miRNAtap example use

Maciej Pajak, Ian Simpson

October 17, 2016

Contents

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 (**?**), Miranda (**?**), PicTar (**?**), TargetScan (**?**), and miRDB (**?**).

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:

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

Then, let's load the required libraries

```
> 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*

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

Let's inspect the top of the prediction list.

```
> 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 (**?**).

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

```
> predictions_min = getPredictedTargets(mir, species = 'hsa',
+                                   method = 'min', min_src = 2)
> head(predictions_min)
```

```
      source_1 source_2 source_3 source_4 source_5 rank_product rank_final
627        103       10      1.0       NA        1            1        2.0
8897         1      183    282.0       NA       NA            1        2.0
79042       NA      107     99.5        1       NA            1        2.0
7182         2       NA       NA       NA      106            2        5.5
10739       NA       42      2.0       NA       NA            2        5.5
79741       NA       NA      8.0        2       NA            2        5.5
```

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*)

```
> 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*

```
> 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 Kolomonogorov Smirnov (K-S) test instead of Fisher exact test which is based only on counts.

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

```
Description:
Ontology: BP
'classic' algorithm with the 'ks' test
599 GO terms scored: 5 terms with p < 0.01
Annotation data:
    Annotated genes: 335
    Significant genes: 335
    Min. no. of genes annotated to a GO: 10
    Nontrivial nodes: 599
```

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

```
> allRes = GenTable(GOdata, KS = results.ks, orderBy = "KS", topNodes = 20)
> allRes[,c('GO.ID','Term','KS')]
```

```
        GO.ID                                       Term     KS
1  GO:0042692                muscle cell differentiation 0.0016
2  GO:0050789           regulation of biological process 0.0023
3  GO:0048518 positive regulation of biological proces... 0.0052
4  GO:0050794             regulation of cellular process 0.0079
5  GO:0065007                      biological regulation 0.0083
6  GO:0044087 regulation of cellular component biogene... 0.0224
```

```
7  GO:0006352     DNA-templated transcription, initiation 0.0247
8  GO:0006367 transcription initiation from RNA polyme... 0.0247
9  GO:0045944 positive regulation of transcription fro... 0.0252
10 GO:0048522     positive regulation of cellular process 0.0263
11 GO:0061061                muscle structure development 0.0264
12 GO:0014070         response to organic cyclic compound 0.0316
13 GO:0010604 positive regulation of macromolecule met... 0.0360
14 GO:0031325 positive regulation of cellular metaboli... 0.0374
15 GO:0080090     regulation of primary metabolic process 0.0401
16 GO:0043254      regulation of protein complex assembly 0.0470
17 GO:0045893 positive regulation of transcription, DN... 0.0483
18 GO:1902680 positive regulation of RNA biosynthetic ... 0.0483
19 GO:1903508 positive regulation of nucleic acid-temp... 0.0483
20 GO:0010629      negative regulation of gene expression 0.0515
```

For more details about GO analysis refer to `topGO` package vignette (**?**).

Finally, we can use our predictions in a similar way for pathway enrichment analysis based on KEGG (**?**), for example using Bioconductor's `KEGGprofile` (**?**).

4 Session Information

- R version 3.3.1 (2016-06-21), `x86_64-apple-darwin13.4.0`
- Locale: `C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: AnnotationDbi 1.36.0, Biobase 2.34.0, BiocGenerics 0.20.0, GO.db 3.4.0, IRanges 2.8.0, S4Vectors 0.12.0, SparseM 1.72, graph 1.52.0, miRNAtap 1.8.0, miRNAtap.db 0.99.10, org.Hs.eg.db 3.4.0, topGO 2.26.0
- Loaded via a namespace (and not attached): DBI 0.5-1, RSQLite 1.0.0, Rcpp 0.12.7, chron 2.3-47, grid 3.3.1, gsubfn 0.6-6, lattice 0.20-34, magrittr 1.5, matrixStats 0.51.0, plyr 1.8.4, proto 0.3-10, sqldf 0.4-10, stringi 1.1.2, stringr 1.1.0, tools 3.3.1