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 BAM files, or from flat text files, although BAM format is recommended.

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.

```r
> 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.

```r
> 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.

```r
> 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 describes the proportion of n-mers fitting with the most likely frameshift. The reading frame can also be readily visualised using the `plotFS` function.

```r
> fS <- readingFrame(rC = fCs); fS

> 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.

```r
> 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.

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

Note the frameshift for 28-mers.

```r
> 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).

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

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

```r
> 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.

```r
> 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.

```r
> 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)
> pD <- getPriors(pD, cl = NULL)
> pD <- getLikelihoods(pD, cl = NULL)
>
> topCounts(pD, "DT", normaliseData = TRUE)
  seqnames start end width strand frame startCodon stopCodon context
<table>
<thead>
<tr>
<th></th>
<th>minus3</th>
<th>plus1</th>
<th>WT.1</th>
<th>WT.2</th>
<th>M.1</th>
<th>M.2</th>
<th>likes</th>
<th>DT</th>
<th>FDR.DT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cre17.g723750.t1.3</td>
<td>0.9411765</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUFF.9523.1</td>
<td>0.9965398</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUFF.9661.1</td>
<td>0.9997965</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUFF.43721.1</td>
<td>0.9999880</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>G</th>
<th>G</th>
<th>11:11</th>
<th>1:1</th>
<th>2:2</th>
<th>1:1</th>
<th>0.05882353</th>
<th>M=WT</th>
<th>0.9411765</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUFF.9661.1</td>
<td>G</td>
<td>G</td>
<td>1:1</td>
<td>1:1</td>
<td>1:1</td>
<td>0:0</td>
<td>0.05882352</td>
<td>M=WT</td>
<td>0.9411765</td>
</tr>
<tr>
<td>CUFF.43721.1</td>
<td>G</td>
<td>C</td>
<td>5:5</td>
<td>4:4</td>
<td>3:3</td>
<td>0:0</td>
<td>0.05882350</td>
<td>M=WT</td>
<td>0.9411765</td>
</tr>
<tr>
<td>CUFF.9523.1</td>
<td>A</td>
<td>G</td>
<td>841:841</td>
<td>592:592</td>
<td>298:298</td>
<td>721:721</td>
<td>0.05882193</td>
<td>M=WT</td>
<td>0.9411769</td>
</tr>
</tbody>
</table>

FWER.DT
<table>
<thead>
<tr>
<th></th>
<th>0.9411765</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUFF.9661.1</td>
<td>0.9965398</td>
</tr>
<tr>
<td>CUFF.43721.1</td>
<td>0.9997965</td>
</tr>
<tr>
<td>CUFF.9523.1</td>
<td>0.9999880</td>
</tr>
</tbody>
</table>
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.
Figure 2: Average alignment of 27-mers to 5' and 3' ends of coding sequences
Figure 3: Average alignment of 28-mers to 5’ and 3’ ends of coding sequences
Figure 4: Alignment to individual transcript
Session Info

```r
> sessionInfo()
R version 3.6.1 (2019-07-05)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.3 LTS

Matrix products: default
BLAS: /home/biocbuild/bbs-3.10-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.10-bioc/R/lib/libRlapack.so

locale:
[1] LC_CTYPE=en_US.UTF-8  LC_NUMERIC=C
[3] LC_TIME=en_US.UTF-8    LC_COLLATE=C
[5] LC_MONETARY=en_US.UTF-8 LC_MESSAGES=en_US.UTF-8
[7] LC_PAPER=en_US.UTF-8   LC_NAME=C
[9] LC_ADDRESS=C          LC_TELEPHONE=C

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

other attached packages:
[1] baySeq_2.20.0 riboSeqR_1.20.0 abind_1.4-5
[4] GenomicRanges_1.38.0 GenomeInfoDb_1.22.0 IRanges_2.20.0
[7] S4Vectors_0.24.0 BiocGenerics_0.32.0

loaded via a namespace (and not attached):
[1] Rcpp_1.0.2    edgeR_3.28.0   knitr_1.25
[4] XVector_0.26.0 zlibbioc_1.32.0 BiocParallel_1.20.0
[7] lattice_0.20-38 rlang_0.4.1   tools_3.6.1
[10] grid_3.6.1     xfun_0.10    seqLogo_1.52.0
[13] htmltools_0.4.0 yaml_2.2.0    digest_0.6.22
[16] GenomeInfoDbData_1.2.2 BiocManager_1.30.9 bitops_1.0-6
[19] RCurl_1.95-4.12 evaluate_0.14 rmarkdown_1.16
[22] limma_3.42.0    compiler_3.6.1 Rsamtools_2.2.0
[25] Biostrings_2.54.0 locfit_1.5-9.1 BiocStyle_2.14.0
```

References