Statistical tools for ultra-deep pyrosequecing of fast evolving viruses

David Knowles
Supervisor: Professor Susan Holmes

September 14, 2008

Contents

Chapter 1

Introduction

When an individual becomes infected by a fast evolving virus, such as Human Immunodeficiency Virus (HIV-1) or Hepatitis B (HBV), minor variants rapidly evolve. Although these variants may exist at very low levels, they are hugely important in determining drug resistance. If a minor variant is resistant to the drug that inhibits the primary strain, it will rapidly proliferate under this new selective pressure, and the benefit of the treatment is likely to be mostly lost [1]. As a result, methods to identify minor variants present in an individual are of great interest for directing treatment. A relatively new method is ultra deep pyrosequencing (UDPS) which allows short reads of viral DNA to be sequenced at enormous coverage (currently around 5000x) at reasonable cost [2]. With limiting dilution Sanger sequencing variants present at 20% or above are detectable: with ultradeep pyrosequencing that limit is pushed down to around 1%.

The aim of this project was to statistically characterise the errors involved in 454 ultra deep pyrosequencing (UDPS) of HBV using available plasmid controls, and to use this understanding to design appropriate methods to reliably detect genuine variants in a dataset from 38 individuals with HBV. The processes involved in analysing the data are summarised in Figure 1.1, with the parts I contributed highlighted.

1.1 454 pyrosequencing process

HBV DNA was extracted from the blood plasma of 38 infected individuals and sequenced using 454 pyrosequencing, along with three HBV-1 genomes of known sequence in plasmid vectors. The processes involved are: extraction, limiting dilution Polymerase Chain Reaction (PCR), amplification PCR, dilution, and pyrosequencing.

Extraction. The RNA/DNA is extracted from patient plasma, which might contain around 100,000 copies per ml. After extraction we hope to have an initial copy number of at least 100.

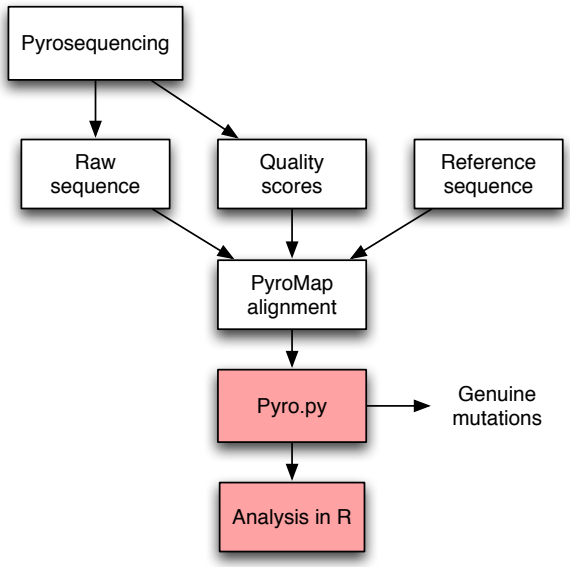


Figure 1.1: Data analysis flow chart.

Limiting dilution PCR. This is used to give a rough estimate of the initial copy number. A 4x serial limiting dilution is done, which just gives an order of magnitude result. The sample is quantitatively diluted until PCR no longer yields product in some batches. Since the number of molecules extracted is Poisson distributed, the proportion with no molecules is $F_0 = e^{-\lambda}$ where λ is the average number of molecules per batch. Thus by estimating F_0 we can estimate λ. Samples with an initial copy number less than 100 were discarded.

Amplification PCR. This is the most troublesome step. PCR enzymes have a very wide fidelity (accuracy of reproduction), of around 2 logs. The main choices currently are:

1. *Taq polymerase.* The original PCR enzyme from the thermophilic bacterium, *Thermus aquaticus*, with an error rate of $\sim 10^{-4}$ [3].

2. *Pfu DNA polymerase.* This enzyme from the thermophilic archaeon *Pyrococcus furiosus* is more stable than Taq at high temperatures and has 3' to 5' exonuclease proofreading activity, which results in a much lower error rate of $\sim 10^{-6}$ [4]. Pfu was used in the group's previous study [5]. However, they found that the yields are typically very poor and the process is slow.

3. *Taq plus error correcting enzyme.* Known as Expand High FidelityPLUS DNA polymerase (Roche Applied Sciences), this combination improves the error rate to around $\sim 10^{-5}$ with acceptable yields and was used for the present study.

For this dataset four slightly overlapping regions ("amplicons") were amplified using eight custom designed primers. An alternative is shotgun sequencing, but this makes alignment more difficult.

Dilution. Following amplification it is necessary to dilute the DNA down so there is just one molecule per bead. Amplification followed by dilution seems wasteful, so in the future it is hoped a more direct method could be developed.

Pyrosequencing. Around 25 to 30 samples are run on one plate. Errors here are less likely to be a problem than in the initial PCR because the PCR in the pyrosequencing process will only result in an observed error if a mismatch occurs in the first round: otherwise the erroneous signal will be hidden by the stronger true signal.

1.2 Analysis issues

Aligning the 454 reads is simplified because the HBV genome is known. The group found that straightforward Smith-Waterman (SW) alignment had problems with the common phenomenon of an insertion closely followed by a deletion (up to around 6bp away), because the score across the small gap is not very significant. The group developped Asymmetric SW [5], which takes into account the quality scores from the pyrosequencer, and helped to alleviate this issue. A further development was the Python based alignment program "Pyromap" which also weights the Sanger sequence. Pyromap is used upstream of my analysis.

The greatest concern is the accuracy of the initial PCR, especially if there are early stage errors: these are the most likely to look like true variants. In the group's previous UDPS study on HIV-1 [5], a Poisson distribution on errors was used in the homopolymeric and non-homopolymeric regions, which was fitted by Expectation Maximisation (EM). However, the increased error rate for the Taq blend seems to result in early errors getting replicated and this results in the error distribution no longer being Poisson.

Chapter 2

Statistical Analysis of Plasmid Controls

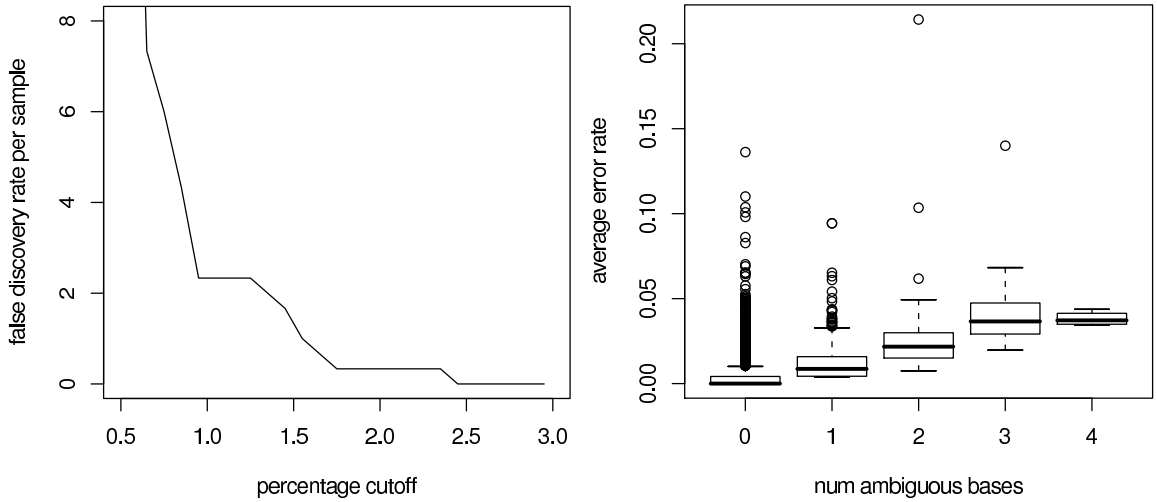
In order to detect which signals in the data represent genuine variants it is necessary to characterise the errors resulting from amplification and pyrosequencing. To facilitate this, three well characterised HBV plasmid vectors were pyro-sequenced using the same experimental method as the patient samples (although the initial copy number was significantly higher, around 100,000). Deviations from the consensus sequence represent either errors occurring during the PCR or the pyrosequencing itself. This data allows us to fit a distribution under the hypothesis that no minor variants are present, which can then be tested. Determining what parametric form this null distribution should take depends on the characteristics of the PCR and sequencing errors, which are analysed in this section. The typical process for this analysis was to develop Python code to manipulate and analyse the aligned read data and output the results in a format easily readable by the statistical programming language R, which was then used for further analysis and plotting.

The level of coverage over the three plasmid controls varies significantly from around 160 to 10,000, with mean just over 3000. Possibly due to primer binding issues, one of the four amplicons is amplified poorly, resulting in low coverage over a significant region.

2.1 Error distribution

In previous studies [5], the criterion used for assigning an observed deviation from the consensus sequence as a genuine mutation was a naive percentage cutoff. If a particular deviation is observed at a given position in more than 1% (for example) of the reads, it is considered a genuine mutation. Thus it is of interest to look at the distribution of these error proportions in the control data, because this gives an empirical estimate of the expected false discovery rate (FDR) for different percentage cutoffs. Figure 2.1(a) shows the empirical FDR across all three controls for varying percentage cutoff. At seven positions across all three controls mismatches occur in more than 1% of reads, so we estimate the FDR per sample is $\frac{7}{3} = 2.3$. Only one position has a mismatch error seen in more than 2% of reads (2.45% actually), so

at this threshold the FDR is reduced to around $\frac{1}{3}$. A simple improvement I made to this basic analysis was to consider mismatches going to different nucleotides separately, which makes a significant difference for the worst errors: for example, what appears to be an error frequency of 4.2% is actually a mismatch to A at 1.7% and to C at 2.4%.



(a) Empirical estimate of FDR versus percentage cutoff.

(b) Error rate against number of ambiguous base calls.

Figure 2.1: Statistical analysis of plasmid control data.

2.2 Homopolymeric regions

454 pyrosequencing is known to be particularly error prone in homopolymeric regions due to carry forward and incomplete extension (CAFIE) errors [2]. Incomplete extension is when the homopolymer is not completed due to insufficient dNTPs. Carry forward errors occur when a nucleotide from the end of a homopolymer is read a few bases later on due to incomplete dNTP flushing. For example, if the true sequence is AAAATCG, it may be read as AAATCGA. We define a homopolymeric region as three or more identical nucleotides and the immediately flanking non-identical nucleotides.

Table 2.1 summarises the error rate for mismatch vs. indels and context. I found a critical bug in the existing code for calculating these error rates that counted a single mismatch error four times. I found that the mismatch error rate is not significantly affected by whether the region is homopolymeric. The increase in the overall error rate in homopolymeric regions is due only to the increase in indel rate (the number of ambiguous base calls, N, is small).

2.3 Quality scores

The quality scores from the pyrosequencing software relate to the probability of CAFIE errors, which is somewhat different to Sanger sequencing *phred* scores. There is a significant

Error rate/10^{-3}	Mismatch	Indel	N	Overall
Homopolymeric	1.125	2.98	1.87	1.811
Non-homopolymeric	1.126	1.76	1.11	1.298
Overall	1.126	2.29	1.32	1.487

Table 2.1: Context specific and non-specific mismatch, indel and overall error rates averaged across the three controls.

correlation ($p < 0.05$) between the mismatch error rate and average quality score, but the effect size is very small: the gradient is -1.2×10^{-5}. I also investigated whether at a given position with mismatch errors the quality score was lower for the specific incorrect base calls, but found no significant effect (results not shown).

2.4 Error distribution over reads

A previous study [6] into the error rates of massively parallel pyrosequencing found that a small number of poor quality reads contained a disproportionate percentage of the total errors. Although this phenomenon seems less severe for our data set, I found the worst 2% of reads still account for 20% of errors. In [6] they also found that reads with lengths outside the main peaks had increased error rate, which I confirmed in our data set. Figure 2.1(b) shows the strong correlation between the error rate and the number of ambiguous base calls in a read. Note that only 2% of reads contain any ambiguous base calls, so discarding these is recommended. I found the correlation is statistically significant ($p < 0.001$) between the error rate and average quality score across a read but the effect size is very small ($\beta = -1.2 \times 10^{-3}$).

2.5 Mismatch rates

We are primarily interested in mismatch errors because indels cause a frameshift which is presumed to make the virus non-functional and therefore not biologically significant. This also suggests that many indels are likely to be PCR or sequencing errors. The mean mismatch error rate is 1.38×10^{-3}, and the maximum mismatch frequency for one position is 4.2×10^{-2}. Table 2.2 shows the mismatch error rates averaged across all three controls, normalised for the relative frequency of each base. This is the conditional probability $P(X|Y)$ where X is the called base (given by the column label) and Y is the true consensus base (given by the row label). As we expect from biochemical constrains, a base is more likely to remain a purine (A or G) or pyrimidine (C or T). For example, $A \rightarrow G$ mismatch errors occur around twenty times more frequently than $A \rightarrow T$ for example.

	A	G	T	C
A	9.99e-01	1.39e-03	7.12e-05	3.39e-05
G	4.53e-04	9.99e-01	3.22e-04	1.87e-05
T	1.69e-04	3.73e-05	9.98e-01	1.54e-03
C	4.14e-04	4.74e-05	4.10e-04	9.99e-01

Table 2.2: Normalised mismatch error rates across controls.

2.6 Error rate along read

Errors occur in 454 pyrosequencing because some proportion of the PCR reactions on a bead get out of sync [2]. We would therefore expect a cumulative effect along the length of a read. To investigate this I plotted error rate against distance from the 5' end of the read, as shown in Figure A.3. There is significant noise, with systematic peaks appearing across all three controls, probably due to homopolymeric regions. However, further investigation showed that this effect is driven almost entirely by the increased indel rather than mismatch rate (results not shown). For our purposes mismatches are more significant so this effect can be ignored.

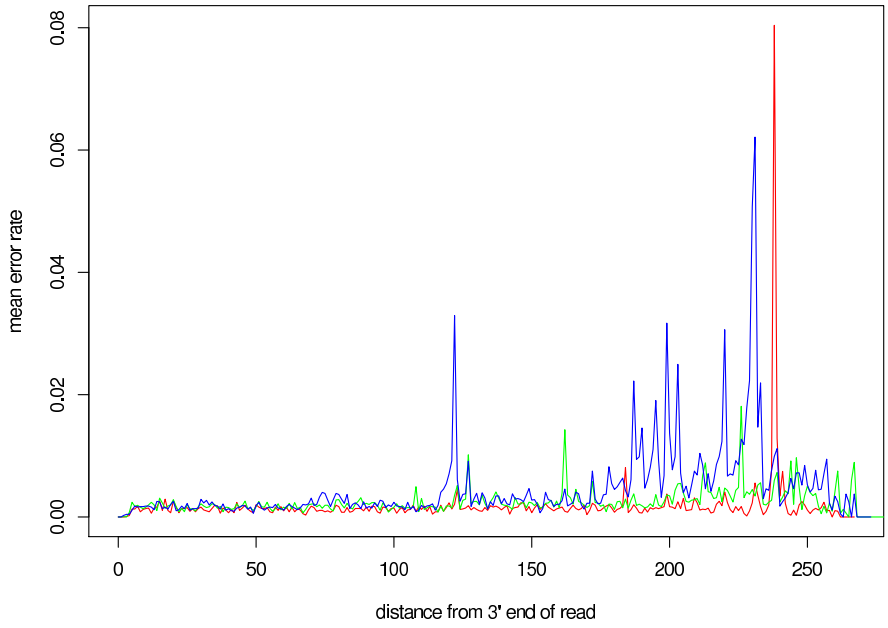
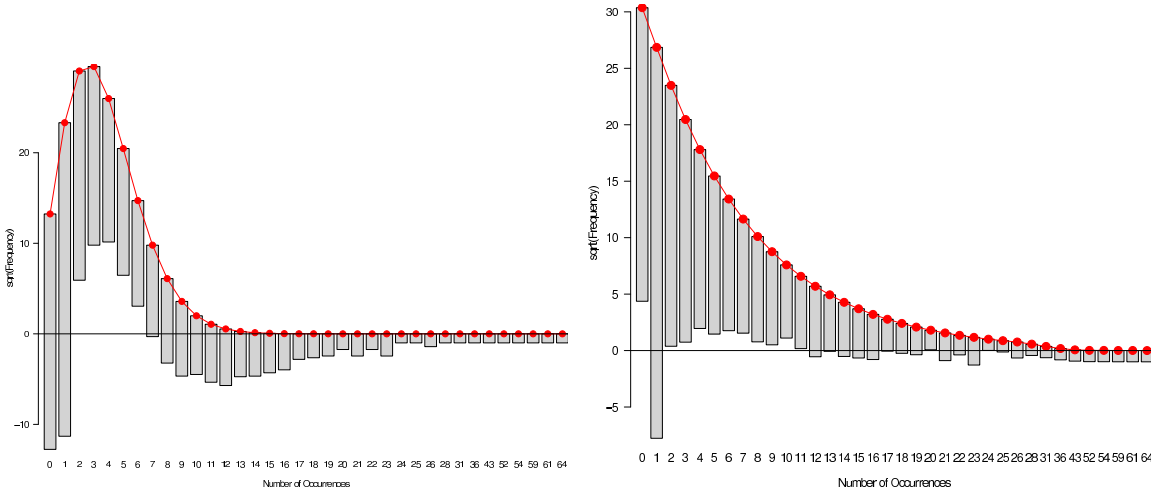


Figure 2.2: Error rate (mismatch and indel) versus distance from 3' end for each control.

2.7 Analytic distributions

A "rootogram" is analogous to a histogram but uses the square root of the frequency to emphasise weight in the tail. A "hanging rootogram" can be used to compare data to a parametric form. The expected frequency under the analytic distribution is plotted, with the rootogram bars "hanging" from it. If the distribution is a good fit the bottom of the bars should line up approximately with the x-axis [7]. Figure 2.2(a) shows a hanging rootogram for a Poisson distribution fitted to the error proportion data, plotted using the `goodfit` function in the R package `vcd`. . The systematic deviation of the bottom of the rootogram bars from the x-axis suggests this model is inappropriate for the data. Figure 2.2(b) shows a hanging rootogram for the negative binomial distribution, a popular choice for overdispersed data [8]. Clearly the fit is much better, although the worst errors are still not accounted for.



(a) Poisson. (b) Negative binomial.

Figure 2.3: Hanging rootograms for Poisson and negative binomial distributions fitted to mismatch error frequency.

Another way of visualising the fit of a discrete distribution are the "Poissonness" plots implemented in the `distplot` function of the `vcd` package [9]. The Poisson distribution is given by:

$$P_\lambda(k) = \frac{e^{-\lambda}k^\lambda}{k!} \text{ for } k \in 0, 1, 2 \ldots \tag{2.1}$$

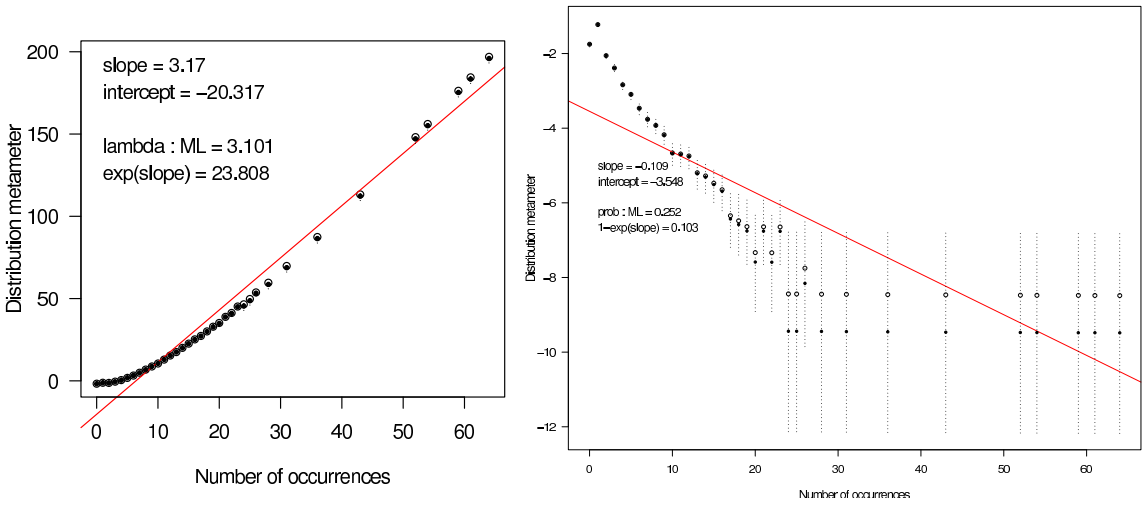
The expected counts are therefore

$$m_k = N\frac{e^{-\lambda}k^\lambda}{k!} \tag{2.2}$$

$$\Rightarrow \log m_k + \log k! = \log N - \lambda + k \log \lambda \tag{2.3}$$

Therefore, or a dataset where the counts x_k equal the expected counts m_k plotting $\log x_k + \log k!$ (known as the "distribution metameter") against k would give a straight line. Analogous metameters are available for the Binomial and Negative binomial distributions.

Figure 2.3(a) shows a Poissonness plot for the total mismatch error rate. The systematic deviation from the straight line shows the Poisson distribution to be inappropriate for the data. Figure 2.3(b) shows the analogous "Negative binomialness" plot. Although the graph is reasonably straight up to around $x = 25$, the model does not cope well with the small number of large proportion errors. It is interesting that different conclusions are drawn from the hanging rootogram vs. distance plots, with the negative binomial looking more appropriate based on the former, and less appropriate with the later. I suspect this is due to how significant the outlying, large proportion errors appear in the two plots.



(a) Poissonness plot.

(b) Negative binomialness plot.

Figure 2.4: Distance plots for Poisson and negative binomial distributions fitted to mismatch error frequency.

2.8 Multinomial regression

The analytic models of the previous section ignored the covariates which are available: the consensus sequence, whether the region is homopolymeric (both categorical variables) and the quality score (a continuous variable). I initially used binomial regression to model the binary outcome: whether or not there is an error (results not shown). The multinomial regression allows which specific error occurs to be modelled. I used the function `multinom` from the package `nnet`, which fits the regression using a neural network and allows counts rather than raw data unlike other packages. The usage in R is shown below:

```
Call:
multinom(formula = cbind(correct, A, G, T, C, N, d, i) ~ consensus +
    as.factor(homo) + qual, data = a)
```

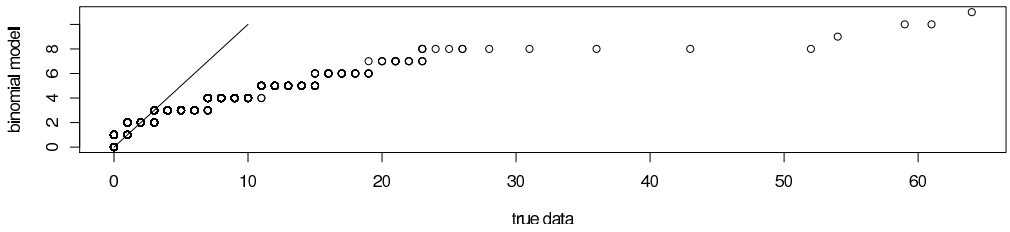
```
  (Intercept) consensusC consensusG consensusT  homo1  homo2 homo3   qual
A       -19.3     11.877     11.994      10.56 -0.314 -0.232 0.346 -0.590
G        -6.1     -3.430    -22.979      -3.63 -0.190 -0.044 0.246 -0.591
T       -10.1      1.738      1.496     -31.83 -0.035  0.404 0.120  0.603
C       -10.3    -26.013     -0.575       3.82 -0.119  0.164 0.017 -0.038
N        -3.5     -0.466      0.049       0.15 -4.445 -3.013 1.005 -7.456
d        -7.7      0.054     -0.724      -0.31  1.353  2.472 1.486 -1.116
i        -4.4     -0.386     -0.153      -0.11 -0.641 -0.205 0.673 -3.561

Residual Deviance: 419028.5
AIC: 419140.5
```

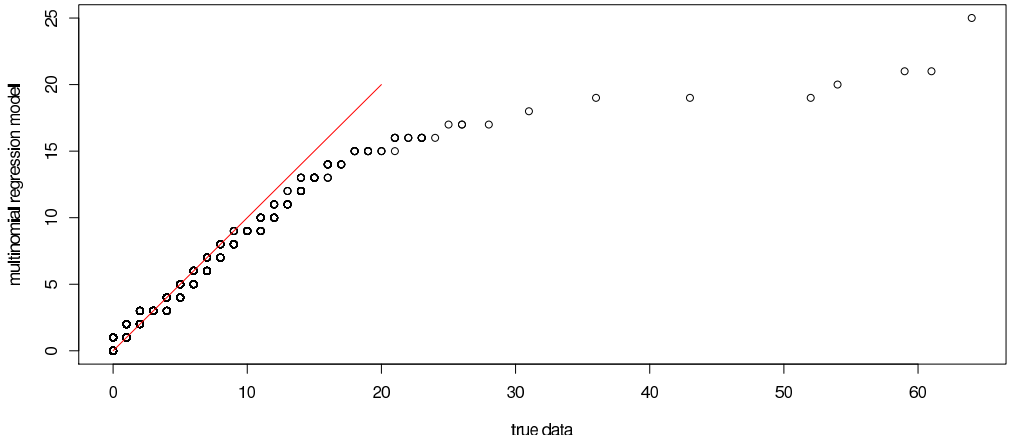
Including the consensus, homopolymeric, and quality covariates all reduce the Akaike Information Criterion (AIC), which implies they are all significant. The table of coefficients has rows corresponding to error types: {A,G,T,C} are mismatches, N is an ambiguous base call, d is a deletion and i is an insertion. Coefficients are positive for a factor that increases the probability of the corresponding error, and negative for those that decrease it. For example, deletion errors are much more likely in homopolymeric regions, hence the coefficient +1.85. Figure 2.4(b) shows a qq plot of data simulated from the multinomial regression model versus the true data, for mismatch errors only (a perfect fit would give a straight line). For comparison, Figure 2.4(a) shows a qq plot of data simulated from a naive binomial model, ignoring all covariates and taking all mismatch errors as equivalent. Clearly the multinomial regression provides a significantly better fit, although the outlying large errors are still not accounted for.

To test the significance of the parameters in the model a bootstrap method can be employed [10]. The probability model is approximated by its empirical distribution: delta functions of mass $\frac{1}{N}$ at each of the observations. The sampling distribution of each parameter is estimated by Monte Carlo by sampling with replacement from the original observations. The corresponding 95%-confidence intervals are shown in Table A.1 in Appendix A.5

Coefficients with confidence intervals which do not include zero are significant at a 95% confidence level. For the mismatch errors the consensus base is significant, as expected from the analysis of mismatch error frequencies in Section 2.5, which showed that the error rates vary depending on the consensus. In Section 2.2 we saw that the mismatch error rate is not increased in homopolymeric regions, and as a result the homopolymeric factor is not significant in determining the mismatch error rate. Interestingly the quality score is also not a significant covariate for the mismatch error rates. Deletion and insertion errors rates are the reverse: the consensus base is not significant, but homopolymeric regions are, as are quality scores for insertion errors.



(a) Binomial.



(b) Multinomial regression.

Figure 2.5: QQ plots of mismatch errors against simulated data.

Chapter 3

PCR Simulation

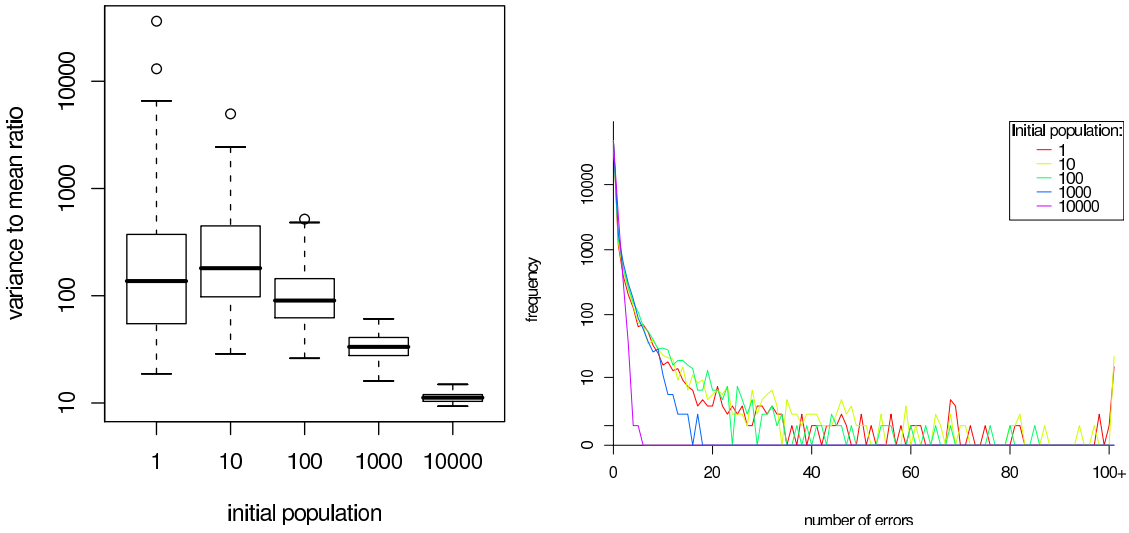
A concern is whether early cycle amplification PCR errors could be amplified to a significant proportion of the population and then resemble a genuine minor variant. This will depend on the initial copy number. Since the plasmid controls had initial copy numbers on the order of 100,000, compared to just 100-1000 for the samples, they cannot be used to answer this question. In order to model the probability of this occuring and the resulting error distribution, I ran computer simulations of the PCR amplification process with a simple binary mutation model. I wrote the simulation algorithm in C for performance reasons.

3.1 Computer simulation

I modelled the PCR amplification process as a simple autocatalytic reaction, using a stochastic simulation algorithm where each step corresponds to a PCR cycle. The reaction is initially exponential as the number of DNA molecules increases, and then becomes limited by the availability of dNTPs. New DNA molecules inherit mutations from their parent molecule, and gain new mutations at random at a specified rate. The simulated molecules are represented as a $N \times M$ binary matrix, where N is the length of the sequence and M is the number of individual molecules. Each column represents a molecule. Although the final population size varies slightly depending on the initial population, it was typically 6000-8000.

The aim of the PCR simulations is to assess the effect of low initial copy number. Figure 3.1(a) shows boxplots of the variance to mean ratio (a measure of overdispersion) for 100 repeats of the simulation, for varying initial copy number. The median is over 100 for initial copy numbers of 1 or 10. The upper range and maximum is considerably higher for an initial copy number of 1 than for 10. The ratio drops significantly as the initial copy number increases, down to around 10 for an initial copy number of 10,000.

Figure 3.1(b) shows histograms of the numbers of errors per position for different initial copy numbers, with the frequency plotted on a $\log(x+1)$ scale. All the distributions have similar frequency of 0, 1 and 2 errors, but the smaller the initial copy number the more density is found in the tail, and the longer the tail is.



(a) Variance to mean ratio.

(b) Histograms.

Figure 3.1: Error distribution from PCR with different initial copy numbers.

3.2 Sparse matrix representation

In my initial implementation the limiting factor in the size of the simulations was the memory requirements. Because stochastic effects are size dependent, it is important for the scale simulation to be as close as possible to the scale of the real reaction. To allow a much larger simulation to be run, a sparse representation of the binary $N \times M$ mutation matrix is required. To do this two arrays, B and C, are defined:

1. Element i of array C gives the row index, i.e. position in the sequence, of the ith mutation. C has length equal to the number of mutations.

2. Element j of array B gives the index in C of the first mutation in column j. B has length $M + 1$.

So $C[B[i], \cdots, B[i+1]-1]$ are the positions of the mutations in individual i, which has $B[i+1] - B[i]$ mutations.

Using this sparse matrix representation it is possible to run considerably larger, more realistic simulations, up to final populations of around 10^9. To mirror the dilution of the PCR product onto beads for pyrosequencing, 5000 "molecules" are sampled at random from this large initial population. This is repeated 100 times to make best use of the information about the final population. Note that the number of repeats was varied depending on the initial copy number n_0. If the error rate is ϵ and there are M positions at which mutations can occur, the expected number of mutations in the first round of replication is $\epsilon n_0 M$, assuming that all the molecules get duplicated. To see the effect of these first round errors, enough repeats need to be run to them. The number of repeats is therefore set to be $\frac{10}{\epsilon n_0 M}$ (rounded up), so that the expected number of first cycle errors observed is 10 (a minimum of 10 repeats is also specified for the larger initial copy numbers).

Figure 3.2 shows the False Discovery Rate per sample against the chosen percentage cut off, assuming only PCR errors, for two different PCR error rates: 10^{-6} on the left and 10^{-5} on the right. To put this in context, recall that previous studies have used a 1% threshold. For an initial copy number of 10, a threshold of 6% is required to make the FDR negligible, whereas for an initial copy number of 100 or more, as required in the experimental samples, a threshold of 1% is sufficient. For an initial copy number of 500 or more even lower thresholds might be possible, but of course the sequencing error must be taken into account as well. The multimodal nature of the low copy number distributions is apparent here manifested as the inflexions in the curve. The cause of the multiple modes is explained in Section 3.3 below: the early discrete cycles result in delta functions at exponentially increasing intervals. These are smoothed by stochastic effects in the simulation.

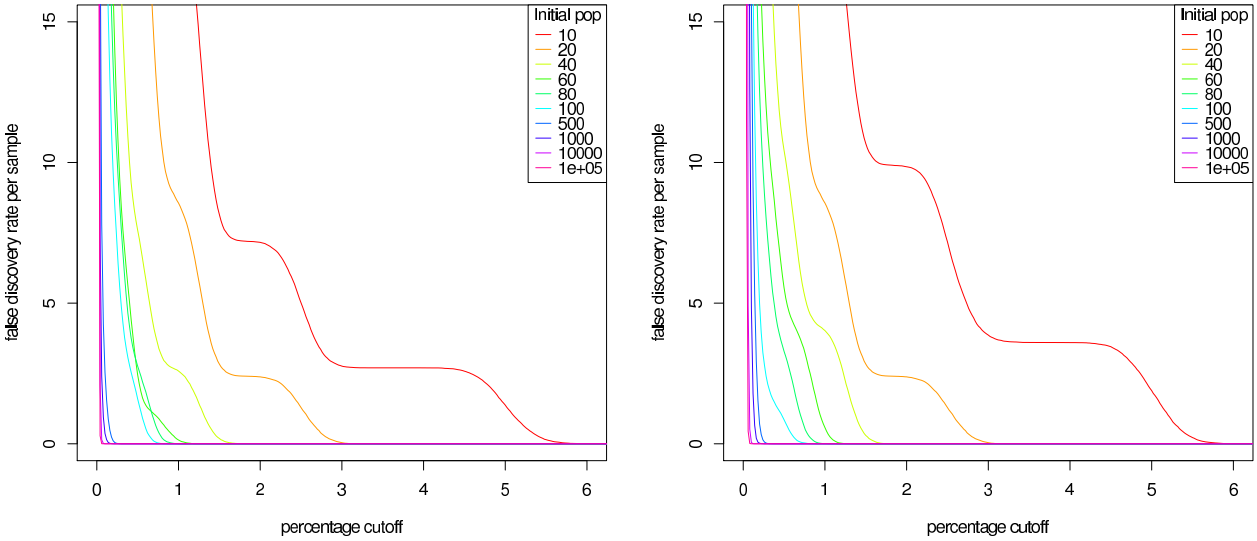


Figure 3.2: False Discovery Rate analysis based on PCR simulations with varying initial copy number. *Left.* Error rate of 10^{-6}. *Right.* Error rate of 10^{-5}.

3.3 Analytic approximations

Although the complexity of the PCR process means an analytic distribution will not exist, it is an interesting thought experiment to consider what happens if the PCR continues exponentially doubling for its entire duration. In this case there will be $n_0 2^{i-1}$ molecules in the ith generation, where n_0 is the initial copy number. If there are N generations there will be $n_0 2^{N-1}$ total molecules. A molecule in the ith generation will have 2^{N-i} daughter molecules. Let the probability of a mutation in the production of any individual is ϵ. Then Binomial$(\epsilon, n_0 2^{i-1})$ mutations will occur in the ith generation, each of which will be transmitted to 2^{N-i} molecules in the final population. Thus the distribution on the number

of times, x, a mutation is duplicated is given by:

$$f(x|N, n_0, \epsilon) = \frac{1}{Z(M, N_0, \epsilon} \sum_{r=2}^{N} \epsilon n_0 2^{r-1} \delta(x - 2^{N-r}) \tag{3.1}$$

Since n_0 and ϵ are just factors we can absorb them into the normalisation constant, which is easily calculated as the sum of a finite geometric series: $Z(N) = 2^N - 2$. The mean is straightforward to calculate:

$$\mathbb{E}x = (N-1)\frac{2^{N-1}}{2^N - 2} \tag{3.2}$$

To calculate the variance we also need $\mathbb{E}[x^2] = 2^{N-2}$. The variance is then given by the standard formula $var(x) = \mathbb{E}[x^2] - (\mathbb{E}x)^2$. The mean, variance and variance to mean ratio for a range of values of N are shown in Figure A.4.

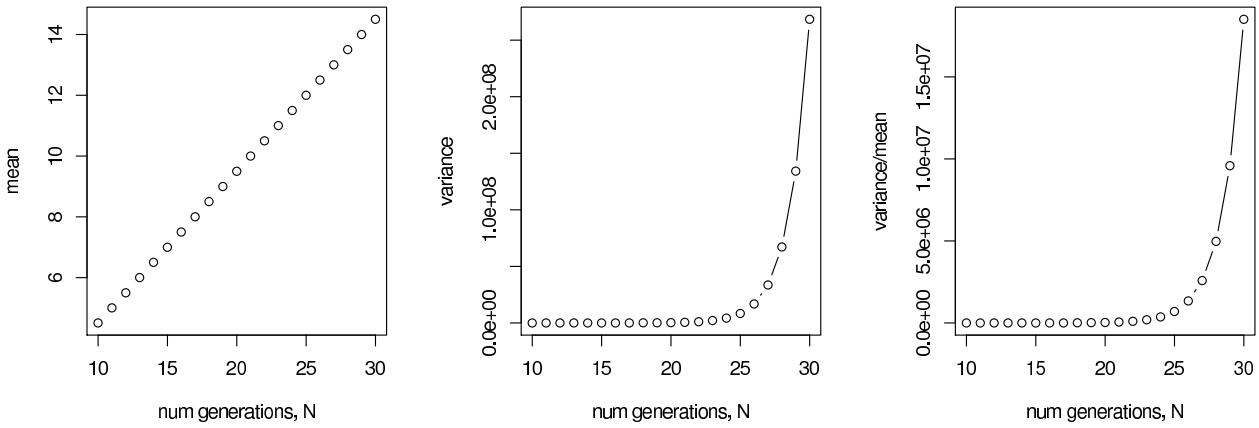


Figure 3.3: Mean, variance and variance over mean ratio for varying number of generations for exponential PCR model.

Chapter 4

Classifying genuine mutations

The simplest method to classify a mismatch as a genuine mutation is to use a percentage cutoff: if the mismatch occurs in more than 1% of reads, for example, classify it as genuine. In this section we develop two more sophisticated methodologies which leverage results from Chapter 2. The first is an essentially frequentist hypothesis testing approach, and the second is a more Bayesian mixture model approach. Since for mismatch error rates we have found that the quality score and homopolymeric state are not significant, we now focus on robust estimation of the 4 by 4 nucleotide transition matrix, $\mathbf{\Theta}$, which will be required for both methods.

4.1 Estimating the nucleotide transition matrix

Element (i, j) of $\mathbf{\Theta}$ gives the probability of observing nucleotide j, given that the true nucleotide was i. A sufficient statistic for $\mathbf{\Theta}$ is the count matrix of nucleotide mismatch errors observed in the controls, $\mathbf{n}$. Element (i, j) of $\mathbf{n}$ is the number of times we observe nucleotide j, when the consensus nucleotide was i. Each row of the $\mathbf{n}$ is a sample from a multinomial distribution with parameters given by the corresponding row of $\mathbf{\Theta}$. Thus the likelihood function for a row i is:

$$P(n_{i.}|\Theta_{i.}) = \prod_j \Theta_{ij}^{n_{ij}} \tag{4.1}$$

So the total likelihood function is

$$P(\mathbf{n}|\mathbf{\Theta}) = \prod_{i,j} \Theta_{ij}^{n_{ij}} \tag{4.2}$$

The maximum likelihood estimate of $\mathbf{\Theta}$ is simply the normalised count matrix:

$$\Theta_{ij}^{\mathrm{ML}} = \frac{n_{ij}}{\sum_k n_{kj}} \tag{4.3}$$

To prevent overfitting it would be better to calculate the posterior distribution of $\mathbf{\Theta}$, given the data. To achieve this we must specify a prior on $\mathbf{\Theta}$. The conjugate prior to the multinomial

distribution is the Dirichlet distibution. Therefore we specify each row i of $\mathbf{\Theta}$ to be drawn from a Dirichlet distribution with parameter vector $\boldsymbol{\alpha}_i$. Thus:

$$P(\Theta_{i.}|\boldsymbol{\alpha}_i) = \frac{\Gamma(\sum_j \alpha_{ij})}{\prod_j \Gamma(\alpha_{ij})} \prod_j \Theta_{ij}^{\alpha_{ij}-1} \tag{4.4}$$

The four vectors form a matrix $\boldsymbol{\alpha}$, whose form is restricted as follows:

$$\alpha_{ij} = \begin{cases} a & \text{if } i = j \\ b & \text{if } i \neq j \end{cases} \tag{4.5}$$

Intuitively we are saying that a prior we do not expect any particular mismatch to be more likely than another, and but the the probability of no mismatch is of course different. An alternative not investigated here would be to have different prior parameters for transition vs. transversion mismatches. We can now express the prior on $\mathbf{\Theta}$ in terms of a and b:

$$P(\mathbf{\Theta}|a,b) = \prod_i P(\Theta_{i.}|a,b) \tag{4.6}$$

$$= \frac{\Gamma(a+3b)^4}{\Gamma(b)^{12}\Gamma(a)^4} \prod_i \Theta_{ii}^{a-1} \prod_{j,j\neq i} \Theta_{ij}^{b-1} \tag{4.7}$$

The joint distribution over the data $\mathcal{D}$ and $\mathbf{\Theta}$ can now be expressed:

$$P(\mathcal{D},\mathbf{\Theta}|a,b) = P(\mathcal{D}|\mathbf{\Theta},a,b)P(\mathbf{\Theta}|a,b) \tag{4.8}$$

$$= \frac{\Gamma(a+3b)^4}{\Gamma(b)^{12}\Gamma(a)^4} \prod_i \Theta_{ii}^{n_{ii}+a-1} \prod_{j,j\neq i} \Theta_{ij}^{n_{ij}+b-1} \tag{4.9}$$

Note that from the form of the posterior it is clear that the maximum a posterior (MAP) estimate of $\mathbf{\Theta}$ is

$$\Theta_{ij}^{\text{MAP}} = \frac{n_{ij}+\alpha_{ij}}{\sum_k (n_{kj}+\alpha_{kj})} \tag{4.10}$$

because $n_{ij}+\alpha_{ij}$ is the effective number of counts of nucleotide i going to j. We can estimate a and b using the evidence framework where we maximise $P(\mathcal{D}|a,b)$, a Type II maximum likelihood method [11].

$$P(\mathcal{D}|a,b) = \int P(\mathcal{D},\mathbf{\Theta}|a,b)d\mathbf{\Theta}$$

$$= \frac{\Gamma(a+3b)^4}{\Gamma(b)^{12}\Gamma(a)^4} \prod_i \int_0^1 \Theta_{ii}^{n_{ii}+a-1} \prod_{j,j\neq i} \Theta_{ij}^{n_{ij}+b-1} d\Theta_i$$

$$= \frac{\Gamma(a+3b)^4}{\Gamma(b)^{12}\Gamma(a)^4} \prod_i \frac{\Gamma(n_{ii}+a)\prod_{j\neq i}\Gamma(n_{ij}+b)}{\Gamma(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b))}$$

It will be easier to maximise the log evidence:

$$\begin{aligned}\log P(\mathcal{D}|a,b) &= 4\log\Gamma(a+3b) - 12\log\Gamma(b) - 4\log\Gamma(a) + \\ &\sum_i \left(\log\Gamma(n_{ii}+a) + \sum_{j\neq i} \log\Gamma(n_{ij}+b) - \log\Gamma(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b)) \right)\end{aligned}$$

To find the minimum of this function with respect to a and b a Newton-Raphson scheme is used, for which the gradient, $\mathbf{g}$, and Hessian matrix, $\mathbf{H}$, are required. To calculate the gradient the digamma function, defined by $\Psi(x) = \frac{d}{dx}\log\Gamma(x)$ is useful. The Hessian will be in terms of the trigamma function $\Psi'(x) = \frac{d\Psi(x)}{dx}$. The details are somewhat laborious, but can be found in the Appendix. The Newton Raphson iteration is:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \lambda \mathbf{H}^{-1}\mathbf{g} \tag{4.11}$$

where $\mathbf{x} = (a,b)^T$ and $\lambda < 1$ is used to help ensure stability.

4.2 Method 1: Hypothesis testing

Once an estimate of the nucleotide transition matrix $\mathbf{\Theta}$ is available the probability of a specific error under the model can be calculated. If we have a position where the consensus nucleotide is i but nucleotide j is observed n_e times out of a coverage of n, then the probability of this occurring is given by the Binomial distribution, since this is the marginal distribution of a multinomial.

$$P(n_{ij} = n_e|\mathbf{\Theta}) = \binom{n}{n_e}\Theta_{ij}^{n_e}(1-\Theta_{ij})^{n-n_e} \tag{4.12}$$

The maximum likelihood or MAP estimate could be used, but it is preferred to integrate over $\mathbf{\Theta}$ using the posterior distribution. Let $\beta_{ij} = n_{ij} + \alpha_{ij}$ be the parameters of the posterior Dirichlet distribution over $\mathbf{\Theta}$. The marginal distribution of Θ_{ij} is then a Beta distribution with parameters $(\beta_{ij}, \beta_{i0} - \beta_{ij})$, where $\beta_{i0} = \sum_j \beta_{ij}$, i.e.

$$P(\Theta_{ij}|\boldsymbol{\beta}) = \frac{1}{B(\beta_{ij}, \beta_{i0}-\beta_{ij})}\Theta_{ij}^{\beta_{ij}-1}(1-\Theta_{ij})^{\beta_{i0}-\beta_{ij}-1} \tag{4.13}$$

where the normalising constant, $B(.,.)$ is the Beta function. To integrate over $\mathbf{\Theta}$:

$$P(n_{ij} = n_e|\boldsymbol{\beta}) = \int P(n_{ij} = n_e|\mathbf{\Theta})P(\Theta_{ij}|\boldsymbol{\beta})d\Theta_{ij} \tag{4.14}$$

$$= \frac{1}{B(\beta_{ij}, \beta_{i0}-\beta_{ij})}\binom{n}{n_e}\int \Theta_{ij}^{n_e+\beta_{ij}-1}(1-\Theta_{ij})^{n-n_e+\beta_{i0}-\beta_{ij}-1}d\Theta_{ij} \tag{4.15}$$

$$= \binom{n}{n_e}\frac{B(n_e+\beta_{ij}, n-n_e+\beta_{i0}-\beta_{ij})}{B(\beta_{ij}, \beta_{i0}-\beta_{ij})} \tag{4.16}$$

Since it is possible to perform this integration analytically there is little additional computational cost compared to using the simple MAP estimate. This expression calculates the

probability of observing n_e mismatches. We classify genuine mutations as mismatches where this probability is lower than some threshold (call this the "likelihood" method), but it is more rigorous to calculate a p-value for each mismatch: the probability of this event, *or any more extreme event*, under the null hypothesis; in this case $P(n_{ij} \geq n_e|\boldsymbol{\beta})$. Since we usually have $n_e \ll n$ it will be cheaper to calculate the p-value as follows:

$$P(n_{ij} \geq n_e|\boldsymbol{\beta}) = 1 - P(n_{ij} < n_e|\boldsymbol{\beta}) \quad (4.17)$$

$$= 1 - \sum_{m=0}^{n_e-1} P(n_{ij} = m|\boldsymbol{\beta}) \quad (4.18)$$

where each term in the sum is evaluated according to Equation 4.16. If the p-value is less than the required significance level then the mismatch is classified as a genuine mutation. Calculating this sum can be quite computationally intensive for large n and moderate n_e, so I used the Python package `ctypes` to interface to compiled, optimised C code to calculate the Binomial coefficients (see Appendix A.7).

4.3 Method 2: A mixture model

The observed mismatches are generated by two processes: PCR/sequencing errors and genuine mutations. A mixture model can be used to represent this, where the mixture proportions correspond to the probability a particular mismatch is a genuine mutation rather than an error. Mixture models can be fitting using the Expectation Maximisation method, which iterates between fitting the mixing proportions and model parameters. Using the controls the error model can be fitted as described in Section 4.1. To model the genuine mutations we will use a codon mismatch matrix, since allows the incorporation three desirable features:

1. Synonymous mutations which do not affect the resulting amino acid are more likely than non-synonymous mutations. Mutations in the third base of a codon are the most likely to be synonymous, leading to the idea of third base "wobble".

2. Non-synonymous mutations which result in amino acids with different physiochemical properties, such as polarity, are less likely because they are more probable to interfere with protein function.

3. Mutations which result in a stop codon are very unlikely to be genuine because the shortened protein will be non-functional.

The reading frame is known for the consensus sequence, so it does not need to be inferred. The number of parameters in the codon model, $64^2 = 4096$, is very large so the risk of overfitting is severe. To avoid this we should use the MAP estimation of the transition matrix, or better still estimate the posterior. The equations are exactly the same as for the nucleotide mismatch matrix, only now the indices are over all codons rather than nucleotides.

Expectation step. This involves calculating the latent variables, which in this case are the mixture proportions. Let m_i be a binary latent variable equal to 1 if codon mismatch i is a genuine mutation, and equal to 0 if it is an error. To calculate the probability that mismatch i is a genuine mutation, π_i, we use Bayes' rule assuming equal priors (i.e. mutation and error are equally likely a prior):

$$\pi_i = P(m_i = 1|\mathcal{D}_i, \Theta_{\text{error}}, \Theta_{\text{mutation}}) = \frac{P(\mathcal{D}_i|m_i = 1, \Theta_{\text{mutation}})}{P(\mathcal{D}_i|m_i = 1, \Theta_{\text{mutation}}) + P(\mathcal{D}_i|m_i = 0, \Theta_{\text{error}})} \tag{4.19}$$

where $\mathcal{D}_i$ is the data associated with mismatch i (i.e. reference and query codon, how many repeats and coverage), and Θ_{mutation} and Θ_{error} are the current estimates of the codon transition matrix for the mutation and error models respectively.

Maximisation step. This step involves updating the Θ_{mutation} using the mixing proportions calculated in the previous step. The count used now are a weighted sum, with the weights given by the mixing proportions.

$$n_{ij} = \begin{cases} \sum_k n_k 1(r_k = i, q_k = j) & \text{if } i = j \\ \sum_k n_k \pi_k 1(r_k = i, q_k = j) & \text{if } i \neq j \end{cases} \tag{4.20}$$

where r_k and q_k are the reference and query codons respectively for mutation k, $1(\cdot)$ is 1 if the statement is true and n_k is the number of times this codon mismatch is observed. Note that for counting the number of times mismatches do not *not* occur for a codon, the mixing proportion is effectively 1 since neither a mutation nor an error has occurred.

4.4 Classification performance

Assessing classification performance is difficult because no ground truth results are available: which mismatches really are genuine mutations? Limiting dilution Sanger sequencing results are available for one of the samples however. This method is not able to detect minor variants at very low levels, so some genuine mutants will not have been detected. Never-theless, this is the closest to ground truth available for comparing the classification methods. The data consists of 95 sequences, which I aligned to the 454 consensus sequence using the `clustalw2` multiple sequence alignment tool [12]. By varying the appropriate thresholds a Receiver Operator Characteristic (ROC) curve (true positive rate against false positive rate, see Appendix A.8 for definitions) can be produced for each method, as shown in Figure 4.1. Better performance is indicated by being closer to the top left corner. Since some genuine mutations will be missing from the "ground truth" set, the number of false positives will be over-estimated and the number of true positives under-estimated, giving a pessimistic view of the performance of these methods. However, the results can still be used to assess the relative performance of the methods.

The more sophisticated methods outperform the percentage cutoff along most of the curve. Somewhat surprisingly, the mixture model performs worse than the p-value and

likelihood methods at most levels. I suspect this may be due to overfitting of the large number of parameters in the mutation model codon mismatch matrix, which for which I used a MAP estimate rather than using the posterior. Given the large number of parameters in the model integrating over the posterior would be prudent. Although the p-value method performs as well as or better than the likelihood method for the important low FPR region to the left of the graph, the likelihood method performs noticeably better when higher FPR is allowed.

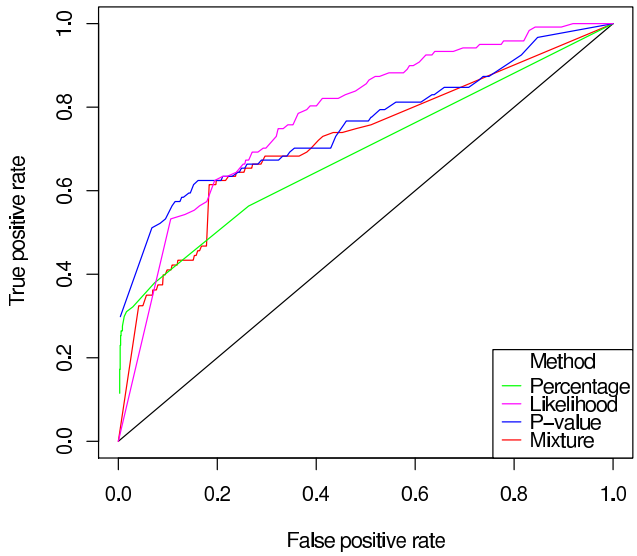


Figure 4.1: ROC curve comparing classification method performance.

Chapter 5

Conclusion

I have statistically characterised the errors inherent in 454 pyrosequencing, and used the results to design methods for detecting genuine variants which outperform the naive threshold method commonly used. The code necessary to run these classification methods is contained in a Python module, `pyro`, which I will make publicly available. I have used computer simulations of the PCR to help understand how initial copy number determines the probability of false positives resulting from early cycle errors. Appendices A.2 and A.3 show some other analyses not directly related to my main project.

There are various avenues of further work I would like to explore. As mentioned in Section 4.4, the somewhat disappointing performance of the mixture model maybe due to overfitting of the large parameter mutation model. To overcome this integration over the posterior of the codon transition matrix should be performed, rather than using a MAP estimate. This would require the `beta` function in Python, which is available as part of the `transcendental` module. A more ambitious aim would be to incorporate more multivariate information into the classification methods. For example, if two mismatches always co-occur, it is highly unlikely they are errors but feasible that they both occur in the same minor variant. A simple way to assess this would be to calculate a co-occurence table of the most frequent errors. A more computationally intensive method would be to attempt to infer the hidden phylogeny of the minor variants. Both these methods are complicated by the fact that the sequences only cover some of the region of interest. Each amplicon would have to be considered separately, but the phylogenies would need to be consistent within each.

Bibliography

[1] Bndicte Roquebert, Isabelle Malet, Marc Wirden, Roland Tubiana, Marc-Antoine Valantin, Anne Simon, Christine Katlama, Gilles Peytavin, Vincent Calvez, and Anne-Genevive Marcelin. Role of hiv-1 minority populations on resistance mutational pattern evolution and susceptibility to protease inhibitors. *AIDS*, 20(2):287–289, Jan 2006.

[2] Marcel Margulies, Michael Egholm, William E Altman, Said Attiya, Joel S Bader, and Jonathan M Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, Sep 2005.

[3] K. A. Eckert and T. A. Kunkel. High fidelity dna synthesis by the thermus aquaticus dna polymerase. *Nucleic Acids Res*, 18(13):3739–3744, Jul 1990.

[4] K. S. Lundberg, D. D. Shoemaker, M. W. Adams, J. M. Short, J. A. Sorge, and E. J. Mathur. High-fidelity amplification using a thermostable dna polymerase isolated from pyrococcus furiosus. *Gene*, 108(1):1–6, Dec 1991.

[5] Chunlin Wang, Yumi Mitsuya, Baback Gharizadeh, Mostafa Ronaghi, and Robert W Shafer. Characterization of mutation spectra with ultra-deep pyrosequencing: application to hiv-1 drug resistance. *Genome Res*, 17(8):1195–1201, Aug 2007.

[6] Susan M Huse, Julie A Huber, Hilary G Morrison, Mitchell L Sogin, and David Mark Welch. Accuracy and quality of massively parallel dna pyrosequencing. *Genome Biol*, 8(7):R143, 2007.

[7] Howard Wainer. The suspended rootogram and other visual displays: An empirical validation. *The American Statistician*, 28(4):143–145, 1974.

[8] Kenneth W. Church and William A. Gale. Poisson mixtures. *Natural Language Engineering*, 1:163–190, 1995.

[9] David C. Hoaglin. A poissonness plot. *The American Statistician*, 34(3):146–149, 1980.

[10] B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

[11] D. J. C. Mackay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1991.

[12] M. A. Larkin, G. Blackshields, N. P. Brown, R. Chenna, P. A. McGettigan, H. McWilliam, F. Valentin, I. M. Wallace, A. Wilm, R. Lopez, J. D. Thompson, T. J. Gibson, and D. G. Higgins. Clustal w and clustal x version 2.0. *Bioinformatics*, 23(21):2947–2948, Nov 2007.

Appendix A

Appendices

A.1 Maximising the evidence

The evidence is given by

$$\begin{aligned}\log P(\mathcal{D}|a,b) &= 4\log\Gamma(a+3b) - 12\log\Gamma(b) - 4\log\Gamma(a) + \\ &\sum_i \left(\log\Gamma(n_{ii}+a) + \sum_{j\neq i}\log\Gamma(n_{ij}+b) - \log\Gamma(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b))\right)\end{aligned}$$

To find the minimum of this function with respect to a and b a Newton-Raphson scheme can be used, for which the gradient and Hessian matrix are required. To calculate the gradient the digamma function, defined by $\Psi(x) = \frac{d}{dx}\log\Gamma(x)$ is useful. The Hessian will be in terms of the trigamma function $\Psi'(x) = \frac{d\Psi(x)}{dx}$.

$$\begin{aligned}\frac{\partial \log P(\mathcal{D}|a,b)}{\partial a} &= -4(\Psi(a) - \Psi(a+3b)) + \sum_i(\Psi(n_i i + a) - \Psi(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b))) \\ \frac{\partial \log P(\mathcal{D}|a,b)}{\partial b} &= -4(3\Psi(b) - 3\Psi(a+3b)) + \sum_i(\sum_{j\neq i}\Psi(n_{ij}+b) - 3\Psi(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b))) \\ \frac{\partial^2 \log P(\mathcal{D}|a,b)}{\partial a^2} &= -4(\Psi'(a) - \Psi'(a+3b)) + \sum_i(\Psi'(n_i i + a) - \Psi'(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b)))) \\ \frac{\partial^2 \log P(\mathcal{D}|a,b)}{\partial b^2} &= -4(3\Psi'(b) - 9\Psi'(a+3b)) + \sum_i(\sum_{j\neq i}\Psi'(n_{ij}+b) - 9\Psi'(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b))) \\ \frac{\partial^2 \log P(\mathcal{D}|a,b)}{\partial a \partial b} &= -4(3\Psi'(b) - 3\Psi'(a+3b)) + \sum_i(-3\Psi'(n_{ii}+a+\sum_{j\neq i}(n_{ij}+b)))\end{aligned}$$

A.2 How much control sequence?

An important question for experimental design is how much control sequence is needed to quantify the error rate accurately. Figure A.1 shows 95% confidence intervals on the nucleotide transition matrix parameters versus the number of bases observed. The graph was generated by simulating mismatch error counts with the same proportions as the existing plasmid controls but with different magnitudes, and calculating corresponding CIs. Note that across all three plasmid controls we have on the order of 10^7 observed bases, which seems suitable based on Figure A.1.

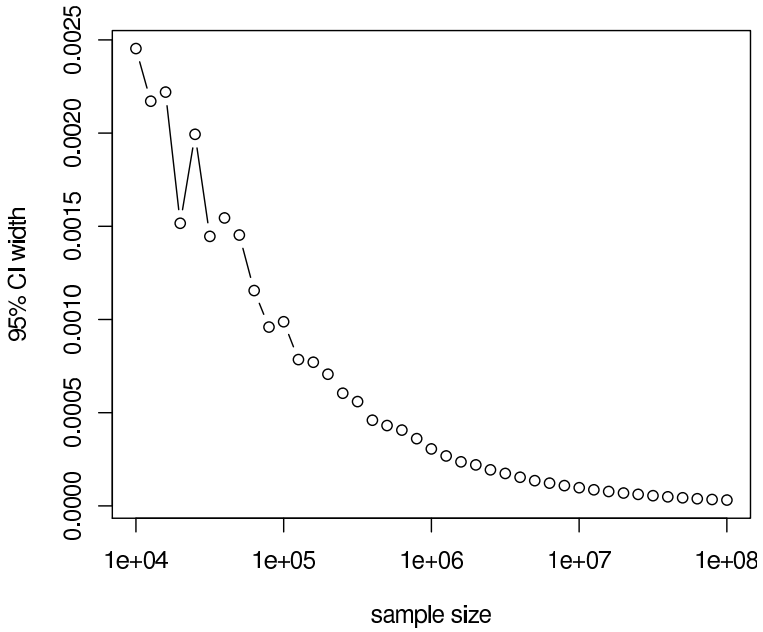


Figure A.1: 95% confidence intervals on transition matrix parameters versus number of bases observed.

A.3 Viral load

As an offshoot of the main project I was asked to look at the relationship between the viral load and the number of non-synonymous (i.e. biologically significant) mixtures in 1186 infected individuals.

A.4 Multinomial regression

Table A.1 shows 95% confidence intervals for the coefficients of the multinomial regression calculated using a non-parametric bootstrap.

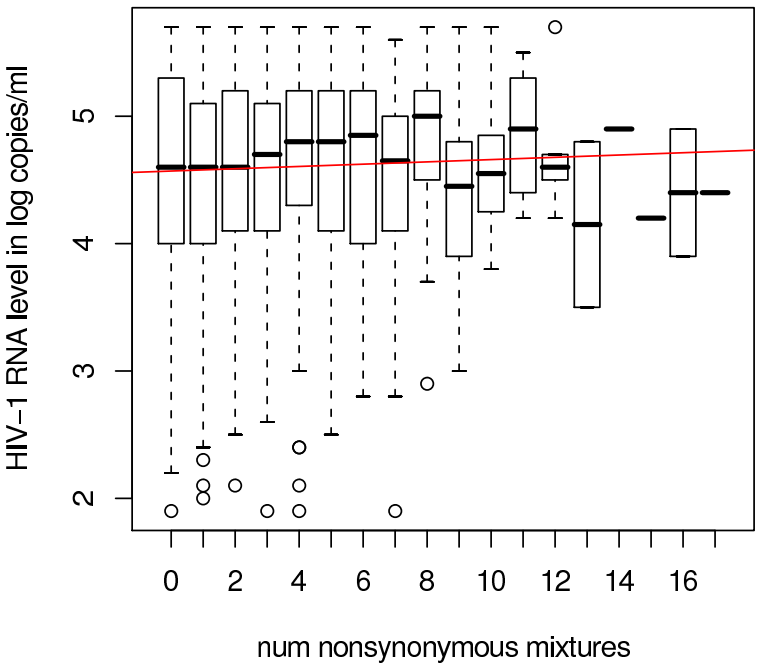


Figure A.2: Viral load versus the number of non-synonymous mixtures.

	(Intercept)	consensusC	consensusG	consensusT	homo1	homo2	homo3	qual
A	-263, -17	10, 257	10, 257	8, 256	-1, 0.003	-2, 0.2	-0.2, 1	-5, 0.2
G	-7, -6	-30, -3	-907, -21	-105, -3	-0.5, 0.1	-0.3, 0.2	-0.1, 0.7	-1, 0.07
T	-136, -9	1, 128	1, 127	-1258, -29	-1, 0.3	-1, 0.8	-0.4, 0.6	-2, 2
C	-423, -10	-1042, -24	-274, 0.3	3, 422	-0.4, 0.1	-0.3, 0.3	-0.2, 0.3	-2, 0.6
N	-9, 9	-2, 0.6	-1, 1	-0.8, 2	-318, -3	-14, 0.5	-0.3, 2	-26, -0.4
d	-10, -5	-1, 1	-1, 0.1	-1, 0.7	-0.2, 2	1, 3	0.05, 2	-4, 0.6
i	-6, -2	-1, 0.3	-0.8, 0.6	-0.8, 0.6	-2, -0.06	-1, 0.6	0.04, 1	-6, -2

Table A.1: Confidence intervals for multinomial regression.

A.5 Calculating a Binomial p-value, fast

To solve a computational bottleneck in calculating p-values under a Binomial distribution with large n, I developed the following function, which calculates terms in the sum successively and handles multiplications as additions in log space to greatly improve performance.

```
double binomialPvalue(int n,int r,double p){
if (p==0.0)
return 0.0;
double logPoverOneMinusP=log(p/(1.0-p));
double y=n*log(1.0-p),sum=exp(y);
int i;
for (i=1;i<r;i++){
y+=log((double)(n-i+1))-log((double)(i))+logPoverOneMinusP;
sum+=exp(y);
}
return 1.0-sum;
}
```

A.6 ROC curves

For a particular classification method and threshold a confusion matrix can be calculated and used to find the true positive rate ($TPR = \frac{TP}{TP+FN}$) and false positive rate ($FPR = \frac{FP}{FP+TN}$), where

- TP is the true positive count: the number of mismatches classified as genuine mutations *and* found in the limiting dilution sequences

  TN is the true negative count: the number of mismatches classified as errors *and* not found in the limiting dilution sequences

- FN is the false negative count: the number of mismatches classified as errors *but* found in the limiting dilution sequences

- FP is the false positive count: the number of mismatches classified as genuine mutations *but* not found in the limiting dilution sequences