Statistical Modelling of Next-generation Sequencing Data from Forensic Genetics

Master Thesis
by Søren B. Vilsen

Department of Mathematical Sciences
Fredrik Bajers Vej 7G
9220 Aalborg Ø
Telephone 99 40 99 40
Fax 99 40 35 48
http://www.math.aau.dk/

Title:
Statistical Modelling of Next-generation Sequencing Data from Forensic Genetics

Project period:
September 1st, 2014 -
June 10th, 2015

Supervisor:
Torben Tvedebrink

Circulation:
6

Number of pages (appendix included):
166

Appendix:
30

Completed:
June 10th, 2015

Synopsis:

This thesis concerns itself with the statistical variation in STR NGS data, with application in forensic genetics. We introduce simple methods for DNA profiling in single contributor samples, and afterwards examine the quality associated with NGS reads. The errors are examined, first the systematic errors, stutters and shoulders, and then the more general noise. The general noise is handled using a noise threshold, which imposes drop-outs in the data. The heterozygote imbalance is therefore examined and a model for full coverage is presented. Thereafter, the probability of drop-out is predicted and the thesis concluded.

Søren Byg Vilsen

PREFACE

This chapter contains a general outline of the thesis, acknowledgements, and an abstract written in danish. The bibliography reference structure, uses Arabic numerals in square brackets, and the authors are shown in order of appearance. Most of the concepts discussed in this thesis, has been implemented using the statistical package R, from this point forth written as **R**. The functions and scripts written for the thesis can be obtained by contacting the author at the following address: `svilsen@math.aau.dk`.

Outline

Chapter 1: Introduction to the basics of DNA, STR, and NGS, as well as the data used throughout the thesis. Furthermore, this chapter also contains some preliminary results, especially with regard to uncontaminated single contributor samples.

Chapter 2: A comprehensive look into the quality generated per base in the NGS process. The chapter discusses potential uses of quality, quality restriction due to preferential detection, and the probability of two strings being equal, based on the quality ratio.

Chapter 3: This chapter aims to categorise the stutters and shoulders observed in the data. Stutters are a well known phenomenon, known to originate from the PCR process, and the main focus will therefore be to examine the hypothesis that the LUS is a better predictor of stutter ratio, than the allele length.

Chapter 4: The general noise of the data is analysed, fitting a one-inflated zero-truncated negative binomial model with the goal of creating a threshold isolating the alleles and systematic noise.

Chapter 5: As the introduction of a noise threshold has the potential to create drop-outs, the probability of drop-out is analysed, by first examining the heterozygote balance for possible covariates. As applying a noise threshold, creates missing values in the data, the covariate is imputed assuming that the full coverage follows a gamma distribution. The probability of drop-out is then predicted.

Chapter 6: This chapter recaps the thesis, adding on a few comments to some of the methods used in the thesis, and a few possibilities for future work is presented.

Appendix A: Contains a short introduction to the incomplete regularised beta function, and derivatives thereof.

Appendix B: A simulation study of the one-inflated zero-truncated negative binomial model. Data is simulated using a fixed set of parameters and the model is then applied is estimating the parameters.

Appendix C: A simulation study examination of the EM imputation using simulated gamma coverage.

Appendix D: Gamma QQ-plots of the real and simulated coverage, including 95% confidence envelopes.

Appendix E: Contains the reference profiles for the two contributors to the dilution series, strings not included.

Acknowledgements

I thank my supervisor Torben Tvedebrink, associate professor at the Department of Mathematical Sciences at Aalborg University, Denmark, for his input on the problems at hand, the ever enlightening discussions, proof reading, providing **R** support, and for dragging me into the world of forensic genetics in the first place.

A huge thanks goes to the Section of Forensic Genetics, the Department of Forensic Medicine, the Faculty of Health and Medical Science, at the University of Copenhagen, Denmark, in particular Niels Morling and Helle Smidt Mogensen, for inviting me into their world, always answering my questions, however ridiculous they may have seemed, and for providing the data used throughout the thesis.

Lastly I would like to thank Søren Højsgaard, associate professor and head of the Department of Mathematical Sciences Aalborg University, Poul Svante Eriksen, associate professor at the Department of Mathematical Sciences Aalborg University, Mikkel Meyer Andersen, assistant professor at the Department of Mathematical Sciences Aalborg University, Steffen Lauritzen, professor at the Department of Mathematical Sciences University of Copenhagen, and Therese Graversen, postdoc at the Department of Mathematical Sciences University of Copenhagen, for their invaluable input at the midway seminar. In particular Poul Svante Eriksen, for his continued input throughout and Mikkel Meyer Andersen, for providing the initial scripts and function my implementations are based on.

Abstract in Danish

The generelle formål med projektet er at opnå en forståelse for den statistiske variation, i short tandem repeat (STR) anden generations sekvensering (NGS) data, i retsgenetiske sammenhæng. Specialet starter med en introduktion til DNA, STR og NGS, hvorefter en metode til identifikation af STR-regioner i NGS data introduceres, ved hjælp af de såkaldte direkte tilstøddende flankregioner. Der bliver herefter vist hvordan en DNA profil kan dannes, hvis det vides, at prøven kun indeholder DNA fra et enkelt individ. Metoden virker dog kun i dette special tilfælde, derfor tages et grundigt blik på de fejl der opstår i NGS data genereringsprocessen, men først undersøges kvaliteten af de kaldte baser.

For en hver base sekveseret i NGS processen, tildeles også en kvalitet, der repræsenterer sandsynligheden for at basen er kaldt forkert. Kvaliteten undersøges i håb om at kunne indkorporere den i vidre analyse, enten til at restringere data (ved at fjerne strenge under et givet kvalitets niveau), eller ved at justere coverage. Det viser sig, at på grund af den måde STR-regionerne findes, så bliver kvaliteten allerede restringeret, da kvaliteten er faldende over tid. Dette medfører at sandsynligheden for fejl stiger. Konsekvensen er, at jo længere STR-region en base findes, desto

mindre bliver sandsynligheden for at den identificeret korrekt. Ydermere, udledes en metode, hvorved sandsynligheden for, at to strenge er ens kan udregnes.

Stutter og shoulder, er produkter af henholdsvis PCR og NGS processen, der begge falder under systematisk støj i data. Stutter er et velkendt fenomen fra PCR og målet ved at analysere disse er derfor at undersøge hypotesen, at LUS er en bedre predikator af stutter frekvensen, end allele længden. Yderligere, undersøges shoulder frekvensen, for at give et threshold til enten at fjerne disse fra data, eller eventuel identifikation i et mikstur tilfælde.

Udover systematisk støj, ses også en mere generel støj i data. Denne generelle støj modelleres ved en one-inlfated zero-truncated negativ binomial fordeling (KINB). KINB modellen, tilpasses ved hjælp af to forskellige metoder, implementationerne af disse undersøges i Appendiks B. De tilpassede parametre bruges derefter til at lave et støj threshold, specifikt til hver locus og sample, ved at bruge et kvartil fra fordelingen med disse parametre og dens standard afvigelse.

Ved introduktionen af et støj threshold, introduceres også drop-outs i vores data. For at bestemme sandsynligheden for at en allele drop-out sker, undersøges først heterozygot balancen. Heterozygot balancen bruges til at finde en pridikator for sandsynligheden for drop-out. Det ses, at der findes en sammenhæng mellem sandsynligheden for drop-out og standard afvigelsen af heterozygot balancen, hvis den fulde information er givet. Dette er dog ikke tilfældet, da nogle alleler ikke længere er obseveret på grund af støj threshold'et. Derfor imputeres disse ved hjælp af EM-algoritmen. Denne implementation undersøges i Appendiks C. Herefter bruges den imputerede standard afvigelse til at prediktere sandsynligheden for drop-out.

Sidst, men ikke mindst, konkluderes og kommenteres der på projektet som helhed, og der bliver givet forslag til eventuelt fremtidigt arbejde.

CONTENTS

CHAPTER

ONE

INTRODUCTION

The aim of this thesis is to investigate the statistical variation of next-generation sequencing (NGS) in forensic genetics. The use of genetic analysis in forensic cases is to create a DNA profile that is highly discriminating. Next-generation sequencing offers a higher resolution than that of capillary electrophoresis presently used in forensic casework. This chapter will serve as an introduction to the general principle of DNA, short tandem repeats (STR), next-generation sequencing, and the data used in the remainder of this thesis, as well as some preliminary results and considerations. Furthermore, Sections 1.1-1.5 will (unless otherwise indicated) be based on *The Fundamentals of Forensic DNA Typing* and *Advanced Topics in Forensic DNA Typing: Methodology* both by John M. Butler [1, 2], as well as *An Introduction to Forensic Genetics* by Goodwin, Linacre, and Hadi [3]. The data used throughout this thesis is supplied by the Section of Forensic Genetics, Department of Forensic Medicine, Faculty of Health and Medical Science, University of Copenhagen.

1.1 The Blueprint of Life

Deoxyribonucleic acid (DNA); often described as the blueprint of life. An organisms DNA contains everything needed for passing down genetic attributes to future generation. Found in every nucleated cell of our bodies it provides a *program* deter-

mining physical features and many other attributes. A DNA strand can be broken down into single units of DNA, called deoxynucleotide triphosphate (dNTP), polymerised together. A dNTP consists of three parts: a nucleobase (nitrogenous bases), a deoxyribose sugar and a phosphate group as seen in Figure 1.1. The information within the DNA is coded as a sequence of these nucleobases. The nucleobases take one of four forms adenine (A), guanine (G), cytosine (C), and thymine (T). Each of the four bases are attracted to its complimentary base, A binds to T and G to C, and form what is commonly called a base pair (bp). This attraction binds two single DNA strands into a double helix structure.

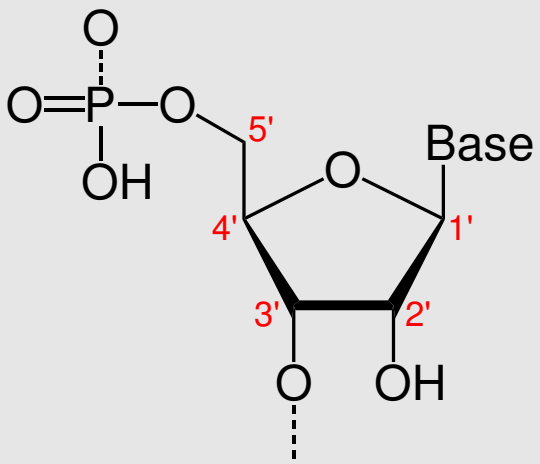


Figure 1.1: A generalized version of a deoxynucleotide triphosphate. The three parts (from right to left): the nucleobase, deoxyribose sugar and phosphate group.

The complete set of instructions necessary for making an organism, i.e. the entirety of the DNA in a cell, is referred to as the genome. In humans it contains approximately 3.2 billion bp of information organised into 22 chromosomes and the sex chromosomes. The chromosomes are named in order of size, from largest to smallest, i.e. chromosome one is the largest and chromosome 22 is the smallest. Humans contain two sets of chromosomes; one set inherited from each parent. The parts of the chromosome where the DNA code and regulate the synthesis of proteins are called genes. The non-coding regions of the chromosome are referred to as junk-DNA. The chromosomal location of a gene or a DNA marker is called a locus (loci plural).

Nomenclature for DNA markers can be split into two categories depending on

whether or not the marker falls within a gene. If it falls within, the name of the gene is used e.g. the short tandem repeat marker TH01; *TH* is short for the gene tyrosine hydroxylase found on chromosome 11, and *01* means that the repeated region is within one intron (exons are the protein-coding parts of a gene, whereas introns are the intervening sequences) of the TH gene. However, if the marker falls outside of a gene e.g. the STR markers D5S818 or DYS19, the name can be broken down as follows: *D* stands for DNA, *5* refers the 5th chromosome (in the case of a sex chromosome it will either X or Y as seen in DYS19), *S* single copy sequence and *818* indicates that it is the 818th locus to found on the specified chromosome.

The alternative forms of a gene or genetic locus are called alleles. Two alleles at a locus on homologous chromosomes (the chromosome pair have the same size and contain the same genetic structure) can be either identical called homozygous, or different, heterozygous. Suppose there are two alleles at a specific locus, A and a. It then follows that there are three different combinations of alleles, AA, Aa and aa (AA and aa are homozygous, and Aa is heterozygous). The specific combination of alleles on a given locus is referred to as a genotype and the combination of genotypes across multiple loci constitutes an individuals DNA profile.

Creating a discriminating DNA profile relies on individuals being different at a genetic level and no two individuals have been found to have the same DNA (with the exception of monozygotic twins). In general loci used for DNA profiling should ideally be highly polymorphic (have a high variation between individuals) and have a low mutation rate (have a low variation from generation to generation). Where the latter is necessary in order to resolve paternity cases.

1.2 Short Tandem Repeats

Repeated sequences come in all sizes and are designated by the length of the core repeated unit (sometimes referred to as motif). From satellites that can contain several thousand base pairs, to mini-satellites also called variable number tandem repeats (VNTRs) of 8-100 bp and micro-satellites more commonly called STRs of 2 to 7 bp. The general structure is however the same; variation between different alleles is caused by a different number of the repeated unit, which results in alleles of different lengths. The majority of forensic cases involves the analysis of STR

polymorphism. In order to analyse STR markers, flanking regions surrounding the repeats have to be determined. Then the region is amplified using a polymerase chain reaction (PCR).

PCR is an enzymatic process that replicates a marked region of DNA by heating and cooling the sample in a very specific cyclic pattern, seen in Figure 1.2, over 28-30 cycles (under optimal conditions 30 cycles yields 2^{30} replicates of the sample). The region for replication is marked by two primers, a forward and a reverse primer. Having well designed primers is a very important component of PCR amplification, it will however not be discussed further in this thesis, for more information see e.g. [1, Chapter 7].

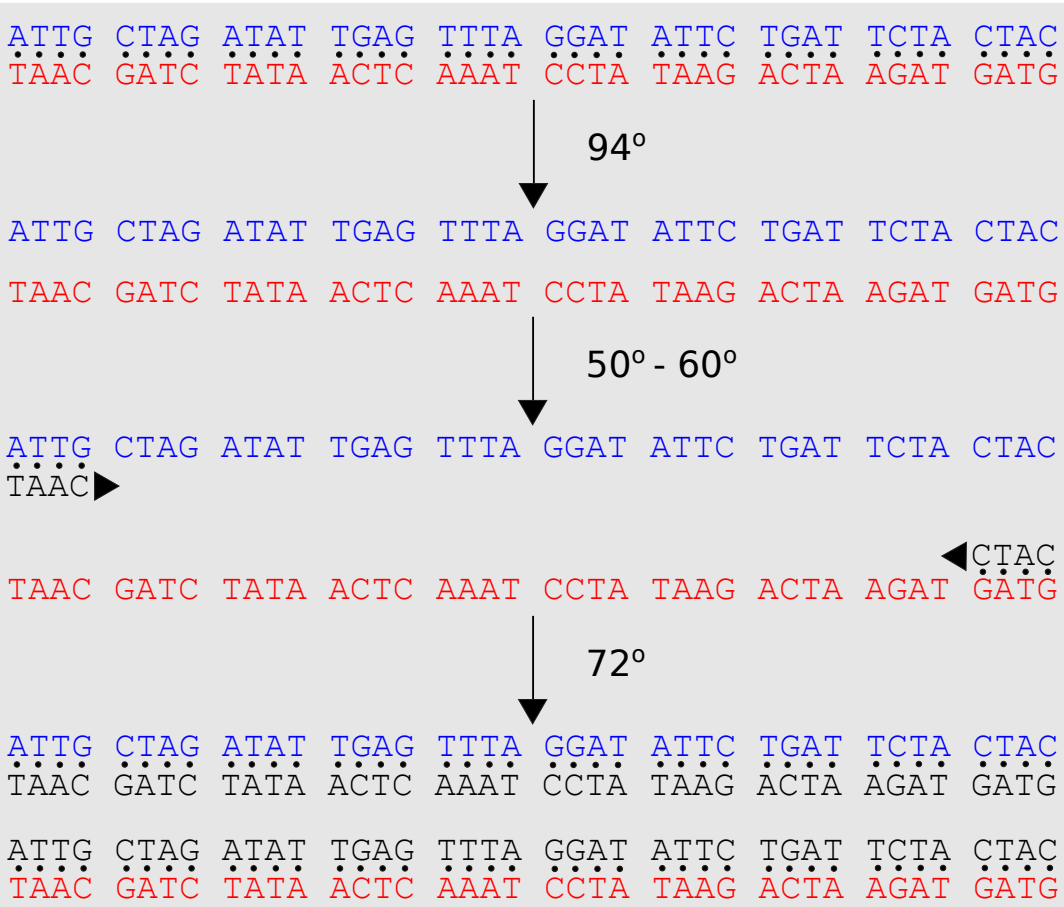


Figure 1.2: A PCR cycle. The sample is heated to 94^o to denature (separate) the two template DNA strands. It is then cooled to $50^o - 60^o$ in order to anneal (bind) the primers (the sideways facing arrows indicate the polymerase enzyme). Finally the sample is once again heated to 72^o to let the DNA polymerase extend the primers to copy the DNA template strands.

STR sequences are named by the length of the core repeated unit. In the case of human identification the most common is tetranucleotides, i.e. four nucleotides in the repeat unit. STR sequences can be split into categories based on the pattern repeated. The three most commonly used in forensic genetics are illustrated and described in Figure 1.3.

Simple repeats; the repeated units are of the same length and sequence.

TGAT TGAT TGAT TGAT TGAT

Compound repeats; made of two or more adjacent simple repeats.

TGAT TGAT TGAT GCCA GCCA

Complex repeats; may contain several repeated blocks of variable bp length.

TGAT GCCA ATCT TC TTCA

Figure 1.3: An illustration and description of simple, compound, and complex repeats.

Furthermore, not all alleles contain complete repeat units. These alleles are called microvariants. One of the most common microvariants is the 9.3 at the TH01 locus (this microvariant is present in approximately 34% of the Danish population). It contains nine tetranucleotides and one repeat unit of 3 bp, because the seventh repeat is missing an A out of its normal AATG sequence as illustrated in Figure 1.4. That is, the *.3* notation refers to the number of bases in the incomplete repeat unit.

———AATG AATG AATG AATG AATG AATG ATG AATG AATG AATG———

Figure 1.4: The 9.3 microvariant at the TH01 locus.

Even though the STR loci are in part chosen due to their low mutation rate, mutations still occur. The rate of mutation at a given marker is estimated by comparing genotypes from parents and their offspring. Most STR mutations occur, as a single repeat unit is either lost or gained, e.g. an allele 14 mutates into a 13 or 15 in the following generation. A list of mutation rates for the most common STR loci can be found in [1, p.402]. It is important to keep the mutation rate in mind when handling paternity cases, as it might otherwise lead to the wrong conclusion.

1.3 Capillary Electrophoresis

The current standard for obtaining a DNA profile is capillary electrophoresis (CE). CE works by placing the sample DNA in an inlet buffer with a negatively charged electric field. This causes the DNA molecules to give up H^+ ions making the DNA molecules negatively charged. The samples are then injected onto the capillary by applying a voltage (this process is also known as electrokenetic injection). The capillary itself is a narrow glass tube filed with a viscous polymer solution acting as a sieve for the DNA molecules. At the other end of the capillary is an outlet buffer carrying a positive charge. The DNA molecules are therefore moving from the inlet buffer through the capillary towards the outlet buffer. A high voltage is applied across the capillary in order to separate the DNA fragments. The polymer chains within the capillary act as obstacles, making smaller fragments move through the capillary faster than larger ones, and it follows that the molecules are separated based on their size. The DNA molecules are then analysed and excited by a laser, as they pass by a detection window, resulting in an electropherogram. Furthermore, fluorescent-dyes are normally added making it possible to analyse multiple loci of similar length simultaneously.

The electropherograms produced are then compared to an allelic ladder (an artificial mixture of common alleles present in the human population for the chosen STR markers; they are generated with the same primers as the sample tested and hence serve as a reference DNA size for every allele included in the ladder) to create a DNA profile of the sample. Ideally this profile (in a non-mixture scenario) contains one (homozygous case) or two (heterozygous case) alleles and equal coverage (the amount a given allele is represented in the sample) of said alleles on any STR loci used. However, this is not an ideal world; the profile can suffer from stutters, drop-ins, drop-outs, and pull-ups, as well as other artefacts. We will only consider the artefacts concerned with the PCR process. Furthermore, the alleles produced by an electopherogram are obtained as the number of repeated tetranucleotides only. It does not take into account that a 7-repeat could be a compound repeat sequence of 4 then 3 or 5 then 2 (other combinations of 7 are also permitted), the resolution is simply not high enough to make that kind of distinction, which is one of the reasons why we turn to next-generation sequencing.

1.4 Artefacts of PCR Amplification

A stutter (in literature also referred to overstutter) is a peak at position $n - j$ bp, for $j \in \mathbb{N}\backslash\{0\}$, where a true allele (or parent allele) was observed at n bp. However, when we say stutter, it will refer to a peak at $n -$ motif length bp, hence a double stutter will refer to $n - 2 \cdot$ motif length bp, and a half stutter $n - 0.5 \cdot$ motif length bp (remember that when we say motif length we assume it to be a tetranucleotide). The peak found at a stutter position will have reduced coverage to that of its parent allele (generally the stutter to parent coverage ratio is around $5-10\%$). Backstutters (also referred to as a forward-stutters) can also occur, however they are not as likely, and are defined in a similar manner, found at $n +$ motif length bp. We can then define double and half backstutters as before.

Allele drop-ins (false positives) occur as additional alleles that are observed in the DNA profile. An allele drop-in can be from sporadic contamination of the sample DNA, which gets amplified during the PCR process along with everything else. That is, drop-ins may not be a direct artefact of the PCR process itself.

Allele drop-outs (false negatives) presents as an allele in the sample that fails to amplify, either at all or sufficiently enough to exceed a pre-set detection threshold. Null alleles, manifest in a similar way to allele drop-outs, occur because of a mutation in the primer-site used to amplify the sample, the consequence being that the Taq-enzyme cannot bind to the DNA fragment and therefore the fragment fails to amplify.

Heterozygote (peak) imbalance, one allele is amplified preferentially compared the other, creating a difference in peak heights between two alleles. This imbalance will occur more often in case with low amounts of template DNA, due to binomial sampling [4]. Furthermore, the shorter of two alleles on an heterozygous locus is more efficiently amplified than the larger fragment, called preferential amplification. Preferential amplification is caused by either enzyme replication failure or increased stuttering of longer fragments, as shown in [5, 6] and in Chapter 3.

1.5 Next-generation Sequencing

Next-generation sequencing (also called second-generation sequencing (SGS) or multiple parallel sequencing (MPS). The name next-generation sequencing when describing second generation sequencing, might be a bit misleading, as third-generation sequencing (TGS) has already been developed [7]) was first introduced in 2003 by the 454 company (now Roche-454). It allows rapid, high-throughput sequencing of short sections of DNA. NGS offers the resolution of, which CE does not, DNA base resolution. This implies that by using NGS instead of CE to create a DNA profile, it is possible to obtain a profile with higher discriminating power.

NGS has two major applications de novo sequencing and resequencing. In de novo sequencing the genome of an organism is sequenced for the first time. Whereas, in resequencing projects, the genome, or parts thereof, are sequenced only where a reference sequence is available. The resequenced samples are then aligned to the reference using an alignment tool. In forensics genetics resequencing is the most common approach [8]. However, alignment is not always necessary.

Generally alignment in the case of STR regions is quite difficult, as the regions by nature are highly repetitive (alignment in general can be difficult) and if the situation is as seen in Figure 1.5 (i.e. we do not see both the forward and reverse primers or flanking regions), we do in general not know how many repeats occur in between. In order to avoid this situation we will stick to regions, where the total bp length (from the beginning of the forward primer to the end of the reverse primer) will be less than the bp maximum of the machine used.

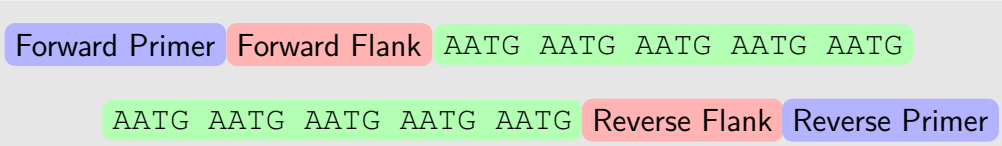


Figure 1.5: Two reads with the same repeat pattern, but only containing, either the forward or reverse primer, of the same locus.

There are a number products on the market that achieves this high-throughput sequencing and they achieve it in slightly different, yet similar ways. The machines available to the forensic geneticists at the University of Copenhagen are the *Roche-454* (454), *LT-IonTorrent PGM* (IonTorrent), and Illumina MiSeq (Illumina). Their

maximum bp read lengths are 600, 400, and 2×250, respectively. In general the work flow for NGS can be seen in Figure 1.6 (inspired by [9, Figure 2]). That is, the DNA template is prepared for use, a library is build of the sample, DNA strings are individually amplified (clonal amplification), and the DNA is sequenced. In order to show how the methods differ, we will add a little more detail.

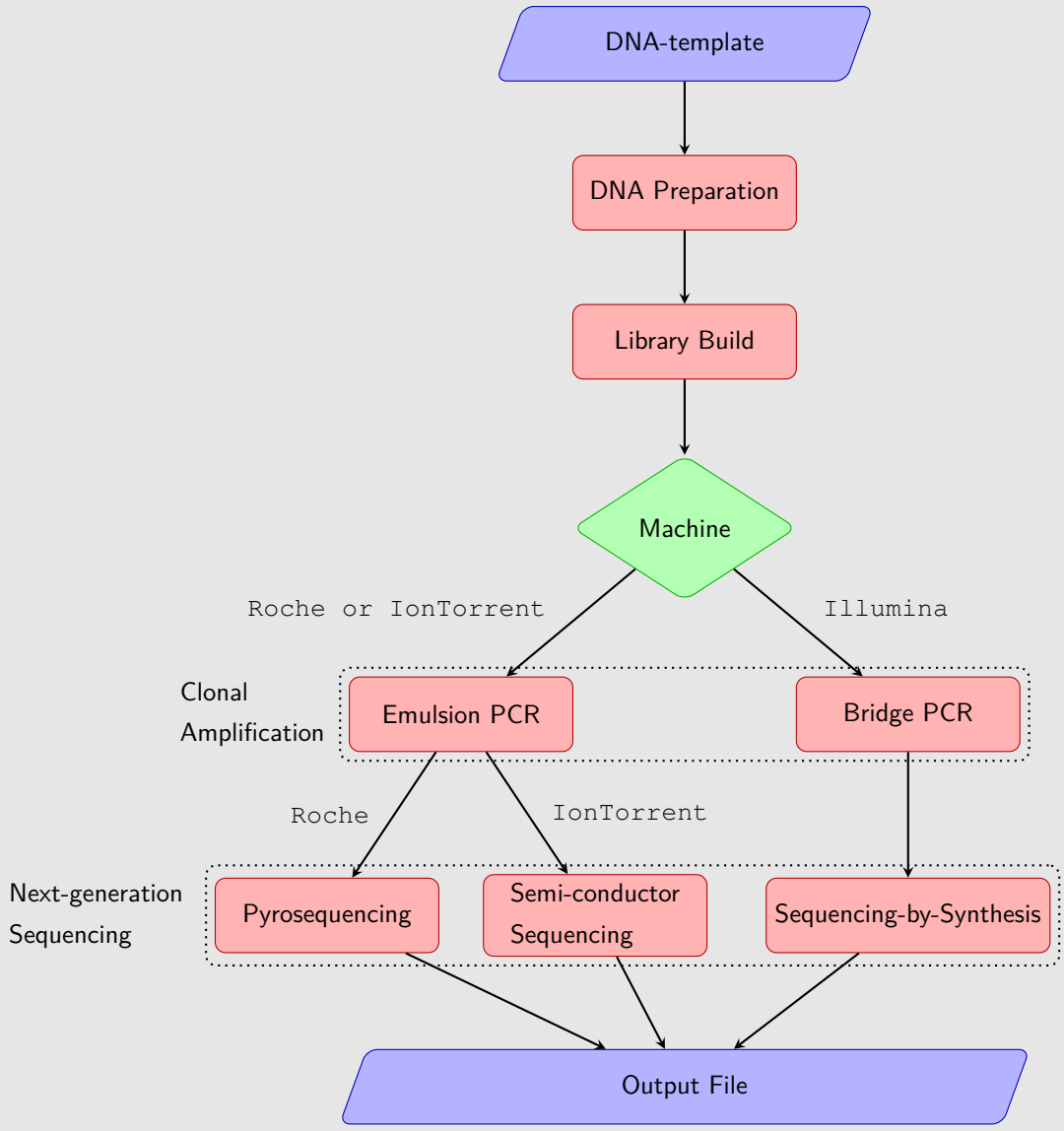


Figure 1.6: A NGS work flow, given the Roche, IonTorrent, or Illumina, sequencing machines.

1.5.1 Roche-454

In the 454, the DNA molecules are amplified inside a water-in-oil emulsion (the mixture of two or more liquids that are normally immiscible) PCR. Each water droplet acts as a microreactor containing the PCR reagents and ideally a single primer-coated bead with a single DNA fragment. Hence, multiple PCRs can be performed simultaneously. After the emulsion breaks, the beads are covered in thousands (or millions) of copies of the original DNA fragment. The beads are then placed in picoliter-volume wells containing the sequencing enzyme (DNA polymerase). The 454 sequencing process is a parallelised version of *pyrosequencing* (parallelised as it can be run on each bead) [8, 10].

The pyrosequencing approach uses unmodified A, T, G and C − dNTPs and add them sequentially to the growing complementary DNA strand. If a complementary dNTP is introduced to the next unpaired nucleotide in the original DNA strand, it is incorporated (pyrophosphate and H^+ ions are released) into the complementary strand by the DNA polymerase. In the case of homopolymer repeats, multiple nucleotides will be incorporated in a single cycle, which leads to a larger amount of pyrophosphate (and hydrogen ions) being released. Through an enzymatic process light is emitted (the amount of light emitted is proportional to the amount of pyrophosphate released), and is then recorded in a pyrogram, as shown in Figure 1.7.

1.5.2 Life Technology IonTorrent PGM

The IonTorrent is based on the *Ion-semiconductor sequencing* (ISS) method [11]. The technique is similar to that of pyrosequencing; microwells on a semiconductor chip containing multiple copies of a single template DNA strand (obtained through PCR), are sequentially flooded with unmodified A, T, G and C dNTPs. However, the techniques differ in their detections method. ISS, instead of detecting the pyrophosphate, uses a hypersensitive ion sensor to detect the release of the hydrogen ions. As in pyrosequencing, the amount of hydrogen ions released corresponds to a proportionally higher electronic signal produced by the ion sensor.

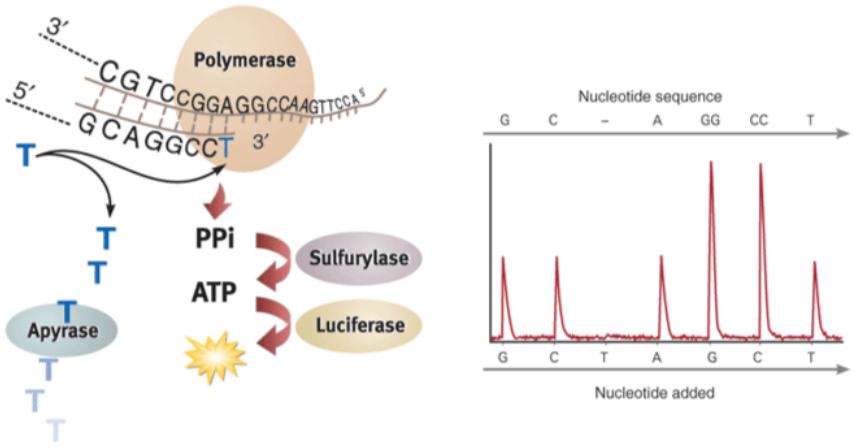


Figure 1.7: Pyrosequencing uses unmodified dNTPs, and try to add them sequentially to growing complementary DNA strand. If a dNTP is incorporated pyrophosphate is released, through an enzymatic process light is emitted, and then recorded as a pyrogram.

1.5.3 Illumina MiSeq

The MiSeq is based on Illuminas TruSeq sequencing method, sequencing-by-synthesis method [12]. The DNA molecules (and primers) are attached to a slide and amplified using PCR, to form DNA clusters. A single fluorescently labelled dNTP is added to the nucleic chain. Each nucleotide label then servers as a terminator for polymerisation, i.e. after a nucleotide is incorporated, the fluorescent dye is imaged in order to identify the base. The dye is then chemically removed and the next cycle begins [13]. A schematic of the process is shown in Figure 1.8.

All three methods are still depended on PCR amplification to build the library of the DNA template, i.e. it follows that they will all suffer from stuttering as explained above. Furthermore, NGS is, as CE or PCR, not perfect and suffers from misreads, which is why a quality score is computed for each called base. Misreads are just one artefact of the NGS process, we will introduce artefacts as they present themselves through the ongoing data analysis. A more comprehensive review of NGS's applications in forensic genetics can be found in [9].

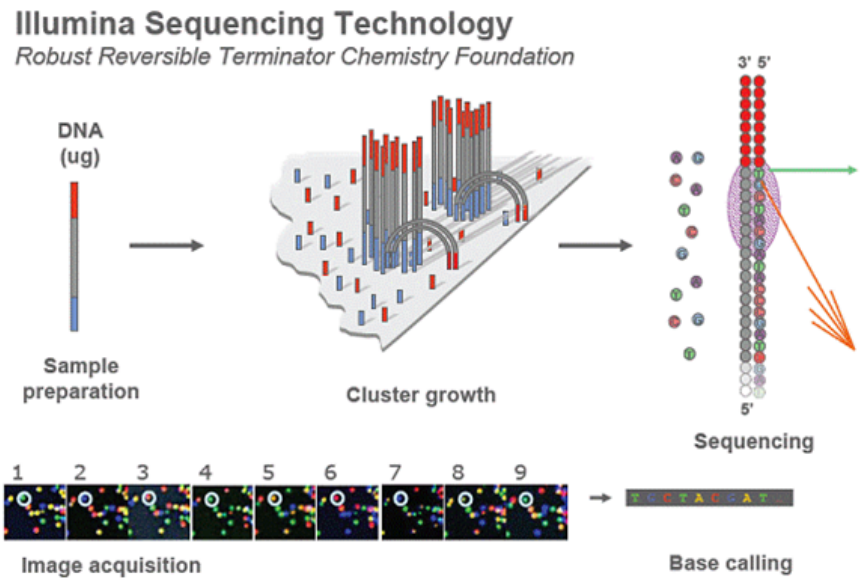


Figure 1.8: A diagram describing Illumina's TruSeq sequencing method. The sample DNA is attached to a slide and amplified to form clusters. Fluorescently labelled dNTPs are added to the growing complementary DNA strand and the fluorescent dye is imaged in order to identify the base.

1.5.4 Data Management

Data analysed in this thesis are sequenced using the IonTorrent, the 454, and Illumina MiSeq. The data in general comes in two parts:

(i) **Dilution series:** The dilution series is a succession of sequencing runs halving the sample DNA with each run. Starting at 2ng and ending at 0.05ng, from two different donors, F and H. Donors for which we know the exact repeat sequence, alleles, and stutters for every loci used (these true values are seen in appendix E). Machines used: IonTorrent PGM.

(ii) **Reference files:** In the case of the reference files we do not know the true profile. Machines used: Ion PGM, Illumina MiSeq, and the Roche 454.

One set of loci used for the IonTorrent data includes CSF1PO, TPOX, D3S1358, D5S818, D16S539, D7S820, D8S1179, TH01, vWA, and AMELX/AMELY, also called the STR-10plex. The 454 is generally used for loci that tend to have longer alleles, such as locus D12S391 or D21S11.

The output format of the 454 is a *FASTA* (`.fna`) file and a *quality* (`.qual`) file, while the IonTorrent and Illumina Systems produces what is called a *FASTQ* (`.fastq` or `.fq`) file (a combination of FASTA and quality file). Furthermore, some of the IonTorrent reference files come in a `.bam`-format.

The FASTA format is simply a string, where each character represents a nucleotide. The quality file stores a quality score for each nucleotide in the sequence. FASTQ stores both a sequence and its quality score. Each sequence and corresponding quality score is represented using four lines as follows:

@title (plus optional description)
Sequence line
+optional repeat of title
Quality line

The first and third line is used as identification, optionally they may also contain a description of the sequence, i.e. the locus, the length of the sequence, et cetera. The second and fourth line contains, the sequence in a FASTA format and the quality of the sequence, respectively. The quality score is calculated in different ways depending on the machine used to sequence the DNA. The quality scores will be defined in section 2.1. Until then, we will not concern ourselves with the quality of the read, just the read itself.

1.5.5 Accessing the data with R

The FASTQ format can easily be read into **R** using the `ShortRead` package (more specifically the **readFastq**-function) available through Bioconductor [14]:

```
> LT_Dil_002_F_2ng <- readFastq(file, quality="Auto")
> LT_Dil_002_F_2ng
class: ShortReadQ
length: 286339 reads; width: 8..360 cycles
```

The nomenclature used in the above code breaks down as follows:

LT: Life Technology (makers of the IonTorrent).

Dil: Dilution; it is part of a dilution series.

002: its a priori designated number.

F: the donor of the DNA.

2ng: the amount of sample DNA used.

The **readFastq**-function creates a `ShortReadQ` object containing the sequenced strings, the length of each string (called width) and corresponding quality of the sequences, as well as an object containing the title attached to each sequence. They are accessed using the **sread**, **width**, **quality**, and **id** functions respectively.

```
> sread(LT_Dil_002_F_2ng)[1:5]
  A DNAStringSet instance of length 5
    width seq
[1]    11 CTATCATCCAT
[2]    11 CTATCATCCAT
[3]    11 CTATCATCCAT
[4]    11 CTATCATCCAT
[5]    32 CCACTGGGCGACAGAGTGAGACTCAGTCTCAA
> quality(LT_Dil_002_F_2ng)[1:5]
class: FastqQuality
quality:
  A BStringSet instance of length 5
    width seq
[1]    11 >:9::/,40,,
[2]    11 ;658>4++(++
[3]    11 /66;;8//(./
[4]    11 955//++2'++
[5]    32 /(////;/=>A???AA@AAC<:::/,,,,,,
```

The objects created by these functions are **DNAStringSet** and **BStringSet** objects (excluding the **width** function) belonging to the `BioStrings` package (also found on Bioconductor). The `BioStrings` package contains functions optimized for working with DNA/RNA strings and will be utilized throughout the thesis.

The Roche files contains 24 profiles each, we load them by first setting the directory, using the `RochePath`-function, to a pair of (`.fna`,`.qual`) files (Note that each folder should only contain one `.fna` file and corresponding `.qual` file). Once the directory is set, there are multiple function that can be used to read from a RochePath object,

we will use the `read454`-function, as it creates a `ShortReadQ` object exactly like the **`readFastq`**-function. In order to identify the 24 samples, we use the multiplex identifiers (MID), included with each file, creating 24 `ShortReadQ` objects per Roche directory. However, for the most part, the data used in this project will be from the Ion Torrent PGM, as Roche has announced that support for the 454 will be discontinued from 2015.

Finally, the `.bam` files; the `.bam` file format is a binary format for storing sequencing data. Each file contains 16 profiles, in order to separate them into `.fastq` files, we have used the SamToFastq java command line tool from Picard [15, Samtools]. The SamToFastq tool can look in the header of the `.bam` file and separate the file accordingly. After the files are separated into `.fastq` files, we can load them into **`R`** as described above.

1.6 DNA Profiling

DNA evidence used in forensic genetic case work can be split into two categories: DNA obtained at crime scenes and paternity/reunification cases. The creation of a DNA profile is normally more difficult in the case of the former, as the sample may only contain small amounts of DNA (referred to as low template DNA), DNA from multiple contributors (DNA mixtures), or the DNA may suffer general contamination or degradation. The analysis of a DNA profile will fall into one of two types: evaluating the strength of evidence for one hypotheses over another and deconvolution of a DNA profile. In general the strength of evidence, comparing one hypothesis, in legal casework the prosecutions hypothesis, $\mathcal{H}_p$, versus hypothesis of the defence, $\mathcal{H}_d$, is usually represented by the likelihood ratio:

$$\text{LR} = \frac{L(\mathcal{H}_p)}{L(\mathcal{H}_d)} = \frac{\mathbb{P}(\mathcal{E}|\mathcal{H}_p)}{\mathbb{P}(\mathcal{E}|\mathcal{H}_d)}, \tag{1.1}$$

where $\mathcal{E}$ is the evidence. In e.g. a paternity case, $\mathcal{H}_p$ would claim a man, K_1, as the father of the child, while $\mathcal{H}_d$ claim that either a different man, K_2, or an unknown man, U, as the father. The numerator and denominator can be decomposed as $\mathbb{P}(\mathcal{E}|\mathcal{H}) = \sum_{\boldsymbol{g}} \mathbb{P}(\mathcal{E}|\boldsymbol{g})\mathbb{P}(\boldsymbol{g}|\mathcal{H})$, where $\boldsymbol{g}$ is given genotype, i.e. we sum over all possible genotypes, or combination of genotypes. It follows that the sum can become very large. The probabilities of interest, in the decomposed numerator/denominator, are the probability of evidence given the genotype, $\mathbb{P}(\mathcal{E}|\boldsymbol{g})$, as $\mathbb{P}(\boldsymbol{g}|\mathcal{H})$ is provided.

Deconvolution of a DNA mixture is the identification of DNA profile(s) of one or multiple unknown contributors to the mixture. That is, we try to extract likely DNA profiles of potential perpetrators from a mixed sample (DNA profiles for unknown contributors found in this manner can then be run against a DNA database).

As the application of NGS is very new in this context, we will start at the beginning. That is, in the remainder of this section, we consider the identification of loci and alleles, examine the reverse complement, define a simple method creating DNA profiles for uncontaminated single contributor sample, and we will end this chapter by more clearly defining the objective of this thesis.

1.6.1 Identifying Loci and STR regions

The first challenge is to identify the loci, the alleles on each locus, and the allele coverage. To identify the locus, L, of a given read $S_i = \{b_{i1}, b_{i2}, ..., b_{in_i}\}$, we will use the flanks seen in Table 1.1. These flanks have the property of being directly adjacent to the repeated regions (i.e. there is no offset between the flanks and the STR region), with the exception of the flanks for D1S1656 and PentaD. The flanks were identified using the STRait Razor Perl software [16].

We will only consider reads where both the forward and reverse flanks are identified, as well as reads, where a given locus is not uniquely identified (i.e. more than one locus has been marked as a match for a given read, implying that the read is too noisy for further processing). It thereby follows that a locus L can be defined as:

$$L = \{S \mid F_{\mathcal{F}}, F_{\mathcal{R}} \in S, F_{\mathcal{F}} \prec F_{\mathcal{R}}, S \notin \tilde{L},\ \forall\ \tilde{L} \neq L\} \tag{1.2}$$

The condition $F_{\mathcal{F}} \prec F_{\mathcal{R}}$, seen in the definition, is used to indicate that the last character of the forward flank, $F_{\mathcal{F}}$, is seen before the first character of the reverse flank, $F_{\mathcal{R}}$.

As both strands of the sample DNA are sequenced, we should also consider the so called reversed complements of the flanks. The **`reverseComplement`**-function from `BioStrings` makes is easy to create these reverse complements from Table 1.1. However, for now we will ignore the reverse complements, approximately halving the amount of DNA we would otherwise use.

We see that the gender specific sequences AMELX/AMELY are not a part of Table

1.1 and is therefore not considered (bringing us down to nine loci in the 10plex). Furthermore, D1S1656 and PentaD are not a part of the 10plex (if they were included, we would have to identify the repeated sequence within the region enclosed by the corresponding flanks), i.e. it follows, that when trying to find alleles in the IonTorrent data, we can then divide the length of the identified reads by the motif length (this is allowed as all loci in the 10plex have the direct adjacency property) and thereby calculate an allele frequency table for each locus.

Table 1.1: Autosomal STR loci detected by STRait Razor as seen in [16, Table 1]. The flanking sequences are directly adjacent to the repeated region, with the exception of the flanks for D1S1656 and PentaD.

Locus	Forward Flank	Reverse Flank
CSF1PO	GATAGATAGATT	AGGAAGTACTTA
TPOX	GAACCCTCACTG	TTTGGGCAAATA
D2S441	TCTATGAAAACT	TATCATAACACC
D3S1358	AGGCTTGCATGT	ATGAGACAGGGT
D5S818	ATTTATACCTCT	TCAAAATATTAC
D13S317	AGATGATTGATT	ATGTATTTGTAA
D18S51	TCCTCTCTCTTT	GAGACAAGGTCT
D16S539	GACAGACAGGTG	TCATTGAAAGAC
D7S820	GAACGAACTAAC	GACAGATTGATA
D8S1179	CACTGTGGGGAA	TACGAATGTACA
TH01	CCCTTATTTCCC	TCACCATGGAGT
vWA	GACTTGGATTGA	TCCATCCATCCT
D21S11	ATAGATAGACGA	AGGCAATTCACT
FGA	GAAAGGAAGAAA	CTAGCTTGTAAA
D2S1338	GGATTGCAGGAG	AGGCCAAGCCAT
D19S433	AAGATTCTGTTG	AGAGAGGTAGAA
PentaD	TTTATGATTCTC	TTGAGATGGTGT
PentaE	TCCTTACAATTT	GAGACTGAGTCT
D10S1248	TATTGTCTTCAT	ACTCACTCATTT
D12S391	AAATCCCCTCTC	ACCTATGCATCC
D1S1656	TAAACACACACA	CATCATACAGTT
D22S1045	TATTTTTATAAC	GAGACTACTATC

Locating the flanking regions has been implemented in **R**, an implementation based on a function made by Mikkel Meyer Andersen. The function outputs the sequence and quality-reads matched to each loci and versions trimmed to only included the region between the flanks (from this point forth the notation S_i will indicate such a string).

In general we can then define a given allele as a realisation of the stochastic variable L. We will denote an allele as $\mathcal{A}_j(L)$ or $\mathcal{A}_j$ if the locus is either not important or clear from the context. The frequency of and allele $\mathcal{A}_j$ is then given by $|\mathcal{A}_j|$. Figure 1.9 shows the frequencies corresponding to locus D16S539, for the `LT_Dil_002_F_2ng` sample.

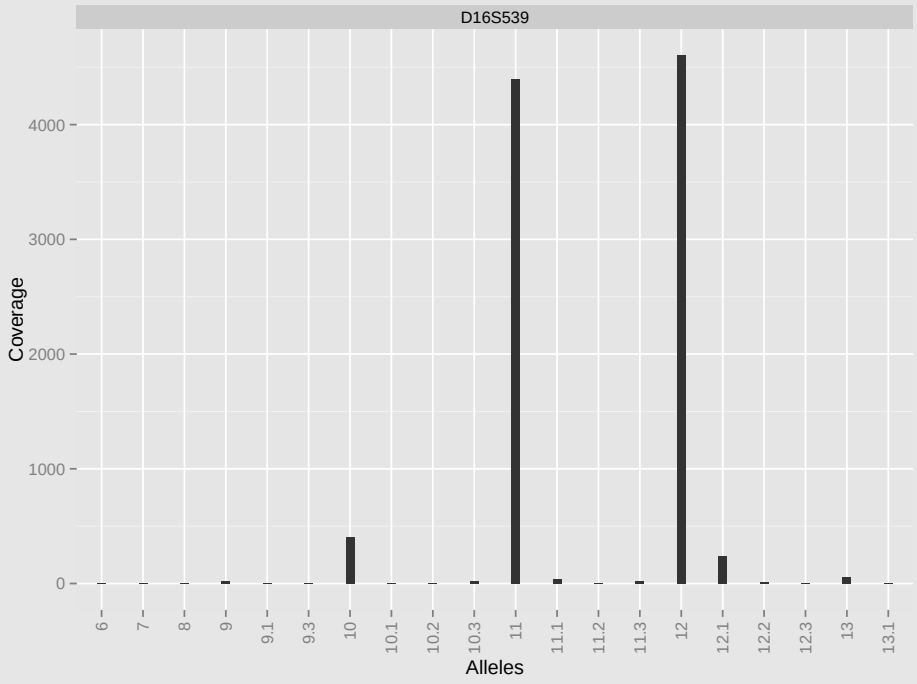


Figure 1.9: A histogram representing the frequency table of the D16S539 locus for the `LT_Dil_002_F_2ng` sample.

Looking at Figure 1.9, we would, based on frequency, also called the coverage, guess that F has alleles at 11 and 12 on the D16S539 locus (which is in agreement with the true values). To obtain the remaining genotypes in a similar manner, we need to account for whether the genotype is heterozygous or homozygous. In the case of sample `LT_Dil_002_F_2ng`, we see in Figure 1.9 that the allele coverage of the true alleles is very high, compared to the remaining candidates. That is we only accept candidates with high coverage. The DNA-profile for sample `LT_Dil_002_F_2ng`

can be seen in Table 1.2.

Comparing the genotypes in Table 1.2 with the true values, we see that they are all the same with the exception of vWA, which according to Table E.1 should be genotype 15,18 (there is in fact no error, it comes down to nomenclature, see e.g. STRBase [17], we will in this thesis refer to its actual length and account for the difference through-out our implementations). Furthermore, from this point onwards all loci, outside gene coding regions, will be referenced using their chromosome designation only, i.e. D16S539 becomes D16.

Table 1.2: The DNA profile for sample `LT_Dil_002_F_2ng` determind based on the frequency of alleles at each locus.

	CSF1PO	TPOX	D3S1358	D5S818	D16S539	D7S820	D8S1179
Allele 1	10	8	14	9	11	8	12
Allele 2	12		17	12	12	10	13
	TH01	vWA					
Allele 1	7	17					
Allele 2	9.3	20					

The Reverse Complement

Up until this moment we have ignored the reverse complement and as evident from Table 1.2 it was easy enough to find the DNA-profile, without using the reverse complement strings.

That being said, in some cases when not including the reverse complement strings we can create a huge allele imbalance, as seen in panel (a) of Figure 1.10, showing the D3 locus for contributor H. H should have a 17,18 genotype on this locus, yet the coverage of allele 18 is very low. However, looking at panel (b) of the same figure, we see that by including the reverse complement, the coverage of allele 18 increases dramatically.

In Figure 1.11, we have split each allele into its reverse complement and non-reverse-complement, to see what proportion of the alleles we can attribute to the non-

reverse-complement and reverse complement respectively. In the case of allele 17 the percentage of reverse complement reads (RC%) is 4.5%, while allele 18 has an RC% of 94%.

It follows that in order to avoid creating artificial allele imbalance we should, and will with the exception of Chapter 2, always include the reverse complement.

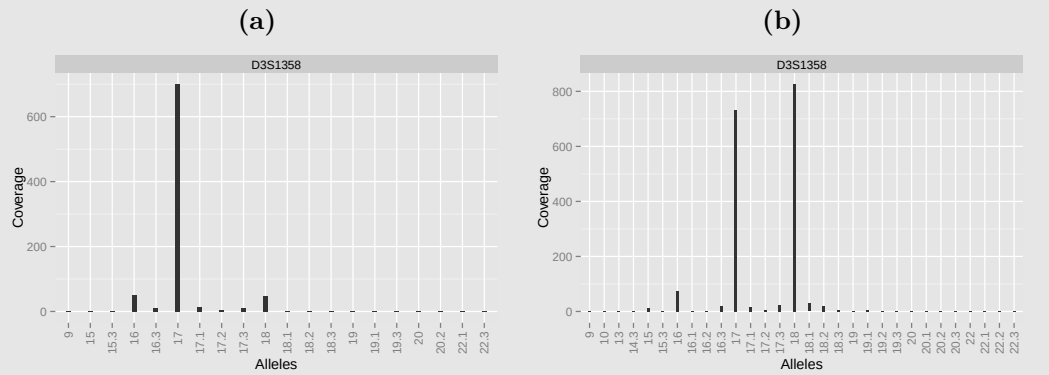


Figure 1.10: A histogram representing the frequency table of the D3 locus for the `LT_Dil_020_H_2ng` sample, where the reverse complement strings were not included and included in panel (a) and (b) respectively.

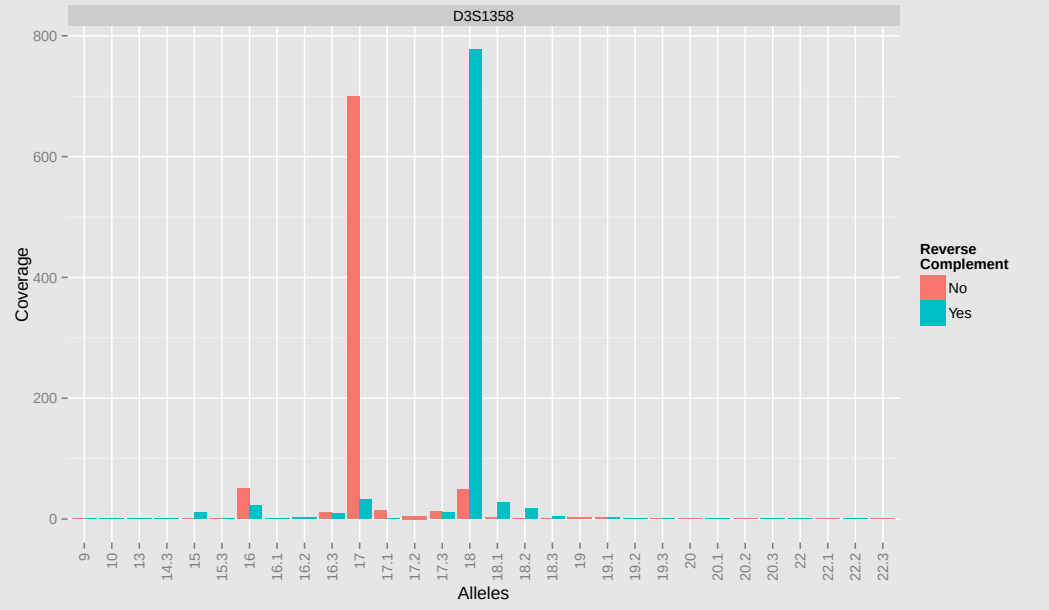


Figure 1.11: A histogram representing the frequency table of the D3 locus for the `LT_Dil_020_H_2ng` sample. The histogram includes both the non-reverse-complement strings, coloured in red, as well reverse complement strings, coloured in blue.

1.6.2 Single Contributor Samples

If we know that the DNA sample contains a single contributor and is otherwise uncontaminated, identification is fairly simple, even with small amounts of input DNA (in this case 50pg). As we see in Figure 1.9 the true alleles will stand out. That is, all we really need to figure out is whether or not a locus is hetero- or homozygous. Therefore, we introduce a heterozygosity threshold, t_H, a real number on the interval $[0, 1]$, which we multiply by the maximum coverage on a given locus. The DNA profile, seen in Table 1.2, is in fact created using $t_H = 0.5$. We have no justification for choosing a threshold of 0.5 and we will therefore calibrate the threshold.

Calibrating the Heterozygosity Threshold

We will examine the drop-in and drop-out rate for a series of thresholds ranging from 0.01 to 1. Figure 1.12 shows the drop-in and drop-out rate for every potential threshold, stratified on the amount of initial DNA (ng) used.

To find the optimal threshold, we would like to minimise the number of false positives (FP) and false negatives (FN). This is part of decision theory, known as minimising competing goals. The easiest way to achieve this goal, is to enforce a constraint on one of the variables and then minimise w.r.t. the other, i.e. minimising FN subject to FP $\leq \alpha$. It can be shown, see [18, Appendix A], that the optimal threshold is the first instance where FP is smaller than α. These thresholds can be seen, marked in black, in Figure 1.12, for $\alpha = 0$. We choose to constrain FP in this manner, as a drop-in would be more difficult to explain, than a drop-out. In order to understand why, we first need to examine how a drop-in can occur:

(i) **Contamination:** Two or more observed alleles that come from a single individual.

(ii) **Allele drop-in:** Two or more observed alleles that come from a multiple individuals.

The problem is that these two are indistinguishable from one another. That is, when a drop-in occurs we will not know whether we are working with a contaminated profile or a mixture of multiple profiles.

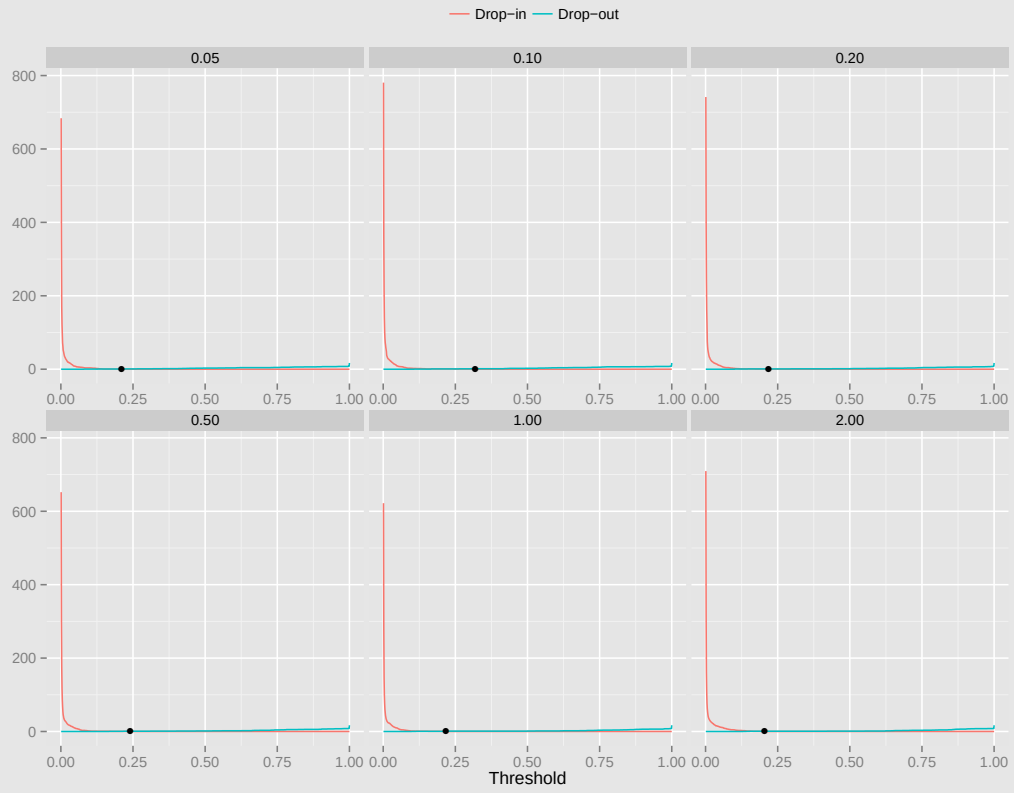


Figure 1.12: The drop-in and drop-out rates, stratified on the amount of DNA used in the initial PCR amplification. The black dot represents the first threshold, where the drop-in is equal to zero.

The overall heterozygosity threshold, t_H, will be taken to be the maximum of these thresholds rounded to the nearest 0.05, which in this case implies $t_H = 0.35$.

1.7 Obstacles, Objective, and Overview

It is readily evident that the method presented above would not work in the case of multiple contributor samples. Even a contamination causing a large drop-in would pose a problem (the drop-in would not even have to be that big just larger than 0.35 times the maximum allele coverage). That is, we need a more comprehensive model to assess $\mathbb{P}(\mathcal{E}|\boldsymbol{g})$. In order to do so, we need to examine the process with which the data was generated.

First off all, we know that the NGS work flow still relies on the PCR amplification,

when building the library for sequencing. Therefore, it follows that our data will suffer the same fate as data generated using CE, i.e. stutters.

Second off, until now we have treated the NGS process as being error free. However, we know that this is not true. The errors introduced during the sequencing process (whether it is do to the actual sequencing or the emPCR process is not entirely clear) can be broken down as follows:

List 1.7.1

(i) A base is miscalled

(ii) A base is skipped (or deleted)

(iii) A base is inserted

Errors of the type, seen in List 1.7.1 item (ii) or (iii), are both referenced to as indel's (a combination of the two words *in*sertion and *del*etion). The effect of item (i) can be seen in Figure 1.13, where we see the coverage of the observed strings with repeat length 12 of locus D16, shown in order of prevalence. We see an abundance of one particular string and a sea of different strings of similar length.

The effect of item (ii)-(iii) can be seen in Figure 1.9, as e.g. the small top seen at allele 12.1. Even though this type of error is, as mentioned, called an indel, we will refer to this particular phenomenon as a left or right shoulder depending on whether the peak is seen on the left or right hand side of the allele, respectively (in this particular case 12.1 is the right shoulder of 12).

The main objective of this thesis will not be to assess $\mathbb{P}(\mathcal{E}|\boldsymbol{g})$, primarily because NGS is still in its infancy. That is, not a lot of groundwork has been done trying to account for the errors arising, new or old, within the NGS framework. Our primary objective is to lay the foundation for future work in paternity and crime scene cases using NGS. The general structure of the thesis looks as follows.

I will look further into the quality of these erroneous reads in Chapter 2, in order to assess if the quality can be used to handle such reads. The relationship between stutters or shoulders and the parent allele (the allele from which they were created) is investigated in Chapter 3, to confirm the hypothesis that the longest uninterrupted

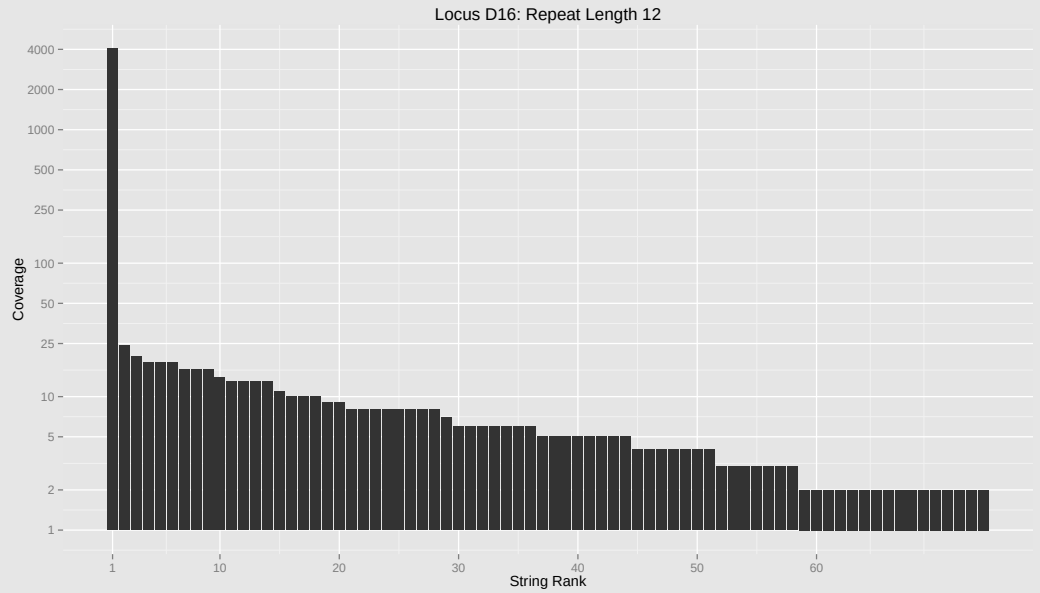


Figure 1.13: The coverage of unique strings with repeat length 12, of locus D16, shown in order of prevalence (from most to least prevalent). Furthermore, the ordinate-axis is shown on $\log_{10}$-scale.

stretch, as a predictor for this relationship. That is, I will consider the systematic noise before I cast a light on the more general noise. In Chapter 4 I examine the general noise generated by the NGS process, with the aim of removing non-systematic noise from our data. A consequence of the methods of removal developed in Chapter 4 is that allele drop-outs are introduced. The probability of a drop-out occurring is considered in Chapter 5, where I also impose a distribution upon the coverage. Finally I have, in Chapter 6, added a few concluding remarks, including comments on refinements, extensions, and future work.

CHAPTER

TWO

QUALITY ANALYSIS

Sequencing is not perfect, which is why a quality score is assigned to the read of every base. We will start by identifying all unique strings, $\mathcal{U}_j$, of every allele, $\mathcal{A}_j(L)$, on a locus L (i.e. $\mathcal{U}_j \subseteq \mathcal{A}_j(L)$). For each string $u \in \mathcal{U}_j$, we define indices for the strings corresponding to u as $\mathcal{I}_u = \{i \mid S_i \in \mathcal{U}_j, S_i = u\}$ and the set of strings as $\{S_i\}_{i \in \mathcal{I}_u}$, or $\mathcal{S}_u$ for short.

If we think back to Figure 1.13, showing the unique strings of allele 12 on locus D16, we see a high prevalence of one particular string, and then a lot of strings, with very low coverage. Figure 2.1 shows the top and bottom four, most prevalent strings, seen in Figure 1.13.

A way we could handle these erroneous reads, would be to set a coverage threshold, treating strings with coverage lower than said threshold as noise, removing them from further analysis. Doing so, however, we could end up removing a lot of data, depending on the threshold. If the threshold is, e.g. 25 we would on allele 12, seen above, lose 572 reads. In situations with low coverage across an entire locus, that might be devastating. Another way to handle such reads, would be to use the quality of a sequence as a weight when defining the measure used to access potential alleles. However, first we formally introduce quality scores.

```
                                                    String Coverage
GATAGATAGATAGATAGATAGATAGATAGATAGATAGATAGATAGATA      4031
GATAGATAGATAGATAGATAGATAGATAGATAGATAGGTAGATAGATA        24
GATAGATAGATAGATAGATAGATAGATAGATAGATAGATAGGTAGATA        20
GATAGATAGATAGGTAGATAGATAGATAGATAGATAGATAGATAGATA        18
...
GATAGATAGATAGATAGATAGATAGTTGGATAGATAGATAGATAGATA         1
GTTAGATAGATAGATAGATAGATAGATAGATAGATAGATAGATGGATA         1
GATAGATAGATAGATAGATAGATAGATAGATAGATAGATGGATGGATA         1
GATAGATAGATAGATAGATAGATAGATGGATAGATAGATAGATATATA         1
```

Figure 2.1: The top and bottom four, most prevalent strings, and their coverage. The differences between the most prevalent string and the remaining strings, are shown marked red. Furthermore, the corresponding *true* base calls are marked as blue in the most prevalent string.

2.1 Quality Scores

The quality score can be represented in different ways depending on the typing technology. In general there are two different ways the quality can be calculated, the *Phred* and *Solexa* methods [19]. The typing machines used for this thesis uses the Phred method, and therefore we will only formally introduce that method.

Definition 2.1.1
The Phred quality score (PQS), Q^{Phred}, is defined using the estimated probability of error, P:

$$Q^{\text{Phred}} = -10\log_{10}(P) \tag{2.1}$$

The probability of error is usually estimated by a pre-computed lookup-table, which is distributed with the machine in question. The estimated probability of error uses n (in the case of the IonTorrent PGM, $n = 6$ [20]; Ewing and Green, introduced an algorithm for creating such a lookup-table, suggested $n = 4$ [21]) predictors of local quality, to find the corresponding quality score in the lookup-table [20–22]. In the case of Ion Torrent, P_2 and P_5, are based on the noise of the surrounding bases, creating a kind of moving average across the quality. Furthermore, P_1 and

P_3, penalise the residual, i.e. the difference between the actual and predicted flow values [20].

The *Solexa quality score* (SQS), Q^{Solexa}, is given in a similar manner to that of PQS:

$$Q^{\text{Solexa}} = -10\log_{10}\left(\frac{P}{1-P}\right). \tag{2.2}$$

It follows from definition 2.1.1 that if a base is assigned a PQS of e.g. 30, the probability of that base being called incorrectly is 10^{-3}, in fact a quality score of $j \times 10$, corresponds to 10^{-j} probability of error, for every j. The full range of the quality (depending on the score and encoding type) can be seen in Table 2.1.

Even though the two quality scores differ as shown in Figure 2.2, they are equivalent and conversion between the two is fairly easy, as we see in the following remark.

Remark 2.1.2

(1) Given a SQS it can be converted to a PQS as follows:

$$Q^{\text{Phred}} = 10\log_{10}\left(10^{Q^{\text{Solexa}}/10} + 1\right) \tag{2.3}$$

(2) Given a PQS it can be converted to a SQS as follows:

$$Q^{\text{Solexa}} = 10\log_{10}\left(10^{Q^{\text{Phred}}/10} - 1\right) \tag{2.4}$$

Furthermore, it is not just the scoring methods that are different, their encoding methods differ as well. Ideally the quality scores are stored as a single character per base. The three standard encoding types are called *Sanger*, Solexa (and early Illumina) and Illumina (as Illumina changed from using SQS to PQS, they changed encoding style as well). All three methods are encoded using ASCII. The range of said encoding methods are seen in Table 2.1, as well as the corresponding quality range.

Even though encoding the quality in this manner might seem unnecessary, it is a way to store what might be double digit number using only one character. Note that as the quality can only be integers we have discretised the probability of error P.

Table 2.1: The encoding and quality ranges for the three encoding methods.

	ASCII Range	Quality Range
Sanger	$(33, 126)$	$(0, 93)$
Solexa	$(59, 126)$	$(-5, 62)$
Illumina	$(64, 126)$	$(0, 62)$

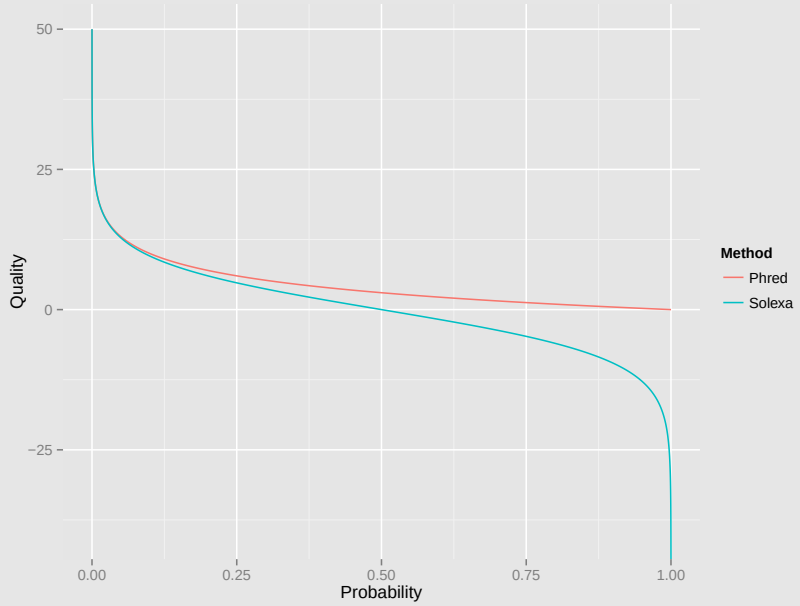


Figure 2.2: The behaviour of Phred and Solexa quality scores given the estimated probability of error.

2.2 Quality Assessment of Strings

We will start by assessing the difference in quality, between the most common variants of a specific allele, by observing the quality of an entire sequence. More precisely given a trimmed sequence read (i.e. a sequence containing only the STR region) $S_i = \{b_{i1}, b_{i2}, ..., b_{in_i}\}$ with corresponding quality scores $\mathcal{Q}_{i,*} = \{q_{i1}, q_{i2}...\}$, we define the quality of the entire sequence read as:

$$Q(S_i) = \left(\prod_{j=1}^{|S_i|} q_{ij} \right)^{1/|S_i|}. \tag{2.5}$$

Using the $\mathcal{I}_u$ we identify the the ten most common strings for each allele on, e.g. the D16 locus, and plot it against the $Q(S_i)$ as shown in Figure 2.3. The choice of locus D16 is completely random.

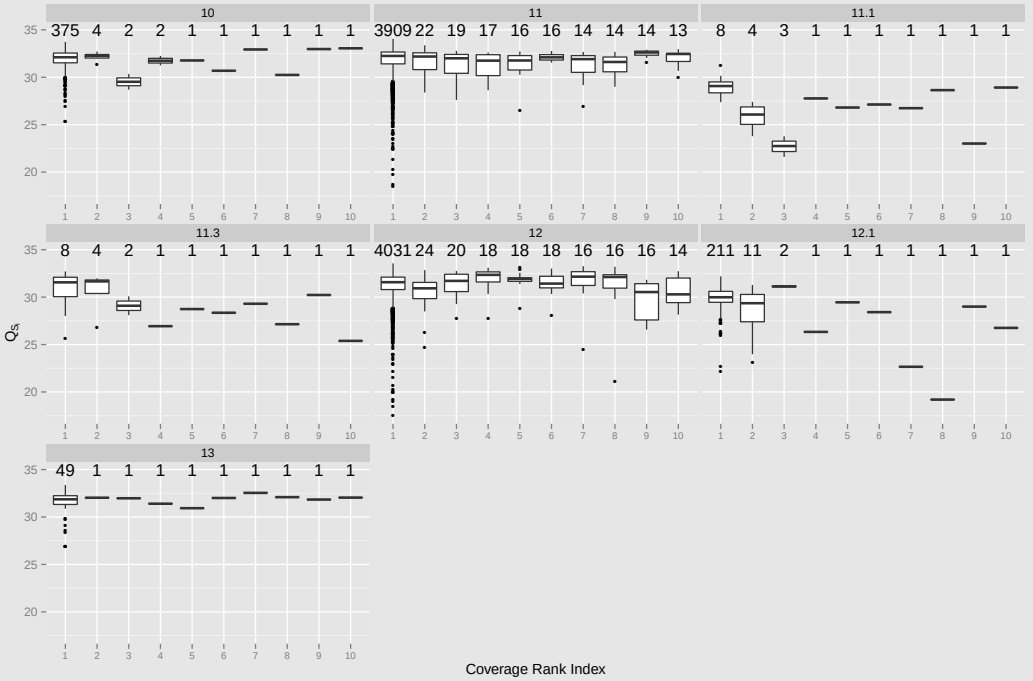


Figure 2.3: Boxplots of Q_{S_i} on locus D16 for the top ten most common strings in each allele. Alleles with very little coverage has been excluded, as they would show nothing of use.

From Figure 2.3, we see that the alleles with low coverage (alleles 11.1, 11.3 and 12.1) decrease in mean quality, as we go from most prevalent to 10th most prevalent string. The alleles with high coverage (alleles 11 and 12), however, seems more steady, contrary to what we had hoped. Furthermore, the deviation on low coverage alleles, is higher and more unstable than the high coverage ones and then there are alleles 10 and 13. Allele 10,13 both seem to be somewhere in between (it is what we will later classify as a stutter and a backstutter of allele 11 and 12. respectively).

A reason why we do not see a downward trend on the high coverage alleles, could be a consequence of us only looking at the top ten most common strings. Therefore, we will look at the top 50 strings for the true alleles, however, as seen in Figure 2.4 the mean and deviance is still very consistent.

In order to more fully understand what is going on, we introduce and examine the second dimension of quality, the quality of bases.

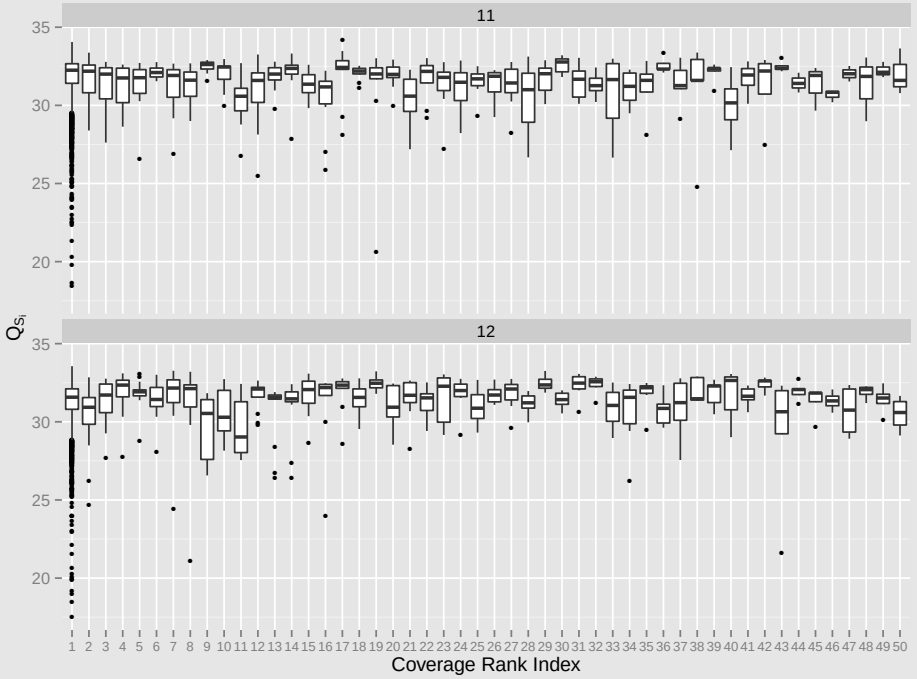


Figure 2.4: Boxplots of Q_{S_i} for alleles 11 and 12 on locus D16 for the top 50 most common strings in each allele. Furthermore the maximum number of base mismatches of the top 50 strings..

2.3 Quality Assessment of Bases

Given n strings, $S_1, ..., S_n$, of similar length m, $S_i = \{b_{i1}, ..., b_{im}\}$, we can define the quality of a given base $B_j = \{b_{1j}, ..., b_{nj}\}$, as $\mathcal{Q}_{*,j} = \{q_{1j}, ..., q_{nj}\}$. That is, we can describe quality in two dimensions. As we did for sequences we can define the quality of a base as:

$$Q(B_j) = \left(\prod_{i=1}^{|B_j|} q_{ij} \right)^{1/|B_j|}. \tag{2.6}$$

The first quality dimension represents the sequences themselves and with the second representing the bases, as shown in Table 2.2. As we can not, by visual inspection, obtain any information from the first dimension, as seen in Section 2.2, we will examine the second dimension.

The representation seen in Table 2.2 gives rise to a quality matrix Q; the (i, j)-th entry of the matrix is quite naturally given by the quality of b_{ij}, i.e. $Q(b_{ij})$ (or q_{ij}). Using the second quality dimension, we will look at the quality of mismatching reads

between strings of the same allele, as illustrated in Figure 2.1, by the red and blue labelled bases. Note that given two identical reads $S_i, S_k \in u$ the quality of a base, even though they are identical per definition of u, is not necessarily equal.

Table 2.2: The two dimensions of quality. The first represented through the strings and the second given by the bases.

	B_1	$\cdots$	B_m	$Q(S_i)$
S_1	b_{11}	$\cdots$	b_{1m}	
S_2	b_{21}	$\cdots$	b_{2m}	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	
S_n	b_{n1}	$\cdots$	b_{nm}	
$Q(B_j)$				

To be more precise, we will chose a reference sequence $u_r \in \mathcal{U}_j$, it will be chosen as the most prevalent sequence of that length. That is, we choose u_r, such that $|\mathcal{I}_{u_r}| \geq |\mathcal{I}_u|$, $\forall u \in \mathcal{U}_j \backslash \{u_r\}$. We then compare the geometric mean quality of bases for all $S_i \in \mathcal{S}_{u_r}$, with the quality of bases in $S_i \in \mathcal{S}_u$, $\forall u \in \mathcal{U}_j \backslash \{u_r\}$. Given rise to the quality ratio (QR) defined as:

$$\mathrm{QR}(B_j; u_k) = \frac{Q(B_j; \mathcal{I}_{u_k})}{Q(B_j; \mathcal{I}_{u_r})}, \tag{2.7}$$

where $Q(B_j; \cdot)$ is the geometric mean quality of B_j, as seen in Equation (2.6), but restricted to the indices provided as the second argument, i.e. $Q(B_j; \mathcal{I}_{u_k}) = \left(\prod_{i \in \mathcal{I}_{u_k}} q_{ij}\right)^{1/|\mathcal{I}_{u_k}|}$. Furthermore, we can restrict j to be in the set containing exactly the bases that mismatch, i.e. $\mathcal{I}_{u_k,u_r} = \{j \mid b_{kj} \neq b_{rj}, b_{kj} \in u_k, b_{rj} \in u_r\}$. For such j, we can define the quality mismatch ratio (QMR) in a similar manner, and denote it by $\mathrm{QMR}(B_j; u_k)$. Note that a QR (or QMR) equal to one, indicates no difference between the quality of the reference and the tested sequence, we would hope that mismatched bases have a lower quality than the reference, i.e. that the QR is strictly less than one.

Using the QR and QMR, we examine the difference in quality of bases, between the most prevalent sequence and all the other sequences variants of similar length (we will restrict ourselves to the ten most prevalent). Figure 2.5, shows the QMR of the

nine most prevalent sequences (excluding the reference sequence) of each allele at locus D16.

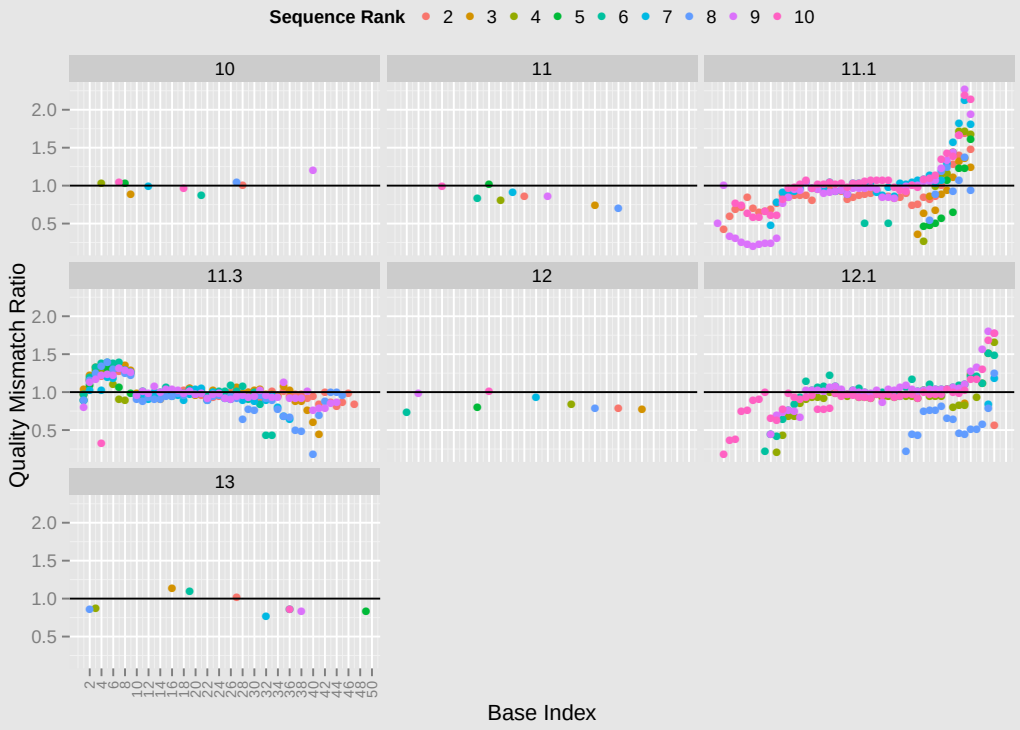


Figure 2.5: The QMR of the ten most prevalent sequences on each allele at D16, plotted against the base, at which it mismatched.

From the figure we see that the higher coverage alleles (10, 11, 12, and 13) in general have few mismatches between the reference and the second-to-tenth most prevalent sequences. Furthermore, we see that the QMR for the high coverage alleles, is pretty evenly distributed around one. Looking at the plots regarding the low coverage alleles (11.1 11.3, and 12.1), we see that there is not much sense in handling them in this manner. The reason being that these alleles occur mostly do to indels. An 11.3 might be a 12 missing a base, however, as it is not the same base being skipped every time, choosing the most prevalent string as reference does not make any sense.

In order to further examine what is happening we will examine the QR of allele 12, shown in Figure 2.6, we see that QR of mismatched bases (indicated by blue points) is generally lower than the QR of matched bases. With that being said it is not always the case, in some cases the dip in quality happens to one of the surrounding bases. This is generally caused by the way the probability of error is estimated, as

discussed in Section 2.1.

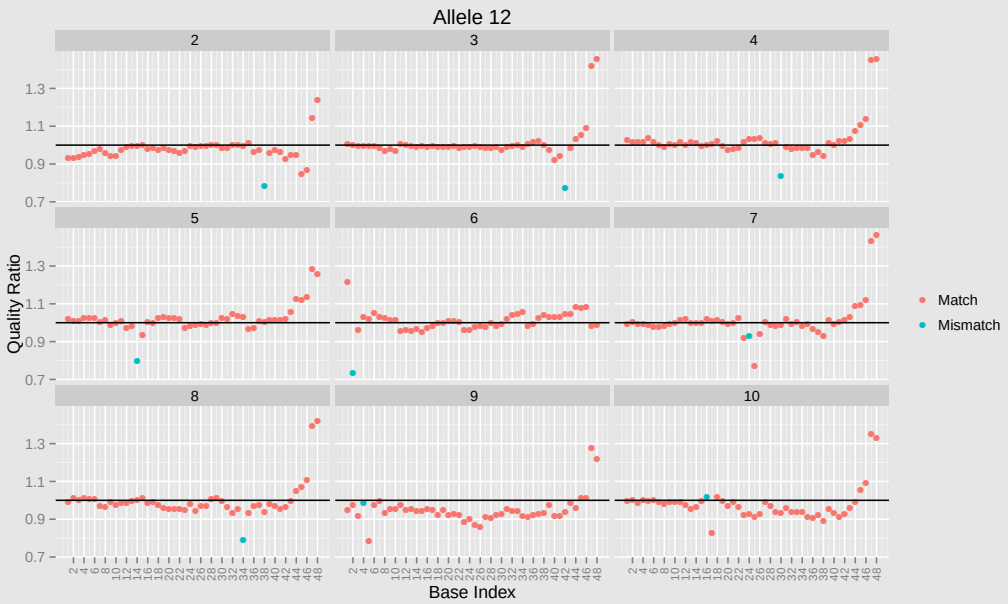


Figure 2.6: The QR of the second to eleventh most prevalent sequences on the 12 allele (the most prevalent string is used as the reference sequence), plotted against the base. Blue points indicates a mismatch, whereas the red points indicates a match, between the base of a given sequence and the reference sequence.

Table 2.3: The difference in geometric mean of the quality between matched and mismatched bases, for the alleles of locus D16.

Allele	10	11	12	13
Match Quality	31.69	31.33	31.33	31.94
Mismatch Quality	31.06	28.11	26.67	28.82
Difference	0.63	3.21	4.66	3.12

If we look at the difference in geometric mean between matched and mismatched bases, shown in Table 2.3, we see that there is a decrease in quality of only 4.66 on allele 12, and with the low number of mismatched bases (e.g. one base in 48, when looking at the second most common string found on allele 12), implies that the quality of the entire string, as defined by Equation (2.5), will remain quite stable.

One solution might be to assign the minimum base quality as the quality of the entire string. However, as seen in Figure 2.7, the quality is so stable that it would not make a difference.

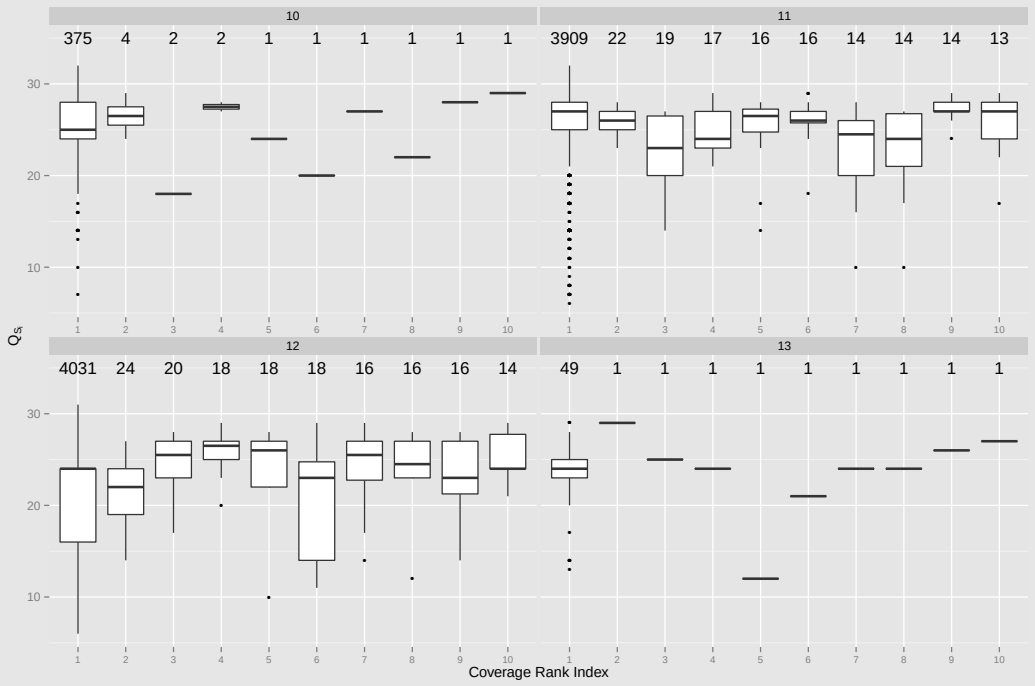


Figure 2.7: Boxplots of $Q_{S_i} = \min_j \{Q(b_{ij}) \in S_i\}$ on locus D16 for the ten most prevalent strings in each allele. Alleles with very little coverage has been excluded, as they would show nothing of use.

Why does the quality remain so stable? The answer to that question is that we are already restricting the quality. To understand why, we will look more generally at the behaviour of the quality scores, specifically the concept of preferential detection.

2.4 Preferential Detection

The principle of preferential detection is quite similar to that of preferential amplification (see Section 1.3), in contrast to preferential amplification, the problem does not stem from the PCR amplification of the template DNA, but from sequencing of the amplified sample. During sequencing the probability of error increases (i.e. the quality decreases) with each called base as seen Figure 2.8. That is, we face a similar consequence to that of preferential amplification; the coverage (and quality) of longer alleles will be smaller (and worse) than those of shorter ones.

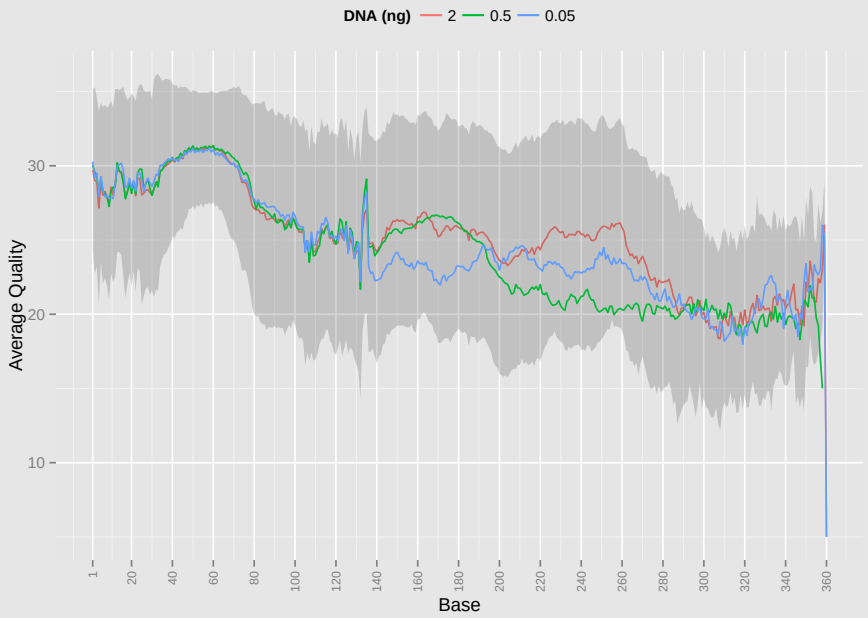


Figure 2.8: The average quality of all bases of the `LT_Dil_002_F_2ng`, `LT_Dil_004_F_05ng`, and `LT_Dil_007_F_005ng` samples plotted against the base. The gray ribbon indicates mean +/− standard deviation of sample `LT_Dil_002_F_2ng`.

The effect of preferential detection, on samples `LT_Dil_002_F_2ng`, `LT_Dil_004_F_05ng`, and `LT_Dil_007_F_005ng`, can be seen in Figure 2.8, as the red, green, and blue lines, respectively. The figure shows that average quality of a given base for each sample, and we clearly see the downward trend we would expect and we see that average quality is fairly stable even across samples. Furthermore, the standard deviation of the `LT_Dil_002_F_2ng` sample, indicated by the gray ribbon, seems quite stable, with the exception of the area around 40-80 bases, and the very last 60 bases. The former is quite interesting, while the later is most likely do to the low number of observations.

If we look at the base quality of the two alleles on the vWA identified reads (we choose this locus, as it in general has long alleles and the difference between the two alleles of our reference profile is large compare to the rest) using the `LT_Dil_002_F_2ng` sample, we see, in Figure 2.9, that the quality drops with each base for both alleles. Furthermore, we see that the longer allele drops faster than the shorter allele.

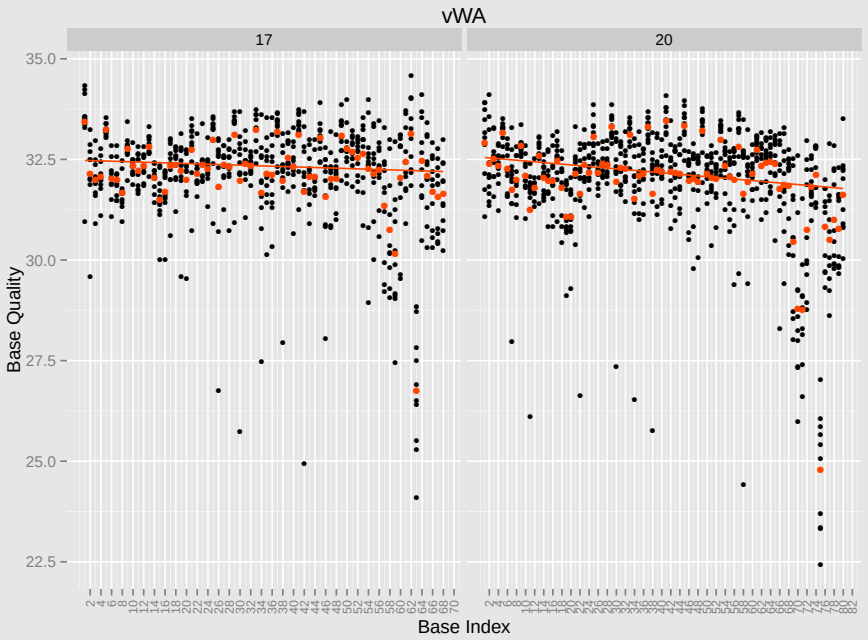


Figure 2.9: The quality plotted against the base. The red points indicate the average quality of a given base and the red line represents a robust linear fit (rlm) of the average quality.

The drop in quality between the two alleles, seen in Figure 2.9, is less than a point, which is not much compared to what is seen in Figure 2.8. In order to more fully understand what is going on, we need to consider exactly how we have identified the loci in Section 1.6. When we identify the loci, we do it based on the flanking regions seen in Table 1.1, because of this, the three statements of List 2.4.1 holds true.

List 2.4.1

(i) All flanks (both forward and reverse) are twelve bases long.

(ii) We only include reads where we have identified both the forward and reverse flank.

(iii) We only allow one mismatched base in both the forward and reverse flank.

As we for the moment are not including reverse complement, List 2.4.1 items (i)-(iii), imply that eleven out of twelve bases, in the reverse flank, has to be called correctly.

That is, the bases in the reverse flank, of included reads, needs a probability of error low enough to have at most a single mismatch. Thus we are implicitly restricting the quality already. This fact is demonstrated in Figure 2.10, where we see exactly just how little we on average use of the reads on a given locus and thus more fully answering the question posed at the end of Section 2.3.

Figure 2.10: The average quality of all bases of the `LT_Dil_002_F_2ng` sample plotted against the base. The red line indicate that the quality was averaged across all reads, whereas the blue line is only averaged using reads identified as belonging to the vWA locus. The first vertical bar (seen from the left) indicate the average end of the forward flank, of all reads on the vWA locus. The last two vertical bars and the corresponding shaded areas indicates average beginning of the reverse flank, $+/-$ standard deviation, of the reverse flank, for allele 17 (orange shaded colour) and 20 (blue shaded colour), respectively, on the vWA locus.

Figure 2.10 also shows that, within the window created by the horizontal bars, the average quality, of the vWA identified reads, is on average high compared to the average quality of all reads. Another consequence of preferential detection, that is also illustrated in figure, is that we might not catch as many of the longer of the two alleles on a heterozygous locus, creating an imbalance.

For us to examine this imbalance, we will use the alleles 17 and 20 from locus vWA and the `LT_Dil_002_F_2ng` sample as before and start by identifying the alleles using the true sequences and the flank regions. We will then trim elements from the reverse flank one at a time (from last till first), finding the coverage of the two alleles each time, we do this 13 times (one for each base in the reverse flank plus one where no trimming occurred), which is also illustrated in Figure 2.11. Using the coverage we calculate the heterozygote balance in all 13 cases, the coverage of allele 20 over the coverage of allele 17. The coverage and H_b can be seen in Table 2.4.

Table 2.4: The heterozygote balance given the number of bases cut from the reverse flank, as well as the coverage for both allele 17 and 20 on the vWA locus.

Number of Trimmed Bases	Coverage Allele 17	Coverage Allele 20	Heterozygote Balance
0	7062	5723	0.8104
1	7092	5749	0.8106
2	7927	7906	0.9974
3	7935	7911	0.9970
4	7965	7943	0.9972
5	8053	8088	1.0043
6	8076	8114	1.0047
7	8090	8123	1.0041
8	8126	8150	1.0030
9	8156	8182	1.0032
10	8186	8210	1.0029
11	8198	8225	1.0033
12	8286	8366	1.0097

As seen in Table 2.4, the heterozygote balance tends towards one as we trim more and more of the reverse flank (the biggest change in H_b happens when trimming exactly two bases). Furthermore, we see that the coverage of allele 17 with untrimmed reverse flank is lower than that allele 20 with 12 trimmed bases, even though they are of the same base length, which is quite curious.

In conclusion we are already restricting the quality, just by locating the STR region

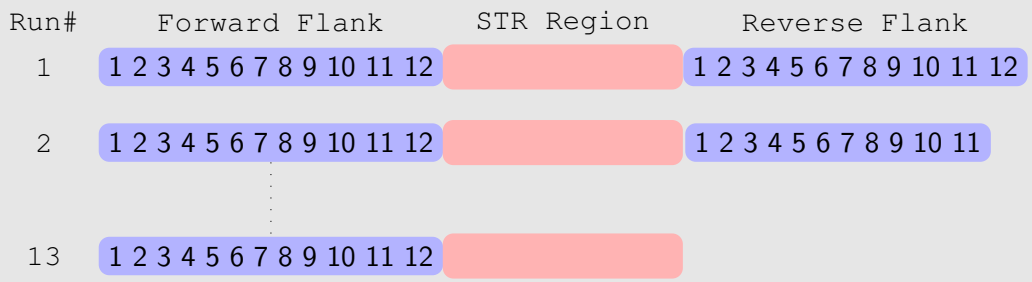


Figure 2.11: An illustration of the trimming experiment performed on the reverse flank. The length of the STR region in our case is 17 and 20, respectively, with a motif length of 4. We start by not trimming any bases from the reverse flank, finding the coverage of the entire sequence including forward flank, the true sequence (for both allele 17 and 20), and the reverse flank. We then, at the start of the second run, trim the reverse flank removing the last base and find the coverage of the entire sequence as before, but using this new trimmed reverse flank. We continue in this fashion until we have removed all bases in the reverse flank, 13 runs in total. Furthermore, at no point during this process do we allow any mismatches, in either the forward flank, allele sequence, or the (trimmed) reverse flank.

of every read. It follows that the remaining reads are of a fairly high quality and therefore even when a base is called incorrectly, the drop in quality will not be much when averaged across an entire read. Therefore, we need a more sophisticated method comparing two unique strings of similar length.

2.5 Comparing Unique Strings of Similar Length

The following ideas are based on the observations made in Section 2.3, more specifically the dip in quality observed surrounding the miscalled base in Figure 2.5. First assume that we have m strings of similar length and we would like to determine the probability that two given strings are equal. A more accurate description of the assumptions can be seen in List 2.5.1. Note that we use the probability of error instead the quality, as we want the probability that two strings are equal it seems more appropriate to use the probability of error (though in reality using the quality would be equivalent).

List 2.5.1

(i) We observe m-strings $s_1, s_2, ..., s_m$, of similar length, such that $s_i \in \mathcal{A}_L(n)$, where $\mathcal{A}_L(n)$ is all possible alleles on locus L of length n.

(ii) We know the indices for which these strings mismatch, the set of these indices is defined like so: $\mathcal{I}_{i,j} = \{k \mid b_{ik} \neq b_{jk}, \forall b_{ik} \in s_i, b_{jk} \in s_j\}$.

(iii) We know the probability of errors $P_{i,j}$ corresponding to the strings s_i and s_j.

We define the neighbourhood surrounding the jth base of the ith string, b_{ij}, as:

$$\partial(j,t) = \{h : h \neq j, h \in [\max\{1,(j-t)\}; \min\{(j+t), n\}]\}\,. \tag{2.8}$$

That is, a base b_{ih} is considered a neighbour to b_{ij} if and only if $h \in \partial(j,\cdot)$. We will generally choose $t = 5$, because of the way the probability of error is estimated by the basecalling algorithm, see Section 2.1. We will write $\partial(j)$ instead of $\partial(j,t)$, when the value of t is either clear from the context, or not important. Furthermore, we define an extended neighbourhood as $\bar{\partial}(j,t) = \partial(j,t) \cup \{j\}$.

Given List 2.5.1 items (i)-(iii), we propose that the probability of two strings being equivalent can be calculated, as the product of the probability of two bases being equivalent. We do so because the probability of two strings being equivalent can be seen as the joint probability that the bases are equivalent. That is:

$$\begin{aligned}
\mathbb{P}(S_k \equiv s_i \mid s_k \neq s_i, \mathcal{I}_{k,i}, P_{k,i}) &= \prod_{j \in \mathcal{I}_{k,i}} \mathbb{P}(B_{kj} \equiv b_{ij} \mid P_{k,i}) \\
&= \prod_{j \in \mathcal{I}_{k,i}} \mathbb{P}(B_{kj} \text{ is called in error} \mid P_{kh}, h \in \bar{\partial}(j)) \\
&= w_i \prod_{j \in \mathcal{I}_{k,i}} P_k(j),
\end{aligned} \tag{2.9}$$

where the probabilities $P_k(j)$ will be calculated as:

$$P_k(j) = \frac{\underset{P_{kh}}{\arg\max}\, \underset{h \in \bar{\partial}(j)}{} \left\{ \dfrac{P_{kh}}{|h-j|+1} \right\}}{\displaystyle\sum_{h \in \bar{\partial}(j)} \left(\frac{P_{kh}}{|h-j|+1} \right)}. \tag{2.10}$$

A graph of the penalised probability $P_{kh}/(|h-j|+1)$ as a function of h is shown in Figure 2.12. Furthermore, the weights w_i, seen in Equation (2.9), will be given as follows:

$$w_i = \frac{\varphi_i}{\varphi_k + \varphi_i}, \tag{2.11}$$

where φ_j is the coverage of string j. That is, the weight w_i is an indication of our belief in s_i compared to S_k.

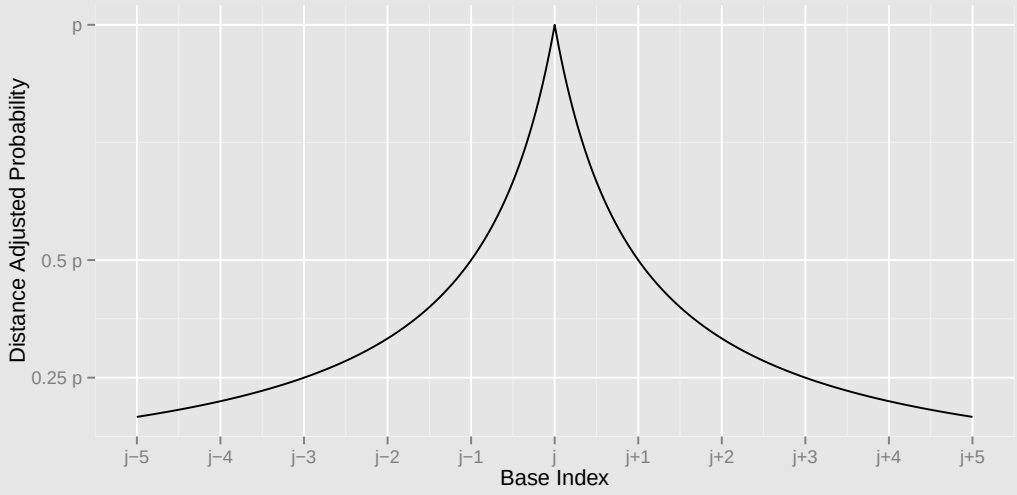


Figure 2.12: The function used to penalise the probability as a function of the distance between the current base j and the elements of the neighbourhood.

In order to examine this approach, we will use the two most prevalent strings of repeat length 20 on locus vWA of the `LT_Dil_002_F_2ng` sample. We will call these two strings s_1 and s_2. The coverage of the two string is 102 and 11626, respectively. The strings are given as follows:

s_1 : ...TCTG TCTG TCTA TCTA TCTA..., Coverage: 102
s_2 : ...TCTG TCTG TCTG TCTA TCTA..., Coverage: 11626

Note that we have truncated the strings, and coloured the mismatching base, which is base $j = 20$.

The probabilities in the neighbourhood of the mismatching base can be seen in Table 2.5. The table shows the probability and adjusted probability for an extended neighbourhood $\bar{\partial}(j)$ setting $t = 5$ in Equation (2.8). We see from the table that using the adjusted probability we swift the max probability from $j+1$ to j for s_2.

Table 2.5: The adjusted and non-adjusted probability of error of the neighbourhood $\bar{\partial}(j)$ using $t = 5$ for the two strings s_1 and s_2.

s_1	$j-5$	$\cdots$	$j-1$	j	$j+1$	$\cdots$	$j+5$
Prob.	6.3e-04	$\cdots$	1.3e-03	1.6e-03	1.3e-03	$\cdots$	6.3e-04
Adj. Prob.	1.1e-04	$\cdots$	6.3e-04	1.6e-03	6.3e-04	$\cdots$	1.1e-04
s_2							
Prob.	6.3e-04	$\cdots$	7.9e-04	7.9e-04	1.0e-03	$\cdots$	7.9e-04
Adj. Prob.	1.1e-04	$\cdots$	4.0e-04	7.9e-04	5.0e-04	$\cdots$	1.3e-04

It follows from the probabilities given above and Equations (2.9), (2.10), and (2.11), we obtain the following results:

$$\begin{aligned} \mathbb{P}(s_2 \equiv s_1 | s_1 \neq s_2, \mathcal{I}_{1,2}, P_{1,2}) &\approx 0.002 \\ \mathbb{P}(s_1 \equiv s_2 | s_1 \neq s_2, \mathcal{I}_{1,2}, P_{1,2}) &\approx 0.362 \end{aligned} \tag{2.12}$$

We see that the probability that s_1 is actually a variation of s_2 is just above 36%, and is more than a hundred times more likely than s_2 being a variation of s_1. These results are highly depended on the choice of neighbourhood, using e.g. $t = 2$ would in Equation (2.12) yield probabilities of 0.003 and 0.456, respectively. Note that from this point fourth we will generally not use s to represent a string, in favour of a, as to avoid confusion when referring to samples.

How can we use this probabilities? We could augment the coverage using the probabilities as weights. An idea first mentioned in the beginning of Chapter 2. Or if the probability that one string is a variation of another is above some threshold T_{Prob}, e.g. 0.05, we could simply remove it from further consideration. In a mixture scenario the probability could help identify whether a string is due to an error in the NGS process or another contributor besides the primary donor. We will let the subject lie for now, in order to take a closer look at some of the systematic noise generated by the NGS workflow, namely stutters and shoulders.

CHAPTER

THREE

STUTTER ANALYSIS

We know from Section 1.4 that amplifying the template DNA using PCR creates stutters in the amplified samples. A consequence of this phenomenon is that, if the difference between the two alleles is one, then the observed coverage at the smaller of the two alleles contains stutter from the larger allele, after amplification.

In order to be more accurate, if we let $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$ be the alleles before amplification, with stutter frequencies ξ_i and ξ_{i+1}, respectively. Let the coverage of $\mathcal{A}_i$, before PCR amplification, be denoted as $\varphi_i^{(0)}$, then the observed coverage, of $\mathcal{A}_i$, in the $(c+1)$th cycle is $\varphi_i^{(c+1)} = (1-\xi_i)\varphi_i^{(c)} + \xi_{i+1}\varphi_{i+1}^{(c)}$. The concept is illustrated in Figure 3.1.

Another thing worth noting is that we, because of NGS, have an additional condition, namely that the strings of the two alleles needs to be equivalent, in order for them to stack as seen in Figure 3.1. Equivalence in this context is more specifically defined as follows: let a_i and a_{i+1} be the true strings of allele $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$, respectively, then a_i and a_{i+1} are equivalent, if the stutter of a_{i+1} is equal to a_i. This additional condition implies that the situation illustrated in Figure 3.1 will not be as common in NGS as it was in CE.

In the context of assessing $\mathbb{P}(\mathcal{E}|\boldsymbol{g})$ and the possibility of multiple contributors, assume that $\mathcal{A}_i$ and $\mathcal{A}_j$ are the true alleles. If $j-1 > i$ and we observe $j-1$, we then need to identify whether the coverage φ_{j-1} is primarily caused by stutter from $\mathcal{A}_j$ or

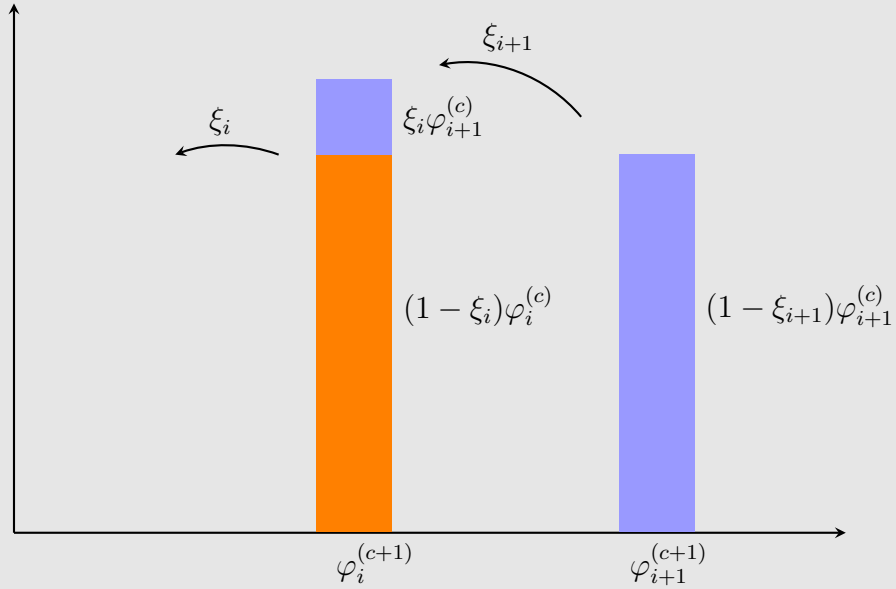


Figure 3.1: A simple illustration of the observed coverage of $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$ after $(c+1)$ cycles of PCR amplification. The orange and blue colours indicates contribution from $\varphi_i^{(c)}$ and $\varphi_i^{(c)}$, respectively. The arrows indicate stuttering with frequency ξ_i and ξ_{i+1} for $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$, respectively.

another contributor to the sample. Therefore, we would like to estimate the stutter frequency ξ, also known as the stutter ratio. The stutter ratio, SR, is defined as follows:

$$SR = \frac{\varphi_{\text{Stutter}}}{\varphi_{\text{Parent}}}. \tag{3.1}$$

We know from Section 1.4 (more specifically [5, 6]), that stuttering increases with allele length, i.e. the stutter ratio increases, this at least holds for simple repeat patterns. However, when working with a compound (or complex) repeat or a microvariant, it has been hypothesised that stutter ratio is more correlated with the the longest uninterrupted stretch (LUS) [5].

The LUS is defined as the longest stretch of simple repeats within the allele. When using CE, one would have to make an educated guess at the value of the LUS, as only the fragment lengths are observed. However, as we, by using NGS, have base resolution, we can actually calculate the LUS explicitly. Figure 3.2 and 3.3 shows the SR against allele length and LUS, respectively.

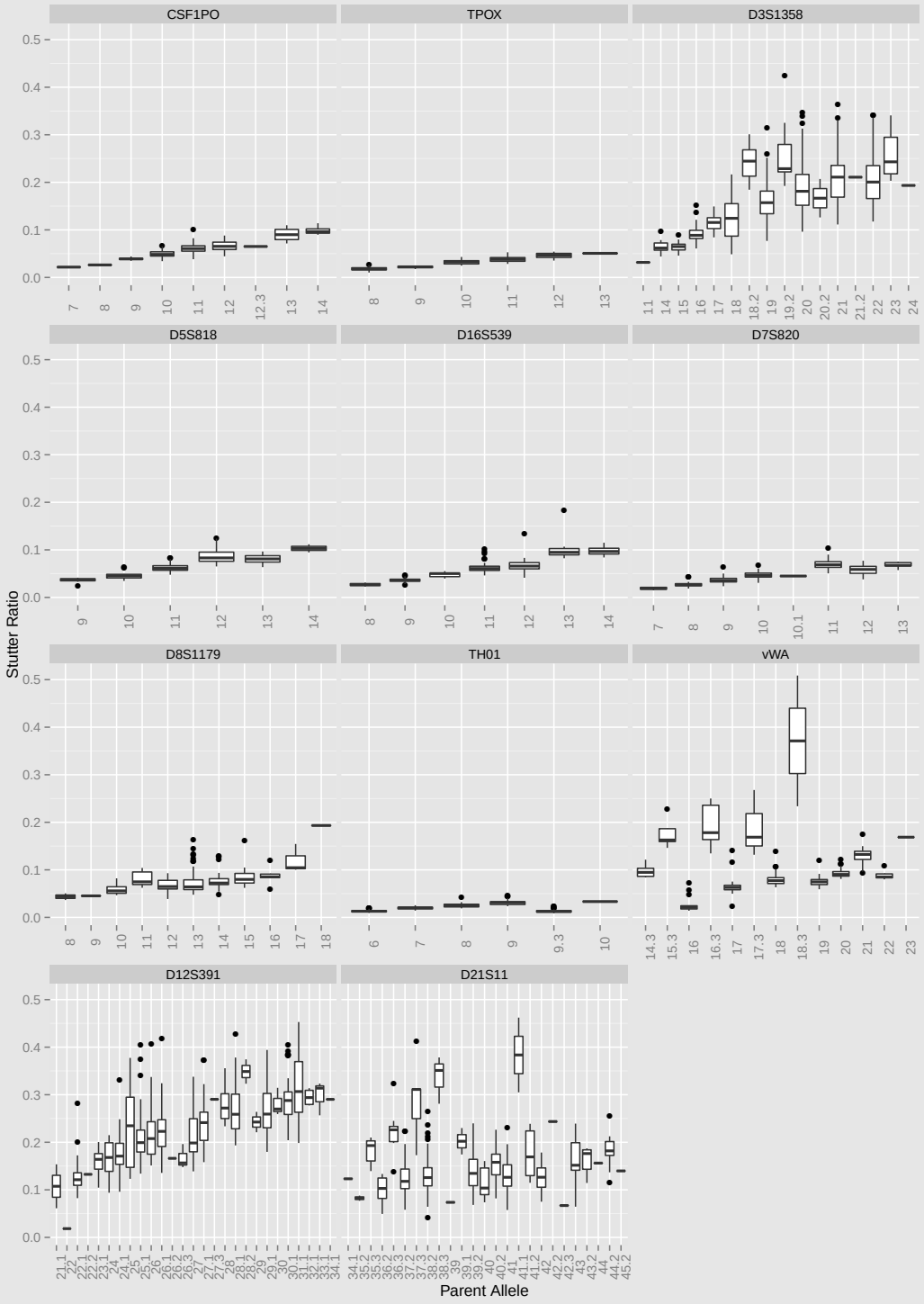


Figure 3.2: Boxplots of stutter ratio against the number of total repeats in the parent allele for all loci in the IonTorrent and Roche reference files.

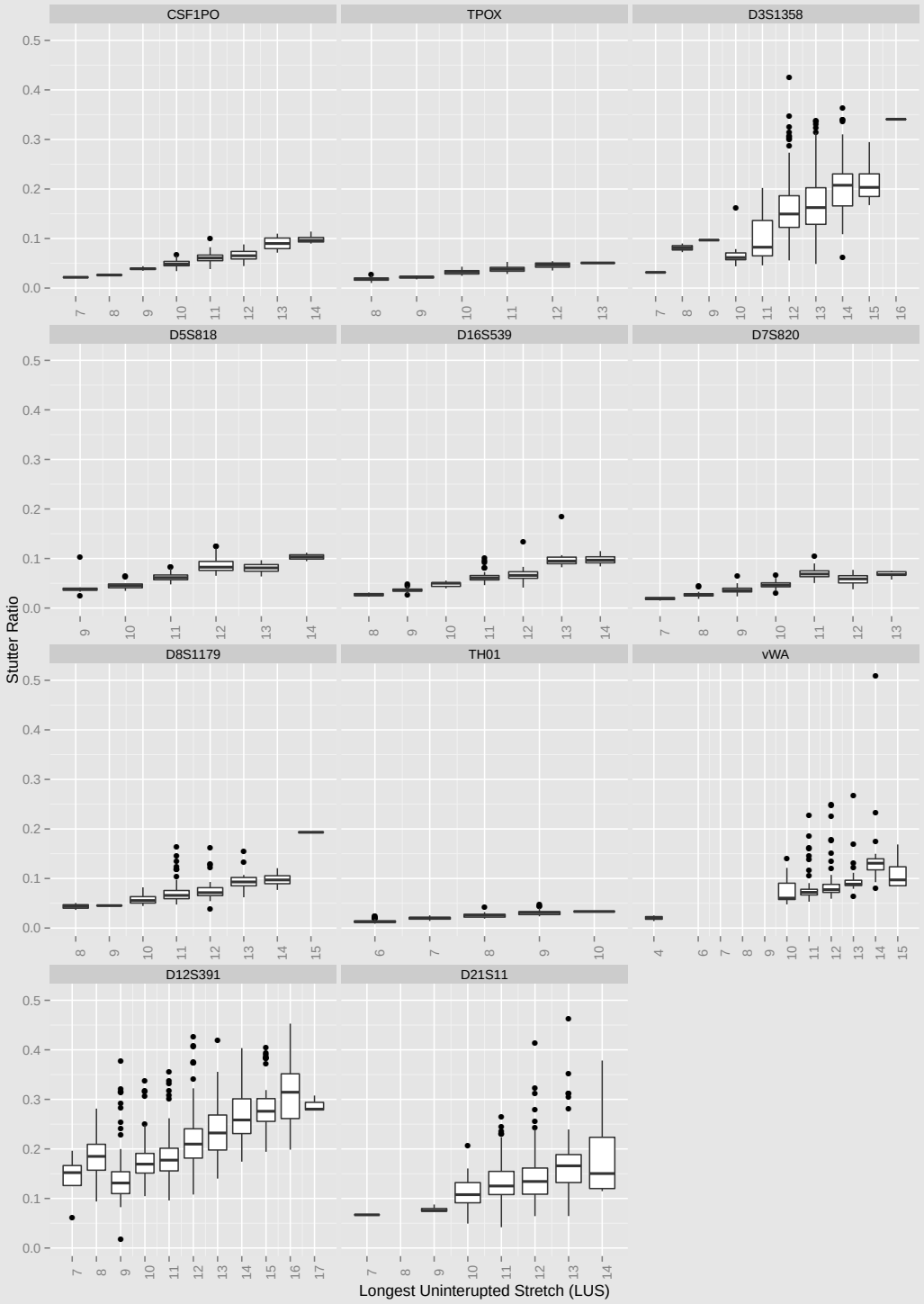


Figure 3.3: Boxplots of stutter ratio against the LUS for all loci in the IonTorrent and Roche reference files.

We will use the IonTorrent and the Roche reference files described in Section 1.5.4. We specifically want to include the Roche files in this analysis, as the loci used

generally contain longer alleles. In order to illustrate the difference between allele length and LUS, we have, in Figure 3.4, shown a boxplot of the SR against both potential explanatory variables, for the TH01 locus (recall, as mentioned in Section 1.2, that $\approx 34\%$ of the Danish population have the 9.3 microvariant on this particular locus).

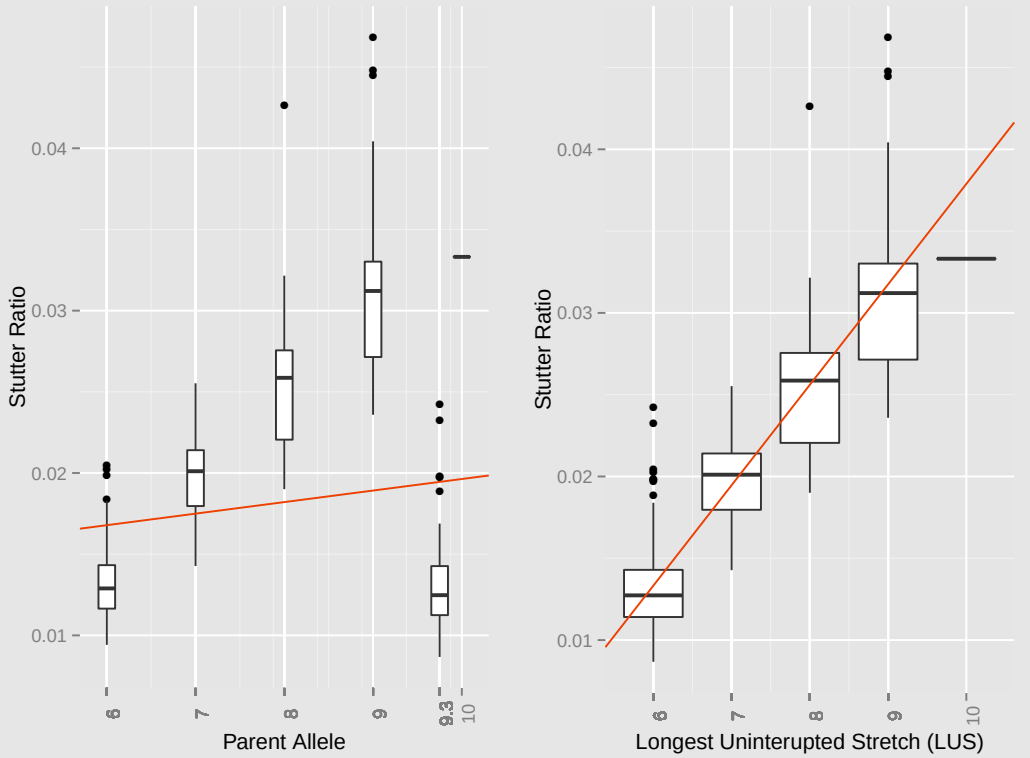


Figure 3.4: Boxplots of stutter ratio against the number of total repeats in the parent allele and LUS, respectively, for the TH01 locus.

Figure 3.4 shows that the relationship between the SR and LUS is fairly linear, with an $R^2 = 0.7075$. That would hold for allele length as well, if we disregarded allele 9.3, however, as it stands it achieves an $R^2 = 0.0403$. The median of the SR, on allele 9.3, is more in line with that of allele 6. The LUS of allele 9.3 is in fact 6, as $[\text{AATG}]_6\text{ATG}[\text{AATG}]_3$. This is why LUS has been proposed as better predictor of SR, than allele length.

In order to better describe this relationship, we fit two simple linear models, and as this relationship is locus dependent, see e.g. [6, 23], the models we are considering take the following form:

$$\log\left(SR_{ij}\right) = \beta_{i,1} + \beta_{i,2}X_{ij} + \varepsilon_{ij}, \tag{3.2}$$

where $\varepsilon_{ij} \sim \mathrm{N}\left(0, \sigma_{ij}^2\right)$, $i = 1, ..., k$, $j = 1, ..., n_i$, with k being the total number of loci, and n_i the number of observations on locus i, making X_{ij} the j'th covariate (allele length or LUS) of the i'th locus, and SR_{ij} the stutter ratio of the corresponding allele. We fit the logarithm of SR_{ij}, because a sum is generally easier to fit than a quotient. Note that we do not weight the variance using $1/\varphi_{\mathrm{Parent},j}$, as suggested by [24, Section 2.2]. The intercept and slope estimates of the two models, can be seen in Table 3.1.

Table 3.1: The intercept and slope estimates for each locus of the allele and LUS covariate models.

	Allele Length				LUS			
Locus	Intercept	SE	Slope	SE	Intercept	SE	Slope	SE
CSF1PO	-4.7817	0.1246	0.1757	0.0112	-4.7946	0.1246	0.1770	0.0113
TPOX*	-5.9936	0.0795	0.2470	0.0086	-5.9936	0.0795	0.2470	0.0086
D3S1358	-5.1203	0.0983	0.1685	0.0052	-4.5963	0.1619	0.2136	0.0129
D5S818	-5.1881	0.1687	0.2190	0.0146	-5.0053	0.1774	0.2035	0.0154
D16S539*	-5.2934	0.1042	0.2228	0.0093	-5.2934	0.1042	0.2228	0.0093
D7S820*	-5.2546	0.1001	0.2136	0.0099	-5.2546	0.1001	0.2137	0.0099
D8S1179	-3.5951	0.1322	0.0747	0.0103	-4.4014	0.1779	0.1546	0.0156
TH01	-4.2860	0.1424	0.0248	0.0174	-6.1276	0.0804	0.3002	0.0117
vWA	-4.8717	0.3739	0.1267	0.0202	-4.5482	0.1001	0.1766	0.0086
D21S11	-2.9706	0.3130	0.0242	0.0079	-3.3428	0.1733	0.1139	0.0149
D12S391	-3.9636	0.0932	0.0915	0.0035	-2.8099	0.0602	0.1034	0.0049

*Loci with equal parameter estimates implying that the loci contains only simple repeats.

We see, by looking at the estimates in Table 3.1, that the estimates for some of the loci (marked with an asterisk) are the same, implying that the allele length and LUS on these loci are identical, i.e. the sequences on those alleles consists of entirely simple repeats, where the LUS and parental allele designation coincide.

In order to examine the fitted model, we have plotted the residuals and made a QQ-plot of the residuals, in Figure 3.5 and 3.6, respectively.

Looking at Figure 3.6, we see that the distribution of the residuals have very heavy

tails, which is also seen in [23, Figure 3]. In order to account for the heavy tails we will try to fit a mixture model with equal mean, and a gamma-model using log as a link function, in accordance with [24].

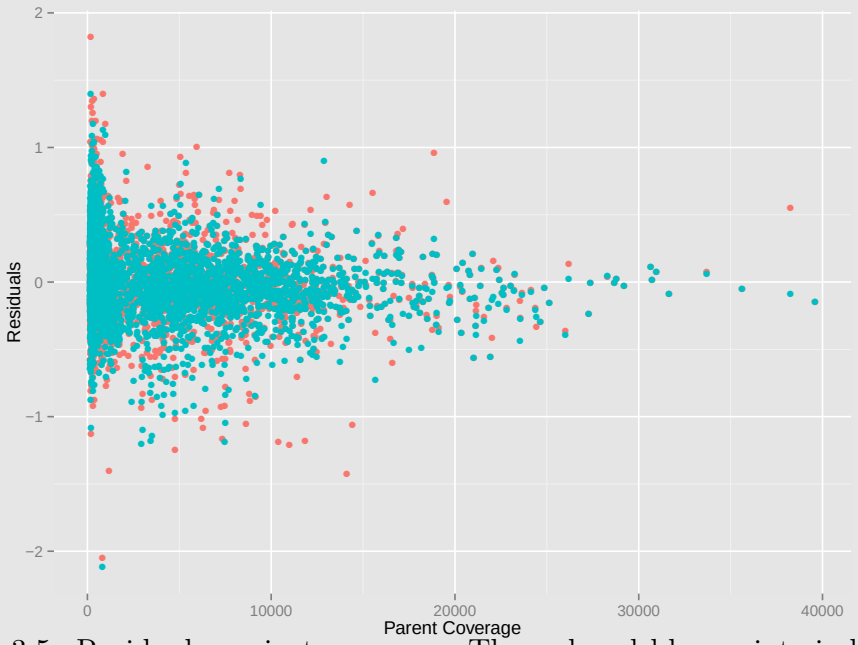


Figure 3.5: Residuals against coverage. The red and blue points indicates a model using allele length and LUS as explanatory variable respectively.

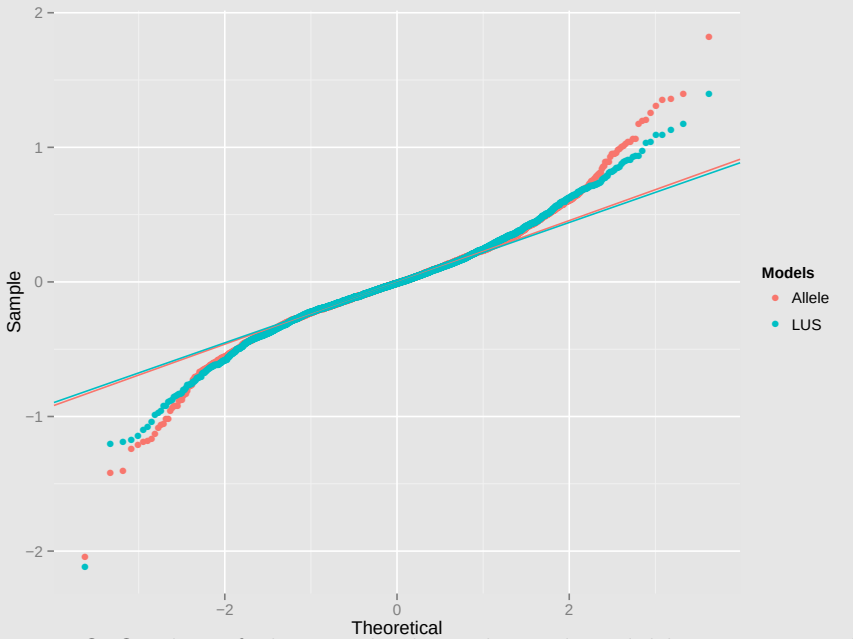


Figure 3.6: Q-Q plot of the residuals. The red and blue points indicates a model using allele length and LUS as explanatory variable respectively.

3.1 The Mixture Model

The mixture model will consist of two variance components, $\varepsilon_{ij(1)} \sim \mathrm{N}\left(0, \sigma^2_{i(1)}\right)$ and $\varepsilon_{i(2)} \sim \mathrm{N}\left(0, \sigma^2_{ij(2)}\right)$, with a mean given by:

$$\mu_{ij} = \beta_{1,i} + \beta_{2,i} x_{ij}. \tag{3.3}$$

Given these two components, the mixture model is given by:

$$SR_{ij} = \mu_{ij} + (1 - z)\varepsilon_{ij(1)} + z\varepsilon_{ij(2)}, \tag{3.4}$$

where $z \in \{0, 1\}$ and $\mathbb{P}(z = 1) = \pi$. Let $\phi_{\boldsymbol{\theta}_k}(\boldsymbol{y})$ denote the normal distribution with parameters $\boldsymbol{\theta}_k = (\mu_{ij}, \sigma^2_{i(k)})$, then the density of SR_{ij} is given by:

$$g_{\boldsymbol{\Psi}}(SR_{ij}) = (1 - \pi)\phi_{\boldsymbol{\theta}_1}(SR_{ij}) + \pi\phi_{\boldsymbol{\theta}_2}(SR_{ij}), \tag{3.5}$$

where $\boldsymbol{\Psi} = (\mu_{ij}, \sigma^2_{i(1)}, \sigma^2_{i(2)})$. However, fitting the variance parameters using maximum likelihood under this model would be rather difficult, as the log-likelihood function of $\boldsymbol{\Psi}$, given n_i observations on the ith locus, would look as follows:

$$\ell(\boldsymbol{\Psi}; \boldsymbol{SR}_i) = \sum_{j=1}^{n_i} \log\left((1 - \pi)\phi_{\boldsymbol{\theta}_1}(SR_{ij}) + \pi\phi_{\boldsymbol{\theta}_2}(SR_{ij})\right). \tag{3.6}$$

If we somehow knew the value of z_j, everything would be easier as $z_j = 0$ would imply $SR_{ij} \sim \mathrm{N}\left(\mu_{ij}, \sigma^2_{i(1)}\right)$ and $z_j = 1$ would imply $SR_{ij} \sim \mathrm{N}\left(\mu_{ij}, \sigma^2_{i(2}\right)$, yielding the following likelihood function:

$$L(\boldsymbol{\Psi}; \boldsymbol{SR}_i, \boldsymbol{z}) = \prod_{j=1}^{n_i} \left((1 - \pi)\phi_{\boldsymbol{\theta}_1}(SR_{ij})\right)^{1-z_j} \left(\pi\phi_{\boldsymbol{\theta}_2}(SR_{ij})\right)^{z_j},$$

given this likelihood function the log-likelihood is greatly simplified:

$$\begin{aligned} \ell(\boldsymbol{\Psi}; \boldsymbol{SR}_i, \boldsymbol{z}) = \; & \sum_{j=1}^{n_i} (1 - z_j)\log\left(\phi_{\boldsymbol{\theta}_1}(SR_{ij})\right) + z_j \log\left(\phi_{\boldsymbol{\theta}_2}(SR_{ij})\right) \\ & + \sum_{j=1}^{n_i} (1 - z_j)\log\left(1 - \pi\right) + z_j \log\left(\pi\right). \end{aligned} \tag{3.7}$$

However, the values of z_j are unknown; we will proceed in an iterative fashion, by substituting z_j with its conditional mean given $\boldsymbol{\Psi}$ and SR_{ij}:

$$z_j^* = \mathbb{E}\left[Z_j | SR_{ij}, \boldsymbol{\Psi}^*\right] = \mathbb{P}(Z_j = 1 | SR_{ij}, \boldsymbol{\Psi}^*),$$

where $\boldsymbol{\Psi}^*$ is the current parameter estimates (z_j^* is also called the responsibility of component two for observation j). We will then maximise the parameters of $\boldsymbol{\Psi}$ using z_j^*. This procedure is known as a special case of the expectation-maximisation (EM) algorithm.

3.1.1 A Special Case of the EM Algorithm

We will only need a special case of the algorithm; as we have a two component Gaussian mixture model, the algorithm simplifies. The algorithm, seen in Algorithm 3.1.1, simplifies even further in our case, as the mean value structure is assumed to be identical in both components. We return to the general EM-algorithm in Section 5.3.

Algorithm 3.1.1 (Special case of the EM Algorithm.)

(1) Make initial guesses of the parameters in $\boldsymbol{\Psi}^* = (\widehat{\mu}_{ij}, \widehat{\sigma}^2_{ij(1)}, \widehat{\sigma}^2_{ij(2)}, \widehat{\pi})$.

(2) **E-step**: Calculate the responsibilities z_j^*.

(3) **M-step**: Calculate the weighted means, variances, and mixing probability using the updated responsibilities.

(4) Repeat steps (2) and (3) until $|\ell(\boldsymbol{\Psi}^*; \boldsymbol{SR}_i, \boldsymbol{z}^*) - \ell(\boldsymbol{\Psi}; \boldsymbol{SR}_i, \boldsymbol{z})| < \varepsilon$, where $\ell(\boldsymbol{\Psi}^*; \boldsymbol{SR}_i, \boldsymbol{z}^*)$ and $\ell(\boldsymbol{\Psi}; \boldsymbol{SR}_i, \boldsymbol{z})$ are the log-likelihood functions, using the current and previous parameter estimates, respectively.

All there is left is to calculate the updated parameters in the E- and M-steps, respectively.

The E-Step:

The responsibilities are easily calculated as:

$$\begin{aligned} z_j^* &= \mathbb{P}(Z_j = 1 | SR_{ij}, \boldsymbol{\Psi}^*) = \frac{\mathbb{P}(SR_{ij} | Z_j = 1, \boldsymbol{\Psi}^*)\mathbb{P}(Z_j = 1 | \boldsymbol{\Psi}^*)}{\mathbb{P}(SR_{ij} | \boldsymbol{\Psi}^*)} \\ &= \frac{\pi \phi_{\boldsymbol{\theta}_2}(SR_{ij})}{(1-\pi)\phi_{\boldsymbol{\theta}_1}(SR_{ij}) + \pi \phi_{\boldsymbol{\theta}_2}(SR_{ij})}. \end{aligned}$$

The M-Step:

Inserting the density functions $\phi_{\boldsymbol{\theta}_1}$ and $\phi_{\boldsymbol{\theta}_2}$, in log-likelihood function seen in Equa-

tion (3.7), yields:

$$\begin{aligned}\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z}) \propto \sum_{j=1}^{n_i}\Bigg\{ & -\frac{(1-z_j)}{2}\left[\frac{(SR_{ij}-\mu_{ij})^2}{\sigma^2_{i(1)}}+\log\left(\sigma^2_{i(1)}\right)\right] \\ & -\frac{z_j}{2}\left[\frac{(SR_{ij}-\mu_{ij})^2}{\sigma^2_{i(2)}}+\log\left(\sigma^2_{i(2)}\right)\right]\Bigg\} \\ & +\sum_{j=1}^{n_i}(1-z_j)\log\left(1-\pi\right)+z_j\log\left(\pi\right)\end{aligned} \tag{3.8}$$

We then maximise the log-likelihood function, with respect to the parameters of $\boldsymbol{\Psi}$, by differentiating Equation (3.8), with respect to each of the parameters:

The mean can be written on vector form as $\mu_{ij} = \boldsymbol{x}_{ij}^T\boldsymbol{\beta}_i$, where $\boldsymbol{x}_{ij}^T = [1 \quad x_{ij}]$ and $\boldsymbol{\beta}_i = [\beta_{1,i} \quad \beta_{2,i}]^T$. Furthermore, from the chain-rule we know that:

$$\frac{\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})}{\partial\boldsymbol{\beta}_i} = \left(\frac{\partial\boldsymbol{\mu}_i}{\partial\boldsymbol{\beta}_i}\right)^T\frac{\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})}{\partial\boldsymbol{\mu}_i}. \tag{3.9}$$

We therefore start by computing $\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})/\partial\boldsymbol{\mu}_i$:

$$\begin{aligned}\frac{\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})}{\partial\mu_{ij}} &= \sum_j\frac{(1-z_j)}{\sigma^2_{i(1)}}(SR_{ij}-\mu_{ij})+\frac{z_j}{\sigma^2_{i(2)}}(SR_{ij}-\mu_{ij}) \\ &= \frac{1}{\sigma^2_{i(1)}\sigma^2_{i(2)}}\sum_j\left((1-z_j)\sigma^2_{i(2)}+z_j\sigma^2_{i(1)}\right)(SR_{ij}-\mu_{ij}).\end{aligned}$$

Inserting the result above and $\partial\mu_i/\partial\boldsymbol{\beta}_i = \boldsymbol{x}_{ij}^T$, in Equation (3.9) we see that (written in matrix form):

$$0 = X^TZ\left(\boldsymbol{SR}_i-\boldsymbol{\beta}_i\right),$$

where Z is a diagonal matrix with $\left((1-z_j)\sigma^2_{i(2)}+z_j\sigma^2_{i(1)}\right)$ as entries. It follows that the estimate of $\boldsymbol{\beta}_i$ become:

$$\widehat{\boldsymbol{\beta}}_i = (X^TZ^*X)^{-1}X^TZ^*\boldsymbol{SR}_i. \tag{3.10}$$

The variance estimates $\widehat{\sigma}^2_{i(1)}$ and $\widehat{\sigma}^2_{i(2)}$ found as follows:

$$\begin{aligned}\frac{\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})}{\partial\sigma^2_{i(1)}} &= \sum_{j=1}^{n_i}(1-z_j)\left(-\frac{1}{2\sigma^2_{i(1)}}+\frac{(SR_{ij}-\mu_{ij})^2}{2\sigma^4_{i(1)}}\right) \\ &= -\frac{1}{2\sigma^2_{i(1)}}\sum_{j=1}^{n_i}(1-z_j)+\frac{1}{2\sigma^4_{i(1)}}\sum_{j=1}^{n_i}(1-z_j)(SR_{ij}-\mu_{ij})^2 \qquad (3.11) \\ \frac{\partial\ell(\boldsymbol{\Psi};\boldsymbol{SR}_i,\boldsymbol{z})}{\partial\sigma^2_{i(2)}} &= \sum_{j=1}^{n_i}z_j\left(\frac{1}{2\sigma^2_{i(2)}}+\frac{(SR_{ij}-\mu_{ij})^2}{2\sigma^4_{i(2)}}\right) \\ &= -\frac{1}{2\sigma^2_{i(2)}}\sum_{j=1}^{n_i}z_j+\frac{1}{2\sigma^4_{i(2)}}\sum_{j=1}^{n_i}z_j(SR_{ij}-\mu_{ij})^2, \qquad (3.12)\end{aligned}$$

setting Equations (3.11) and (3.11) equal to to zero and multiplying them by $2\sigma_{i(1)}^4$ and $2\sigma_{i(2)}^4$, respectively, we obtain estimates of $\sigma_{i(1)}^2$ and $\sigma_{i(2)}^2$:

$$\widehat{\sigma}_{i(1)}^2 = \frac{\sum_{j=1}^{n_i}(1-z_j^*)\left(SR_{ij}-\widehat{\mu}_{ij}\right)^2}{\sum_{j=1}^{n_i} 1-z_j^*} \tag{3.13}$$

$$\widehat{\sigma}_{i(2)}^2 = \frac{\sum_{j=1}^{n_i} z_j^*\left(SR_{ij}-\widehat{\mu}_{ij}\right)^2}{\sum_{j=1}^{n_i} z_j^*} \tag{3.14}$$

The mixture parameter π is found fairly easily as:

$$\frac{\partial \ell(\boldsymbol{\Psi}; \boldsymbol{SR}_i, \boldsymbol{z})}{\partial \pi} = \sum_j -\frac{1-z_j}{1-\pi} + \frac{z_j}{\pi} = \sum_j \frac{-\pi + \pi z_j + z_j - \pi z_j}{\pi(1-\pi)} = \sum_j \frac{z_j - \pi}{\pi(1-\pi)},$$

implying that the estimate of π is:

$$\widehat{\pi} = \sum_{j=1}^{n_i} \frac{z_j^*}{n_i}. \tag{3.15}$$

Fitting the model using the EM algorithm, we use the **`regmixEM`**-function from the `mixtools` package. The **`regmixEM`**-function takes a parameter `arbmean`, that if set equal to `FALSE` assumes that the regression coefficients of the two components are equal. The parameter estimates can be seen in Tables 3.2 and 3.3.

Table 3.2: The parameter estimates of the Gaussian mixture model, using allele length as explanatory variable.

Locus	π_i	Intercept	Slope	σ_{i1}	σ_{i2}
CSF1PO	0.7465	-4.7868	0.1762	0.1535	0.2002
TPOX	0.5435	-5.9936	0.2470	0.1752	0.1752
D3S1358	0.6516	-5.1516	0.1704	0.1764	0.3244
D5S818	0.7811	-5.1658	0.2162	0.0492	0.2020
D16S539	0.8568	-5.2849	0.2214	0.1185	0.3383
D7S820	0.8803	-5.2874	0.2162	0.0794	0.2272
D8S1179	0.5389	-3.5647	0.0694	0.1369	0.3211
TH01	0.8009	-4.2860	0.0248	0.3908	0.3908
vWA	0.6626	-5.1531	0.1398	0.1274	0.8238
D21S11	0.5061	-3.2585	0.0314	0.2212	0.4168
D12S391	0.9947	-3.8980	0.0891	0.2112	1.1353

Table 3.3: The parameter estimates of the Gaussian mixture model, using LUS as explanatory variable.

Locus	π_i	Intercept	Slope	σ_{i1}	σ_{i2}
CSF1PO	0.6734	-4.8013	0.1776	0.1486	0.1969
TPOX	0.5226	-5.9936	0.2470	0.1752	0.1752
D3S1358	0.5849	-4.6074	0.2147	0.3251	0.4143
D5S818	0.7086	-5.0891	0.2094	0.0575	0.2271
D16S539	0.8568	-5.2849	0.2214	0.1185	0.3383
D7S820	0.8811	-5.2879	0.2164	0.0793	0.2271
D8S1179	0.7037	-4.4604	0.1557	0.1186	0.3886
TH01	0.5209	-6.1352	0.3006	0.1583	0.2219
vWA	0.7554	-4.5771	0.1720	0.1287	0.6021
D21S11	0.5936	-3.3270	0.1124	0.2137	0.3700
D12S391	0.8991	-2.8796	0.1084	0.2017	0.4904

Figure 3.7 shows QQ-plots of the residuals with respect to the two mixture models. The QQ-plots looks very similar.

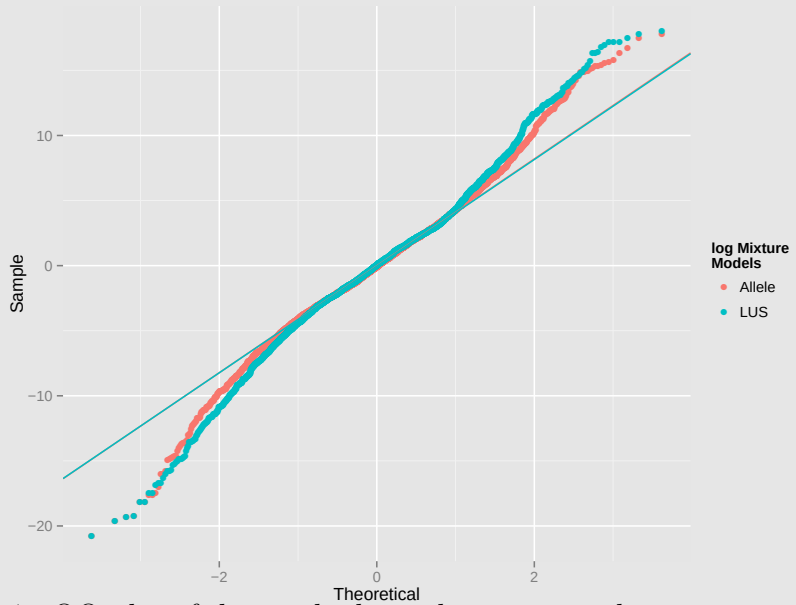


Figure 3.7: QQ-plot of the residuals, with respect to the mixture models using allele length (red) and LUS (blue).

3.2 The Gamma Model

The gamma model assumes that $SR_{ij} \sim \Gamma(\alpha_{ij}, \eta_{ij})$, where α_{ij} and η_{ij} are the shape and scale, respectively. Note that the scale is sometimes denoted as β, however, we have chosen η, as to avoid confusion w.r.t. the regression coefficients. Given $SR_{ij} \sim \Gamma(\alpha_{ij}, \eta_{ij})$, we have $\mathbb{E}[SR_{ij}] = \alpha_{ij}\eta_{ij}$ and $\mathbb{V}\text{ar}[SR_{ij}] = \alpha_{ij}\eta_{ij}^2$. This gives rise to the mean/dispersion parametrisation, where $\mathbb{E}[SR_{ij}] = \mu_{ij}$ and $\mathbb{V}\text{ar}[SR_{ij}] = \mu_{ij}\phi$. It is evident from this representation, that the gamma model can be used to capture over dispersion in the data. Furthermore, when fitting the model, we will use the logarithm as a link function for the gamma distribution. That is,

$$\log(\mu_{ij}) = \mathbb{E}[\log(SR_{ij})] = \beta_{i,1} + \beta_{i,2}X_{ij},$$

where X_{ij} is once again either the observed allele length or the LUS. Using a log-link implies that the variance will equal $\exp(\beta_{i,1} + \beta_{i,2}X_{ij})\,\phi$. The two gamma models are easily fitted using the **glm**-function in **R**, and the intercept, slope, and dispersion are seen in Table 3.4.

Table 3.4: The intercept, slope, and dispersion estimates for each locus of the allele and LUS covariate gamma models.

	Allele Length			LUS		
Locus	Intercept	Slope	ϕ	Intercept	Slope	ϕ
CSF1PO	-4.7762	0.1765	0.0285	-4.7886	0.1777	0.0283
TPOX	-5.9644	0.2455	0.0314	-5.9644	0.2455	0.0314
D3S1358	-5.1182	0.1705	0.0812	-4.3668	0.2008	0.1367
D5S818	-5.2070	0.2221	0.0345	-4.9001	0.1960	0.0464
D16S539	-5.3087	0.2255	0.0337	-5.3087	0.2255	0.0337
D7S820	-5.2624	0.2168	0.0490	-5.2626	0.2168	0.0490
D8S1179	-3.6067	0.0783	0.0787	-4.3958	0.1566	0.0732
TH01	-4.3887	0.0475	0.1780	-6.1052	0.2997	0.0417
vWA	-3.5738	0.0629	0.3452	-4.5499	0.1818	0.1831
D21S11	-2.7404	0.0198	0.1344	-3.3900	0.1223	0.1123
D12S391	-3.8651	0.0887	0.0505	-2.6780	0.0951	0.0636

We see from Table 3.4, that by using the LUS instead of the allele length, we lower generally lower the dispersion parameter, on loci such as TH01, vWA or D21.

3.3 Comparing the Stutter Models

In order for us to compare the models discussed above, we will examine the mean square error (MSE). The MSE will be locus specific, as we saw in Table 3.1 that the models will equivalent on some of the loci. Furthermore, we will exploit that we know the references are of Danish origin, and that we know the allele frequency within the Danish population, these frequencies have been provided by Susanne Lunøe Friis (the results have not yet been published and are therefore not included in this thesis). That is, we are looking at an expected MSE (EMSE), of locus L, defined as follows:

$$\text{EMSE}_L = \sum_{j \in A(L)} p_{Lj} \left(\bar{SR}_{Lj} - \widehat{SR}_{Lj} \right)^2, \tag{3.16}$$

where p_{Lj} is the observed allele frequency in the population, $A(L)$ indicates observed alleles on locus L, $\bar{SR}_{Lj}$ is the median of SR across the jth allele on locus L. Furthermore, $\widehat{SR}_{Lj}$ will be the predicted value of jth allele on locus L, using either the allele length or the LUS. As the LUS of an allele can take multiple values, we will use the mean across the allele.

Table 3.5, shows that for loci where the LUS and allele length differ greatly, i.e. TH01 and vWA, we can achieve a smaller error, by using the LUS. Furthermore, we do not see much difference in EMSE between the simple linear model, the Gaussian mixture model, or the gamma model.

Table 3.5: The EMSE of the linear models and Gaussian Mixture Models using allele length and LUS as covariates.

	Simple Linear Model		Mixture Model		Gamma Model	
Locus	Allele	LUS	Allele	LUS	Allele	LUS
CSF1PO	2.912e-03	2.917e-03	2.912e-03	2.921e-03	3.214e-03	3.246e-03
TPOX	6.272e-04	6.272e-04	6.272e-04	6.272e-04	1.043e-03	1.043e-03
D3S1358	8.355e-03	1.162e-01	5.753e-04	6.763e-03	1.081e-02	1.744e-01
D5S818	1.137e-02	1.171e-02	1.052e-02	1.101e-02	1.322e-02	1.358e-02
D16S539	6.100e-03	6.100e-03	4.414e-03	4.414e-03	6.870e-03	6.870e-03
D7S820	3.391e-02	3.390e-02	1.724e-02	1.721e-02	3.354e-02	3.355e-02
D8S1179	1.917e-02	1.300e-02	5.292e-03	3.356e-03	2.451e-02	1.838e-02
TH01	9.235e-02	4.026e-03	9.235e-02	4.163e-03	6.978e-02	3.755e-03
vWA	5.094e-01	2.256e-01	5.451e-01	2.655e-01	4.305e-01	2.052e-01
D21S11	2.048e-01	1.341e-01	2.069e-01	1.353e-01	1.933e-01	1.182e-01
D12S391	3.254e-01	3.505e-01	3.295e-01	3.425e-01	3.372e-01	3.692e-01

3.4 Shoulders

We will start out by treating the shoulders, as we did the stutters, namely by assuming that there is a linear relationship between the shoulder ratio (defined in analogue to Equation (3.1), using shoulder coverage). Figure 3.8 shows boxplots of the the shoulder ratio plotted against the parent allele length, for each locus in the IonTorrent data.

As seen in Figure 3.8 the relationship between the shoulder ratio and the parent allele length is maybe not as clear, or non-existing. Furthermore, we see on the reference files, as on the dilution series, that the shoulder ratio on the CSF1PO locus, is a lot higher than any other locus.

As the figure does not show use anything, other than the difference in shoulder ratio between loci is substantial, we will set locus depend thresholds, based on the median shoulder ratio plus three times the standard deviation. The resulting thresholds can

be seen in Table 3.6 and are represented by the blue line in Figure 3.8.

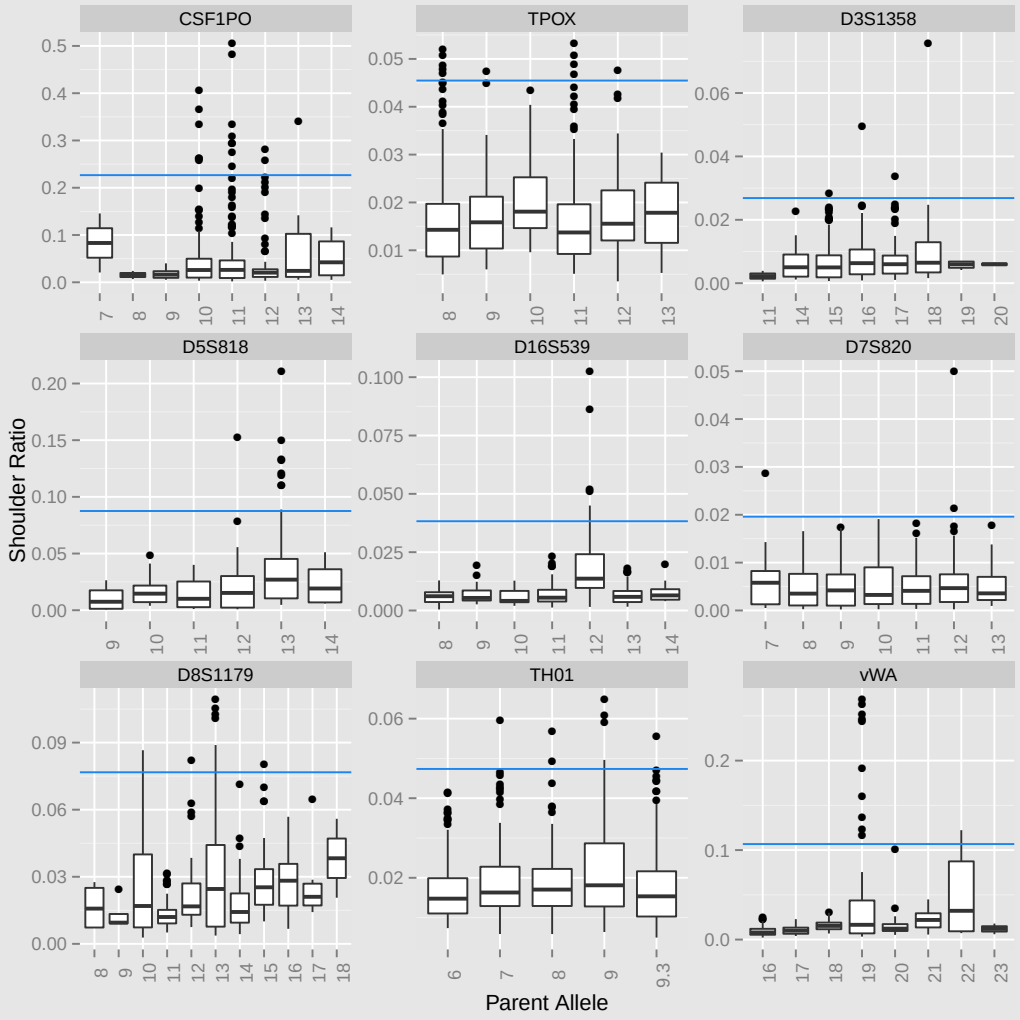


Figure 3.8: The shoulder ratio plotted against the allele length of the parent string, for each locus on the Ion PGM reference samples. The blue line represents the locus specific thresholds seen in Table 3.6.

Table 3.6: The locus specific shoulder thresholds.

CSF1PO	TPOX	D3S1358	D5S818	D16S539	D7S820	D8S1179	TH01	vWA
0.2267	0.0455	0.0269	0.0875	0.0383	0.0196	0.0767	0.0474	0.1067

CHAPTER

FOUR

NOISE ANALYSIS

The easiest way to deal with the observed noise, i.e. the insertion, deletions and erroneously called bases, would be to simply enforce a threshold upon the coverage, limiting ourselves to strings with a coverage above the threshold. The concept is well known from CE, where the threshold of 50 RFU (relative fluorescence units). The reason we enforce a threshold is that, in a mixture scenario, we would otherwise have to account for the observed drop-ins (which would be numerous, even with some being classified as stutters or shoulders) by allowing for a large number of unknown contributors. In a naïve approach to creating such a thresholds, one removes every string with 5%-10% of the total coverage of a given locus. However, we would instead like to model the string coverage on a given locus.

As the coverage of a string, is just a synonym for the number of occurrences of said string, we will use the negative binomial distribution. We use the negative binomial and not the Poisson distribution, as the Poisson distribution assumes that the mean and variance are equal. If we look at the coverage counts on a given locus, we will see an abundance of one's, as evident from in Table 4.1.

Our approach to modelling the noise, will therefore resemble modelling data with excess zero's, i.e. zero-inflated count models (in our case the zero-inflated negative binomial (ZINB) model [25]), which we will extend to a k-inflated negative binomial model, specifically using $k = 1$.

Table 4.1: The counts of coverages on the CSF1PO locus of the `LT_Dil_002_F_2ng` sample.

Coverage	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Count	508	62	37	26	12	15	12	7	3	5	5	10	6	7
Coverage	15	16	17	18	19	20	21	22	25	26	27	32	33	34
Count	3	4	6	1	3	5	4	1	1	2	2	1	2	1
Coverage	35	44	49	60	64	133	159	170	183	224	414	699	6882	8196
Count	1	1	1	1	1	1	1	1	1	1	1	1	1	1

4.1 k-Inflated Negative Binomial Model

The general idea of the k-inflated negative binomial (KINB) model, follows directly from the ZINB, that is it is a two component mixture model. The first component models the excess frequency of k's, while the second component models the remainder according to a negative binomial distribution. It should be noted, that the second component also generates k's, i.e. the word *excess*, used when describing the first component should be taken quite literally in this context. Furthermore, we assume that k is the smallest possible value. The two components can be described more precisely as follows:

$$f(x;\theta,\lambda,\pi,k) = \begin{cases} \pi + (1-\pi)f(k;\theta,\lambda), & \text{if } x = k \\ (1-\pi)f(x;\theta,\lambda), & \text{if } x > k \end{cases} \tag{4.1}$$

where x is a non-negative integer, λ is the mean, θ is the shape parameter (sometimes referred to as the dispersion parameter), π is the mixture parameter, and $f(\cdot;\theta,\lambda)$ is the probability mass function (pmf) of the negative binomial distribution. We will use the mean parametrisation, i.e. it follows that we use the following pmf the negative binomial distribution:

$$f(x;\theta,\lambda) = \left(\frac{\theta}{\theta+\lambda}\right)^{\theta} \frac{\Gamma(\theta+x)}{x!\Gamma(\theta)} \left(\frac{\lambda}{\theta+\lambda}\right)^{x}. \tag{4.2}$$

A consequence of the restriction $x \geq k$ is, that we are not only looking at a k-inflated model, but also a $k-1$ and k truncated model, for $x = k$ and $x > k$, respectively. That is, we have to substitute $f(x;\theta,\lambda,k)$ with $g(x;\theta,\lambda,k)$, where g is the truncated

distribution function (tdf) given as follows:

$$g(x;\theta,\lambda,k) = \mathbb{P}(x|x>k) = \frac{f(x;\theta,\lambda)}{1-F(k;\theta,\lambda)}, \tag{4.3}$$

where $F(\cdot;\theta,\lambda)$ is cumulative density function (cdf), of the negative binomial distribution. The cdf of the negative binomial distribution can be written as $F(k;\theta,\lambda) = 1 - I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)$, where $I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)$ is the regularised incomplete beta function, see Appendix Section A for more details.

4.1.1 Implementation of the KINB Method

We will consider two implementations of the KINB method: A direct maximum likelihood approach and what we will call the weighted fractile method, maximising the parameters based on two fractiles of the distribution, to isolate the extreme values (the alleles, stutters, and other systematic noise).

The Maximum Likelihood Approach

The first implementation of KINB (the **`kinb`**-function), is based on the **`zeroinfl`**-function from the `pscl`-package. The **`kinb`**-function, can handle truncation through the `truncation`-argument. If `TRUE` the log-likehood function is truncated as seen in Equation 4.3. The **`zeroinfl`**-function uses the **`optim`**-function with a pre-calculated gradient function of the log-likelihood, which as a consequence of the extension and truncation has to be recalculated. The log-likelihood of the truncated KINB model can be written as follows:

$$\begin{aligned}\ell(\theta,\lambda,\pi;\boldsymbol{x},k) &= \sum_{i\,:\,x_i=k} \log\left(\pi + (1-\pi)f(k;\theta,\lambda)\right) \\ &\quad + \sum_{i\,:\,x_i\neq k} \log\left((1-\pi)\right) + \log\left(\frac{f(x_i;\theta,\lambda)}{I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)}\right)\end{aligned} \tag{4.4}$$

We can expand Equation (4.4), using the pdf of the negative binomial distribution and differentiate with respect to the parameters θ, λ, and π. We split the differentiation into two parts, i.e. for $x_i = k$ and $x_i \neq k$, creating a vector for each case, the gradient then becomes the sum of these two vectors. We will begin with the simpler case, $x_i \neq k$:

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial\lambda} = -\frac{\theta+x_i}{\theta+\lambda} + \frac{x_i}{\lambda} - \frac{\partial}{\partial\lambda}\left[\log\left(I_{\frac{\lambda}{\lambda+\theta}}\left(k+1,\theta\right)\right)\right]$$

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial \theta} = \log(\theta) - \log(\theta+\lambda) + 1 - \frac{\theta+x_i}{\theta+\lambda} + \frac{\partial}{\partial\theta}\left[\log\left(\Gamma(\theta+x_i)\right)\right] - \frac{\partial}{\partial\theta}\left[\log\left(\Gamma(\theta)\right)\right] - \frac{\partial}{\partial\theta}\left[\log\left(I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)\right)\right]$$

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial \pi} = -\frac{1}{1-\pi}$$

For $x_i = k$ we need to differentiate the logarithm of sums:

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial \lambda} = \frac{\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)}{\pi+\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)}\left(-\frac{\theta+k}{\theta+\lambda}+\frac{k}{\lambda}\right)$$

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial \theta} = \frac{\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)}{\pi+\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)} \times \left\{\log(\theta)-\log(\theta+\lambda)+1-\frac{\theta+k}{\theta+\lambda}+\frac{\partial}{\partial\theta}\left[\log\left(\Gamma(\theta+k)\right)\right]-\frac{\partial}{\partial\theta}\left[\log\left(\Gamma(\theta)\right)\right]\right\}$$

$$\frac{\partial \ell(\theta,\lambda,\pi;x_i,k)}{\partial \pi} = \frac{\left(1-\frac{\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)}{1-\pi}\right)}{\pi+\exp\left(\log(1-\pi)+\log\left(f(k;\theta,\lambda)\right)\right)}$$

The only thing missing is to insert the derivatives of $\log(\Gamma(\theta+k))$, $\log(\Gamma(\theta+x_i))$, and $I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)$. Neither of these expressions is easily differentiated, however, the derivative of the logarithm of the gamma functions, is a well known function known as the digamma function and approximations of the digamma function has been implemented in **R**. As for the derivatives of the incomplete regularised beta function, $I_{\frac{\lambda}{\lambda+\theta}}(k+1,\theta)$, they can be found in Appendix A.

The Weighted Fractile Approach

The weighted fractile approach will be implemented using an EM mixture model angle. The log-likelihood function of the KINB model, seen in Equation (4.1), can also be written as a mixture model, in accordance with [26], as follows:

$$\ell(\theta,\lambda,\pi;\boldsymbol{x},\boldsymbol{z}) = \sum_{i=1}^{n} z_i \log(\pi)\,\mathbb{I}[x_i=k] + (1-z_i)[\log((1-\pi)f(x_i;\theta,\lambda))] \qquad (4.5)$$

Maximising the log-likelihood, we turn to the EM-algorithm and therefore take a closer look at the E- and M-step.

E-Step:

Using Equation (4.5) we can find z^*, as we did in Section 3.1.1:

$$z^* = \mathbb{E}\left[z_i | x_i, \theta, \lambda, \pi\right] = \frac{\pi \mathbb{I}\left[x_i = k\right]}{\pi \mathbb{I}\left[x_i = k\right] + (1-\pi) f(x_i; \theta, \lambda)} \tag{4.6}$$

M-Step:

The parameter π is updated by maximising the π-restricted log-likelihood function $\ell_\pi = \sum_{i=1}^n z_i \log(\pi)\, \mathbb{I}\left[x_i = k\right] + (1-z_i)\log(1-\pi)$. It follows that the derivative w.r.t. π is:

$$\begin{aligned}
\frac{\partial \ell_\pi}{\partial \pi} &= \sum_{i=1}^n \frac{z_i \mathbb{I}\left[x_i = k\right]}{\pi} - \frac{1-z_i}{1-\pi} \\
&= \sum_{i=1}^n \frac{z_i \mathbb{I}\left[x_i = k\right] - \pi z_i \mathbb{I}\left[x_i = k\right] - \pi + z_i \pi}{\pi(1-\pi)} \\
&= \sum_{i=1}^n \frac{z_i \mathbb{I}\left[x_i = k\right] - \pi(z_i \mathbb{I}\left[x_i = k\right] - z_i + 1)}{\pi(1-\pi)}
\end{aligned}$$

implying that the estimate of π is:

$$\widehat{\pi} = \frac{\sum_{i=1}^n z_i^* \mathbb{I}\left[x_i = k\right]}{\sum_{i=1}^n 1 - z_i^* \mathbb{I}\left[x_i \neq k\right]}. \tag{4.7}$$

In order to find θ and λ, we could maximise $\ell_{\theta,\lambda} = \sum_{i=1}^n (1-z_i)\log\left(f(x_i; \theta, \lambda)\right)$, with respect to θ and λ. However, as the aim is noise filtering and we want to isolate the high coverage strings, we fit the parameters λ and θ based on two fractiles. The fractiles will be weighted by $1 - z_i^*$ in accordance with $\ell_{\theta,\lambda}$. The fractiles used for fitting the model is given as pair $\boldsymbol{q} = (q_{lower}, q_{upper})$. As a consequence, we are no longer working with the MLE of the parameters. That is, we are no longer ensured non-decreasing log-likelihood values in the iterations of the EM-algorithm.

The lower fractile will be set to 0.1, as the excess one's are already handled by the first component, and we will use 0.9 for the upper fractile. The upper fractile was found by experimenting with different values choosing what seemed the most appropriate. We then calculate the corresponding weighted fractiles $\boldsymbol{q}^w = (q^w_{lower}, q^w_{upper})$, based on our data, and minimise the following function, using `solnp` in **R**:

$$L(\boldsymbol{q}^w, \boldsymbol{q}, \lambda, \theta) = \left(\left[\left\{ \sum_{i=1}^{q^w_{lower}} f(i; \lambda, \theta) \right\} - q_{lower} \right] + \left[\left\{ \sum_{i=1}^{q^w_{upper}} f(i; \lambda, \theta) \right\} - q_{upper} \right] \right)^2,$$

where $f(\cdot; \lambda, \theta)$ is the density of the negative binomial distribution. Analogous to the non-mixture approach we switch $f(\cdot; \lambda, \theta)$ with $f(\cdot; \lambda, \theta)/I_{\lambda/(\lambda+\theta)}(k, \theta)$, i.e. the truncated negative binomial distribution.

We have in Appendix B examined how well these two methods estimate the parameters of simulated one inflated (and zero truncated) negative binomial data. We examine both a truncated and non-truncated **`kinb`**-function, as well as a truncated weighted fractile function, called **`fitnbinom.weighted.truncated`**. A non-truncated version of the weighted fractile method has also been implemented, the **`fitnbinom.weighted`**-function, and tested. However, we have not included it in this project, due to inferior performance.

4.2 Noise Threshold

Using the estimates of the mean and size parameters (λ and θ respectively), obtained from the **`kinb`**-method, we will create a noise threshold, t^{kinb}, in order to filter out anything that is noise components and not observations caused by more systematic mechanisms. The threshold will be based on a quantile, of the corresponding distribution, plus some scalar, $s \in \mathbb{N}_0$, times the standard deviation:

$$t^{\text{kinb}} = q + s\left(\lambda + \frac{\lambda^2}{\theta}\right)^{1/2},$$

where the quantile q is depends on a probability $p \in [0.9, 0.99995]$, as well as the parameters θ and λ. Likewise we create a threshold t^{wf} based on the parameters $\tilde{\theta}$ and $\tilde{\lambda}$ estimated using the weighted fractile method.

We calibrate the thresholds indirectly, by calibrating the parameters s and p. The parameters will be chosen such that the number of drop-ins are as low as possible in accordance with Section 1.6.2. We can divide the drop-ins into four types as seen in List 4.2.1.

List 4.2.1

(i) Stutters

(ii) Shoulders (high coverage .1 or .3 strings seen around an allele)

(iii) Strings of similar length as the true alleles

(iv) General noise

We do not care if stutters or shoulders pass through the noise threshold, as they are systematic errors for which we can account. A drop-in of type (iii) could be: (1) a homozygote with two different strings, (2) a string from another contributor in the case of DNA mixtures, or (3) a variation on the true allele (i.e. a string with an erroneously called base(s)). The calibration of the parameters will still be based on achieving as few drop-ins as possible, but we will allow drop-ins of type (i)-(iii), leaving general noise (iv), from this point forth referred to as adjusted drop-ins. We have already proposed solutions for types (i)-(iii), in Chapter 3 and Section 2.5, leaving general noise.

If ties occur we take p and s to be the smallest values possible. Furthermore, as there is still quite a difference in coverage between loci, these parameters will be locus specific.

Calibrating the thresholds

We create a grid for both $p = 0.9, 0.95, 0.99, 0.995, ..., 0.99995$ and $s = 0, 0.1, 0.2, ..., 15$. Furthermore, we do this for the entirety of the dilution series, and take the $\widehat{p}_L$ and $\widehat{s}_L$ as the maximum value across samples.

Table 4.2: The calibrated scalar and probability parameters, for both the KINB and weighted fractile thresholds.

	KINB		Weighted Fractile	
Locus	$\widehat{s}_L$	$\widehat{p}_L$	$\widehat{s}_L$	$\widehat{p}_L$
CSF1PO	0	0.9999	13.9	0.99995
TPOX	0	0.9950	6.5	0.99995
D3S1358	0	0.9999	5.9	0.99995
D5S818	0	0.9995	3.3	0.99995
D16S539	0	0.9999	1.1	0.99995
D7S820	0	0.9950	5.9	0.99995
D8S1179	0	0.9900	0.0	0.99000
TH01	0	0.9900	0.0	0.95000
vWA	0	0.9995	8.8	0.99995

The calibrated values $\widehat{p}_L$ and $\widehat{s}_L$, can be seen in Table 4.2, for both the t^{kinb} and t^{wf}. We see that $\widehat{s}_L$ for the **kinb**-method is calibrated to zero for all loci, i.e. the threshold t^{kinb} is based on the quantile alone.

Using the calibrated values, $\widehat{s}_L$ and $\widehat{p}_L$, we can recalculated the drop-in and drop-out rate for each sample. The resulting thresholds and rates can be seen in Table 4.3, for sample `LT_Dil_002_F_2ng`. Note: string drop-ins, refers to drop-ins of List 4.2.1 type (iii).

We see from Table 4.3 that the some of the thresholds, the once corresponding to locus D5 and vWA in particular, are quite similar even though their calibrated parameters are different. Furthermore, the thresholds for sample `LT_Dil_002_F_2ng` can be seen in Figure 4.1, for the TPOX locus.

The total number of drop-ins and drop-outs per file can be seen in Table 4.4. We see that we impose three drop-outs using **kinb**, but eight drop-outs (with three locus drop-outs) using the weighted fractile method. The coverage of the alleles dropping out using the **kinb**-method ranges 37 to 125 with a median of 67, which makes the drop-outs quite understandable. The coverage range of the dropped alleles using the weighted fractile method, is 37 to 402, with a median of 172. Furthermore, we see that seven of seven adjusted drop-ins found with the **kinb**-method, occur on the same file, actually on the same locus (CSF1PO), and as seen in Figure 4.2, this particular locus has a lot of .3 strings with higher coverage than we would expect. In fact 18 of the 19 adjusted drop-ins seen with weighted fractile method occur as .3 on the CSF1PO locus as well. These .3 strings are an artefact of the PCR process, and we might have to adjust the threshold on this locus in general to ensure that such drop-ins do not occur.

Comparing our approach to the more naïve method described in the introduction of the chapter, setting a threshold at 5% of the total coverage on a given locus yields: 0 drop-outs, 32 drop-ins, and 0 adjusted drop-ins. We see that this way imposes no drop-outs and has no adjusted drop-ins, yet also includes very little of the systematic noise (only 32 drop-ins on a total of 216 loci). That is, if one would like to include the systematic noise in ones evaluation, for the reasons mentioned in the beginning of Chapter 3, our method would be preferred, if not, the naïve approach seems like a fair choice.

Table 4.3: The t^{kinb} and t^{wf} thresholds, as well as the number of drop-ins (all types) and drop-outs, for the `LT_Dil_002_F_2ng` sample using the calibrated $\widehat{s}_L$ and $\widehat{p}_L$ parameters. Each row of the four rightmost columns, sums to the value of Drop-in in the corresponding row.

k-inflated Negative Binomial Model

Locus	t^{kinb}	Drop-out	Drop-in	String	Stutter	Shoulder	Adjusted Drop-in
CSF1PO	256.00	0	2.00	0.00	2	0	0
TPOX	89.00	0	2.00	0.00	1	1	0
D3S1358	275.00	0	2.00	0.00	2	0	0
D5S818	185.00	0	2.00	0.00	2	0	0
D16S539	271.00	0	1.00	0.00	1	0	0
D7S820	103.00	0	2.00	0.00	2	0	0
D8S1179	62.00	0	7.00	1.00	1	5	0
TH01	50.00	0	0.00	0.00	0	0	0
vWA	164.00	0	2.00	0.00	2	0	0

Weighted Fractile Negative Binomial Model

Locus	t^{wf}	Drop-out	Drop-in	String	Stutter	Shoulder	Adjusted Drop-in
CSF1PO	183.59	0	3.00	0.00	2	1	0
TPOX	162.77	0	2.00	0.00	1	1	0
D3S1358	203.23	0	2.00	0.00	2	0	0
D5S818	175.43	0	2.00	0.00	2	0	0
D16S539	143.79	0	2.00	0.00	1	1	0
D7S820	82.41	0	2.00	0.00	2	0	0
D8S1179	46.00	0	9.00	2.00	1	6	0
TH01	9.00	0	4.00	0.00	2	2	0
vWA	250.84	0	2.00	0.00	2	0	0

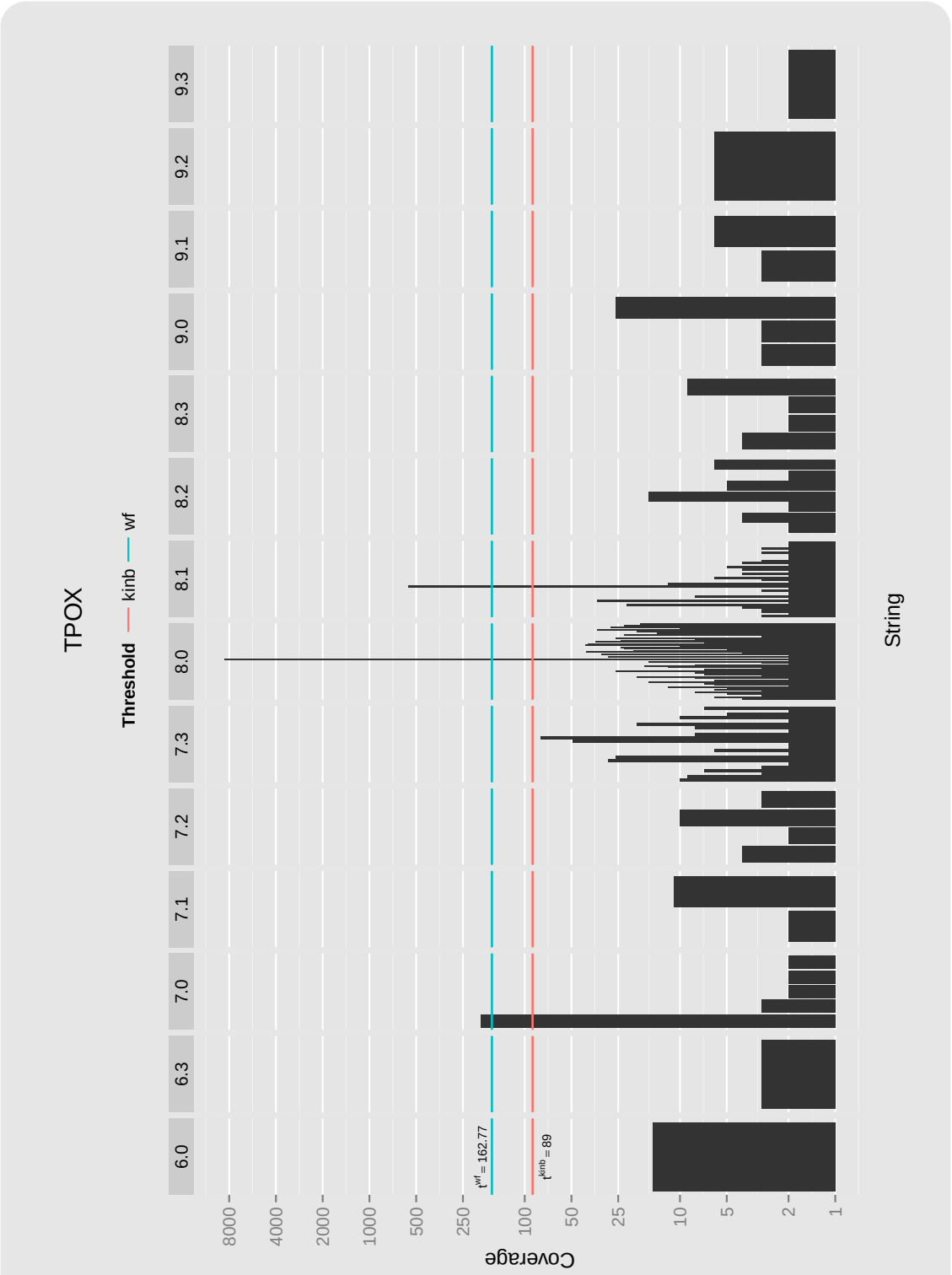


Figure 4.1: The coverage of all strings on locus TPOX, including the t^{kinb} and t^{wf} thresholds, for the `LT_Dil_002_F_2ng` sample. Furthermore, the ordinate-axis is shown on the $\log_{10}$-scale.

Table 4.4: Total number of drop-ins and -outs, for each sample using the calibrated $\widehat{s}_L$ and $\widehat{p}_L$ parameters.

	One-inflated Negative Binomial				Weighted Fractile			
Sample	Drop-out	Drop-in	Adj. Drop-in	Locus Drop-out	Drop-out	Drop-in	Adj. Drop-in	Locus Drop-out
002 F 2000pg	0	20	0	0	0	28	0	0
003 F 1000pg	0	13	0	0	0	24	0	0
004 F 500pg	0	16	0	0	0	25	0	0
005 F 200pg	0	11	0	0	2	19	0	1
006 F 100pg	1	15	0	0	2	29	1	1
007 F 50pg	0	21	0	0	0	33	0	0
008 H 2000pg	0	21	0	0	0	31	1	0
009 H 1000pg	0	17	0	0	0	29	0	0
010 H 500pg	0	17	0	0	0	27	0	0
011 H 200pg	0	22	0	0	0	30	0	0
012 H 100pg	0	19	0	0	0	29	0	0
013 H 50pg	0	18	0	0	0	33	0	0
014 F 2000pg	0	20	7	0	0	40	10	0
015 F 1000pg	0	6	0	0	0	17	0	0
016 F 500pg	0	5	0	0	0	20	0	0
017 F 200pg	0	7	0	0	0	19	0	0
018 F 100pg	0	4	0	0	2	14	0	1
019 F 50pg	0	6	0	0	0	21	0	0
020 H 2000pg	0	10	0	0	0	24	0	0
021 H 1000pg	0	10	0	0	0	23	0	0
022 H 500pg	0	23	0	0	0	46	7	0
023 H 200pg	2	0	0	0	8	4	0	3
024 H 100pg	0	9	0	0	0	19	0	0
025 H 50pg	0	7	0	0	0	16	0	0
Total	3	317	7	0	14	600	19	6

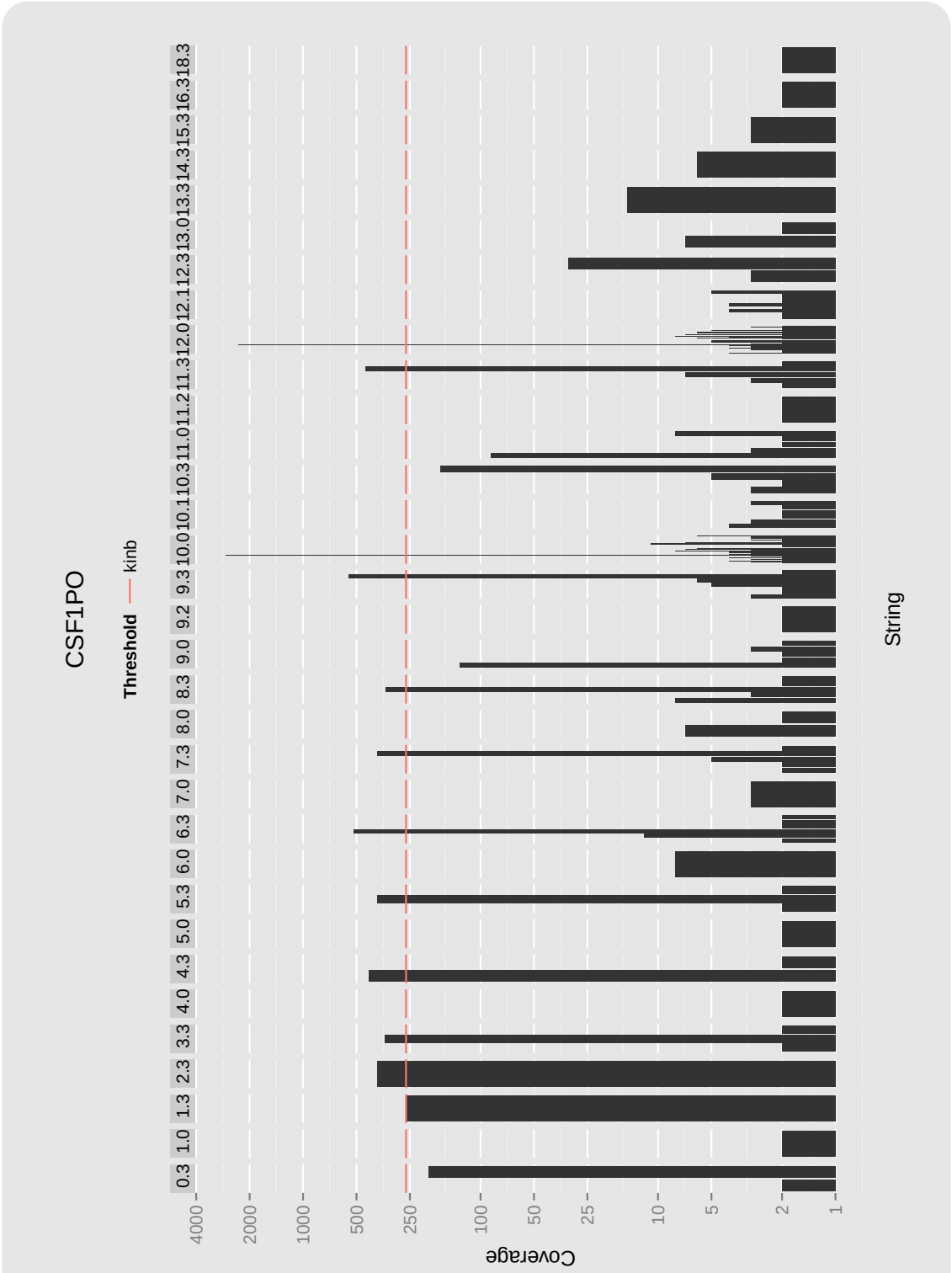


Figure 4.2: The coverage of all strings on locus CSF1PO, including the t^{kinb} threshold, for the `LT_Dil_014_F_2ng` sample. Note: the ordinate-axis is shown on the $\log_{10}$-scale.

CHAPTER

FIVE

PROBABILITY OF DROP-OUT

In Chapter 4 we, due to applying a threshold, induce drop-outs in data that otherwise would not have had any drop-outs. The goal in this chapter is to estimate the probability of such a drop-out occurring. In order to achieve this goal, we first examine the heterozygote balance, Section 5.1, in hopes of finding a possible explanatory variable. Using what we learn in Section 5.1, we further define the probability of drop-out in Section 5.2. Noting that we have missing information, we introduce, apply, and implement a version of the general EM-algorithm in Sections 5.3-5.1 and finally estimate the probability of drop-out in Section 5.6.

5.1 Heterozygote Imbalance

To examine the imbalance between two alleles, we will genotype the entirety of our dilution series (in accordance with Section 1.6), and then observe the relative height between the two peaks of a heterozygote, known as the heterozygote balance, H_b:

$$H_b = \frac{\varphi_{\text{HMW}}}{\varphi_{\text{LMW}}}, \tag{5.1}$$

where HMW and LMW refers to the highest and lowest molecular weight alleles, respectively, and φ is the coverage. There is another common way of defining H_b: $H_b' = \varphi_{\text{smaller}}/\varphi_{\text{larger}}$, which ensures that $H_b' \in [0, 1]$. However, Equation (5.1) offers

more detail, as its definition is fixed (the HMW is always the numerator) and it is therefore preferred, for further explanation see [27, Section 5.1].

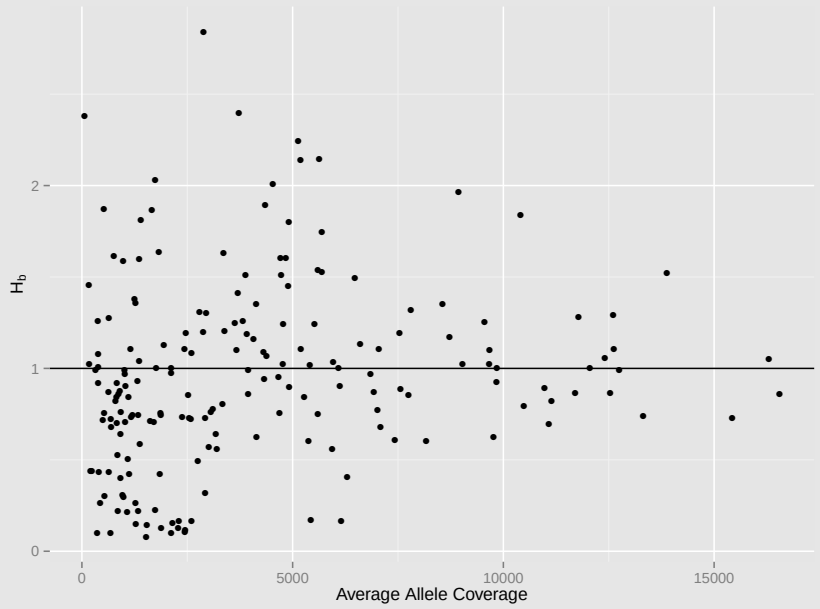


Figure 5.1: The heterozygote balance against the average coverage of the alleles or more specifically $(\varphi_{\text{HMW}} + \varphi_{\text{LMW}})/2$.

The shape, seen in Figure 5.1, is not what we have come to expect from CE, which has of a more trumpet shape, see e.g. [27, Figure 1]. That is, we would normally expect an increase in variability, as the amount of DNA decreases. This seems to indicate that the coverage is not be an apt indicator for the amount of input DNA, which is even more evident when plotting the average allele coverage against the amount of DNA, as seen in Figure 5.2.

The simple answer is that between the PCR and the emPCR processes, the sample is normalised yielding approximately the same amount of sequences across a sample no matter the amount of input DNA. In order to more fully understand, we need to consider how we have sequenced the dilution series. Each series ranging from 2ng to 0.5ng has been sequenced on a single chip, using MID's to identify the samples. The sample is then normalised based on these MID sequences. That is, we select approximately the same amount of reads per MID. It thereby follows that the coverage is not a good indicator for the amount of input DNA. Note in the case of IonTorrent the MID's are named barcodes.

However, even with MID sampling (normalisation), theoretically as the amount

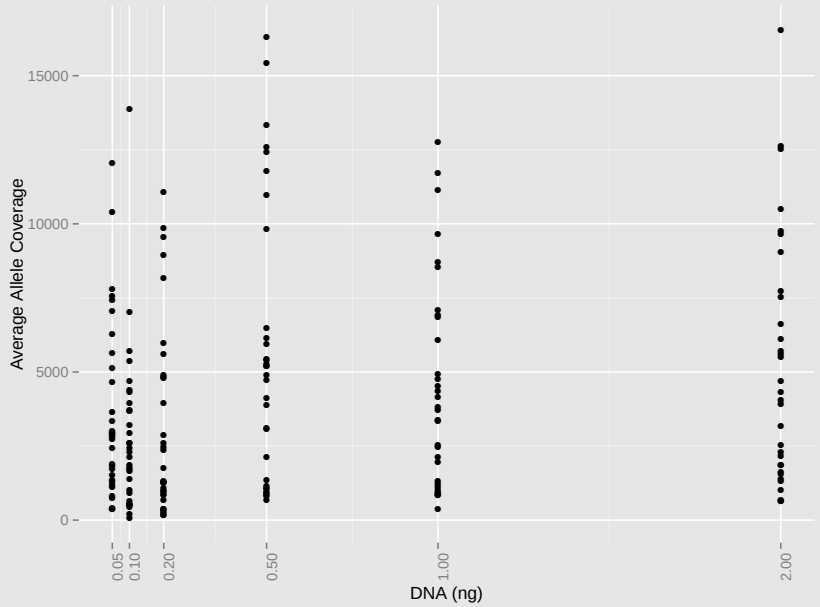


Figure 5.2: The observed average allele coverage against the amount of DNA (ng), of both dilution series from contributors F and H.

of input DNA decreases, the variability of the heterozygote balance should still increase. We will examine this hypothesis using simulated data. We can simulate the PCR process by using the binomial sampling model first introduced in [4]. Given n copies of input DNA, the sampling process can be broken down as follows:

(i) n_0 DNA copies are extracted for PCR amplification. This process is simulated by a binomial model, and a molecule is selected with probability $\pi_{aliquot}$.

(ii) The PCR process is also simulated using a binomial model, where the number of DNA copies in cycle t is given as:

$$n_t = n_{t-1} + \text{Bin}\left(n_{t-1}, \pi_{PCReff}\right),$$

where π_{PCReff} indicates the PCR efficiency, which in [4] is assumed to continue throughout the PCR process, i.e. independent of cycle number.

The values of the parameters, $\pi_{aliquot}$ and π_{PCReff}, will be chosen in accordance with [4]. That is, $\pi_{aliquot} = 20/66$ and $\pi_{PCReff} = 0.8$. The amount of DNA copies and the number of PCR cycles, will be 1000 and 25, respectively. Furthermore, we dilute the sample 6 times, replicate the entire experiment a thousand times, and assume we have 9 loci (as we would with a 10plex sample), denoted by L.

As an extension of the process described in [4], we add on MID sampling as a final step. In order to simulate the MID sampling, we first calculate, given n_{MID}, the number of samples per MID, MID_{chip}, sampling from a gamma distribution with shape and scale equal n_{MID} and one, respectively. The number of reads, MID_{chip}, gets distributed among the L loci using a multinomial distribution with size equal to MID_{chip} and probability vector $\boldsymbol{\pi}_{PCRLocusEff}$. The $\boldsymbol{\pi}_{PCRLocusEff}$ vector indicates the proportion of MID_{chip} assigned to each locus. These proportions are found, for a given locus l, by dividing the number of reads of locus l by the total number of reads.

Input: N, $\pi_{aliquot}$, π_{PCReff}, $\boldsymbol{\pi}_{PCRLocusEff}$, t, n_{MID}, DNA_0, L, and d.
replicate N **time**(s)
  for i **from** 0 **to** d :
    $\text{DNA} = \textbf{round}\left(\frac{\text{DNA}_0}{2^i}\right)$
    for l **from** 1 **to** L :
      for a **from** 1 **to** 2 :
        $\text{n}^l_{0,a} = \text{Bin}\left(\text{DNA}, \pi_{aliquot}\right)$
        for j **from** 1 **to** t :
          $\text{n}^l_{j,a} = \text{n}^l_{j-1,a} + \text{Bin}\left(\text{n}^l_{j-1,a}, \pi_{PCReff}\right)$
        end
      end
      $\text{MID}_{reads,l} = \text{n}^l_{t,1} + \text{n}^l_{t,2}$
    end
    $\text{MID}_{chip} = \Gamma\left(\text{n}_{\text{MID}}, 1\right)$
    $\text{MID}_{reads} = \sum_l \text{MID}_{reads,l}$
    $\textbf{MID}_{chip,L} = \text{Mult}(\text{MID}_{chip}, \boldsymbol{\pi}_{PCRLocusEff})$
    for l **from** 1 **to** L :
      for s **from** 1 **to** $\text{MID}^l_{chip,L}$:
        $\text{MID}_{sample} = \mathcal{U}\left(1, \text{MID}^l_{reads}\right)$
        $\text{allele}_{s,1} = \text{MID}_{sample} < \text{n}^l_{t,1}$
      end
      $\text{Coverage}^l_1 = \sum_s \text{allele}^1_s$
      $\text{Coverage}^l_2 = \text{MID}^l_{chip} - \text{Coverage}^l_1$
    end
    $\textbf{Hb}_{\text{d}} = \textbf{Coverage}_2/\textbf{Coverage}_1$
  end
Output: N replicates of the simulated heterozygote balance for every dilution d.

Listing 5.1: Pseudo code simulating the PCR and MID sampling process.

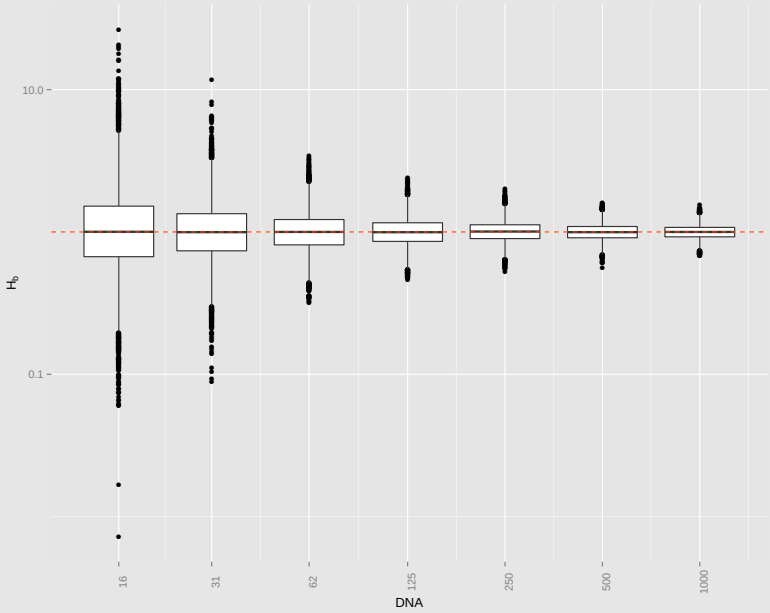


Figure 5.3: H_b against the amount of DNA, of data simulated using a variation of the binomial PCR sampling model [4], modified to include MID sampling. The ordinate and abscissa-axis is shown on a $\log_{10}$-scale. The red dashed line indicates a value of one.

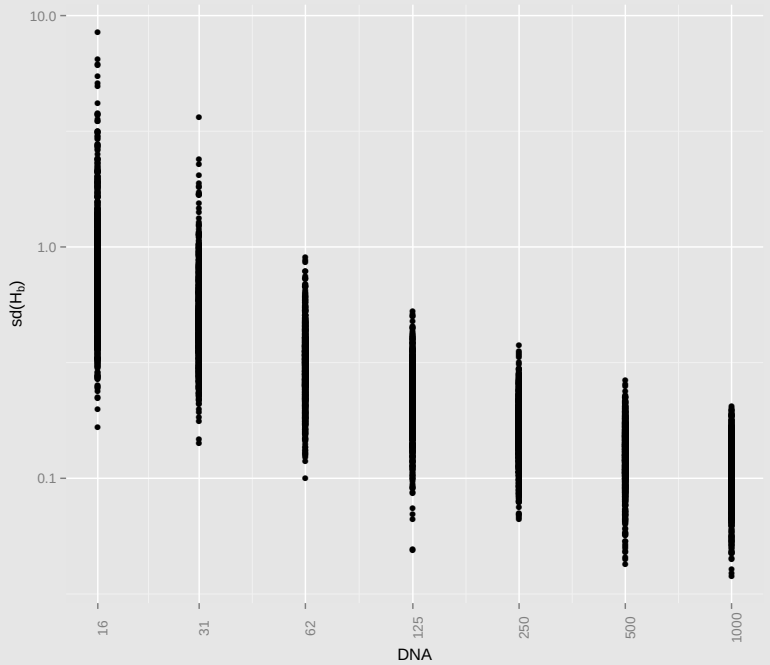


Figure 5.4: The standard deviation of the simulated heterozygote balance plotted against the amount of template DNA. The ordinate and abscissa-axis are shown on a $\log_{10}$-scale.

We then sample the coverage of both alleles for each locus and calculate the heterozygote balance. The pseudo code for everything discussed above can be seen in listing 5.1. The simulated H_b can be seen in Figure 5.3. The figure shows that the hypothesis should hold, i.e. as the amount of input DNA decreases, the variability of the heterozygote balance increases. An observation further supported by Figure 5.4 showing the standard deviation of H_b plotted against the amount of DNA.

5.2 Probability of Drop-out

One of the consequences of using a noise threshold, as seen in Section 4.2, is the introduction of more drop-outs (as a drop-out may occur when an allele fails to amplify or when the coverage is below some preset threshold T). By imposing a threshold $T = 250$ on our simulated data, we can calculate the drop-out frequency $\mathbb{P}(D)$. Note that the choice of this threshold value is based solely on values produced by the simulations.

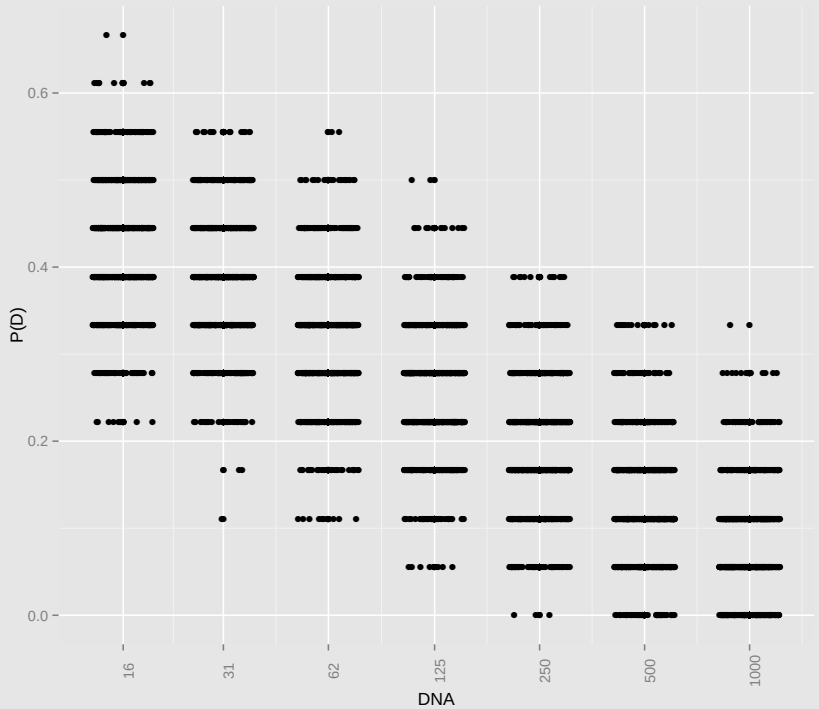


Figure 5.5: The probability of drop-out plotted against the amount of template DNA for simulated data.

Plotting $\mathbb{P}(D)$ against the amount of template DNA, seen in Figure 5.5, we see

that the probability of drop-out decreases when the amount of DNA increases. We have combined this fact with the information observed in Figure 5.4, plotting the probability of drop-out against $\mathrm{sd}\,(H_b)$, shown in Figure 5.6.

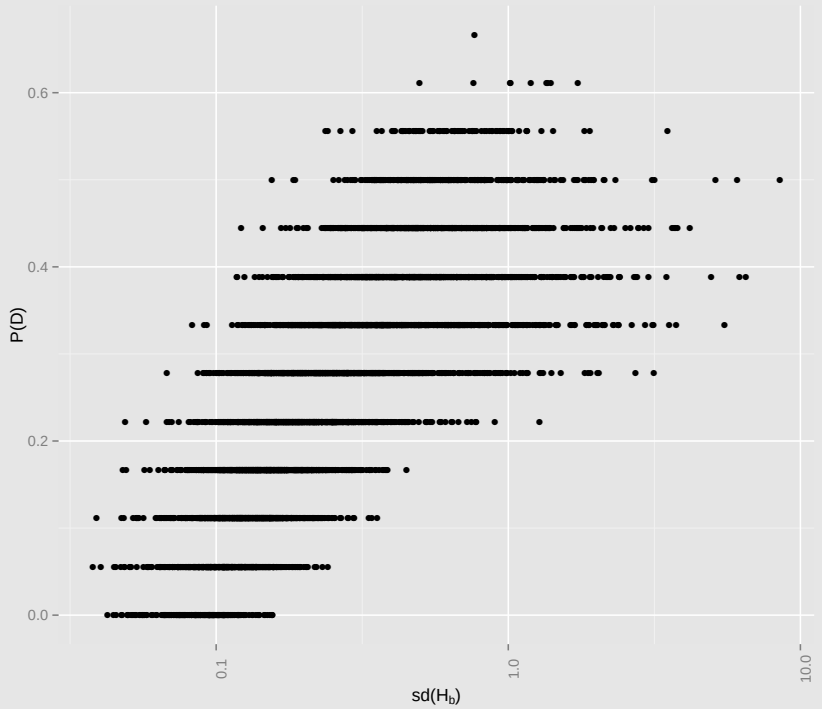


Figure 5.6: The probability of drop-out plotted against the standard deviation of the heterozygote balance. The abscissa-axis is shown on a $\log_{10}$-scale.

Figure 5.6 shows that as the probability of drop-out increases so does the standard deviation of the heterozygote balance. With that being said, this only holds if we use the complete information, φ, when calculating the standard deviation. However, the given a threshold T, we do not observe φ, but C defined as follows:

$$C = \begin{cases} \varphi, & \text{if } \varphi \geq T \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

The definition of C and relationship between $\mathbb{P}(D)$ and $\mathrm{sd}\,(H_b)$, implies that the probability of drop-out can be written like so:

$$\mathbb{P}(D) = \mathbb{P}(C = 0 \mid \mathrm{sd}\,(H_b)). \tag{5.3}$$

In order to obtain an estimate of $\mathrm{sd}\,(H_b)$ based on the complete information, φ, rather than only the observed information, C, we once again turn to the EM-algorithm. We will not estimate $\mathrm{sd}\,(H_b)$ directly. Instead we choose a distribution

for the complete coverage, φ, and impute the missing coverage. The main justification is that if just one allele, on a given locus, is not observed then H_b is not observable. That is, by trying to impute H_b directly, we would ignore perfectly good information.

5.3 The General EM-Algorithm

Until now we have only used the EM-algorithm the parameters in finite mixture models in order to impute indicator variables. However, another frequent use of the EM-algorithm, is to estimate the parameters of a model, when some of the observations are either missing or unobservable [28, 29].

We denote the full data by $\boldsymbol{Y}$ and define it such that $\boldsymbol{Y} = (\boldsymbol{Y}_{\text{obs}}, \boldsymbol{Y}_{\text{mis}})$, where $\boldsymbol{Y}_{\text{obs}}$ and $\boldsymbol{Y}_{\text{mis}}$ denotes the observed and missing data, respectively. Furthermore, we will assume that our data is missing at random (MAR), see e.g. [30], as opposed to missing completely at random (MCAR) or missing not at random (MNAR). The implication of MAR is that the likelihood of the complete data can be decomposed as follows:

$$f_{\boldsymbol{Y}}(\boldsymbol{y};\boldsymbol{\theta}) = f_{\boldsymbol{Y}}(\boldsymbol{y}_{\text{obs}}, \boldsymbol{y}_{\text{mis}};\boldsymbol{\theta}) = f_{\boldsymbol{Y}_{\text{obs}}}(\boldsymbol{y}_{\text{obs}};\boldsymbol{\theta}) f_{\boldsymbol{Y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}}(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}};\boldsymbol{\theta}). \tag{5.4}$$

That is, given the observed data, the missing data is independent of the observed data (i.e. they are conditionally independent). The general EM-algorithm [31], is shown in Algorithm 5.3.1.

Algorithm 5.3.1 (The EM Algorithm.)

(1) Make initial guesses of the parameters $\boldsymbol{\theta}^{(0)}$

(2) **E-step**: At the jth iteration, compute:

$$q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(j)}) = \mathbb{E}\left[\ell_Y(\boldsymbol{\theta}) \mid y_{obs}, \boldsymbol{\theta}^{(j)}\right]$$

(3) **M-step**: Determine the new estimate $\boldsymbol{\theta}^{(j+1)}$ as:

$$\boldsymbol{\theta}^{(j+1)} = \arg\max_{\boldsymbol{\theta}}\{q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)})\}$$

(4) Repeat steps (2) and (3) until convergence, i.e. until $L(\boldsymbol{\theta}^{(j+1)}) - L(\boldsymbol{\theta}^{(j)}) < \varepsilon$.

A consequence of using this approach is that we need the distribution of the complete data. If the distribution of the complete data belongs to the exponential family, the calculations in the E-step of the algorithm simplifies greatly. To be more precise it simplifies to calculating the expectation of the sufficient statistics of the complete data. A nice feature of the EM-algorithm is, that it creates a non-decreasing series of likelihoods. Our proof follows that seen in [28].

Proof: (Non-decreasing Likelihood)
We assume that the data is MAR, and it follows by Equation (5.4) that the log-likelihood of the observed data can be written as follows:

$$\ell(\boldsymbol{\theta}; \boldsymbol{y}_{\text{obs}}) = \ell(\boldsymbol{\theta}; \boldsymbol{y}) - \log\left(f(\boldsymbol{Y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta})\right). \tag{5.5}$$

If we then take the mean of the above equation w.r.t. $\boldsymbol{Y}_{\text{mis}}$ given the incomplete data $\boldsymbol{y}_{\text{obs}}$ and the current parameters $\boldsymbol{\theta}^{(j)}$, we have:

$$\begin{aligned}
\ell(\boldsymbol{\theta}; \boldsymbol{y}_{\text{obs}}) &= \mathbb{E}_{\boldsymbol{y}_{mis}}\left[\ell(\boldsymbol{\theta}; \boldsymbol{y}) - \log\left(f(\boldsymbol{Y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta})\right) \mid \boldsymbol{y}_{\text{obs}}, \boldsymbol{\theta}^{(j)}\right] \\
&= \int \ell(\boldsymbol{\theta}; \boldsymbol{y}) f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)}\right) d\boldsymbol{y}_{\text{mis}} \\
&\qquad - \int \log\left(f(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta})\right) f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)}\right) d\boldsymbol{y}_{\text{mis}}.
\end{aligned}$$

The first term in this last equation is clearly $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)})$, the last term we will define as $H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)})$. Before we continue it is worth noting the following:

$$\begin{aligned}
H\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right) - H\left(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)}\right) &= \int \log\left(\frac{f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)}\right)}{f(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta})}\right) f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)}\right) d\boldsymbol{y}_{\text{mis}} \\
&\geq -\log\left(\int \frac{f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}\right)}{f(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)})} f\left(\boldsymbol{y}_{\text{mis}}|\boldsymbol{y}_{\text{obs}}; \boldsymbol{\theta}^{(j)}\right) d\boldsymbol{y}_{\text{mis}}\right) \\
&= 0.
\end{aligned}$$

We have here used Jensen's inequality of convex functions and that a density function integrates to 1. Furthermore, this inequality holds for all $\boldsymbol{\theta}$. Given a sequence of parameter estimates, we have:

$$\begin{aligned}
\ell(\boldsymbol{\theta}^{(j+1)}) - \ell(\boldsymbol{\theta}^{(j)}) &= \left[q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right) - q\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right)\right] - \left[H\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right) - H\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right)\right] \\
&\geq \left[q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right) - q\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right)\right],
\end{aligned}$$

where we have used that $H\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right) > H\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right)$. As $\boldsymbol{\theta}^{(j+1)}$ is found by

$$\boldsymbol{\theta}^{(j+1)} = \arg\max_{\boldsymbol{\theta}} q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)}),$$

it follows that

$$q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right) \geq q\left(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j)}\right)$$

which implies:

$$\ell(\boldsymbol{\theta}^{(j+1)}) \geq \ell(\boldsymbol{\theta}^{(j)}).$$

That is, the EM-algorithm ensures that the likelihood is non-decreasing. □

One disadvantage of using the EM-algorithm, as seen in [31], is that the convergence is linear with a rate proportional to the amount of observed data, i.e. the more missing data, the slower the rate of convergence.

5.4 The Distribution of the Complete Coverage

We will, for the choice of distribution, seek inspiration in the work which has been done in DNA profiling with regard to CE, in particular [32–34]. That is, we assume (in contrast to [32–34]), that the coverage follows a gamma distribution and not the florescent intensities (peak heights). As seen in Figure 5.7, showing a gamma QQ-plot of the coverage (from the dilution series), this assumption is not entirely unfounded. The QQ-plots corresponding to the remaining samples, as well as QQ-plots for the simulated data, can be seen in Appendix D.

In particular for a sample s, locus l, and allele a, we assume that the coverage $\varphi_{sla} \sim \Gamma(\alpha_s, \eta_{sl})$, and that the number of alleles on locus l is given by n_l. We let both parameters be dependent on the sample as this method of imputation should hold on a sample by sample basis. Generally we drop the sample index s when it is clear from context. That is, the pdf of φ_{la} is given as:

$$g(\varphi_{la}; \alpha, \eta_l) = \frac{\varphi_{la}^{\alpha-1}\exp\left(-\frac{\varphi_{la}}{\eta_l}\right)}{\eta_l^{\alpha}\Gamma(\alpha)}.$$

Furthermore, as we remove coverage below a threshold T, the distribution of the complete coverage looks as follows:

$$f(\boldsymbol{\varphi}_l; \alpha, \eta_l) = \prod_{a:\varphi_{la}\geq T} g(\varphi_{la}; \alpha, \eta_l) \prod_{a:\varphi_{la}<T} G(T; \alpha, \eta_l),$$

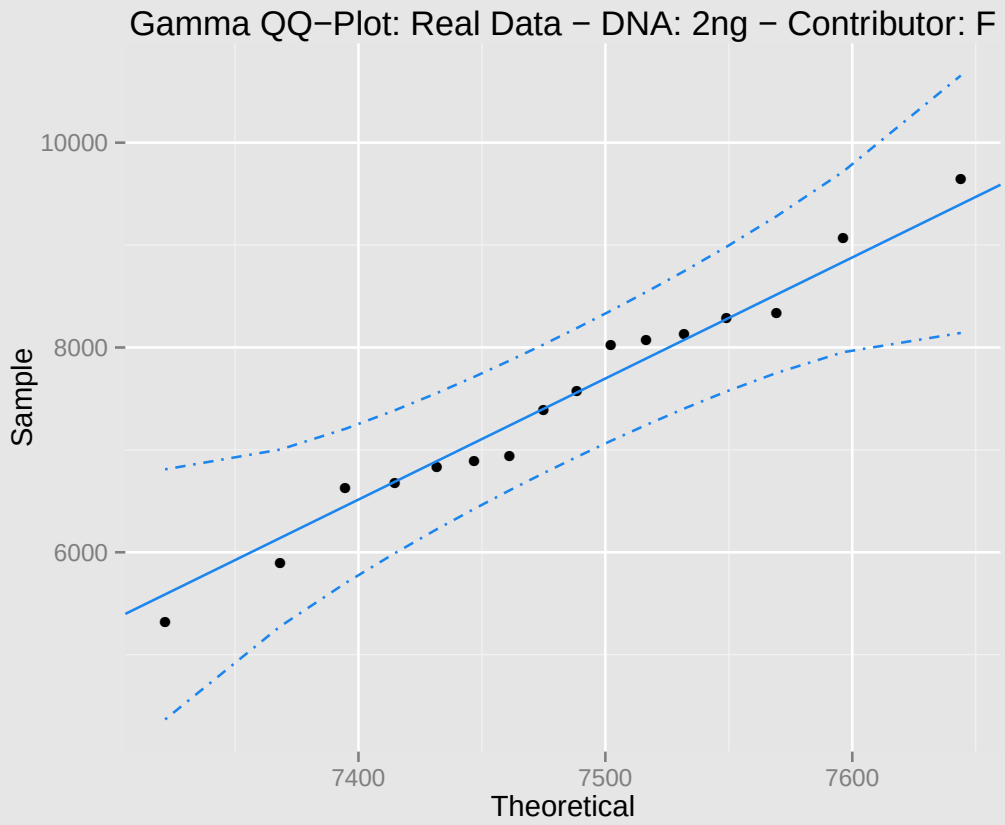


Figure 5.7: A Gamma QQ-plot of the coverage in sample `LT_Dil_002_F_2ng`. The dashed blue lines indicates a point-wise 95%-confidence envelope.

where G, is the cdf of a gamma distributed random variable. We have assumed that the alleles are independent and if we further assume that the loci are independent, then the likelihood is nothing but the product of $f(\boldsymbol{\varphi}_l; \alpha, \eta_l)$:

$$L_s(\alpha, \boldsymbol{\eta}; \varphi) = \prod_l f(\boldsymbol{\varphi}_l; \alpha, \eta_l). \tag{5.6}$$

Again we drop the subscript s, when it is clear from context. Note that if we have L loci this will yield $L+1$ parameters to be estimated, as $\boldsymbol{\eta} = (\eta_1, ..., \eta_L)^T$, contains L parameters, and we have a single α. For single contributor samples each locus will consist of zero to two alleles, some that needs to be imputed, for these loci the true value of η_l will be difficult to ascertain. Therefore, we will take a slightly different approach, we will maximise the profile likelihood. However, first a few notes on the the E-step in our particular application.

5.4.1 The E-Step

As the gamma distribution is an exponential family, we only need the sufficient statistics in the E-step of the algorithm. The sufficient statistics for the gamma-distribution are $\sum_i \log(y_i)$ and $\sum_i y_i$. The expectation of the sufficient statistics given α and η_l are found as follows:

$$\mathbb{E}\left[\sum_a \varphi_{la}|\alpha,\eta_l\right] = \sum_{a:\varphi_{la}\geq T} \varphi_{la} + \sum_{a:\varphi_{la}<T} \mathbb{E}\left[\varphi_{la}|\varphi_{la}<T,\alpha,\eta_l\right] \tag{5.7}$$

$$\mathbb{E}\left[\sum_a \log(\varphi_{la})\,|\alpha,\eta_l\right] = \sum_{a:\varphi_{la}\geq T} \log(\varphi_{la}) + \sum_{a:\varphi_{la}<T} \mathbb{E}\left[\log(\varphi_{la})\,|\varphi_{la}<T,\alpha,\eta_l\right] \tag{5.8}$$

The expected values, seen in Equations (5.7) and (5.8), can be further simplified, in accordance with [35]. We will start with Equation (5.7):

$$\begin{aligned}
\mathbb{E}\left[\varphi_{la}|\varphi_{la}<T,\alpha,\eta_l\right] &= \int_0^T \varphi_{la} g(\varphi_{la};\varphi_{la}<T,\alpha,\eta_l)d\varphi_{la} \\
&= \frac{\int_0^T \varphi_{la} g(\varphi_{la};\alpha,\eta_l)d\varphi_{la}}{G(T;\alpha,\eta_l)} \\
&= \frac{\int_0^T \frac{1}{\eta_l^\alpha\Gamma(\alpha)}\varphi_{la}^\alpha \exp\left(-\frac{\varphi_{la}}{\eta_l}\right)d\varphi_{la}}{\int_0^T \frac{1}{\eta_l^\alpha\Gamma(\alpha)}\varphi_{la}^{\alpha-1}\exp\left(-\frac{\varphi_{la}}{\eta_l}\right)d\varphi_{la}} \\
&= \eta_l \frac{\int_0^{T/\eta_l} u^\alpha \exp(-u)\,du}{\int_0^{T/\eta_l} u^{\alpha-1}\exp(-u)\,du} \\
&= \eta_l \frac{\gamma(\alpha+1,T/\eta_l)}{\gamma(\alpha,T/\eta_l)}.
\end{aligned} \tag{5.9}$$

The second to last equality holds by substituting $u = x/\eta_l$. Furthermore, γ denotes the lower incomplete gamma function, using the notation of [36]. A simplified version of Equation (5.8) can be found in a similar manner:

$$\begin{aligned}
\mathbb{E}\left[\log(\varphi_{la})\,|\varphi_{la}<T,\alpha,\eta_l\right] &= \int_0^T \log(\varphi_{la})\, g(\varphi_{la};\varphi_{la}<T,\alpha,\eta_l)d\varphi_{la} \\
&= \frac{\int_0^T \log(\varphi_{la})\,\varphi_{la}^{\alpha-1}\exp\left(-\frac{\varphi_{la}}{\eta_l}\right)d\varphi_{la}}{\int_0^T \varphi_{la}^{\alpha-1}\exp\left(-\frac{\varphi_{la}}{\eta_l}\right)d\varphi_{la}} \\
&= \frac{\int_0^{T/\eta_l} \log(u\eta_l)\,u^{\alpha-1}\exp(-u)\,du}{\gamma(\alpha,T/\eta_l)}
\end{aligned}$$

$$= \log(\eta_l) + \frac{1}{\gamma(\alpha, T/\eta_l)} \int_0^{T/\eta_l} \frac{\partial}{\partial \alpha} \left(u^{\alpha-1} \exp(-u) \right) du$$
$$= \log(\eta_l) + \frac{1}{\gamma(\alpha, T/\eta_l)} \frac{\partial}{\partial \alpha} \gamma(\alpha, T/\eta_l). \tag{5.10}$$

Implementation of the Lower Incomplete Gamma Function

Some care has be taken when implementing the incomplete gamma functions, as large values of either α or T/η_l, will correspond to very large values of the lower γ-function. In fact for α-values of 170 and above, γ yields a value larger than 10^{310} which in **R** results in an `Inf`-value (recall that η_l and thereby the upper limit T/η_l, in our case, will depend heavily on α). That is, for implementation purposes, we use the regularised lower incomplete gamma function and its derivatives (which quite conveniently is implemented in **R**):

$$P(\alpha, T/\eta_l) = \frac{1}{\Gamma(\alpha)} \gamma(\alpha, T/\eta_l). \tag{5.11}$$

The consequence being that the implementations of Equations (5.9) and (5.10), looks as follows:

$$\mathbb{E}\left[\varphi_{la} | \varphi_{la} < T, \alpha, \eta_l\right] = \eta_l \frac{\Gamma(\alpha+1)}{\Gamma(\alpha)} \frac{P(\alpha+1, T/\eta_l)}{P(\alpha, T/\eta_l)}$$
$$= \eta_l \alpha \frac{P(\alpha+1, T/\eta_l)}{P(\alpha, T/\eta_l)}$$

The last equality holds, as if the difference between $\Gamma(t+k)$ and $\Gamma(t)$ is an integer, $k \in \mathbb{Z}$, we can use the identity $\Gamma(t+1) = t\Gamma(t)$ recursively. We will exploit this fact even further, when we estimate the standard deviation. In order to implement Equation (5.10), first note that the derivative of Equation (5.11) w.r.t. α is:

$$\frac{\partial P(\alpha, T/\eta_l)}{\partial \alpha} = \frac{\partial}{\partial \alpha} \frac{\gamma(\alpha, T/\eta_l)}{\Gamma(\alpha)}$$
$$= \frac{1}{\Gamma(\alpha)} \frac{\partial \gamma(\alpha, T/\eta_l)}{\partial \alpha} - \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} P(\alpha, T/\eta_l) \tag{5.12}$$

Dividing Equation (5.12) by $P(\alpha, T/\eta_l)$ we have:

$$\frac{1}{P(\alpha, T/\eta_l)} \frac{\partial P(\alpha, T/\eta_l)}{\partial \alpha} = \frac{1}{\gamma(\alpha, T/\eta_l)} \frac{\partial \gamma(\alpha, T/\eta_l)}{\partial \alpha} - \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} \tag{5.13}$$

That is, using Equation (5.13) we can rewrite Equation (5.10) as follows:

$$\mathbb{E}\left[\log(\varphi_{la}) | \varphi_{la} < T, \alpha, \eta_l\right] = \log(\eta_l) + \frac{1}{P(\alpha, T/\eta_l)} \frac{\partial P(\alpha, T/\eta_l)}{\partial \alpha} + \frac{\Gamma'(\alpha)}{\Gamma(\alpha)}.$$

5.4.2 The M-Step

In the M-step, for the tth iteration of the algorithm, we could use Newton-Raphson to solve the following score equations, see [35]:

$$\begin{aligned} n_l \log\left(\eta_l^{(t+1)}\right) - n_l \frac{\Gamma'(\alpha^{(t+1)})}{\Gamma(\alpha^{(t+1)})} + \mathbb{E}\left[\sum_a \log\left(\varphi_{la}\right) | \alpha^{(t)}, \eta_l^{(t)}\right] = 0 \\ -\frac{n_l \alpha^{(t+1)}}{\eta_l^{(t+1)}} + \frac{1}{\left(\eta_l^{(t+1)}\right)^2} \mathbb{E}\left[\sum_a \varphi_{la} | \alpha^{(t)}, \eta_l^{(t)}\right] = 0 \end{aligned}$$

These equations can be found by simply differentiating log of Equation (5.6) for α and η_l, respectively. However, we will instead exploit that we know that the MLE of η_l, is given as:

$$\widehat{\eta}_l = \frac{\mathbb{E}\left[\sum_a \varphi_{la} | \alpha^{(t)}, \eta_l^{(t)}\right]}{n_l \alpha}. \tag{5.14}$$

That is, we can represent the log-likelihood of Equation (5.6) solely using the shape parameter, yielding the profile log-likelihood:

$$\begin{aligned} \ell_{\boldsymbol{\eta}}(\alpha; \varphi) = \sum_l -n_l \alpha \left\{ \log\left(\mathbb{E}\left[\sum_a \varphi_{la} | \alpha^{(t)}, \eta_l^{(t)}\right]\right) - \log\left(\alpha\right) - \log\left(n_l\right) + 1 \right\} \\ - n_l \log\left(\Gamma(\alpha)\right) + (\alpha - 1)\mathbb{E}\left[\sum_a \log\left(\varphi_{la}\right) | \alpha^{(t)}, \eta_l^{(t)}\right] \end{aligned} \tag{5.15}$$

We maximise $\ell_{\boldsymbol{\eta}}$ with respect to α and then update the scale parameter using Equation (5.14). Another approach would be to use a modification of the EM-algorithm called the Expectation-Conditional-Maximisation algorithm (ECM-algorithm), see e.g. [28, 37].

In order to maximise $\ell_{\boldsymbol{\eta}}$, we use the **`optim`**-function in **`R`**, specifically we will use the `BFGS`-method (the Broyden–Fletcher–Goldfarb–Shanno algorithm [38–41]). That is, we need the gradient of $\ell_{\boldsymbol{\eta}}$, which is given as:

$$\begin{aligned} \frac{\partial}{\partial \alpha} \ell_{\boldsymbol{\eta}} = \sum_l -n_l \left\{ \log\left(\mathbb{E}\left[\sum_a \varphi_{la} | \alpha^{(t)}, \eta_l^{(t)}\right]\right) - \log\left(\alpha\right) - \log\left(n_l\right) \right\} \\ - n_l \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} + \mathbb{E}\left[\sum_a \log\left(\varphi_{la}\right) | \alpha^{(t)}, \eta_l^{(t)}\right]. \end{aligned}$$

We have, in Appendix C, examined this implementation on simulated gamma distributed data. The implementation performs fairly well overall and we therefore continue onwards.

5.5 Estimating the Standard Deviation of the Heterozygote Balance

As we now have the imputation of the complete coverage in place, we turn our attention to its use in estimating the standard deviation of H_b. In general the sample variance of H_b is given as:

$$s^2(H_b) = \frac{1}{L}\sum_l \left(\frac{\varphi_{l,\text{HMW}}}{\varphi_{l,\text{LMW}}}\right)^2 - \left(\frac{1}{L}\sum_l \frac{\varphi_{l,\text{HMW}}}{\varphi_{l,\text{LMW}}}\right)^2. \tag{5.16}$$

An unbiased estimate can be obtained in the usual fashion, multiplying s^2 by $L/(L-1)$. We see that the sample variance not only depends on φ, but also $1/\varphi$, φ^2, and $1/\varphi^2$. In order to obtain a more accurate estimate we therefore compute the conditional expectation of these remaining variables, using the final parameter estimates of the EM-algorithm, $\boldsymbol{\eta}^*$ and α^*:

$$\mathbb{E}\left[\frac{1}{\varphi_{la}} \mid \alpha^*, \eta_l^*\right] = \eta_l^* \frac{\gamma(\alpha^* - 1, T/\eta_l^*)}{\gamma(\alpha^*, T/\eta_l^*)} \tag{5.17}$$

$$\mathbb{E}\left[\varphi_{la}^2 \mid \alpha^*, \eta_l^*\right] = \eta_l^* \frac{\gamma(\alpha^* + 2, T/\eta_l^*)}{\gamma(\alpha^*, T/\eta_l^*)} \tag{5.18}$$

$$\mathbb{E}\left[\frac{1}{\varphi_{la}^2} \mid \alpha^*, \eta_l^*\right] = \eta_l^* \frac{\gamma(\alpha^* - 2, T/\eta_l^*)}{\gamma(\alpha^*, T/\eta_l^*)} \tag{5.19}$$

These are found analogous to that of $\mathbb{E}\left[\varphi_{la}|\alpha^*, \eta_l^*\right]$, and are implemented in the same way, i.e. using the regularised lower incomplete gamma function. Define, using Equation (5.9), C_l^* as follows:

$$C_{la}^* = \begin{cases} C_{la}, & \text{if } C_{la} > T \\ \mathbb{E}\left[\varphi_{la}|\varphi_{la} < T, \alpha^*, \eta_l^*\right], & \text{else} \end{cases} \tag{5.20}$$

Furthermore, we define $1/C_{la}^*$, ${C_{la}^*}^2$, and $1/{C_{la}^*}^2$, using Equations (5.17), (5.18), and (5.19), respectively.

Given these expressions we can now also estimate $\text{sd}\,(H_b)$. However, before we continue, note that if both alleles on a locus are missing $H_b = 1$, as the imputed values are equal for all alleles on a give locus. As we have seen in Figure 5.3, H_b is centred around 1, i.e. imputing a value of 1 might result in underestimation of the standard deviation. Therefore, we examine the effect of the three types of removal, seen in List 5.5.1.

List 5.5.1

(i) **No removal**

(ii) **Outer removal**: Dropped loci are removed before coverage imputation. That is, they will not have any influence on the parameter estimates of $\boldsymbol{\eta}$ and α.

(iii) **Inner removal**: Dropped loci are used in the imputation of coverage, but removed when estimating the standard deviation. That is, the they will have influence on the parameter estimates of $\boldsymbol{\eta}$ and α, but not directly on the standard deviation.

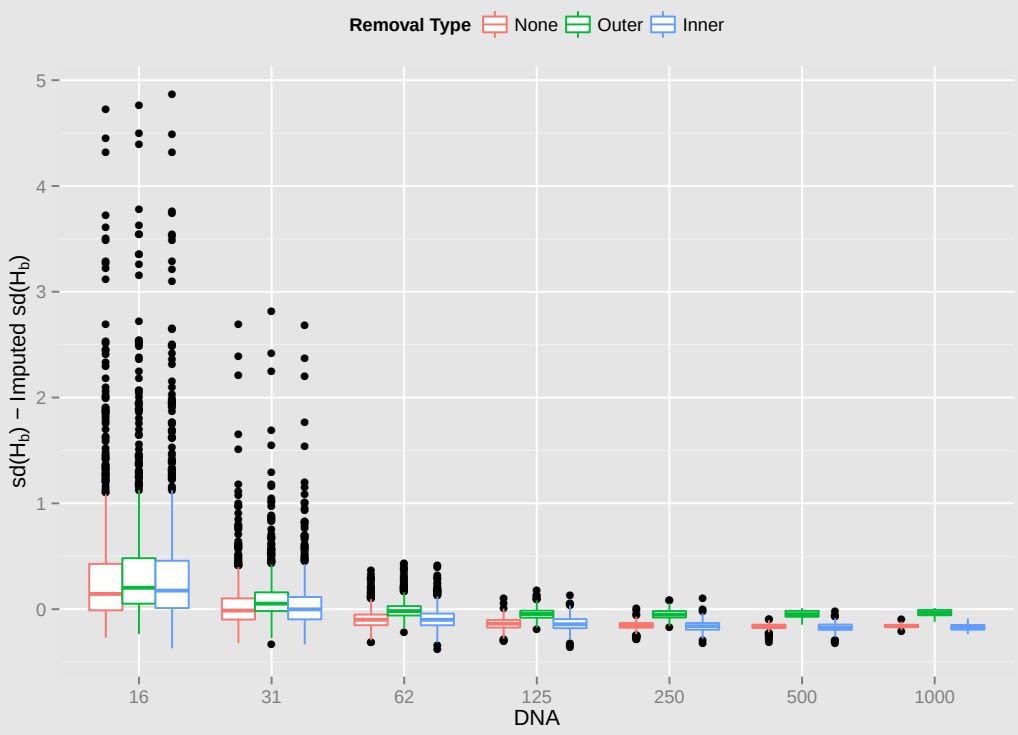


Figure 5.8: The estimation error of sd (H_b), for outer, inner, and no removal of dropped loci.

Figure 5.8 shows the estimation error, for each dilution of our simulated data, and all three removal types used (no-removal included). Furthermore, we only use the replicates where a locus drop out is observed, as the three methods would be equal on

every other replication. The figure shows that all three methods underestimates the standard deviation for small amounts of DNA. However, the outer removal method gets very close to true standard deviation as the amount of DNA increases. Hence, we will choose the outer removal method.

The fact that all three methods underestimate the standard deviation for smaller amounts of DNA is not surprising, as both the number of drop-outs and the standard deviation increases when the amount of DNA decreases. The fact that our method underestimates the standard deviation is further supported by the Bland-Altman inspired plot (difference versus average plot), seen in Figure 5.9. The plot includes only values imputed using the outer removal and shows a clear increasing trend.

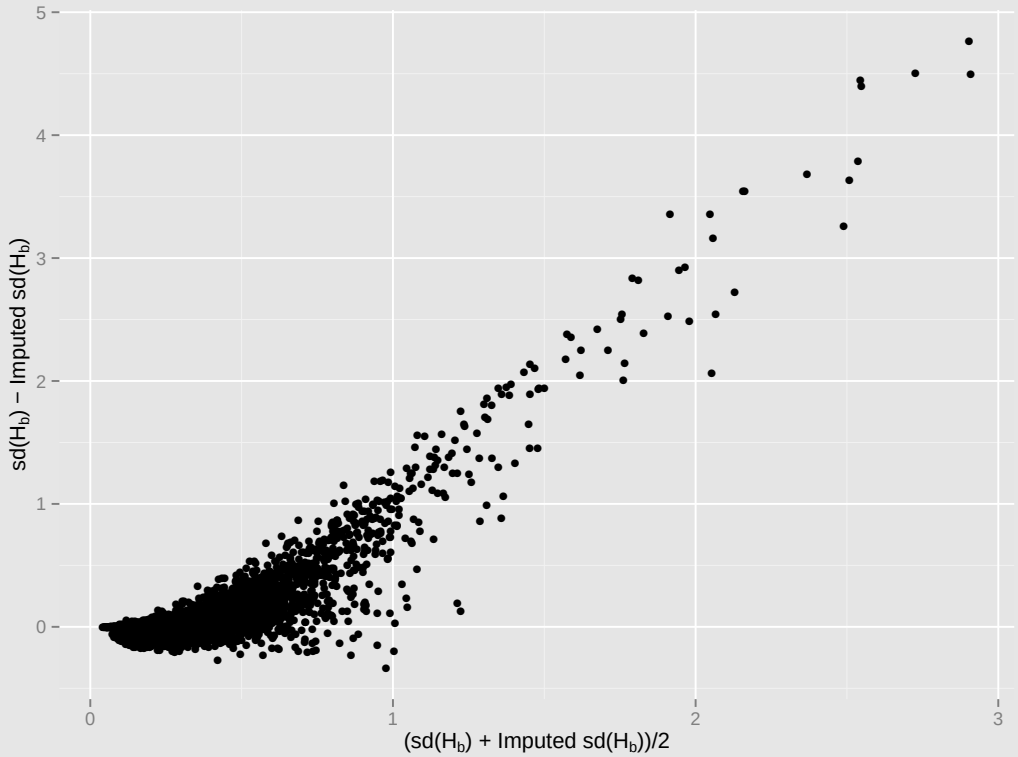


Figure 5.9: A Bland-Altman plot (the difference plotted against the average) of the true and imputed standard deviation.

5.6 Estimating Probability of Drop-out

In order to model $\mathbb{P}(D)$ we will use logistic regression. That is, given the estimated standard deviation, $\mathbb{P}(D)$ is logit-linear:

$$\text{logit}\left(\mathbb{P}(D)\right) = \beta_0 + \beta_1 \log\left(\widehat{\text{sd}}\left(H_b\right)\right), \tag{5.21}$$

Before we fit the model note that the Bland-Altman plot, Figure 5.9, further shows that the order of underestimation increases as the standard deviation increases. The consequence is, as seen in Figure 5.10, that log of the estimated standard deviation has a steep incline (steeper than the true standard deviation, as evident by panel (b) showing a violin plot). That is, using a log-transformation of the standard deviation might be suboptimal when estimating the probability of drop-out.

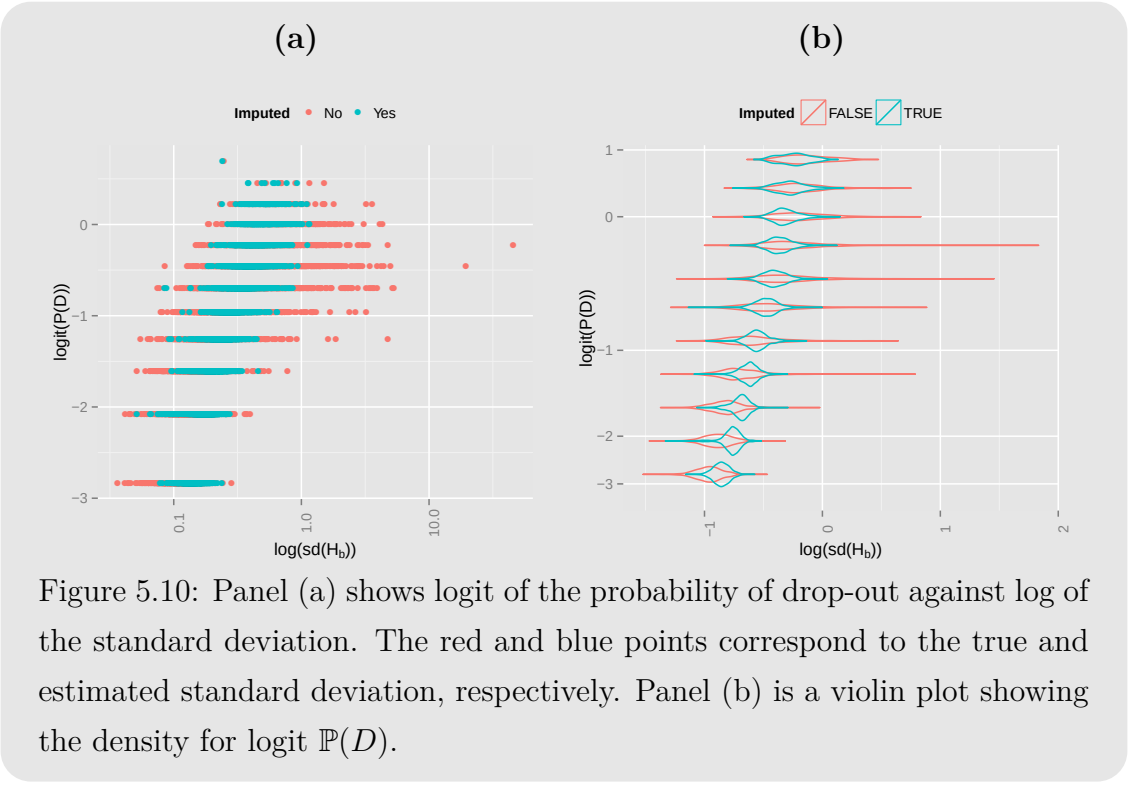


Figure 5.10: Panel (a) shows logit of the probability of drop-out against log of the standard deviation. The red and blue points correspond to the true and estimated standard deviation, respectively. Panel (b) is a violin plot showing the density for logit $\mathbb{P}(D)$.

Therefore, we would like to choose a more appropriate transformation, if such a transformation exists, and in order to do so we will examine Box-Cox transformations [42], of $\widehat{\text{sd}}\left(H_b\right)$. Given λ a Box-Cox transformation is defined as:

$$y_i^{(\lambda)} = \begin{cases} \dfrac{y_i^{\lambda} - 1}{\lambda}, & \text{if } \lambda \neq 0 \\ \log\left(y_i\right), & \text{if } \lambda = 0 \end{cases} \tag{5.22}$$

The parameter λ is estimated using the profile likelihood function, once the MLE of λ is found we choose an appropriate value within (or close to) the 95% confidence interval (CI). The CI is created using the fact that two times the log-likelihood ratio asymptotically follows a χ^2 distribution with one degree of freedom. The Box-Cox transformations can be extended to a two parameter version which includes an offset. Furthermore, as seen in our definition of the Box-Cox transformation, Equation (5.22), the transformation is normally performed on the response variable adjusting for any possible covariates. However, we will use it for our covariate and adjust using the response. Note that from this point forth, any $\mathbb{P}(D) = 0$ has been excluded when fitting $\mathbb{P}(D)$, as they would not occur in reality.

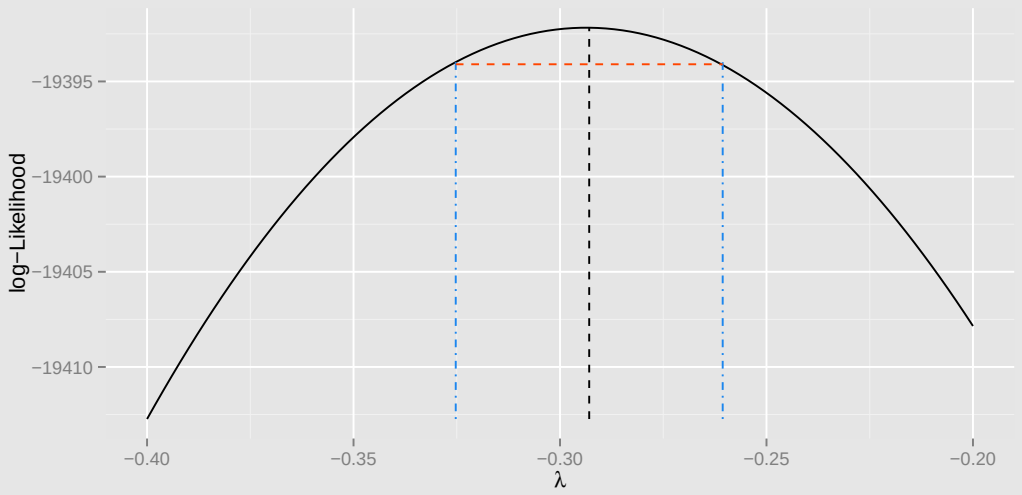


Figure 5.11: .The profile log-likelihood of Box-Cox transformed standard deviation adjusted with the probability of drop-out. The black, blue, and red dashed lines indicate the MLE, lower/upper bound of the 95%-CI, and the range of the 95%-CI, respectively.

The profile likelihood of the Box-Cox transformed $\widehat{\text{sd}}\,(H_b)$ is seen in Figure 5.13. We have adjusted the profile likelihood using logit of $\mathbb{P}(D)$. The MLE (the black dashed line) is -0.293, and the lower/upper bound of the 95% CI (the blue dashed lines) are -0.325 and -0.261, respectively (all rounded to 3 digits). There does not seem to be any *appropriate* values within the CI, we therefore venture slightly outside and choose $\lambda = -1/3$.

Using the value $\lambda = -1/3$, we will fit a logistic model to $\mathbb{P}(D)$ using $\widehat{\text{sd}}\,(H_b)^{(-1/3)}$ as a covariate. The superscript encased in parentheses indicates that the variable has undergone a Box-Cox transformation with the given value, in accordance with

Equation (5.22).

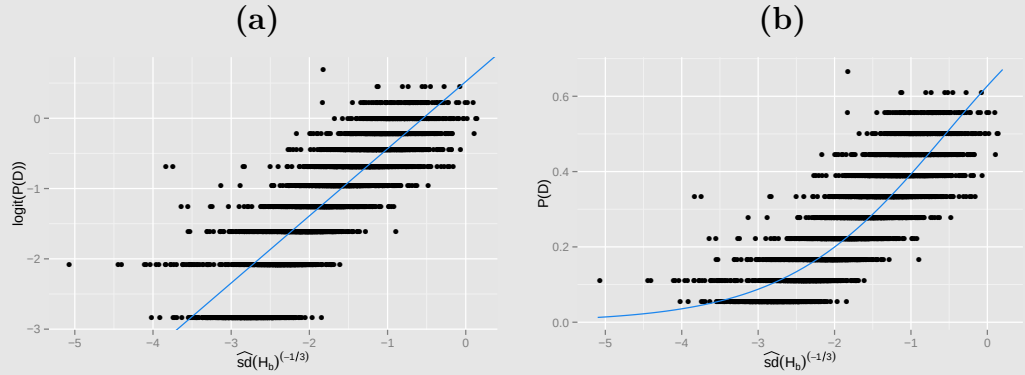


Figure 5.12: Panel (a) shows the logit transformed probability of drop-out against the Box-Cox transformed imputed standard deviation of the heterozygote balance. The blue line is the regression line with respect to the estimated parameters $\widehat{\beta}_0 \approx 0.5216$ and $\widehat{\beta}_1 \approx 0.9551$. Panel (b) is nothing but panel (a) transformed using the inverse link function (the logistic function).

In order to estimate the parameters in Equation (5.21), we fit a generalised linear model (GLM) with a binomial family using logit as a link-function. The parameters are estimated as $\widehat{\beta}_0 \approx 0.5216$ and $\widehat{\beta}_1 \approx 0.9551$. We have, in Figure 5.12, plotted $\text{logit}\left(\mathbb{P}(D)\right)$ against $\widehat{\text{sd}}\left(H_b\right)^{(-1/3)}$. The blue line is the regression line with respect to $\widehat{\beta}_0$ and $\widehat{\beta}_1$. The mean square error (MSE) of the fitted logistic regression model is ≈ 0.1817. Note that the MSE is, when the response variable is dichotomous, sometimes referred to as the Brier score.

We are, to be more precise, trying to fit a sigmoid curve and logistic regression is not the only way to achieve such a noble goal. Therefore, in order to see if the fit can be improved, we will also try a couple of the other link-functions available for the binomial family, namely the probit and the complementary log-log (cloglog) functions, as well as, a five-parameter logistic (fpl) regression model [43]. The logit, probit, and cloglog-functions all have the same disadvantage, that the curve is always symmetric, a disadvantage the fpl-model does not share.

The fpl-model transforms the covariates using the following non-linear function:

$$\text{fpl}(\boldsymbol{X}, \boldsymbol{p}) = p_1 + \frac{p_2 - p_1}{1 + f(\boldsymbol{X})\exp\left(p_3(p_4 - \boldsymbol{X})\right) + (1 - f(\boldsymbol{X}))\exp\left(p_5(p_4 - \boldsymbol{X})\right)} \tag{5.23}$$

where f and C are given as follows:

$$f(\boldsymbol{X}) = \frac{1}{1 + \exp\left(-C(P_4 - \boldsymbol{X})\right)}$$
$$C = \frac{2P_3P_5}{|P_3 + P_5|},$$

The parameters $\boldsymbol{p}$ are interpreted as follows:

- p_1 and p_2 controls the lower and upper asymptote, respectively.
- p_3 and p_5 controls the curvature below and the above the infliction point, respectively.
- p_4 controls the position of the infliction point.

The fpl-model, Equation (5.23), simplifies in our case as we are dealing with probabilities, i.e. $p_1 = 0$ and $p_2 = 1$. Furthermore, it is easy to see that the curve is symmetric if $p_3 = p_5$ (for more information see e.g. [44, 45]). In order to fit the model we will use the `nls`-function in **R**.

We will compare the four models using cross-validation (CV) in order to examine their predictive properties. Figure 5.13 shows the $\mathbb{P}(D)$ against $\widehat{\mathrm{sd}}\,(H_b)^{(-1/3)}$ for all four models fitted on the training set. We will compare the area under the curve (AUC) and Brier score for each model. Note that we have used $\widehat{\mathrm{sd}}\,(H_b)^{(-1/3)}$ as a covariate in all four models. We see that the only real difference between the glm based models and the fpl-model, is that the fpl-model bends toward the upper asymptote very early, comparatively.

The Brier score and AUC for both the training and validation sets, are shown in Table 5.1, as well as the BIC for each model. We see that the BIC for the fpl-model is around 3000 larger than the BICs w.r.t. the glm based models. Focusing only on the glm-based models, we see that the probit-model has the lowest BIC and Brier score on the validation set, with the logit-model close behind. We see that the fpl-model achieves the smallest Brier score on the validation set and would therefore be our model of choice, though it would seem, based on the BIC, that it comes at the cost of added complexity.

The Brier score, of the fpl-model, on the test set is ≈ 0.1816. We note that the AUC is very consistent across all four models, given the data, and we see why in the Brier score, the models predictive properties are virtually identical, making the

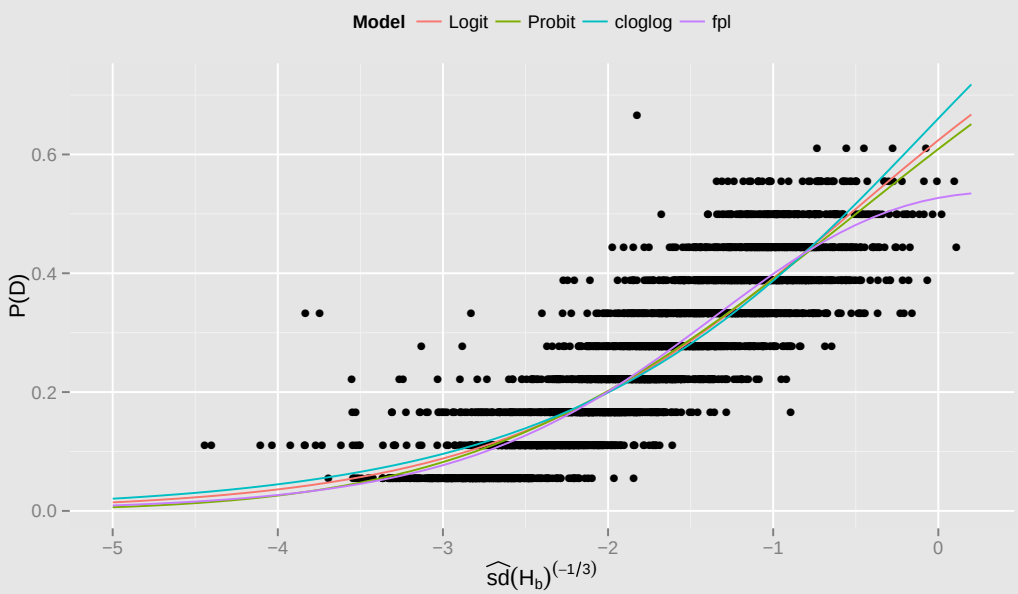


Figure 5.13: The probability of drop-out plotted against the Box-Cox transformed estimated standard deviation, from the training set. The curved lines represent the four models discussed in this section, fitted on the training set.

added complexity introduced by the fpl-model very unnecessary. We will therefore choose the regular logistic regression model.

Table 5.1: The training and validation errors of all four models discussed in this section. The tinted cells show the smallest validation errors.

		Training Set		Validation Set	
Model	BIC	Brier Score	AUC	Brier Score	AUC
logit	79,072	0.1813	0.6708	0.1827	0.6687
probit	79,056	0.1812	0.6708	0.1826	0.6687
cloglog	79,152	0.1815	0.6708	0.1829	0.6687
fpl	82,319	0.1809	0.6708	0.1824	0.6687

The test Brier score of the logit-model is ≈ 0.1819, i.e. the difference in test error, between the logit and fpl model, is less than 1%. The AUC of the logit-model based on the test set is approximately 0.6712, and the corresponding ROC curve can be seen in Figure 5.14.

An AUC of 0.6712 is not considered good, though it is better than random guessing. If we choose a classification rule of 0.5, then the percentage of correct classifications is 73.38%, which is close to one minus the percentage of drop-outs in the data, at 73.47%.

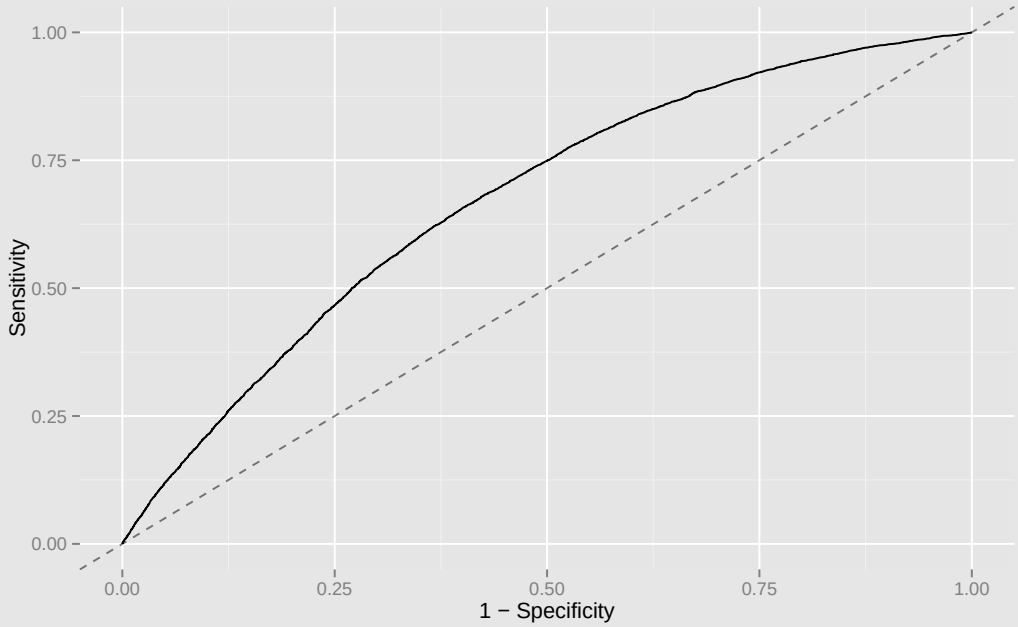


Figure 5.14: ROC curve, based on the test set, for the logit-model. The dashed line is an indication of guessing randomly. The AUC is 0.6712.

There are two things left noting. First, the coefficients $\boldsymbol{\beta}$, are here not dependent on locus, which works with the simulated data, but for the real data the coefficients would most likely be locus dependent. Second, we know from Figure 5.4, that the standard deviation decreases when the amount DNA increases, i.e. using the method, described in Sections 5.3-5.5 imputing the standard deviation, we are able to estimate the amount of DNA, which may be useful when handling mixtures.

CHAPTER

SIX

EPILOGUE

This chapter will start with a short recap of the thesis, the aim, problems, and possible solutions, presented, as well as how these fit together. We then comment on a few of the presented methods and a possible extension of the LUS. We end the chapter listing and describing potential future work.

6.1 Recap

The general aim of the thesis has been to investigate the statistical variation of short tandem repeat (STR) next-generation sequencing (NGS) data in a forensic genetics framework. The variation is examined as a prelude to future work evaluating the strength of evidence when comparing two hypothesis, $LR = \mathbb{P}(\mathcal{E}|\mathcal{H}_p)/\mathbb{P}(\mathcal{E}|\mathcal{H}_d)$, or deconvoluting a DNA profile. We introduce a method for extracting the STR-regions from the NGS data, using directly adjacent flanking regions. If the sample is known to contain only one contributor and is otherwise uncontaminated creating a profile is simple. We show that the DNA profile can be obtained using a heterozygote threshold, which determines whether a locus is homo- or heterozygous, based on the most prevalent string on the locus. The threshold was calibrated such that the number of drop-outs is minimised given that the number of drop-ins is equal to zero. This method is, however, only apt in this specific case. That is, to find a more

general estimate of $\mathbb{P}(\mathcal{E}|\mathcal{H})$, we needed to thoroughly inspect the noise, systematic or otherwise. However, before doing so, we took a look at the string quality generated, when using NGS.

The quality of a called base is an indication of the probability of error associated with the call. That is, if the quality is low we expect that the probability of the base being called erroneously is high. We show that the quality of our locus identified reads is fairly stable, as a consequence of the way we have identified them, and conclude that further restriction of the reads is therefore unnecessary. Furthermore, we have introduced a method of assigning probability to two strings of similar length being equal, based on the quality and the bases in which the strings mismatch. This method may prove useful if multiple strings of similar length slipped through a noise threshold.

Stutters and shoulders is some of the more systematic noise encountered in the data. Stutters is an old favourite generated by the PCR amplification process, while shoulders seems to a product of NGS. It has been shown previously that the stutter ratio, $\varphi_{\text{Stutter}}/\varphi_{\text{Parent}}$, is highly dependent on the length of the parent allele. Furthermore, it has been assumed, and somewhat verified, that the longest uninterrupted stretch (LUS) was a better predictor, than the allele length. A hypothesis for which we have now provided extra evidence. We have fitted simple linear models, gaussian mixture models, and gamma models of the stutter ratio versus both allele length and LUS. These models have clearly shown that, the LUS out performs allele length, for loci with non-simple repeat patterns. Furthermore, we have determined that the shoulders are fairly stable versus the allele length and the creations of a threshold to subsequently remove or identify potential shoulders is set at the median plus three times the standard deviation (the threshold is shown to be highly locus dependent).

We have seen that STR NGS data, because of the miscalled bases and insertions/deletion (indels), suffers from plenty of strings with very little coverage. In particular we see a lot of strings occurring only once in the data. We have therefore considered a negative binomial (NB) model, as the coverage of a string can be seen as over dispersed count data, in order to model this sort of general noise. We have observed that the data suffers from both one-inflation and zero-truncation and have because of this fact modified the NB model to account for this kind of data. Therefore, we have generalised the concept as a k-inflated negative binomial (KINB) model, and introduced two approaches to estimating the parameters of such

a model. Using quantiles of the KINB model, we have created a sample and locus specific threshold differentiate the general noise from alleles and more systematic noise (such as stutters and shoulders).

With the introduction of a noise threshold, we inadvertently create drop-outs in our data. In order to examine the probability of such a drop-out occurring, we investigate the imbalance of the alleles on a heterozygous locus. We simulate allele coverage, by first simulating PCR amplification using binomial sampling and then add what we call chip sampling, or MID sampling, to represent the NGS process. We observed that we cannot use the coverage as a predictor for the probability of drop-out, however, that the standard deviation, of the heterozygote balance (defined as the coverage of the high molecular weight allele over the coverage of the low molecular weight allele), increases with the probability of drop-out. However, it only holds if no threshold is imposed. That is, we need the complete coverage, which we would not. Therefore, we turned to the expectation-maximisation (EM) algorithm to impute the standard deviation of the complete coverage. The consequence being that the distribution of the complete coverage is needed. We have chosen the gamma distribution, inspired by the work which has been done in capillary electrophoresis. The imputed standard deviation was then used as the covariate in a logistic regression model, estimating the probability of drop-out.

6.2 Comments

6.2.1 String Coverage with Artefacts

The following is heavily inspired by [34]. First note that we can decompose the coverage of a string, φ_{la}, directly after PCR, as $\varphi_{la} = \varphi_{la}^{(0)} + \varphi_{la}^{(s)}$. Assume that $\varphi_{la}^{(0)}$ and $\varphi_{la}^{(s)}$ are independent, then:

$$\varphi_{la}^{(0)} \sim \Gamma(\alpha(1-\xi_a), \eta_l), \quad \varphi_{la}^{(s)} \sim \Gamma(\alpha\xi_a, \eta_l),$$

where ξ_a is the stutter ratio of allele a given the LUS of a. It therefore follows that, if the string allele a and the string of the stutter belonging to $a+1$, then:

$$\varphi_{la} = \varphi_{la}^{(0)} + \varphi_{l(a+1)}^{(s)}.$$

This only includes the stutter of $(a+1)$, however, one could also consider the double stutter of $(a+2)$, triple stutter of $(a+3)$ et cetera. As we have assumed the alleles are independent, φ_{la} is still gamma distributed:

$$\varphi_{la} \sim \Gamma\left(\alpha\{(1-\xi_a)+\xi_{(a+1)}\}, \eta_l\right).$$

Note that when calculating the probability of the evidence given a hypothesis, genotypes are provided. It is customary in CE to multiply the shape by n_{la}, which indicates the number of alleles of type a on locus l, i.e. $n_{la} \in \{0,1,2\}$ (n_{la} is related to n_l seen in Chapter 5, as $n_l = \sum_a n_{la}$), and we will use n_{la} in a similar manner, by defining $g_{la} = \mathbb{I}[n_{la} > 0]$. Using the probability of drop-out, found in Chapter 5, we can write the conditional likelihood of C_{la} as:

$$L_{la}(\alpha, \eta_l, \boldsymbol{\xi}; C_{la}, \boldsymbol{g}) = \begin{cases} g(C_{la}; \alpha\{(1-\xi_a)g_{la} + \xi_{(a+1)}g_{l(a+1)}\}, \eta_l), & \text{if } C_{la} \geq T \\ \dfrac{1}{1+\exp\left(\beta_{0,l} + \beta_{1,l}\widehat{\mathrm{sd}}(H_b)^{(-1/3)}\right)}, & \text{otherwise} \end{cases}$$

The probability of the evidence conditioned on a given hypothesis $\mathcal{H}$ is then:

$$L(\mathcal{H}) = \mathbb{P}(\mathcal{E}|\mathcal{H}) = \sum_{\boldsymbol{g}} \left(\prod_l \prod_a L_{la}(\alpha, \eta_l, \boldsymbol{\xi}; C_{la}, \boldsymbol{g}) \right) \mathbb{P}(\boldsymbol{g}|\mathcal{H}).$$

This is a very short overview of how the artefacts discussed in the thesis, could be weaved together in a fashion nearly identical, to what has been done in CE, [32–34].

6.2.2 Expanding the LUS

The stutter ratio calculated in Chapter 3, is found using the sum of the coverage for every *true* stutter of a given allele. If our allele sequence takes the form:

$$[\mathrm{AATG}]_{10}[\mathrm{GTTA}]_4[\mathrm{AATG}]_2, \tag{6.1}$$

some of the possible stutter variations include:

(i) $[\mathrm{AATG}]_9[\mathrm{GTTA}]_4[\mathrm{AATG}]_2$

(ii) $[\mathrm{AATG}]_{10}[\mathrm{GTTA}]_3[\mathrm{AATG}]_2$

(iii) $[\mathrm{AATG}]_{10}[\mathrm{GTTA}]_4[\mathrm{AATG}]$,

and most likely in that order of prevalence. We are not guaranteed that our sequence will stutter so *nicely*. In reality we hypothesise that any tetra-nucleotide within the allele sequence can stutter. Given our allele sequence $[\text{AATG}]_{10}[\text{GTTA}]_4[\text{AATG}]_2$, it could stutter at AATG, ATGA, TGAA, GAAT, GGTT etc.

Until now these have all been attributed to the same LUS (i.e. 10 in this case). However, given that the stuttering unit might not come from the LUS, we would like to, instead of lumping all the observed variations together, consider them individually. That is, given a sequence $\mathcal{A}$ we do the following:

- We use the **`pairwiseAlignment`**-function in order aligning all strings four bases shorter than $\mathcal{A}$, to $\mathcal{A}$. We use the function in such a way that all strings missing exactly four consecutive bases achieves a unique score. We classify these as *true* stutters of $\mathcal{A}$.
- Assuming that we have m *true* stutters $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m$, we calculate the stutter ratios for all m stutters all w.r.t. the coverage of $\mathcal{A}$.
- We find the actual missing unit of $\mathcal{S}_i$, in the parent $\mathcal{A}$, for every $i = 1, ..., m$, and then its repeat unit length (RUL) within the parent.

The RUL is nothing but a finer version of the LUS. If the allele in Equation (6.1) looses e.g. unit AATG its observed parent RUL is either 10 or 2, depending on where the missing unit is found, while the LUS is always 10. The hypothesis is, as in the case of the LUS, that the stutter ratio will increase as the RUL increases.

The score created by **`pairwiseAlignment`** depends on the number of matches, if the function has to open a gap (`gapOpening`), and if the function has to extend a gap (`gapExtension`). The score increases by one if two bases match and decreases every a gap has to either opened or extended (there should be a penalty on both). We use a penalty of six for opening a gap and 1 for extending a gap, i.e. the score of a true stutter $\mathcal{S}_i$ equals the number of bases in $\mathcal{S}_i$ minus ten. Note that in order to avoid mismatched bases, we set the penalty for mismatching extremely high.

In Table 6.1, we see the RUL tabulated against the LUS, the tinted cells indicate that the two are equal (i.e. the diagonal of the table). We see that a lot of the missing units actually has an RUL of 1, or an RUL in the 6-9 range. Figure 6.2, shows heatmaps of the corresponding tabulations for every locus. That is, the

heatmap under facet D12 corresponds to Table 6.1. We see that for most of our loci the RUL will correspond to the LUS. This could indicate a simple repeat STR structure. However, looking at D3 or D21, we see a distinct circular pattern above the diagonal.

Table 6.1: The RUL against the LUS of locus D12. The tinted boxes indicates RUL equal to LUS.

		LUS										
		7	8	9	10	11	12	13	14	15	16	17
RUL	1	2	1	28	25	86	151	128	99	45	20	2
	2	0	0	0	0	2	1	0	0	1	0	0
	3	0	0	0	1	2	0	0	0	0	0	0
	4	0	1	0	0	0	0	0	0	1	0	0
	5	0	10	0	1	3	29	15	8	0	0	0
	6	6	13	35	20	92	92	58	24	4	1	0
	7	5	1	15	29	14	27	29	22	1	1	0
	8	0	14	0	0	7	6	14	20	22	4	2
	9	0	0	50	4	4	21	28	28	26	14	1
	10	0	0	0	54	2	3	4	7	2	0	0
	11	0	0	0	0	122	1	2	1	0	0	0
	12	0	0	0	0	0	179	0	3	0	0	0
	13	0	0	0	0	0	0	150	1	0	0	0
	14	0	0	0	0	0	0	0	114	0	0	0
	15	0	0	0	0	0	0	0	0	56	0	0
	16	0	0	0	0	0	0	0	0	0	20	0
	17	0	0	0	0	0	0	0	0	0	0	3

Figure 6.1 shows the stutter ratio plotted against observed parent RUL. We see that, for loci containing shorter RUL, the relationship still seems linear. However, looking at D3 and D12 we see more ambiguous, with a lot of outliers, and for vWA and D21, we see an enormous variation for long RUL's. However, Figure 6.1 shows, that the stutter ratio increases as the RUL increases, as hoped though it seems that the relationship between RUL and stutter ratio is not linear as in the case of the LUS.

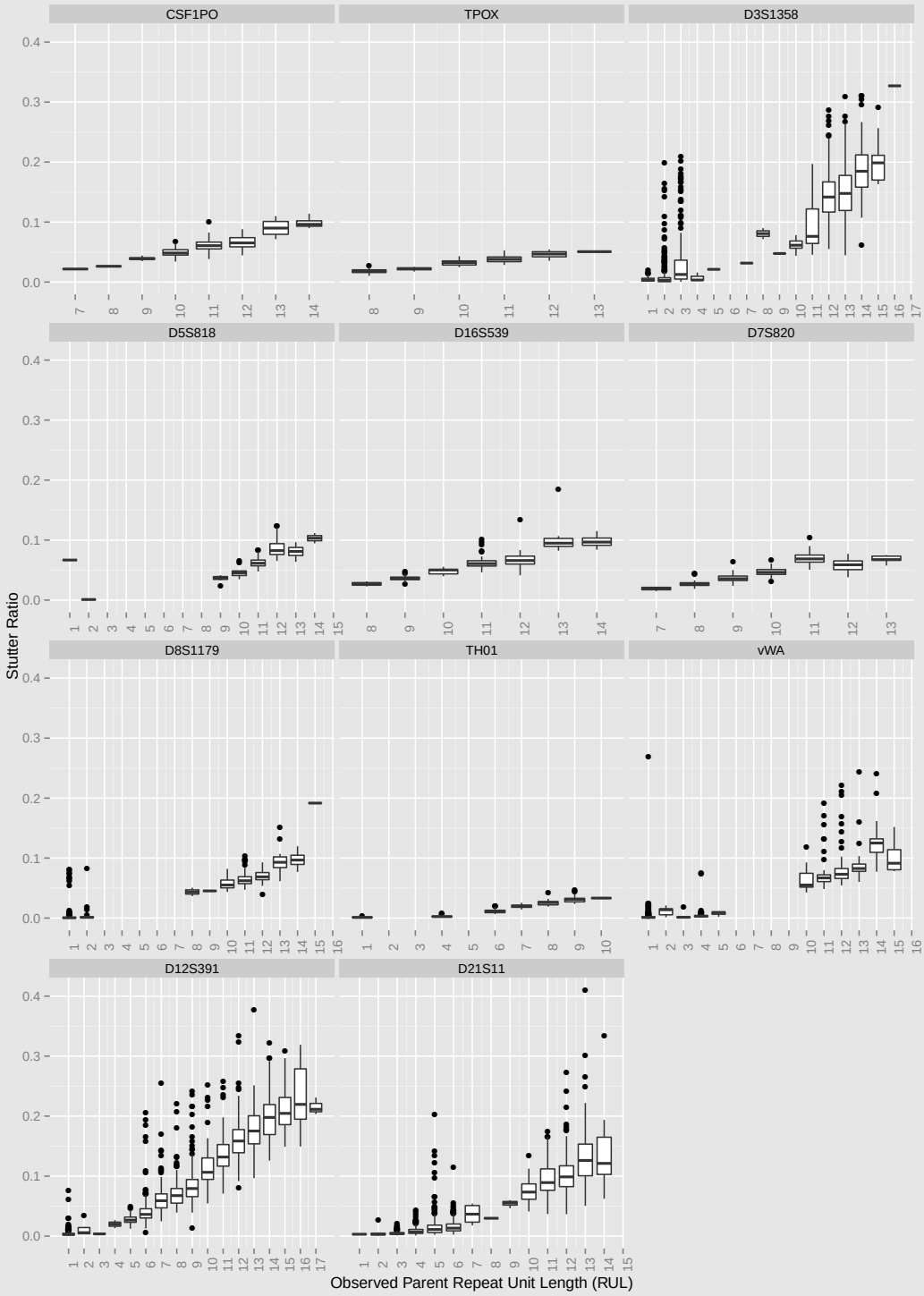


Figure 6.1: Boxplots of stutter ratio against the observed parent RUL, for every loci in the IonTorrent and Roche reference files.

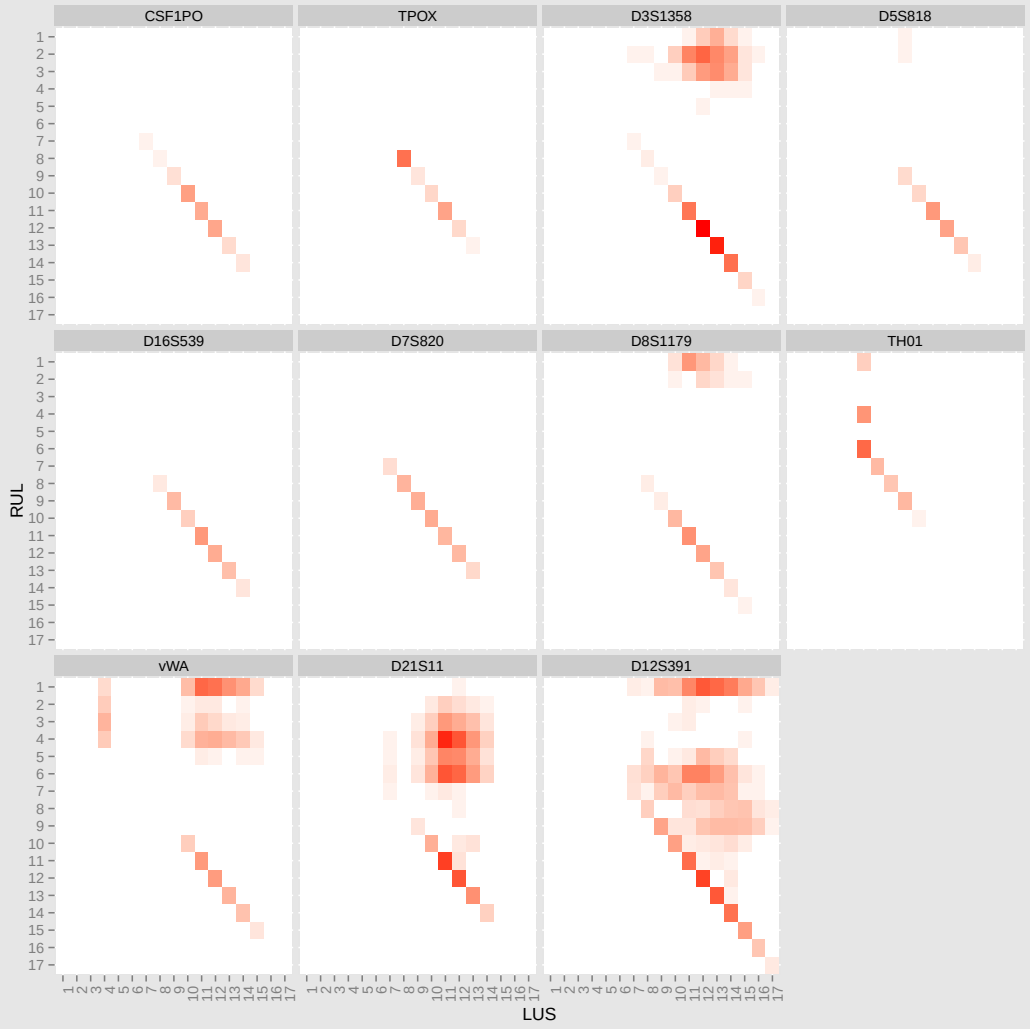


Figure 6.2: Heatmaps corresponding to tabulations of RUL against LUS, for every locus.

6.2.3 Using Profile-likelihood in the EM-algorithm

As the profile likelihood is not a true likelihood function, the choice of using a profile likelihood in the maximisation step of the EM-algorithm, is unconventional. The EM-algorithm and profile likelihood are no strangers, the profile likelihood can be used in the creation of standard errors and confidence intervals, for the parameter estimates, yet has not been used for estimating the parameters. We would have to show that the profile likelihood is concave, for a given iteration of the algorithm, ensuring a local maximum is found when maximised, thereby yielding a proper EM-algorithm.

Another approach to estimating the parameters of the gamma distribution, as mentioned in Section 5.4.2 and generally introduced in [37], is to substitute the maximising step, with a conditional-maximisation step. That is, assume we after t iterations have parameter estimates, $\boldsymbol{\theta}^{(t)}$, then in the $(t+1)$th iteration the parameter $\theta_i^{(t+1)}$ is estimated, given the parameters $\boldsymbol{\theta}_{-i}^{(t)}$.

6.2.4 The KINB Methods

The implementation of the KINB methods, both the maximum likelihood (ML) approach (which we through-out the thesis refer to as the **`kinb`**-method) and the weighted fractile method, could use some work, particularly the weighted fractile method. The only real problem with the ML approach is the drop-ins occurring on the CSPOF1 locus, the solution could be as simple as adjusting the threshold to compensate, for this particular locus, and maybe provide both adjusted and non-adjusted classifiers.

The problem with the weighted fractile function on the other hand, is that the estimate of the mean value parameter, λ, very poor when the size parameter is close to zero, as seen in Appendix B. However, looking at the density plot in Figure 6.3, we see that the density of the negative binomial distribution using both true and estimated parameters, of the simulated data from Appendix B, looks virtually identical.

So why does the shape of the density seem that closely related? In order to understand why, we take a closer look at negative binomial distribution. In Section 4.1, we introduce the pmf of the negative binomial distribution by its mean representation, it is however more generally defined as follows:

$$f(k;p,\theta) = \binom{k-\theta+1}{k} p^k (1-p)^\theta,$$

where p is the probability of success and θ is the number of failures until the experiment is stopped. The mean λ is given as $p\theta/(1-p)$, which implies that $p = \lambda/(\lambda+\theta)$. It follows, that when the true size parameter is close to zero, or when the true mean is much larger than the size, the probability of success is close to 1. This is an important observation, as a higher probability of success would yield an increase in the number of extreme observations, and when modelling the noise we see a lot of extreme values (alleles, stutters, and shoulders).

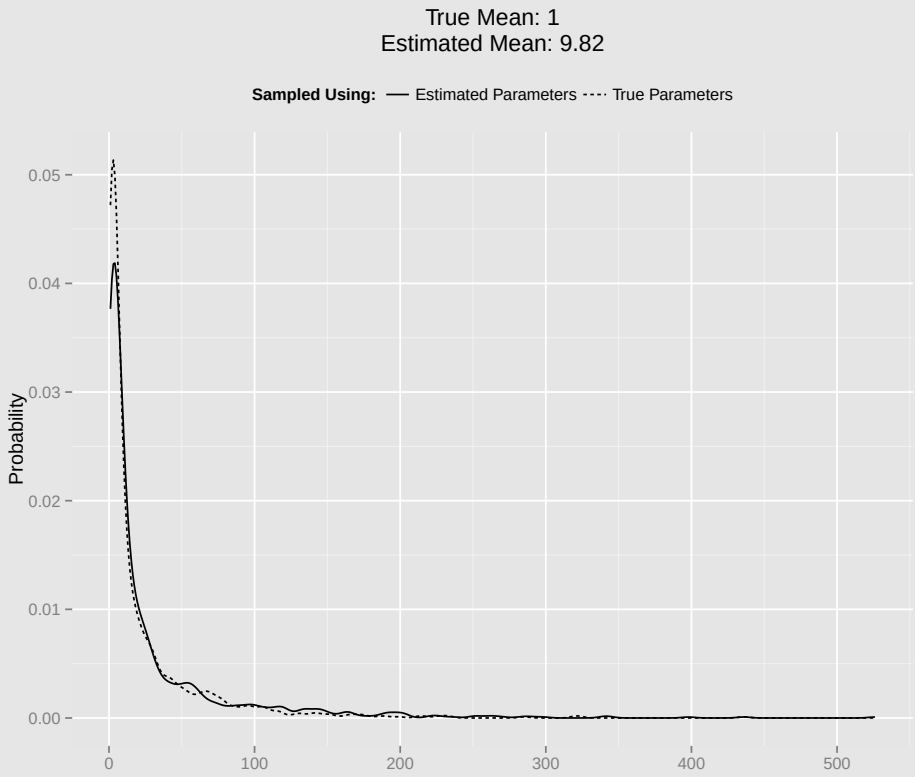


Figure 6.3: The density of the negative binomial distribution, using both the true parameters and parameters estimated using the weighted fractile method.

In general the weighted fractile method has a tendency to overestimate the size parameter (the median is approximately 70 times the true size value), therefore in order to ascertain a higher probability of success, the the weighted fractile function tries to compensate by increasing the estimated mean value. Figure 6.4, shows that the relationship between the true and the estimated mean value is fairly linear. An observation which we might be able to use, to adjust the weighted fractile method.

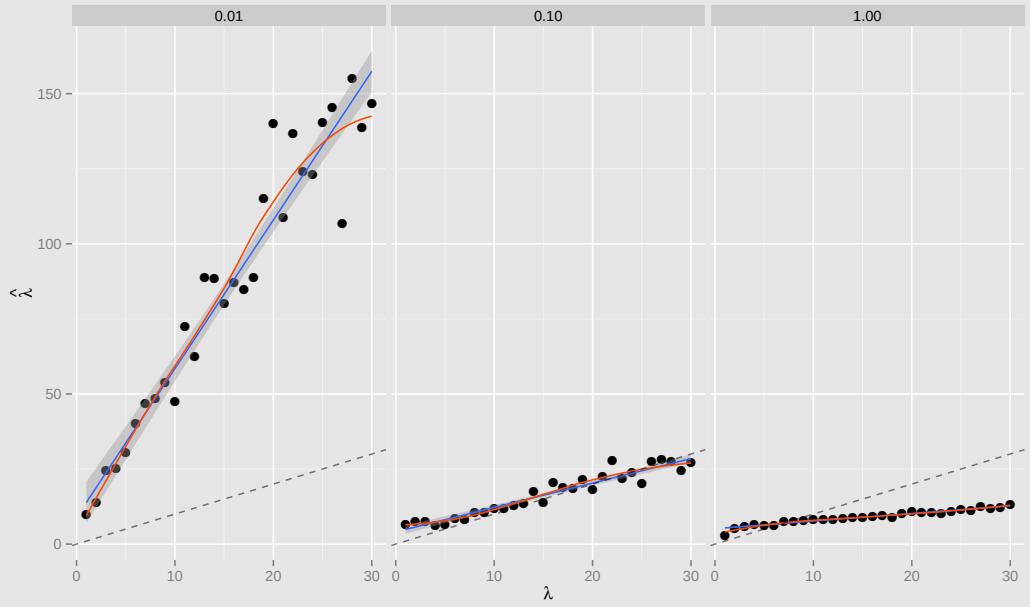


Figure 6.4: The estimated mean plotted against the true mean of simulated negative binomial data, with a true size parameter of 0.01, 0.1, and 1. The blue line and gray band, are a robust linear model fit and the accompanying standard error, respectively. The red line is a loess-fit, and the gray dashed line indicates a one-to-one fit.

6.3 Future Work

This section will serve as a short introduction to potential future work based on the this thesis.

Modelling String Coverage with Artefacts and Mixtures:
A way of incorporating artefacts has already been discussed above, in Section 6.2.1. The addition of mixtures, introduces another parameter ϕ_i, indicating the fraction of the coverage belonging to individual i. The artefact adjusted model discussed in Section 6.2.1, is a convenient extension based on previous work in CE, more appropriate methods may yet be discovered.

Development of R-package
This is, probably, the easiest of our proposals, as most of the code is already written, though some could benefit from an overhaul (or conversion to C++) to optimise its speed.

Examination of RUL

The first question regarding the RUL is, do we even need a refinement of the LUS? From a purely statistical standpoint, it will depend on the predictive performance of the RUL compared to the LUS. We see from Figure 6.1, that the relationship between the stutter ratio and the RUL is not necessarily linear, which implies a possible increase in model complexity. From a more practical standpoint, the gain obtained by this further refinement, is most likely not that substantial.

Verification of the EM-Algorithm using Profile-likelihood

As mentioned in Section 6.2.3, we need to prove that the use of a profile-likelihood in the M-step of the EM-algorithm, still ensures a non-decreasing likelihood. This is achieved by showing the profile-likelihood is concave for every iteration of the algorithm.

BIBLIOGRAPHY

[1] John M. Butler. *Fundamentals for Forensic DNA Typing.* Academic Press, 2010.

[2] John M. Butler. *Advanced Topics in Forensic DNA Typing: Methodology.* Academic Press, 2012.

[3] William Goodwin, Adrian Linacre, and Sibte Hadi. *An Introduction to Forensic Genetics.* Wiley-Blackwell, 2011.

[4] Peter Gill, James Curran, and Keith Elliot. A Graphical Model of the entire DNA Process Associated with the Analysis of Short Tandem Repeat Loci. *Nucleic Acids Research*, 2005.

[5] Clare Brookes, Jo-Anne Bright, SallyAnn Harbison, and John Buckleton. Characterising Stutter in Forensic STR Multiplexes. *Forensic International Science: Genetics*, 2011.

[6] Torben Tvedebrink, Helle Smidt Mogensen, Maria Charlotte Stene, and Niels Morling. Performence of two 17 Locus Forensic Identification STR Kits - Biosystems's AmpFℓSTR NGMSElect and Promega's PowerPlex ESI17 Kits. *Forensic International Science: Genetics*, 2011.

[7] Eric E. Schadt, Steve Turner, and Andrew Kasarskis. A Window into Third-generation Sequencing. *Human Molecular Genetics*, 2010.

[8] Eva C. Berglund, Anna Kiialainen, and Ann-Christine Syvanen. Next Generation Sequencing Technologies and Applications for Human Genetic History and Forensics. *Investigative Genetics*, 2011.

[9] Claus Børsting and Niels Morling. Next Generation Sequencing and its Applications in Forensic Genetics. *Forensic Science International: Genetics*, 2015.

[10] Roche-454. http://www.454.com/. Accessed: 2014-09-16.

[11] LT-IonTorrent. http://www.lifetechnologies.com/dk/en/home/brands/ion-torrent.html/, . Accessed: 2014-09-16.

[12] Illumina MiSeq. http://systems.illumina.com/systems/miseq/technology.ilmn/. Accessed: 2014-09-26.

[13] Illumina TruSeq Chemistry. http://truseq.illumina.com/truseq.html/. Accessed: 2014-09-26.

[14] Bioconductor. http://www.bioconductor.org/. Accessed: 2014-09-22.

[15] Picard. http://broadinstitute.github.io/picard/. Accessed: 2014-12-09.

[16] David H. Warshauer, David Lin, Kumar Hari, Ravi Jain, Carey Davis, Bobby LaRue, Jonathan L. King, and Bruce Budowle. STRait Razor: A Length-based Forensic STR Allele-calling Tool for Use with Second-generation Sequencing Data. *Forensic Science International: Genetics*, 2013.

[17] Strbase. http://www.cstl.nist.gov/strbase/. Accessed: 2015-01-08.

[18] Peter Müller, Giovanni Parmigiani, Christian Robert, and Judith Rousseau. Optimal Sample Size for Multiple Testing: The Case of Gene Expression Microarrays. *American Statistical Association*, 2005.

[19] Peter J. A. Cock, Christopher J. Fields, Naohisa Goto, Michael L. Heuer, and Peter M. Rice. The Sanger FASTQ File Format for Sequences with Quality Scores and the Solexa-Illumina FASTQ Variants. *Nucleic Acids Research*, 2010.

[20] IonTorrent Technical Note: The Per-Base Quality Score System. http://mendel.iontorrent.com/ion-docs/Technical-Note---Quality-Score_6128102.html/, . Accessed: 2014-12-27.

[21] Brent Ewing and Phil Green. Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. *Genome Research*, 1998.

[22] Lauren M. Bragg, Glenn Stone, Margaret K. Butler, Philip Hugenholtz, and Gene W. Tyson. Shining a Light on Dark Sequencing: Characterising Errors in Ion Torrent PGM Data. *PLOS Computational Biology*, 2013.

[23] Jo-Anne Bright, Duncan Taylor, James M. Curran, and John Buckleton. Developing Allelic and Stutter Peak Height Models for a Continues Method of DNA Interpretation. *Forensic International Science: Genetics*, 2012.

[24] Jo-Anne Bright, James M. Curran, and John Buckleton. Investigation into the Performance of different Models for Predicting Stutter. *Forensic International Science: Genetics*, 2013.

[25] William H. Greene. Accounting for Excess Zeros and Sample Selection in Poisson and Negative Binomial Regression Models. *NYU Working Paper*, 1994.

[26] Hwa Kyung Lim, Wai Keung Li, and Philip L.H. Yu. Zero-inflated Poisson Mixture Model. *Computational Statistics and Data Analysis*, 2013.

[27] Hannah Kelly, Jo-Anne Bright, James M. Curran, and John Buckleton. Modelling Heterozygote Balance in Forensic DNA Profiles. *Forensic International Science: Genetics*, 2012.

[28] Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. Wiley, 2002.

[29] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*. Springer-Verlag New York, 2009.

[30] Donald B. Rubin. Inference and Missing Data. *Biometrika*, 1976.

[31] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, 1977.

[32] R. G. Cowell, S. L. Lauritzen, and J. Mortera. A Gamma Model for DNA Mixture Analyses. *Bayesian Analysis*, 2007.

[33] R. G. Cowell, S. L. Lauritzen, and J. Mortera. Probabilistic Expert Systems for Handling Artifacts in Complex DNA Mixtures. *Forensic Science International: Genetics*, 2011.

[34] R. G. Cowell, T. Graversen, S. L. Lauritzen, and J. Mortera. Analysis of Forensic DNA Mixtures with Artifacts. *Journal of the Royal Statistical Society*, 2013.

[35] Benjamin Milo Bolstad. Comparing some Iterative Methods of Parameter Estimation for Censored Gamma Data. Master's thesis, The University of Waikato, 1998.

[36] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, 1964.

[37] Xiao-Li Meng and Donald B. Rubin. Maximum Likelihood Estimation via the ECM Algorithm: A General Framework. *Journal of the Royal Statistical Society*, 1977.

[38] C.G. Broyden. A new Double-rank Minimization Algorithm. *Notice of the American Mathematical Society*, 1969.

[39] R. Fletcher. A new Approach to Variable Metric Methods. *The Computer Journal*, 1970.

[40] D. Goldfarb. A Family of Variable Metric Methods Derived by Variational Means. *Mathematics of Computation*, 1970.

[41] D.F. Shanno. Conditioning of quasi-Newton Methods for Function Minimization. *Mathematics of Computation*, 1970.

[42] G. E. P. Box and D. R. Cox. An Analysis of Transformations. *Royal Statistical Society*, 1964.

[43] James H. Ricketts and Geoffrey A. Head. The Five-parameter Logistic Equation for Investigating Asymmetry of Curvature in Baroreflex Studies. *The American Physiological Society*, 1999.

[44] Jesús Giraldo, Nuria M. Vivas, Elisabet Vila, and Alber Badia. Assessing the (a)symmetry of Concentration-effect Curves: Empirical versus Mechanistic Models. *Pharmacology and Therapeutics*, 2002.

[45] Paul G. Gottschalk and John R. Dunn. The Five-parameter Losgistic: A Characterization and Comparison with the Four-paramter Logistic. *Analytical Biochemistry*, 2005.

APPENDIX

A

THE INCOMPLETE REGULARISED BETA FUNCTION

This chapter will contain a short look at the incomplete regularised beta function, including derivatives thereof, which we will need in the main thesis. The beta function (sometimes also referred to as the Euler function of the first kind) is given as:

$$\mathrm{B}(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt, \tag{A.1}$$

where $a, b \in \mathbb{R}$ and $a, b > 0$. The beta function is a symmetric function and can be written as a product of gamma functions:

$$\mathrm{B}(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}, \tag{A.2}$$

where the gamma function takes is usual form:

$$\Gamma(t) = \int_0^\infty x^{t-1}\exp(-x)\,dx. \tag{A.3}$$

The beta function can be extended to an incomplete beta function defined as follows:

$$\mathrm{B}(x;a,b) = \int_0^x t^{a-1}(1-t)^{b-1}dt. \tag{A.4}$$

We see that the beta function is a special case of the incomplete beta function with $x = 1$. The incomplete regularised beta function (also called regularised beta function) is a regularisation of the incomplete beta function, using the beta function.

That is, the regularised beta function is given as:

$$I_x(a,b) = \frac{\mathrm{B}(x;a,b)}{\mathrm{B}(a,b)}. \tag{A.5}$$

In Section 4.1.1, we need the derivatives of a regularised beta function, where $x = \frac{\lambda}{\lambda+\theta}$, $a = k$, and $b = \theta$, with respect to λ and θ. Furthermore, we take the logarithm of the regularised beta function. The derivative w.r.t. λ is:

$$\begin{aligned}\frac{\partial}{\partial\lambda}\left[\log\left(I_{\frac{\lambda}{\lambda+\theta}}(k,\theta)\right)\right] &= \frac{\partial}{\partial\lambda}\left[\log\left(\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)\right)\right] \\ &= \frac{1}{\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)}\frac{\partial}{\partial\lambda}\left[\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)\right]\end{aligned} \tag{A.6}$$

The derivative w.r.t. the θ parameter is found as follows:

$$\begin{aligned}\frac{\partial}{\partial\theta}\left[\log\left(I_{\frac{\lambda}{\lambda+\theta}}(k,\theta)\right)\right] &= \frac{\partial}{\partial\theta}\left[\log\left(\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)\right) - \log\left(\mathrm{B}\left(k,\theta\right)\right)\right] \\ &= \frac{\frac{\partial}{\partial\theta}\left[\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)\right]}{\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right)} - \frac{\frac{\partial}{\partial\theta}\left[\mathrm{B}\left(k,\theta\right)\right]}{\mathrm{B}\left(k,\theta\right)}\end{aligned} \tag{A.7}$$

The derivatives of the incomplete beta functions, with respect to λ and θ, are given as follows:

$$\begin{aligned}\frac{\partial}{\partial\lambda}\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right) &= \frac{\partial}{\partial\lambda}\int_0^{\frac{\lambda}{\lambda+\theta}} t^{k-1}(1-t)^{\theta-1}dt \\ &= \left(\frac{\lambda}{\lambda+\theta}\right)^{k-1}\left(1-\frac{\lambda}{\lambda+\theta}\right)^{\theta-1}\frac{\theta}{(\lambda+\theta)^2} \\ &= \frac{\lambda^{k-1}\theta^{\theta}}{(\lambda+\theta)^{k+\theta}}\end{aligned} \tag{A.8}$$

The derivative w.r.t. θ is a bit more complicated as the integrand also depends on θ:

$$\begin{aligned}\frac{\partial}{\partial\theta}\mathrm{B}\left(\frac{\lambda}{\lambda+\theta};k,\theta\right) &= \frac{\partial}{\partial\theta}\int_0^{\frac{\lambda}{\lambda+\theta}} t^{k-1}(1-t)^{\theta-1}dt \\ &= -\left(\frac{\lambda}{\lambda+\theta}\right)^{k-1}\left(1-\frac{\lambda}{\lambda+\theta}\right)^{\theta-1}\frac{\lambda}{(\lambda+\theta)^2} + \int_0^{\frac{\lambda}{\lambda+\theta}} t^{k-1}(1-t)^{\theta-1}\log\left(1-t\right)dt \\ &= -\frac{\lambda^{k}\theta^{\theta-1}}{(\lambda+\theta)^{k+\theta}} + \int_0^{\frac{\lambda}{\lambda+\theta}} t^{k-1}(1-t)^{\theta-1}\log\left(1-t\right)dt\end{aligned} \tag{A.9}$$

All that remains is to calculate the derivative of the beta function:

$$\begin{aligned}\frac{\partial}{\partial\theta}\mathrm{B}\left(k,\theta\right) &= \frac{\partial}{\partial\theta}\left[\frac{\Gamma(k)\Gamma(\theta)}{\Gamma(k+\theta)}\right] \\ &= \frac{\Gamma(k)\Gamma'(\theta)}{\Gamma(k+\theta)} - \frac{\Gamma(k)\Gamma(\theta)\Gamma'(k+\theta)}{\Gamma(k+\theta)^2} \\ &= \frac{\Gamma(k)\Gamma(\theta)}{\Gamma(k+\theta)}\left(\frac{\Gamma'(\theta)}{\Gamma(\theta)} - \frac{\Gamma'(k+\theta)}{\Gamma(k+\theta)}\right) \end{aligned} \tag{A.10}$$

We include the last equality as it will simply Equation (A.7). We will have solve the last integral in Equation (A.9) numerically, and as it is a one-dimensional integral we can use the `integrate`-function in **R**. Furthermore, $\Gamma'(x)/\Gamma(x)$ is the digamma function, and is already implemented in **R**.

APPENDIX

B

NOISE SIMULATION

We want to carry out a simulation study of our homebrew functions for modelling noise, in order to investigate how well they perform. First, however, a comment on the implementation of weighted fractile method.

As we in our data have extreme outliers, with respect to the noise distribution, i.e. the coverage of the true alleles, the density values under the noise model of these outliers would be very small. In fact, the density is so small in these points that it is assigned a value of zero. The consequence being that either z^*, seen in Equation (4.6), is divided by zero or we take the logarithm of zero when calculating the log-likelihood, resulting in `NaN` and `-Inf` values, in **R**, respectively. The first case would result in an error, where the latter would make the log-likelihood function diverge to negative infinity. In order to avoid these consequences, we restrict the values used when calculating the log-likelihood to values smaller than the 99%-fractile. Furthermore, we ensure that the upper quantile, q_{upper}, used is smaller than the 99%-fractile.

We will generate one inflated negative binomial data, using different values of the sample size, mean, and size parameters (N, λ, and θ, respectively), and fit the parameters using the **kinb** (both truncated and non-truncated, to examine the difference) and `weighted.fractile` methods. We will not use different values of the mixing parameter, π, as previous simulations have already shown that π is well

estimated by all three methods. The data is simulated as follows:

(i) The amount of excess one's is calculated using a binomial distribution with π as the probability of success in each trial and N as the number of trials. That is, $n_{\text{infl}} \sim \text{Bin}\,(N, \pi)$, leaving an effective sample size $n_{\text{eff}} = N - n_{\text{infl}}$.

(ii) We then sample from a negative binomial distribution with parameters λ and θ, i.e. $x_i \sim \text{NegBin}\,(\lambda, \theta)$, where $i = 1, \ldots, n_{\text{eff}}$. Note: we use the mean representation of the negative binomial distribution.

The process in items (i)-(ii) will be replicated ten times. The total number of unique strings per locus range somewhere between 400 and 2,000 reads. Based on previous simulations and estimates of noise seen in the files from dilution series, we know $\pi \in [0.5; 0.7]$, $\lambda \in [2; 7]$, and $\theta \in [0.001; 2]$. Therefore, we let $N \in \{500, 1{,}500, 2{,}500\}$, $\pi = 0.6$, $\lambda = \{1, 2, \ldots, 10\}$, and $\theta \in \{0.01, 0.1, 1\}$. The median and standard error of the parameter estimates can be seen in Tables B.1 - B.3, (though these tables are rather large, making it difficult to glean any relevant information, they have been included for the sake of completeness).

Figure B.1 shows boxplots of the difference between the true mean value and the estimate of the mean, $d(\widehat{\lambda}) = \lambda - \widehat{\lambda}$, plotted against the true mean, λ, shown for the three sample sizes, $N = 500$, 1,500 and 2,500. We see that as the true mean increases the difference, $d(\widehat{\lambda})$, and the variance of the $\widehat{\lambda}$-estimates, using the weighted fractile method, increases. We see this effect across all three sample sizes, in fact we do not see any difference between the three different sample sizes for either of the three methods used.

As the sample size does not seem to have an effect on the mean, we have tried stratifying on the values of the true size parameter instead, the resulting boxplots can be seen in Figure B.2. The figure shows that the smallest size parameter 0.01, has a huge effect on the mean estimates of the weighted fractile method. Furthermore, we also see that the weighted fractile method has a general tendency to overestimate the mean parameter, whereas the KINB methods generally underestimates the parameter.

The difference between the true size parameter and the estimated parameter, $d(\widehat{\theta}) = \theta - \widehat{\theta}$ is worse when the mean parameter, λ, is close to 1, as seen in Figure B.3, which makes sense as our data is one inflated implying that the two mixtures would be

harder to tell apart. Furthermore, we see as we increase the size parameter the variation gets worse as well, which leads to the extreme difference of -713.32 for the truncated KINB method.

Figure B.4 shows the difference $d(\widehat{\theta})$ plotted against the sample size, in order to show that the extreme values for $\theta = 1$, is not a product of a small sample size, but is purely affected by the true θ parameter.

In general we do not see much difference between the two KINB methods, though it does seem that the truncated version gets closer to the true size parameter than the non-truncated KINB. As the mean increases the variance of $d(\widehat{\theta})$ seems to decrease all three methods (or at the very least stabilise). All three methods overestimate the size, however as the mean increases, the absolute difference, $|d(\widehat{\theta})|$, decreases.

We see that when estimating the size the truncated methods are clearly superior to the non-truncated. We have not included a non-truncated version of the weighted fractile method in this chapter, even though it has also been examined, as it generally performs worse than the non-truncated **`kinb`**-function and the truncated weighted fractile function, when estimating the size and mean respectively.

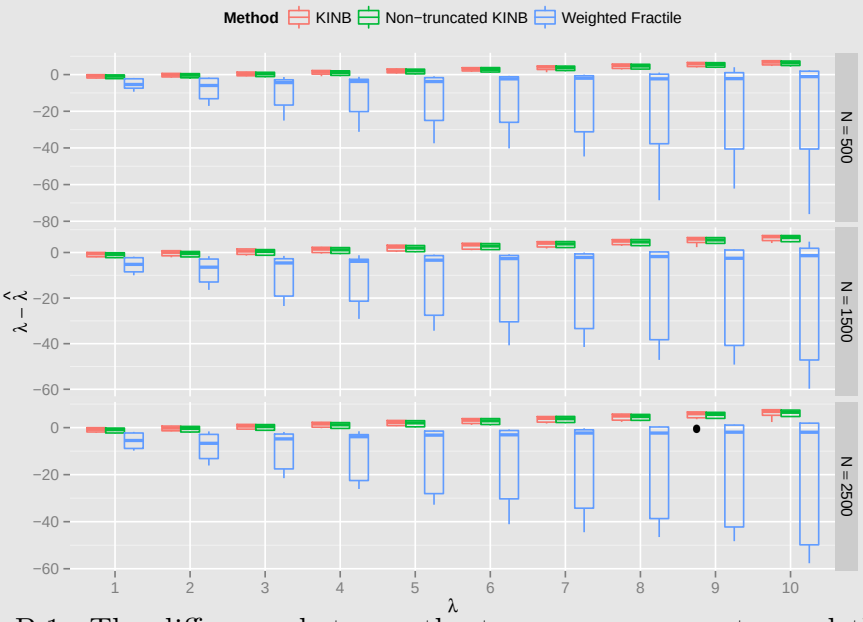


Figure B.1: The difference between the true mean parameter and the estimated mean parameter, $d(\widehat{\lambda}) = \lambda - \widehat{\lambda}$, plotted against the true mean λ, for each sample size N.

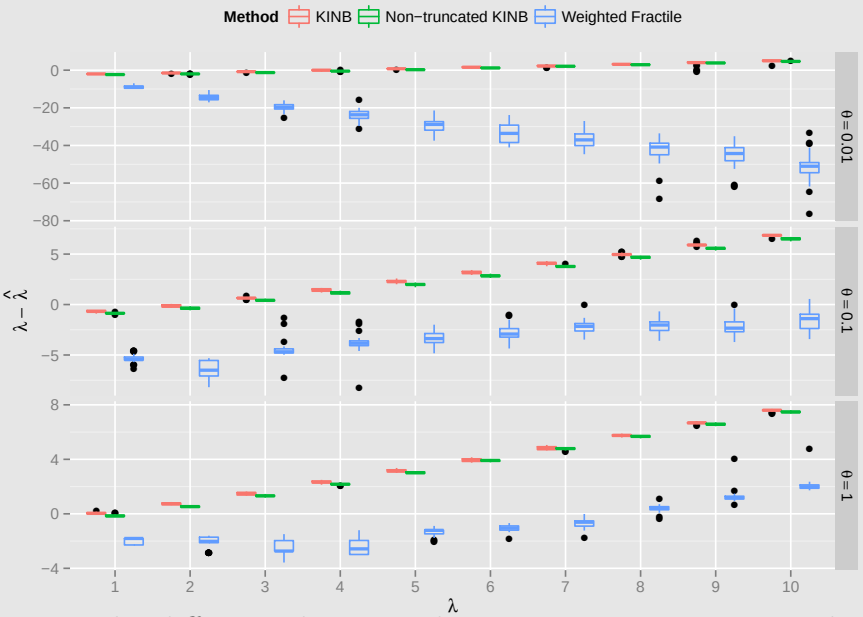


Figure B.2: The difference between the true mean parameter and the estimated mean parameter, $d(\widehat{\lambda}) = \lambda - \widehat{\lambda}$, plotted against the true mean λ, for value of the true size parameter θ.

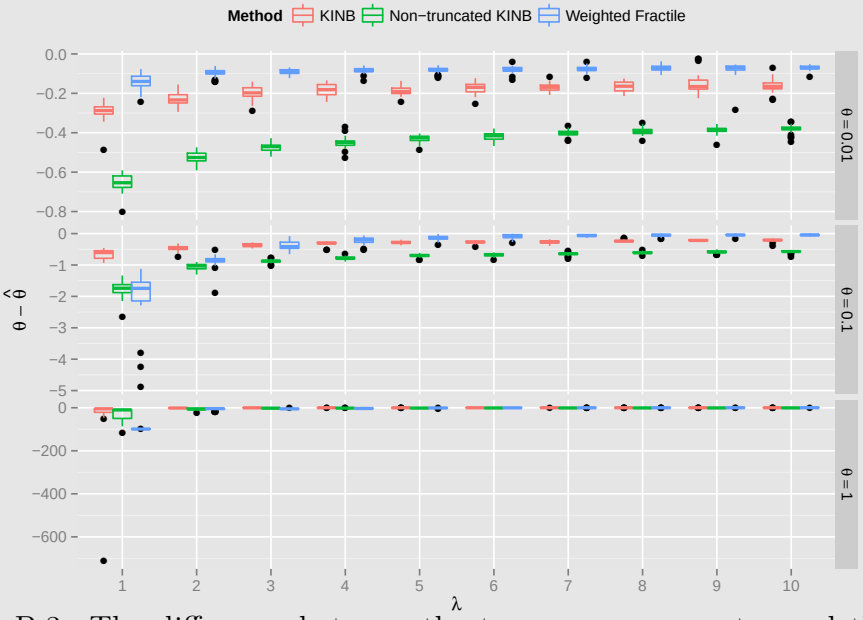


Figure B.3: The difference between the true mean parameter and the estimated mean parameter, $d(\widehat{\theta}) = \theta - \widehat{\theta}$, plotted against the true mean λ, for value of the true size parameter θ.

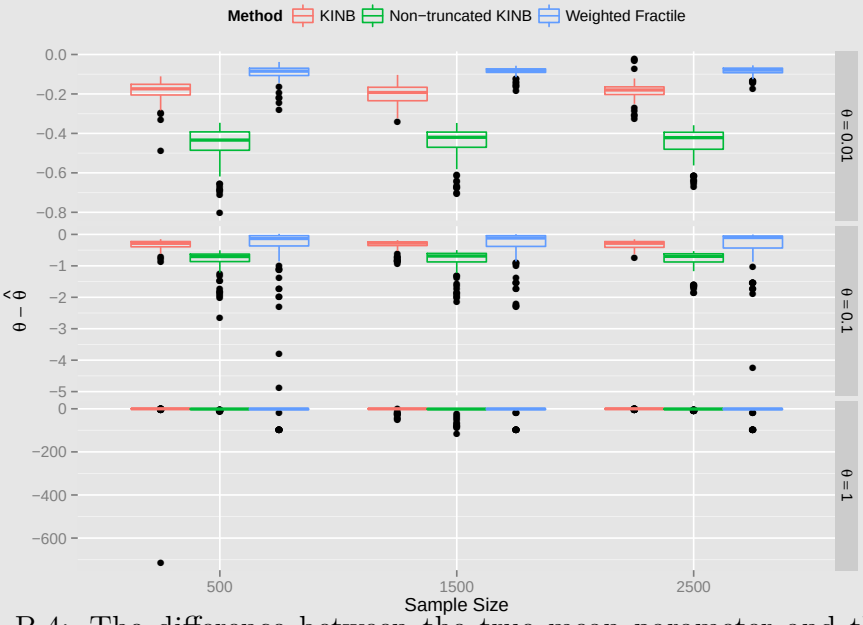


Figure B.4: The difference between the true mean parameter and the estimated mean parameter, $d(\widehat{\theta}) = \theta - \widehat{\theta}$, plotted against the true size θ, for each sample size N.

Table B.1: The median and the standard error of the parameters estimated using the **kinb**-function.

KINB

$N = 500$

	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	2.9859	0.0349	0.2910	0.0215	1.7189	0.0293	0.6790	0.0429	0.9211	0.0322	5.3488	70.9295
2	3.4778	0.0652	0.2344	0.0105	2.1851	0.0329	0.5529	0.0229	1.2708	0.0160	2.7165	0.1456
3	3.7393	0.0429	0.1926	0.0077	2.3530	0.0294	0.4964	0.0177	1.6100	0.0257	1.1188	0.0592
4	3.9484	0.1222	0.1841	0.0075	2.6212	0.0508	0.4142	0.0230	1.7025	0.0299	0.9257	0.1153
5	4.1409	0.0555	0.1905	0.0080	2.6990	0.0530	0.3575	0.0209	1.8813	0.0313	0.9592	0.0884
6	4.2144	0.0724	0.1622	0.0118	2.8757	0.0446	0.3838	0.0209	2.1434	0.0247	1.0581	0.0330
7	4.7489	0.1823	0.1651	0.0073	2.8397	0.0341	0.3433	0.0251	2.1772	0.0340	0.8235	0.0441
8	4.7701	0.0746	0.1613	0.0090	3.0259	0.0431	0.3428	0.0153	2.2224	0.0200	0.9212	0.0642
9	4.8344	0.0660	0.1813	0.0106	3.0868	0.0468	0.3214	0.0115	2.3070	0.0151	0.8439	0.0492
10	4.9659	0.0748	0.1764	0.0098	3.1825	0.0248	0.3049	0.0218	2.3993	0.0147	0.8961	0.0464

$N = 1500$

	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	3.0102	0.0261	0.3079	0.0062	1.5864	0.0065	0.9351	0.0224	0.9215	0.0141	25.9498	4.7805
2	3.6444	0.0720	0.2628	0.0100	2.0104	0.0367	0.5631	0.0249	1.1930	0.0132	1.9412	0.0329
3	4.0024	0.0693	0.2445	0.0121	2.3772	0.0319	0.3993	0.0092	1.4063	0.0130	1.5050	0.0438
4	4.2832	0.0651	0.2198	0.0063	2.5341	0.0453	0.4133	0.0221	1.6048	0.0300	1.1796	0.0393
5	4.4846	0.0606	0.2071	0.0068	2.6525	0.0277	0.3699	0.0139	1.7557	0.0174	0.9602	0.0545
6	4.7429	0.0581	0.1913	0.0053	2.6968	0.0369	0.3645	0.0081	1.8873	0.0255	0.8408	0.0595
7	4.7966	0.0799	0.1911	0.0052	2.9287	0.0317	0.3624	0.0085	2.0182	0.0422	0.9080	0.0694
8	4.8268	0.0819	0.1688	0.0066	3.0332	0.0346	0.3495	0.0075	2.3134	0.0221	0.8710	0.0727
9	5.0661	0.2354	0.1696	0.0099	3.1311	0.0214	0.3251	0.0074	2.3783	0.0276	0.8940	0.0740
10	5.0868	0.1277	0.1744	0.0102	3.1195	0.0378	0.3218	0.0064	2.4446	0.0320	0.8779	0.0869

$N = 2500$

	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	3.0083	0.0399	0.2693	0.0120	1.7213	0.0101	0.6563	0.0100	1.0008	0.0050	5.4638	0.1463
2	3.4869	0.0354	0.2275	0.0078	2.2107	0.0350	0.5403	0.0311	1.3336	0.0153	2.5462	0.0626
3	3.7428	0.0273	0.1953	0.0061	2.3640	0.0098	0.5278	0.0126	1.6128	0.0223	1.1938	0.0345
4	3.8982	0.0445	0.1701	0.0071	2.4640	0.0149	0.3803	0.0065	1.6491	0.0249	1.0194	0.0260
5	4.1994	0.0342	0.1955	0.0051	2.8481	0.0483	0.3947	0.0075	1.9164	0.0139	0.9524	0.0250
6	4.3018	0.0833	0.1820	0.0076	2.8638	0.0340	0.3646	0.0069	2.0719	0.0207	0.9276	0.0385
7	4.9143	0.0958	0.1818	0.0067	3.0566	0.0599	0.3701	0.0184	2.2792	0.0245	0.9770	0.0346
8	5.0082	0.0867	0.1970	0.0063	3.0779	0.0383	0.3268	0.0071	2.2230	0.0242	0.9233	0.0349
9	5.1020	0.6804	0.1748	0.0222	3.0917	0.0221	0.3065	0.0069	2.3100	0.0092	0.8791	0.0121
10	4.9826	0.2766	0.1741	0.0107	3.1341	0.0185	0.2953	0.0074	2.3789	0.0056	0.8754	0.0147

Table B.2: The median and the standard error of the parameters estimated using the non-truncated **kinb**-function.

Non-truncated KINB

	$N = 500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	3.2696	0.0235	0.6891	0.0181	1.8507	0.0175	1.9192	0.1189	1.1594	0.0108	10.6096	0.5879
2	3.9847	0.0597	0.5421	0.0115	2.3240	0.0311	1.1310	0.0362	1.4342	0.0137	8.0284	0.9492
3	4.2111	0.0670	0.4831	0.0095	2.5820	0.0329	0.9541	0.0193	1.6713	0.0293	2.9232	0.2310
4	4.5211	0.0714	0.4575	0.0142	2.8906	0.0362	0.8745	0.0228	1.8468	0.0253	2.8339	0.1510
5	4.7362	0.0447	0.4444	0.0090	3.0101	0.0449	0.7797	0.0231	2.0123	0.0220	2.4995	0.1532
6	4.8427	0.0614	0.4395	0.0081	3.1703	0.0402	0.7767	0.0240	2.1129	0.0239	2.0874	0.0597
7	4.9141	0.0613	0.4082	0.0083	3.2577	0.0392	0.7425	0.0232	2.2580	0.0324	1.7559	0.0576
8	5.0574	0.0582	0.4036	0.0081	3.3024	0.0327	0.7112	0.0197	2.3532	0.0203	1.9194	0.0719
9	5.1477	0.0716	0.3902	0.0096	3.4059	0.0392	0.6990	0.0180	2.4287	0.0243	1.5793	0.0650
10	5.2539	0.0848	0.3936	0.0100	3.5571	0.0464	0.6704	0.0198	2.5423	0.0185	1.6090	0.0448
	$N = 1500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	3.3511	0.0386	0.6781	0.0102	1.8732	0.0211	1.9675	0.0736	1.0646	0.0228	75.3268	7.4097
2	4.0216	0.0263	0.5216	0.0110	2.4793	0.0408	1.0950	0.0465	1.5391	0.0163	2.9878	2.3632
3	4.3133	0.0276	0.4792	0.0066	2.5569	0.0229	0.9786	0.0167	1.6876	0.0118	2.6800	0.0823
4	4.5257	0.0436	0.4566	0.0046	2.8326	0.0368	0.9015	0.0139	1.8259	0.0185	2.8129	0.0463
5	4.7235	0.0291	0.4395	0.0056	3.0542	0.0438	0.7800	0.0165	1.9167	0.0100	2.4929	0.0625
6	4.7804	0.0258	0.4219	0.0058	3.1549	0.0328	0.7760	0.0147	2.0653	0.0110	2.1391	0.0398
7	4.8874	0.0368	0.4161	0.0029	3.2349	0.0210	0.7324	0.0145	2.1708	0.0137	2.0212	0.0250
8	5.0678	0.0355	0.4016	0.0067	3.3726	0.0311	0.7113	0.0060	2.2829	0.0132	1.9002	0.0233
9	5.1373	0.0414	0.3998	0.0040	3.4943	0.0210	0.6724	0.0110	2.4136	0.0213	1.8448	0.0318
10	5.3464	0.0421	0.3867	0.0069	3.4648	0.0248	0.6698	0.0094	2.4801	0.0220	1.7561	0.0356
	$N = 2500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	3.3499	0.0296	0.6389	0.0062	1.8822	0.0128	1.7603	0.0300	1.1826	0.0050	9.9395	0.1992
2	3.9448	0.0397	0.5361	0.0050	2.3725	0.0303	1.1712	0.0237	1.4656	0.0070	7.9949	0.3042
3	4.2683	0.0411	0.4843	0.0059	2.6097	0.0125	0.9811	0.0178	1.6702	0.0136	3.5427	0.1280
4	4.4089	0.0286	0.4609	0.0061	2.8043	0.0260	0.8811	0.0098	1.8245	0.0167	2.9538	0.0817
5	4.7542	0.0310	0.4336	0.0031	2.9848	0.0233	0.8123	0.0065	2.0311	0.0086	2.5022	0.1118
6	4.8981	0.0543	0.4268	0.0040	3.1057	0.0212	0.7697	0.0091	2.1238	0.0189	1.8341	0.0564
7	4.9966	0.0454	0.4098	0.0031	3.2201	0.0218	0.7416	0.0057	2.2338	0.0123	1.8285	0.0370
8	5.0951	0.0488	0.4109	0.0046	3.3082	0.0327	0.7095	0.0098	2.3515	0.0209	1.7709	0.0235
9	5.1957	0.0379	0.3937	0.0039	3.3919	0.0075	0.6891	0.0095	2.4723	0.0223	1.7072	0.0383
10	5.3684	0.0287	0.3879	0.0025	3.4780	0.0394	0.6750	0.0067	2.5164	0.0102	1.6769	0.0425

Table B.3: The median and the standard error of the parameters estimates using the weighted fractile method.

Weighted Fractile

	$N = 500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	9.2881	0.2828	0.1633	0.0182	6.3988	0.1853	1.9638	0.3834	2.8097	0.0779	100.0001	0.0000
2	16.0940	0.6604	0.1010	0.0090	7.9388	0.2558	0.9190	0.0495	3.6141	0.1303	5.0158	1.6526
3	23.0441	0.8704	0.1002	0.0060	7.3691	0.3355	0.3773	0.0543	5.7351	0.1712	6.2608	0.4154
4	27.6139	1.4645	0.0937	0.0075	7.7448	0.2721	0.2685	0.0432	6.2108	0.1930	3.5996	0.4362
5	34.0411	1.7084	0.1020	0.0081	8.7946	0.2733	0.2640	0.0351	6.3351	0.1369	1.8477	0.2634
6	35.7828	2.0285	0.0945	0.0084	8.3256	0.3510	0.1263	0.0369	7.1166	0.0996	1.3715	0.0299
7	43.5811	1.9516	0.0809	0.0086	8.9729	0.2866	0.1462	0.0144	7.5766	0.1547	1.1964	0.0401
8	52.8507	3.4059	0.0894	0.0077	10.3033	0.2989	0.1743	0.0185	7.6016	0.0923	1.0950	0.0311
9	58.3091	3.0651	0.0948	0.0208	11.2115	0.3867	0.1431	0.0171	7.8308	0.2976	0.8086	0.0705
10	63.3892	4.2044	0.0820	0.0059	11.1059	0.3782	0.1398	0.0125	7.8713	0.0660	0.7451	0.0338
	$N = 1500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	10.2590	0.2239	0.1438	0.0082	6.2110	0.1452	1.8450	0.1145	3.0481	0.0795	100.0001	0.0000
2	17.1514	0.5062	0.1027	0.0060	8.4423	0.2149	0.9586	0.0290	4.0224	0.1444	4.6810	2.5445
3	22.9450	0.6046	0.0971	0.0039	7.5891	0.2681	0.4298	0.0443	5.5226	0.2035	5.7583	0.4983
4	26.4544	0.8424	0.0917	0.0038	7.8798	0.5072	0.3677	0.0441	6.7552	0.2313	4.3293	0.2994
5	33.4447	0.8566	0.0926	0.0027	8.3707	0.1922	0.2572	0.0207	6.2113	0.0974	1.8461	0.0640
6	39.5903	1.3924	0.0935	0.0040	8.6808	0.2017	0.2168	0.0141	7.0445	0.0811	1.3931	0.0311
7	43.1789	1.1445	0.0906	0.0023	9.1741	0.1703	0.1537	0.0098	7.6091	0.1029	1.1961	0.0313
8	47.6546	1.1962	0.0805	0.0035	9.7857	0.1976	0.1289	0.0148	7.5717	0.0781	1.0177	0.0260
9	50.5247	1.0625	0.0868	0.0034	11.5347	0.2036	0.1403	0.0073	7.8370	0.0500	0.8608	0.0296
10	60.1438	1.7283	0.0792	0.0025	11.3963	0.1883	0.1376	0.0086	7.9448	0.2781	0.7857	0.0495
	$N = 2500$											
	$\theta = 0.01$				$\theta = 0.1$				$\theta = 1$			
λ	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE	$\widehat{\lambda}$	SE	$\widehat{\theta}$	SE
1	10.2261	0.1504	0.1430	0.0069	6.4773	0.0720	1.6538	0.2644	2.8097	0.0779	100.0001	0.0000
2	16.5792	0.3424	0.1069	0.0056	8.6740	0.2688	0.9453	0.1052	4.0234	0.1443	4.6963	2.5517
3	22.2492	0.4722	0.1006	0.0041	7.7892	0.2506	0.5422	0.0265	5.7351	0.1487	6.5147	0.4372
4	28.5206	0.4776	0.0959	0.0023	7.8685	0.0776	0.2983	0.0170	6.7619	0.1606	4.3738	0.3195
5	34.2358	0.6748	0.0889	0.0028	8.1960	0.1683	0.2183	0.0136	6.2186	0.0609	1.8461	0.0402
6	40.0952	1.3766	0.0823	0.0024	9.0602	0.0943	0.1895	0.0071	7.0501	0.0632	1.3812	0.0235
7	44.9317	1.2992	0.0841	0.0032	9.3134	0.1310	0.1735	0.0075	7.8766	0.0689	1.1649	0.0173
8	48.1110	1.0018	0.0850	0.0031	10.3209	0.2360	0.1512	0.0049	7.6049	0.0825	0.9833	0.0283
9	54.0522	1.3015	0.0718	0.0024	10.9490	0.1613	0.1450	0.0075	7.8125	0.0314	0.8552	0.0158
10	61.7173	0.8891	0.0783	0.0017	11.9694	0.3135	0.1437	0.0059	8.0425	0.0230	0.7464	0.0120

APPENDIX

C

EXAMINING THE EM-IMPLEMENTATION USING SIMULATED GAMMA COVERAGE

In order to examine the performance of our implementation of the EM-Algorithm, we will simulate data from the a gamma distribution, trying to emulate the real data, using multiple loci. That is, we will have three, nine, and twenty loci (L_1, L_2, and L_3, respectively), each locus containing two observations. The true α parameters used to simulate the gamma distributed data are $\{350, 1, 500, 3, 500\}$, creating nine simulated datasets. Furthermore, we use three distinct $\{\boldsymbol{\eta}_i\}_{i=1}^{3}$ vectors, one for each L_i, the vectors are created drawing L_i observations from a normal distribution with $\mu = 0.9$ and $\sigma = 0.2$. The values for μ and σ were chosen to recreate allele and locus drop-outs.

After the data is simulated, we truncate it using a threshold equal to 250, $1,000$, and $2,500$, respectively. Furthermore, we replicate this process $1,000$ times, and estimate the parameters using both no removal and outer removal of dropped loci (we do not need to include inner removal in our evaluation, as the parameter estimates would be the same as no removal). We do this to examine the effect, on estimating the shape-parameter, when dropped loci are removed.

For the moment we will just concern ourselves with the case where $\alpha = 350$, and nine, L_2, loci. Figure C.1 panel (a), shows quite clearly that the shape parameters are the

same using both removal types. As a consequence the $\boldsymbol{\eta}$ parameters would be equal. Wherein lies the difference between the types? As seen in panel (b) the number of iterations needed to achieve the estimates differs greatly, when the number number of dropped loci is larger than zero. In fact we benefit by removing the dropped loci before estimating the parameters, as the median number of iterations drops from 51 to 2. Furthermore, the average absolute error of the estimated shape parameter is approximately 18.26.

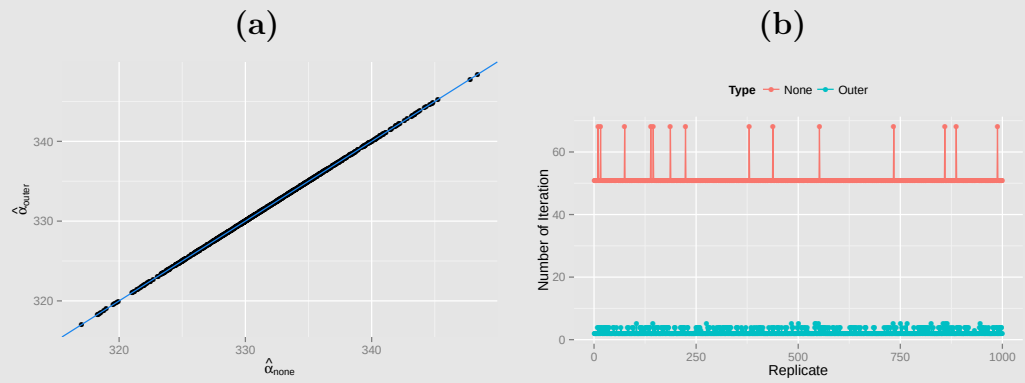


Figure C.1: Panel (a) shows the shape parameters estimated using no removal plotted against the shape parameters estimated using outer removal. The blue line shows a one-to-one relationship. Panel (b) shows the number of iterations needed to converge for no removal and outer removal, here shown in red and blue, respectively. The two panels show data simulated using $\alpha = 350$ and number of loci equal to L_2.

We see, in Figure C.2, a boxplot of the estimation error regarding the scale parameter, stratified for each locus. The estimation error looks fairly consistent, with the exception of locus 8, though this could be a consequence of loci having been dropped (944 out of $1,000$). We see that it makes quite a difference whether locus is dropped or not, in fact it seems algorithm tends to underestimate the the scale when the locus does drop. From this point fourth, we will not limit ourselves to $\alpha = 350$ and nine, L_2, loci.

In Figure C.3 we see boxplots of the relative shape error against the true α parameter. We see that the median relative shape error stays consistent, given the number of loci, though the inter-quantile range shrinks.

Figure C.4, shows the scale estimation error against the true scale parameter. From

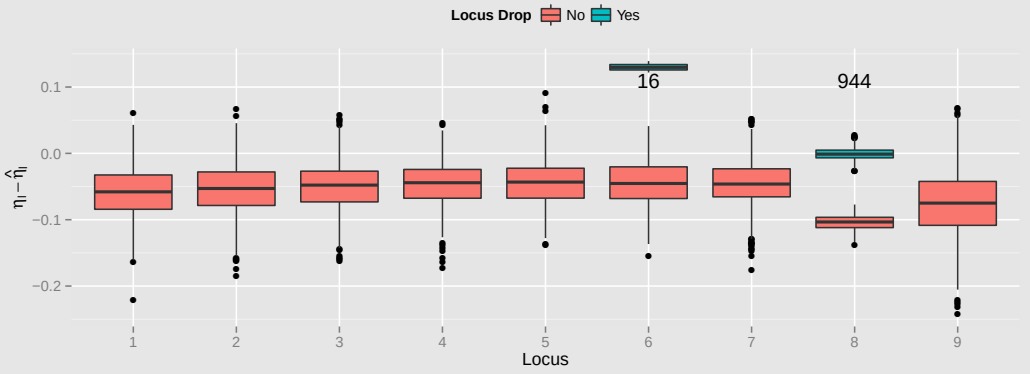


Figure C.2: A boxplot showing the estimation error $\eta_l - \widehat{\eta}_l$, for each locus l. The estimation error when the locus is drop and not-dropped is coloured using red and blue, respectively. The number shown at locus 6, and 8, is the total number of dropped loci out of 1000 replications. The data simulated using $\alpha = 350$ and number of loci equal to L_2.

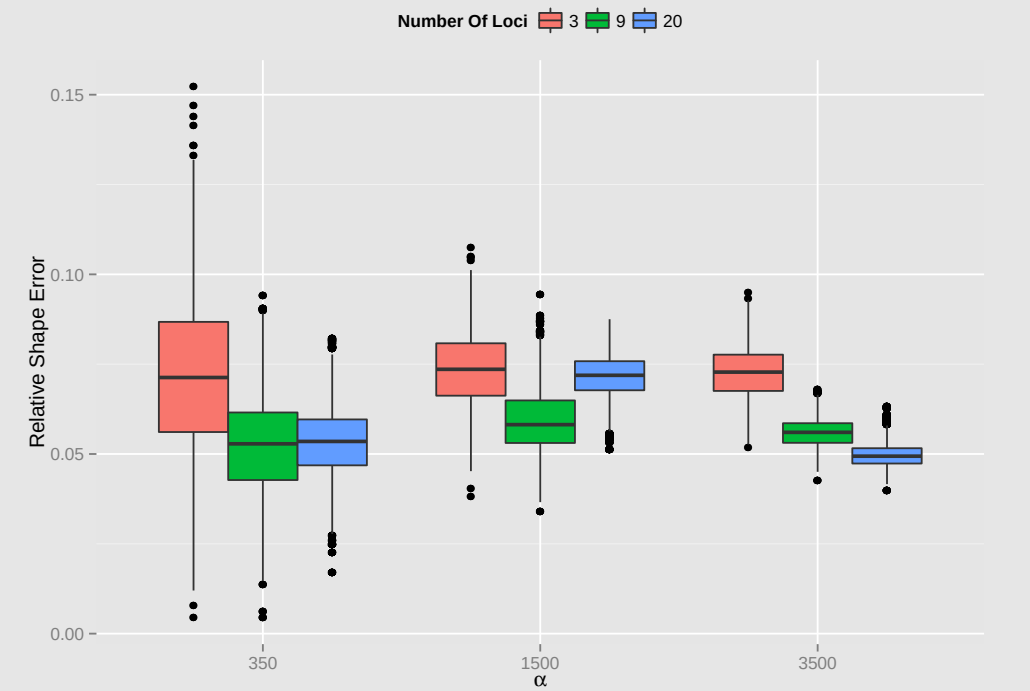


Figure C.3: A boxplot of the relative shape error against the true shape parameter. The number of loci is shown in red, green, and blue, for 3, 9, and 20 loci, respectively.

the figure we see that the standard deviation of the scale estimation error increases with the true parameter, as does its median. Furthermore, we see that the absolute

error is fairly consistent across all three number of loci.

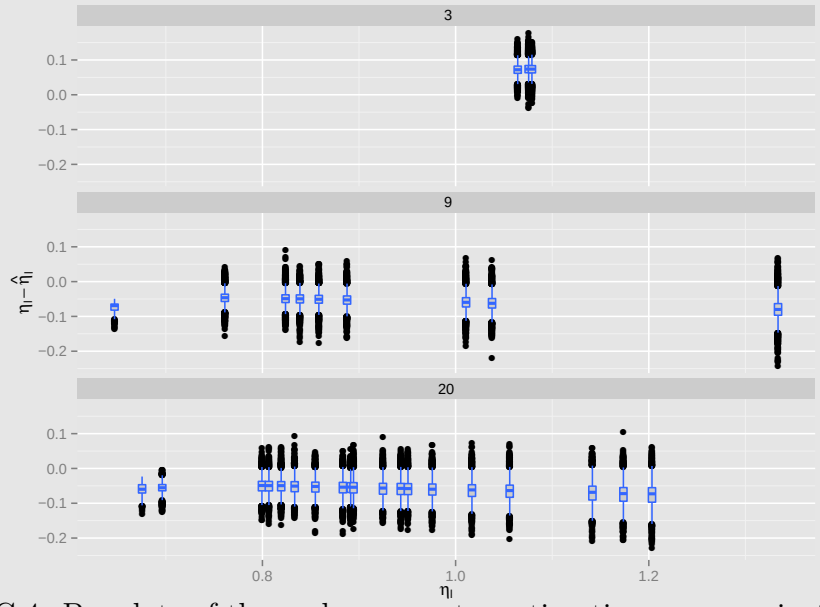


Figure C.4: Boxplots of the scale parameter estimation error against the true scale parameter.

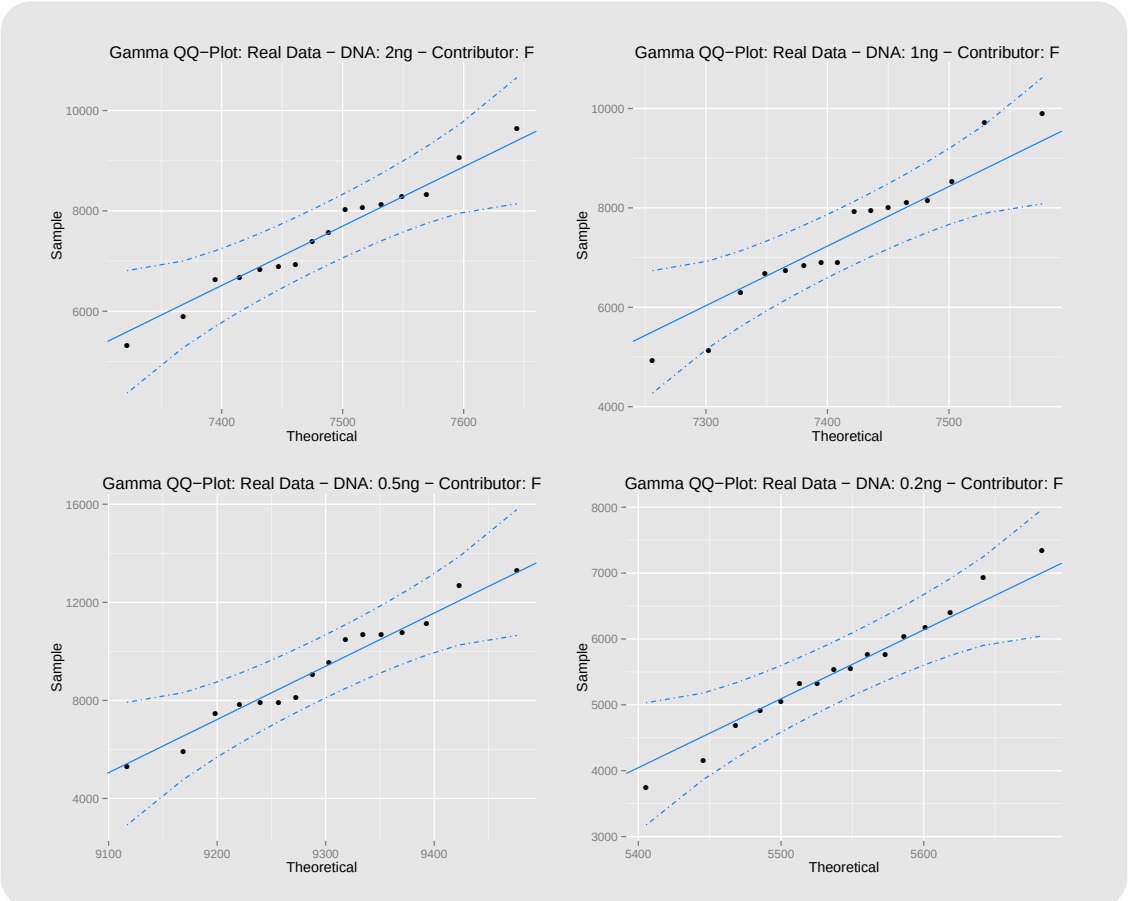
As a last note, it is a bit troubling that the shape parameter does not change, dependent on the removal type and on further inspection we see that the shape parameter rarely change from the initial chosen value (most likely do to the use of the profile-likelihood in the M-step), the algorithm just adjusts the scale parameter. Therefore, we need to choose the shape parameter with care, in a way which makes sense. We choose the to set it as the mean, of the observed coverage across all loci. That is, the shape will act as a sample mean and the scale will then adjust accordingly.

APPENDIX C. EXAMINING THE EM-IMPLEMENTATION USING SIMULATED GAMMA COVERAGE

APPENDIX

D

GAMMA QQ-PLOTS OF REAL AND SIMULATED DATA




Gamma QQ−Plot: Real Data − DNA: 0.1ng − Contributor: F
Gamma QQ−Plot: Real Data − DNA: 0.05ng − Contributor: F
Gamma QQ−Plot: Real Data − DNA: 2ng − Contributor: H
Gamma QQ−Plot: Real Data − DNA: 1ng − Contributor: H
Gamma QQ−Plot: Real Data − DNA: 0.5ng − Contributor: H
Gamma QQ−Plot: Real Data − DNA: 0.2ng − Contributor: H
Gamma QQ−Plot: Real Data − DNA: 0.1ng − Contributor: H
Gamma QQ−Plot: Real Data − DNA: 0.05ng − Contributor: H
Sample
Theoretical


Gamma QQ−Plot: Real Data − DNA: 2ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 1ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 0.5ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 0.2ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 0.1ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 0.05ng − Contributor: F
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 2ng − Contributor: H
Sample
Theoretical
Gamma QQ−Plot: Real Data − DNA: 1ng − Contributor: H
Sample
Theoretical

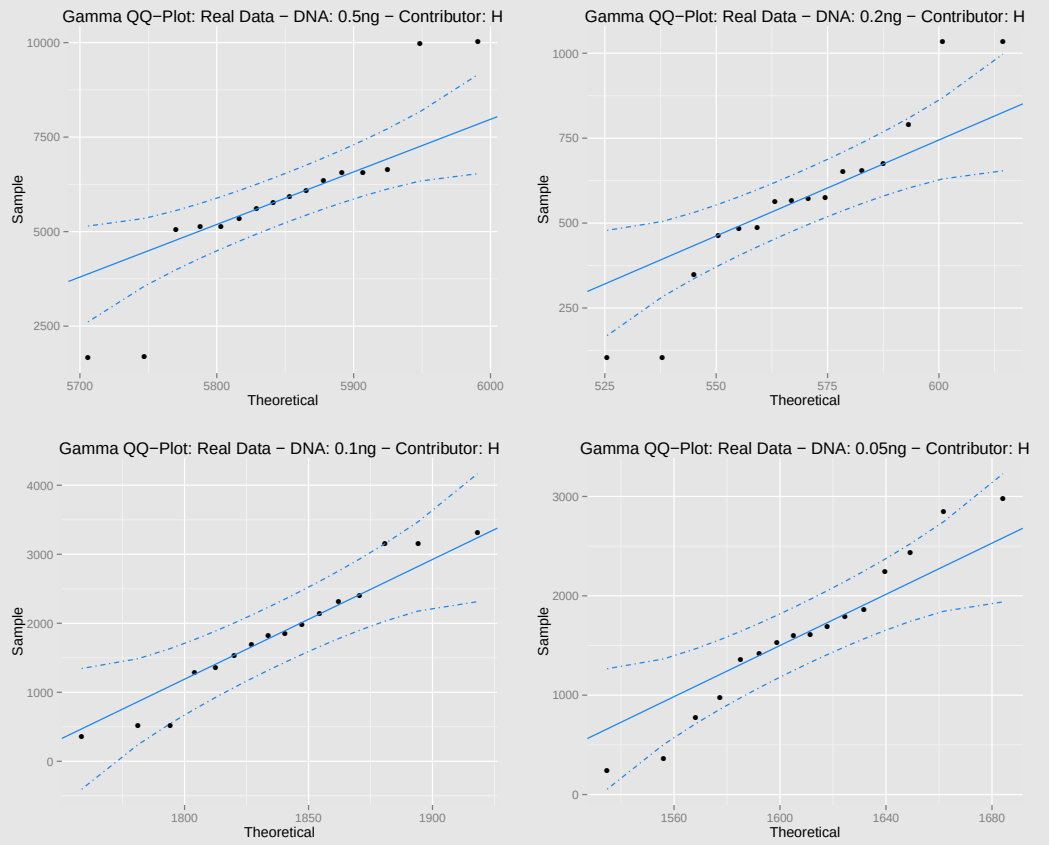


Figure D.1: Gamma QQ-plots of the coverage for each sample in the dilution series. The solid line is a straight line through the first and third quartile. The dashed lines are point-wise 95%-confidence envelopes.

In Figure D.1, we see gamma QQ-plots of the allele coverage for every sample of our dilution series. We see that on eight out of twenty-four plots, a few of the more extreme observations does not necessarily adhere to the assumption that the coverage is gamma distributed. We have examined these observations in order to find any communality. Table D.1 shows the number of extreme observations for each locus and it would see the extreme observations tend to occur on locus D16 and vWA.

Table D.1: The number of extreme observations for each locus.

CSF1PO	D16S539	D3S1358	D5S818	D7S820	D8S1179	TH01	vWA
0	14	2	4	1	0	6	13

Figure D.2: Gamma QQ-plots of the coverage given the input DNA from the simulated data. The solid line is a straight line through the first and third quartile. The dashed lines are point-wise 95%-confidence envelopes.

In Figure D.2, we see gamma QQ-plots for the simulated coverage, one for each dilution. We can aggregate in this manner as samples of our simulated data will have approximately the same shape, given the dilution. The plots generally show good fits, maybe with the exception of the plot w.r.t. the smallest amount of input DNA.

APPENDIX

E

DILUTION SERIES REFERENCES

Table E.1: The reference profile of the dilution series for contributor F.

Allele	Length	Locus	Zygotic
1	1	AMELX	
1	1	AMELY	
10	40	CSF1PO	Heterozygotic
12	48	CSF1PO	
11	44	D16S539	Heterozygotic
12	48	D16S539	
14	56	D3S1358	Heterozygotic
17	68	D3S1358	
12	48	D5S818	Heterozygotic
9	36	D5S818	
10	40	D7S820	Heterozygotic
8	32	D7S820	
12	48	D8S1179	Heterozygotic
13	52	D8S1179	
7	28	TH01	Heterozygotic
9,3	39	TH01	
8	32	TPOX	Homozygotic
15	60	vWA	Heterozygotic
18	72	vWA	

Table E.2: The reference profile of the dilution series for contributor H.

Allele	Length	Locus	Zygotic
1	1	AMELX	
10	40	CSF1PO	Heterozygotic
9	36	CSF1PO	
9	36	D16S539	Homozygotic
17	68	D3S1358	Heterozygotic
18	72	D3S1358	
10	40	D5S818	Heterozygotic
12	48	D5S818	
10	40	D7S820	Heterozygotic
9	36	D7S820	
13	52	D8S1179	Heterozygotic
15	60	D8S1179	
8	32	TH01	Heterozygotic
9,3	39	TH01	
11	44	TPOX	Heterozygotic
8	32	TPOX	
14	56	vWA	Heterozygotic
17	68	vWA	