riboSeqR

Thomas J. Hardcastle, Betty Y.W. Chung

October 17, 2016

Introduction

Ribosome profiling extracts those parts of a coding sequence currently bound by a ribosome (and thus, are likely to be undergoing translation). Ribosomes typically cover between 20-30 bases of the mRNA (dependant on conformational changes) and move along the mRNA three bases at a time. Sequenced reads of a given length are thus likely to lie predominantly in a single frame relative to the start codon of the coding sequence. This package presents a set of methods for parsing ribosomal profiling data from multiple samples and aligned to coding sequences, inferring frameshifts, and plotting the average and transcript-specific behaviour of these data. Methods are also provided for extracting the data in a suitable form for differential translation analysis. For a fuller description of these methods and further examples of their use, see Chung & Hardcastle *et al* (2015) [1].

Getting Data

`riboSeqR` currently reads alignment data from flat text files that contain (as a minimum), the sequence of the read, the name of the sequence to which the read aligns, the strand to which it aligns, and the starting position of alignment. A *Bowtie* alignment (note that *Bowtie*, rather than *Bowtie2*, is recommended for short reads, which ribosome footprints are) using the option "–suppress 1,6,7,8" will generate this minimal data. It is by default assumed that the data are generated in this way, and the default columns specification for the default `readRibodata` function (see below) reflects this.

Workflow Example

Begin by loading the riboSeqR library.

```
> library(riboSeqR)
```

Identify the data directory for the example data.

```
> datadir <- system.file("extdata", package = "riboSeqR")
```

The `fastaCDS` function can be used to guess at potential coding sequences from a (possibly compressed; see `base::file`) fasta file containing mRNA transcripts (note; do not use this on a genome!). These can also be loaded into a *GRanges* object from an annotation file.

```
> chlamyFasta <- paste(datadir, "/rsem_chlamy236_deNovo.transcripts.fa", sep = "")
> fastaCDS <- findCDS(fastaFile = chlamyFasta,
+                     startCodon = c("ATG"),
+                     stopCodon = c("TAG", "TAA", "TGA"))
```

The ribosomal and RNA (if available) alignment files are specified.

```
> ribofiles <- paste(datadir,
+                    "/chlamy236_plus_deNovo_plusOnly_Index", c(17,3,5,7), sep = "")
> rnafiles <- paste(datadir,
+                    "/chlamy236_plus_deNovo_plusOnly_Index", c(10,12,14,16), sep = "")
```

The aligned ribosomal (and RNA) data can be read in using the `readRibodata` function. The columns can be specified as a parameter of the `readRibodata` function if the data in the alignment files are differently arranged.

```
> riboDat <- readRibodata(ribofiles, replicates = c("WT", "WT", "M", "M"))
```

The alignments can be assigned to frames relative to the coding coordinates with the `frameCounting` function.

```
> fCs <- frameCounting(riboDat, fastaCDS)
```

The predominant reading frame, relative to coding start, can be estimated from the frame calling (or from a set of coordinates and alignment data) for each n-mer. The weighting decribes the proportion of n-mers fitting with the most likely frameshift. The reading frame can also be readily visualised using the `plotFS` function.

```
> fS <- readingFrame(rC = fCs); fS
           26    27    28   29   30
         1030  8261 16355 2379 1346
         2847 36011  3582 1634  436
         3352  1687  3331  701  609
frame.ML    2     1     0    0    0
```

```
> plotFS(fS)
```

These can be filtered on the mean number of hits and unique hits within replicate groups to give plausible candidates for coding. Filtering can be limited to given lengths and frames, which may be inferred from the output of the `readingFrame` function.

```
> ffCs <- filterHits(fCs, lengths = c(27, 28), frames = list(1, 0),
+                    hitMean = 50, unqhitMean = 10, fS = fS)
```

We can plot the total alignment at the 5' and 3' ends of coding sequences using the `plotCDS` function. The frames are colour coded; frame-0 is red, frame-1 is green, frame-2 is blue.

```
> plotCDS(coordinates = ffCs@CDS, riboDat = riboDat, lengths = 27)
```

Note the frameshift for 28-mers.

```
> plotCDS(coordinates = ffCs@CDS, riboDat = riboDat, lengths = 28)
```

We can plot the alignment over an individual transcript sequence using the `plotTranscript` function. Observe that one CDS (on the right) contains the 27s in the same phase as the CDS (they are both red) while the putative CDSes to the left are not in phase with the aligned reads, suggesting either a sequence error in the transcript or a misalignment. The coverage of RNA sequenced reads is shown as a black curve (axis on the right).

```
> plotTranscript("CUFF.37930.1", coordinates = ffCs@CDS,
+                riboData = riboDat, length = 27, cap = 200)
```

```
NULL
```

We can extract the counts from a *riboCoding* object using the `sliceCounts` function

```
> riboCounts <- sliceCounts(ffCs, lengths = c(27, 28), frames = list(0, 2))
```

Counts for RNA-sequencing can be extracted using from the riboData object and the coding coordinates using the `rnaCounts` function. This is a relatively crude counting function, and alternatives have been widely described in the literature on mRNA-Seq.

```
> rnaCounts <- rnaCounts(riboDat, ffCs@CDS)
```

These data may be used in an analysis of differential translation through comparison with the RNA-seq data. See the description of a beta-binomial analysis in the *baySeq* vignettes for further details.

```
> library(baySeq)
> pD <- new("countData", replicates = ffCs@replicates,
+           data = list(riboCounts, rnaCounts),
+           groups = list(NDT = c(1,1,1,1), DT = c("WT", "WT", "M", "M")),
+           annotation = as.data.frame(ffCs@CDS),
+           densityFunction = bbDensity)
> libsizes(pD) <- getLibsizes(pD)
```

```
            26    27    28    29    30
          1030  8261 16355  2379  1346
          2847 36011  3582  1634   436
          3352  1687  3331   701   609
frame.ML     2     1     0     0     0
```
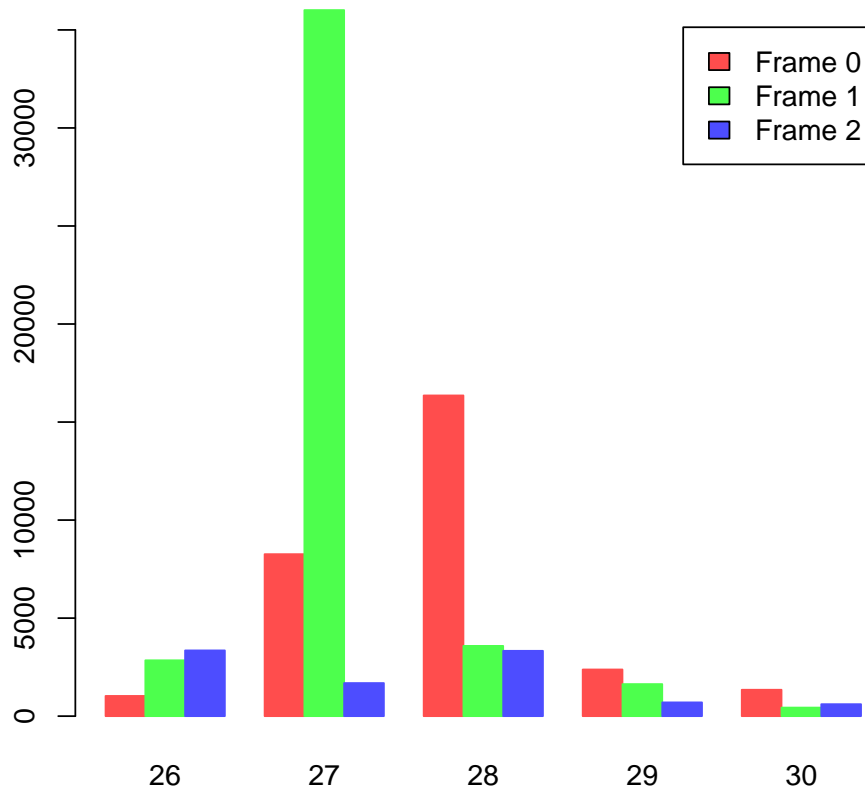

Figure 1: Number of n-mers in each frame relative to coding start. 27-mers are predominantly in frame-1, while 28-mers are chiefly in frame-0.

```
> pD <- getPriors(pD, cl = NULL)
> pD <- getLikelihoods(pD, cl = NULL)

.

> topCounts(pD, "DT", normaliseData = TRUE)
           seqnames start  end width strand frame startCodon stopCodon context minus3
1        CUFF.28790.1   165  530   366      *     2        ATG       TAA CACATGC      C
2        CUFF.26092.1     7  450   444      *     0        ATG       TGA ACCATGG      A
3   Cre17.g723750.t1.3   516  638   123      *     2        ATG       TAA GGCATGA      G
4   Cre12.g554250.t1.1   580 3951  3372      *     0        ATG       TGA GAAATGG      G
5            g18028.t1   192 2108  1917      *     2        ATG       TGA GAAATGG      G
6   Cre17.g717750.t1.2   106  828   723      *     0        ATG       TGA ACCATGG      A
7         CUFF.9523.1    78 1040   963      *     2        ATG       TAA AAAATGG      A
8             g9881.t1   219 2030  1812      *     2        ATG       TGA GAAATGG      G
9   Cre03.g160200.t1.2    87  848   762      *     2        ATG       TGA ACAATGC      A
10        CUFF.26944.1   612 1049   438      *     2        ATG       AAA GCCATGC      G
```
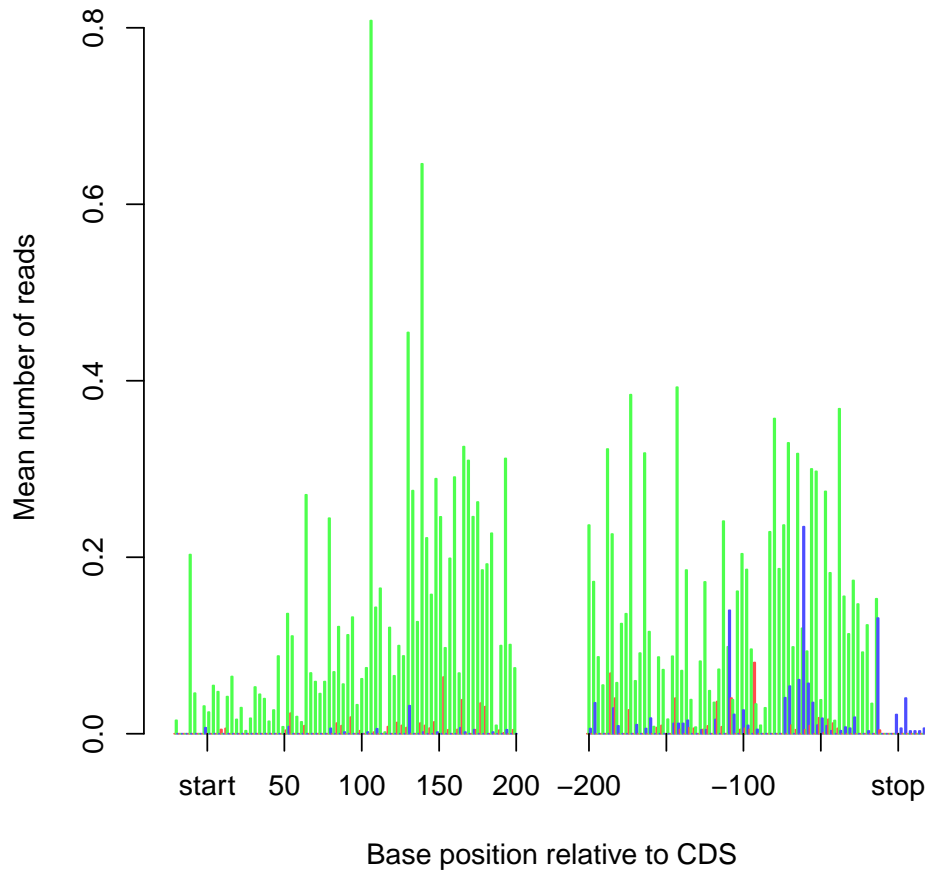
Figure 2: Average alignment of 27-mers to 5' and 3' ends of coding sequences.

```
    plus1  WT.1   WT.2    M.1    M.2  Likelihood ordering     FDR.DT    FWER.DT
1       C   3:3    1:1    0:0    0:0 0.0006090042    M=WT  0.9993910  0.9993910
2       G   1:1    1:1    0:0    0:0 0.0006090041    M=WT  0.9993910  0.9999996
3       A   3:3    3:3    0:0    0:0 0.0006090036    M=WT  0.9993910  1.0000000
4       G   0:0    0:0    2:2    0:0 0.0005777916    M=WT  0.9993988  1.0000000
5       G   1:1    1:1    2:2    3:3 0.0005633553    M=WT  0.9994064  1.0000000
6       G   1:1    1:1    2:2    3:3 0.0005633552    M=WT  0.9994114  1.0000000
7       G 10:10  14:14  59:59  33:33 0.0005633550    M=WT  0.9994150  1.0000000
8       G   1:1    0:0    2:2    0:0 0.0005633550    M=WT  0.9994177  1.0000000
9       C   4:4    0:0    2:2    5:5 0.0005633547    M=WT  0.9994198  1.0000000
10      C   3:3    1:1    4:4    3:3 0.0005633547    M=WT  0.9994215  1.0000000
```

Session Info

```
> sessionInfo()

R version 3.3.1 (2016-06-21)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 16.04.1 LTS

locale:
```
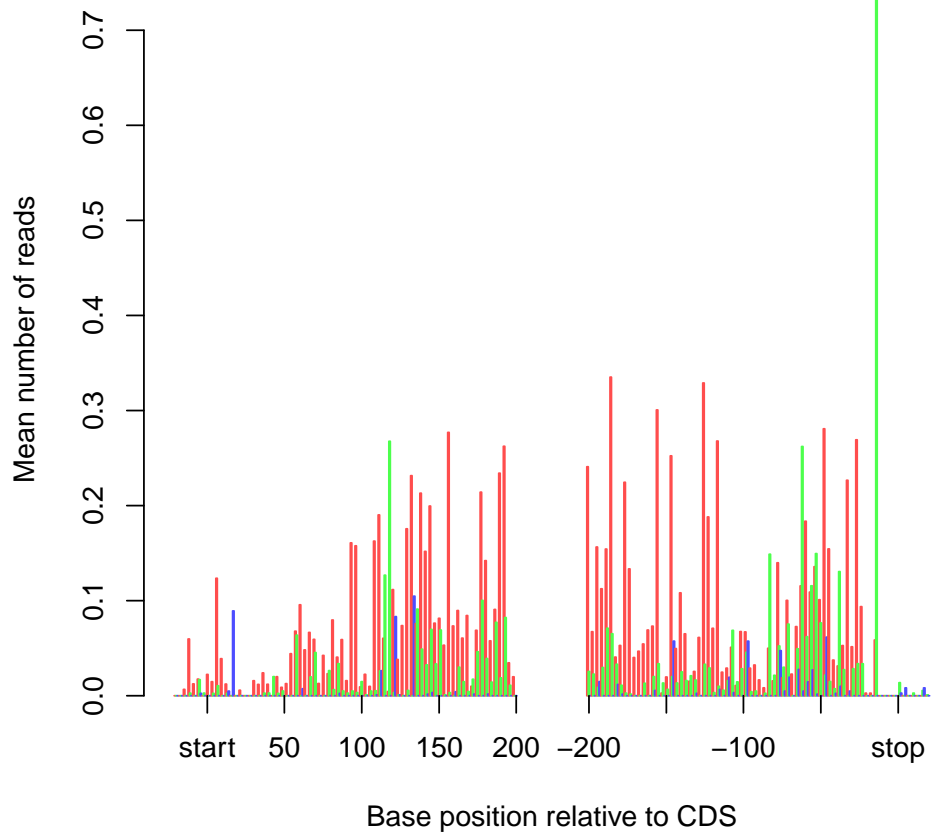
Figure 3: Average alignment of 28-mers to 5' and 3' ends of coding sequences.

```
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C              LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=C              LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8      LC_NAME=C                 LC_ADDRESS=C
[10] LC_TELEPHONE=C            LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] parallel  stats4    stats     graphics  grDevices utils     datasets  methods
[9] base

other attached packages:
[1] baySeq_2.8.0         riboSeqR_1.8.0      abind_1.4-5          GenomicRanges_1.26.0
[5] GenomeInfoDb_1.10.0  IRanges_2.8.0       S4Vectors_0.12.0     BiocGenerics_0.20.0

loaded via a namespace (and not attached):
 [1] locfit_1.5-9.1      edgeR_3.16.0        lattice_0.20-34     Rsamtools_1.26.0
 [5] Biostrings_2.42.0   bitops_1.0-6        grid_3.3.1          zlibbioc_1.20.0
 [9] XVector_0.14.0      limma_3.30.0        BiocStyle_2.2.0     BiocParallel_1.8.0
[13] tools_3.3.1
```
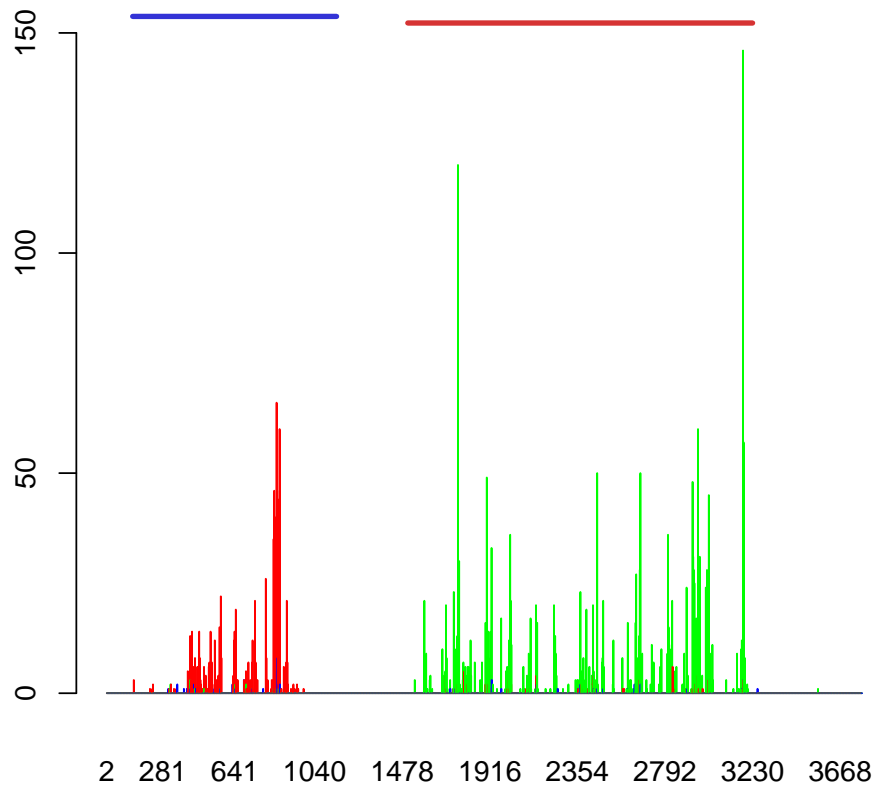
NULL

chlamy236_plus_deNovo_plusOnly_Index17 :: CUFF.37930.1

Figure 4: Alignment to individual transcript.

References

[1] BY Chung and TJ Hardcastle and JD Jones and N Irigoyen and AE Firth and DC Baulcombe and I Brierley *The use of duplex-specific nuclease in ribosome profiling and a user-friendly software package for Ribo-seq data analysis.* RNA (2015).