MiXCR QC Analysis

Wes Horton

May 6, 2016

Objective

We need to determine if we can use MiXCR clonotype count outputs as a proxy for depth of coverage. To do so, we need to figure out how reads are aligned and assembled. Are they grouped together too often for our purposes, what is the reasoning behind the grouping, what happens if we change certain parameters, etc.

MiXCR Summary

Alignment

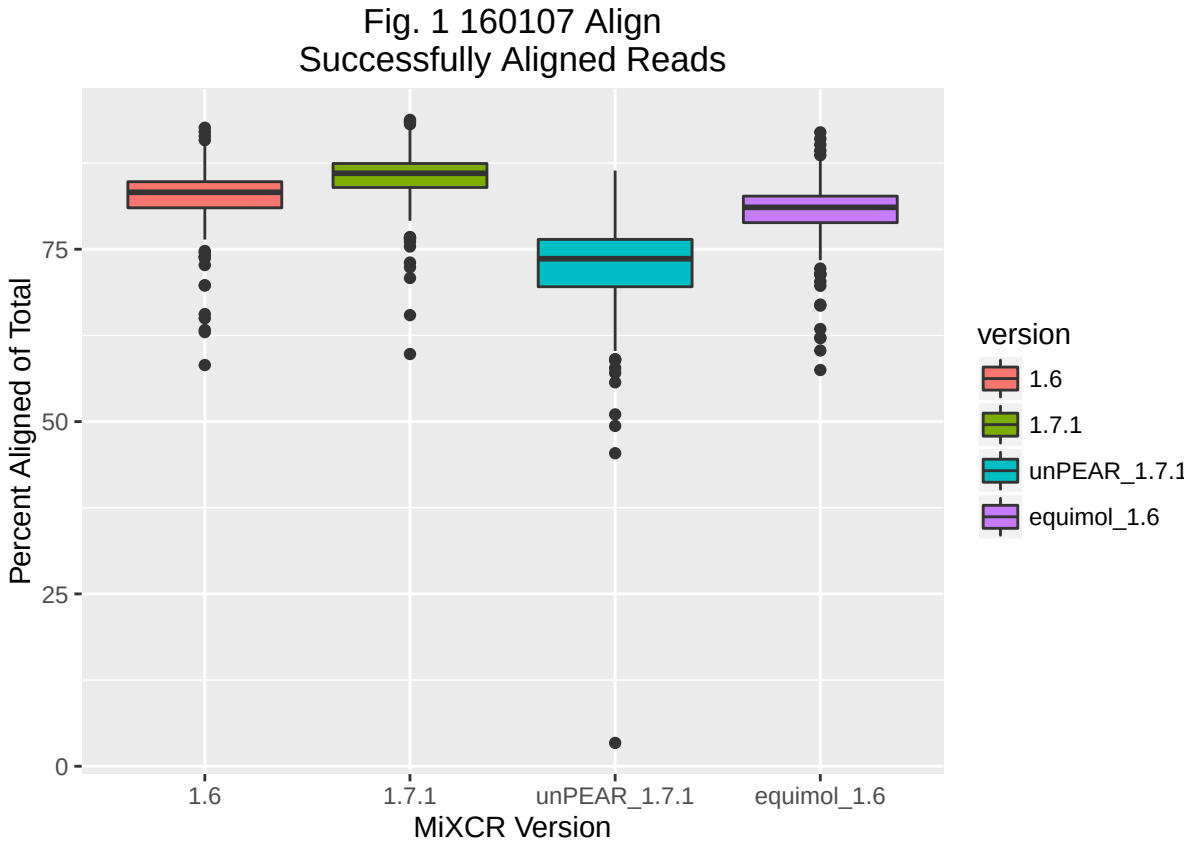
The first command in the MiXCR pipeline is the alignment step. In this step, sequencing reads are aligned to reference (GenBank) V, D, J and C genes of T-cell receptors. We specify the genes of the TRB (T-Cell Receptor Beta chain) locus in our analysis. We use the **default regions**.

Our analysis pipeline collects the report generated by each align run into a QC summary table. This table contains the following notable columns:

1. Total Reads (in original file)
2. Aligned Reads (number of reads aligned to reference genes)
3. Aligned Percent (reads aligned to reference genes as percent of total reads)
4. Alignment failed because of absence of V hits (as percent of total reads)
5. Alignment failed because of absence of J hits (as percent of total reads)
6. Alignment failed because of low quality score (as percent of total reads)

Below are the results taken from a few different MiXCR runs using the batch 160107LC.

First, lets look at a boxplot of the percentages of aligned reads for each sample as well as a summary of the distribution:



The first three use the equivolume sequencing run. 1.6 refers to mixcr version 1.6, 1.7.1 is version 1.7.1, unPEAR_1.7.1 uses unmerged paired-end reads and version 1.7.1, and equimol_1.6 is equimolar sequencing run using version 1.6. (Equimolar refers to sample prep during Illumina sequencing. Samples are normalized by concentration, rather than taking an equal volume of each sample, regardless of concentration. We use equivolume to preserve biological differences in T-cell concentration between samples.)

Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   58.20   81.01   83.26   82.39   84.80   92.63
```

Version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   59.80   83.96   86.02   85.19   87.44   93.77
```

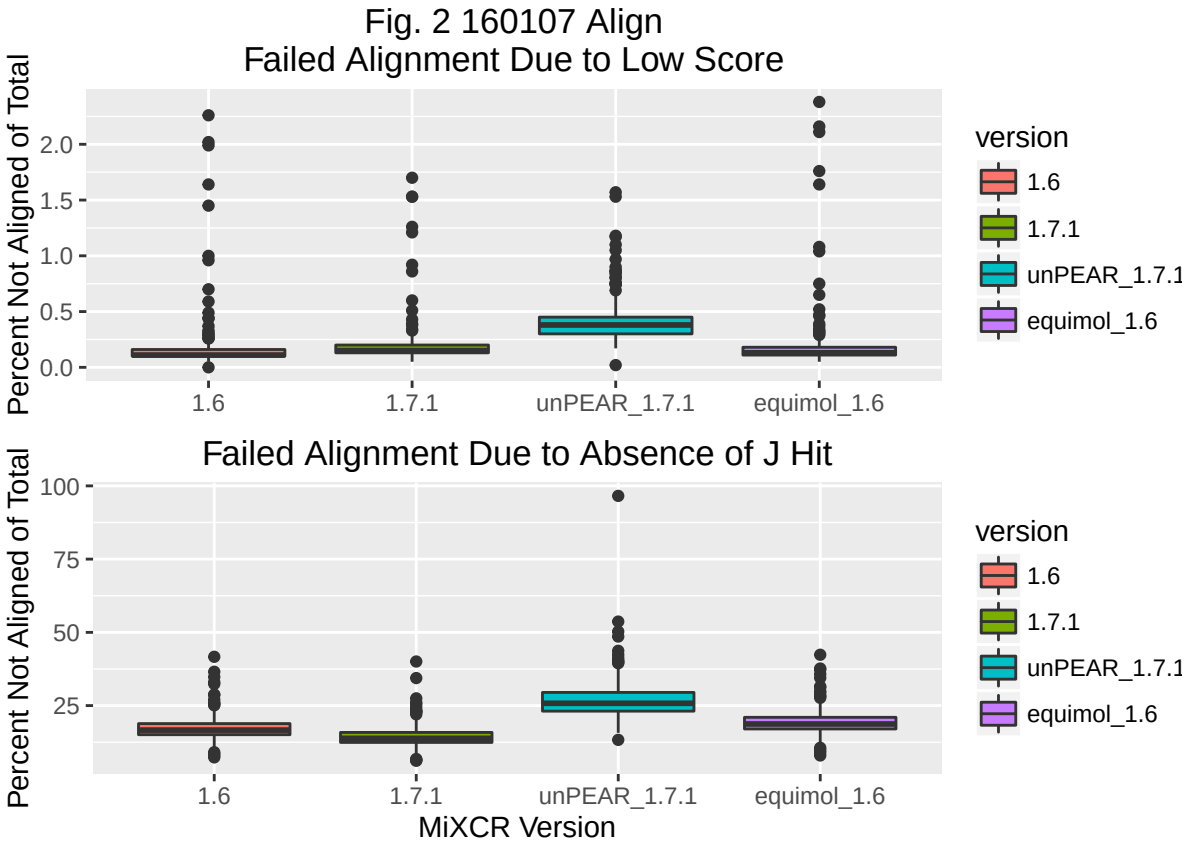
Unassembled reads, version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.38   69.56   73.63   71.91   76.43   86.41
```

Equimolar sequencing, version 1.6

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   57.48   78.86   81.08   80.32   82.71   91.96
```

We see that most of the samples aligned greater than 80% of their reads, but a few have relatively poor alignments, with one extreme outlier in the unassembled reads run. We can look at a few of the explanatory columns to see why the rest aren't aligning.



Failed Due to Low Score:

Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.1000  0.1100  0.2017  0.1600  2.2600
```

Version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Unassembled reads, version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0500  0.1100  0.1300  0.2225  0.1800  2.3800
```

Failed Due to No J Hit:

Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.33   15.07   16.64   17.41   18.86   41.66
```

Version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Unassembled reads, version 1.7.1

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.98   17.03   18.72   19.46   21.00   42.36
```

Looks like most reads are not aligning due to a lack of J hit. This is a little concerning. In theory, all reads assembled by PEAR should have both V and J regions. This is because we use a V primer as our forward primer and a J as our reverse. After PEAR assembly, they should be on either end of the same read.

Does this mean we have off-target amplification? Are V primers binding where we should have J binding? Is PEAR mis-assembling our reads?

Should we relax the parameters for calling a hit?

Moving Forward

1. Perform primer specificity experiment - amplify synthetic templates with 1 V primer and all J primers, repeating for each V primer. Repeat process with 1 J primer and all V primers.
    - See figure 3 of Carlson et al. 2013 - "Using synthetic templates to design an unbiased multiplex PCR assay"
2. Extract all unaligned reads, and re-run them through MiXCR align using relaxed parameters.
    - use -a to save the description line from fastq file.
    - Export alignments with descriptions
    - remove all aligned reads from fastq file
    - re-run remaining reads with relaxed parameters
    - See if we align more of remaining reads

Assemble

After reads are aligned to reference genes, the assemble command extracts specific gene regions (CDR3 in our case) and builds a set of clones.

1. Assembler extracts clonal sequence (CDR3) from read

A) Read dropped if lacking clonal sequence
B) Read deferred to mapping if contains at least 1 low-quality nucleotide
C) Read dropped if contains too many low-quality nucleotides (.7% of total)
D) Read retained as core clonotype if it has CDR3 and high quality nucleotides

2. Assembler builds core clonotypes by grouping reads from section 1D that have identical clonal sequence (CDR3). Two important properties:

   A) Clonal sequence (CDR3 identity)
   B) Count - number of reads of this clonotype

3. Assembler maps deferred reads (1B) to core clonotypes

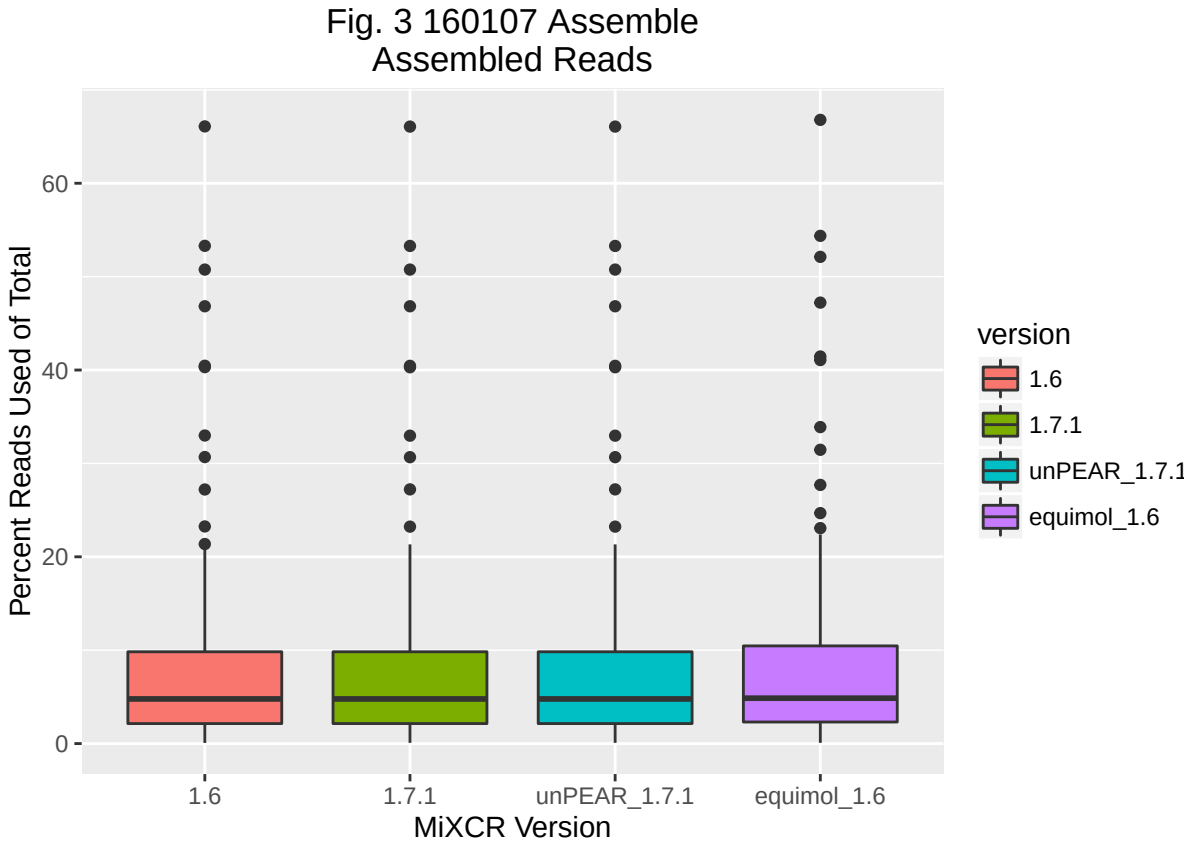
   A) Deferred read is aggregated to a core clonotype if it has a "fuzzy" match
      - What is fuzzy match? Can't find in docs
      - Possibly same as parameters for clustering
   B) If read matches multiple core clonotypes, one is chosen at random based on their abundances
   C) Read dropped if it doesn't match any core clonotype

4. Assembler clusters core clonotypes based on abundances

   A) Finds fuzzy matches between core clonotypes
      - Default allows 1 mutation in N regions in order to cluster
        - N regions are VD, DJ, and VJ junctions
      - Default allows 2 mismatches or indels between clones in different tree layers
        - parent and direct child
        - total of 2
   B) Aggregates into a hierarchical tree based on relative abundances
      - Head clone has highest abundance
      - Child layer 1 has almost identical sequence and an order of magnitude (or 2 or 3) fewer counts than head clone
      - Child layer 2 has almost identical sequence to Child 1 and an order of magnitude (or 2 or 3) fewer counts than Child 1
   C) Only head clones are considered final clones and only those counts are used.
   D) Align clonal sequences to reference V, D, J, and C genes

Our analysis pipeline collects the report generated by each pipeline and aggregates them into a QC summary. Notable columns that provide information on assembly:

1. Clonotype Count - how many (unique?) clonotypes identified
2. Total Reads Used in Assembly
3. Percent Reads Used (percent of total)
4. Percent Reads Used as Core Clonotypes (percent of reads used)
5. Low quality reads successfully mapped (percent of reads used)
6. Reads clustered in PCR error correction (percent of reads used)
7. Clonotypes eliminated by PCR error correction
8. Reads dropped due to lack of clonal sequence (percent of total)
9. Reads dropped due to low quality score (percent of total)
10. Reads dropped due to failed mapping (percent of total)



Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.070   2.140   4.780   8.071   9.830  66.090
```

Version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

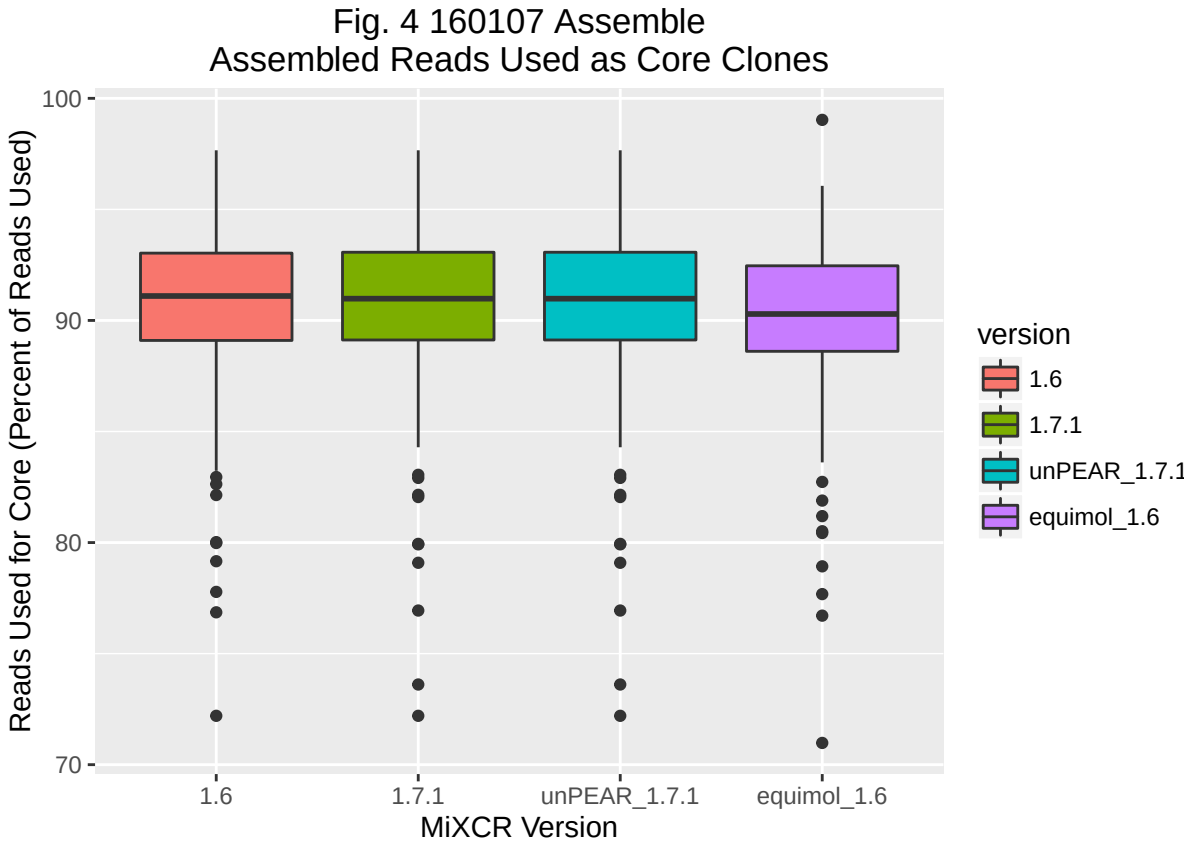
Unassembled reads, version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.080   2.310   4.860   8.474  10.460  66.790
```

This is concerning that not very many reads are assembled into clonotypes. We can look at some of the reasons that MiXCR gives us to see if we can figure out why.



Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   72.20   89.10   91.10   90.60   93.03   97.66
```

Version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

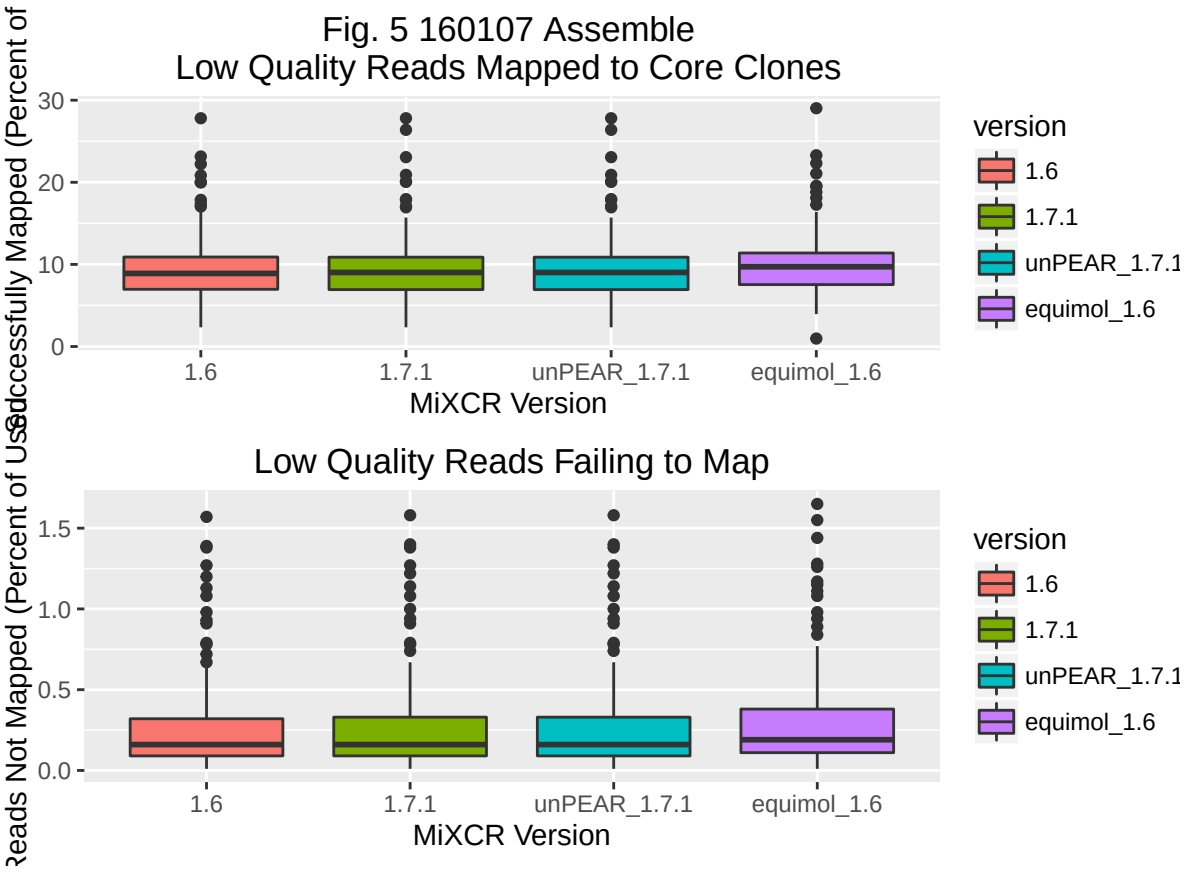
Unassembled reads, version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   70.98   88.61   90.29   90.09   92.46   99.03
```

Here we see that most of the reads that are used are core clonotypes, meaning they have high sequence quality and contain CDR3 sequences. So what happened to the rest of the reads assembled, but not identified as a core clone?



Summary for Percent Mapped

Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.340   6.970   8.900   9.401  10.900  27.800
```

Version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Unassembled reads, version 1.7.1:

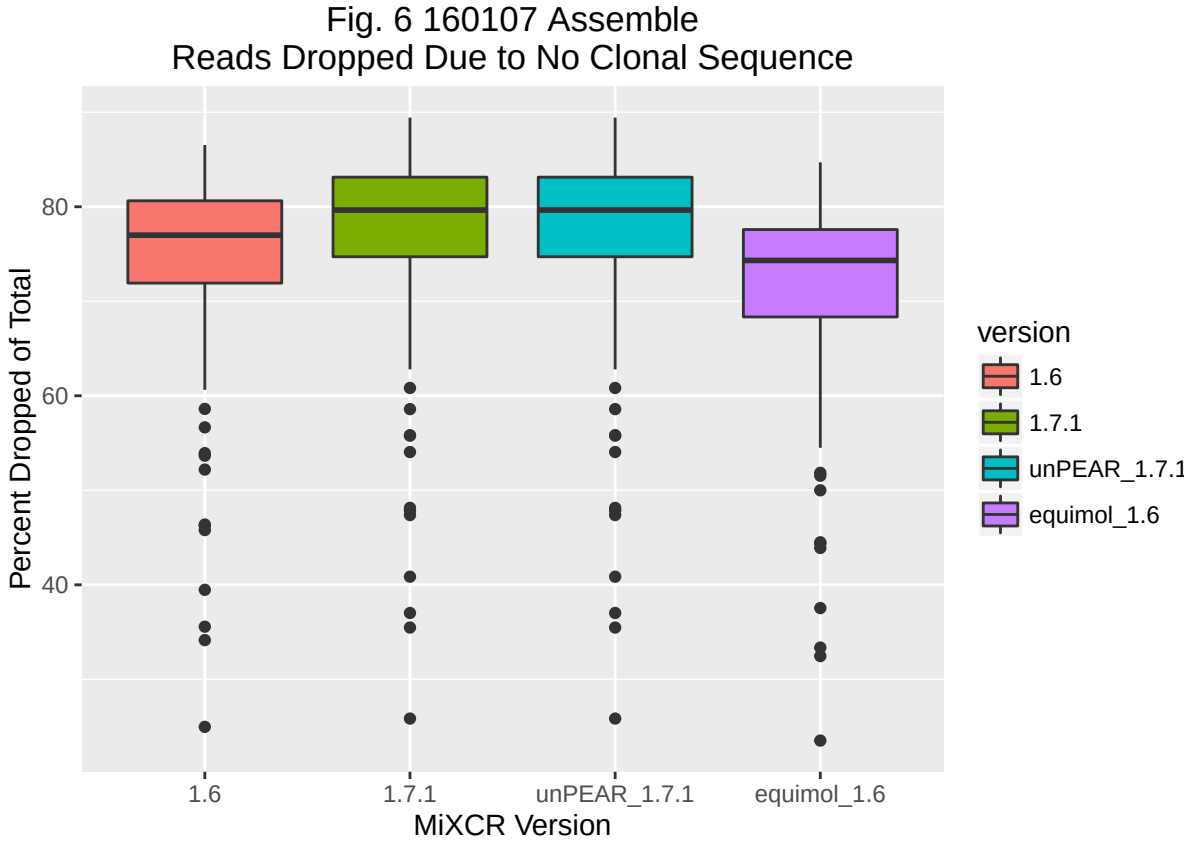
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.970   7.540   9.710   9.906  11.390  29.020
```

Looks like almost all reads that were deferred for mapping successfully mapped back to the core clonotypes. One would think that there should be a few reads that are deferred that are unable to be mapped back. Does

this suggest that our alignment parameters are too strict so that we're eliminating too many reads? We can look at the various reasons for dropped reads to get some clues.



Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   24.97   71.92   76.99   74.05   80.63   86.53
```

Version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Unassembled reads, version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   23.52   68.34   74.32   71.54   77.59   84.70
```

These results are puzzling. If around 80% of our reads are aligning to V and J regions, one would assume that they would also contain a CDR3 region to be extracted. According to one of the developers, assembly is just clustering records into clonotypes and all "markup" is done at the alignment step. That still doesn't fit with our high alignment percentages and low assembly percentages though. Are we certain our sample preparation is correct?

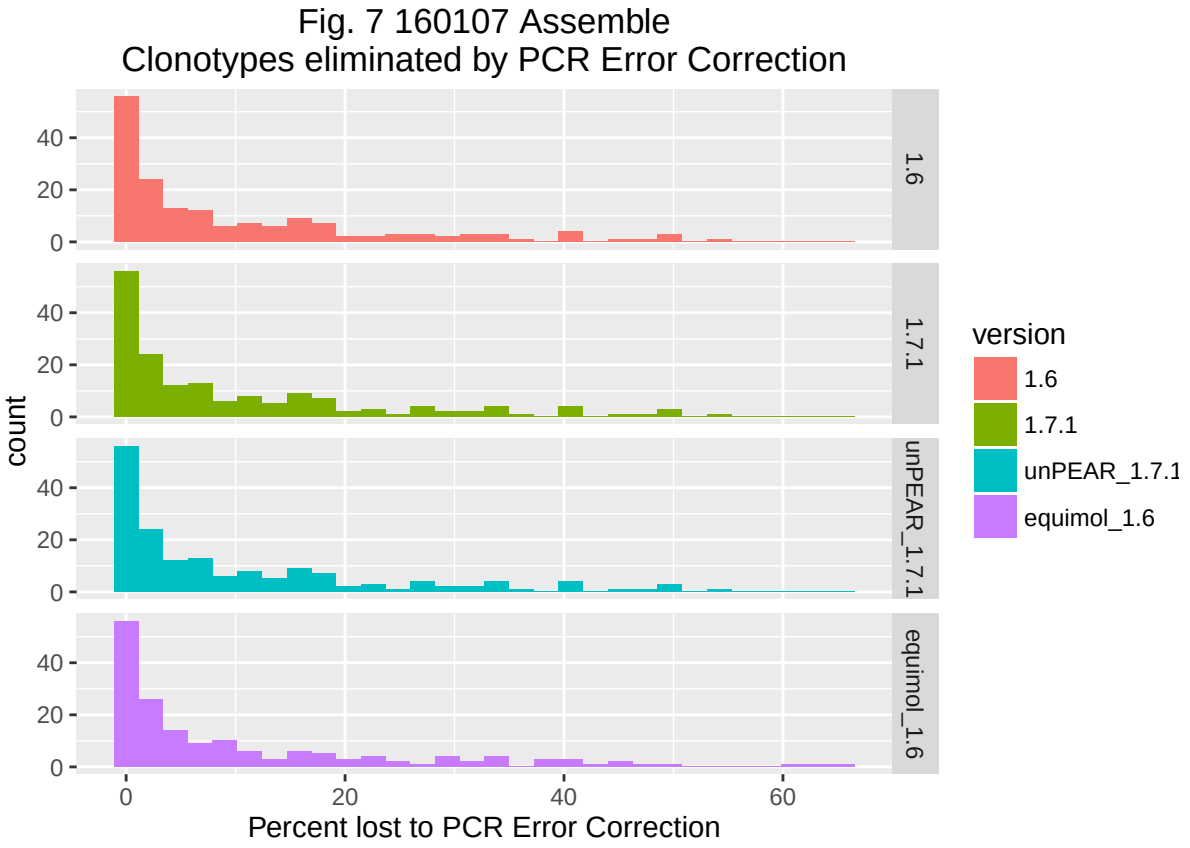
Moving Forward

We need to determine where the problem of read dropping is occurring. These reports suggest somewhere in the assembly step (because of the high alignment percentages), but the information on how MiXCR runs suggests that the important part of clone identification actually occurs when reads are aligned.

1. Use IMGT library instead of GenBank and see if that increases our assembly percentages
2. Leverage exported clones file to map back to raw reads somehow?
    - Need to do some more thinking on this.

A few extra plots

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    24.0   177.0   876.7   625.0 15380.0
```

Version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Unassembled reads, version 1.7.1:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

Equimolar sequencing, version 1.6:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0      18     148    1537     669   50410
```

These histograms show the percentage of clonotypes lost during the clustering stage. We can see that most are pretty negligible, although a non-trivial amount of our samples are losing 20-60% of clonotypes because of this clustering.

Moving Forward

We can turn off clustering completely as well as alter its parameters. The one problem is that we don't know if these clonotypes that we're losing are real clonotypes or if the program is correctly identifying PCR errors and they're actually erroneous clonotypes. Need to do some more thinking on this. Also may not be relevant right now because our assembly percentage is a more important problem

Export

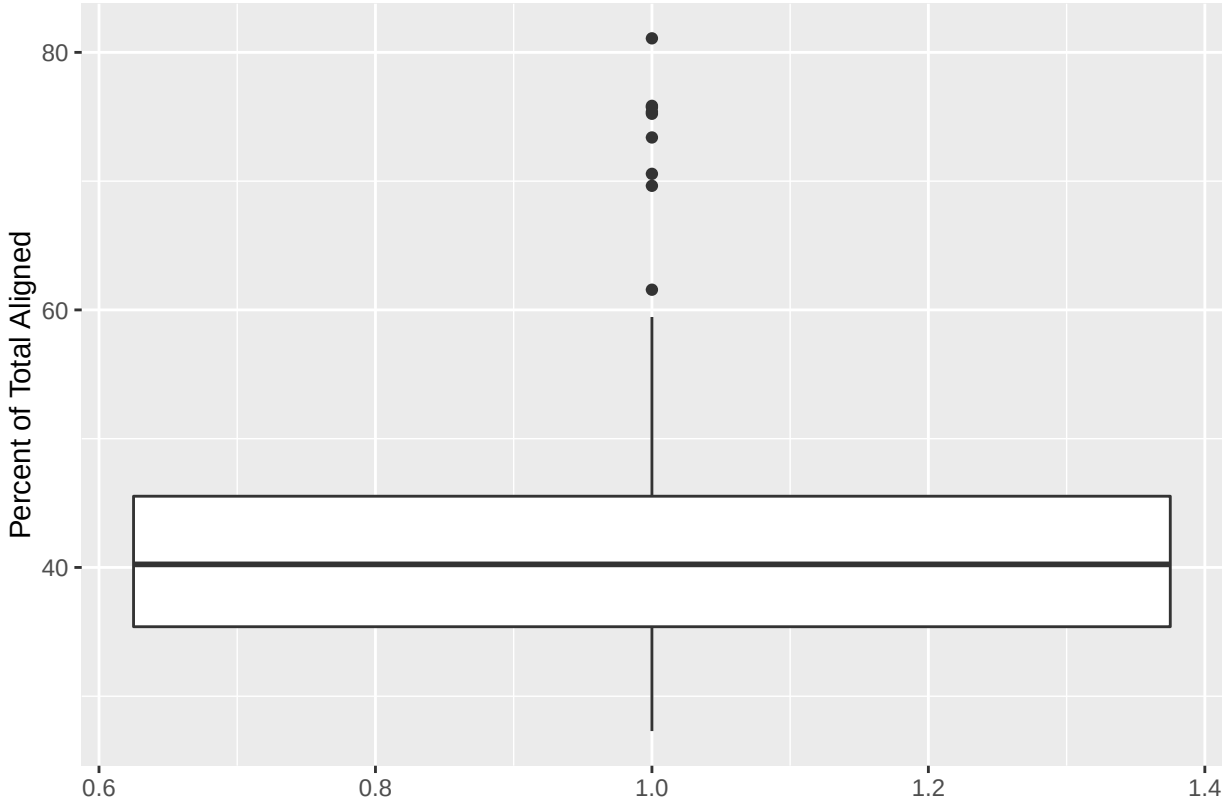
We can export alignments and assemblies to tab-separated files for manual and programmatic inspection. I need to take a look at the outputs of these files and see how we might be able to leverage them to help solve some of these problems. Can maybe see if alignment is capturing D regions or not, as well as in assembly. More to come.

New Data

New Analysis of equivolume_DNA160107 using read-id capture

We would expect all aligned reads to have a V, D, and a J region. MiXCR only requires a V and a J hit for alignment, but if our sequencing is correct, we should always have a D region when we have a V and a J region.

Fig. 8 160107 Aligned Reads Missing D Region



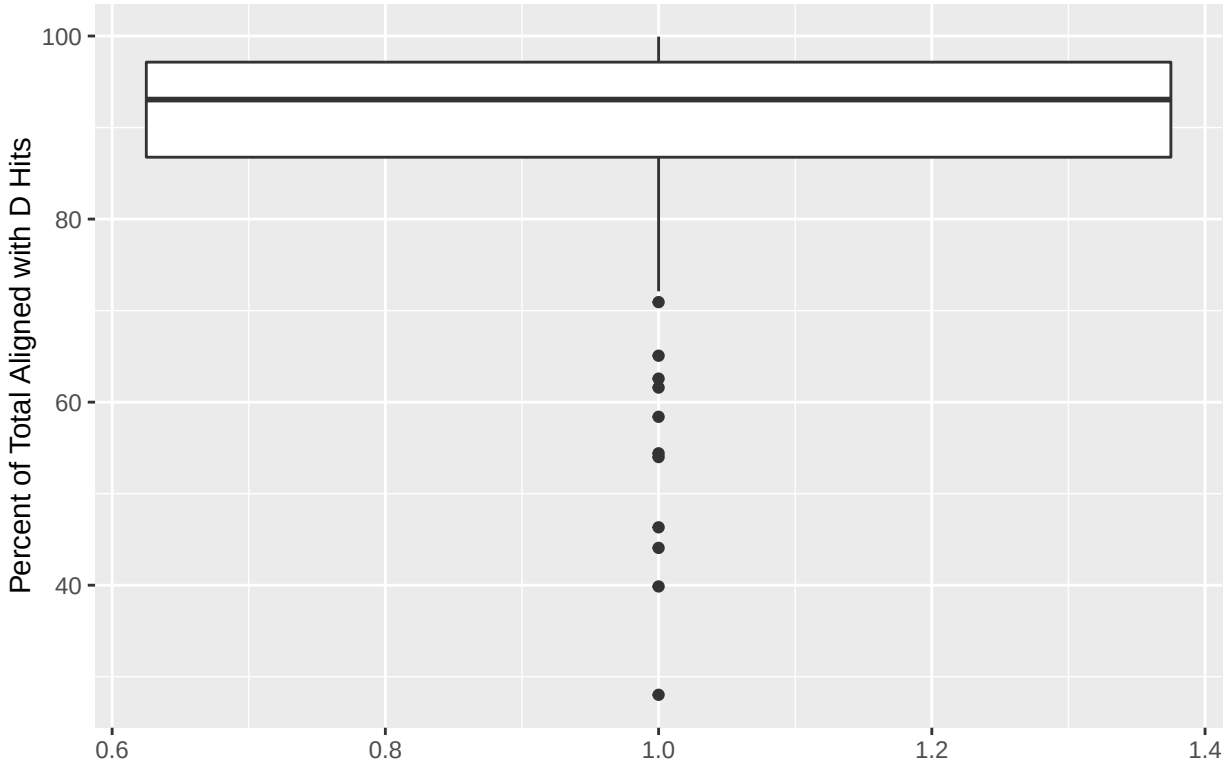

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   27.29   35.40   40.24   42.24   45.53   81.09
```

Around 40 percent of our reads are being aligned to reference V and J segments, but do not have a D region hit. The developers list two main regions why a D gene does not get aligned:

- The D gene was too short (fewer than 5 nucleotides) and alignment fails due to low score
  - Can change absoluteMinScore and relativeMinScore to see if this is the case
- The D gene was completely "trimmed" during VDJ recombination, apparently a relatively common event.
  - How common is this? Surely not 40% of reads common, but may account for part of this.

Another issue that we've been having is many of our aligned reads don't assemble. Out of the aligned reads that have D hits, how many of those failed to assemble?

Fig. 9 160107 Aligned Reads With D Region
But Failed to Assemble

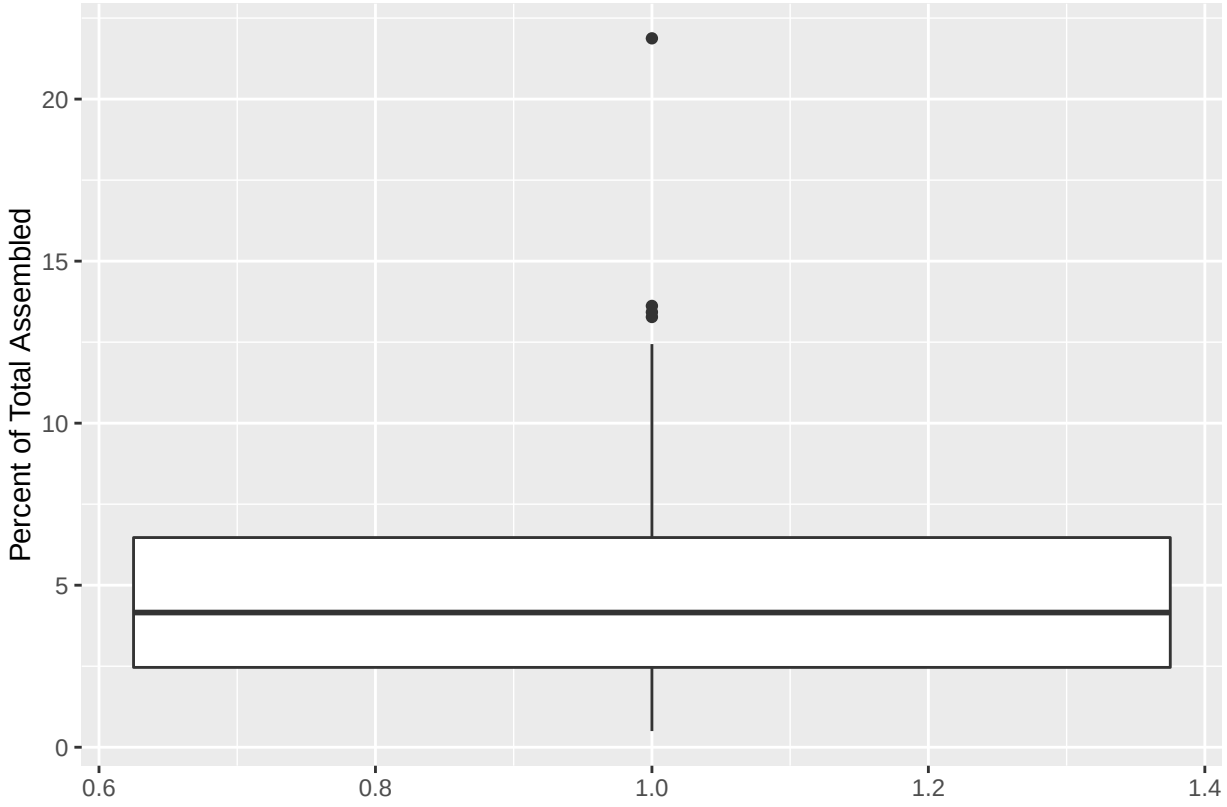



```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   28.03   86.76   93.06   89.45   97.15   99.94
```

Here we see that around 90% of our reads that align with D regions (meaning they have a V, D, and J alignment), don't assemble! This does not make much sense to me. It also suggests that there is a problem both at the alignment stage - we're not catching D regions in alignment, but also at the assembly stage - we're not assembling the D regions that we do catch.

One final thing to note is that not all of our assembled reads have D hits:

Fig. 10 160107 Assembled Reads Missing D Region




```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4988  2.4640  4.1570  4.9020  6.4710 21.8800
```

While this is a smaller fraction than some of the other distributions, this result is puzzling. The assembly stage extracts CDR3 regions, so how do we extract CDR3 regions from reads that supposedly don't have them?

I think sending some data to the MiXCR people will be very enlightening for us.