Genome-wide assessment of differential translations with ribosome profiling data – the xtail package

Zhengtao Xiao[1−3], Qin Zou[1,3], Yu Liu[1−3], and Xuerui Yang[1−3]

[1]MOE Key Laboratory of Bioinformatics,

[2]Tsinghua-Peking Joint Center for Life Sciences,

[3]School of Life Sciences, Tsinghua University, Beijing 100084, China.

May 10, 2016

xtail version: 1.1.5

If you use xtail in published research, please cite:
Z. Xiao, Q. Zou, Y. Liu, X. Yang: **Genome-wide assessment of differential translations with ribosome profiling data**. *Nat Commun* 2016, **7**:11194.
http://www.ncbi.nlm.nih.gov/pubmed/27041671

1 Introduction

This package, Xtail, is for identification of genes undergoing differential translation across two conditions with ribosome profiling data. Xtail is based on a simple assumption that if a gene is subjected to translational dyresgulation under certain exprimental or physiological condition, the change of its RPF abundance should be discoordinated with that of mRNA expression. Specifically, `Xtail` consists of three major steps: (1) modeling of ribosome profiling data using negative binomial distribution (NB), (2) estabilishment of probability distributions for fold changes of mRNA or RPF (or RPF-to-mRNA ratios), and (3) evaluation of statistical significance and magnitude of differential translations. The differential translation of each gene is evaluated by two pipelines: in the first one, `Xtail` calculated the posterior probabilities for a range of mRNA or RPF fold changes, and eventually estabilished their probability distributions. These two distributions, represented as probability vectors, were then used to estabilish a joint probability distribution matrix, from which a new probability distribution were generated for differential translation. The P-values, point estimates and credible intervals of differential tranlsations were then calculated based on these results. In the other parallel pipline, `Xtail` established probability distributions for RPF-to-mRNA ratios in two conditions and derived another distribution for differential translation. The more conserved set of results from these two parallel piplines was used as the final result. With this strategy, `Xtail` performs quantification of differential translation for each gene, i.e., the extent to which a gene's translational rate is not coordinated with the change of the mRNA expression.

By default, `Xtail` adapts the strategy of DESeq2 [1] to normalize read counts of mRNA and RPF in all samples, and fits NB distributions with dispersions α and μ.

This guide provides step-by-step instructions on how to load data, how to excute the package and how to interpret output.

2 Data Preparation

The `Xtail` package uses read counts of RPF and mRNA, in the form of rectangular table of values. The rows and columns of the table represent the genes and samples, respectively. Each cell in the *g-th* row and the *i-th* columns is the count number of reads mapped to gene *g* in sample *i*.

Xtail takes in raw read counts of RPF and mRNA, and performs median-of-ratios normalization by default. This normalization method is also recommend by Reddy R. [2]. Alternatively, users can provide normalized read counts and skip the built-in normalization in Xtail.

In this vignette, we select a published ribosome profiling dataset from human prostate cancer cell PC3 after mTOR signaling inhibition with PP242 [3]. This dataset consists of mRNA and RPF data for 11391 genes in two replicates from each of the two conditions("treatment" vs. "control").

3 An Example

Here we run `Xtail` with the ribosome profiling data described above. First we load the library and data.

```
library(xtail)
data(xtaildata)
```

Next we can view the first five lines of the mRNA (`mrna`) and RPF (`rpf`) elements of `xtaildata`.

```
mrna <- xtaildata$mrna
rpf <- xtaildata$rpf
head(mrna,5)

##                 control1 control2 treat1 treat2
## ENSG00000000003      825      955    866   1039
## ENSG00000000419     1054      967    992    888
## ENSG00000000457       71       75    139     95
## ENSG00000000460      191      162    199    201
## ENSG00000000971       81        2     88     11

head(rpf,5)

##                 control1 control2 treat1 treat2
## ENSG00000000003      143      302    197    195
## ENSG00000000419      234      481    383    306
## ENSG00000000457       12       17     17     15
## ENSG00000000460       45       88     63     37
## ENSG00000000971       31        7     36      2
```

We assign condition labels to the columns of the mRNA and RPF data.

```
condition <- c("control","control","treat","treat")
```

Next, we run the main function, `xtail()`. By default, the second condition (here is "treat") would be compared against the first condition (here is "control"). Those genes with the minimum average expression of mRNA counts and RPF counts among all samples larger than 1 are used (can be changed by setting `minMeanCount`). All the available CPU cores are used for running program. The argument `"bins"` is the number of bins used for calculating the probability densities of log2FC and log2R. This paramater will determine accuracy of the final pvalue. Here, in order to keep the run-time of this vignette short, we will set `bins` to "1000". Detailed description of the arguments of the `xtail` function can be found by typing `?xtail` or `help(xtail)` at the **R** prompt.

```
test.results <- xtail(mrna,rpf,condition,bins=1000)
```

We can summarize some basic information of xtail results using the summary function (type `?summary` for further information).

```
summary(test.results)

##
## The total number of gene is: 11391
## Number of the log2FC and log2R used in determining the final p-value:
##      log2FC: 9192
##      log2R : 2199
##
## adjusted pvalue <  0.1
##      log2FC_TE > 0 (up)   : 2
##      log2FC_TE < 0 (down): 108
```

Now we can extract a results table using the function `resultsTable`, and examine the first five lines of the results table.

```
test.tab <- resultsTable(test.results)
head(test.tab,5)

##                 log2FC_TE_v1 pvalue_v1 log2FC_TE_v2 pvalue_v2 log2FC_TE_final
## ENSG00000000003    0.0623457 0.8079550   0.05867991 0.8162414      0.05867991
## ENSG00000000419    0.3592684 0.1360275   0.35599462 0.1536697      0.35599462
## ENSG00000000457   -0.2897093 0.6409663  -0.29931892 0.5865027     -0.28970931
## ENSG00000000460   -0.3109160 0.4249511  -0.31079662 0.4203029     -0.31091597
## ENSG00000000971   -0.6457720 0.7443768  -0.62232332 0.7105031     -0.64577202
##                 pvalue_final pvalue.adjust
## ENSG00000000003    0.8162414     0.9957191
## ENSG00000000419    0.1536697     0.9957191
## ENSG00000000457    0.6409663     0.9957191
## ENSG00000000460    0.4249511     0.9957191
## ENSG00000000971    0.7443768     0.9957191
```

The results of fist pipline are named with suffix "_v1", which are generated by comparing mRNA and RPF log2 fold changes: The element `log2FC_TE_v1` represents the log2 fold change of TE; The `pvalue_v1` represent statistical significance. The sencond pipline are named with suffix "_v2", which are derived by comparing log2 ratios between two conditions: `log2FC_TE_v2`, and `pvalue_v2` are log2 ratio of TE, and pvalues. Finally, the more conserved results (with larger-Pvalue) was select as the final assessment of differential translation, which are named with suffix "_final". The `pvalue.adjust` is the estimated false discovery rate corresponding to the `pvalue_final`.

Users can also get the log2 fold changes of mRNA and RPF, or the log2 ratios of two conditions by setting "log2FCs" or "log2Rs" as "TRUE" in resultsTable. And the results table can be sorted by assigning the "sort.by". Detailed description of the `resultsTable` function can be found by typing `?resultsTable`.

Finally, the plain-text file of the results can be exported using the functions *write.csv* or *write.table*.

```
write.table(test.tab,"test_results.txt",quote=F,sep="\t")
```

We also provide a very simple function, `write.xtail` (using the write.table function), to export the `xtail` result (test.results) to a tab delimited file.

```
write.xtail(test.results,"test_results.txt",quote=F,sep="\t")
```

4 Visualization

4.1 plotFCs

In `Xtail`, the function `plotFCs` shows the result of the differential expression at the two expression levels, where each gene is a dot whose position is determined by its log2 fold change (log2FC) of transcriptional level (`mRNA_log2FC`), represented on the x-axis, and the log2FC of translational level (`RPF_log2FC`), represented on the y-axis (Figure 1). The optional input parameter of `plotFCs` is `log2FC.cutoff`, a non-negative threshold value that will divide the genes into different classes:

- `blue`: for genes whoes `mRNA_log2FC` larger than `log2FC.cutoff` (transcriptional level).
- `red`: for genes whoes `RPF_log2FC` larger than `log2FC.cutoff` (translational level).
- `green`: for genes changing homodirectionally at both level.
- `yellow`: for genes changing antidirectionally at two levels.

```
plotFCs(test.results)
```

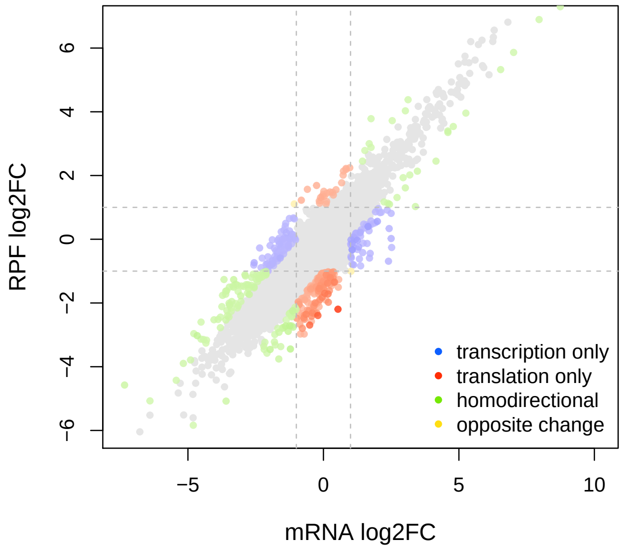

Figure 1: Scatter plot of log2 fold changes

Those genes in which the difference of `mRNA_log2FC` and `RPF_log2FC` did not exceed more than `log2FC.cutoff` are excluded. The points will be color-coded with the `pvalue_final` obtained with `xtail` (more significant p values having darker color). By default the `log2FC.cutoff` is 1.

4.2 volcanoPlot

It can also be useful to evaluate the fold changes cutoff and p values thresholds by looking at the volcano plot. A simple function for making this plot is `volcanoPlot`, in which the `log2FC_TE_final` is plotted on the x-axis and the negative log10 `pvalue_fianl` is plotted on the y-axis (Figure 2).

```
volcanoPlot(test.results)
```

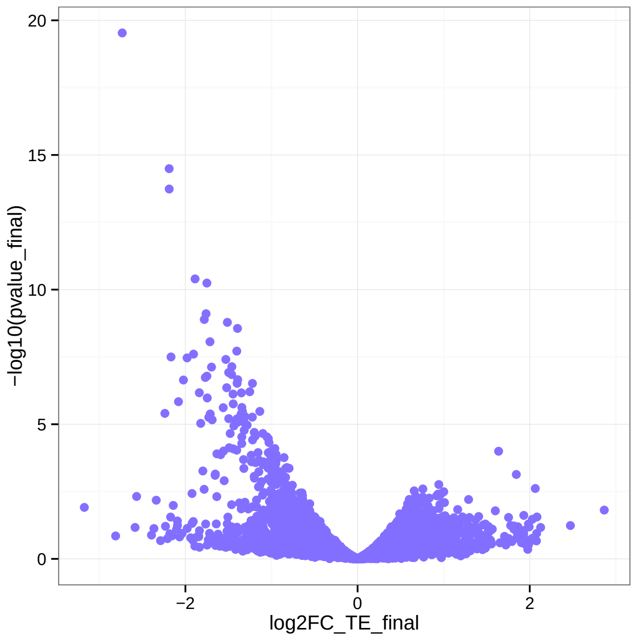

Figure 2: volcano plot.

Session Info

```
sessionInfo()

## R version 3.2.4 (2016-03-10)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.11.4 (El Capitan)
##
## locale:
## [1] C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets  methods
## [9] base
##
## other attached packages:
##  [1] ggplot2_1.0.1              scales_0.3.0               xtail_1.1.5
##  [4] DESeq2_1.10.1              RcppArmadillo_0.6.100.0.0  Rcpp_0.12.1
##  [7] SummarizedExperiment_1.0.0 Biobase_2.30.0             GenomicRanges_1.22.0
## [10] GenomeInfoDb_1.6.0         IRanges_2.4.1              S4Vectors_0.8.0
## [13] BiocGenerics_0.16.0        knitr_1.11
##
## loaded via a namespace (and not attached):
##  [1] RColorBrewer_1.1-2   formatR_1.2.1        futile.logger_1.4.1  highr_0.5.1
##  [5] plyr_1.8.3           XVector_0.10.0       futile.options_1.0.0 tools_3.2.4
##  [9] zlibbioc_1.16.0      rpart_4.1-10         digest_0.6.8         RSQLite_1.0.0
## [13] annotate_1.48.0      evaluate_0.8         gtable_0.1.2         lattice_0.20-33
## [17] DBI_0.3.1            proto_0.3-10         gridExtra_2.0.0      genefilter_1.52.0
## [21] cluster_2.0.3        stringr_1.0.0        locfit_1.5-9.1       nnet_7.3-12
## [25] grid_3.2.4           AnnotationDbi_1.32.0 XML_3.98-1.3         survival_2.38-3
## [29] BiocParallel_1.4.0   foreign_0.8-66       latticeExtra_0.6-26  Formula_1.2-1
## [33] geneplotter_1.48.0   reshape2_1.4.1       lambda.r_1.1.7       magrittr_1.5
## [37] Hmisc_3.17-0         codetools_0.2-14     splines_3.2.4        MASS_7.3-45
## [41] xtable_1.7-4         BiocStyle_1.8.0      colorspace_1.2-6     labeling_0.3
## [45] stringi_1.0-1        acepack_1.3-3.3      munsell_0.4.2
```

References

[1] Love MI, Huber W, Anders S: *Moderated Estimation of Fold Change and Dispersion for RNA-Seq Data with DESeq2*. Genome Biology 2014, 15:550. A Comparison of Methods: Normalizing High-Throughput RNA Sequencing Data.

[2] Reddy R: *A Comparison of Methods: Normalizing High-Throughput RNA Sequencing Data. Cold Spring Harbor Labs Journals*. bioRxiv 2015:1-9.

[3] Hsieh AC, Liu Y, Edlind MP, et al.: *The translational landscape of mTOR signaling steers cancer initiation and metastasis*. Nature 2012, 485:55-61.