Expected Alignment Lengths

Wes Horton

June 28, 2016

Overview

We recently received a new batch (DNA160609LC) of TCR sequences and want to determine the extent to which MiXCR correctly and incorrectly identifies V and J alignments. We have expected alignment lengths for all V and J regions that we can compare to actual alignment lengths output by MiXCR.

We're using Sample 100 (which is a blood sample with spike-ins) for this analysis. Additionally, all aligments with AA..Seq..CDR3 == "CASSDAGGRNTLYF" have been removed due to a contamination of monoclonal p14 DNA.

Data Wrangling and Set Up

```r
## For testing: subset data
align.data <- align.data[1:10000,]

## Clean up
align.data$Best.V.hit <- gsub("^TRB|\\*00", '', align.data$Best.V.hit)
align.data$Best.J.hit <- gsub("^TRB|\\*00", '', align.data$Best.J.hit)

##  Calculate V and J alignment Lengths
align.lengths <- get.lengths(align.data)

## Combine to original data
align.data <- cbind.data.frame(align.data, align.lengths)

## Subset only alignments that assembled to clones
clone.data <- align.data[complete.cases(align.data$Clone.Id),]

## Calculate total alignments for QC later
total.alignments <- length(align.data[,1])

## Calculate total assembled alignments for QC later
total.clones <- length(clone.data[,1])
```

Summary of Current Status

Right now we have two dataframes of interest. In both, each row corresponds to an alignment (aka read) from the file and each column corresponds to a particular piece of information about it. The columns of most interest to us are the lengths of the V and J alignments, as well as V,D, and J identity/presence. The first dataframe consists of all of the alignments present in the file, whereas the second consists of all alignments that assembled to a clone during the assemble step.

Subset by estimated length range

First, let's subset by the expected total alignment length range of 170-240 nucleotides. This is the range that we expect our alignments to reside in. An alignment with an alignment length outside of this range is expected to be a false/bad alignment, and those within the range are expected to be true/good alignments.

```
align.outside.range <- align.data[align.data$tot.length < 170 | align.data$tot.length > 240,]
num.outside.range <- length(align.outside.range[,1])
align.in.range <- align.data[align.data$tot.length >= 170 & align.data$tot.length <= 240,]
num.in.range <- length(align.in.range[,1])
```

Just based on total length criteria, 86.03% of our alignments are incorrect. This is possible, although not likely. One reason that using a blanket 170-240 length filter is that the expected alignment lengths for the 20 different V genes and 13 different J genes vary. A short V coupled with a short J may not reach the 170 bp minimum.

Let's try and use the expected alignment lengths to figure out if this is accurate or not. For each alignment within the in/outside range tables, we want to see if the V alignment length is within 10% of the expected length for that V gene. A summary of the possibilities are:

1. True Positive: Alignment length is within range, and V alignment is of expected length
2. False Positive: Alignment length is within range, but V alignment is not of expected length
3. True Negative: Alignment length is outside range, and V alignment is not of expected length
4. False Negative: Alignment length is outside range, but V alignment is of expected length

Check alignments using expecteded V lengths (subsetted by estimated total length)

```
## Negatives
## First Check how many that are outside the normal range have good V alignments
## (false negatives)
outside.good.v <- compare.lengths.V(align.outside.range, expected.data)$good
num.outside.good.v <- length(outside.good.v[,1])
## how many are outside normal range and have bad V alignments (true negatives)
outside.bad.v <- compare.lengths.V(align.outside.range, expected.data)$bad
num.outside.bad.v <- length(outside.bad.v[,1])
##  Postives
## how many are inside normal range and have good V alignments (true positives)
inside.good.v <- compare.lengths.V(align.in.range, expected.data)$good
num.inside.good.v <- length(inside.good.v[,1])
## how many are inside the normal range, but have bad V alignments
inside.bad.v <- compare.lengths.V(align.in.range, expected.data)$bad
num.inside.bad.v <- length(inside.bad.v[,1])
```

Now, using the expected V alignment length, we see that actually, 54.794839% of the reads that we intially flagged as bad due to our total alignment range, are actually good. Only 44.7634546% were correctly identified as bad alignments. To summarize:

1. True Positives: 86.972083%
2. False Positives: 13.027917%
3. True Negatives: 44.7634546%
4. False Negatives: 54.794839%

Check alignments using clone IDs (subsetted by total length)

As a further check, we can look at the clone IDs from these subsets. A clone ID means a read successfully assembles, and is further evidence for a correct alignment.

```r
## Negatives
# How many of our "false negatives" have clone IDs
outside.good.v.assembled <- outside.good.v[complete.cases(outside.good.v$Clone.Id),]
num.outside.good.v.assembled <- length(outside.good.v.assembled[,1])

# How many of our "true negatives" have clone IDs
outside.bad.v.assembled <- outside.bad.v[complete.cases(outside.bad.v$Clone.Id),]
num.outside.bad.v.assembled <- length(outside.bad.v.assembled[,1])

## Positives
inside.good.v.assembled <- inside.good.v[complete.cases(inside.good.v$Clone.Id),]
num.inside.good.v.assembled <- length(inside.good.v.assembled[,1])
inside.bad.v.assembled <- inside.bad.v[complete.cases(inside.bad.v$Clone.Id),]
num.inside.bad.v.assembled <- length(inside.bad.v.assembled[,1])
```

Double checking with Clone IDs, we see that most (95.2481969%) alignments that we determined are false positives using the V alignment length, are also considered false positives using Clone ID. Conversely, only 0.8828876% of our so-called true negatives have clone IDs. These numbers suggest that using V alignment length may be a good filter for bad reads. In addition to how correctly V length determines bad reads, we can take a look at how accurately it identifies good reads. Of the reads that we listed as true positives (inside range and good V length), 78.6831276% assembled to clones. Additionally, of the reads that we said were false positives (inside range, but bad V), 4.9450549% actually assembled to clones.

It appears that appropriate V length is a better determinant of a bad read than it is for a good read. Subsetting by total length first and then checking V alignment quality makes it difficult to isolate the exact role of V alignment length in determining good reads. Let's go back to the full data and just use the V alignment length as our filter.

Check alignments using expected V lengths (full data)

So let's see how everything looks if we don't subset by the 170-240 length and instead just subset based on V length. A revised set of possible results is:

1. True Positive: Appropriate V length, and assembles to a clone
2. False Positive: Appropriate V length, but doesn't assemble to a clone
3. True Negative: Bad V length, and doesn't assemble to a clone
4. False Negative: Bad V length, but assembles to a clone

```r
good.v <- compare.lengths.V(align.data, expected.data)$good
num.good.v <- length(good.v[,1])
bad.v <- compare.lengths.V(align.data, expected.data)$bad
num.bad.v <- length(bad.v[,1])
```

Using the expected V alignment lengths, we estimate that 59.29% of our reads are quality alignments. Now we want to try and see how accurate this estimate is.

Check alignments using clone IDs (full data)

So using just the V alignment length to determine good and bad alignments, how many false negatives and false positives do we observe (using Clone ID as the condition for good alignment)?

```r
bad.v.assembled <- bad.v[complete.cases(bad.v$Clone.Id),]
num.bad.v.assembled <- length(bad.v.assembled[,1])
good.v.assembled <- good.v[complete.cases(good.v$Clone.Id),]
num.good.v.assembled <- length(good.v.assembled[,1])
```

So 1.0662038% of our "bad" alignments are actually false negatives, so we're missing out on that information (which isn't much). Alternatively 91.8536009% of our good alignments are true positives, meaning that we do have a decent amount of false positives that we may want to eliminate somehow.

Check alignments using both V and J expected lengths (full data)

Now let's see how we do if we use both the V and the J alignment length criteria:

```r
good.both <- compare.lengths.both(align.data, expected.data)$good
num.good.both <- length(good.both[,1])
bad.both <- compare.lengths.both(align.data, expected.data)$bad
num.bad.both <- length(bad.both[,1])
```

Here we see a huge drop in the number of alignments that we specify as good. Based on the above calculations, many of these are going to be false negatives if we look at how many assemble into clones.

```r
good.both.assembled <- good.both[complete.cases(good.both$Clone.Id),]
num.good.both.assembled <- length(good.both.assembled[,1])
bad.both.assembled <- bad.both[complete.cases(bad.both$Clone.Id),]
num.bad.both.assembled <- length(bad.both.assembled[,1])
```

As expected, we now see that 31.4374693% of the reads flagged as bad reads are actually false negatives. Interestingly, only 92.4889925% of reads flagged as good reads are true positives. This is less than just using the V alignment length as our good/bad criterion.

Check alignments using J lengths

Let's see if the J lengths give us similar results, or if it's just the combination of the two.

```r
good.j <- compare.lengths.J(align.data, expected.data)$good
num.good.j <- length(good.j[,1])
bad.j <- compare.lengths.J(align.data, expected.data)$bad
num.bad.j <- length(bad.j[,1])
```

Using the J alignments gives us similar numbers to using both alignments. Let's check against Clone IDs as well

```r
good.j.assembled <- good.j[complete.cases(good.j$Clone.Id),]
num.good.j.assembled <- length(good.j.assembled[,1])
bad.j.assembled <- bad.j[complete.cases(bad.j$Clone.Id),]
num.bad.j.assembled <- length(bad.j.assembled[,1])
```

Again we see that 31.7147193% of the reads flagged as bad are actually false negatives. Why is the J alignment length not a good predictor of alignment quality. If it were that the lengths are too short and that we're identifying primers as good alignments when they're not, we should have an increase in false positives, not false negatives.

See how the assembled alignments look

Now, if we subset for only those alignments that assembled, do they meet the V and J alignment length criteria?

```r
clone.good.v <- compare.lengths.V(clone.data, expected.data)$good
clone.num.good.v <- length(clone.good.v[,1])
clone.bad.v <- compare.lengths.V(clone.data, expected.data)$bad
clone.num.bad.v <- length(clone.bad.v[,1])
```

So 99.1985428% of reads that successfully assemble meet the V alignment critera. How about J?

```r
clone.good.j <- compare.lengths.J(clone.data, expected.data)$good
clone.num.good.j <- length(clone.good.j[,1])
clone.bad.j <- compare.lengths.J(clone.data, expected.data)$bad
clone.num.bad.j <- length(clone.bad.j[,1])
```

We see similar numbers once again: 34.2622951% of reads that successfully assemble are considered bad reads based on J alignment length.

To Do: determine if these sucessfully assembled reads with bad J lengths are true clones, or if the assembler is messing up

Preliminary Conclusions

Expected V alignment length is a good predictor of a quality alignment. Total alignment length and J alignment length are not. Based on the V alignment length, I would conclude that MiXCR is accurately aligning and assembling the data. However, Clone ID is not a perfect indicator of a successful clonotype. There are many assembled reads that do not have D hits. We need to incorporate D alignment length as well as just presence of a D alignment into these quality control checks in order to make a final conclusion on the quality of MiXCR.

Incorporating D Presence

So if we use presence of a D hit as our criterion for determining a good alignment, instead of clone ID, what do the data look like?

First, let's see how many clones don't have a D region.

```r
clone.with.d <- clone.data[complete.cases(clone.data$Best.D.hit),]
num.clone.with.d <- length(clone.with.d[,1])
```

So only 63.8251366% of our assembled alignments (aka clones) actually have D hits.

Based on this, let's re-check false positives and false negatives using both V and J alignment lengths:

```r
with.d.good.both <- compare.lengths.both(clone.with.d, expected.data)$good
with.d.num.good.both <- length(with.d.good.both[,1])
with.d.bad.both <- compare.lengths.both(clone.with.d, expected.data)$bad
with.d.num.bad.both <- length(with.d.bad.both[,1])
```

Which V and J regions contribute to the most bad reads?

Using the reads flagged as bad by incorrect V alignment length, we can see if there are a few primers (or primer combinations) that are the cause.

```
bad.v.primers <- data.frame("V" = bad.v$Best.V.hit, "J" = bad.v$Best.J.hit,
                            "VJ" =paste(bad.v$Best.V.hit, bad.v$Best.J.hit, '-'),
                            "value" = rep(1, times = length(bad.v[,1])),
                            stringsAsFactors = F)
```