# ReQON (Tutorial)

***Christopher R. Cabanski***

**October 29, 2019**

## Contents



## 1 Summary

*ReQON* is a tool for recalibrating the nucleotide quality scores for aligned sequence data in BAM format. This document provides a tutorial of how to use *ReQON*. **THERE HAVE BEEN SEVERAL CRITICAL UPDATES TO THE ReQON PACKAGE. MAKE SURE THAT YOU HAVE DOWNLOADED AND ARE RUNNING ReQON VERSION 1.3.7 OR GREATER TO ACHIEVE ALL FUNCTIONALITY DESCRIBED IN THIS TUTORIAL.**

## 2 Introduction

Next-generation sequencing technologies have become important tools for genome-wide studies. Each nucleotide that is called is reported with a quality score, which represents the probabilty of a sequencing error. These quality scores are often reported on a log-transformed Phred scale [1]. Unfortunately, these reported quality scores do not truly reflect the probability of a sequencing error ([2] and Subsection 4.3).

*ReQON* (Recalibrating Quality Of Nucleotides) is a tool for recalibrating the nucleotide quality scores to more accurately reflect sequencing error probabilities. *ReQON* uses logistic regression to recalibrate the quality scores for a supplied BAM file with aligned sequence data that has been sorted and indexed. Output includes a new BAM file with the original quality scores replaced with the new recalibrated scores, along with diagnostic plots which show the effectiveness of the recalibration on the training set and a data object of the plotted diagnostic data.

# 3 Tutorial

The input file to `ReQON` must be a BAM file of either single-end or paired-end sequencing data that has been aligned using any alignment tool. This BAM file must be sorted, and the corresponding index file must be present in the same directory. For help with sorting and indexing BAM files, we recommend using functions `sortBam` and `indexBam` from package *Rsamtools*.

For this tutorial, we will recalibrate a BAM file that is included with the *seqbias* package, which uses data taken from Mortazavi et al. [3] (NCBI accession number SRR001358).

```
> library( ReQON )
> library( seqbias )
> library( Rsamtools )
> reads_fn <- system.file( "extra/example.bam", package = "seqbias" )
> # sort BAM
> sorted <- sortBam( reads_fn, tempfile() )
> # index BAM
> indexBam( sorted )
```

When recalibrating a BAM file, users are given the option of removing known SNPs from the training set. This file of known SNP locations can be either a text or Rdata file (with variable name `snp`) with no header and two columns: (1) chromosome number and (2) position. For this example, we will not remove any known SNP positions from our training set.

Users are also given the option of supplying a file of reference sequence for the training set to help identify sequencing errors. This file can be either a text or Rdata file (variable name `ref`) with no header and 3 columns: (1) chromosome, (2) position, (3) nucleotide (A,C,G,T). Computations will speed up if the RefSeq file only covers the positions of the training region. If a reference file is not supplied, then sequencing errors are identified as nucleotides not matching the major allele(s) for coverage $> 2$. We will construct a file of reference sequence data for our example from a reference sequence FASTA file provided in the *seqbias* package. We will save the reference sequence text file as '`ref\_seq.txt`'.

```
> ref_fn <- system.file( "extra/example.fa", package = "seqbias" )
> ref_f <- FaFile( ref_fn )
> open.FaFile( ref_f )
> seqs <- scanFa( ref_f )
> len <- length( seqs[[1]] )
> ref <- matrix( nrow = len, ncol = 3 )
> # chromosome/sequence name in column 1
> ref[,1] <- rep( "seq1", len )
> # sequence position in column 2
```

```
> ref[,2] <- c( 1:len )
> # reference base in column 3
> str <- toString( subseq( seqs[[1]], 1, len ) )
> s <- strsplit( str, NULL )
> ref[,3] <- s[[1]]
> # look at reference file
> ref[1:5,]
     [,1]   [,2] [,3]
[1,] "seq1" "1"  "A"
[2,] "seq1" "2"  "A"
[3,] "seq1" "3"  "A"
[4,] "seq1" "4"  "G"
[5,] "seq1" "5"  "T"
> # save reference file
> write.table( ref, file = "ref_seq.txt", sep = "\t", quote = FALSE,
+    row.names = FALSE, col.names = FALSE )
```

When specifying SNP or RefSeq files, there are available R packages that provide much of the needed information. For human, see *SNPlocs.Hsapiens.dbSNP.20101109* for dbSNP version 132 positions and *BSgenome.Hsapiens.UCSC.hg19* for hg19 reference sequence.

It is possible when working with complex genomes with a lot of variability that some bases in the training set identified as sequencing errors are actually correct. For example, this may be a novel variant or a systematic mapping error. In an attempt to remedy this issue, we set two different thresholds to remove positions from the training set most likely to contain incorrect error calls. First, we remove any positions with a non-reference (error) allele frequency greater than `nraf` (default = 0.05). Second, if the coverage at a position times `nraf` is less than `nerr`, we remove positions with more than `nerr` called errors (default = 2). These thresholds set a maximum number of allowable called errors for positions with low coverage (`nerr`) and a maximum frequency of non-reference/error bases for positions with high coverage (`nraf`).

Since this example BAM file is very small, we will train the model on the entire data set. For large data sets containing hundreds of millions of reads, which are becoming increasingly common, we suggest training the model on at least 10 million bases (`max_train` > 10,000,000). In practice, we have obtained very accurate results by training on approximately 25 million bases.

For this tutorial, we would like to train on the entire BAM file (`region` = seq1:1-len), supply a reference sequence (`RefSeq`), set error thresholds (`nerr` = 20, `nraf` = 0.25) and plot the results (`plotname`). We will save the diagnostic data output as `diagnostics`.

```
> # set training region
> reg <- paste( "seq1:1-", len, sep = "" )
> diagnostics <- ReQON( sorted, "Recalibrated_example.bam", region = reg,
+    RefSeq = "ref_seq.txt", nerr = 20, nraf = 0.25,
+    plotname = "Recalibrated_example_plots.pdf" )
> # delete reference sequence file
> unlink( "ref_seq.txt" )
```

The output of `ReQON` is a BAM file that replaces the original quality scores in the QUAL field of the input BAM file with the recalibrated Phred-scaled (Phred+33, also called "Sanger") quality scores.

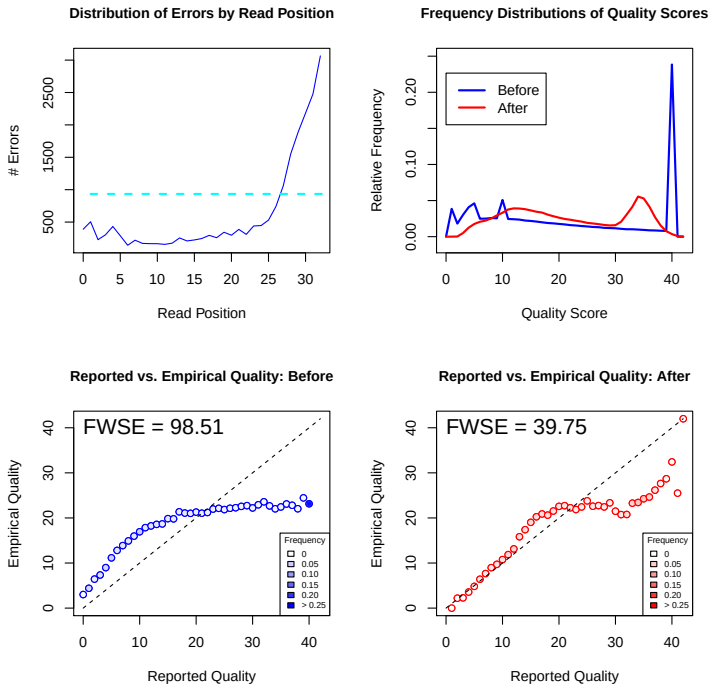


**Figure 1:** Graphical output of `ReQON`

# 4 Description of Diagnostic Plots

In addition to the recalibrated BAM file, the output of `ReQON` consists of diagnostic data and plots. The data is output as an R object and the plots are saved as a pdf file if the plotname option is set. Figure 1 shows the graphical output produced by `ReQON` after recalibrating the example data. The data used in the diagnostic plots and provided as output come from the training set and therefore should not be treated as an overall quality assessment for the entire BAM file, especially when the training set is small. These diagnostic plots are provided as evidence that recalibration is needed for the quality scores to accurately reflect the probability of a sequencing error. The following subsections describe and interpret the four diagnostic plots that are produced.

## 4.1 Distribution of Errors by Read Position

The top left plot shows the distribution of sequencing errors by read position. Generally, this line is very high at one or both ends of the plot, which corresponds to sequencing accuracy decreasing at the beginning/end of reads. Occasionally, there are additional spikes in this line, which may be due to issues with the sequencer at a specific cycle. Our recalibration algorithm incorporates the information about which read positions have many errors by adding additional variables to the model for all read positions that are above a threshold (the dashed cyan line at 1.5 times the average number of errors per read position). For this example, the majority of the sequencing errors occur after read position 25.

This plot can also be produced using the vector `$ReadPosErrors` and the plotting function `ReadPosErrorPlot`. We will set the `startpos` option to 0 because we want the read positions to begin at 0 (as opposed to 1).

```
> # show the error counts by read position
> diagnostics$ReadPosErrors

 [1]  390  505  228  303  432  289  142  220  171  167  166  156  174  255  209
[16]  224  246  295  259  343  297  387  310  440  446  530  740 1062 1552 1889
[31] 2182 2477 3065

> # plot
> ReadPosErrorPlot( diagnostics$ReadPosErrors, startpos = 0 )
```

## 4.2 Frequency Distribution of Quality Scores

The top right plot shows relative frequency distributions of the quality scores both before (blue) and after (red) recalibration. Notice that the recalibrated quality scores tend to have a larger spread than the original scores. In this specific example, we can see that 25% of the original quality scores had a value of 40, corresponding to an error rate of 1 in 10,000 nucleotides. After recalibration, the largest peak is around 35 (corresponding to an error rate of approximately 3 in 10,000), with less than 10% of the nucleotides assigned this value.

This plot can also be produced using the vectors `$QualFreqBefore` and `$QualFreqAfter` and the plotting function `QualFreqPlot`.

```
> # plot
> QualFreqPlot( diagnostics$QualFreqBefore, diagnostics$QualFreqAfter )
```

## 4.3 Reported vs. Empirical Quality

The bottom two plots show the reported quality score on the x-axis and the empirical quality score on the y-axis. To calculate the empirical quality, the error rate was calculated for all nucleotides with a given quality score then converted to the Phred-scale. If the reported quality scores are accurate, then all points should fall on the $45^\circ$ line. After recalibration, we see that the points are much closer to the $45^\circ$ line.

To get a better sense of how closely the points are to the $45^\circ$ line, each plot also reports the Frequency-Weighted Squared Error (FWSE). FWSE is calculated by squaring the vertical distance between each point and the $45^\circ$ line (the error), weighting this squared error by the relative frequency (shown in the top right plot), then summing over all points. To help visualize which points have a small relative frequency, and thus do not have a large contribution to FWSE, points are shaded according to their relative frequency. If the quality scores accurately reflect the probability of a sequencing error, then FWSE should be close to zero.

The before and after recalibration FWSE scores can be compared and we see that for this example, the quality scores more accurately reflect the true sequencing error rate after recalibration, reflected by the much smaller FWSE score. Remember that this example data set was very small ($<$ 500,000 nucleotides). Even with this small training size, we see a signifi-

cant decrease in FWSE. If we had a larger training set, we would expect FWSE to decrease even more. In practice, when recalibrating large BAM files and training on approximately 25 million bases, we commonly see FWSE decrease by over 90

This plot can also be produced using the vectors `$ErrRateBefore` and `$ErrRateAfter` and the plotting function `FWSEplot`.

```
> # report FWSE from ReQON output
> diagnostics$FWSE

[1] 98.51158 39.74702

> # plot before recalibration and report FWSE
> f1 <- FWSEplot(diagnostics$ErrRatesBefore, diagnostics$QualFreqBefore,
+    col = "blue", main_title = "Reported vs. Empirical Quality: Before")
> f1

[1] 98.51158

> # plot after recalibration and report FWSE
> f2 <- FWSEplot(diagnostics$ErrRatesAfter, diagnostics$QualFreqAfter,
+    col = "red", main_title = "Reported vs. Empirical Quality: After")
> f2

[1] 39.74702
```

# 5 Additional options for `ReQON` function

- **max_train** The maximum number of nucleotides to include in the training region. The default is to use all nucleotides from the training region. This option is useful if you want to train on e.g. the first 5 million nucleotides of chromosome 10.
- **temp_files** Option for keeping temporary files. 0 (default) = remove temporary files. 1 = keep temporary files.

# 6 Troubleshooting

This section descibes issues that may arise when running *ReQON* and how to solve these issues.

1. **Error in ReQON. Unused argument(s) (nerr = 2, nraf = 0.05).** You are most likely running an out-of-date version of ReQON. Parameters `nerr` and `nraf` were added in version 1.3.0. We strongly recommend using version 1.3.7 or later as some earlier verions are not fully stable. To check the version of ReQON installed on your computer, use the R command `citation("ReQON")`. The lateststst version of ReQON can be downloaded at http://bioconductor.org/packages/devel/bioc/html/ReQON.html.
2. **WARNING: This BAM's header does not say it is sorted (perhaps it was sorted by samtools?)** This error occurs if either your BAM file is not sorted, or it was sorted using *SAMTools*. If it was sorted using *SAMTools*, you can ignore this error, as this

warning message will not affect the results. However, if your BAM file is not sorted, you should ignore the results and re-run `ReQON` after sorting and re-indexing the BAM file.

3. **net.sf.samtools.SAMException: No index is available for this BAM file.** This error occurs when the corresponding index file (.bai file) is not in the same directory as the BAM file.

4. **java.lang.RuntimeException: SAM validation error: ERROR: Record XXX, Read name XXX, Mapped mate should have mate reference name.** This error occurs when trying to recalibrate paired-end reads and the two mapped mates have different reference names. The easiest solution is to run "`samtools fixmate <in.nameSrt.bam> <out.bam>`", then re-index the file before recalibrating. For more information, see the *SAMTools* manual [4].

5. **SAM validation error: ERROR: Record XXX, Read name XXX, CIGAR should have zero elements for unmapped read.** This error occurs when the CIGAR string has more than zero elements for an unmapped read. There are only few specific aligners that produce this error when aligning under certain conditions. Currently, there is no simple fix for this error. We recommend that you realign the reads using a different aligner such as *Bowtie* [5] or *MapSplice* [6].

# 7 Manuscript

For a more in depth discussion of quality score recalibration, see our manuscript describing the *ReQON* algorithm in further detail [7]. This paper outlines the specific model used, how recalibration improves quality score accuracy and the ability to detect true variants, and a comparison with other recalibration algorithms. Please cite this manuscript if you use *ReQON*.

# References

[1] Ewing, B., Green, P. (1998) Base-calling of automated sequencer traces using phred II. Genome Research 8, 186−194.

[2] Li, R., Li, Y., Fang, X., et al. (2009) SNP detection for massively parallel whole-genome resequencing. Genome Research 19, 1124−1132.

[3] Mortazavi, A., Williams, B., McCue, K., et al. (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nature Methods, 5, 621−628.

[4] Li, H., Handsaker, B., Wysoker, A. (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078−2079.

[5] Langmead, B., Trapnell, C., Popm M., et al. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biology 10, R25.

[6] Wang, K., Singh, D., Zeng, Z., et al. (2010) MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery. Nucleic Acids Research 38, e178.

[7] Cabanski, C., Cavin, K., Bizon, C., et al. (2012) ReQON: a Bioconductor package for recalibrating quality scores from next-generation sequencing data. BMC Bioinformatics 13(221). doi:10.1186/1471-2105-13-221.