Contents

1 Introduction 2
2 Installation 2
3 Workflow 3
4 Session Information 5
References 6
1 Introduction

miRNAtap package is designed to facilitate implementation of workflows requiring miRNA prediction. Aggregation of commonly used prediction algorithm outputs in a way that improves on performance of every single one of them on their own when compared against experimentally derived targets. microRNA (miRNA) is a 18-22nt long single strand that binds with RISC (RNA induced silencing complex) and targets mRNAs effectively reducing their translation rates.

Targets are aggregated from 5 most commonly cited prediction algorithms: DIANA (Maragkakis et al., 2011), Miranda (Enright et al., 2003), PicTar (Lall et al., 2006), TargetScan (Friedman et al., 2009), and miRDB (Wong and Wang, 2015).

Programmatic access to sources of data is crucial when streamlining the workflow of our analysis, this way we can run similar analysis for multiple input miRNAs or any other parameters. Not only does it allow us to obtain predictions from multiple sources straight into R but also through aggregation of sources it improves the quality of predictions.

Finally, although direct predictions from all sources are only available for Homo sapiens and Mus musculus, this package includes an algorithm that allows to translate target genes to other speices (currently only Rattus norvegicus) using homology information where direct targets are not available.

2 Installation

This section briefly describes the necessary steps to get miRNAtap running on your system. We assume that the user has the R program (see the R project at http://www.r-project.org) already installed and is familiar with it. You will need to have R 3.2.0 or later to be able to install and run miRNAtap. The miRNAtap package is available from the Bioconductor repository at http://www.bioconductor.org To be able to install the package one needs first to install the core Bioconductor packages. If you have already installed Bioconductor packages on your system then you can skip the two lines below.

> source("http://bioconductor.org/biocLite.R")
> biocLite()

Once the core Bioconductor packages are installed, we can install the miRNAtap and accompanying database miRNAtap.db package by

> source("http://bioconductor.org/biocLite.R")
> biocLite("miRNAtap")
> biocLite("miRNAtap.db")
3 Workflow

This section explains how miRNAtap package can be integrated in the workflow aimed at predicting which processes can be regulated by a given microRNA.

In this example workflow we’ll use miRNAtap as well as another Bioconductor package topGO together with Gene Ontology (GO) annotations. In case we don’t have topGO or GO annotations on our machine we need to install them first:

```r
> source("http://bioconductor.org/biocLite.R")
> biocLite("topGO")
> biocLite("org.Hs.eg.db")
```

Then, let’s load the required libraries

```r
> library(miRNAtap)
> library(topGO)
> library(org.Hs.eg.db)
```

Now we can start the analysis. First, we will obtain predicted targets for human miRNA miR-10b

```r
> mir = 'miR-10b'
> predictions = getPredictedTargets(mir, species = 'hsa',
+ method = 'geom', min_src = 2)
```

Let’s inspect the top of the prediction list.

```r
> head(predictions)

  source_1 source_2 source_3 source_4 source_5 rank_product rank_final
627     103    10.0     1.0      NA       1     1.416281       1
79741   NA      NA     8.0      2       NA     2.000000       2
6095     5     2.5   73.5      NA       5     2.058173       3
348980   NA     2.5  20.0      NA      NA     3.535534       4
51365   NA    53.0    3.0     12      27     3.766392       5
7022    88    17.5    5.0    149       3     4.058725       6
```

We are using geometric mean aggregation method as it proves to perform best when tested against experimental data from MirBase (Griffiths-Jones et al., 2008).

We can compare it to the top of the list of the output of minimum method:

```r
> predictions_min = getPredictedTargets(mir, species = 'hsa',
+ method = 'min', min_src = 2)
> head(predictions_min)
```
Where predictions for rat genes are not available we can obtain predictions for mouse genes and translate them into rat genes through homology. The operation happens automatically if we specify species as `rno` (for *Rattus norvegicus*):

```r
> predictions_rat = getPredictedTargets(mir, species = 'rno', +   method = 'geom', min_src = 2)
```

Now we can use the ranked results as input to GO enrichment analysis. For that we will use our initial prediction for human *miR-10b*:

```r
> rankedGenes = predictions[, 'rank_product']
> selection = function(x) TRUE
> # we do not want to impose a cut off, instead we are using rank information
> allGO2genes = annFUN.org(whichOnto='BP', feasibleGenes = NULL, +   mapping="org.Hs.eg.db", ID = "entrez")
> GOdata = new('topGOdata', ontology = 'BP', allGenes = rankedGenes, +   annot = annFUN.GO2genes, GO2genes = allGO2genes, +   geneSel = selection, nodeSize=10)
```

In order to make use of the rank information we will use Kolomorogorov-Smirnov (K-S) test instead of Fisher exact test which is based only on counts.

```r
> results.ks = runTest(GOdata, algorithm = "classic", statistic = "ks")
> results.ks
```

Description:

Ontology: BP

'classic' algorithm with the 'ks' test

611 GO terms scored: 4 terms with p < 0.01

Annotation data:

- Annotated genes: 340
- Significant genes: 340
- Min. no. of genes annotated to a GO: 10
- Nontrivial nodes: 611

We can view the most enriched GO terms (and potentially feed them to further steps in our workflow):

```r
> allRes = GenTable(GOdata, KS = results.ks, orderBy = "KS", topNodes = 20)
> allRes[,c('GO.ID', 'Term', 'KS')]
```
For more details about GO analysis refer to topGO package vignette (Alexa and Rahnenfuhrer, 2010).

Finally, we can use our predictions in a similar way for pathway enrichment analysis based on KEGG (Kanehisa and Goto, 2000), for example using Bioconductor’s KEGGprofile (Zhao, 2012).

4 Session Information

- R version 3.4.0 (2017-04-21), x86_64-pc-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Running under: Ubuntu 16.04.2 LTS
- Matrix products: default
- BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
- LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
Other packages: AnnotationDbi 1.38.0, Biobase 2.36.0, BiocGenerics 0.22.0, GO.db 3.4.1, IRanges 2.10.0, S4Vectors 0.14.0, SparseM 1.77, graph 1.54.0, miRAnalyzer 1.10.0, miRAnalyzer.db 0.99.10, org.Hs.eg.db 3.4.1, topGO 2.28.0

Loaded via a namespace (and not attached): DBI 0.6-1, RSQLite 1.1-2, Rcpp 0.12.10, chron 2.3-50, compiler 3.4.0, digest 0.6.12, grid 3.4.0, gsubfn 0.6-6, lattice 0.20-35, magrittr 1.5, matrixStats 0.52.2, memoise 1.1.0, plyr 1.8.4, proto 1.0.0, sqldf 0.4-10, stringi 1.1.5, stringr 1.2.0, tools 3.4.0

References


