].

`Piggy` [Thorpe et al., 2018] is the first and only tool designed to identify clusters of intergenic regions. It emulates `Roary`, but applies to the regions identified as intergenic by `Roary`.

Multiple sequence alignments for intergenic region clusters based on 228 *E. coli* ST131 genome sequences were generated with Piggy by [Thorpe et al., 2018] for this publication, and made available on request. Whilst this panel of intergenic sequences does not reflect the full diversity within *E. coli*, we were keen to include them as an initial starting point.

3.5.3 Construction of PanRG

We used `Nextflow` [Di Tommaso et al., 2017] to construct local graphs in parallel using a python script and the full pipeline used is available at `https://github.com/rmcolq/make_prg`. We used up to 10 iterations, resulting in up to 10 degrees of nesting, and a minimum match length of 7.

We included some basic handling of common human input errors, including replacing sequences containing non-AGCT IUPAC letters with copies containing each of the corresponding bases represented by the letter. Despite this, a small fraction

of multiple sequence alignments failed to build into graphs, mainly due unexpected symbols which hadn't been accounted for and could not be handled. The resulting PanRG contains local graphs corresponding to 23052/23107 genes and 14374/14400 intergenic regions downloaded.

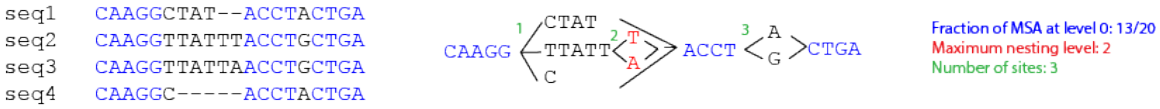
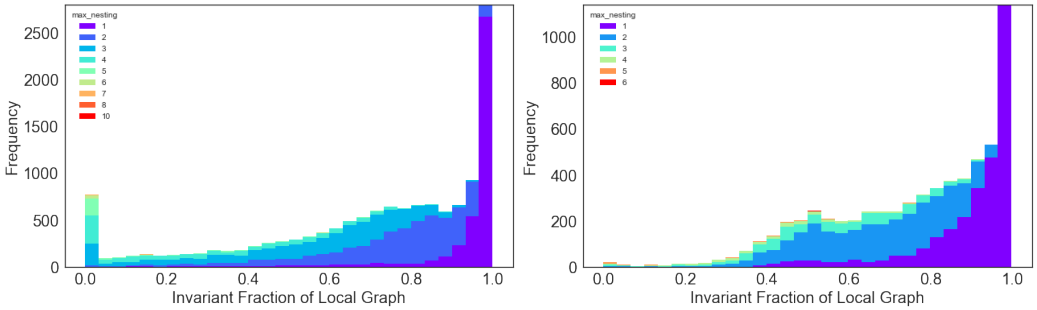


Figure 3.4: *Three metrics to describe the complexity of a Local Graph.* In blue we describe the fraction of the MSA which was 'invariant' or common to all sequences in the MSA and captured at level 0 in the resulting graph. In red we describe the maximum nested level reached in the graph. In green we describe the number of variant sites in the graph.

Of the local graphs in the PanRG, 58.9% of gene graphs and 43.8% of intergenic graphs consisted of a single sequence with no added variation. We know that most genes are rare (see gene frequency spectrum Figure 2.2) and therefore we expected more than half of gene clusters to correspond to genes for which there was only a single exemplar in our reference dataset.

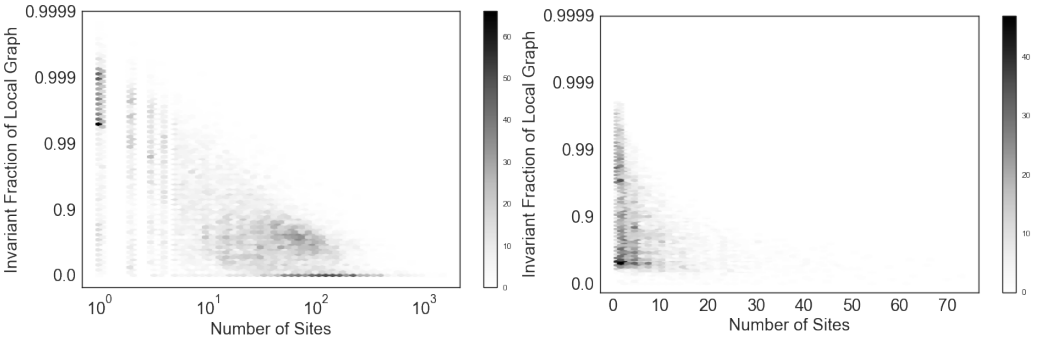
Figure 3.4 defines the 3 metrics we use to describe the resulting graphs. Figure 3.5 shows that the graphs generated for intergenic regions had fewer nested levels and more sequence at level 0 in the graph (common to all input sequences in slices of the MSA) than genes. The most levels used by intergenic graphs was 6, compared to 10 for genes. There is a peak close to 0 in Figure 3.5a, showing that a significant proportion of the gene MSA input had almost no sequence that was shared by all sequences in a slice. This could be due to a single insertion event in one allele or could arise if many diverged versions of these genes were included. Despite this, for the majority of such cases 3-5 nested levels were sufficient to describe the variation seen. There is also a slight hump at 0.5 in the distribution for intergenic regions. This could be explained by the minimum percentage nucleotide identity threshold of 50% used by Piggy during clustering.

Figure 3.6 shows that local graphs with a high fraction of sequence at level 0 also had fewer sites. Most intergenic local graphs had few sites (0-10), even when they had a low fraction of sequence at level 0. This could indicate clusters of distinct sequences which appear in the same genetic context. Many gene local graphs had either very few sites (< 10), or many sites, numbering in the 100s, but there was a spread of number of sites as the fraction of sequence at level 0 varied.



(a) (b)

Figure 3.5: Stacked histogram showing the number of levels of nesting used in Local Graphs corresponding to (a) genes and (b) intergenic regions as the fraction of sequence at level 0 in the graph (common to all input sequences) varies.



(a) (b)

Figure 3.6: Number of sites in the local graphs corresponding (a) genes and (b) intergenic regions as the fraction of sequence at level 0 in the graph (common to all input sequences) varies.

Chapter 4

Indexing and Quasi-mapping to the PanRG with Pandora

4.1 Introduction

The process of mapping reads to a single reference genome performs poorly in regions where there is high sequence or structural diversity between the reference and sample [Dilthey et al., 2015]. Reference graph methods have been successful in resolving this issue when applied to short read sequence data from human genomes [Garrison et al., 2018] and pathogens with limited structural diversity [Maciuca et al., 2016]. However, so far no method has been developed which can handle long noisy sequence data or the degree of structural diversity seen in the *E. coli* pangenome.

Full sequence alignments of long noisy reads are computationally expensive. By comparison, approximate mapping approaches, such as `Minimap2` [Li, 2018] and `MashMap` [Jain et al., 2018] are able to rapidly find approximate mapping locations for such reads against a panel of reference sequences. These tools *quasi-map* query sequences against the reference sequences and provide information about the strand and position from which the query may have likely originated. They do not provide base-by-base information about how the query aligns against the reference at a position (although `Minimap2` can subsequently align).

In the previous chapter, we described a novel pangenome reference graph (PanRG) structure capable of storing reference sequences from diverse panels of genomes. In this chapter we describe methods to index the PanRG and quasi-map both long and short reads to it in order to infer the presence or absence of loci (such as genes or intergenic regions) within the reads. We will demonstrate in subsequent chapters that we can work directly with these approximate mappings to infer sequences and call variation between samples.

4.2 The Pandora software implementation

Methods for interacting with the novel PanRG structure defined in Chapter 3 have been developed in the software package `Pandora` [`https://github.com/rmcolq/pandora`]. These include:

- `pandora index` – indexes the PanRG with (w, k)-minimizers
- `pandora map` – quasi-maps reads from a single sample to the index and infers the presence or absence of PanRG loci in the sample, infers the sequence at loci as a mosaic of reference sequences and outputs a set of variant calls for the sample with respect to the graph and optionally identifies *de novo* variants not present in the graph.
- `pandora compare` – quasi-maps reads from several samples to the index and infers the presence and sequence of PanRG loci in each sample, then calls variants between the samples with respect to the graph.
- `pandora random_path` – outputs sequences from a specified number of random paths through each local graph in the PanRG

In this chapter we will focus on indexing and quasi-mapping to infer the presence or absence of loci in a sample. This forms the first step in both the `map` and `compare` methods. In Chapter 5 we look at inferring sequences as a mosaic of references and graph-based genotyping using `pandora map` and finally in Chapter 6 we look at how to describe variation between samples using `pandora compare`.

`Pandora` is implemented in C++ and includes approximately 12,000 lines each of code and tests. The vast majority of code is my own work, with the exception of the *de novo* discovery section, which is predominantly the work of Michael Hall (another PhD student in our group). Robyn Ffrancon identified and fixed a number of memory leaks in tests and updated the CMake files to allow inclusion of external libraries required by *de novo* discovery. In addition, paired programming with Robyn significantly improved the readability of code and use of modern C++ techniques. Subsequent to this thesis, major performance improvements have been made to this code base by Leandro Ishi Soares De Lima.

4.3 Algorithms

4.3.1 Using (w,k)-minimizers to index a graph

Indexing with k-mers

When indexing a panel of reference sequences it is common to index reference sequences with fixed length k-mers, which are stored in a hash table. Read k-mers are looked up in this hash table and candidate mapping regions within the reference sequences are identified.

When the reference panel contains a large number of distinct k-mers (such as when it contains multiple references from a diverse species), it is not feasible to index every k-mer, and instead different techniques are used to select a sparser set of representative k-mers. This has been done in a variety of ways

`BLAST` [Altschul et al., 1997] and `BLAT` [Kent, 2002] use one k-mer hash function, and hash k-mers at fixed-width positions 1, $w + 1$, $2w + 1$, ... in the reference sequences. The size of this index grows linearly with the total length of reference sequences but the hashed values of all read k-mers must be queried against this index to avoid offset errors.

`MASH` [Ondov et al., 2016], also uses 1 k-mer hash function, but stores only the n smallest hashed canonical k-mers for each reference or query sequence. Similarly, the MinHash Alignment Process (MHAP) [Berlin et al., 2015] used by `Canu` [Koren et al., 2017] defines a fixed number of k-mer hash functions $\{\phi_j\}_{1\leq j\leq m}$ and stores a minimum hashed k-mer for each, $\{h_j\}_{1\leq j\leq m}$ where

$$h_j = \min\{\phi_j(s_i^k) : 1 \leq i \leq \|s\| - k + 1\}.$$

However, using a fixed size index can negatively affect sensitivity or can waste space depending on the length of input reference sequences.

Both `Minimap2` [Li, 2018] and `MashMap` [Jain et al., 2018] make use of a winnowing technique first introduced as a way of allowing similarity searches between web pages. For a single fixed hash function, these methods sketch a string by including the smallest hashed item in each consecutive fixed size window. The resulting index scales in size with the total length of reference sequences. Using this method, the query sequence is also sketched so there is a much smaller set of query k-mers than for `BLAST`. We will extend this technique to allow indexing of a graph.

Minimizers of strings

We define (w,k)-*minimizers* of strings as in [Li, 2016]. Let $\phi : \Sigma^k \to \mathbb{R}$ be a k-mer hash function and let $\pi : \Sigma^* \times \{0,1\} \to \Sigma^*$ be defined such that $\pi(s,0) = s$ and $\pi(s,1) = \bar{s}$.

Consider any integers $w, k > 0$. For window start position $0 \leq j \leq |s| - w - k + 1$, let

$$T_j = \{\pi(s_{p,p+k}, r') : j \leq p < j + w \text{ and } r' \in \{0,1\}\}$$

be the set of forward and reverse-complement k-mers of s in this window. We define (w,k)-minimizer(s) to be any triple (h,p,r) such that

$$h = \phi(\pi(s_{p,p+k}, r)) = \min\{\phi(t) : t \in T_j\}.$$

The set $W(s)$ of (w,k)-minimizers for s, is the union of minimizers over all such windows.

$$W(s) = \bigcup_{0 \leq j \leq |s|-w-k+1} \{(h,p,r) : h = \min\{\phi(t) : t \in T_j\}\}$$

Minimizers of a sequence graph

We extend this definition intuitively to an acyclic sequence graph $G = (V, E)$.

Let $\mathbf{i} = (v, a, b)$ represent the sequence interval $[a, b)$ on node $v \in V$ and define $|v|$ to be the length of the sequence associated with v.

We define a *path* in G by

$$\mathbf{p} = \{(\mathbf{i}_1, ..., \mathbf{i}_m) : (v_j, v_{j+1}) \in E \text{ and } b_j == |v_j| \text{ for } 1 \leq j < m\}.$$

This matches the intuitive definition for a path in a sequence graph except that we allow for inclusion of only part of the sequence associated with the first and last nodes on the path. We will use $s_{\mathbf{p}}$ to refer to the sequence along path $\mathbf{p}$ in the graph.

Let $\mathbf{p}$ be a path of length $w + k - 1$ in G. The string $s_{\mathbf{p}}$ contains w consecutive k-mers for which we can find the (w,k)-minimizer(s) as before. We therefore define the (w,k)-minimizer(s) of the graph G to be the union of minimizers over all paths of length $w + k - 1$ in G.

$$W(G) = \bigcup_{\mathbf{p} \in G\,:\, \|\mathbf{p}\| = w+k-1} \{(h, \mathbf{p}', r) : h = \min\{\phi(t) : t \in T_{\mathbf{p}}\}\}$$

Sketching a Local Graph

To find these minimizers for a graph, we use a streaming algorithm as described in Algorithm 2. For each minimizer found, it simply finds the next minimizer(s) until the end of the graph has been reached. The method is illustrated in Figure 4.1 for a toy example.

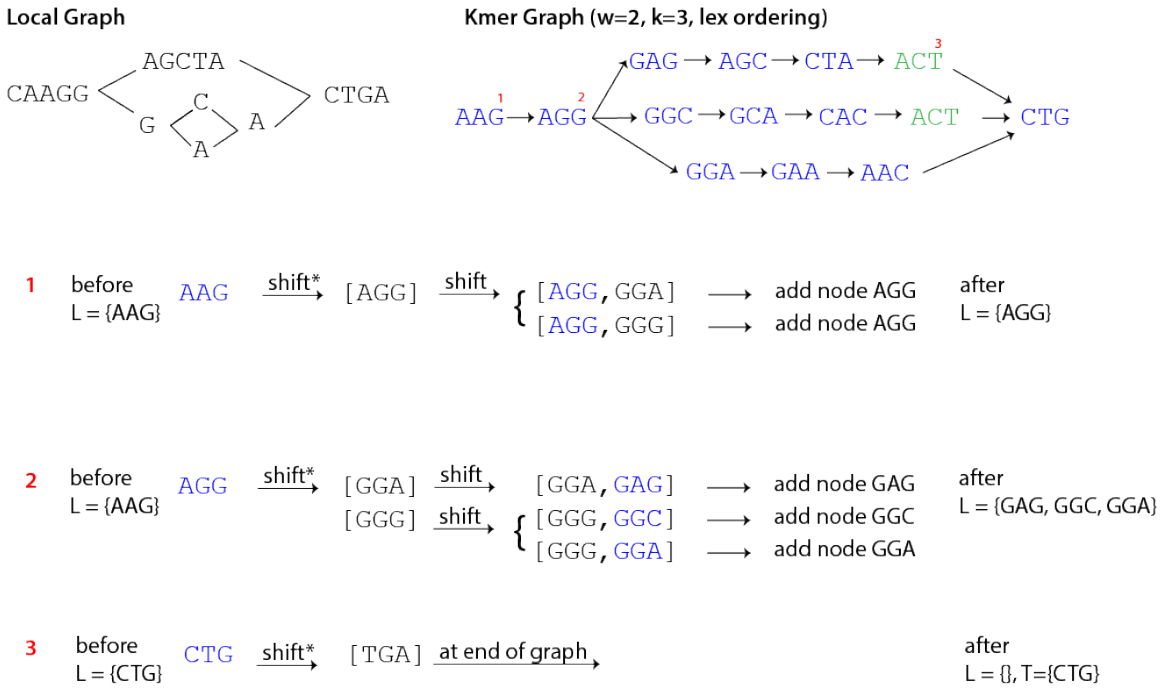


Figure 4.1: *Sketching a graph with (w, k)-minimizers.* For a toy local graph we show what the k-mer graph of minimizers would look like. All minimizers are coloured. We use `shift*` to denote the function which finds the next k-mer along the graph. We detail the process of finding the next minimizer(s) at 3 stages of the k-mer graph construction process. Note that `ACT`, highlighted in green, appears twice in the k-mer graph but these nodes are not merged as they correspond to different paths in the local graph.

Borrowing from Graph Theory, for any graph (V, E) we define the in-degree of a vertex $v \in V$ to be

$$deg^-(v) = \|\{(u, v) \in E : u \in V\}\|$$

and the out-degree to be

$$deg^+(v) = \|\{(v, w) \in E : w \in V\}\|.$$

We define the *leaves* of a graph to be

$$\{v \in V : deg^+(v) = 0\}.$$

We define K to be the *k-mer graph* with nodes corresponding to minimizers $(h, \mathbf{p}, r)$. We add an edge (u, v) to K if there exists a path through G on for which u and v are both minimizers and v is the first minimizer after u along the path. This sketching algorithm constructs K, traversing the local graph from left to right. At each point, it considers the leaves of current iteration of K, and for each leaf it either adds new vertices and edges to the next minimizer(s) (so that it is no longer a leaf), or labels it as a *tip* of the final tree which cannot be extended.

We define walk(v, i, w, k) be a function which returns all vectors of w consecutive k-mers in G starting at position i on node v. Suppose we have a vector of k-mers x. Let shift(x) be the function which returns all possible vectors of k-mers which extend x by one k-mer in the graph. It does this by considering possible ways walk one letter in G from the end of the final k-mer of x. For a vector of k-mers of length w, the function minimize(x) returns the minimizing k-mers of x.

Let $K \leftarrow \text{add}(s, t)$ denote the addition of nodes s and t and the directed edge (s, t) to K if they do not already exist. Let $K \leftarrow \text{add}(s, T)$ denote the addition of nodes s and $t \in T$ to K as well as directed edges $(s, t) : t \in T$, and define $K \leftarrow \text{add}(S, t)$ similarly.

Indexing the PanRG

We index the PanRG with (w, k)-minimizers, storing a map from each minimizing k-mer hash value to the positions in all local graphs where that minimizer occurred as described in Algorithm 3.

4.3.2 Quasi-mapping to the Index

Quasi-mapping is the process of identifying approximate mapping locations between each read and the PanRG. To do this, each read is sketched with (w, k)-minimizers and these are looked up in the index as described in Algorithm 4. For every (w, k)-minimizer shared between a position in the read and a local graph in the PanRG index, we define a *hit* as the coordinates of the minimizer in the read and local graph.

To handle noise which arises both as a result of frequent sequencing errors in Nanopore reads, and due to non-unique k-mers in the index, we filter these hits. Firstly, we define clusters of hits between a single read and local graph, where enough hits are located close together on a read and retain only hits contained in such a cluster. These clusters are then filtered further so that only one local graph can be identified as present in a single region of a read : if the first and last hit of a cluster are contained on the read within a larger cluster, we remove it.

ALGORITHM 2: Find the minimizing k-mers for a graph

Input : G – a local graph
Output: K – a k-mer-graph of minimizing k-mers

```
1  Function MinimizerSketch(G)
2      L ← ∅                                              /* set of current leaves in K */
3      T ← ∅                                              /* set of tips of K */
4      X ← walk(0, 0, w, k)  /* all vectors of w consecutive k-mers starting at node 0 pos 0 */
5      for x ∈ X do
6          S ← minimize(x)                  /* find the minimizing k-mers for this vector */
7          K ← add(∅, S)                        /* join null start node to first k-mers */
8          L ← L ∪ S
9      while L ≠ ∅ do
10         m := (h, p, r) ← L.extract()  /* remove a minimizing k-mer from current leaves */
11         if m == G.end() then
12             T ← T ∪ {m}
13             continue
14         X ← {x \ m for x ∈ shift(m)}
15         while X ≠ ∅ do
16             x ← X.extract_last()
17             y ← x.get_last()
18             if min{φ(π(s_y), 0), φ(π(s_y), 1)} ≤ min{φ(π(s_p), 0), φ(π(s_p), 1)} then
19                 K ← add(m, y)
20                 L ← L ∪ {y}
21             else if |x| == w then
22                 S ← minimize(x)              /* find the minimizing k-mers for this vector */
23                 K ← add(m, S)
24                 L ← L ∪ S
25             else if y == G.end() then
26                 T ← T ∪ {m}
27             else
28                 X ← X ∪ {shift(x)}
29     K ← add(T, ∅)                                  /* join tips to a null end node in K */
30     return K
```

ALGORITHM 3: Index the PanRG

Input : $\mathcal{G}$ – a set of Local Graphs
Output: $\mathcal{I}$ – an Index

Function `Index`($\mathcal{G}$)
  $\mathcal{I} \leftarrow \emptyset$
  for $\mathsf{G}_i \in \mathcal{G}$ **do**
    $\mathsf{K}_i \leftarrow$ `MinimizerSketch`(G_i)
    for $(h, \mathbf{p}, r) \in V(\mathsf{K}_i)$ **do**
      $\mathcal{I}[h] \leftarrow \mathcal{I}[h] \cup \{(i, \mathbf{p}, r)\}$
  return $\mathcal{I}$

A future extension to remove noise could enforce an additional co linearity criterion requiring hits in a cluster to appear in the same order on a read as on a (single) path through the local graph.

Setting the minimum cluster size

When the minimum size of a cluster is set too low, we have more false positive local graphs identified as present in the dataset, and also have to handle more noise downstream when inferring a mosaic sequence and genotyping. When it is set too high, we have less sensitivity to discover loci that are present.

To handle reads and local graphs of very different lengths, the threshold for number of hits in a cluster is defined relative to both. For short reads, the length of the read may be shorter than the length of the local graph sequence. The number of (w, k)-minimizers in a sequence of length $\|s\|$ is approximately $2\|s\|/w$ [Schleimer et al., 2003], and for perfect sequencing covering the entire local graph, we would therefore expect a cluster to contain roughly this number of hits.

When reads are longer, the best we can say about the number of hits between a perfect read and the local graph is that it must be at least the minimum number of (w, k)-minimizers along a path in the local graph.

To account for sequencing errors, we estimate that if errors occur independently as a Poisson distribution with rate ϵ along a sequence, the probability that a k-mer has been sequenced with no errors is $\exp(-k\epsilon)$. The expected size of a cluster of (w, k)-minimizer hits between a read of length $\|s\|$ and local graph with minimum k-mer path length l is at least:

$$t \geq \min(2\|s\|/w, l) * \exp(-k\epsilon).$$

ALGORITHM 4: Quasi-map

Input : R – a read
$\mathcal{I}$ – the PanRG Index
m – the maximum gap length (in bases) between k-mers in a cluster on a read
t – the minimum number of hits in a cluster

Output: $\mathcal{C}$ – Clusters of hits corresponding to mapped regions

```
1  Function QuasiMap(R, I)
2      M ← MinimizerSketch(R)
3      A ← ∅
4      for (h, p, r) ∈ M do
5          for (j, q, r') ∈ I[h] do
6              if r ≡ r' then
7                  Append (j, 0, p, q) to A
8              else
9                  Append (j, 1, p, q) to A
10     Sort A = {(j, r, p, q)} by the 4 values of tuple
11     C ← ∅
12     Cluster ← ∅
13     a_prev ← None
14     for a_curr = (j, r, p, q) ∈ A do
15         if j_curr ≠ j_prev  or  r_curr ≠ r_prev  or  ||p_curr − p_prev|| > m then
16             if Cluster.size() > t then
17                 C.append(Cluster)
18             Cluster ← ∅
19         Cluster.append(a_curr)
20         a_prev = a_curr
21     return C
```

To allow for partial matches on reads, we set the cluster size threshold to be half this lower bound on the expected value. This is equivalent to a threshold of 10 hits for a gene length of 900bps and either an Illumina read length of 150bps with errors at rate 0.001, or a long Nanopore read with errors at rate 0.11.

Filtering false positives

Once we have identified and filtered clusters of hits, each cluster corresponds to a local graph which we have identified as present in our dataset. In the next section we describe how a mosaic sequence is inferred using the k-mer counts. As a sanity check, we further filter out local graphs when the global coverage is greater than 20X and the mosaic sequence inferred has a mean k-mer coverage either less than 1/20th or more than 10 times the global coverage, or where both the mean and median k-mer coverage are less than 3.

4.3.3 A pangenome (multi)sample graph

The PanRG does not store information about the order of loci within it, and is therefore agnostic to gene rearrangements. Instead, such information is inferred directly from reads. While quasi-mapping reads, we construct a *pangenome (multi)sample graph* as depicted in Figure 4.2. In this graph, each node corresponds to a locus in the PanRG which has been discovered, and edges connect loci in the order they were discovered on reads. Within `Pandora`, discovered clusters of hits are stored at the relevant node in this pangenome sample graph. Downstream analysis can then operate independently on each of these nodes.

We describe in Section 4.8 how information from this pangenome sample graph could be used to infer the order of identified loci, and enable deconvolution of mixtures.

4.4 Methods

We first demonstrate how choices of parameter affect precision and recall when identifying the presence or absence of loci in the PanRG using simulations. We then apply this method to identify the presence of PanRG loci in real reads from a lab strain of *E. coli* K12.

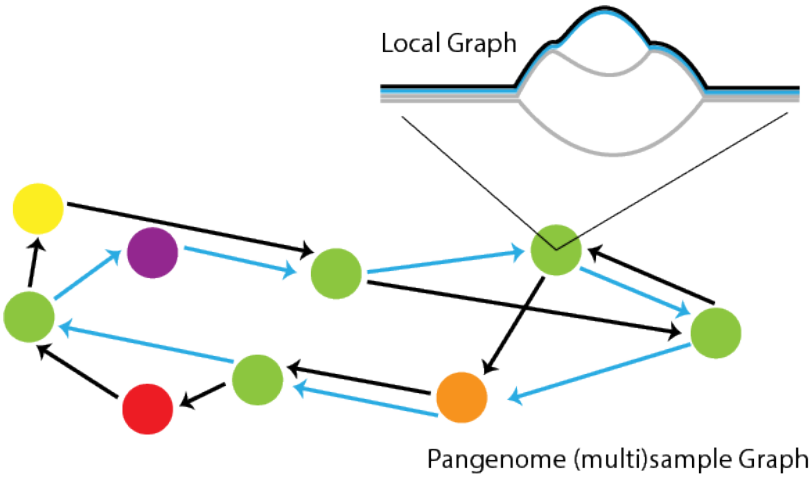


Figure 4.2: *Schematic describing pangenome (multi)sample graph for two samples.* Two samples are depicted in blue and black. A coloured node is depicted for each locus found in a sample (green for core genes) and arrows represent the order in which nodes were found in reads for each sample. Downstream analyses comparing sequences found in each sample are performed independently on each common node.

4.4.1 Simulating reads with `ART` and `NanoSim-H`

A number of good read simulators exist for both long and short read sequence data and these vary in the complexity of the models they use. A summary of the available short read simulators can be found in [Olson et al., 2015]. To evaluate our methods using simulated data it was important that simulated reads captured some of the characteristics of true sequence data, e.g. error rates may not be uniform over a read, and may be higher at the ends, near homopolymers or in GC-rich regions.

To simulate Illumina reads, we therefore use `ART` [Huang et al., 2012] and for Nanopore reads we use `NanoSim-H` [Yang et al., 2017] [Karel Brinda, `http://doi.org/10.5281/zenodo.1341249`]. Both of these contain in-built technology-specific read error models and base quality value profiles parameterized empirically from sequencing datasets.

We ran these simulators in singularity images to generate approximately 300X coverage using the commands:

```
nanosim-h -p <profile> -n 150000 <sim_genome_fa> --unalign-rate 0
--max-len <max_len> --circular
art_illumina -ss <profile> -i <sim_genome_fa> -l <read_len> -f 300
-o simulated_illumina
```

For Illumina, we used profiles `MSv3` and `HS25` which represent the MiSeq v3 and the HiSeq 2500 technologies. For Nanopore reads, we use the `ecoli_R9_2D` and

`ecoli_R9_1D` profiles which represent 1D and 2D read protocols sequenced using the R9 flow cell. These are the most up to date profiles available in NanoSim-H and represent lower and medium accuracy Nanopore reads.

Current flow cells (R9.4.1 and R9.5.1, with R10 available to early access program) only support 1D reads, but are reported to have higher accuracy than older 1D reads due to the combination of updated pore and basecalling algorithms ([Rang et al., 2018] and Figure 4.3). The majority of our sequence data was generated using the R9.4 1D protocol, and was re-basecalled with the improved basecaller Albacore 2.1.7. We expect it to have a read accuracy between those represented by the two profiles. We note that estimates for the read accuracy of each technology vary greatly: [Jain et al., 2017] report a median read identity of 89% for 1D and 94% for 2D experiments using R9 flow cells, Nick Loman reports a median read identity of 83% with a rapid 1D kit and R9 flow cell [`http://lab.loman.net/2016/07/30/nanopore-r9-data-release/`], [Wick et al., 2019] reports a median read identity of 88% using R9.4 1D and Albacore 2.0.1. Despite this, error rates have and continue to improve (e.g. [`https://nanoporetech.com/about-us/news/1d-squared-kit-available-store-boost-accuracy-simple-prep`]). It seems reasonable therefore to tune `Pandora` for medium accuracy Nanopore data.

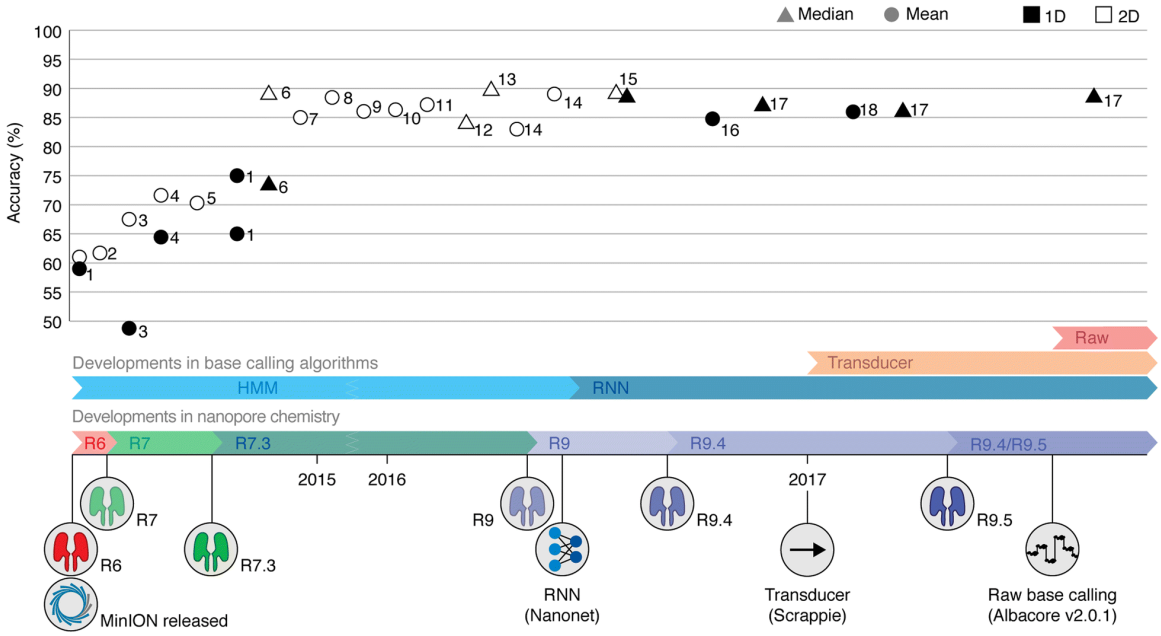


Figure 4.3: *Timeline of MinION read accuracies and Oxford Nanopore Technologies technological developments.* Source:[Rang et al., 2018]. Reprinted from Rang *et al.*, 'From squiggle to basepair: Computational approaches for improving nanopore sequencing read accuracy', Genome Biology (2018) under CC BY.

4.4.2 Simulating a 'genome'

We simulated 'genomes' by selecting local graphs corresponding to 5000 genes and 5000 intergenic regions from the *E. coli* PanRG described in Section 3.5. We chose a single sequence from each of these local graphs by randomly walking through the graph using the command:

```
pandora random_path <PanRG> 1
```

We concatenated sequences, selecting without replacement and alternating genes and intergenic sequences, to form our 'genome'. By construction, we therefore know the genes and intergenic sequences which are present in this 'genome' and can use this to classify loci as true positives, false positives or false negatives.

4.4.3 Effect of technology and read coverage on precision and recall of loci

We used the in-built profiles of several different sequencing technologies (Illumina HiSeq 2500, Illumina MiSeq v3 , Nanopore R9 1D and Nanopore R9 2D) to simulate reads with different length distributions and error profiles. Simulated reads were down sampled to a range of depths, and `Pandora` was used to identify the presence/absence of loci from the PanRG in each simulated dataset.

```
pandora map -p <PanRG> -r <simulated_nano_reads>  --genotype
--max_covg <max_covg>
pandora map -p <PanRG> -r <simulated_illumina_reads>  --genotype --illumina
--max_covg <max_covg>
```

We evaluated the number of local graphs correctly and incorrectly identified as present or absent in the reads regardless of sequence identity.

4.4.4 Effect of w and k on precision and recall of loci

For different values of w and k we indexed the PanRG using `Pandora` and quasi-mapped 50X of simulated Illumina or Nanopore reads to this index.

```
pandora index -w <w> -k <k> <PanRG>
pandora map -p <PanRG> -r <simulated_nano_reads> --genotype
--max_covg 50 -w <w> -k <k>
pandora map -p <PanRG> -r <simulated_illumina_reads> --genotype  --illumina
--max_covg 50 -w <w> -k <k>
```

Time and memory requirements were measured using the in-built Unix `time` module. We also evaluated the number of local graphs correctly and incorrectly identified as present or absent in the simulated reads.

4.4.5 Effect of cluster parameters on precision and recall of loci

We repeated the above analysis, fixing the default parameters of w and k, and varying the minimum number of hits in a cluster, and the maximum distance on a read of one hit from another within a cluster.

```
pandora map -p <PanRG> -r <simulated_nano_reads> --genotype
--max_covg 50 --min_cluster_size <min_size> --max_diff <max_diff>
pandora map -p <PanRG> -r <simulated_illumina_reads> --genotype --illumina
--max_covg 50  --min_cluster_size <min_size> --max_diff <max_diff>
```

4.4.6 Precision of genes and intergenic regions from real *E. coli* sequence data

We used a reference strain of *E. coli* K12 originally isolated in 1922, for which publicly available Nanopore and Illumina sequence datasets and a reference assembly are available [Goodwin et al., 2015], [`http://lab.loman.net/2017/03/09/ultrareads-for-nanopore/`]. Each of these was generated from a separate culture and extraction of this standard lab isolate, so we expect some small differences between the Illumina and Nanopore sequences and the truth assembly as a result of evolution [Kuhnert et al., 1995]. However, this data has been used extensively by developers in the Nanopore community as a means to evaluate methods for Nanopore and hybrid assembly.

Reads were quasi-mapped to the PanRG with `Pandora` using default parameters. The inferred mosiac sequences were mapped back to the reference assembly using `bwa mem`. Sequences for which there was no match with a 0 or 16 flag in the SAM file were deemed to be *false positives* and all others *true positives*. In this way we estimated the precision of presence or absence calls.

4.5 Results

4.5.1 The effect of technology and read coverage on precision and recall of loci

We simulated reads with the length distribution and error profiles of Illumina HiSeq 2500 with 150bp reads, Illumina MiSeq v3 with 150bp and 250bp reads and Nanopore R9 1D and 2D with maximum read lengths of 10,000bps and 50,000bps. Figure 4.4 shows the precision and recall when finding genes and intergenic regions from a simulated genome with `Pandora` (default parameters) at different depths of coverage. Nanopore 2D reads achieved the highest combination of precision and recall for gene presence, with Illumina technologies resulting in lower precision and Nanopore 1D resulting in lower recall. This is perhaps unsurprising, since all three were run with the same parameters, tuned on medium accuracy (2D) simulated Nanopore sequence data.

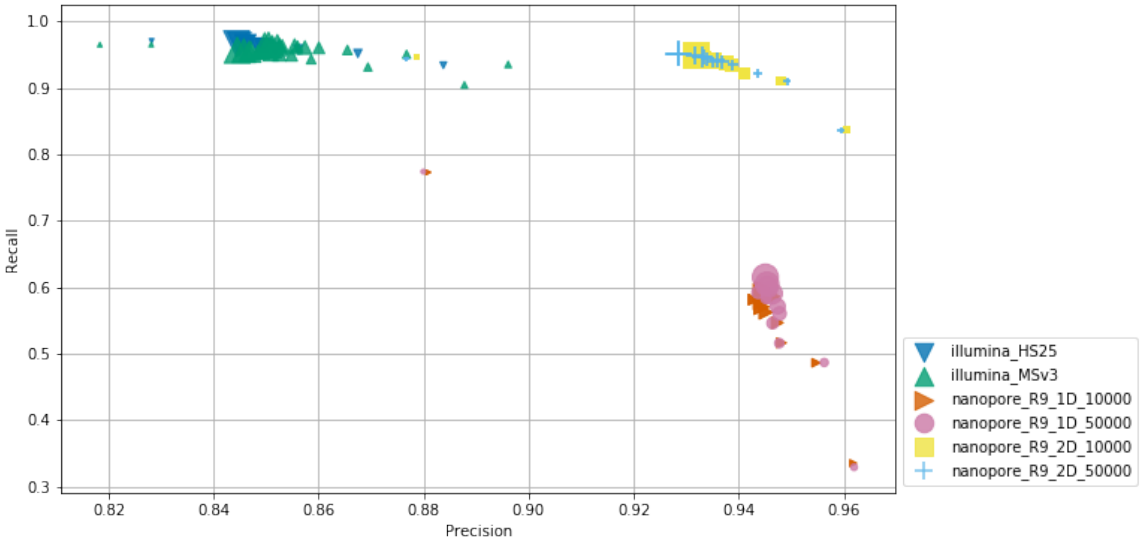


Figure 4.4: *Precision and recall for PanRG loci given sequence data from different technologies.* Using a simulated reads with the read length distributions and error profiles of different sequencing technologies, we evaluate the precision and recall when identifying the presence or absence of loci from the PanRG. Marker size represents the coverage, varying between 10X and 300X.

Both Illumina technologies resulted in a slightly higher recall and a lower precision for gene presence than Nanopore technologies at the same depth of coverage. For example, at 30X coverage, Illumina technologies achieved precision of 0.867-0.877 and recall of 0.932-0.952 by comparison with Nanopore 2D which achieved precision

of 0.949-0.951 and recall of 0.910-0.911. It seems likely that the initial definition and filtering of clusters of hits is permitting too many false positive clusters to be considered for further analysis, resulting both the slightly higher recall and the higher number of false positive genes. Nanopore 1D reads have a higher error rate than 2D reads. Whilst this increased noise is being filtered out such that precision is maintained, some true clusters of hits are also being filtered, resulting in the lower recall (0.486-0.487 at 30X coverage). It is possible that the default parameters for minimum cluster size may need to be relaxed when there is evidence of a high error rate.

There was very little difference between the precision and recall achievable from Nanopore data with a maximum read length of 10,000 or 50,000bps, or from Illumina data with read lengths of 150 or 250bp.

For each technology, the point corresponding to the precision and recall at 10X coverage is an outlier. The secondary filtering of genes and intergenic regions, based on whether the coverage along the inferred sequence for that locus is reasonable, is only enforced when the global genome coverage reaches 20X. As a result, when less than 20X sequence data is provided, this results in a higher recall and lower precision.

In all cases, additional coverage above 50X did not improve results, leading to only a small gain in recall and a small loss of precision.

4.5.2 The effect of w and k parameters on precision and recall of loci

We evaluated the affect of parameters w and k on precision and recall when identifying presence or absence of loci from the PanRG with `Pandora`. For a simulated genome, we generated 50X simulated reads with the characteristics matching 3 different profiles. Figure 4.5 demonstrates that in all cases, as w or k increases, recall decreases and precision increases. This is because the index contains fewer k-mers leading both to fewer noisy hits and fewer true hits. However, the optimal choice of w and k parameters is not consistent between read profiles. For medium accuracy Nanopore reads, the default parameters of $k = 15, w = 14$ are close to optimal, whilst for lower accuracy reads, it looks like $w = 10$ would provide extra sensitivity and for Illumina increasing to $k = 31, w = 19$ would provide moderate gains of precision.

As the values of w or k increase, the time and maximum memory required for quasi-mapping decay exponentially due to the sparsity of the index. However, the time and memory required to construct the index increase exponentially due to the complexity of pulling out long paths which cross many junctions in a local graph.

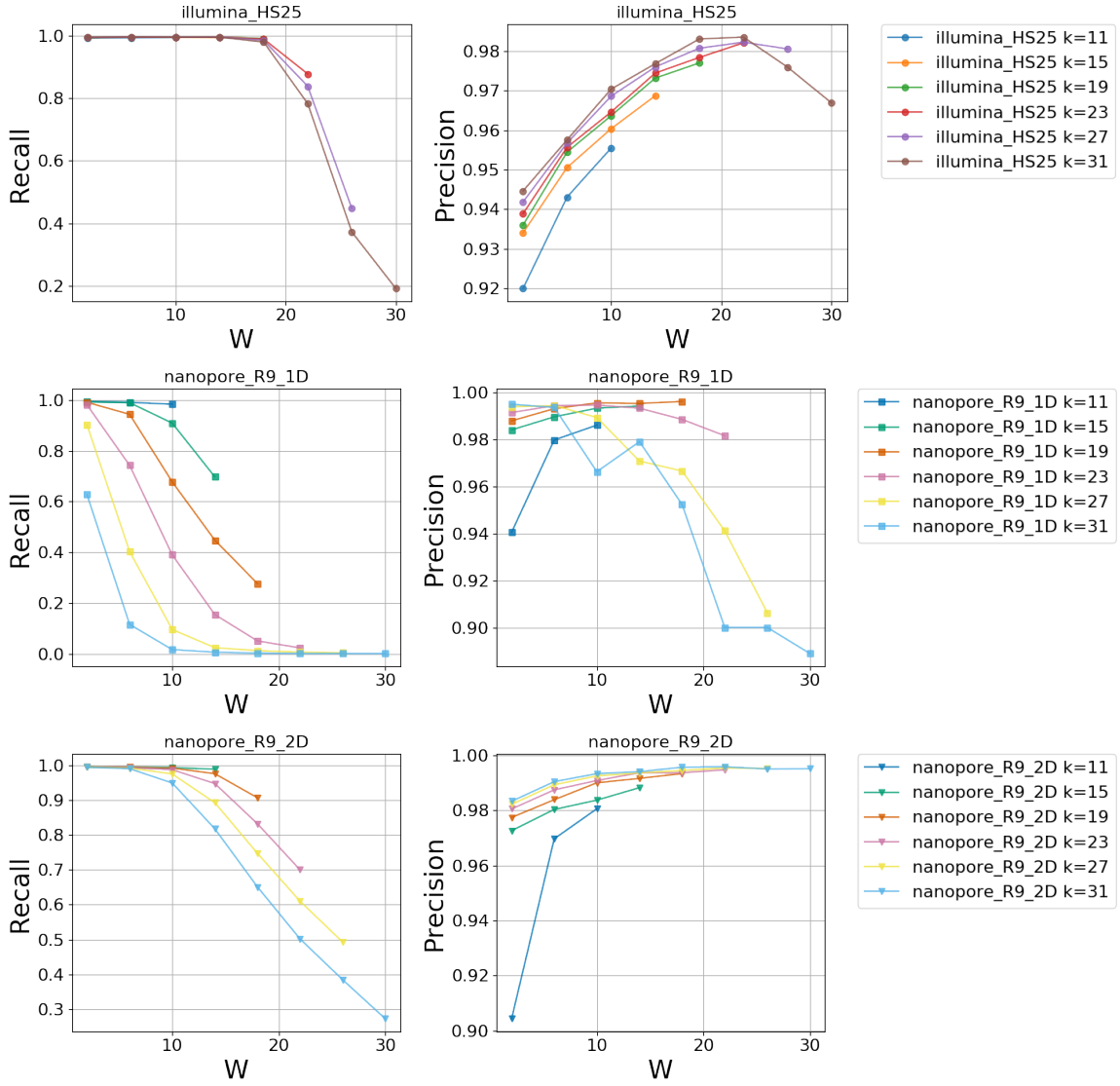


Figure 4.5: The effect varying parameters w and k on the precision and recall when identifying presence or absence of loci from the PanRG in simulated reads with error profiles and lengths matching (a) Illumina HiSeq 2500 (b) Nanopore R9 1D and (c) Nanopore R9 2D.

4.5.3 The effect of cluster threshold parameters on precision and recall of loci

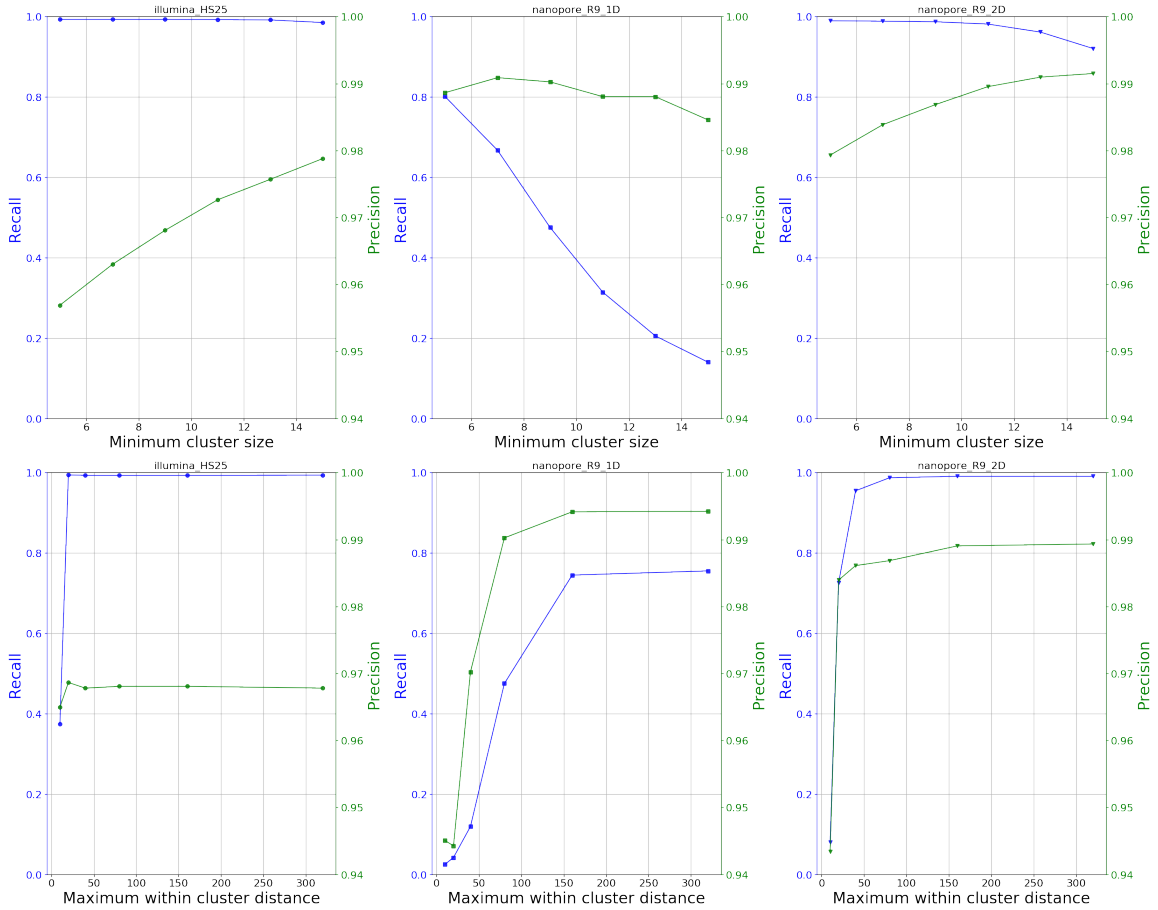


Figure 4.6: The effect varying parameters for (a) minimum number of hits in a cluster and (b) maximum distance on reads between consecutive hits in a cluster on the precision and recall when identifying presence or absence of loci from the PanRG in simulated reads.

We evaluated the precision and recall when identifying the presence of loci from the PanRG for different values of minimum cluster size and maximum within cluster distance input parameters. Figure 4.6 shows how these parameters affect these metrics using simulated reads matching 3 different read profiles.

Surprisingly, the results from simulated Illumina HiSeq 2500 and Nanopore R9 2D reads were remarkably similar, although precision was generally slightly lower given short reads which multimap more often. In both cases, as the minimum cluster size increased, the precision increased and the recall decreased. For cluster sizes above 11, recall decreased more rapidly, and precision increased only marginally. There was a sharp gain in both recall and precision as the maximum cluster distance was

increased from 10 to 20 bps. Given that the k-mer size for these results was 15bps, this is the equivalent of allowing 1 k-mer to be missed through sequencing errors or novel mutation. For Nanopore 2D, both precision and recall improve further as the maximum within cluster distance is increased. For Illumina, further increase of maximum cluster distance results in a slight dip in precision, whilst recall remains unchanged.

For Nanopore 1D reads, as the minimum cluster size is increased, recall decreases rapidly to almost 0, and precision fluctuates between 0.984 and 0.991. As maximum cluster distance increases up to 150bps, both precision and recall increase.

The default parameters in `Pandora` are a minimum cluster size of 10 and a maximum within cluster distance of 250bps for Nanopore and $2k + 1$ for Illumina. This analysis suggests that these default values are probably close to optimal for both Illumina HiSeq 2500 reads and medium accuracy Nanopore reads. However, lower accuracy Nanopore reads will require a higher maximum cluster distance and a lower minimum cluster size.

4.5.4 Precision of genes and intergenic regions from real *E. coli* sequence data

We quasi-mapped real Nanopore and Illumina reads to the PanRG with `Pandora` and default parameters, and estimated the fraction of the mosaic sequences which mapped back to the reference assembly with `bwa mem`.

For Illumina reads, `Pandora` inferred 6753 gene and intergenic sequences as present. Of these only 2 did not map back to the assembly and these were both very short (27 and 92bps). 97.9% mapped back unambiguously to a single location in the assembly. The final 142 sequences (2.1%) had duplicate records, corresponding to either sequences which appear multiple times in the genome (probably true positives) or split matches (probably false positives, but possibly where mobile elements have inserted within the gene).

For Nanopore reads, `Pandora` inferred 6041 gene and intergenic sequences, 98.2% of which mapped back unambiguously to a single location in the assembly, the same 2 again did not map back, and 103 had multiple SAM records.

In both cases, the majority of multi-mapping sequences (65%/75%) corresponded to short intergenic sequences rather than gene sequences. This suggests that setting a minimum length for PanRG input sequences could improve precision.

There was a noticeable discrepancy between the number of sequences identified as correctly present (flagged as mapped with `bwa mem`) with each technology: 96.7% of

the sequences found with Nanopore were also found with Illumina, but only 84.9% of sequences found with Illumina were found in the Nanopore data. The sequences identified with only one technology were again dominated by short intergenic sequences: 78.3% of sequences found only with Illumina and 78.7% of sequences found only with Nanopore were intergenic, and the small number of genes found with only one technology were usually less than 300bp long. These short sequences have fewer minimizing k-mers in the index. As a result, it is more likely that we fail to achieve the minimum number of hits required for detection, especially with the noisier Nanopore data.

Using `dnadiff`, sequences output by `Pandora` from Illumina reads aligned against 92.72% of bases in the reference genome and 90.66% from sequences output by `Pandora` from Nanopore reads.

4.6 Discussion

In this chapter, we described methods to index and quasi-map to a pangenome reference graph. This infrastructure provides a means to compare genomes where a single reference would make part of one genome or another inaccessible. We have shown for a single *E. coli* genome that more than 90% of the genome was covered by loci present in the PanRG and that 98% of sequences from the PanRG inferred to be present were true positives.

We do not attempt to demonstrate that more of a single genome is accessible via mapping than with conventional methods, as this is unlikely to be true. In Chapter 6 however we will demonstrate that for sets of diverse samples this infrastructure makes a greater proportion of between sample variation accessible.

We have also demonstrated the effects of different choices of parameter on the precision and recall of the loci identified as present or absent by `Pandora`. We have shown that based on simulations the default parameters are optimal for medium accuracy Nanopore sequence data and are not far from optimal for high accuracy Illumina sequence data. Given that all of the data considered in this thesis is of one of these types, the majority of the following work will use `Pandora` with default parameters. When considering only Illumina sequence data however, we will use the higher values of $w = 19$ and $k = 31$.

4.7 Limitations

4.7.1 A fundamental limitation

A pangenome reference is always incomplete

The primary limitation of any reference-based approach is that it does not generally permit discovery of novel variation. In a later section we will describe work by Michael Hall to enable discovery of short novel variants where there is evidence that a gene or intergenic region exists in the sample, but the reference mosaic sequence has poor support from reads in some interval. However this will not enable discovery of entire novel genes or insertion sequences.

For species with an open pangenome, every new genome is likely to contain some new genes not seen before (see Figure 2.2). The number of such genes can be controlled by inclusion of many reference sequences in the PanRG. Additionally, careful curation of input sequences by the user can ensure that features of interest are well represented in the PanRG. For example, it could include all known mobile elements, or all orthologous gene clusters found in a large collection of samples from a clinically characterized strain. In this way, the pangenome reference graph makes accessible a much larger fraction of the pangenome for comparison than previous reference-based methods.

4.7.2 Current limitations

Minimizer hit data structure uses too much RAM

At time of writing this thesis, a major limitation is the amount of RAM required to quasi-map reads. The RAM use scales linearly with the number of minimizer hits, and this is a prime candidate for development to improve the memory requirement.

At present, hits found between the index and the reads which are identified as part of a cluster are stored for the duration of the single sample mapping process. Each hit includes a `uint32_t` to represent each read identifier, read start position, local graph identifier and k-mer node identifier, a bool to represent the relative orientations of the k-mer in the read and graph, and a vector of `uint32_t` to represent the path in the local graph from which the k-mer originated. The coordinate information allows regions of the relevant reads to be extracted for *de novo* discovery. However there is clearly redundancy in the amount of information stored, and once counts have been added to the k-mer graph nodes, much of the information ceases to be required.

In Section 6.6.2.1 I will briefly describe work that has been done subsequent to this thesis, to improve this performance.

Scaling to multi-species pangenomes

The current values of w and k enable effective indexing of the PanRG and identification of those loci which are present. Within *Enterobacteriaceae*, genetic material is shared between several species, and it might be interesting to construct a PanRG at this family level. In order to scale to a PanRG which includes the pangenome of multiple species, it is likely that sections of the PanRG will need to be considered in parallel. Part of the filtering of clusters involves considering if a cluster is contained within a larger cluster, as may be the case if there are many k-mers in common between several local graphs. For this process to work when the sections of the PanRG are considered separately, we may need to sort the collection of local graphs in the PanRG so that those with a high degree of k-mer sharing are placed close together in the PanRG file, and are considered as part of the same PanRG chunk. We may also need to increase the size of w and k so that there are fewer (w, k)-minimizers shared between graphs.

4.8 Future work

4.8.1 Inferring the order of local graphs in a sample

Bacterial genomes are typically very gene-dense, and often more than 90% of the DNA sequence is made up of protein-coding regions [Rosenberg and Rosenberg, 2012]. These genes typically vary in length between 500-1500bps. As a result, long reads of length 10,000bps and up are expected to cover numerous genes and this long range information can be used to infer the order of loci from the PanRG in the sample. Such an inference can provide valuable information about the context of individual genes in the sample, such as whether it lies in the chromosome or a plasmid. When we consider making use of sequence data from multiple infections or metagenomic datasets, this long range order information would also aid deconvolution of reads into the respective genomes.

We have started developing an approach to infer the order of local graphs using a de Bruijn graph. We use an 'alphabet' of local graph identifiers. Each node is a fixed size tuple of oriented local graphs, and edges correspond to adjacency of consecutive tuples in reads.

A very recent preprint [Ruan and Li, 2019] for new rapid long read *de novo* assembler `wtdbg2` used a very similar idea. They construct a 'fuzzy de Bruijn' graph by chopping reads into 1024bp segments, divided into 4 256bp bins (like a k-mer with $k = 4$, and 256bp 'words' forming the alphabet). The 4-bin segments may be merged if they are aligned together based on all-vs-all read alignment.

Based on the fact that most genes in the genome appear uniquely, we are confident that with further development, ordering could be successfully implemented in this way within `Pandora`.

4.8.2 Handling mixed sequence data

Bacteria live in mixed communities and often many versions of the same species co-exist in the same environmental niche unless some external factor (e.g. antibiotic use) selects for one form [Didelot et al., 2016]. However, most sequencing preparations include picking a single bacterial clone to culture and extract and most assembly and variant calling methods assume *a priori* that only a single clonal genome is present.

Allowing for the presence of multiple genomes of a species when variant calling would do two things. Firstly, it would add to our knowledge and understanding of within-host evolution and carriage vs disease. Secondly, it would remove the necessity of a culture step before extraction, potentially enabling more rapid diagnostics from sequencing. For *M. tuberculosis*, [Votintseva et al., 2017] have already demonstrated that it is possible to provide same day drug-susceptibility diagnostics directly from sputum samples.

When designing these methods, much thought was put into making them extensible to mixtures. The long range information available from long read sequence data ought to make it much easier to deconvolve reads which show evidence of having different genes, or a different ordering of genes using graph operations. The analyses described in the next chapters could then be applied independently to partitioned sets of reads. In particular, we allow the same local graph from the PanRG to be present multiple times in the pangenome sample graph.

Chapter 5

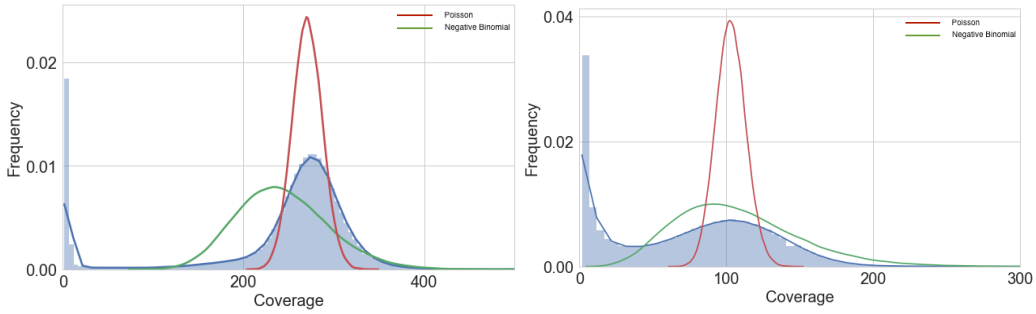
Local sequence inference and genotyping for a single haploid clonal sample

5.1 Introduction

The accessory part of the pangenome plays an important role in enabling bacteria to adapt to different environments and as such is biologically relevant in many problems. However, differences in the presence and order of pangenome elements can often render parts of this accessory genome inaccessible to variant calling (see Figure 1.1).

In this chapter we describe methods to use quasi-mapped reads to rapidly infer the sequence at a locus in the PanRG as a mosaic of reference sequences (the *mosaic sequence*). If the PanRG reference panel is sufficiently well chosen, and given that new variation is introduced into bacterial genomes by mutation and recombination mechanisms, we expect this to be a reasonable approximation. For Nanopore data in particular, we show that such an approach can yield sequences with a per base accuracy of at least 99.8% in less than 1 CPU hour, which is many hours faster than existing approaches.

Traditional reference based variant calling uses a single 'closely related' reference genome as a first approximation of the sample genome and identifies the first order corrections from reads where they differ from this reference. In this chapter we detail algorithms to instead use a path through the PanRG as this first approximation of the sequence for a single sample. We then detail how genotyping is performed with respect to this graph using a Poisson model, allowing SNPs, indels and complex features to be described. We show that this model results in highly accurate and sensitive results, even with noisy data. In section 6.6.2, we describe work by another



(a) Illumina k-mer coverage distribution (b) Nanopore k-mer coverage distribution

Figure 5.1: The k-mer coverage distribution and the fitted Poisson and Negative Binomial distributions.

student to extend this approach with novel variant discovery, bringing it in line with standard approaches.

These steps are implemented in our software `Pandora` as part of the `map` protocol.

5.2 Algorithms

Each of the operations described in this chapter are performed independently on each local graph in the PanRG. As such, we will describe each method for a single local graph.

5.2.1 Mosaic sequence inference

To infer the closest mosaic of reference sequences for each locus from a local graph, we use a dynamic programming algorithm on the k-mer graph constructed by the graph indexing process.

In the previous chapter, we described the process of identifying the presence of local graphs in sequence data by finding clusters of hits between the reads and local graphs. For each minimizing k-mer in the k-mer graph, we store the number of hits we have found in these clusters.

We model k-mer coverage with a negative binomial distribution to allow for greater dispersion than in the standard Lander-Waterman/Poisson model, and use the conventional simplifying assumption that k-mers are read independently.

Let $r + s$ be the number of times the underlying DNA was read by the machine, generating a k-mer coverage of s, and r instances where the k-mer was sequenced with errors. Let $1 - p$ be the probability that a given k-mer was sequenced correctly

when read. The probability mass function for parameters $r > 0$ and $0 \leq p \leq 1$

$$f(s; r, p) = \frac{\Gamma(r+s)}{\Gamma(r)s!} p^r (1-p)^s$$

has mean $\frac{(1-p)r}{p}$ and variance $\frac{(1-p)r}{p^2}$. We solve these for r and p using the observed k-mer coverage mean and variance across all k-mers in all graphs for the sample.

Let Θ be the set of possible paths through the k-mer graph which could correspond to the true genomic sequence from which reads were generated. For $\theta \in \Theta$ let $\{X_1, ..., X_m\}$ be independent and identically distributed random variables representing the k-mer coverages along this path. Let $\mathcal{D}$ be the k-mer coverages seen in the read dataset. We use a dynamic programming algorithm (Algorithm 5) to find the path which maximizes log-likelihood-inspired score

$$\hat{\theta} = \{ \arg\max_{\theta \in \Theta} l(\theta | \mathcal{D}) \}$$

where

$$l(\theta | \mathcal{D}) = \frac{1}{m} \sum_{i=1}^{m} \log f(x_i; r, p).$$

By construction, the k-mer graph K is directed and acyclic. We can therefore find an ordering of the nodes such that for any directed edge from x to y, x comes before y in this ordering (this is called a *topological ordering*). Such an order can be found in time linear in the number of nodes and edges using Kahn's algorithm [Kahn, 1962], but in our implementation we use the C++ `stdlib` sort algorithm. Let $\{x_1, ..., x_n\}$ be the coverage counts for a topological ordering of the n nodes of K, and define

$$prob(i) = \log f(x_i; \hat{r}, \hat{p}),$$

where $\hat{r}, \hat{p}$ are parameters estimated from the global k-mer coverage distribution as outlined above.

For some genes, there exist versions of the sequence which appear truncated by comparison with others. The result in our graph is that there are some k-mer nodes with both other k-mers and the graph *terminus* as outnodes. In these cases we must allow for both of these options, but the graph terminus by definition has no sequence to have coverage on. To handle this, we assign a threshold probability to the terminus, and require the score for extending the sequence to be greater than this threshold.

Each k-mer node in K is uniquely defined by a path of length k in the local graph, and for $w \leq k$ every letter in a sequence is contained in at least one (w, k)-minimizer of that sequence (except in the first and last $w + k - 1$ bases). The found k-mer node

ALGORITHM 5: Find the mosaic path

Input : K – a directed acyclic k-mer graph with n nodes
t – terminus probability threshold
Output: $\mathbf{p} = \{p_1, ..., p_m\}$ – a path through K

```
Function FindConsensus (K)
    M ← arr(0, n)                    /* initialize intermediate score array of n zeroes */
    len ← arr(0, n)                  /* initialize length array of n zeroes */
    prev ← arr(n, n)                 /* initialize previous array of n n's */
    for i = n, ..., 1 do
                                     /* consider the nodes in reverse topological order */
        max_mean ← −inf
        max_length ← 0
        for j ∈ outnodes(i) do
            if (j.is_terminus() and t > max_mean)
             or M[j]/len[j] > max_mean
             or (M[j]/len[j] ≡ max_mean and len[j] > max_len) then
                if j.is_terminus() then
                    max_mean = t
                else
                    max_mean = M[j]/len[j]
                    max_len = len[j]
                M[i] = M[j] + prob(i)
                len[i] = len[j] + 1
                prev[i] = j
    prev_node = prev[0]
    p ← ∅
    while prev_node < n do
        p.append(prev_node)
        prev_node = prev[prev_node]
    return p
```

path therefore corresponds to overlapping paths of length k through the local graph, and can be transformed back to a path through the local graph. The string along this path is the mosaic sequence for the sample, and the path may be used as the reference sequence for VCF construction.

5.2.2 Genotyping

Genetic variation is usually described with respect to a single reference sequence. For a prepared panel of variants described with respect to this reference, *genotyping* solves the classification problem, statistically inferring which alleles in the panel are supported by the sequence data for a sample.

By construction, a pangenome reference graph contains within it a description of variation we have previously seen in a population. By selecting a path through the graph to act as the reference sequence, we can describe all other variation in the graph with respect to this path. In this way, we intuitively extend the idea of genotyping to *graph genotyping* as described in Figure 5.2.

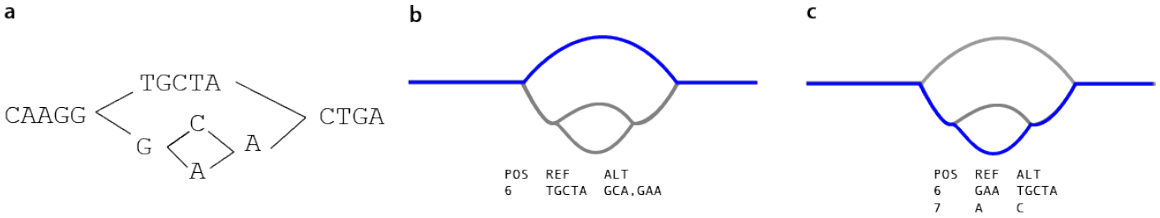


Figure 5.2: *Variation as described with respect to different paths*: for toy local graph in (a) we summarize how graph variation would be described with respect to different choices of reference path shown in blue. In (b) the reference path is the top path and in (c) the bottom path.

We first construct a VCF describing variation in a local graph with respect to a predefined reference path, which can be user-specified or inferred (the mosaic path). Bubbles in the local graph result in records and alleles in the VCF. We collect information about the coverage on indexed k-mers which overlap each allele, and store their mean, mode and sum. We use these statistics to genotype each site in the VCF.

We model the mean k-mer coverage on the true allele as Poisson distributed with rate equal to the expected depth of coverage d. To account for noise, we assume that coverage on alternative alleles arises as a result of an error process with rate ϵ. We also have information about the proportion of k-mers along each allele with zero (or near-zero) coverage. If we assume k-mer coverage is Poisson distributed with mean

d, then the probability that a given k-mer has non-zero coverage is

$$p = 1 - \text{Poisson}(d, 0) = 1 - \exp(-d).$$

We scale the probability of an allele proportional to the probability of seeing z zero-covg k-mers out of l with the term

$$(1-p)^{z/l} p^{(l-z)/l}.$$

Let A be the set of alleles, and for $a \in A$, let c_a be the mean k-mer coverage over allele a, and let c_b be the sum of the mean k-mer coverage over each other allele in A. Let z_a be the number of k-mers covering allele a with zero coverage, and l_a be the number of k-mers covering allele a. The the probability of the observed data assuming a is the true allele is

$$e^{-d} \frac{d^{c_a}}{c_a!} \ \epsilon^{c_b} \ (1-p)^{z_a/l_a} p^{(l_a - z_a)/l_a} \tag{5.1}$$

hence the log likelihood of an allele is

$$\begin{aligned} \log \mathcal{L}(a|\text{observed data}) = &- d + c_a \log(d) - \log(c_a!) + c_b \log(\epsilon) \\ &- d z_a / l_a + \log(1 - \exp(-d))(l_a - z_a)/l_a. \end{aligned}$$

We ascribe the called genotype to the most likely allele a, and define the confidence to be

$$\min_{b \in A \setminus \{a\}} \| \log \mathcal{L}(a|\text{observed data}) - \log \mathcal{L}(b|\text{observed data}) \|.$$

Because the VCF by default contains records representing nested levels of variation, there are combinations of records where alleles called in this independent way could conflict. We therefore parse the VCF once more for conflicting calls and keep only the most confident call in such cases.

The resulting VCF includes ALT calls where a sample differs from the chosen reference path, and REF calls where we have identified it follows the reference path. In this way, we continue to describe the sample with respect to all the variation contained in the pangenome reference graph.

5.3 Methods

5.3.1 Evaluation of mosaic sequence accuracy

We first demonstrate that given a 'perfect read' corresponding to a path in the PanRG we can correctly infer the sequence. We then use simulations to confirm that for

sequences that are contained in some local graph in the PanRG, we can accurately retrieve the original sequence from noisy reads. Finally we use publicly available data for a reference strain of *E. coli* to show that we can maintain this mosaic sequence accuracy using real whole genome sequence data.

Note that for $w \leq k$,every letter of each sequence is guaranteed to be covered by at least one minimising k-mer. The exception is in the first and last $w + k - 1$ bases of any sequence, where there may only be one minimizing k-mer and so the inferred sequence for these regions may be randomly chosen from the graph, leading to lower accuracy in these flanks. In this work we have built graphs which precisely match genes or intergenic regions of interest, and so we have lower accuracy at the ends of these sequences. A simple workaround would be to add w+k-1 bases of flanking sequence during graph construction.

To evaluate the output mosaic sequences, we first map them back to the truth using `bwa mem` [Li, 2013]. We then extract from the SAM file the length of the matched sequences and the number of mismatches in the `NM` tag for each alignment. We plot a histogram for the number of mismatches and estimate the per-base error rate as the total number of mismatch bases divided by the total length of aligned bases. For each aligned sequence we also note whether the cigar includes an insertion or deletion.

5.3.1.1 Accuracy of mosaic sequences from simulated read data

We generated 5 different random paths through each local graph in the *E. coli* PanRG (except for simple graphs where there were fewer possible paths). For 10000 paths of at least 200bps in length, we simulated 100X coverage of Illumina reads with `ART` [Huang et al., 2012]. Due to limitations of the simulator, to allow simulation of Nanopore reads we added 15000bp of the letter 'A' to the beginning and end of each sequence. We then simulated 250 reads using `NanoSim-H` [Yang et al., 2017] [Karel Brinda, `http://doi.org/10.5281/zenodo.1341249`], and restricted to those simulated reads which mapped back to the original random path sequence with Minimap2 [Li, 2018]. This resulted in approximately 100X coverage of simulated Nanopore reads. To get 'perfect reads' we used 30 copies of the random path sequence. Each simulated dataset was mapped to the PanRG with `Pandora` and the resulting mosaic path compared with `bwa mem` to the original path.

5.3.1.2 Accuracy of mosaic sequences from *E. coli* K12 whole genome sequence data

We used the Nanopore and Illumina sequence data and reference assembly for the reference strain of *E. coli* K12 described in Section 4.4.6. Whilst the Nanopore and Illumina read datasets will have evolved slightly from the reference genome, the reference sequence is expected to lie in the graph. Since this graph gives an upper bound for variation we are able to detect, this provides an approximate positive control for our methods.

Inferred mosaic sequences were mapped using `bwa mem`. to the reference assembly. Sequences which multimapped were eliminated, to reduce confounding results. These arose both when sequence associated with a locus appeared multiple times in the genome, and when shared k-mers between loci resulted in a locus being incorrectly identified as present (often shorter sequences). Sequences for which there was no match in the SAM file were deemed to be *false positives* and all others *true positives*.

5.3.2 Evaluation of genotyping accuracy

We use simulations to evaluate the precision and recall of `Pandora` and to compare with other variant callers. We compare performance with Illumina data to SNP caller `Snippy` [`https://github.com/tseemann/snippy`], a wrapper for Freebayes. For Nanopore data we compare with `nanopolish`, using their `variants` protocol [Loman et al., 2015]. We described these methods further in Background section 2.4.2.1.

5.3.3 Precision and recall of genotype calls from error-free reads

We simulated a reference genome from the PanRG as in Section 4.4.2. We simulated long and short error-free reads with the commands

```
nanosim-h --perfect --circular -n 30000 <genome.fa> --unalign-rate 0
--max-len 10000
nanosim-h --perfect --circular -n 550000 <genome.fa> --unalign-rate 0
--max-len 300 --min-len 250
```

To evaluate the recall of single-reference variant callers, a standard approach is to take a reference genome, and reads from that genome. By introducing variants to the reference and calling variants from the reads with respect to this mutated reference,

we know the truth about the variants that should be called. We can then evaluate recall by calculating which of these introduced variants can be discovered from calls. Since `Pandora` does not require a reference, we select the panel of variants in the same way and handle the fact that the VCF output by `Pandora` is with respect to a different (graph) reference to the other methods, still checking which variants in the panel can be discovered from calls.

We used `Pandora` to construct a panel of variation present in the PanRG with respect to the simulated reference genome. A candidate panel of variants was selected by filtering for size (< 10bp) and discarding variants which overlapped. We then randomly selected variants from this panel, including each independently with probability $p = 0.1$. A new simulated 'reference' genome was created by updating the original reference with selected variants. The simulated reads now represent sequence data from a genome which is closely related to this simulated reference.

We called variants with respect to the mutated reference with `Snippy` and `Nanopolish`, and with respect to the graph with `Pandora` and looked for correct calls at the sites where we introduced variants.

These simulations represent a 'best case' scenario for all tools - for `Pandora` we know *a priori* that the variants are in the graph and so should be called, and for `Snippy` and `Nanopolish` we have guaranteed that the reference sequence is not too divergent from the sample genome.

We evaluated recall as described in Section 2.4.3.2 and precision as described in Section 2.4.3.1.

5.3.3.1 Precision and recall of genotype calls from real *E. coli* K12 sequence data with respect to a simulated reference

We used `Pandora` to construct a panel of variation present in the PanRG with respect to the *E. coli* K12 reference sequence described in Section 4.4.6. We then repeated the above analysis using real sequence data instead of simulations. This introduced 5201 variants to the 5Mb genome, of which 3423 were isolated SNPs, 32 were indels and 1733 were clustered SNPs. This is a best case simulation; in practice for *E. coli* we might expect to see a nucleotide divergence of $< 3\%$ in core genes [Touchon et al., 2009] but an average of more than 10 SNPs/kb [Schloissnig et al., 2013].

In all cases we used 100X sequence data, and we additionally call variants with `Pandora` from just 30X sequence data.

We evaluated recall as described in Section 2.4.3.2 and precision as described in Section 2.4.3.1.

5.3.3.2 Running Snippy and Nanopolish

We ran Illumina SNP caller `snippy` using the command:

```
snippy --cpus 8 --outdir snippy_outdir
    --reference <reference_assembly> --se <illumina_read_fq>
```

and Nanopore variant caller `nanopolish` using the command:

```
nanopolish index -d <raw_fast5_dir>
    -s <albacore_summary_file> <nanopore_read_fq>
minimap2 -ax map-ont -t 8 <reference_assembly> <nanopore_read_fq>
    | samtools sort -o reads.sorted.bam -T reads.tmp
samtools index reads.sorted.bam
mkdir -p nanopolish.results/vcf
python3 /nanopolish/scripts/nanopolish_makerange.py
    <reference_assembly> | parallel --results nanopolish.results
    -P 2 nanopolish variants
      -t 8
      -w 1
      --reads <nanopore_read_fq>
      --bam reads.sorted.bam
      --genome <reference_assembly>
      -o nanopolish.results/vcf/nanopolish.1.vcf
      -q dam,dcm
      --ploidy 1
```

Pandora and `snippy` both output a genotype likelihood for each allele of all SNPs and indels, and from this we were able to define a genotype confidence. For `nanopolish`, which does not specifically define allele likelihoods or genotype confidences, we used support fraction in lieu of a confidence to stratify results.

5.3.4 Pandora

We ran `Pandora` for each analyses using the singularity image and default parameters using command:

```
singularity exec shub://rmcolq/pandora:pandora pandora map
-p <pangenome_prg> -r <reads_fa> --max_covg <100|30>
```

with additional argument

```
--illumina
```

when the input data was Illumina and

```
--genotype
```

to genotype. When using real sequence data (from *E. coli* K12), we added

```
--max_diff 16
```

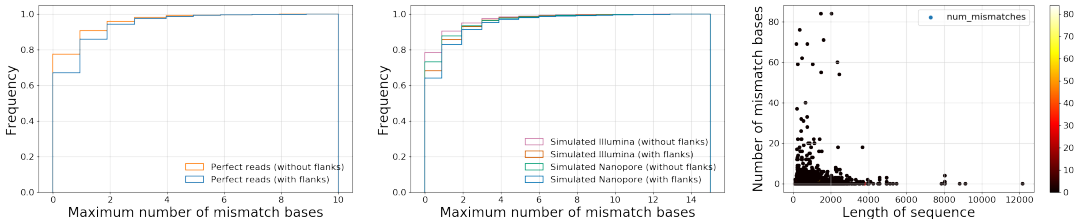
for Illumina data based on the parameter results in Chapter 4.

5.4 Results

5.4.1 Evaluation of mosaic sequence accuracy

For genome sequences that lie in the PanRG, we expect the mosaic sequences inferred by `Pandora` to be an exact match, both from Illumina and Nanopore sequence data. We evaluate this with perfect and imperfect simulated reads and real sequence data.

5.4.1.1 Accuracy of mosaic sequences from simulated read data



(a) (b) (c)

Figure 5.3: (a) Cumulative distribution for the number of mismatch bases between inferred mosaic sequences and expected sequences for perfect reads. Analysis of inferred mosaic sequences was repeated with $w + k - 1$ bp flanks (28bps) removed from the start and end of each gene/locus just before alignment (b) Cumulative distribution for the number of mismatch bases between inferred mosaic sequences and expected sequences for simulated Nanopore and Illumina reads, with and without flanks of 28bps. (c) For each inferred sequence, we plot the number of mismatch bases by comparison with the truth and the length of the sequence and colour by the length of the longest insertion or deletion identified in the SAM cigar, here shown from Illumina simulations.

To evaluate mosaic accuracy, we first simulated 30X perfect and 100X noisy reads from random paths through local graphs in the PanRG, and compared the sequence inferred by `Pandora` to the reference sequence used to generate the reads.

Perfect reads represent the ceiling for per base accuracy of sequences inferred with Pandora. For perfect reads the full inferred sequences mapped back with 99.93% average per base identity. Figure 5.3a shows the distribution of the number of mismatch

bases per inferred local graph mosaic sequence. In 66.7% of cases the sequences inferred were a perfect match and 93.8% had up to 2 base mismatches by comparison with the truth. We investigated whether these were due to lower sensitivity in the $w+k-1$ base pair flanks, where not every letter is covered by a minimizer, leading to lower sensitivity to distinguish between alternative sequences. For 10% of sequences where the inferred mosaic had a mismatch base by comparison with the reference, these mismatches were only in the first or last $w+k-1$ of the sequence. However, the overall per base accuracy when flanks were excluded was only slightly higher at 99.94%. We will discuss the cause of this apparent ceiling later.

From simulated reads matching the length and error profiles of Nanopore R9 2D and Illumina HiSeq 2500 technologies, we got very similar results. Simulated Illumina/Nanopore reads mapped back to the truth with 99.89%/99.90% average per base identity. This is comparable with the accuracy of *de novo* Nanopore assemblies (99.84% [Wick et al., 2019]), although considerably less accurate than typical Illumina assemblies (99.98% [Zerbino and Birney, 2008]).

Figure 5.3b shows the distribution of the number of mismatch bases per inferred local graph mosaic sequence. For Nanopore/Illumina simulations, 90.8%/91.9% of inferred sequences had up to 2 mismatch bases by comparison with the truth, rising to 92.9%/94.2% when flanks were omitted. Results from Illumina simulations were marginally more similar to those from perfect reads, most likely due to the low error rate.

For all types of read, the mismatch distribution had a long tail, with a very small number of individual sequences having a larger number of mismatch bases by comparison with the true sequence. Figure 5.3c shows that none of these can be explained by the presence of a large deletion or insertion in the middle of the inferred sequence by comparison with the truth. Instead it is likely to be due to large soft clipped flanks. We know that a number of local graphs in the PanRG we constructed from MSA including sequences of very different lengths due to real insertions or deletions or truncations seen in the reference sequences. The dynamic programming algorithm is slightly vulnerable to the occasional incorrect inference across these regions in the graph if the mean coverage on k-mers in the true allele is not sufficiently high. Further investigation is needed into whether better calibration of the dynamic programming algorithm can reduce these errors.

The results from this section demonstrate that when the true sequence exists as a path through a graph in the PanRG we can infer it as accurately from noisy sequence data as from perfect reads.

Experiment	flanks	Avg id (%)	Identical (%)	$\leq$ 2 mismatches (%)
Perfect Reads	yes	99.93	66.7	93.8
Perfect Reads	no	99.94	77.3	95.6
Simulated Nanopore	yes	99.90	63.8	90.8
Simulated Nanopore	no	99.91	72.8	92.9
Simulated Illumina	yes	99.89	67.4	91.9
Simulated Illumina	no	99.92	77.8	94.2
K12 Nanopore	yes	99.73	33.5	75.3
K12 Nanopore	no	99.77	45.1	81.5
K12 Illumina	yes	99.84	54.0	90.9
K12 Illumina	no	99.88	70.1	94.0

Table 5.1: Evaluation of inferred mosaic sequences for different input data

5.4.1.2 Accuracy of mosaic sequences from *E. coli* K12 whole genome sequence data

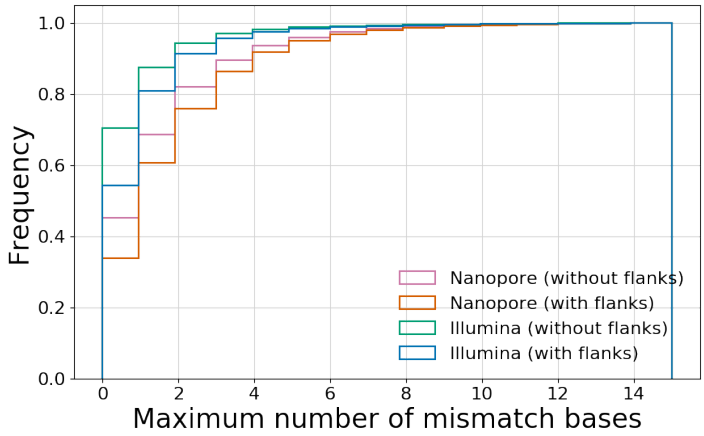


Figure 5.4: Cumulative distribution for the number of mismatch bases between inferred mosaic sequences and expected sequences for real sequence *E. coli* K12 Nanopore and Illumina reads. Analysis of inferred mosaic sequences was repeated with $w+k-1$ bp flanks (28bps) removed before alignment.

We repeated this analysis for 100X of real Nanopore and Illumina whole genome sequence data generated from *E. coli* K12, achieving a very similar per base accuracy of 99.73/99.84% for Nanopore/Illumina reads despite expecting some additional errors as a result of evolution between the reference isolate and read isolates. Mapping used 12GB of RAM and took just 15 minutes from the raw Nanopore reads. Table 5.1 shows how the results compare with the simulations, and Figure 5.4 shows the distribution of the number of mismatch bases per local graph mosaic sequence. When flanks are

included, a lower proportion of those local graphs present in the genome were inferred with 0 mismatch bases, but this proportion is increased to 81.5/94.0% of sequences for Nanopore/Illumina when flanks are omitted.

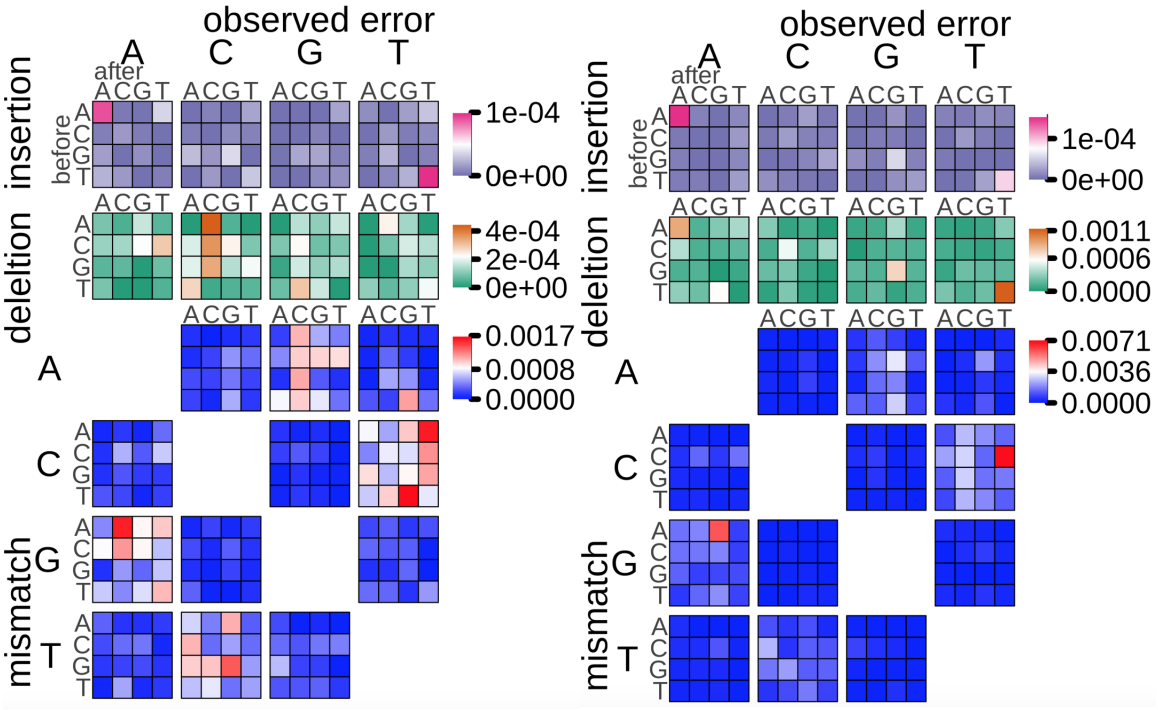


Figure 5.5: AlignQC heatmap showing the context of base errors in inferred mosaic sequences for (a) Illumina, (b) Nanopore sequence data. The large type text describes possible types of base error, and the small type represents the base before and after in the truth. Note that the scales are automatically generated by AlignQC and so are different for each data type.

We investigated the sequence context of mismatch bases using AlignQC [Weirather et al., 2017], a tool which reports information about the quality of mappings, error rates and error biases from BAM format alignment data. Figure 5.5 shows heatmaps generated by AlignQC from the alignment of the inferred mosaic sequences to the truth assembly. For Nanopore reads, the largest source of contextual bias is towards CCT being called as CTT in the mosaic sequence. This bias in Nanopore is likely to be as a result of methylated bases which typically occur in this way [Wick et al., 2019], leading to sufficient support from reads for this incorrect allele. In the results from both Nanopore and Illumina data, indels tended to occur in the context of homopolymers, although this was more common in Nanopore mosaic sequences than Illumina. Mismatch errors in Illumina mosaic sequences were much rarer than in

Nanopore mosaic sequences (not the different scales) and occurred in a wider variety of contexts, reflecting the more uniform error profiles typical in Illumina reads.

5.4.2 Evaluation of genotyping accuracy

5.4.2.1 Precision and recall of genotype calls from error-free reads

We simulated a genome by concatenating sequences from random paths for genes and intergenic regions in the PanRG. We simulated 30X perfect short (250bp) or long (max length 10,000) reads from this genome to represent Illumina and Nanopore reads. We modified the simulated genome with small variants from the PanRG and evaluated `Pandora` with the simulated reads. Figure 5.6 shows the precision and recall of introduced variants.

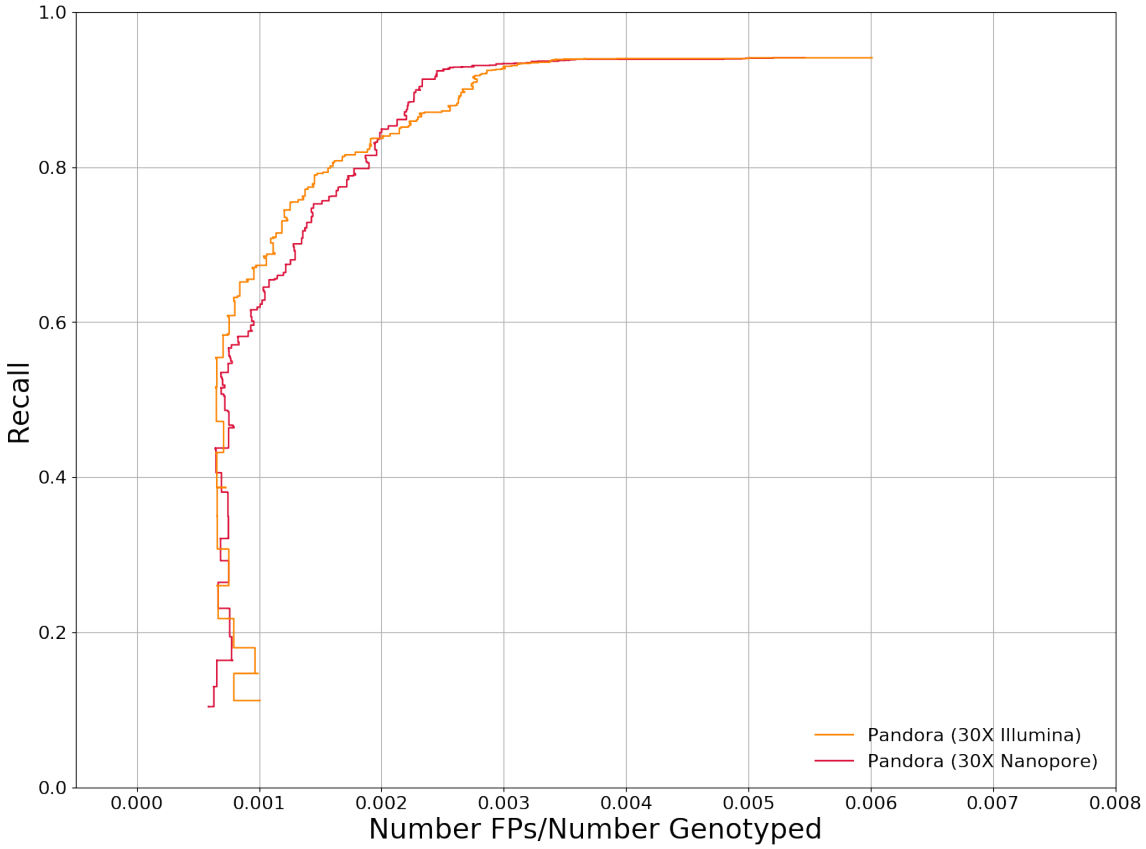


Figure 5.6: *Precision and recall of genotype calls from error-free reads.* For simulated short and long reads, we show the recall of simulated reference mutations and 1 - precision based on all calls. Results are stratified by genotype confidence thresholds.

As expected, given the same coverage and no errors, results are very similar for both read length distributions. From both long and short error-free reads, `Pandora`

achieved 90% recall with 99.8% precision. A maximum recall of 94% was achieved by both. This represents better accuracy than the equivalent mosaic results, with a slightly lower recall. Of the 2327 simulated variants, 2198 were discovered from error free long reads in the full VCF. Of those missed, in 53 out of 129 the incorrect allele was called. In 74 cases, no match was found in the VCF. This occurs in `Pandora` when sites are discarded because the called genotype is incompatible with the called genotype of an overlapping site with a higher likelihood.

5.4.2.2 Precision and recall of genotype calls from real *E. coli* K12 sequence data with respect to a simulated reference

We modified the reference assembly for our *E. coli* K12 datasets with variants from the PanRG, and evaluated `Pandora`, `Snippy` and `Nanopolish` for precision and recall of the introduced variants (thus evaluating real data, but with a simulated reference genome for reference dependent methods).

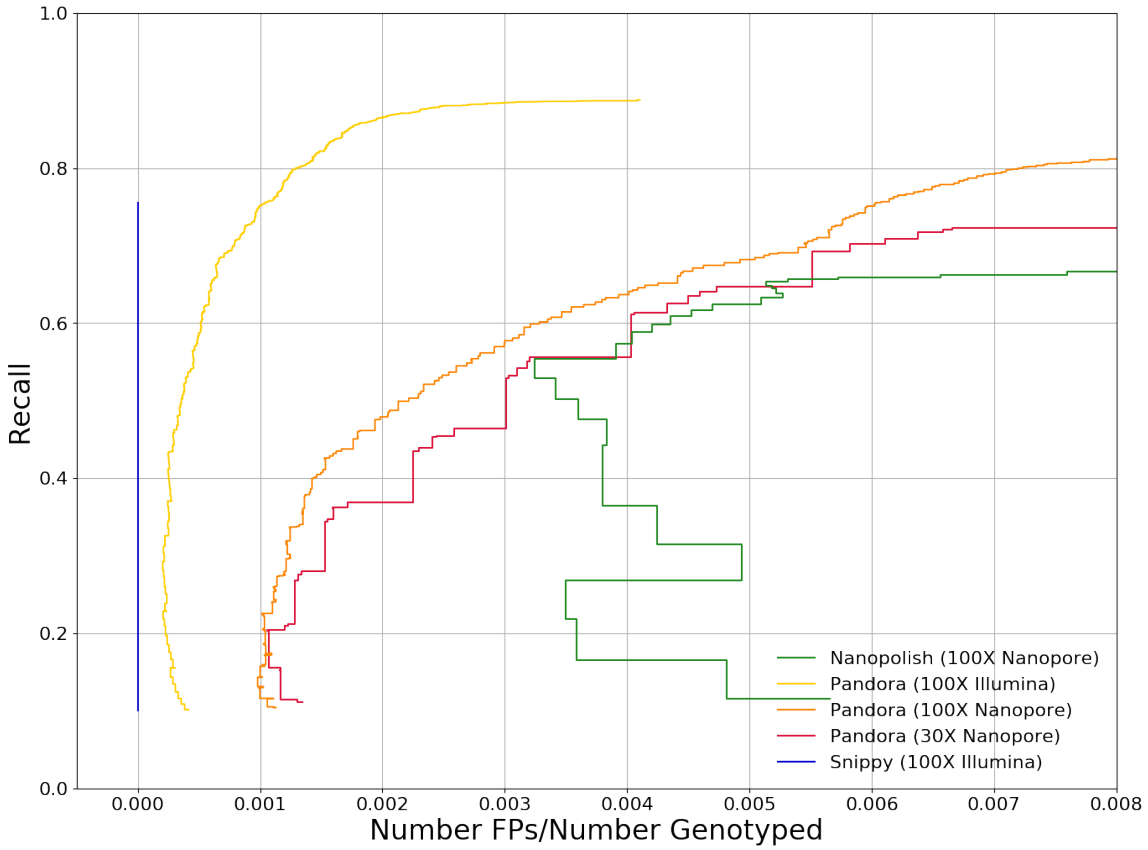


Figure 5.7: *Precision and recall of different methods evaluated with E. coli K12 reads and a simulated reference.* For each method, we show the recall of simulated reference mutations and 1 - precision based on all calls. Results are stratified by genotype confidence thresholds.

Figure 5.7 shows the precision and recall of introduced variants achieved by each method, stratified by genotype confidence. The highest precision was achieved by `Snippy` which had 75.5% recall with 100% precision. By comparison, `Pandora` achieved 88.6% recall with Illumina data. The lower recall with `Snippy` was due to difficulties calling non-SNP variants. While it did succeed in calling such variants in some cases, in 613/1270 of the variants missed, there was a record in the VCF but the allele called was not a match (likely a SNP was called where a slightly longer variant was present). In a further 562 cases there was no record in the VCF matching the site, and of these only 13 were SNP sites.

As expected, genotyping with Illumina data achieved higher precision and recall than genotyping with Nanopore sequence data. With 99.7% precision `Pandora` was able achieve, 61.4% recall with 100X of Nanopore data, and 55.5% with 30X of Nanopore data. `Pandora` outperformed `Nanopolish`, which achieved results with 100X coverage very similar to `Pandora` with just 30X coverage: 55.4% recall with just under 99.7% precision.

As with error free reads, recall with `Pandora` plateaus, unable to call the final 10% of simulated variants despite them being selected to lie in the graph. In some places, nesting in the graph gives rise to multiple records in the VCF. If the genotype calls made for such records are incompatible, the most likely record is kept, and the genotype call for the other is omitted. As a result, it is the case that not all sites in the graph are genotyped in a given sample. This is an area for future development.

Genotyper	**Data**	**CPU time (h)**	**Max memory (GB)**
Snippy	Illumina	0.25	0.97
Nanopolish	Nanopore	5.25	3.8
Pandora	Illumina	0.70	22.2
Pandora	Nanopore	0.61	11.9

Table 5.2: CPU time and memory requirement of each method to call variants in *E. coli* K12 from 100X of either Nanopore or Illumina sequence data.

Table 5.2 shows the time and memory requirements of each method. `Snippy` was both the fastest and most memory efficient tool, with `Pandora` requiring approximately twice as much time and much more memory to run. Nanopolish was an order of magnitude slower than both `Pandora` and `Snippy` (and this is know to scale very badly with increased depth of coverage e.g. `https://github.com/jts/nanopolish/issues/228`). However, with 30X Nanopore sequence data, `Pandora` requires only twice the RAM (6.72GB) to call variants with greater precision and recall than the

current state of the art Nanopore caller `Nanopolish` in 1/10th the time (0.56 h). Finally, we highlight that this is not a like-for-like comparison. `Snippy` and `Nanopolish` are both performing variant discovery, and intend to get as close as possible to the sample genome. However for `Pandora` we have been evaluating how close the nearest mosaic in the graph is to the sample. The Illumina, Nanopore and truth reference for K12 all come from different cultured isolates, and the reference in particular is old; thus some or all of them may have sequence outside the graph.

5.5 Discussion

In this chapter, we described methods to infer the sequence at a locus as a mosaic of reference sequences and to genotype with respect to a path in the pangenome reference graph. These methods provides a substrate to allow genotyping across the pangenome. By comparison, traditional single-reference variant calling methods are only able to access variation in the core part of the genome.

Our results for mosaic sequence inference and genotyping from error free reads suggest that there is still some room for improvement of the underlying algorithms. However, we have shown that we are able to achieve 99.5% precision with 70.6%/88.8% recall when genotyping from Nanopore/Illumina sequence data.

With Illumina sequence data, we infer sequences and genotype with similar accuracy as with error free reads. In addition, genotyping recall appears to be higher than can be achieved with other methods and we will show in the next chapter that there are even more significant gains of recall when comparing multiple samples.

For Nanopore sequence data, we are able to infer the sequence of genes and intergenic regions in a fraction of the memory and time that a full assembly would require and with comparable per base accuracy. In addition we are able to make variant calls with greater precision and recall than the only currently available alternative `Nanopolish`, and can do so in a fraction of the time.

5.5.1 Current Limitations and Future Development

5.5.1.1 Lower accuracy in $w + k - 1$ bp flanks

We noted in the results section that the accuracy of inferred sequences was lower in the $w + k - 1$ bps at the start and end of each sequence. Provided $w \leq k$, outside of these flanks every letter of each path through the local graph is guaranteed to be contained in a minimizing k-mer, which leads to greater accuracy when differentiating between different alleles. For genes or other loci of interest where we care about having this

higher accuracy throughout the entire sequence, adding additional flanking sequence to either end of each graph could ensure this.

5.5.2 Mosaic sequence inference errors

When inferring sequences as a mosaic of references, at each point in the dynamic programming algorithm we choose the outnode with the highest mean log probability. As we progress through the nodes towards the first nodes in the k-mer graph, new k-mer coverages only slightly change the mean k-mer coverage along the extended path. In addition to this, when the algorithm is implemented in C++, to avoid errors when comparing floats, we consider a float equal to another if it is within a ± 0.000001 range.

As a result we have less sensitivity to distinguish between alleles towards the start of sequences as shown in Figure 5.8. Modifying the algorithm to distinguish between alternatives with very similar mean log probabilities based instead on the kmer coverages of a smaller number of recent kmers on the graph paths would enable us to further improve sensitivity of these mosaics.

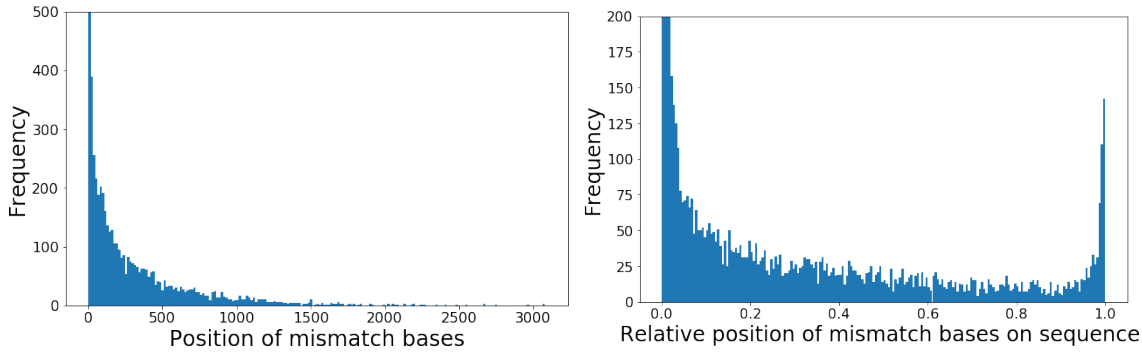


Figure 5.8: Positions within mosaic sequences at which mismatch base errors occurred, shown (a) raw and (b) scaled by sequence length. Positions taken from the SAMFILE generated when comparing inferred mosaics from Illumina *E. coli* K12 to the truth assembly.

5.5.2.1 An improved genotyping model

The current genotyping model achieves reasonable results for both Illumina and Nanopore sequence data. Rerunning the analysis in Section 5.4.2.2 without the second term (penalising coverage on alternate alleles) or the third term (penalising based on missing coverage) in genotype probability equation 5.1 reduces the precision of results and so we conclude that each term is beneficial. However, from observation we

have seen that precision decreases when the path used as the reference sequence is more distant in the graph to the sample path. In this situation, alleles are longer and some contain long shared flanks. As a result the assumption that all coverage on alternate alleles is due to errors is incorrect. To handle this, we propose an improved model as follows:

For each allele $a \in A$ with non-zero (or sufficient) coverage on its k-mers, we identify the set of unique k-mers on the allele $k(a)$. We then work with $c_{a'}$, the mean k-mer coverage over unique k-mers on the allele and $c_{b'}$ the sum of the mean k-mer coverage over unique k-mers on alternate alleles.

The probability of the observed data assuming a is the true allele is then

$$e^{-d}\frac{d^{c_{a'}}}{c_{a'}!}\ \epsilon^{c_{b'}}\ (1-p)^{z_a/l_a}p^{(l_a-z_a)/l_a}.$$

We have not had time to implement and test this new model, and expect some corner cases to have to be considered further such as when there are no unique k-mers on an allele. This could be handled either by filtering out alternate alleles with very low support, or considering equivalence classes of alleles.

5.5.2.2 Error biases in Nanopore sequence data

We noted above that for Nanopore sequence data, the errors in the mosaic sequences were biased towards certain motifs, in particular CCT identified as CTT. The conversion of cytosine to 5-methylcytosine as part of the CC(A/T)GG motifs, known as Dcm methylation, is common in *E. coli* and other species of *Enterobacteriaceae* [Gomez-Eichelmann et al., 1991]. It was recently demonstrated by [Wick et al., 2019] that training a neural network for ONT basecalling on taxon-specific data resulted in a much higher consensus sequence accuracy for *Enterobacteriaceae* samples, largely because they could account for these methylation motifs.

Future development for `Pandora` could include allowing a user-input error bias model to be used as a prior as part of the mosaic sequence algorithm. Alternatively, running methylation aware `Nanopolish` on the inferred mosaic sequences would further improve their accuracy and would not add too much of an additional time cost, due the the high accuracy of `Pandora` mosaic sequences.

5.5.2.3 An upper bound for mosaic sequence accuracy and the case for *de novo* discovery

We have demonstrated that when a sequence is contained in the PanRG, the mosaic sequence inferred by `Pandora` has a per base accuracy comparable to polished

nanopore assemblies. However, as the method infers the sequence as a mosaic of the reference sequences, new variation which is not present in the PanRG cannot be inferred. This means that the distance of the sample from the graph provides an upper bound on the per base accuracy that can be achieved without a process of *de novo* discovery. In Section 6.6.2 we will describe work by another PhD student to add this feature.

5.5.2.4 Multi-copy genes and mixed datasets

All the algorithms described in this section assume that only a single path through a local graph can be present in the read dataset. For multi-copy genes this assumption is violated. As a result we anticipate that our results are likely to be less accurate in this situation. We plan to handle multi-copy genes and mixed datasets in a combination of ways:

Firstly, the context of a local graph can be used to partition reads ('synteny'). This is something which would be easier given long read sequence data, where it is expected that a read covers numerous local graphs. Once reads have been partitioned, a path can be inferred separately for each set of reads.

Secondly, we can look for evidence of multiple paths through a local graph such as the presence of extraordinary coverage. In this case, we can output a shortlist of paths which are supported by coverage and solve an expectation maximisation problem (similar to RNA transcript quantification e.g. SALMON [Patro et al., 2017]) to identify the true paths present and partition reads accordingly.

We have partial work for both of these approaches, and expect this to aid future work inferring the order of local graphs within a sample.

Chapter 6

Variation inference for collections of haploid clonal samples

6.1 Introduction

The comparison of collections of bacterial genomes is central to microbiology and is increasingly used to aid our understanding about how a species evolves, transmission within an outbreak, the mechanisms of antimicrobial resistance and virulence, and the differences between pathogenic and commensal isolates.

However comparison of divergent samples is hard (recall Table 3.1), and typically analysis is constrained to either focus on (small) variation in the part of the pangenome core to all samples, or larger scale variation such as presence/absence of genes.

In this section we describe methods introduced in `Pandora` as part of the `compare` protocol which allow a hierarchy of variation between samples to be captured. This includes both a presence/absence matrix showing which local graphs have been found in each sample, as well as a VCF representing differences within each local graph between any samples that contain it.

We compare `Pandora` and two SNP calling tools and evaluate how well each enables differences between genomes to be captured. We show by successively adding samples that `Pandora` is able to capture variation across the pangenome whereas single reference-based approaches progressively fail. We demonstrate that `Pandora` enables variation in shared regions of the pangenome to be described, even when other regions of the genome are more divergent. To our knowledge this is the first time that this has been achieved.

6.2 Algorithms

6.2.1 Workflow for comparison of a dataset

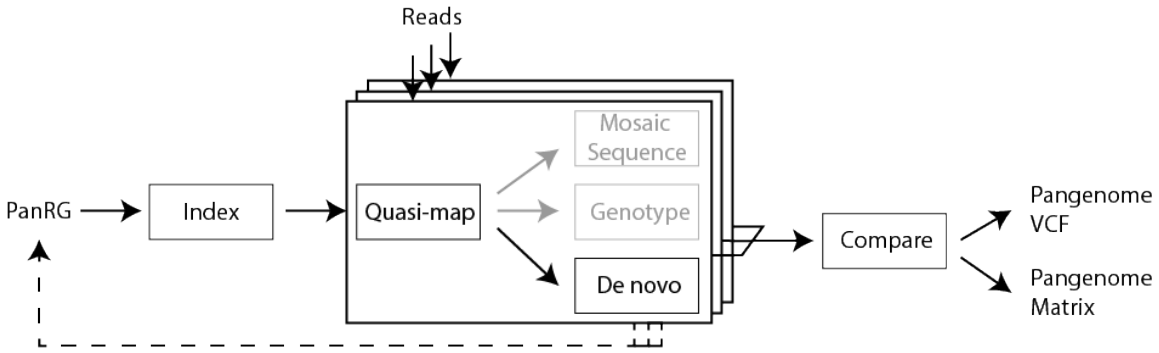


Figure 6.1: *Workflow in* `pandora compare`

Figure 6.1 outlines the workflow used by `pandora compare` to enable comparison of read datasets for several samples. Firstly each sample in turn is quasi-mapped to the PanRG, and their mosaic sequences inferred. Coverage information is stored in a multisample pangenome graph. Then, for each local graph in the multisample pangenome graph, a VCF is constructed to represent the graph variation with respect to some reference path and the relevant samples are genotyped. Finally `Pandora` outputs a pangenome matrix and multisample VCF. When *de novo* discovery is included, after the inital mapping phase, the PanRG is updated with variants discovered in each sample. We will describe this in detail in Section 6.6.2.

6.2.2 Choice of graph reference path

As outlined in Section 3.4, when no user-defined reference sequence is provided, a path through the local graph is chosen to be the VCF reference sequence for each local graph based on the input sequence data.

When comparing samples, it is helpful for small differences between sequences to look small. Figure 6.2 shows for a toy example how a SNP difference between two samples might appear to be a larger variant when described with respect to a different choice of reference path.

For this reason, the default reference path is chosen to be maximally close to the sample paths. To do this we first make a copy of the k-mer graph. For each sample, we increment the node coverage by one along the path covered by the sample mosaic sequence. We then re-use the maximum-likelihood algorithm from Algorithm

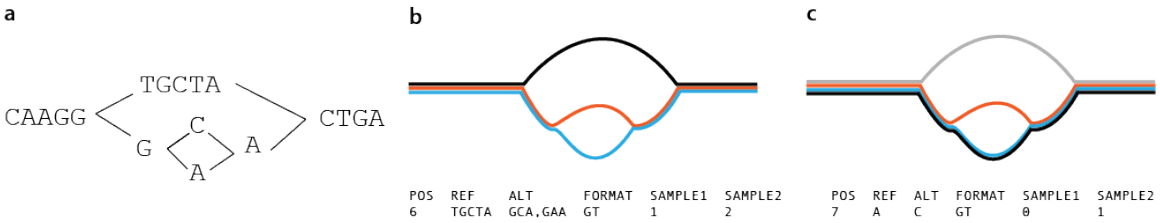


Figure 6.2: *Choice of reference path may disguise small variants with shared flanking sequence*: for toy local graph in (a) and two samples shown in orange and blue, figures (b) and (c) show VCF details describing the SNP difference between these two samples with respect to 2 different choices of reference path, shown in black. In (c) the difference is explicitly written as a SNP, whilst in (b) it is nested within longer alleles.

5 with a modified probability function, defined such that the probability of a node is proportional to the number of samples covering it.

6.3 Methods

We first confirm using simulations that by using a hierarchical approach we are able to describe variation between divergent sets of genomes. We go on to demonstrate how our pangenome reference approach compares to two single reference genotypers using a range of reference sequences, both when comparing two closely related samples and when comparing increasing numbers of divergent samples of the same species.

6.3.1 Describing variation between divergent sets of genomes

For a species like *E. coli*, the multiple mechanisms of inheritance mean that there can be a lot of structural diversity between genomes. And yet as we described in Section 2.2.3 it is useful to be able to describe similarities in accessory parts of the genome. To demonstrate `Pandora`'s ability to describe similarities between diverse and similar genomes, we simulated a toy example dataset. Pairs of random paths were selected from 400 local graphs to create sequences $\{r_1, ..., r_{400}\}$ and $\{r'_1, ..., r'_{400}\}$, which differ by variants contained within the PanRG. We then concatenated the following sets of sequences to get 4 simulated 'genomes':

$$\begin{aligned} g_1 &= \{r_1, ..., r_{200}\} \\ g_2 &= \{r'_1, ..., r'_{200}\} \\ g_3 &= \{r_1, ..., r_{50}, r_{201}, ..., r_{250}, r_{301}, ..., r_{400}\} \\ g_4 &= \{r_{251}, ..., r_{300}, r'_{51}, ..., r'_{100}, r'_{301}, ..., r'_{400}\} \end{aligned}$$

By this construction, genomes g_1 and g_2 have the same genes in the same order and differ predominantly by SNPs and indels. Genomes g_3 and g_4 share half their genomes,

in which they differ predominantly by SNPs and indels, and also each contain both unique material and some shared sections of either genomes g_1 or g_2.

Whilst this construction is highly constrained and artificial, it allows us to evaluate whether SNP differences between each pair of samples are identifiable. Under this construction, random paths $\{r_1, ..., r_{100}\}$ are each present in all but one sample, $\{r_{101}, ..., r_{200}\}$ and $\{r_{301}, ..., r_{400}\}$ are shared between 2 samples and $\{r_{201}, ..., r_{300}\}$ are uniquely present.

We simulated 30X each of Nanopore reads using `NanoSim-H` [Yang et al., 2017] [Karel Brinda, `http://doi.org/10.5281/zenodo.1341249`] and of Illumina reads using `ART` [Huang et al., 2012]. Read datasets were compared using `pandora compare`.

We first produce a graphical representation of the pangenome matrix and confirm it contains the above genes in the correct combination for each genome. For each pair of genomes we also evaluated the fraction of confident SNP differences as identified with the `dnadiff` module from `MUMMER3` [Kurtz et al., 2004], which were identifiable from the multisample VCF, and the fraction of VCF sites with a call for both samples where the calls were correct. We produce a heatmap of these pairwise fractions.

6.3.2 The effect of diversity of a dataset on pan/reference genome approaches

For reference based methods, the choice of a reference sequence affects how accessible regions of the sample genome are for variation detection. We demonstrate this effect first for two related samples of *E. coli* from a hospital outbreak, and then when an additional 2 unrelated samples of the same species are added to the set of samples being compared (details below). Figure 6.3 shows the proportion of sequence shared between pairs of these 4 samples.

We used `pandora compare` to produce a multi-sample VCF of variation between the samples across the PanRG for the two outbreak samples, and again for all 4 samples. As `Snippy` and `nanopolish` do not have a multisample mode, we ran them separately on each sample using a selection of reference genomes. This created single-sample VCFs with respect to each reference, with variation between any two samples captured as differences between the variants called in each sample VCF.

6.3.2.1 Outbreak dataset

The two outbreak samples were selected from a hospital outbreak dataset [Decraene et al., 2018] which we investigate further in Section 7.1. This outbreak was primarily

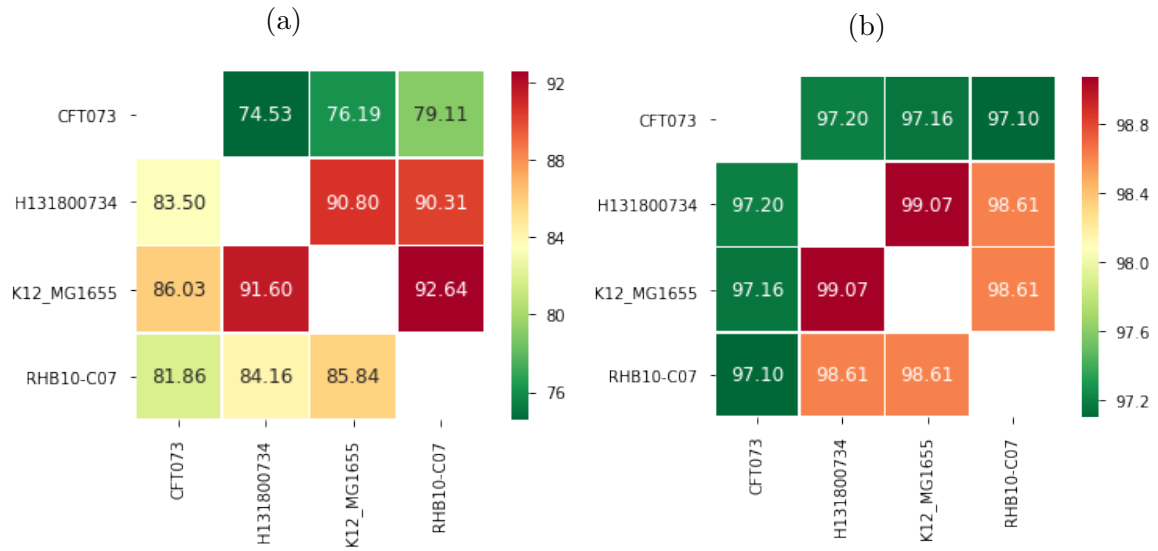


Figure 6.3: *Sequence similarity between the assemblies of 4 E. coli samples as estimated using* ***dnadiff***. (a) For each genome (y-axis) we show the percentage of that genome which aligns against each other genome on (x-axis). (b) The percentage of bases which agree between each pair of genomes within 1-1 aligned segments.

concentrated in two wards of a hospital and Illumina sequencing had already been performed as part of ongoing research into the outbreak. DNA was re-extracted and sequenced for 16 samples with Oxford Nanopore MinION for this project by Sophie George, and a subset of 5 of these samples were sent for PacBio SMRT sequencing.

The Pacbio raw reads were assembled using `canu v1.6` [Koren et al., 2017] to produce the draft assembly. The contigs from the draft assembly were then aligned to a standard reference genome (accession: NZ_CP025268.1) using the `nucmer` utility from the `MUMMER3` [Kurtz et al., 2004] package. The contigs were oriented to match the reference, trimmed based on the `nucmer` alignments and circularized using `minimus2` [Sommer et al., 2007]. This work was done by Srividya Ramakrishnan and Michael Schatz.

Nanopore sequencing was performed in March and April 2017 using the most up to date protocols, kits and flow cells. DNA was prepared for all but one sample using the '1D Native barcoding genomic DNA' protocol for multiplex sequencing, and the remaining sample was prepared according to the '1D Genomic DNA by ligation' protocol. Preparation used SQK-LSK108 and EXP-NBD103 kits and R9.4 flow cells. Multiplexed samples were sequenced in batches of 2-5 samples.

Nanopore reads were first basecalled with ONTs Albacore command line tool v2.7.1, then `Porechop` [`https://github.com/rrwick/Porechop`] was used to demultiplex (where necessary) and trim adaptors.

PacBio assemblies were polished with Illumina data using 6 rounds of `Pilon` [Walker et al., 2014]. Illumina reads were then mapped with `bwa mem` [Li, 2013] against the polished assembly, and positions where the major allele fraction fell below a threshold of 95% were excluded from further analyses. All further analyses use this polished PacBio assembly as a truth for the sample.

We used two samples (identifiers H131800734 and H151080744) for the analysis in this Section, and perform a full comparison the samples in the next chapter. H131800734 was collected in 2013 from the ward where the outbreak was concentrated, and H151080744 was collected in 2015 from a neighbouring ward and belongs to the same sequence type, ST216. Previous analysis of these two samples suggests they are very closely related.

6.3.2.2 Comparison of divergent samples

We compared a diverse set of 4 samples, including outbreak sample H131800734 from the previous analysis, and the *E. coli K12* strain used in previous chapters and described in Section 4.4.6. We will refer to this sample with id K12_MG1655. In addition, we included 2 additional samples sequenced by the REHAB consortium as part of a study to compare hybrid assembly methods for complex bacterial genomes [Maio et al., 2019]. Hybrid assemblies combine information about the genome structure from long read data with accurate sequence from short reads to generate fully resolved, accurate genome assemblies. This study included 18 *Enterobacteriaceae* isolates from farm animals and the environment and 2 reference strains. Of these 20 samples, 4 were from *E. coli*.

For each sample they have made available Nanopore, Illumina and PacBio sequence data. They constructed hybrid assemblies from long and short read sequence data using `Unicycler` [Wick et al., 2017a], which has been shown to be the optimal hybrid tool for fully closed genomes. They considered a variety of long read filtering methods and found that random subsampling was the most effective read filtering method. We therefore used this resulting assembly for our sample truth. Unfortunately, the signal level data generated during Nanopore sequencing is not available, and so `Nanopolish` is unable to call variants for these samples.

We used CFT073 (ST73, a reference strain) and RHB10-C07 (isolated from sheep faeces) from this dataset. There were very few differences between the PacBio and Nanopore based hybrid assemblies for these samples, so for consistency with the other two samples, we used the PacBio hybrid assemblies for our analysis.

In all cases, we subsampled read files to contain 100X or 30X coverage, using the first 100X coverage for Nanopore read sets which are already randomly ordered, and a random subset of reads for Illumina.

6.3.2.3 Running Pandora

We ran `Pandora` using the singularity image and default parameters using command:

```
singularity exec shub://rmcolq/pandora:pandora Pandora compare
-p <pangenome_prg> -r <read_index> --genotype
```

with additional argument

```
--max_covg 30
```

when using just the first 30X coverage in each read file, and

```
--illumina -w 19 -k 31
```

when the input data was Illumina (based on recommended parameters in Chapter 4).

A single sample VCF was generated for each sample from the output multisample VCF so that downstream analysis could process the results of each method in the same way.

6.3.2.4 Running `Snippy` and `nanopolish`

To capture the differences in performance of `Snippy` and `nanopolish` when calling variation with respect to different references, we used a range of reference assemblies. We downloaded 228 RefSeq complete *E. coli* reference genomes. This set was composed by Nabil-Fareed Alikhan to include one reference per ribosomal MLST, although it is not entirely representative because many clonal complexes do not have a complete genome available.

We ran Illumina SNP caller `Snippy` [`https://github.com/tseemann/snippy`] and Nanopore variant caller `nanopolish` [Loman et al., 2015] on each outbreak sample with each reference individually as in 5.3.3.1.

We first selected a panel of 10 reference assemblies based on the `Snippy` calls for these two samples, selected to cover the full range of estimated precision and recall fractions. To do this we set a threshold for recall (60%), ordered references by precision, and selected 10 evenly spaced members of this list. Due to the huge number ($\tilde{2}0$) of CPU hours required for each `nanopolish` run, we were unable to select references in the same way for `nanopolish`.

When we extended the initial analysis from the 2 outbreak samples to 4 samples, we used MASH [Ondov et al., 2016] to add additional references to this panel, including the reference most closely related to each sample and the top 3 references most closely related to the collection of samples as a whole.

6.3.2.5 Evaluating precision and recall

To evaluate precision and recall, we used the polished/hybrid PacBio assembly for each sample as a truth assembly. We masked out regions of each of these assemblies where Illumina reads were less than 95% concordant.

We evaluated the precision for each sample VCF as described in Section 2.4.3.1. The results tables of true positive calls and false positive calls stratified by genotype confidence were amalgamated.

Since there is no truth panel describing the variation between these diverse samples, for each pair of samples we used `dnadiff` [Kurtz et al., 2004] to compare the truth assemblies and construct a panel of confident SNP differences. We evaluated the number of these which could be identified from the pair of VCFs as described in Section 2.4.3.3. We combined results over all pairs of samples to estimate the total pairwise SNP recall. By this measure, a SNP site which segregates all 4 samples would be counted 6 times, whilst a SNP site which segregates 2 samples only would be counted once. The primary reason for this is to avoid having to merge these distinct call sets (and the inevitable problems of translating coordinates). In core regions of the genome, this 'double counting' is likely to bias in favour of the most accurate methods. Variation in the accessory genome will be identified more rarely and will segregate pairs of samples less often, hence is less likely to be 'double counted'. As a result we do not expect this metric to bias in favour of `Pandora`.

Analysis was performed using in house software `minos` [`https://github.com/iqbal-lab-org/minos`].

6.3.2.6 Plotting results

For each choice of genotyping method and reference genome we evaluated the precision and recall at different genotype confidence thresholds. For method and reference combinations which were able to achieve a minimum recall of 10%, we plot a step function to represent the precision and recall as genotype confidence is increased.

6.4 Results

6.4.1 Comparing diverse simulated genomes

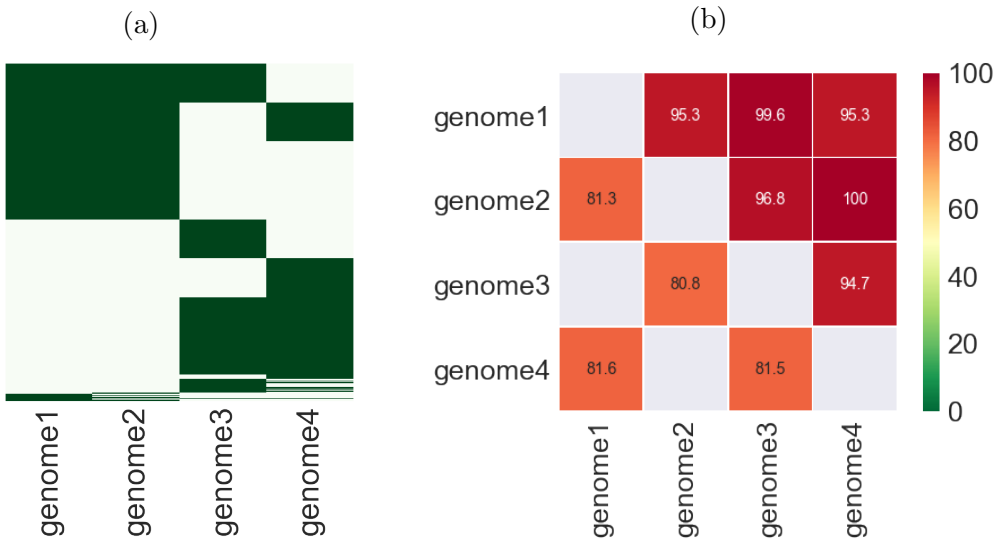


Figure 6.4: (a) Presence/absence matrix output by `Pandora`, sorted so that the 400 genes from which random paths $r_1, ..., r_{400}$ were generated are in order, followed by any additional called genes. Dark green represents presence, and white represents absence. (b) For each pair of simulated genomes we show in the bottom half of the heatmap the proportion of `dnadiff` SNP differences which were correctly identified by `Pandora` in the multisample VCF. The top half of the graph gives for each pair of genomes the fraction of VCF sites where both genomes were genotyped and were called correctly.

We used `pandora compare` to analyse 4 simulated genomes. The pangenome matrix is represented pictorally in Figure 6.4a, sorted so that the 400 simulated genes appear in order starting at the top. It shows that all 400 simulated genes are inferred to be present or absent correctly by `Pandora`, although a small number of additional genes were incorrectly identified as present in each simulated genome (13 for Nanopore and 35 for Illumina data).

Figure 6.4b shows that for 4 pairs of simulated genomes where `dnadiff` identified SNPs, 80.8-81.6% of these could be identified from the resulting multisample VCF. Additionally for any pair of simulated genomes, 94.7-100% of sites where a call was made for both were correct. As in other analyses where we evaluate recall using dnadiff SNPs, there is a ceiling at around 80% sensitivity. We will address possible reasons for this in the discussion.

This demonstrates that we are able to discriminate between regions of the pangenome

which are unique to samples, or common to a subset of samples as well as to call variants in regions of the genome that are shared between at least two samples.

6.4.2 Comparison of genotyping methods on a pair of outbreak samples

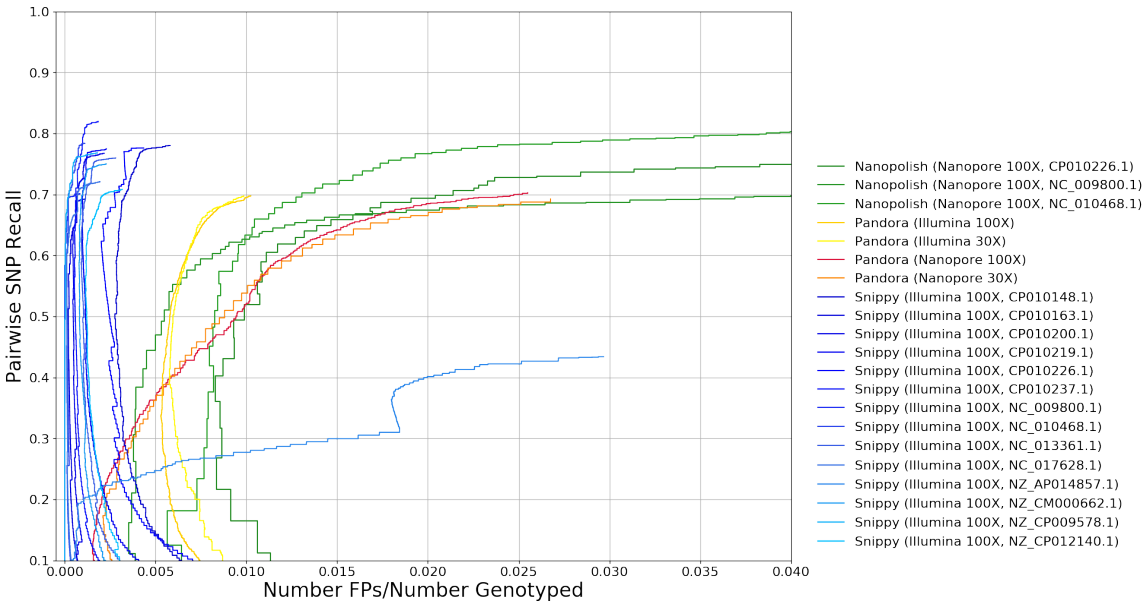


Figure 6.5: *Precision-Recall comparison of methods for a choice of reference genomes stratified by genotype confidence.* The y-axis shows the fraction of high confidence SNP differences between samples H131800734 and H151080744 which were captured by each method with at least this confidence and the x-axis shows the fraction of VCF calls of at least this confidence which are false positives.

We evaluated the precision and recall of different genotyping methods for a range of reference genomes.

Figure 6.5 shows that for both `Snippy` and `nanopolish`, choice of reference affected both the precision and recall of results. For `Snippy`, 13/15 references resulted in a maximum recall of 69.2-82.0%, one reference achieved a maximum recall of 43.4% and the final reference failed for sample H131800734. At 60% recall, the fraction of false positives ranged between 0.0000269 and 0.00287, lower than any other method. For `nanopolish` only 3 references were similar enough to both samples to allow any calls, but these achieved a maximum recall of 70.5-81.0%. However, these calls also had a high proportion of false positive calls, and at 60% recall the three references resulted in the fractions of 0.00813-0.0111 false positives, increasing rapidly for any gain in recall.

With `Pandora` the recall achieved was at the lower end of the spectrum typically achieved with successful runs by other methods. `Pandora` had a maximum recall of 69.7% for Illumina data, and 69.2-70.3% for Nanopore data. In this analysis, recall is estimated based only on confident SNPs identified between the sample assemblies. There is therefore a contrast between this analysis, measuring recall of SNPs and showing better recall for `Nanopolish` and `Snippy`, and that shown in Figure 5.7, which measured better recall for `Pandora` when considering both SNPs and longer variants. This contrast emphasises the strengths of both `Snippy` and `Nanopolish` when describing SNPs between two closely related samples as opposed to longer variants.

At 60% recall, the fraction of false positives was 0.00669 with 100X Illumina and 0.0120 with 100X Nanopore, higher than the other methods. For Nanopore sequence data this performance is unsurprising given that `Nanopolish` uses raw signal data in addition to the basecalled reads, allowing for the inclusion of more complex models to handle methylated bases. Whilst we do not intend to rely on raw signal, this is perhaps an indication of the gains that could be achieved if a user-defined error bias model were added to `Pandora`.

Finally, we note that `Pandora` achieved almost identical results with just 30X coverage as it did with 100X. This is a real strength, particularly for Nanopore sequence data where the high error rate has typically prevented genotyping except at high coverages.

6.4.3 Comparison of genotyping methods on a more diverse set of 4 samples

We repeated this comparison on a diverse set of 4 samples, using the same panel of 15 reference sequences for the single-reference genotyping methods, and calling variants once for the panel of 4 with `pandora compare` allowing a reference path to be inferred. Figures 6.6, 6.7 and 6.8 show the precision and recall of these methods for increasing numbers of divergent samples.

Figure 6.6 shows the comparison of 2 samples, with results looking very similar to Figure 6.5. Both `Nanopolish` and `Snippy` achieve higher pairwise SNP recalls with a lower fraction of false positive calls than `Pandora`. Whilst these two samples are not related, much of their genomes align.

When we add in the third sample CFT073 in Figure 6.7, we find that the recall achievable with `Snippy` drops dramatically with most of the reference choices to less than 50%. This is indicative of there being regions of the individual genomes which are

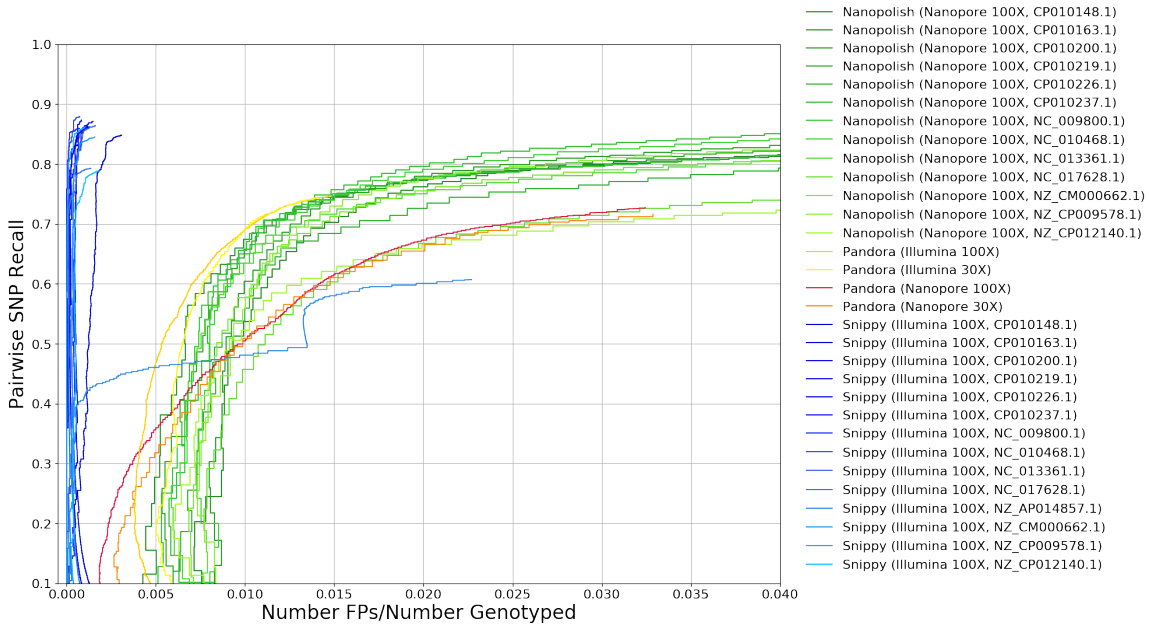


Figure 6.6: *Precision-Recall comparison of methods for a choice of reference genomes with 2 samples.* For a range of genotype confidence thresholds, the y-axis shows the fraction of high confidence SNP differences between samples H131800734 and K12_MG1655 which were captured by each method and the x-axis shows the fraction of VCF calls of at least this confidence which are false positives.

absent from each reference, rendering variation invisible. By contrast, the recall with `Pandora` remains at more than 70%. We were unable to compare with `Nanopolish` since we did not have signal level Nanopore data, but would expect it to have a similar drop in recall to `Snippy`, because it was calling against the same reference panel.

When the 4th sample RHB10-C07 is added to the comparison, we see another noticeable drop in recall when genotyping with `Snippy` as further shared sequence is absent from the reference. For most choices of reference, recall is now ≈20-30%, with a maximum recall of 50%. By comparison, calls made with `Pandora` again continue to achieve a maximum recall of 75.5/74.2% with 100X/30X Illumina data, and 74.3/73.1% with 100X/30X Nanopore. At 70% recall, the fraction of false positives with `Pandora` was 0.0244 with 100X Nanopore data and 0.0113 with 100X Illumina.

Table 6.1 includes the run times and memory requirements for a single successful genotyping run on one or all of these samples. For Nanopore data, `Pandora` is able to compare the 4 samples in an order of magnitude less time than `nanopolish`, using a comparable amount of memory (assuming the 4 samples are run in parallel for `nanopolish`). This represents a significant improvement. Whilst it is possible that providing more sequence data would further improve the false positive rate of calls

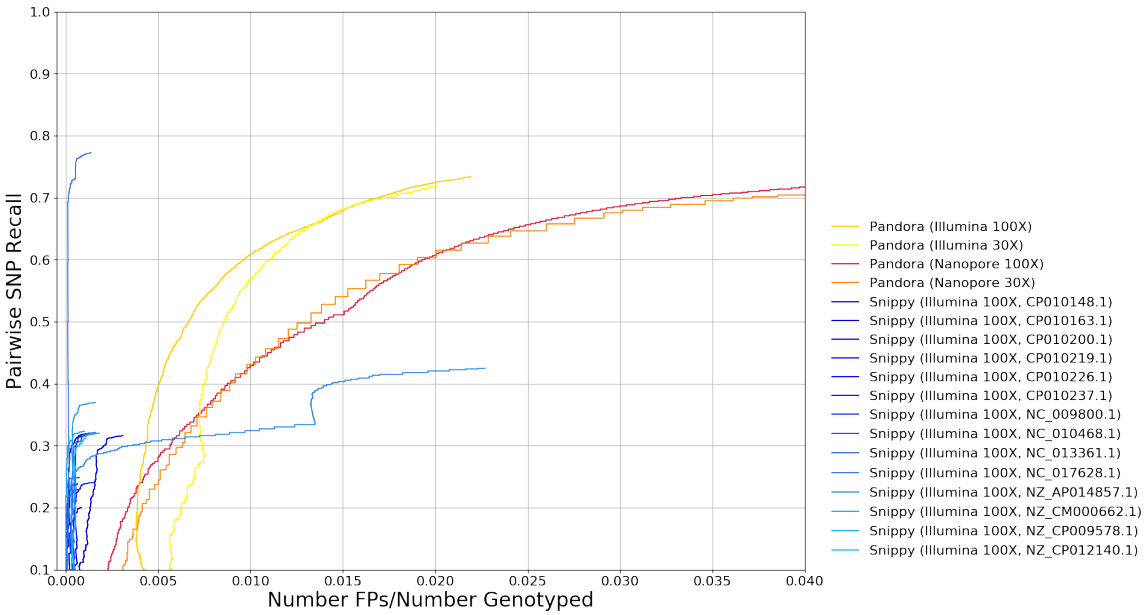


Figure 6.7: *Precision-Recall comparison of methods for a choice of reference genomes with 3 samples.* For a range of genotype confidence thresholds, the y-axis shows the fraction of all high confidence SNP differences between pairs of samples H131800734, K12_MG1655 and CFT073 which were captured by each method and the x-axis shows the fraction of VCF calls of at least this confidence which are false positives.

with `nanopolish`, we note that when this analysis was previously attempted with 300X sequence data per sample rather than 100X, each `Nanopolish` genotyping run required almost 200 CPU hours to complete.

6.5 Discussion

In this chapter we described methods to use the PanRG as a substrate to describe variation between sets of genomes. This allows us to describe both variation at the coarse scale of gene presence/absence, and at the fine scale of SNPs and indels across the

Genotyper	Data	Covg	CPU time (h)	Peak RAM (GB)	# samples
Snippy	Illumina	100	0.283	1.79	1
Nanopolish	Nanopore	100	20.9	3.86	1
Pandora	Illumina	30	5.63	38.5	4
Pandora	Nanopore	30	2.70	25.3	4

Table 6.1: CPU time and memory requirement of a single successful genotyping run. For `Snippy` and `Nanopolish` the time or memory requirement needs to be multiplied by the number of samples for comparison.

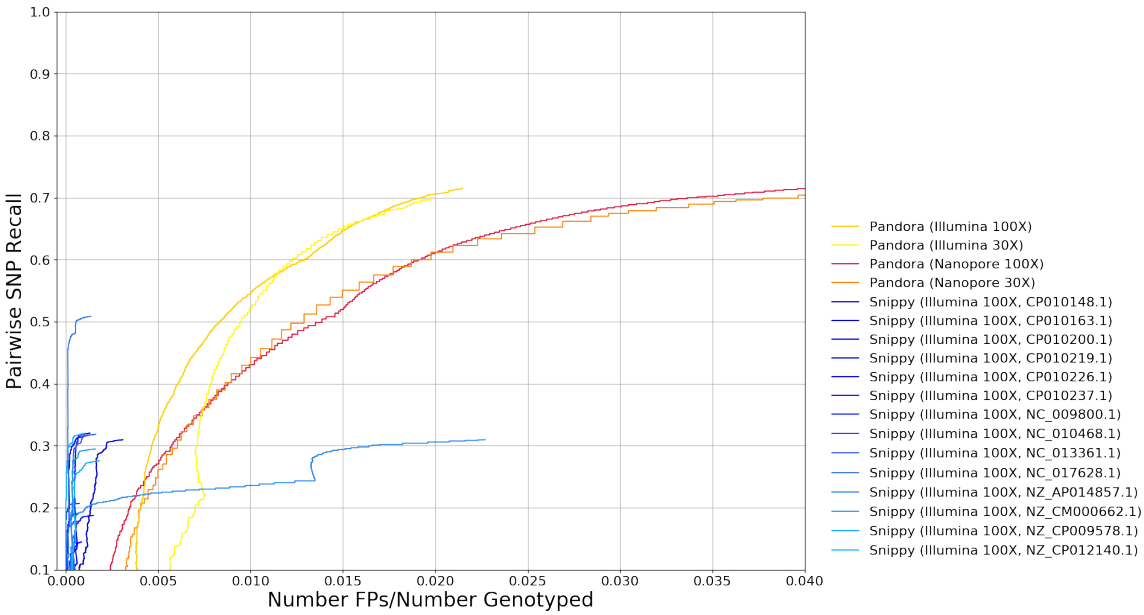


Figure 6.8: *Precision-Recall comparison of methods for a choice of reference genomes with 4 samples.* For a range of genotype confidence thresholds, the y-axis shows the fraction of all high confidence SNP differences between pairs of samples H131800734, K12_MG1655, CFT073 and RHB10-C07 which were captured by each method and the x-axis shows the fraction of VCF calls of at least this confidence which are false positives.

pangenome, facilitating comparison of diverse collections of genomes. We have demonstrated that by enabling SNP genotyping in the accessory part of the pangenome we are able to maintain a high pairwise SNP recall of 70-75% as the number of divergent samples increases. In the same context, we have shown that single-reference genotypers progressively fail to describe this variation. We have demonstrated that genotyping is possible with as little as 30X Nanopore sequence data per sample, and that we can do this an order of magnitude faster than the only published alternative, `Nanopolish`.

6.5.1 When is there a 'best' reference?

For the large number of bacterial and viral species where there is considerable structural variation, the question of how to choose a best reference to enable comparison is a hard one.

We have demonstrated that for reference-based methods, the choice of reference genome affects both the precision and recall of the resulting variation analysis. For this reason, it is increasingly common when evaluating clonal outbreaks to construct an 'outbreak reference' by sequencing a single sample and assembling e.g.[Haley et al.,

2016], [van der Graaf-van Bloois et al., 2016] and [Decraene et al., 2018].

To do this requires preliminary analysis of the samples to decide which are part of the clonal outbreak, and potentially a time cost while a new 'reference' is sequenced and assembled. In the process there is a risk of eliminating samples which seem unrelated, but where the accessory or mobile genome is in fact related.

For many other situations, the 'best reference' is just the reference which is closest on average to samples, evaluated with preliminary analysis using e.g. MASH [Ondov et al., 2016]. For many other studies still, only a single standard reference is ever considered.

We make the case that in the majority of these situations, now that we have such an extensive database of reference assemblies for many species, the best reference is in fact a pangenome reference.

6.5.2 Bacterial evolution and phylogenetics

We know that in bacteria like *E. coli*, evolution occurs as a result of multiple competing processes, each altering the genome on different scales. Commonly we approximate the evolutionary history of samples by looking at nucleotide level differences (in the core genome), which enables us to construct trees. Whilst a tree is a good model for vertical inheritance, it cannot capture the full picture. It is not within the scope of this thesis to address the issue of how we can instead represent the history of a sample in such a way as to describe both vertically and horizontally inherited material in an intelligible manner (a very complicated ARG, perhaps). Instead this work aims to make all variation systematically available for such an analysis.

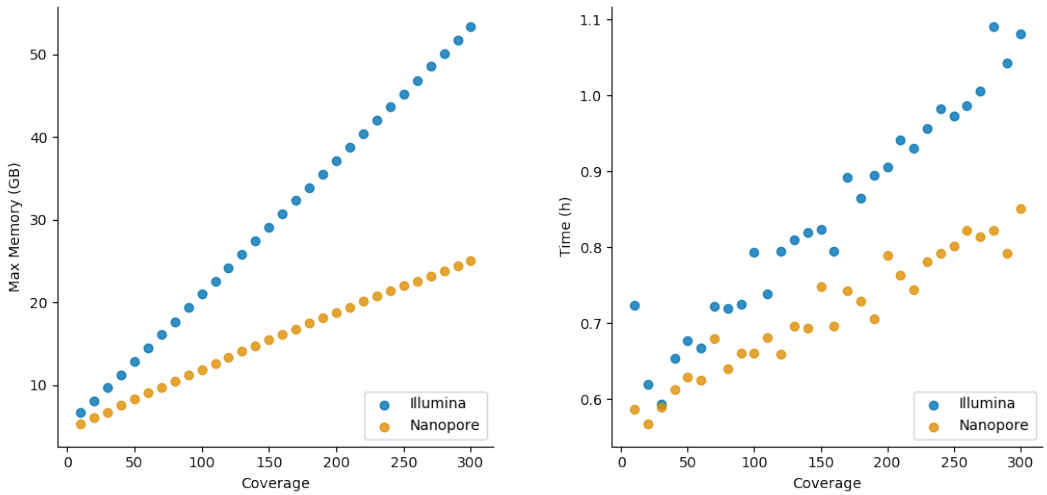
6.5.3 Comparison of samples from different read technologies

It would be possible to use `Pandora` to compare samples using reads generated on different sequencing platforms, provided a single PanRG index is used. We demonstrated in Chapter 4 that the optimal parameter setting vary slightly for different sequencing technologies. It would be possible to allow the use of different presets when defining and filtering clusters (e.g. for Illumina and Nanopore data), the index parameters of w and k would remain fixed for the analysis. Therefore, as is the case with all variant calling methods, caution must be used when considering the results, bearing in mind that the precision and recall from reads of each platform are likely to be different.

6.6 Limitations

6.6.1 Scaling

Whilst `Pandora` is extremely fast by comparison with assembly, at the time of writing this thesis it has not been optimised for memory consumption.

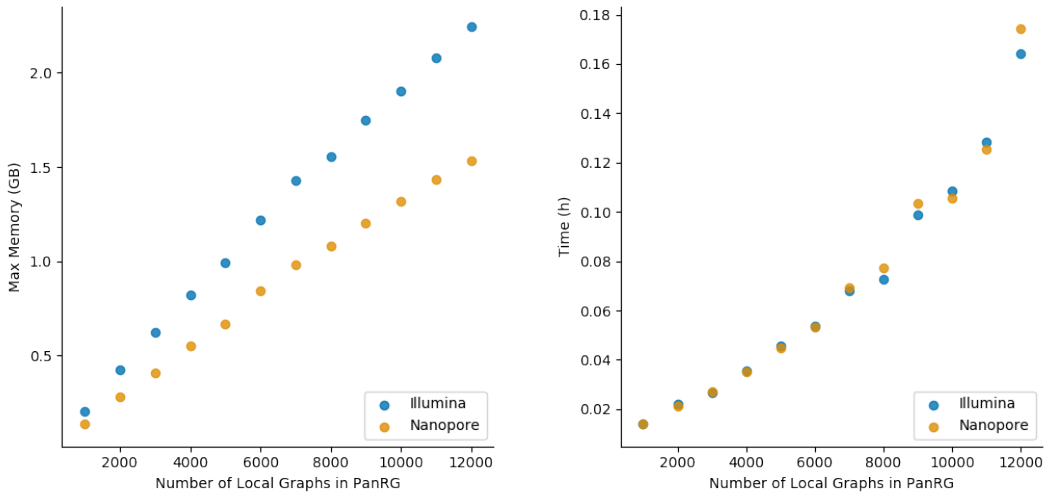


(a) Max memory vs read coverage (b) CPU time vs read coverage

Figure 6.9: Maximum memory and CPU time scale linearly with read coverage. We subsampled the read datasets from Section 4.4.6 and used `pandora map` to infer mosaic sequences against our full *E. coli* PanRG. Time and memory requirements were measured using the inbuilt Unix `time` utility.

Figure 6.9 shows that the maximum memory used by `Pandora` to map a set of reads to the PanRG scales linearly with genome coverage by reads, and the time approximately linearly. Due to the error rate in Nanopore data, a larger proportion of read data is discarded as noise during the initial mapping phase and so the memory and time requirements of `Pandora` to map these reads are both lower than for Illumina data.

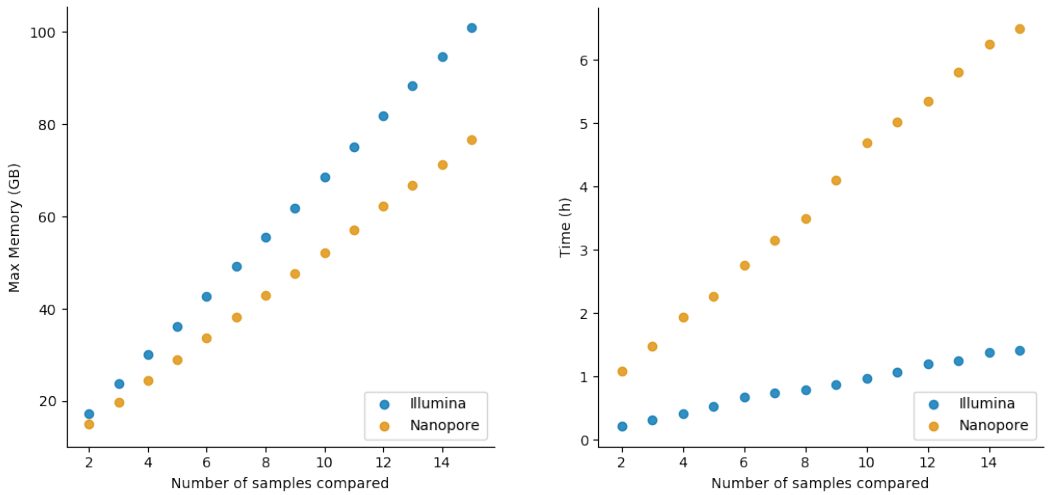
Figure 6.10 shows a similar linear relationship between the maximum memory used by `Pandora` to map a set of reads to the PanRG and the number of local graphs in the PanRG. However CPU time scales more quadratically. As the number of local graphs in the PanRG increases, there are more pairs of local graphs with intersecting sets of k-mers in the index leading to more false positive clusters of hits being found between reads and the PanRG. After the mosaic sequence is inferred, many of these false positive clusters are filtered out, but until this point the memory requirement will be higher.



(a) Max memory vs number of local graphs in PanRG

(b) CPU time vs number of local graphs in PanRG

Figure 6.10: Maximum memory scales linearly, and CPU time scales greater than linearly with number of local graphs in PanRG. We subsampled the PanRG from Section 3.5 and used `pandora map` to infer mosaic sequences for 30X of the read datasets described in Section 4.4.6 . Time and memory requirements were measured using the inbuilt Unix `time` utility.



(a) Max memory vs number samples compared

(b) CPU time vs number of samples compared

Figure 6.11: Maximum memory and CPU time scale linearly with number of samples. We ran `pandora compare` with the same PanRG described in Section 3.5 on increasing numbers of the samples described in Section 6.3.2.1, using a maximum of 30X coverage from each read dataset. Time and memory requirements were measured using the inbuilt Unix `time` utility.

Figure 6.11 shows a linear relationship between both the maximum memory and CPU time used by `pandora compare` and the number of samples being compared. Since fewer hits are discarded as noise, the maximum memory requirement for Illumina sequence data is higher than for the same coverage of Nanopore sequence data. However the time taken by Nanopore datasets increases more rapidly with the number of samples. It is likely that the added noise in the Nanopore reads results in more time being spent copying noise coverage data between data structures, and filtering results.

It is clear from this graph, that such a large memory requirement prohibits comparison of many genomes. In the short term this has been worked around by partitioning the PanRG, and using Nextflow to run analysis for each partition in parallel and combine results at the end. Further profiling of `Pandora` could be used to reduce the memory requirements and in particular to improve the data structures used to store hits and kmer coverages. There also are many steps within `Pandora` where the same operation is performed on each local graph separately, or on each sample separately and with further development these could be parallelised within `Pandora` so that the large memory requirement can be distributed over several threads or CPUs.

we briefly mention in Section 6.6.2.1, the performance improvements that have been made by others since this work to solve the problems identified here.

6.6.2 *De novo* discovery of variants

One major limitation of the methods described in this work is the fact that only variants captured by the PanRG can be called. Given that bacteria are evolving new mutations all the time, the result is that no matter how large the input panel of reference genomes is, over time we expect this PanRG to become outdated.

To address this limitation, another PhD student, Michael Hall has been working on adding *de novo* variant calling to `Pandora`. Regions along the inferred mosaic sequence with poor support from reads are flagged and slices of reads corresponding to each of these regions (plus padding) are extracted. In each region a de Bruijn graph is created with GATB [Drezen et al., 2014] using a smaller k-mer size k_d and requiring 2X coverage of a node for inclusion in the graph.

A path start k_d-mer is generated by successively querying the first k_d k_d-mers from the original padded slice in the inferred mosaic sequence in the de Bruijn graph until one is found. In the same way a path end k_d-mer is generated working from the end of the slice in the inferred mosaic sequence. If both a start and end k_d-mer exist in the de Bruijn graph, a depth first search is performed to find paths between

them. These paths are enumerated by walking the depth first search tree, killing walks if they become too long (∼70bps) or if more than k_d nodes on the path have less than 10% the expected coverage. If too many paths (∼ 30) are found, a stricter requirement for the coverage on nodes is enforced. The filters at each step ensure that the method can complete with reasonable time and memory requirements, and that the graph does not become oversaturated with improbable alleles.

The initial code to identify and extract read slices was written collaboratively by Michael, Robyn and I. Since then, Michael has fully implemented this method in C++ in `Pandora` and is currently in the process of testing different parameter choices.

6.6.2.1 Performance improvements to `Pandora` subsequent to this thesis

Subsequent to the work described in this thesis, the performance of `Pandora` has been drastically improved by Leandro Ishi Soares De Lima. He has addressed many of the redundancies identified in this thesis and more. Namely, he has implemented memoization during indexing, improved some data structures and added multi-threading capabilities. At the point of submission the command `pandora index -w 19 -k 31 -t 16 <PanRG>` required a total of 19.8 GB and 1 hour to reindex the *E. coli* PanRG used throughout this thesis (down from more than a week). Genotyping a single sample with `pandora map` now typically takes 35 minutes and requires 10.9 GB RAM. The comparison of 151 *K. pneumoniae* samples (each with 45X Illumina data) with `pandora compare` as considered in the next chapter now requires a total of 13.5 GB and completes in just over 5 hours on 16 threads.

Chapter 7

Applications

The primary purpose of the PanRG data structure and the `Pandora` software package is to enable comparison of diverse sets of bacterial genomes, including base-level resolution across the pangenome and from Nanopore reads. In this section we will apply `Pandora` to investigate a number of questions. Firstly we investigate whether the accessory genome can add resolution the analysis of an outbreak dataset. Secondly we apply our methods to investigate the site frequency spectrum outside of the core genome. Finally we apply our methods to investigate the population structure in *K. pneumoniae*.

7.1 Case Study 1: Outbreak of carbapenem resistant *E. coli*

7.1.1 Introduction

In Section 6.3.2.1 we used samples from a hospital outbreak published in [Decraene et al., 2018]. This study initially included samples from the clonal *E. coli* outbreak which was concentrated in 2 wards, and was later extended to include additional carbapenem resistant *E. coli* collected over an 8 year period across the the wider Central Manchester University Hospital NHS Foundation Trust (CMFT). The paper described the effects of different measures implemented to control the outbreak, as well as describing how the samples in the outbreak were related to the additional environmental and contextual samples.

The software `Pandora` both enables genotyping from Nanopore sequence data, and provides a framework for calling variation in the accessory part of the pangenome. For a subset of 16 closely related samples from the clonal outbreak for which we have Nanopore sequence data, we therefore first compare the phylogenies constructed

based on core SNPs from Illumina or Nanopore sequence data are equivalent, then show how the relationship between samples varies when SNPs within accessory genes are included.

7.1.2 Methods

7.1.2.1 Data

The Nanopore and PacBio sequence data were generated as described in section 6.3.2.1 and the Illumina data was generated as described in [Decraene et al., 2018]. Since Illumina read files are sometimes sorted, we randomly downsampled 45X coverage of reads to input to `Pandora` using a python script.

7.1.2.2 Running `Pandora`

We used `Pandora` to compare the 16 samples separately using Nanopore data, and Illumina data as follows:

```
singularity exec shub://rmcolq/pandora:pandora pandora index -w 19 -k 31 <PanRG>
singularity exec shub://rmcolq/pandora:pandora pandora compare
-p <PanRG>
-w 19 -k 31
-r <illumina_read_index>
--genotype
--illumina

singularity exec shub://rmcolq/pandora:pandora pandora index -w 14 -k 15 <PanRG>
singularity exec shub://rmcolq/pandora:pandora pandora compare
-p <PanRG>
-w 14 -k 15
-r <nanopore_read_index>
--genotype
--max_covg 40
```

7.1.2.3 Processing results and generating trees

Genes were categorized as 'core' or 'accessory' based on the pangenome matrix describing their presence or absence in each sample genome. Those loci present in all 16 samples were identified as 'core' and all others 'accessory'.

For each category of genes, relevant records in the multisample VCF were identified. These records were filtered based on allele lengths (restricting to only SNPs). An alignment was constructed for each sample by concatenating the alleles called in that sample. For each record, each alignment sequence was updated with the allele called in that sample, or 'N' if no call was made for that sample.

We used FastTree [Price et al., 2009], [Price et al., 2010] to construct generalized time-reversible maximum-likelihood trees for each alignment.

```
fasttree -gtr -nt <alignment_file> > <tree_file>
```

In R, alignment-based trees were loaded and rooted using a known outgroup (H151080744) with package `ape`.

```
load_my_tree <- function(tree_file, outgroup="H151080744"){
    tree <- read.tree(file=tree_file)
    tree <- root(tree, outgroup=outgroup, resolve.root = TRUE)
    dend <- chronos(tree)
    dend <- as.dendrogram(dend)
    return(dend)
}
```

The pangenome matrix file was hierarchically clustered using packages `cluster`, `tidyverse` and `factoextra` and used to construct a tree based on the presence or absence of accessory elements.

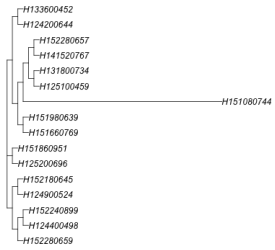
```
load_my_matrix <- function(df_file){
    df <- read.csv(df_file, row.names=1, header=TRUE, strip.white = TRUE)
    dft <- as.data.frame(t(df))
    res.dist <- dist(dft, method = "euclidean")
    hc <- hclust(res.dist, method = "ward.D2" )
    dend <- as.dendrogram (hc)
    return(dend)
}
```

We used package `dendextend` to construct tanglegrams for pairs of trees, colouring labels on both trees based on groups defined by cutting the left hand tree.

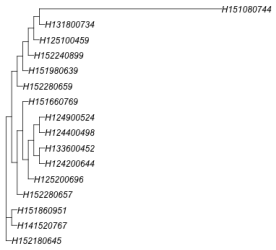
```
facing_trees <- function(left_tree, right_tree, k=5){
    left_tree_a <- color_labels(left_tree, groupLabels=TRUE, k=k)
    left_cols <- labels_colors(left_tree_a,labels=TRUE)
    left_labels <- labels(left_tree)
    right_labels <- labels(right_tree)
    right_cols <-  left_cols[order(left_labels)][rank(right_labels)]
    right_tree_a <- color_labels(right_tree, col=right_cols)

    dl <- dendlist(left_tree_a, right_tree_a)

    tanglegram(dl,
           common_subtrees_color_lines = FALSE,
           common_subtrees_color_branches = FALSE,
           highlight_distinct_edges  = TRUE,
           highlight_branches_lwd=FALSE,
           margin_inner=7, lwd=2)
}
```

All work processing the output files from `Pandora` were completed in a jupyter notebook. We used a minimal threshold for genotype confidence, set to 10 for the results with Illumina sequence data, and 20 with Nanopore. We included only SNP sites which had been called in at least 80% of the samples in the core, or at least 20% of samples when considering both core and accessory genes.

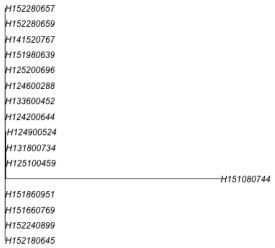
7.1.3 Results



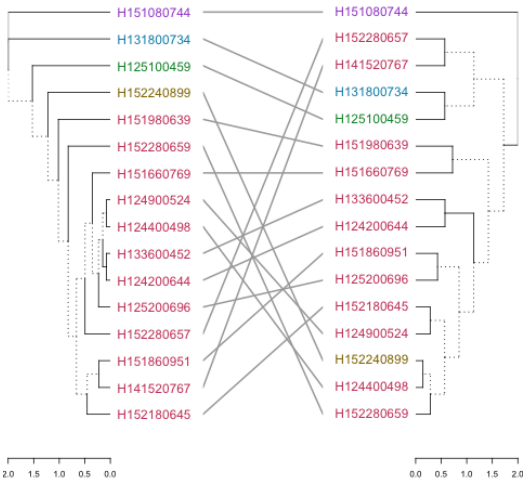
(a) Phylogenetic tree based on core SNPs identified from Illumina data.



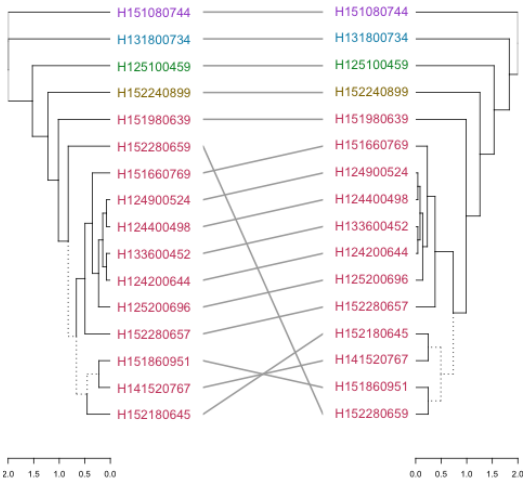
(b) Phylogenetic tree based on core SNPs identified from Nanopore data.



(c) Phylogenetic tree from [Decraene et al., 2018], downsampled in dendroscope.



(d) Phylogenetic trees based on core SNPs identified from Illumina data (left) and Nanopore data (right).



(e) Phylogenetic trees based on core SNPs (left) and all SNPs (right) identified from Illumina data.

Figure 7.1: *Comparison of phylogenetic trees constructed with* ***`FastTree`*** *based on SNPs identified with* ***`Pandora`*** *from either Illumina or Nanopore sequence data.*

We first compared the core SNP trees identified by `Pandora` to the tree constructed during the initial analysis of the published outbreak. Figures 7.1a and 7.1b show the maximum likelihood phylogenetic tree generated from the `Pandora` results with Illumina and Nanopore data respectively. Figure 7.1c shows the outbreak tree constructed by Hang Phan, downsampled to our subset in Dendroscope [Huson and Scornavacca, 2012]. All three trees contain a single outlying sample, H151080744. In the analysis in [Decraene et al., 2018], this was sample was identified as belonging to a different clade to the remaining samples in this dataset, as a result of a 1Mb recombination event. They identified that the remaining samples were extremely closely related ($\leq$65 SNPs separating the 112 isolates of which these are a subset). We note that the trees constructed from `Pandora` VCFs show more discrimination between the remaining samples. This is possibly because a smaller set of genes were core to the larger set of samples, but could also potentially indicate that `Pandora` found variation in genes that were absent from the reference genome used for the outbreak analysis but common to samples.

Next we compared the two core SNP trees constructed from `Pandora` VCFs with Illumina and Nanopore data. The tanglegram in Figure 7.1d shows that, aside from the outlier, the remainder of each `Pandora` phylogenetic tree looks different with each technology. Part of this variation can be explained as a consequence of slight variation in the genes identified as present with each type of data. From the Illumina data, `Pandora` identified 5298 core genes, and 1258 accessory genes, by comparison with 5178 core genes and 1643 accessory genes from Nanopore data. Given how closely related these samples are, it is likely to also reflect the different precision and recall rates in each technology. We know that we have a higher rate of false positives when using Nanopore data from the analysis in the previous chapter.

Finally, we compared the tree inferred from core SNPs to the tree inferred from all SNPs as shown in Figure 7.1e for Illumina data. Extending from core SNPs to all SNPs added a further 717 bases to each sample SNP alignment, extending from 107007bps to 107724bps in length (note that `Pandora` includes reference calls, and we included all PanRG SNP sites in analysis, even when they were non-segregating). We found that the topology of the tree with the extended panel of SNPs was well preserved when additional sites were included. However the additional bases resulted in improved resolution at the deepest level of the tree, with a slight rearrangement of 4 samples.

7.1.4 Discussion

In this section we used `Pandora` to describe variation between 16 closely related samples from an outbreak. We showed that the inclusion of SNP variation in a larger panel of core genes added resolution to the tree by comparison with the original downsampled outbreak tree. Further, we showed that the inclusion of additional SNP variation within the accessory genome added resolution to the tree structure at the deepest level of the tree.

We also demonstrated that in this context, where samples are all very closely related, the topology of inferred trees is very sensitive to the data included. In it's current state, `Pandora` has a moderate false positive rate for both SNP and gene presence/absence calls with both technologies, but especially with Nanopore, and this was reflected in the different tree topologies inferred with each. With hindsight, evaluating our methods with such a discriminating dataset was a poor choice.

We have shown that the additional accessory SNP or indel information made accessible with `Pandora` could be included in phylogenetic analyses of such datasets. However, given pangenome-wide single nucleotide resolution variation of a less related dataset, a single phylogenetic tree will no longer be sufficient to describe the complex genetic relationships that exist between samples. One possible way to explore a larger and more diverged dataset would be to restrict downstream analysis of the output VCF to a smaller subset of genes for the full dataset, and larger subsets when considering specific clades, or sequence types found. This approach of 'zooming in/out' is already used for surveillance datasets such as [Gorrie et al., 2017], but currently requires many steps classifying samples into sequence types, and performing separate variant calling with lineage specific references for each cluster. The next problem to be solved is how best to describe these relationships (ideally pictorially) in such a way that is both biologically accurate and enables useful inferences to be made.

7.2 Case Study 2: Site Frequency Spectrum in accessory genes

7.2.1 Introduction

Variation within a collection of genomes is often used to infer the ancestral relationships between those genomes as well as the nature of evolution within the species. In both of these cases, models are used to describe how such variation arises in the

population. Estimation of the model parameters from a dataset allows us to either construct a tree with appropriate branch lengths, or to study the rates of mutation, genetic drift, selection and recombination within the species.

In [Baumdicker et al., 2010], a model was introduced to describe the gene frequency spectrum in the pangenome and results were shown to provide a realistic fit for a small set of 9 *Prochlorococcus* genomes despite simplifying assumptions. In [Baumdicker, 2015], an extension of this work described a model for the site frequency spectrum outside of the core genome and proved that in this context, both Watterson's and Tajima's estimators for mutation rate are negatively biased. However, the model was not demonstrated to be realistic on a dataset, perhaps due to the difficulty of calling variation outside of the core genome.

`Pandora` provides a framework in which variation can be described across the pangenome. In this section there therefore use a datset of 151 diverse *Klebsiella pneumoniae*, to investigate whether there is evidence to support this model.

7.2.2 Background

7.2.2.1 The Kingman coalescent

The Kingman coalescent (or n-coalescent) defined in [Kingman, 1982] is a continuous time Markov process $\{R_t : t \geq 0\}$ on Π_n, the set of partitions of $\{1, ..., n\}$ with infinitesimal generator $Q = \{q_{\xi\eta} : \eta \in \Pi_n\}$ defined such that

$$q_{\xi\eta} = \begin{cases} -\frac{k(k-1)}{2}, & \xi = \eta \\ 1, & \xi \prec \eta \\ 0, & \text{otherwise} \end{cases}$$

where $k = \|\xi\|$ and $\xi \prec \eta$ if and only if η is obtained by combining two equivalence classes in the partition ξ.

It describes the construction of a binary tree, starting with n leaf nodes and going backwards in time. Branch lengths represent waiting times between coalescence events and are exponentially distributed with rate $\binom{k}{2}$. At a coalescence event, 2 lineages are picked uniformly and merged.

7.2.2.2 Modelling the gene frequency spectrum for the pangenome of a species

Baumdicker et al. [2010] introduced a model to describe the gene frequency spectrum seen in the pangenome. They divide the pangenome into the *core* genes which are essential for survival and are always conserved, and the *dispensable* genes which can

be gained and lost. These definitions allow some dispensable genes to appear in every genome within a sample from the population and hence differ subtly from the definitions of the core and accessory components of the pangenome seen in previous sections.

Reproduction is assumed to follow a neutral Wright-Fisher model with fixed population of size N. Before reproduction of an individual, 'mutation' is allowed in the form of *gene gain*, where with probability μ a new gene (never before seen) is taken up from the environment, and *gene loss*, where each gene in the dispensable genome of an individual is lost with probability ν.

After a time re-scaling by N, the genealogy of a sample of size n under the Wright-Fisher model converges to the Kingman coalescent started with n lines e.g. [Durrett, 2008]. Setting $\theta_1 = \lim_{N\to\infty} 2N\mu$ and $\rho = \lim_{N\to\infty} 2N\nu$, genes are gained in the population at rate $\theta_1/2$ and lost at rate $\rho/2$. This *infinitely many genes* model can be considered to be equivalent to the *infinitely many sites* when $\rho = 0$.

For dispensable genome $\mathcal{G}$, and n sample genomes $B_1, ..., B_n$, the gene frequency spectrum $\{G_1, ..., G_n\}$ is defined as

$$G_k = \|\{g \in \mathcal{G} : g \in B_i \text{ for exactly } k \text{ different } i\}\|.$$

Under this model, Baumdicker et al. proved that the expected values of the gene frequency spectrum are

$$\mathrm{E}[G_k] = \frac{\theta_1}{\rho}\frac{n\cdots(n-k+1)}{(n-1+\rho)\cdots(n-k+\rho)}, \qquad k = 1, ..., n.$$

7.2.2.3 Modelling the site frequency spectrum of the dispensable genome of a species

For an essential gene (the core genome) C, the classical population genetics model assumes that the ancestry of the gene in n sampled genomes can be modelled with a coalescent tree. Mutations occur at rate $\theta_2/2$ along the branches of this tree. As a result, the expected number of mutated sites present in exactly $s = 1, .., n-1$ out of the n genomes is given by

$$\mathrm{E}[C_s] = \frac{\theta_2}{s}.$$

Building on their gene frequency spectrum result [Baumdicker, 2015] describe the joint gene and site frequency spectrum in the dispensable genome. Suppose as before that genes are gained at rate $\theta_1/2$ and lost with rate $\rho/2$. Let $G_{k,s}$ describe the number

of mutated sites occurring in exactly s gene sequences, while the corresponding gene is present in exactly k out of n genomes. The expectation of $G_{k,s}$ for $k < s$ is

$$\mathrm{E}[G_{k,s}] = \frac{\theta_1}{k}\frac{(n-k+1)\cdots n}{(n-k+\rho)\cdots(n-1+\rho)}\frac{\theta_2}{s}\frac{k}{n}\binom{n-1}{s}^{-1}\sum_{j=0}^{n-s-1}\frac{j+1}{j+1+\rho}\binom{n-j-2}{s-1}$$

and for $k = s$

$$\mathrm{E}[G_{k,s}] = \frac{\theta_1}{k}\frac{(n-k+1)\cdots n}{(n-k+\rho)\cdots(n-1+\rho)}\frac{\theta_2}{s}sk\sum_{j=1}^{n-s+1}\frac{1}{j(j-1+\rho)}\binom{n}{j}^{-1}\binom{n-k}{j-1}.$$

For gene g, let $F(g)$ be the number of individuals which contain g out of the n genomes. Let S_s^g be the number of mutated sites in gene g present in exactly s genomes. The conditional site frequency spectrum for gene g, present in exactly k genomes out of n with $s < k$ is:

$$\mathrm{E}[S_s^g|F(g)=k] = \frac{\theta_2}{s}\frac{k}{n}\binom{n-1}{s}^{-1}\sum_{j=0}^{n-s-1}\frac{j+1}{j+1+\rho}\binom{n-j-2}{s-1}$$

and for $k = s$

$$\mathrm{E}[S_s^g|F(g)=k] = \frac{\theta_2}{s}sk\sum_{j=1}^{n-s+1}\frac{1}{j(j-1+\rho)}\binom{n}{j}^{-1}\binom{n-k}{j-1}.$$

As an aside, we note that there is a standard simplifying assumption in population genetics that a genome can be represented by the interval $[0, 1]$, and under the infinitely many sites model a mutation occurs at a position along this interval that has never before been mutated. Baumdicker therefore models each gene as an interval $(0, 1]$, and so his model does not include a parameter to control for the length of a gene. Instead, it models mutations as occurring at some rate dependant on the time that the gene is in the population for.

7.2.3 Methods

7.2.3.1 Dataset

We made use of 151 samples of from a study into gastrointestinal carriage and infection with *K. pneumoniae* in an at-risk cohort at an intensive care unit [Gorrie et al., 2017]. This dataset contains a lot of diversity, including more than 20 characterized sequence types.

7.2.3.2 PanRG construction

We make use of the dataset of 285 annotated *K. pneumoniae* genome assemblies from [Holt et al., 2015]. Orthologous gene clusters were identified with Roary [Page et al., 2015] and intergenic regions identified were then clustered using Piggy [Thorpe et al., 2018] with min percentage length identity of $len\_id = 10$. Multiple sequence alignments were then generated for each cluster with MAFFT [Katoh and Standley, 2013]. This work was done by Harry Thorpe.

We constructed a local graph from each MSA in parallel using the `Nextflow` [Di Tommaso et al., 2017] pipeline available at `https://github.com/rmcolq/make_prg` with default parameters.

7.2.3.3 Variant calling with Pandora

We ran `pandora index` and `compare` in parallel over chunks of 4000 genes and merged the resulting pangenome matrix and VCF files to get a single result.

7.2.3.4 Finding the Gene Frequency Spectrum

We used the pangenome matrix output by `Pandora` to count the frequency of each orthologous gene cluster from the PanRG in the samples.

7.2.3.5 Finding the Site Frequency Spectrum

For genes at each frequency within the dataset, we probed the VCF for biallelic segregating SNPs in these genes, and counted how many samples each allele was present in. For gene g which is present in exactly k samples, we do not know the ancestral allele at any given site and so cannot distinguish between SNP sites at frequency s and at frequency $k - s$ in the population. We therefore combine these and for $s = 1, ..., \lfloor k/2 \rfloor$ we define

$$\tilde{S_s^g} = \begin{cases} S_s^g + S_{k-s}^g, & s < k/2 \\ S_s^g, & s = k/2. \end{cases}$$

From these counts, we could summarize the mean number of SNP sites with an allele present in s or $k - s$ samples over genes which were present in exactly k samples out

of n. Note that

$$\begin{aligned}
&\mathrm{E}\Big[\frac{1}{\|\{g \in \mathcal{G} : F(g) = k\}\|} \sum_{g \in \mathcal{G}: F(g)=k} \tilde{S}_s^g\Big] \\
&= \frac{1}{\|\{g \in \mathcal{G} : F(g) = k\}\|} \sum_{g \in \mathcal{G}: F(g)=k} \mathrm{E}[\tilde{S}_s^g | F(g) = k] \\
&= \mathrm{E}[\tilde{S}_s^g | F(g) = k].
\end{aligned}$$

We performed a least squares fit of parameters between the observed mean frequencies and this expectation.

7.2.3.6 Estimating model parameters

We first performed a least squares fit between the observed values of G_k and $\mathbb{E}[G_k]$ to estimate model parameters θ_1 and ρ. This was implemented using using the python 'scipy.optimize' module. We plot the observed gene frequency spectrum and the expected gene frequency spectrum using different loss functions defined in the module.

We then used the same method to perform a least squares fit between observed average values of S_s^g over dispensable genes g at frequency k, and $\mathbb{E}[\tilde{S}_s^g | F(g) = k]$, to estimate model parameters θ_2 and ρ. For a range of gene frequencies k, we plot the average site frequencies seen in genes at frequency k and the expected conditional site frequency spectrum with parameters again estimated using different loss functions defined in the module.

In all cases, we constrained parameter estimates to be non-negative.

7.2.4 Results

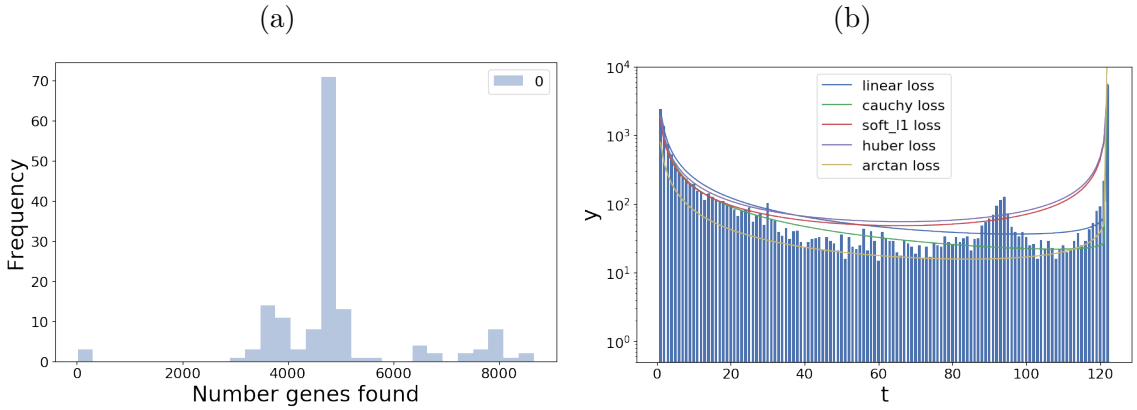


Figure 7.2: (a) Distribution of the number of genes identified in each sample by `Pandora`. (b) Gene frequency spectrum with fits estimated using least squares regression.

Figure 7.2a shows the number of genes identified in each of the 151 samples with `Pandora`. Most fall within the expected range for *K. pneumoniae* of 3500-6000 genes, and the 29 samples which had gene counts falling outside this range were excluded from further analysis.

Figure 7.2b shows the gene frequency spectrum seen in this subset of 122 samples. The distribution has the characteristic U-shape we expect (see Figure 2.2). Least squares parameter fits using each of the loss functions look to have a sensible fit, and estimate the parameters of $\hat{\theta_1}$ and $\hat{\rho}$ as described in Table 7.1. Each of these estimates is of the same order of magnitude, but there is a degree of variation between them

Loss model	$\hat{\theta_1}$	$\hat{\rho}$
Linear	2347	0.7401
Soft L1	1641	0.1469
Huber	1896	0.1591
Cauchy	1786	0.8632
Arctan	784.5	0.5497

Table 7.1: Estimates for θ_1 and ρ parameters using different loss functions in a least squares fit.

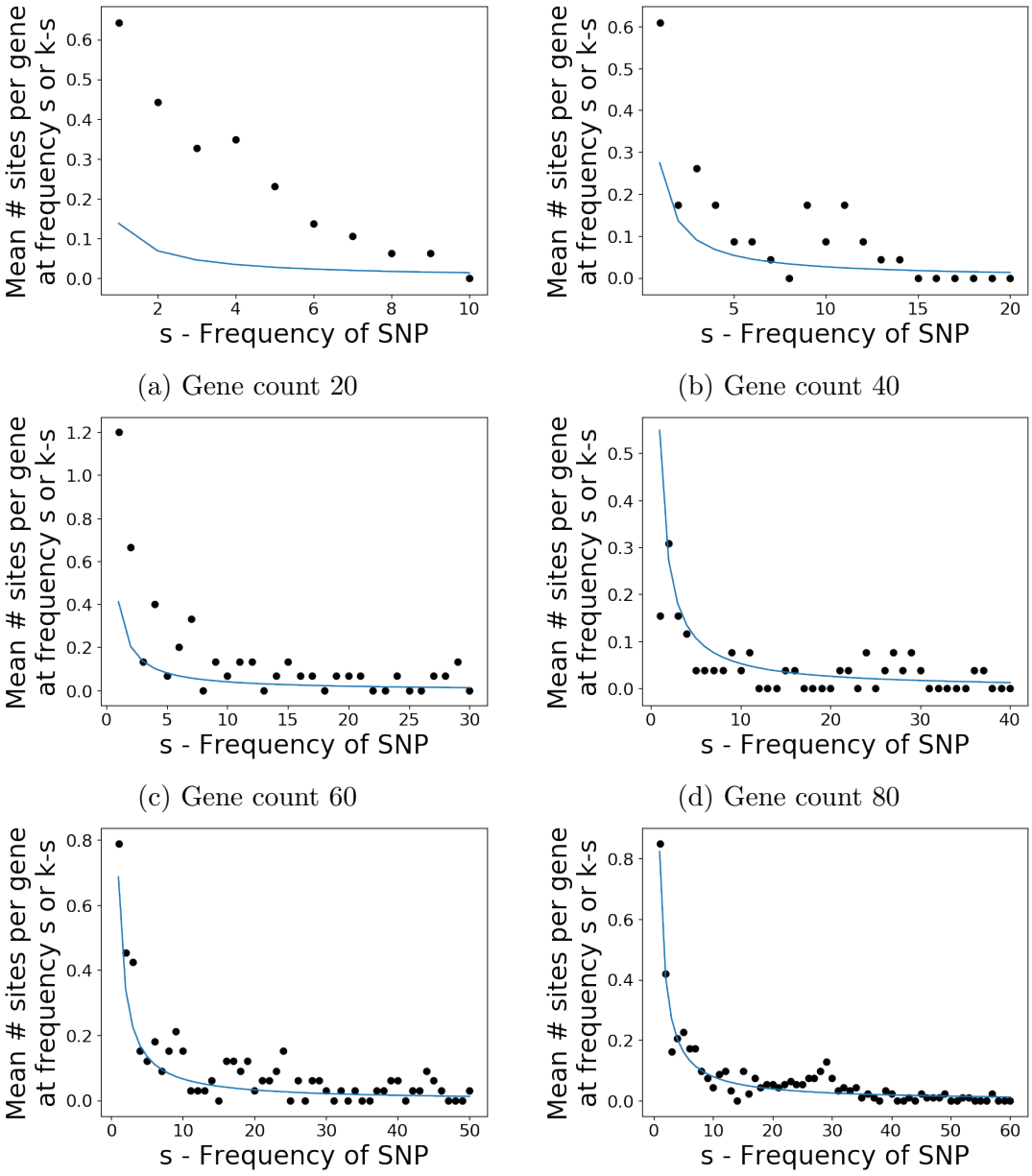
Next we estimate the parameters of ρ and θ_2 based on the site frequency spectrum, with results shown in Table 7.2. These estimates are all much more consistent, with $\hat{\rho} \approx 0.25$, within the range estimated using the gene frequency spectrum, and $\hat{\theta_2} \approx 0.84$.

Loss model	$\hat{\rho}$	$\hat{\theta_2}$
Linear	0.2789	0.8469
Soft L1	0.2528	0.8364
Huber	0.2785	0.8468
Cauchy	0.2299	0.8271
Arctan	0.2239	0.8313

Table 7.2: Estimates for ρ and θ_2 parameters using different loss functions in a least squares fit.

Figure 7.3 shows the fit of the site frequency model using these estimated parameters to the observed mean site frequency spectrum for genes present in 10, 40, 60, 80, 100 and 120 samples within the population of 122. In many cases, the fit seems reasonable, although the observed mean site counts remain noisy. The fit appears less good for genes at lower frequencies. This could be because genes at higher frequencies

contributed more data to the least squares fit. Even if this is the case, it suggests that the model is not explaining all of the site frequency variation we see.



(a) Gene count 20

(b) Gene count 40

(c) Gene count 60

(d) Gene count 80

(e) Gene count 100

(f) Gene count 120

Figure 7.3: *Site frequency spectrum for genes at 6 frequencies within the population.*

7.2.5 Discussion

In this section we investigated whether the site frequency distribution model proposed in [Baumdicker, 2015] was supported by the variation seen in a real population of

Klebsiella pneumoniae. Using parameters estimated from the data, we fit the model and found that it did capture the general shape of the distribution remarkably well for genes at intermediate to high frequencies within the population. It is entirely unsurprising that such a simple model should not be able to explain all of the site frequency variation which we see: it is only by fitting models such as these, we are able to begin to look for signatures of recombination and selection on variation within the dispensable genome. By providing a structured framework in which fine scale variation can be described across the pangenome of a species, `Pandora` provides a starting point from which such further analyses can be performed.

Chapter 8

Conclusion

Comparison of bacterial genomes is fundamental to our understanding of bacterial populations and how they evolve or adapt. As sequencing technologies have become cheaper and more accessible, increasingly larger and more diverse collections of genomes are being sequenced. However, the extensive structural diversity within many species presents a real challenge for variant calling.

In this thesis, we have extended the concept of a population reference graph to bacterial pangenomes. In this way, we have introduced hierarchical framework in which all variation between sets of bacterial genomes can be described. This facilitates the comparison of more diverse collections of genomes by describing nucleotide level variation throughout the pangenome, not just in the core. In particular, it enables variation calling in the accessory genome for the first time. As a result, we have shown that whilst single reference genotypers progressively fail when comparing increasing numbers of diverse genomes, our PanRG approach is able to consistently achieve a considerably higher recall. Our methods are applicable to both Illumina and Nanopore sequence data, and we have shown that they outperform the only published Nanopore variant caller.

We have only briefly touched on the many possibilities that such analyses open up. We have shown that we can call variation in a collection of 151 *K. pneumoniae* genomes collected by routine surveillance during admission to a hospital ward. With this *K. pneumoniae* dataset, we were able to measure the site frequency spectrum in dispensable genes for the first time, and to estimate the parameters for a model proposed in [Baumdicker, 2015]. By fitting such models we can start to look more widely for signatures of selection or for recombination hotspots within species.

As improvements are made to the scalability of our methods, we will be able to perform comparisons of 1000s of possibly unrelated genomes of a species. This could be used for example for surveillance purposes, or to investigate asymptomatic vs

symptomatic enteropathogenic *E. coli*. With this variation accessible and quantified, we will have to find new ways to best interpret and represent the genetic relationships between such diverse genomes.

Bibliography

Alkan, C., Coe, B. P., and Eichler, E. E. (2011). Genome structural variation discovery and genotyping. *Nat. Rev. Genet.*, 12(5):363–376. 16, 17

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. 40

Altshuler, D. M., Durbin, R. M., Abecasis, G. R., Bentley, D. R., Chakravarti, A., Clark, A. G., Donnelly, P., Eichler, E. E., Flicek, P., Gabriel, S. B., Gibbs, R. A., Green, E. D., Hurles, M. E., Knoppers, B. M., Korbel, J. O., Lander, E. S., Lee, C., Lehrach, H., Mardis, E. R., Marth, G. T., McVean, G. A., Nickerson, D. A., Schmidt, J. P., Sherry, S. T., Wang, J., Wilson, R. K., Dinh, H., Kovar, C., Lee, S., Lewis, L., Muzny, D., Reid, J., Wang, M., Fang, X., Guo, X., Jian, M., Jiang, H., Jin, X., Li, G., Li, J., Li, Y., Li, Z., Liu, X., Lu, Y., Ma, X., Su, Z., Tai, S., Tang, M., Wang, B., Wang, G., Wu, H., Wu, R., Yin, Y., Zhang, W., Zhao, J., Zhao, M., Zheng, X., Zhou, Y., Gupta, N., Clarke, L., Leinonen, R., Smith, R. E., Zheng-Bradley, X., Grocock, R., Humphray, S., James, T., Kingsbury, Z., Sudbrak, R., Albrecht, M. W., Amstislavskiy, V. S., Borodina, T. A., Lienhard, M., Mertes, F., Sultan, M., Timmermann, B., Yaspo, M. L., Fulton, L., Fulton, R., Weinstock, G. M., Balasubramaniam, S., Burton, J., Danecek, P., Keane, T. M., Kolb-Kokocinski, A., McCarthy, S., Stalker, J., Quail, M., Davies, C. J., Gollub, J., Webster, T., Wong, B., Zhan, Y., Auton, A., Yu, F., Bainbridge, M., Challis, D., Evani, U. S., Lu, J., Nagaswamy, U., Sabo, A., Wang, Y., Yu, J., Coin, L. J., Fang, L., Li, Q., Li, Z., Lin, H., Liu, B., Luo, R., Qin, N., Shao, H., Wang, B., Xie, Y., Ye, C., Yu, C., Zhang, F., Zheng, H., Zhu, H., Garrison, E. P., Kural, D., Lee, W. P., Fung Leong, W., Ward, A. N., Wu, J., Zhang, M., Griffin, L., Hsieh, C. H., Mills, R. E., Shi, X., Von Grotthuss, M., Zhang, C., Daly, M. J., Depristo, M. A., Banks, E., Bhatia, G., Carneiro, M. O., Del Angel, G., Genovese, G., Handsaker, R. E., Hartl, C., McCarroll, S. A., Nemesh, J. C., Poplin, R. E.,

Schaffner, S. F., Shakir, K., Yoon, S. C., Lihm, J., Makarov, V., Jin, H., Kim, W., Cheol Kim, K., Rausch, T., Beal, K., Cunningham, F., Herrero, J., McLaren, W. M., Ritchie, G. R., Gottipati, S., Keinan, A., Rodriguez-Flores, J. L., Sabeti, P. C., Grossman, S. R., Tabrizi, S., Tariyal, R., Cooper, D. N., Ball, E. V., Stenson, P. D., Barnes, B., Bauer, M., Keira Cheetham, R., Cox, T., Eberle, M., Kahn, S., Murray, L., Peden, J., Shaw, R., Ye, K., Batzer, M. A., Konkel, M. K., Walker, J. A., MacArthur, D. G., Lek, M., Herwig, R., Shriver, M. D., Bustamante, C. D., Byrnes, J. K., De La Vega, F. M., Gravel, S., Kenny, E. E., Kidd, J. M., Maples, B. K., Moreno-Estrada, A., Zakharia, F., Halperin, E., Baran, Y., Craig, D. W., Christoforides, A., Homer, N., Izatt, T., Kurdoglu, A. A., Sinari, S. A., Squire, K., Xiao, C., Sebat, J., Bafna, V., Ye, K., Burchard, E. G., Hernandez, R. D., Gignoux, C. R., Haussler, D., Katzman, S. J., James Kent, W., Howie, B., Ruiz-Linares, A., Dermitzakis, E. T., Lappalainen, T., Devine, S. E., Liu, X., Maroo, A., Tallon, L. J., Rosenfeld, J. A., Michelson, L. P., Min Kang, H., Anderson, P., Angius, A., Bigham, A., Blackwell, T., Busonero, F., Cucca, F., Fuchsberger, C., Jones, C., Jun, G., Li, Y., Lyons, R., Maschio, A., Porcu, E., Reinier, F., Sanna, S., Schlessinger, D., Sidore, C., Tan, A., Kate Trost, M., Awadalla, P., Hodgkinson, A., Lunter, G., Marchini, J. L., Myers, S., Churchhouse, C., Delaneau, O., Gupta-Hinch, A., Iqbal, Z., Mathieson, I., Rimmer, A., Xifara, D. K., Oleksyk, T. K., Fu, Y., Liu, X., Xiong, M., Jorde, L., Witherspoon, D., Xing, J., Browning, B. L., Alkan, C., Hajirasouliha, I., Hormozdiari, F., Ko, A., Sudmant, P. H., Chen, K., Chinwalla, A., Ding, L., Dooling, D., Koboldt, D. C., McLellan, M. D., Wallis, J. W., Wendl, M. C., Zhang, Q., Tyler-Smith, C., Albers, C. A., Ayub, Q., Chen, Y., Coffey, A. J., Colonna, V., Huang, N., Jostins, L., Li, H., Scally, A., Walter, K., Xue, Y., Zhang, Y., Gerstein, M. B., Abyzov, A., Balasubramanian, S., Chen, J., Clarke, D., Fu, Y., Habegger, L., Harmanci, A. O., Jin, M., Khurana, E., Jasmine Mu, X., Sisu, C., Degenhardt, J., Stütz, A. M., Keira Cheetham, R., Church, D., Michaelson, J. J., Blackburne, B., Lindsay, S. J., Ning, Z., Frankish, A., Harrow, J., Mu, X. J., Fowler, G., Hale, W., Kalra, D., Barker, J., Kelman, G., Kulesha, E., Radhakrishnan, R., Roa, A., Smirnov, D., Streeter, I., Toneva, I., Vaughan, B., Ananiev, V., Belaia, Z., Beloslyudtsev, D., Bouk, N., Chen, C., Cohen, R., Cook, C., Garner, J., Hefferon, T., Kimelman, M., Liu, C., Lopez, J., Meric, P., O'Sullivan, C., Ostapchuk, Y., Phan, L., Ponomarov, S., Schneider, V., Shekhtman, E., Sirotkin, K., Slotta, D., Zhang, H., Barnes, K. C., Beiswanger, C., Cai, H., Cao, H., Gharani, N., Henn, B., Jones, D., Kaye, J. S., Kent, A., Kerasidou, A., Mathias, R., Ossorio, P. N., Parker, M., Reich, D., Rotimi, C. N., Royal, C. D., Sandoval, K.,

Su, Y., Tian, Z., Tishkoff, S., Toji, L. H., Via, M., Wang, Y., Yang, H., Yang, L., Zhu, J., Bodmer, W., Bedoya, G., Ming, C. Z., Yang, G., Jia You, C., Peltonen, L., Garcia-Montero, A., Orfao, A., Dutil, J., Martinez-Cruzado, J. C., Brooks, L. D., Felsenfeld, A. L., McEwen, J. E., Clemm, N. C., Duncanson, A., Dunn, M., Guyer, M. S., Peterson, J. L., and Lacroute, P. (2012). An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491(7422):56–65. 21, 33

Angiuoli, S. V. and Salzberg, S. L. (2011). Mugsy: Fast multiple alignment of closely related whole genomes. *Bioinformatics*, 27(3):334–342. 17

Baumdicker, F. (2015). The site frequency spectrum of dispensable genes. *Theor. Popul. Biol.*, 100:13–25. 107, 108, 113, 115

Baumdicker, F., Hess, W. R., and Pfaffelhuber, P. (2010). The diversity of a distributed genome in bacterial populations. *Ann. Appl. Probab.*, 20(5):1567–1606. 107

Berlin, K., Koren, S., Chin, C.-S., Drake, J. P., Landolin, J. M., and Phillippy, A. M. (2015). Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. *Nat. Biotechnol.*, 33(6):623–30. 40

Bertels, F., Silander, O. K., Pachkov, M., Rainey, P. B., and van Nimwegen, E. (2014). Automated Reconstruction of Whole-Genome Phylogenies from Short-Sequence Reads. *Mol. Biol. Evol.*, 31(5):1077–1088. 17, 21

Bradley, P., Gordon, N. C., Walker, T. M., Dunn, L., Heys, S., Huang, B., Earle, S., Pankhurst, L. J., Anson, L., de Cesare, M., Piazza, P., Votintseva, A. A., Golubchik, T., Wilson, D. J., Wyllie, D. H., Diel, R., Niemann, S., Feuerriegel, S., Kohl, T. A., Ismail, N., Omar, S. V., Smith, E. G., Buck, D., McVean, G., Walker, A. S., Peto, T. E. A., Crook, D. W., and Iqbal, Z. (2015). Rapid antibiotic-resistance predictions from genome sequence data for Staphylococcus aureus and Mycobacterium tuberculosis. *Nat. Commun.*, 6:10063. 15

Brinda, K., Callendrello, A., Cowley, L., Charalampous, T., Lee, R. S., MacFadden, D. R., Kucherov, G., O'Grady, J., Baym, M., and Hanage, W. P. (2018). Lineage calling can identify antibiotic resistant clones within minutes. *bioRxiv*. 15

Buchfink, B., Xie, C., and Huson, D. H. (2014). Fast and sensitive protein alignment using DIAMOND. 35

Bush, S. J., Foster, D., Eyre, D. W., Clark, E. L., Maio, N. D., Shaw, L. P., Stoesser, N., Peto, T. E. A., Crook, D. W., and Walker, A. S. (2019). Genomic diversity affects the accuracy of bacterial SNP calling pipelines. *bioRxiv*. 16

Castro-Wallace, S. L., Chiu, C. Y., John, K. K., Stahl, S. E., Rubins, K. H., McIntyre, A. B., Dworkin, J. P., Lupisella, M. L., Smith, D. J., Botkin, D. J., Stephenson, T. A., Juul, S., Turner, D. J., Izquierdo, F., Federman, S., Stryke, D., Somasekar, S., Alexander, N., Yu, G., Mason, C. E., and Burton, A. S. (2017). Nanopore DNA Sequencing and Genome Assembly on the International Space Station. *Sci. Rep.*, 7(1):18022. 13

Chewapreecha, C., Harris, S. R., Croucher, N. J., Turner, C., Marttinen, P., Cheng, L., Pessia, A., Aanensen, D. M., Mather, A. E., Page, A. J., Salter, S. J., Harris, D., Nosten, F., Goldblatt, D., Corander, J., Parkhill, J., Turner, P., and Bentley, S. D. (2014). Dense genomic sampling identifies highways of pneumococcal recombination. *Nat. Genet.*, 46(3):305–309. 28

Collins, R. E. and Higgs, P. G. (2012). Testing the infinitely many genes model for the evolution of the bacterial core genome and pangenome. *Mol. Biol. Evol.*, 29(11):3413–3425. 34

Cornish, A. and Guda, C. (2015). A Comparison of Variant Calling Pipelines Using Genome in a Bottle as a Reference. *Biomed Res. Int.*, 2015:1–11. 16, 20

Croll, D. and McDonald, B. A. (2012). The Accessory Genome as a Cradle for Adaptive Evolution in Pathogens. *PLoS Pathog.*, 8(4):e1002608. 9

Croucher, N. J., Coupland, P. G., Stevenson, A. E., Callendrello, A., Bentley, S. D., and Hanage, W. P. (2014). Diversification of bacterial genome content through distinct mechanisms over different timescales. *Nat. Commun.*, 5(1):5471. 28

Croucher, N. J., Finkelstein, J. A., Pelton, S. I., Mitchell, P. K., Lee, G. M., Parkhill, J., Bentley, S. D., Hanage, W. P., and Lipsitch, M. (2013). Population genomics of post-vaccine changes in pneumococcal epidemiology. *Nat. Genet.*, 45(6):656–663. 28

Croxen, M. A., Law, R. J., Scholz, R., Keeney, K. M., Wlodarska, M., and Finlay, B. B. (2013). Recent advances in understanding enteric pathogenic Escherichia coli. 12

Danecek, P., Auton, A., Abecasis, G., Albers, C. A., Banks, E., DePristo, M. A., Handsaker, R. E., Lunter, G., Marth, G. T., Sherry, S. T., McVean, G., and Durbin, R. (2011). The variant call format and VCFtools. *Bioinformatics*, 27(15):2156–2158. vii, 21, 22, 33

Decraene, V., Phan, H. T. T., George, R., Wyllie, D. H., Akinremi, O., Aiken, Z., Cleary, P., Dodgson, A., Pankhurst, L., Crook, D. W., Lenney, C., Walker, A. S., Woodford, N., Sebra, R., Fath-Ordoubadi, F., Mathers, A. J., Seale, A. C., Guiver, M., McEwan, A., Watts, V., Welfare, W., Stoesser, N., Cawthorne, J., and TRACE Investigators' Group, t. T. I. (2018). A Large, Refractory Nosocomial Outbreak of Klebsiella pneumoniae Carbapenemase-Producing Escherichia coli Demonstrates Carbapenemase Gene Outbreaks Involving Sink Sites Require Novel Approaches to Infection Control. *Antimicrob. Agents Chemother.*, 62(12):e01689–18. 85, 96, 101, 102, 104, 105

Delcher, A. L., Kasif, S., Fleischmann, R. D., Peterson, J., White, O., and Salzberg, S. L. (1999). Alignment of whole genomes. *Nucleic Acids Res.*, 27(11):2369–76. 15

den Bakker, H. C., Allard, M. W., Bopp, D., Brown, E. W., Fontana, J., Iqbal, Z., Kinney, A., Limberger, R., Musser, K. A., Shudt, M., Strain, E., Wiedmann, M., and Wolfgang, W. J. (2014). Rapid whole-genome sequencing for surveillance of salmonella enterica serovar Enteritidis. *Emerg. Infect. Dis.*, 20(8):1306–1314. 10

Depristo, M. A., Banks, E., Poplin, R., Garimella, K. V., Maguire, J. R., Hartl, C., Philippakis, A. A., Del Angel, G., Rivas, M. A., Hanna, M., McKenna, A., Fennell, T. J., Kernytsky, A. M., Sivachenko, A. Y., Cibulskis, K., Gabriel, S. B., Altshuler, D., and Daly, M. J. (2011). A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.*, 43(5):491–501. 16, 21

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nat. Biotechnol.*, 35(4):316–319. 3, 35, 110

Didelot, X. and Falush, D. (2007). Inference of bacterial microevolution using multilocus sequence data. *Genetics*, 175(3):1251–1266. 10

Didelot, X. and Maiden, M. C. J. (2010). Impact of recombination on bacterial evolution. 7

Didelot, X., Méric, G., Falush, D., and Darling, A. E. (2012). Impact of homologous and non-homologous recombination in the genomic evolution of Escherichia coli. *BMC Genomics*, 13(1):256. 7

Didelot, X., Walker, A. S., Peto, T. E., Crook, D. W., and Wilson, D. J. (2016). Within-host evolution of bacterial pathogens. *Nat. Rev. Microbiol.*, 14(3):150–162. 17, 60

Didelot, X. and Wilson, D. J. (2015). ClonalFrameML: Efficient Inference of Recombination in Whole Bacterial Genomes. *PLoS Comput. Biol.*, 11(2):e1004041. 10

Dilthey, A., Cox, C., Iqbal, Z., Nelson, M. R., and McVean, G. (2015). Improved genome inference in the MHC using a population reference graph. *Nat. Genet.*, 47(6):682–8. 18, 29, 38

Ding, W., Baumdicker, F., and Neher, R. A. (2017). panX: pan-genome analysis and exploration. *Nucleic Acids Res.*, 46(1):e5–e5. 30, 35

Drezen, E., Rizk, G., Chikhi, R., Deltel, C., Lemaitre, C., Peterlongo, P., and Lavenier, D. (2014). GATB: Genome Assembly & Analysis Tool Box. *Bioinformatics*, 30(20):2959–2961. 99

Durand, G., Javerliat, F., Bes, M., Veyrieras, J.-B., Guigon, G., Mugnier, N., Schicklin, S., Kaneko, G., Santiago-Allexant, E., Bouchiat, C., Martins-Simões, P., Laurent, F., Van Belkum, A., Vandenesch, F., and Tristan, A. (2018). Routine Whole-Genome Sequencing for Outbreak Investigations of Staphylococcus aureus in a National Reference Center. *Front. Microbiol.*, 9:511. 10

Durrett, R. (2008). *Probability Models for DNA Sequence Evolution*. Probability and its Applications. Springer New York, New York, NY. 108

Eggertsson, H. P., Jonsson, H., Kristmundsdottir, S., Hjartarson, E., Kehr, B., Masson, G., Zink, F., Hjorleifsson, K. E., Jonasdottir, A., Jonasdottir, A., Jonsdottir, I., Gudbjartsson, D. F., Melsted, P., Stefansson, K., and Halldorsson, B. V. (2017). Graphtyper enables population-scale genotyping using pangenome graphs. *Nat. Genet.*, 49(11):1654–1660. 18

Eid, J., Fehr, A., Gray, J., Luong, K., Lyle, J., Otto, G., Peluso, P., Rank, D., Baybayan, P., Bettman, B., Bibillo, A., Bjornson, K., Chaudhuri, B., Christians,

F., Cicero, R., Clark, S., Dalal, R., DeWinter, A., Dixon, J., Foquet, M., Gaertner, A., Hardenbol, P., Heiner, C., Hester, K., Holden, D., Kearns, G., Kong, X., Kuse, R., Lacroix, Y., Lin, S., Lundquist, P., Ma, C., Marks, P., Maxham, M., Murphy, D., Park, I., Pham, T., Phillips, M., Roy, J., Sebra, R., Shen, G., Sorenson, J., Tomaney, A., Travers, K., Trulson, M., Vieceli, J., Wegener, J., Wu, D., Yang, A., Zaccarin, D., Zhao, P., Zhong, F., Korlach, J., and Turner, S. (2009). Real-time DNA sequencing from single polymerase molecules. *Science (80-.).*, 323(5910):133–138. 13

Eldholm, V. and Balloux, F. (2016). Antimicrobial Resistance in Mycobacterium tuberculosis: The Odd One Out. *Trends in Microbiology*, 24(8):637–648. vii, 9

Emms, D. M. and Kelly, S. (2015). OrthoFinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. *Genome Biol.*, 16(1):157. 34

Enright, A. J., Van Dongen, S., and Ouzounis, C. A. (2002). An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res.*, 30(7):1575–84. 35

Faria, N. R., Quick, J., Claro, I. M., Thézé, J., De Jesus, J. G., Giovanetti, M., Kraemer, M. U., Hill, S. C., Black, A., Da Costa, A. C., Franco, L. C., Silva, S. P., Wu, C. H., Raghwani, J., Cauchemez, S., Du Plessis, L., Verotti, M. P., De Oliveira, W. K., Carmo, E. H., Coelho, G. E., Santelli, A. C., Vinhal, L. C., Henriques, C. M., Simpson, J. T., Loose, M., Andersen, K. G., Grubaugh, N. D., Somasekar, S., Chiu, C. Y., Muñoz-Medina, J. E., Gonzalez-Bonilla, C. R., Arias, C. F., Lewis-Ximenez, L. L., Baylis, S. A., Chieppe, A. O., Aguiar, S. F., Fernandes, C. A., Lemos, P. S., Nascimento, B. L., Monteiro, H. A., Siqueira, I. C., De Queiroz, M. G., De Souza, T. R., Bezerra, J. F., Lemos, M. R., Pereira, G. F., Loudal, D., Moura, L. C., Dhalia, R., França, R. F., Magalhães, T., Marques, E. T., Jaenisch, T., Wallau, G. L., De Lima, M. C., Nascimento, V., De Cerqueira, E. M., De Lima, M. M., Mascarenhas, D. L., Neto, J. P., Levin, A. S., Tozetto-Mendoza, T. R., Fonseca, S. N., Mendes-Correa, M. C., Milagres, F. P., Segurado, A., Holmes, E. C., Rambaut, A., Bedford, T., Nunes, M. R., Sabino, E. C., Alcantara, L. C., Loman, N. J., and Pybus, O. G. (2017). Establishment and cryptic transmission of Zika virus in Brazil and the Americas. *Nature*, 546(7658):406–410. 13

Feijao, P., Yao, H.-T., Fornika, D., Gardy, J., Hsiao, W., Chauve, C., and Chindelevitch, L. (2018). MentaLiST A fast MLST caller for large MLST schemes. *Microb. Genomics*, 4(2).

Fleischmann, R. D., Adams, M. D., White, O., Clayton, R. A., Kirkness, E. F., Kerlavage, A. R., Bult, C. J., Tomb, J.-F., Dougherty, B. A., Merrick, J. M., Mckenney, K., Sutton, G., Fitzhugh, W., Fields, C., Gocyne, J. D., Scott, J., Shirley, R., Liu, L.-I., Glodek, A., Kelley, J. M., Weidman, J. F., Phillips, C. A., Spriggs, T., Hedblom, E., Cotton, M. D., and Utterback, T. R. (1995). Whole-Genome Random Sequencing and Assembly of Haemophilus Influenzae. *Science (80-.).*, 269(5223):496–498. 12

Furuya, E. Y. and Lowy, F. D. (2006). Antimicrobial-resistant bacteria in the community setting. 6

Garrison, E. and Marth, G. (2012). Haplotype-based variant detection from short-read sequencing. *arXiv*. 16

Garrison, E., Sirén, J., Novak, A. M., Hickey, G., Eizenga, J. M., Dawson, E. T., Jones, W., Garg, S., Markello, C., Lin, M. F., Paten, B., and Durbin, R. (2018). Variation graph toolkit improves read mapping by representing genetic variation in the reference. 18, 29, 38

Gomez-Eichelmann, M. C., Levy-Mustri, A., and Ramirez-Santos, J. (1991). Presence of 5-Methylcytosine in CC(A/T)GG sequences (Dcm Methylation) in DNAs from different bacteria. 80

Goodwin, S., Gurtowski, J., Ethe-Sayers, S., Deshpande, P., Schatz, M. C., and Mccombie, W. R. (2015). Oxford Nanopore sequencing, hybrid error correction, and de novo assembly of a eukaryotic genome. *Genome Res.*, 25(11):1750–6. 51

Goodwin, S., McPherson, J. D., and McCombie, W. R. (2016). Coming of age: ten years of next-generation sequencing technologies. *Nat. Rev. Genet.*, 17(6):333–351. 13

Gorrie, C. L., Mirc Eta, M., Wick, R. R., Edwards, D. J., Thomson, N. R., Strugnell, R. A., Pratt, N. F., Garlick, J. S., Watson, K. M., Pilcher, D. V., McGloughlin, S. A., Spelman, D. W., Jenney, A. W., and Holt, K. E. (2017). Gastrointestinal Carriage Is a Major Reservoir of Klebsiella pneumoniae Infection in Intensive Care Patients. *Clin. Infect. Dis.*, 65(2):208–215. 28, 106, 109

Griffiths, R. C. (1981). Neutral two-locus multiple allele models with recombination. *Theor. Popul. Biol.*, 19(2):169–186. 10

Haley, B. J., Kim, S. W., Pettengill, J., Luo, Y., Karns, J. S., and Van Kessel, J. A. S. (2016). Genomic and evolutionary analysis of two salmonella enterica serovar Kentucky sequence types isolated from bovine and poultry sources in North America. *PLoS One*, 11(10):e0161225. 95

Hall, B. G. (2004). Predicting the evolution of antibiotic resistance genes. *Nat. Rev. Microbiol.*, 2(5):430–435. 11

Highnam, G., Wang, J. J., Kusler, D., Zook, J., Vijayan, V., Leibovich, N., and Mittelman, D. (2015). An analytical framework for optimizing variant discovery from personal genomes. *Nat. Commun.*, 6(1):6275. 20

Holt, K. E., Wertheim, H., Zadoks, R. N., Baker, S., Whitehouse, C. A., Dance, D., Jenney, A., Connor, T. R., Hsu, L. Y., Severin, J., Brisse, S., Cao, H., Wilksch, J., Gorrie, C., Schultz, M. B., Edwards, D. J., Nguyen, K. V., Nguyen, T. V., Dao, T. T., Mensink, M., Minh, V. L., Nhu, N. T. K., Schultsz, C., Kuntaman, K., Newton, P. N., Moore, C. E., Strugnell, R. A., and Thomson, N. R. (2015). Genomic analysis of diversity, population structure, virulence, and antimicrobial resistance in Klebsiella pneumoniae, an urgent threat to public health. *Proc. Natl. Acad. Sci. U. S. A.*, 112(27):E3574–81. 11, 110

Huang, W., Li, L., Myers, J. R., and Marth, G. T. (2012). ART: A next-generation sequencing read simulator. *Bioinformatics*, 28(4):593–594. 48, 67, 85

Hudson, R. R. (1983). Properties of a neutral allele model with intragenic recombination. *Theor. Popul. Biol.*, 23(2):183–201. 10

Huson, D. H. and Scornavacca, C. (2012). Software for Systematics and Evolution Dendroscope 3: An Interactive Tool for Rooted Phylogenetic Trees and Networks. *Syst. Biol.*, 61(6):1061–1067. 105

Hwang, S., Kim, E., Lee, I., and Marcotte, E. M. (2015). Systematic comparison of variant calling pipelines using gold standard personal exome variants. *Sci. Rep.*, 5(1):17875. 16, 20

Iqbal, Z., Caccamo, M., Turner, I., Flicek, P., and McVean, G. (2012). De novo assembly and genotyping of variants using colored de Bruijn graphs. *Nat. Genet.*, 44(2):226–232. 15, 17, 29

Jackson, B. R., Tarr, C., Strain, E., Jackson, K. A., Conrad, A., Carleton, H., Katz, L. S., Stroika, S., Gould, L. H., Mody, R. K., Silk, B. J., Beal, J., Chen, Y., Timme, R., Doyle, M., Fields, A., Wise, M., Tillman, G., Defibaugh-Chavez, S., Kucerova, Z., Sabol, A., Roache, K., Trees, E., Simmons, M., Wasilenko, J., Kubota, K., Pouseele, H., Klimke, W., Besser, J., Brown, E., Allard, M., and Gerner-Smidt, P. (2016). Implementation of Nationwide Real-time Whole-genome Sequencing to Enhance Listeriosis Outbreak Detection and Investigation. *Clin. Infect. Dis.*, 63(3):380–386. 10

Jain, C., Koren, S., Dilthey, A., Phillippy, A. M., and Aluru, S. (2018). A fast adaptive algorithm for computing whole-genome homology maps. *Bioinformatics*, 34(17):i748–i756. 38, 40

Jain, M., Fiddes, I. T., Miga, K. H., Olsen, H. E., Paten, B., and Akeson, M. (2015). Improved data analysis for the MinION nanopore sequencer. *Nat. Methods*, 12(4):351–356. 16

Jain, M., Tyson, J. R., Loose, M., Ip, C. L., Eccles, D. A., O'Grady, J., Malla, S., Leggett, R. M., Wallerman, O., Jansen, H. J., Zalunin, V., Birney, E., Brown, B. L., Snutch, T. P., and Olsen, H. E. (2017). MinION Analysis and Reference Consortium: Phase 2 data release and analysis of R9.0 chemistry. *F1000Research*, 6:760. 49

Jolley, K. A. and Maiden, M. C. J. (2010). BIGSdb: Scalable analysis of bacterial genome variation at the population level. *BMC Bioinformatics*, 11(1):595. 14

Kahn, A. B. (1962). Topological sorting of large networks. *Commun. ACM*, 5(11):558–562. 63

Kallonen, T., Brodrick, H. J., Harris, S. R., Corander, J., Brown, N. M., Martin, V., Peacock, S. J., and Parkhill, J. (2017). Systematic longitudinal survey of invasive Escherichia coli in England demonstrates a stable population structure only transiently disturbed by the emergence of ST131. *Genome Res.*, 27(8):1437–1449. 28

Kaper, J. B., Nataro, J. P., and Mobley, H. L. T. (2004). Pathogenic Escherichia coli. *Nat. Rev. Microbiol.*, 2(2):123–140. 34

Katoh, K. and Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Mol. Biol. Evol.*, 30(4):772–80. 110

Katz, L. S., Griswold, T., Williams-Newkirk, A. J., Wagner, D., Petkau, A., Sieffert, C., Domselaar, G. V., Deng, X., and Carleton, H. A. (2017). A comparative analysis of the Lyve-SET phylogenomics pipeline for genomic epidemiology of foodborne pathogens. *Front. Microbiol.*, 8(MAR):375. 16

Kent, W. J. (2002). BLAT–the BLAST-like alignment tool. *Genome Res.*, 12(4):656–64. 40

Kingman, J. F. C. (1982). The coalescent. *Stoch. Process. their Appl.*, 13(3):235–248. 107

Koboldt, D. C., Chen, K., Wylie, T., Larson, D. E., McLellan, M. D., Mardis, E. R., Weinstock, G. M., Wilson, R. K., and Ding, L. (2009). VarScan: Variant detection in massively parallel sequencing of individual and pooled samples. *Bioinformatics*, 25(17):2283–2285. 16

Kolmogorov, M., Yuan, J., Lin, Y., and Pevzner, P. A. (2019). Assembly of long, error-prone reads using repeat graphs. *Nat. Biotechnol.*, 37(5):540–546. 13

Koren, S., Schatz, M. C., Walenz, B. P., Martin, J., Howard, J. T., Ganapathy, G., Wang, Z., Rasko, D. A., McCombie, W. R., Jarvis, E. D., and Phillippy, A. M. (2012). Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nat. Biotechnol.*, 30(7):693–700. 13

Koren, S., Walenz, B. P., Berlin, K., Miller, J. R., Bergman, N. H., and Phillippy, A. M. (2017). Canu: Scalable and accurate long-read assembly via adaptive κ-mer weighting and repeat separation. *Genome Res.*, 27(5):722–736. 13, 40, 86

Köser, C. U., Holden, M. T., Ellington, M. J., Cartwright, E. J., Brown, N. M., Ogilvy-Stuart, A. L., Hsu, L. Y., Chewapreecha, C., Croucher, N. J., Harris, S. R., Sanders, M., Enright, M. C., Dougan, G., Bentley, S. D., Parkhill, J., Fraser, L. J., Betley, J. R., Schulz-Trieglaff, O. B., Smith, G. P., and Peacock, S. J. (2012). Rapid Whole-Genome Sequencing for Investigation of a Neonatal MRSA Outbreak. *N. Engl. J. Med.*, 366(24):2267–2275. 10

Kuhnert, P., Nicolet, J., and Frey, J. (1995). Rapid and accurate identification of Escherichia coli K-12 strains. *Appl. Environ. Microbiol.*, 61(11):4135–9. 51

Kurtz, S., Phillippy, A., Delcher, A. L., Smoot, M., Shumway, M., Antonescu, C., and Salzberg, S. L. (2004). Versatile and open software for comparing large genomes. *Genome Biol.*, 5(2):R12. 17, 25, 85, 86, 89

Kurtzer, G. M., Sochat, V., and Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PLoS One*, 12(5):e0177459. 3

Land, M., Hauser, L., Jun, S. R., Nookaew, I., Leuze, M. R., Ahn, T. H., Karpinets, T., Lund, O., Kora, G., Wassenaar, T., Poudel, S., and Ussery, D. W. (2015). Insights from 20 years of bacterial genome sequencing. 34

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., Fitzhugh, W., Funke, R., Gage, D., Harris, K., Heaford, A., Howland, J., Kann, L., Lehoczky, J., Levine, R., McEwan, P., McKernan, K., Meldrim, J., Mesirov, J. P., Miranda, C., Morris, W., Naylor, J., Raymond, C., Rosetti, M., Santos, R., Sheridan, A., Sougnez, C., Stange-Thomann, N., Stojanovic, N., Subramanian, A., Wyman, D., Rogers, J., Sulston, J., Ainscough, R., Beck, S., Bentley, D., Burton, J., Clee, C., Carter, N., Coulson, A., Deadman, R., Deloukas, P., Dunham, A., Dunham, I., Durbin, R., French, L., Grafham, D., Gregory, S., Hubbard, T., Humphray, S., Hunt, A., Jones, M., Lloyd, C., McMurray, A., Matthews, L., Mercer, S., Milne, S., Mullikin, J. C., Mungall, A., Plumb, R., Ross, M., Shownkeen, R., Sims, S., Waterston, R. H., Wilson, R. K., Hillier, L. W., McPherson, J. D., Marra, M. A., Mardis, E. R., Fulton, L. A., Chinwalla, A. T., Pepin, K. H., Gish, W. R., Chissoe, S. L., Wendl, M. C., Delehaunty, K. D., Miner, T. L., Delehaunty, A., Kramer, J. B., Cook, L. L., Fulton, R. S., Johnson, D. L., Minx, P. J., Clifton, S. W., Hawkins, T., Branscomb, E., Predki, P., Richardson, P., Wenning, S., Slezak, T., Doggett, N., Cheng, J. F., Olsen, A., Lucas, S., Elkin, C., Uberbacher, E., Frazier, M., Gibbs, R. A., Muzny, D. M., Scherer, S. E., Bouck, J. B., Sodergren, E. J., Worley, K. C., Rives, C. M., Gorrell, J. H., Metzker, M. L., Naylor, S. L., Kucherlapati, R. S., Nelson, D. L., Weinstock, G. M., Sakaki, Y., Fujiyama, A., Hattori, M., Yada, T., Toyoda, A., Itoh, T., Kawagoe, C., Watanabe, H., Totoki, Y., Taylor, T., Weissenbach, J., Heilig, R., Saurin, W., Artiguenave, F., Brottier, P., Bruls, T., Pelletier, E., Robert, C., Wincker, P., Rosenthal, A., Platzer, M., Nyakatura, G., Taudien, S., Rump, A., Smith, D. R., Doucette-Stamm, L., Rubenfield, M., Weinstock, K., Hong, M. L.,

Dubois, J., Yang, H., Yu, J., Wang, J., Huang, G., Gu, J., Hood, L., Rowen, L., Madan, A., Qin, S., Davis, R. W., Federspiel, N. A., Abola, A. P., Proctor, M. J., Roe, B. A., Chen, F., Pan, H., Ramser, J., Lehrach, H., Reinhardt, R., McCombie, W. R., De La Bastide, M., Dedhia, N., Blöcker, H., Hornischer, K., Nordsiek, G., Agarwala, R., Aravind, L., Bailey, J. A., Bateman, A., Batzoglou, S., Birney, E., Bork, P., Brown, D. G., Burge, C. B., Cerutti, L., Chen, H. C., Church, D., Clamp, M., Copley, R. R., Doerks, T., Eddy, S. R., Eichler, E. E., Furey, T. S., Galagan, J., Gilbert, J. G., Harmon, C., Hayashizaki, Y., Haussler, D., Hermjakob, H., Hokamp, K., Jang, W., Johnson, L. S., Jones, T. A., Kasif, S., Kaspryzk, A., Kennedy, S., Kent, W. J., Kitts, P., Koonin, E. V., Korf, I., Kulp, D., Lancet, D., Lowe, T. M., McLysaght, A., Mikkelsen, T., Moran, J. V., Mulder, N., Pollara, V. J., Ponting, C. P., Schuler, G., Schultz, J., Slater, G., Smit, A. F., Stupka, E., Szustakowki, J., Thierry-Mieg, D., Thierry-Mieg, J., Wagner, L., Wallis, J., Wheeler, R., Williams, A., Wolf, Y. I., Wolfe, K. H., Yang, S. P., Yeh, R. F., Collins, F., Guyer, M. S., Peterson, J., Felsenfeld, A., Wetterstrand, K. A., Myers, R. M., Schmutz, J., Dickson, M., Grimwood, J., Cox, D. R., Olson, M. V., Kaul, R., Raymond, C., Shimizu, N., Kawasaki, K., Minoshima, S., Evans, G. A., Athanasiou, M., Schultz, R., Patrinos, A., and Morgan, M. J. (2001). Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921. 12

Le, V. T. M. and Diep, B. A. (2013). Selected insights from application of whole-genome sequencing for outbreak investigations. 14

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv*. 15, 67, 87

Li, H. (2015). FermiKit: Assembly-based variant calling for Illumina resequencing data. *Bioinformatics*, 31(22):3694–3696. 15

Li, H. (2016). Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics*, 32(14):2103–2110. 41

Li, H. (2018). Minimap2: Pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100. 13, 16, 38, 40, 67

Li, H. and Durbin, R. (2010). Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, 26(5):589–595. 25

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and Durbin, R. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079. 15, 16

Li, L., Stoeckert, C. J., and Roos, D. S. (2003). OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Res.*, 13(9):2178–89. 34

Loman, N. J., Quick, J., and Simpson, J. T. (2015). A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nat. Methods*, 12(8):733–735. 13, 16, 68, 88

Lovett, S. T. (2004). Encoded errors: Mutations and rearrangements mediated by misalignment at repetitive DNA sequences. 5

Lunter, G. and Goodson, M. (2011). Stampy: a statistical algorithm for sensitive and fast mapping of Illumina sequence reads. *Genome Res.*, 21(6):936–9. 15

Luo, R., Sedlazeck, F. J., Lam, T.-W., and Schatz, M. (2018). Clairvoyante: a multi-task convolutional deep neural network for variant calling in Single Molecule Sequencing. *bioRxiv*. 17

Maciuca, S., del Ojo Elias, C., McVean, G., and Iqbal, Z. (2016). A Natural Encoding of Genetic Variation in a Burrows-Wheeler Transform to Enable Mapping and Genome Inference. In *Algorithms Bioinformatics. WABI 2016. Lect. Notes Comput. Sci. vol 9838*, pages 222–233. Springer. 18, 29, 38

MacQueen, J. (1967). Some Methods for classification and Analysis of Multivariate Observations. In *5th Berkeley Symp. Math. Stat. Probab.*, volume 1, pages 281–297. University of California Press. 31

Maiden, M. C., Bygraves, J. A., Feil, E., Morelli, G., Russell, J. E., Urwin, R., Zhang, Q., Zhou, J., Zurth, K., Caugant, D. A., Feavers, I. M., Achtman, M., and Spratt, B. G. (1998). Multilocus sequence typing: a portable approach to the identification of clones within populations of pathogenic microorganisms. *Proc. Natl. Acad. Sci. U. S. A.*, 95(6):3140–5. 14

Maiden, M. C. J., van Rensburg, M. J. J., Bray, J. E., Earle, S. G., Ford, S. A., Jolley, K. A., and McCarthy, N. D. (2013). MLST revisited: the gene-by-gene approach to bacterial genomics. 14

Maio, N. D., Shaw, L. P., Hubbard, A., George, S., Sanderson, N., Swann, J., Wick, R., AbuOun, M., Stubberfield, E., Hoosdally, S. J., Crook, D. W., Peto, T. E. A., Sheppard, A. E., Bailey, M. J., Read, D. S., Anjum, M. F., Walker, A. S., and Stoesser, N. (2019). Comparison of long-read sequencing technologies in the hybrid assembly of complex bacterial genomes. *bioRxiv*. 87

McInerney, J. O., McNally, A., and O'Connell, M. J. (2017). Why prokaryotes have pangenomes. *Nat. Microbiol.*, 2(4):17040. 34

McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., and DePristo, M. A. (2010). The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.*, 20(9):1297–303. 16, 21

Molina, N. and Van Nimwegen, E. (2008). Universal patterns of purifying selection at noncoding positions in bacteria. *Genome Res.*, 18(1):148–160. 35

Nathan, C. and Cars, O. (2014). Antibiotic Resistance Problems, Progress, and Prospects. *N. Engl. J. Med.*, 371(19):1761–1763. 11

Nielsen, R., Paul, J. S., Albrechtsen, A., and Song, Y. S. (2011). Genotype and SNP calling from next-generation sequencing data. *Nat. Rev. Genet.*, 12(6):443–51. 16

Oliveira, P. H., Touchon, M., Cury, J., and Rocha, E. P. C. (2017). The chromosomal organization of horizontal gene transfer in bacteria. *Nat. Commun.*, 8(1):841. 6

Olson, N. D., Lund, S. P., Colman, R. E., Foster, J. T., Sahl, J. W., Schupp, J. M., Keim, P., Morrow, J. B., Salit, M. L., and Zook, J. M. (2015). Best practices for evaluating single nucleotide variant calling methods for microbial genomics. vii, 15, 19, 27, 48

Ondov, B. D., Treangen, T. J., Melsted, P., Mallonee, A. B., Bergman, N. H., Koren, S., and Phillippy, A. M. (2016). Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biol.*, 17(1):132. 40, 89, 96

Page, A. J., Cummins, C. A., Hunt, M., Wong, V. K., Reuter, S., Holden, M. T., Fookes, M., Falush, D., Keane, J. A., and Parkhill, J. (2015). Roary: Rapid large-scale prokaryote pan genome analysis. *Bioinformatics*, 31(22):3691–3693. 30, 34, 110

Page, A. J. and Keane, J. A. (2018). Rapid multi-locus sequence typing direct from uncorrected long reads using Krocus. *PeerJ*, 6:e5233. 15

Patro, R., Duggal, G., Love, M. I., Irizarry, R. A., and Kingsford, C. (2017). Salmon provides fast and bias-aware quantification of transcript expression. *Nat. Methods*, 14(4):417–419. 81

Pérez-Losada, M., Arenas, M., and Castro-Nallar, E. (2017). Multilocus Sequence Typing of Pathogens. In *Genet. Evol. Infect. Dis.*, pages 383–404. Elsevier, second edi edition. 14

Petkau, A., Mabon, P., Sieffert, C., Knox, N. C., Cabral, J., Iskander, M., Iskander, M., Weedmark, K., Zaheer, R., Katz, L. S., Nadon, C., Reimer, A., Taboada, E., Beiko, R. G., Hsiao, W., Brinkman, F., Graham, M., and Van Domselaar, G. (2017). SNVPhyl: a single nucleotide variant phylogenomics pipeline for microbial genomic epidemiology. *Microb. Genomics*, 3(6). 16

Pightling, A. W., Petronella, N., and Pagotto, F. (2014). Choice of Reference Sequence and Assembler for Alignment of Listeria monocytogenes Short-Read Sequence Data Greatly Influences Rates of Error in SNP Analyses. *PLoS One*, 9(8):e104579. 17, 20, 27

Pightling, A. W., Petronella, N., and Pagotto, F. (2015). Choice of reference-guided sequence assembler and SNP caller for analysis of Listeria monocytogenes short-read sequence data greatly influences rates of error. *BMC Res. Notes*, 8(1):748. 16

Pightling, A. W., Pettengill, J. B., Luo, Y., Baugher, J. D., Rand, H., and Strain, E. (2018). Interpreting whole-genome sequence analyses of foodborne bacteria for regulatory applications and outbreak investigations. *Front. Microbiol.*, 9:1482. 10

Powers, J. G., Weigman, V. J., Shu, J., Pufky, J. M., Cox, D., and Hurban, P. (2013). Efficient and accurate whole genome assembly and methylome profiling of E. coli. *BMC Genomics*, 14(1):675. 12

Price, M. N., Dehal, P. S., and Arkin, A. P. (2009). FastTree: computing large minimum evolution trees with profiles instead of a distance matrix. *Mol. Biol. Evol.*, 26(7):1641–50. 103

Price, M. N., Dehal, P. S., and Arkin, A. P. (2010). FastTree 2 - Approximately maximum-likelihood trees for large alignments. *PLoS One*, 5(3):e9490. 103

Quick, J., Loman, N. J., Duraffour, S., Simpson, J. T., Severi, E., Cowley, L., Bore, J. A., Koundouno, R., Dudas, G., Mikhail, A., Ouédraogo, N., Afrough, B., Bah, A., Baum, J. H. J., Becker-Ziaja, B., Boettcher, J. P., Cabeza-Cabrerizo, M., Camino-Sánchez, Á., Carter, L. L., Doerrbecker, J., Enkirch, T., Dorival, I. G., Hetzelt, N., Hinzmann, J., Holm, T., Kafetzopoulou, L. E., Koropogui, M., Kosgey, A., Kuisma, E., Logue, C. H., Mazzarelli, A., Meisel, S., Mertens, M., Michel, J., Ngabo, D., Nitzsche, K., Pallasch, E., Patrono, L. V., Portmann, J., Repits, J. G., Rickett, N. Y., Sachse, A., Singethan, K., Vitoriano, I., Yemanaberhan, R. L., Zekeng, E. G., Racine, T., Bello, A., Sall, A. A., Faye, O., Faye, O., Magassouba, N., Williams, C. V., Amburgey, V., Winona, L., Davis, E., Gerlach, J., Washington, F., Monteil, V., Jourdain, M., Bererd, M., Camara, A., Somlare, H., Camara, A., Gerard, M., Bado, G., Baillet, B., Delaune, D., Nebie, K. Y., Diarra, A., Savane, Y., Pallawo, R. B., Gutierrez, G. J., Milhano, N., Roger, I., Williams, C. J., Yattara, F., Lewandowski, K., Taylor, J., Rachwal, P., J. Turner, D., Pollakis, G., Hiscox, J. A., Matthews, D. A., Shea, M. K. O., Johnston, A. M., Wilson, D., Hutley, E., Smit, E., Di Caro, A., Wölfel, R., Stoecker, K., Fleischmann, E., Gabriel, M., Weller, S. A., Koivogui, L., Diallo, B., Keïta, S., Rambaut, A., Formenty, P., Günther, S., and Carroll, M. W. (2016). Real-time, portable genome sequencing for Ebola surveillance. *Nature*, 530(7589):228–232. 13

Rang, F. J., Kloosterman, W. P., and de Ridder, J. (2018). From squiggle to basepair: Computational approaches for improving nanopore sequencing read accuracy. viii, 49

Reuter, J. A., Spacek, D. V., and Snyder, M. P. (2015). High-throughput sequencing technologies. *Mol. Cell*, 58(4):586–97. 12, 13

Richardson, E. J., Bacigalupe, R., Harrison, E. M., Weinert, L. A., Lycett, S., Vrieling, M., Robb, K., Hoskisson, P. A., Holden, M. T., Feil, E. J., Paterson, G. K., Tong, S. Y., Shittu, A., van Wamel, W., Aanensen, D. M., Parkhill, J., Peacock, S. J., Corander, J., Holmes, M., and Fitzgerald, J. R. (2018). Gene exchange drives the ecological success of a multi-host bacterial pathogen. *Nat. Ecol. Evol.*, 2(9):1468–1478. 12

Rimmer, A., Phan, H., Mathieson, I., Iqbal, Z., Twigg, S. R., Wilkie, A. O., Mcvean, G., and Lunter, G. (2014). Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. *Nat. Genet.*, 46(8):912–918. 15, 16

Rocha, E. P. (2018). Neutral theory, microbial practice: Challenges in bacterial population genetics. *Mol. Biol. Evol.*, 35(6):1338–1347. vi, 6, 8

Roetzer, A., Diel, R., Kohl, T. A., Rückert, C., Nübel, U., Blom, J., Wirth, T., Jaenicke, S., Schuback, S., Rüsch-Gerdes, S., Supply, P., Kalinowski, J., and Niemann, S. (2013). Whole Genome Sequencing versus Traditional Genotyping for Investigation of a Mycobacterium tuberculosis Outbreak: A Longitudinal Molecular Epidemiological Study. *PLoS Med.*, 10(2):e1001387. 10

Rosenberg, L. E. and Rosenberg, D. D. (2012). Structure of Genes, Chromosomes, and Genomes. In *Hum. Genes Genomes*, pages 75–96. Academic Press. 59

Ruan, J. and Li, H. (2019). Fast and accurate long-read assembly with wtdbg2. *bioRxiv*. 13, 60

Sakoparnig, T., Field, C., and van Nimwegen, E. (2019). Whole genome phylogenies reflect long-tailed distributions of recombination rates in many bacterial species. *bioRxiv*. 7

Schatz, M. C., Delcher, A. L., and Salzberg, S. L. (2010). Assembly of large genomes using second-generation sequencing. 13

Schierup, M. H. and Hein, J. (2000). Recombination and the molecular clock. 10

Schleimer, S., Wilkerson, D. S., and Aiken, A. (2003). Winnowing: local algorithms for document fingerprinting. In *Proc. 2003 ACM SIGMOD Int. Conf. Manag. data*, pages 76–85. 45

Schloissnig, S., Arumugam, M., Sunagawa, S., Mitreva, M., Tap, J., Zhu, A., Waller, A., Mende, D. R., Kultima, J. R., Martin, J., Kota, K., Sunyaev, S. R., Weinstock, G. M., and Bork, P. (2013). Genomic variation landscape of the human gut microbiome. *Nature*, 493(7430):45–50. 69

Segata, N., Waldron, L., Ballarini, A., Narasimhan, V., Jousson, O., and Huttenhower, C. (2012). Metagenomic microbial community profiling using unique clade-specific marker genes. *Nat. Methods*, 9(8):811–814. 14

Shen, D., Ma, G., Li, C., Jia, X., Qin, C., Yang, T., Wang, L., Jiang, X., Ding, N., Zhang, X., Yue, L., Yin, Z., Zeng, L., Zhao, Y., Zhou, D., and Chen, F. (2019). Emergence of a Multidrug-Resistant Hypervirulent Klebsiella pneumoniae Sequence Type 23 Strain with a Rare bla CTX-M-24 -Harboring Virulence Plasmid. *Antimicrob. Agents Chemother.*, 63(3):e02273–18. 11

Shen, Y., Wan, Z., Coarfa, C., Drabek, R., Chen, L., Ostrowski, E. A., Liu, Y., Weinstock, G. M., Wheeler, D. A., Gibbs, R. A., and Yu, F. (2009). A SNP discovery method to assess variant allele probability from next-generation resequencing data. *Genome Res.*, 20(2):273–280. 21

Sheppard, A. E., Stoesser, N., Wilson, D. J., Sebra, R., Kasarskis, A., Anson, L. W., Giess, A., Pankhurst, L. J., Vaughan, A., Grim, C. J., Cox, H. L., Yeh, A. J., Sifri, C. D., Walker, A. S., Peto, T. E., Crook, D. W., Mathers, A. J., and Mathers, A. J. (2016). Nested Russian Doll-Like Genetic Mobility Drives Rapid Dissemination of the Carbapenem Resistance Gene bla KPC. *Antimicrob. Agents Chemother.*, 60(6):3767–3778. 11, 34

Snitkin, E. S., Zelazny, A. M., Thomas, P. J., Stock, F., Henderson, D. K., Palmore, T. N., and Segre, J. A. (2012). Tracking a hospital outbreak of carbapenem-resistant Klebsiella pneumoniae with whole-genome sequencing. *Sci. Transl. Med.*, 4(148):148ra116–148ra116. 10

Sommer, D. D., Delcher, A. L., Salzberg, S. L., and Pop, M. (2007). Minimus: A fast, lightweight genome assembler. *BMC Bioinformatics*, 8(1):64. 86

Sović, I., Šikić, M., Wilm, A., Fenlon, S. N., Chen, S., and Nagarajan, N. (2016). Fast and sensitive mapping of nanopore sequencing reads with GraphMap. *Nat. Commun.*, 7(1):11307. 13

Stoesser, N., Giess, A., Batty, E. M., Sheppard, A. E., Walker, A. S., Wilson, D. J., Didelot, X., Bashir, A., Sebra, R., Kasarskis, A., Sthapit, B., Shakya, M., Kelly, D., Pollard, A. J., Peto, T. E., Crook, D. W., Donnelly, P., Thorson, S., Amatya, P., and Joshi, S. (2014). Genome sequencing of an extended series of NDM-producing Klebsiella pneumoniae isolates from neonatal infections in a Nepali hospital characterizes the extent of community- Versus hospital- associated transmission in an endemic setting. *Antimicrob. Agents Chemother.*, 58(12):7347–7357. 28

Talwalkar, A., Liptrap, J., Newcomb, J., Hartl, C., Terhorst, J., Curtis, K., Bresler, M., Song, Y. S., Jordan, M. I., and Patterson, D. (2014). SMaSH: A benchmarking toolkit for human genome variant calling. *Bioinformatics*, 30(19):2787–2795. 21

Thomas, C. M. and Nielsen, K. M. (2005). Mechanisms of, and barriers to, horizontal gene transfer between bacteria. 6

Thorpe, H. A., Bayliss, S. C., Sheppard, S. K., and Feil, E. J. (2018). Piggy: a rapid, large-scale pan-genome analysis tool for intergenic regions in bacteria. *Gigascience*, 7(4):1–11. 30, 35, 110

Touchon, M., Cury, J., Yoon, E. J., Krizova, L., Cerqueira, G. C., Murphy, C., Feldgarden, M., Wortman, J., Clermont, D., Lambert, T., Grillot-Courvalin, C., Nemec, A., Courvalin, P., and Rocha, E. P. (2014). The genomic diversification of the whole Acinetobacter genus: Origins, mechanisms, and consequences. *Genome Biol. Evol.*, 6(10):2866–2882. 7

Touchon, M., Hoede, C., Tenaillon, O., Barbe, V., Baeriswyl, S., Bidet, P., Bingen, E., Bonacorsi, S., Bouchier, C., Bouvet, O., Calteau, A., Chiapello, H., Clermont, O., Cruveiller, S., Danchin, A., Diard, M., Dossat, C., El Karoui, M., Frapy, E., Garry, L., Ghigo, J. M., Gilles, A. M., Johnson, J., Le Bouguénec, C., Lescat, M., Mangenot, S., Martinez-Jéhanne, V., Matic, I., Nassif, X., Oztas, S., Petit, M. A., Pichon, C., Rouy, Z., Ruf, C. S., Schneider, D., Tourret, J., Vacherie, B., Vallenet, D., Médigue, C., Rocha, E. P., and Denamur, E. (2009). Organised genome dynamics in the Escherichia coli species results in highly diverse adaptive paths. *PLoS Genet.*, 5(1):e1000344. vi, 8, 69

Truong, D. T., Tett, A., Pasolli, E., Huttenhower, C., and Segata, N. (2017). Microbial strain-level population structure & genetic diversity from metagenomes. *Genome Res.*, 27(4):626–638. 14

van der Graaf-van Bloois, L., Duim, B., Miller, W. G., Forbes, K. J., Wagenaar, J. A., and Zomer, A. (2016). Whole genome sequence analysis indicates recent diversification of mammal-associated Campylobacter fetus and implicates a genetic factor associated with H2S production. *BMC Genomics*, 17(1):713. 96

Vaser, R., Sović, I., Nagarajan, N., and Šikić, M. (2017). Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Res.*, 27(5):737–746. 13

Vaughan, T. G., Welch, D., Drummond, A. J., Biggs, P. J., George, T., and French, N. P. (2017). Inferring ancestral recombination graphs from bacterial genomic data. *Genetics*, 205(2):857–870. 10

Venter, C. J., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., Smith, H. O., Yandell, M., Evans, C. A., Holt, R. A., Gocayne, J. D., Amanatides, P., Ballew, R. M., Huson, D. H., Wortman, J. R., Zhang, Q., Kodira, C. D., Zheng, X. H., Chen, L., Skupski, M., Subramanian, G., Thomas, P. D., Zhang, J., Gabor Miklos, G. L., Nelson, C., Broder, S., Clark, A. G., Nadeau, J., McKusick, V. A., Zinder, N., Levine, A. J., Roberts, R. J., Simon, M., Slayman, C., Hunkapiller, M., Bolanos, R., Delcher, A., Dew, I., Fasulo, D., Flanigan, M., Florea, L., Halpern, A., Hannenhalli, S., Kravitz, S., Levy, S., Mobarry, C., Reinert, K., Remington, K., Abu-Threideh, J., Beasley, E., Biddick, K., Bonazzi, V., Brandon, R., Cargill, M., Chandramouliswaran, I., Charlab, R., Chaturvedi, K., Deng, Z., di Francesco, V., Dunn, P., Eilbeck, K., Evangelista, C., Gabrielian, A. E., Gan, W., Ge, W., Gong, F., Gu, Z., Guan, P., Heiman, T. J., Higgins, M. E., Ji, R. R., Ke, Z., Ketchum, K. A., Lai, Z., Lei, Y., Li, Z., Li, J., Liang, Y., Lin, X., Lu, F., Merkulov, G. V., Milshina, N., Moore, H. M., Naik, A. K., Narayan, V. A., Neelam, B., Nusskern, D., Rusch, D. B., Salzberg, S., Shao, W., Shue, B., Sun, J., Yuan Wang, Z., Wang, A., Wang, X., Wang, J., Wei, M. H., Wides, R., Xiao, C., Yan, C., Yao, A., Ye, J., Zhan, M., Zhang, W., Zhang, H., Zhao, Q., Zheng, L., Zhong, F., Zhong, W., Zhu, S. C., Zhao, S., Gilbert, D., Baumhueter, S., Spier, G., Carter, C., Cravchik, A., Woodage, T., Ali, F., An, H., Awe, A., Baldwin, D., Baden, H., Barnstead, M., Barrow, I., Beeson, K., Busam, D., Carver, A., Center, A., Lai Cheng, M., Curry, L., Danaher, S., Davenport, L., Desilets, R., Dietz, S., Dodson, K., Doup, L., Ferriera, S., Garg, N., Gluecksmann, A., Hart, B., Haynes, J., Haynes, C., Heiner, C., Hladun, S., Hostin, D., Houck, J., Howland, T., Ibegwam, C., Johnson, J., Kalush, F., Kline, L., Koduru, S., Love, A., Mann, F., May, D., McCawley, S., McIntosh, T., McMullen, I., Moy, M., Moy, L., Murphy, B., Nelson, K., Pfannkoch, C., Pratts, E., Puri, V., Qureshi, H., Reardon, M., Rodriguez, R., Rogers, Y. H., Romblad, D., Ruhfel, B., Scott, R., Sitter, C., Smallwood, M., Stewart, E., Strong, R., Suh, E., Thomas, R., Ni Tint, N., Tse, S., Vech, C., Wang, G., Wetter, J., Williams, S., Williams, M., Windsor, S., Winn-Deen, E., Wolfe, K., Zaveri, J., Zaveri, K., Abril, J. F., Guigo, R., Campbell, M. J., Sjolander, K. V., Karlak, B., Kejariwal, A., Mi, H., Lazareva, B., Hatton, T., Narechania, A., Diemer, K., Muruganujan, A., Guo, N., Sato, S., Bafna, V., Istrail, S., Lippert, R., Schwartz,

R., Walenz, B., Yooseph, S., Allen, D., Basu, A., Baxendale, J., Blick, L., Caminha, M., Carnes-Stine, J., Caulk, P., Chiang, Y. H., Coyne, M., Dahlke, C., Deslattes Mays, A., Dombroski, M., Donnelly, M., Ely, D., Esparham, S., Fosler, C., Gire, H., Glanowski, S., Glasser, K., Glodek, A., Gorokhov, M., Graham, K., Gropman, B., Harris, M., Heil, J., Henderson, S., Hoover, J., Jennings, D., Jordan, C., Jordan, J., Kasha, J., Kagan, L., Kraft, C., Levitsky, A., Lewis, M., Liu, X., Lopez, J., Ma, D., Majoros, W., McDaniel, J., Murphy, S., Newman, M., Nguyen, T., Nguyen, N., Nodell, M., Pan, S., Peck, J., Peterson, M., Rowe, W., Sanders, R., Scott, J., Simpson, M., Smith, T., Sprague, A., Stockwell, T., Turner, R., Venter, E., Wang, M., Wen, M., Wu, D., Wu, M., Xia, A., Zandieh, A., and Zhu, X. (2001). The sequence of the human genome. *Science (80-.).*, 291(5507):1304–1351. 12

Vos, M. and Didelot, X. (2009). A comparison of homologous recombination rates in bacteria and archaea. *ISME J.*, 3(2):199–208. 7

Votintseva, A. A., Bradley, P., Pankhurst, L., del Ojo Elias, C., Loose, M., Nilgiriwala, K., Chatterjee, A., Smith, E. G., Sanderson, N., Walker, T. M., Morgan, M. R., Wyllie, D. H., Walker, A. S., Peto, T. E. A., Crook, D. W., and Iqbal, Z. (2017). Same-Day Diagnostic and Surveillance Data for Tuberculosis via Whole-Genome Sequencing of Direct Respiratory Samples. *J. Clin. Microbiol.*, 55(5):1285–1298. 60

Walker, B. J., Abeel, T., Shea, T., Priest, M., Abouelliel, A., Sakthikumar, S., Cuomo, C. A., Zeng, Q., Wortman, J., Young, S. K., and Earl, A. M. (2014). Pilon: An integrated tool for comprehensive microbial variant detection and genome assembly improvement. *PLoS One*, 9(11):112963. 13, 87

Watson, M. and Warr, A. (2019). Errors in long-read assemblies can critically affect protein prediction. *Nat. Biotechnol.*, page 1. 13

Weirather, J. L., de Cesare, M., Wang, Y., Piazza, P., Sebastiano, V., Wang, X.-J., Buck, D., and Au, K. F. (2017). Comprehensive comparison of Pacific Biosciences and Oxford Nanopore Technologies and their applications to transcriptome analysis. *F1000Research*, 6:100. 74

Welch, R. A., Burland, V., Plunkett, G., Redford, P., Roesch, P., Rasko, D., Buckles, E. L., Liou, S.-R., Boutin, A., Hackett, J., Stroud, D., Mayhew, G. F., Rose, D. J., Zhou, S., Schwartz, D. C., Perna, N. T., Mobley, H. L. T., Donnenberg, M. S., and Blattner, F. R. (2002). Extensive mosaic structure revealed by the

complete genome sequence of uropathogenic Escherichia coli. *Proc. Natl. Acad. Sci.*, 99(26):17020–17024. 34

WHO (2014). ANTIMICROBIAL RESISTANCE Global Report on Surveillance. Technical report. 11, 34

Wick, R. R., Judd, L. M., Gorrie, C. L., and Holt, K. E. (2017a). Completing bacterial genome assemblies with multiplex MinION sequencing. *Microb. Genomics*, 3(10). 87

Wick, R. R., Judd, L. M., Gorrie, C. L., and Holt, K. E. (2017b). Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads. *PLoS Comput. Biol.*, 13(6):e1005595. 13

Wick, R. R., Judd, L. M., and Holt, K. E. (2019). Performance of neural network basecalling tools for Oxford Nanopore sequencing. *bioRxiv*. 49, 72, 74, 80

Wood, D. E. and Salzberg, S. L. (2014). Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.*, 15(3):R46. 14

Wyres, K. L., Nguyen, T. N., Lam, M. M., Judd, L. M., Chau, N. v. V., Dance, D. A., Ip, M., Karkey, A., Ling, C. L., Miliya, T., Newton, P., Nguyen, L., Sengduangphachanh, A., Turner, P., Veeraraghavan, B., Vinh, P. V., Vongsouvath, M., Thomson, N. R., Baker, S., and Holt, K. E. (2019). Genomic surveillance for hypervirulence and multi-drug resistance in invasive Klebsiella pneumoniae from south and southeast Asia. *bioRxiv*. 11

Yang, C., Chu, J., Warren, R. L. R. L., and Birol, I. (2017). NanoSim: Nanopore sequence read simulator based on statistical characterization. Technical Report 4. 48, 67, 85

Zankari, E., Hasman, H., Cosentino, S., Vestergaard, M., Rasmussen, S., Lund, O., Aarestrup, F. M., and Larsen, M. V. (2012). Identification of acquired antimicrobial resistance genes. *J. Antimicrob. Chemother.*, 67(11):2640–2644. 14

Zerbino, D. R. and Birney, E. (2008). Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Res.*, 18(5):821–829. 72