RNAprobR: An R package for analysis of the massive parallel sequencing based methods of RNA structure probing

Lukasz Jan Kielpinski, Nikos Sidiropoulos, Jeppe Vinther

Modified: 27 October, 2014. Compiled: May 2, 2019

Contents

1 Introduction

RNA structure probing is more and more often conducted with the use of high-throughput sequencing. Analysis of data coming from those experiments can be challenging and time consuming. Here we describe an R package - *RNAprobR* - which intends to standardize and simplify processing of experiments for which information on RNA susceptibility to different probing conditions is encoded in a location of sequenced reads termini.

2 Galaxy workflow

RNAprobR takes as input "Unique Barcodes" and "k2n" (or "Read Counts") files generated by "RNA probing" Galaxy workflow [1]. The package comes with sample data from HRF-Seq experiment [2]. This data was generated from raw sequencing reads (available under: http://people.binf.ku.dk/jvinther/data/HRF-Seq/) with the Galaxy workflow specifying: barcode sequence to NNNNNNN, 3' trimming length to 0 and using trimming untemplated nucleotides.

3 RNAprobR workflow

The package was designed with a specific processing workflow in mind (Fig. 1). It reads-in the Unique Barcodes files and calculates EUCs (`readsamples()`), compiles positional information based on sequenced fragments (`comp()`), performs data normalization (`dtcr()`, `slograt()`, `swinsor()`) and exports data in various formats (`GR2norm_df()`, `plotRNA()`, `norm2bedgraph()`)

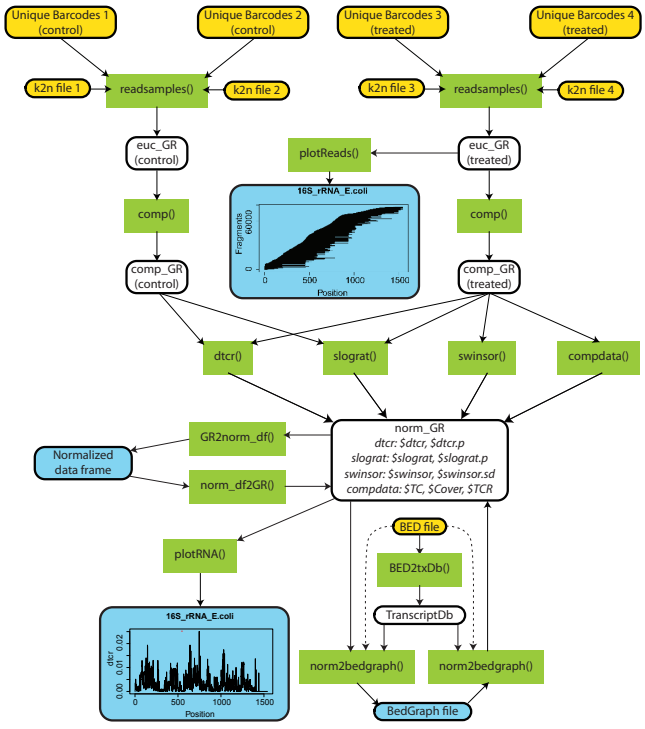


Figure 1: RNAprobR workflow

4 EUC Calculation

Proposed data analysis takes advantage of removing PCR duplicates by looking at the sequences of random barcodes attached to the beginning of each read. Counting each unique barcode only once is correct for fragments present in low counts range but leads to erroneous quantification in the high range (higher probability of physically distinct barcodes bearing the same sequence). We define Estimated Unique Counts (EUCs) as the number of molecules expected to give rise to the observed number of unique barcodes. In function `readsamples()` we have implemented two methods to calculate EUCs: `euc="Fu"` [3] or `euc="HRF-Seq"` (Kielpinski and Vinther, 2014). The latter requires supplying precomputed k2n files (generated by "RNA probing" Galaxy workflow if "Produce k2n file" checked or `k2n_func()` function from the package).

Note: if no random barcode was used in experiment, "Unique Barcodes" and "Read Counts" files are identical and each can be used for subsequent analysis. In this case for the `readsamples()` function use option: `euc="counts"`.

5 Reading in datasets

The first step in data processing is reading in the Unique Barcodes files, combining samples of the same treatment (if we had e.g. repeated control) and calculating EUCs. All of those steps are performed by `readsamples()` function. Start with specifying paths to different Unique Barcodes files and their respective k2n files (order matters!), followed by reading-in the data and calculating EUCs according to HRF-Seq method:

```
> library(RNAprobR)
```

```
> treated <- c(system.file("extdata", "unique_barcodes14.gz", package="RNAprobR"),
+              system.file("extdata", "unique_barcodes22.gz", package="RNAprobR"))
> control <- c(system.file("extdata", "unique_barcodes16.gz", package="RNAprobR"),
+              system.file("extdata", "unique_barcodes24.gz", package="RNAprobR"))
> k2n_treated <- c(system.file("extdata", "k2n_14", package="RNAprobR"),
+                  system.file("extdata", "k2n_22", package="RNAprobR"))
> k2n_control <- c(system.file("extdata", "k2n_16", package="RNAprobR"),
+                  system.file("extdata", "k2n_24", package="RNAprobR"))
> control_euc <- readsamples(control, euc="HRF-Seq", k2n_files=k2n_control)
> treated_euc <- readsamples(treated, euc="HRF-Seq", k2n_files=k2n_treated)
```

`control_euc` and `treated_euc` are *GenomicRanges* (GRanges) objects holding information on sequenced fragments span and EUC.

6 Compiling positional data

`comp()` function takes as input an object imported by `readsamples()` function and uses it to compile needed data for each nucleotide in analyzed RNA molecules. Specifically it computes:

- termination count (TC),
- coverage (Cover),

- termination-coverage ratio (TCR),
- priming count (PC).

If one specifies path to FASTA file which was used for mapping (option: `fasta_file`) it also adds nucleotide identity (nt). By specifying "cutoff" value, function discards fragments which are shorter than the provided value. Usage:

```
> treated_comp <- comp(treated_euc, cutoff=101, fasta_file =
+                          system.file("extdata", "hrfseq.fa", package="RNAprobR"))
> control_comp <- comp(control_euc, cutoff=101, fasta_file =
+                          system.file("extdata", "hrfseq.fa", package="RNAprobR"))
```

`treated_comp` and `control_comp` are *GRanges* objects with each range being of length 1.

7 Normalization

Positionally compiled GRanges objects can be used as input for normalization. In this package we have implemented three normalization functions:

- `dtcr()`, which performs ΔTCR normalization as described in (Kielpinski and Vinther, 2014),
- `slograt()`, which calculates smooth log-ratio as described in (Wan et al., 2014),
- `swinsor()`, which calculates smooth Winsor normalization. First it calculates Winsorized values as described in (Rouskin et al., 2013) but in 1-nt sliding windows, and then for each nucleotide it returns mean and standard deviation of all predictions that overlapped given nucleotide.

Single *GRanges* object can hold data normalized by all of the abovementioned methods - usually the first call of normalization functions will create the GRanges object and subsequent calls will add the data to already existing object (via `add_to` option). Let's normalize HRF-Seq data with all three methods:

```
> hrfseq_norm <- dtcr(control_GR=control_comp, treated_GR=treated_comp,
+                     window_size=3, nt_offset=1)
```

```
> hrfseq_norm <- slograt(control_GR=control_comp, treated_GR=treated_comp,
+                        add_to=hrfseq_norm)
```

```
> hrfseq_norm <- swinsor(Comp_GR=treated_comp, add_to=hrfseq_norm)
```

One can add data from compiled *GRanges* object (TC, Cover, TCR) to normalized *GRanges* object with `compdata()` function:

```
> hrfseq_norm <- compdata(Comp_GR=treated_comp, add_to=hrfseq_norm)
```

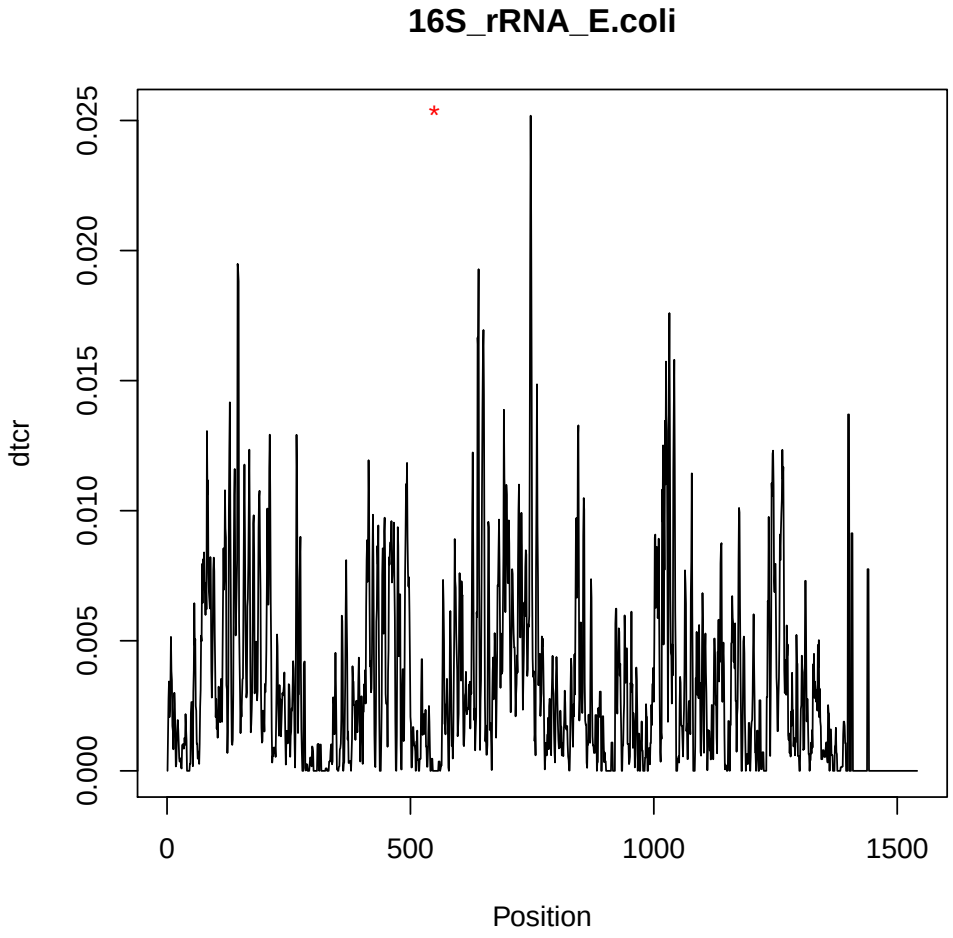
8 Export

The package allows for data export in multiple formats. First, let say we are interested in obtaining the slograt normalized table for RNase_P. Simply type:

```
> norm_df <- GR2norm_df(hrfseq_norm, RNAid = "RNase_P", norm_methods = "slograt")
```

Or, we could make a plot of ΔTCR values over 16S rRNA:

```
> plotRNA(norm_GR=hrfseq_norm, RNAid="16S_rRNA_E.coli", norm_method="dtcr")
```



At last, if we want to visualize the data in UCSC Genome Browser we can generate the BedGraph file:

```
> RNase_P_BED <- system.file("extdata", "RNaseP.bed", package="RNAprobR")
> norm2bedgraph(hrfseq_norm, bed_file = RNase_P_BED, norm_method = "dtcr",
+               genome_build = "baciSubt2", bedgraph_out_file = "RNaseP_dtcr",
+               track_name = "dtcr", track_description = "deltaTCR Normalization")
```

9 Session Info

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: OS X El Capitan 10.11.6

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

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

other attached packages:
 [1] RNAprobR_1.16.0        plyr_1.8.4             GenomicFeatures_1.36.0
 [4] AnnotationDbi_1.46.0   Biobase_2.44.0         GenomicRanges_1.36.0
 [7] GenomeInfoDb_1.20.0    IRanges_2.18.0         S4Vectors_0.22.0
[10] BiocGenerics_0.30.0

loaded via a namespace (and not attached):
 [1] SummarizedExperiment_1.14.0 progress_1.2.0
 [3] xfun_0.6                    lattice_0.20-38
 [5] htmltools_0.3.6             rtracklayer_1.43.3
 [7] yaml_2.2.0                  blob_1.1.1
 [9] XML_3.98-1.19               rlang_0.3.4
[11] DBI_1.0.0                   BiocParallel_1.18.0
[13] bit64_0.9-7                 matrixStats_0.54.0
[15] GenomeInfoDbData_1.2.1      stringr_1.4.0
[17] zlibbioc_1.30.0             Biostrings_2.52.0
[19] memoise_1.1.0               evaluate_0.13
[21] knitr_1.22                  biomaRt_2.40.0
[23] Rcpp_1.0.1                  BiocManager_1.30.4
[25] DelayedArray_0.10.0         XVector_0.24.0
[27] bit_1.1-14                  Rsamtools_2.0.0
[29] BiocStyle_2.12.0            hms_0.4.2
[31] digest_0.6.18               stringi_1.4.3
[33] grid_3.6.0                  tools_3.6.0
[35] bitops_1.0-6                magrittr_1.5
[37] RCurl_1.95-4.12             RSQLite_2.1.1
[39] crayon_1.3.4                pkgconfig_2.0.2
[41] Matrix_1.2-17               prettyunits_1.0.2
[43] assertthat_0.2.1            rmarkdown_1.12
[45] httr_1.4.0                  R6_2.4.0
[47] GenomicAlignments_1.20.0    compiler_3.6.0
```

References

[1] Jeremy Goecks, Anton Nekrutenko, and James Taylor. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome biology*, 11(8):R86, January 2010. URL: http://genomebiology.com/2010/11/8/R86, doi:10.1186/gb-2010-11-8-r86.

[2] Lukasz Jan Kielpinski and Jeppe Vinther. Massive parallel-sequencing-based hydroxyl radical probing of RNA accessibility. *Nucleic acids research*, 42(8):e70, April 2014. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4005689&tool=pmcentrez&rendertype=abstract, doi:10.1093/nar/gku167.

[3] Glenn K Fu, Jing Hu, Pei-Hua Wang, and Stephen P A Fodor. Counting individual DNA molecules by the stochastic attachment of diverse labels. *Proceedings of the National Academy of Sciences of the United States of America*, 108(22):9026–31, May 2011. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3107322&tool=pmcentrez&rendertype=abstract, doi:10.1073/pnas.1017621108.