miRNAmeConverter-vignette

Stefan Haunsberger

2016-05-15

Contents

The miRBase database (Griffiths-Jones, 2004; Griffiths-Jones, Grocock, Dongen, Bateman, & Enright, 2006; Griffiths-Jones, Saini, Dongen, & Enright, 2008; Kozomara & Griffiths-Jones, 2011, 2014) is the official repository for miRNAs and includes a miRNA naming convention (AMBROS et al., 2003; Meyers et al., 2008). Over the years of development miRNAs have been added to, or deleted from the database, while some miRNA names have been changed. As a result, each version of the miRBase database can differ substantially from previous versions. If working with just one or two miRNAs, these can be manually searched on the miRBase website. With larger sets of miRNAs, however, this work becomes labour intensive and prone to mistakes. When working with a set of miRNAs, therefore, useful information includes

- if a certain miRNA name exists in the miRBase database.
- if a given miRNA is still considered to be a miRNA.
- the miRNA names from the most recent miRBase version
- the respective miRNA name from a different miRBase version, for using other services, such as miRNA target databases.

The *miRNAmeConverter* R package has been developed for handling naming challenges of mature miRNAs.

In addition we have developed a web interface that enables one to use the `translateMiRNAName` function via web interface and is based on shiny (miRNAmeConverter-web).

The application runs on port 3838, so if the application does not show up please make sure your device is able to connect through port 3838.

1. Introduction

The *miRNAmeConverter* package delivers results in a fast and transparent way. The main functions

- check for validity of miRNA names,

- determine the most likely miRBase version of a given set of miRNAs and
- translate miRNA names to different versions.

The core function, translating miRNA names to different versions, is especially useful when dealing with miRNA tools other than miRBase. To retrieve targets from miRTarBase, for example, the miRNA name from version 20 is required, whereas the website miRecords only accepts version 17. The miRNAmeConverter can manage large sets of miRNA names and hence can be easily implemented into workflows.

The data set included in the package (*miRNAs*) is a collection of human miRNA names. It consists of valid miRNA names (some duplicates), incorrect names and features used as controls in experiments, such as the RNU44 as a house keeping gene for HT-qPCR assay plates from Applied Biosystems.

To load the package and gain access to the functions and sample dataset of the miRNAmeConverter package just run the following command:

```
library(miRNAmeConverter)
```

```
## Loading required package: miRBaseVersions.db
```

Vignette Info

This vignette has been generated using an R Markdown file with `knitr:rmarkdown` as vignette engine (Boettiger, 2015; Francois, 2014; Xie, 2014, 2015b, 2015a).

Database information

The data used in the *miRNAmeConverter* package is obtained from the `miRBaseVersions.db` annotation package (Haunsberger, 2016).

miRNA name input values

According to the nomenclature used in the miRBase repository `hsa-mir-29a` is a stem-loop sequence name. If we substitute the 'r' by a capital 'R' it codes for the mature 3' miRNA `hsa-miR-29a` (or `hsa-miR-29a-5p` in the current version). The `miRNAs` input value for the functions has to be in form of a `character` vector. Algorithms of the package are **not** case sensitive. This has the effect, that for example in the case `hsa-mir-29a` and `hsa-miR-29a` as input values both will be considered as valid mature miRNA names.

2. The MiRNANameConverter class

The MiRNANameConverter class is coded in S4 style. All functions offered by the miRNAmeConverter package are methods of that class. The following figure is a simplified class diagram with the exported methods and class attributes:

All methods can be displayed by

```
ls("package:miRNAmeConverter");
```

```
##  [1] "MiRNANameConverter" "assessVersion"      "checkMiRNAName"
##  [4] "currentVersion"     "example.miRNAs"     "nOrganisms"
##  [7] "nTotalEntries"      "saveResults"        "show"
## [10] "translateMiRNAName" "validOrganisms"     "validVersions"
```

The slot names (attributes) of the class can be displayed by

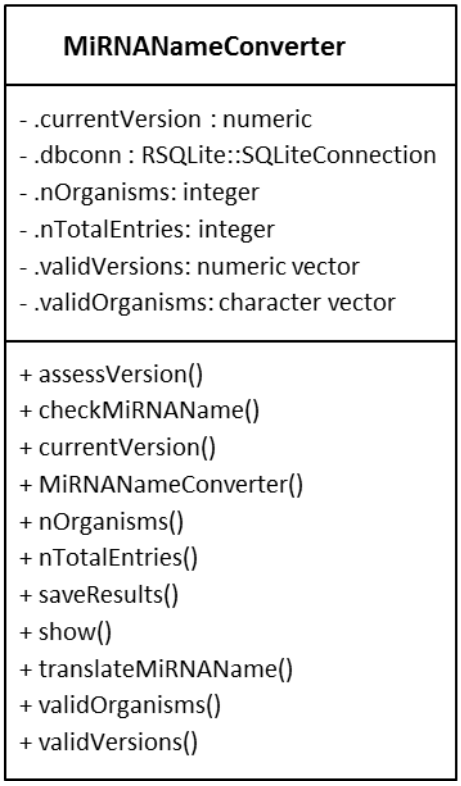

Figure 1: class-diagram

```
slotNames("MiRNANameConverter");
```

```
## [1] ".dbconn"         ".currentVersion" ".validVersions"  ".nOrganisms"
## [5] ".nTotalEntries"  ".validOrganisms"
```

An instance of the `MiRNANameConverter` class can be created by calling the `MiRNANameConverter` function:

```
MiRNANameConverter();
```

```
## An object of class: MiRNANameConverter
## - Most recent miRBase version provided in the package:[1] 21
## - Valid miRBase versions:  [1] 21.0 20.0 19.0 18.0 17.0 16.0 15.0 14.0 13.0 12.0 11.0 10.1 10.0  9.2
## [15]  9.1  9.0  8.2  8.1  8.0  7.1  6.0
## - Number of species: [1] 228
## - Number of database entries among all versions: [1] 242728
```

3. Use Cases

3.1 Check for valid miRNA name

Valid input values

We have given a set of mature miRNA names `miRNAs` and would like to check if the names are actual miRNA names that are or were listed in any miRBase version. Our `miRNAs` have the following values:

```
miRNAs = c("hsa-miR-422b", "mmu-mir-872", "gra-miR157b", "cel-miR-56*");
```

Our first step is to create an instance of the `MiRNANameConverter` class by calling the constructor and saving it to the variable `nc`.

```
nc = MiRNANameConverter();
```

Now we check if the names are valid names listed in any of the provided miRBase versions:

```
checkMiRNAName(nc, miRNAs);
```

```
## [1] "cel-miR-56*"  "gra-miR157b"  "hsa-miR-422b" "mmu-miR-872"
```

The function `checkMiRNAName()` has a parameter called `correct`, which is set to `TRUE` by default. The input name `mmu-mir-872` has therefore been
corrected to `mmu-miR-872`. If we run the same command again with `correct = FALSE`, the output name will be equal to the input name.

```
checkMiRNAName(nc, miRNAs, correct = FALSE);
```

```
## [1] "hsa-miR-422b" "mmu-mir-872"  "gra-miR157b"  "cel-miR-56*"
```

Mixed input values

The following set of miRNA names contains valid as well as invalid names.

```
# "RNU-6" and "bpcv1-miR-B23" are not valid
miRNAs = c("hsa-miR-422b", "RNU-6", "mmu-miR-872", "gra-miR157b", "bpcv1-miR-B23", "bpcv1-miR-B1");
nc = MiRNANameConverter();    # Create MiRNANameConverter object
checkMiRNAName(nc, miRNAs);   # check names
```

```
##  The following miRNAs are not listed in the miRBase repository:

## RNU-6; bpcv1-miR-B23

## [1] "bpcv1-miR-B1" "gra-miR157b"  "hsa-miR-422b" "mmu-miR-872"
```

This time the function prints information for the features that did not pass the check and therefore are not included in the return value.

3.2 Assess most likely miRBase version

Sometimes it is useful to know the miRBase version that a given set of miRNA names is from. In this case one can use the `assessVersion` function to receive the most likely miRBase version. The following example makes use of the provided `example-miRNAs`-dataset.

```
nc = MiRNANameConverter();              # Create MiRNANameConverter object
assessVersion(nc, example.miRNAs);      # Assess most likely miRBase version
```

```
##  Input: 355 unique miRNA names

##  The following miRNAs are not listed in the miRBase repository:

## RNU48; RNU44; RNU6B; hsa-miR-517; 4343438

##  miRNAs that could not be found in identified max version 9.0:

## hsa-miR-213; hsa-miR-425; hsa-miR-493

## -> 347 out of 350 valid miRNAs assigned to max version 9.0.

##    version frequency
## 1      9.0       347
## 2      8.2       347
## 3      8.1       347
## 4      9.2       346
## 5      9.1       346
## 6     10.0       276
## 7     10.1       272
## 8     14.0       271
## 9     13.0       271
## 10    12.0       271
## 11    11.0       271
## 12    15.0       270
## 13    16.0       267
## 14     8.0       267
## 15    17.0       266
## 16     7.1       264
## 17     6.0       180
## 18    18.0       133
## 19    19.0       127
## 20    20.0        93
## 21    21.0        92
```

The console output shows that from the 384 input names there were 355 unique values. Five of these values are not listed in any miRBase version and were neglected. The return value is a data.frame object with the two columns `version` and `frequency`. It is ordered by frequency and version. It shows that 347 out of 350 valid input miRNAs could be assigned to miRBase version 9.0. This is the highest score and therefore the most likely miRBase version of the input mature miRNA names.

3.3 Translate miRNA name(s)

Translating miRNA names to different versions is the most required function. Let us assume we found a paper that shows that the miRNA `"hsa-miR-422b"` is significantly differentially expressed under certain conditions. Assume further, that we did a similar experiment but this miRNA does not show up in our analysis. Instead we got `"hsa-miR-378a-3p"` from miRBase version 21. We see, that the previous paper was released a couple of years ago so there might be a chance that their miRNA could now run under a different name. We apply the `translateMiRNAName` function with `hsa-miR-422b` as miRNA parameter and no version. With no given version the function returns the miRBase version 21 by default.

```
miRNA.paper = "hsa-miR-422b";
nc = MiRNANameConverter();              # Create MiRNANameConverter object
translateMiRNAName(nc, miRNA.paper);    # Translate miRNA names
```

```
##                      mimat        input           v21.0
## MIMAT0000732 MIMAT0000732 hsa-miR-422b hsa-miR-378a-3p
```

The result shows that these two miRNAs are actually the same. This is because `"hsa-miR-422b"` is last listed in miRBase version 9.2. In version 10.0 it was named `"hsa-miR-378"` which in the current version 21.0 runs under the name `"hsa-miR-378a-3p"`.

Another example with more diverse input names is shown below, with the respective console output and the entry in the attribute `description`.

```
miRNAs = c("hsa-miR-128b", "ebv-miR-BART3-5p", "mmu-miR-302b*",
           "mmu-miR-872", "ebv-miR-BART5", "bpcv1-miR-B23");
nc = MiRNANameConverter();                # Create MiRNANameConverter object
result = translateMiRNAName(nc, miRNAs,
                            versions = c(17, 19, 21));   # Translate names
```

```
##  Following miRNA will be neglected (not listed in miRBase):
```

```
## bpcv1-miR-B23
```

```
##  Following miRNA is not listed in the current miRBase version 21.0.
```

```
## hsa-miR-128b
```

The console output shows us that one of the input values is not a miRNA (`"bpcv1-miR-B23"`) and another is not listed in miRBase version 21 (`"hsa-miR-128b"`).

```
result;
```

```
##                      mimat             input          v17.0             v19.0
## MIMAT0003373 MIMAT0003373    mmu-miR-302b*  mmu-miR-302b*  mmu-miR-302b-5p
## MIMAT0003410 MIMAT0003410 ebv-miR-BART3-5p ebv-miR-BART3* ebv-miR-BART3-5p
## MIMAT0003413 MIMAT0003413    ebv-miR-BART5  ebv-miR-BART5 ebv-miR-BART5-5p
## MIMAT0004934 MIMAT0004934      mmu-miR-872    mmu-miR-872   mmu-miR-872-5p
##                         v21.0
## MIMAT0003373  mmu-miR-302b-5p
## MIMAT0003410 ebv-miR-BART3-5p
## MIMAT0003413 ebv-miR-BART5-5p
## MIMAT0004934   mmu-miR-872-5p
```

The information, whether a miRNA is OK or not, is stored in form of an attribute of the return value of the function. This `data.frame` object provides information about every single input value and can be accessed via the `attr` command followed by `'description'`.

```
attr(result, 'description');
```

```
##                         input.miRNA
## bpcv1-miR-B23       bpcv1-miR-B23
## ebv-miR-BART3-5p ebv-miR-BART3-5p
## ebv-miR-BART5       ebv-miR-BART5
## hsa-miR-128b         hsa-miR-128b
## mmu-miR-302b*       mmu-miR-302b*
## mmu-miR-872           mmu-miR-872
##                                                                  information
## bpcv1-miR-B23                 This name is not listed in any miRBase version.
## ebv-miR-BART3-5p                                                          OK
## ebv-miR-BART5                                                             OK
## hsa-miR-128b     This miRNA is not listed in the current miRBase version 21.0.
## mmu-miR-302b*                                                             OK
## mmu-miR-872                                                               OK
```

Another option that can be passed on to the function is the Boolean parameter `current`. By setting this value to `TRUE` the `translateMiRNAName` function includes translation results from the current miRBase version in the return value (which is the highest supported version contained in the package).

3.4 Other functions

Save translation results

Sometimes it is useful to save a variable to a file. Taking the translation `result` from above we can do so by running the following command with default settings:

```
saveResults(nc, result);
```

Two files will be saved, one for the translation data frame and one for the description. By default the file is a tab-separated file without the parameter `quote` set to `FALSE`. Other options can be applied and will be passed on to the `utils::write.table` function.

Getter functions

Other functions are so called getter-functions to receive the values of the class attributes `current.version` and `valid.versions` respectively.

```
nc = MiRNANameConverter();    # Create MiRNANameConverter object
currentVersion(nc);           # Receive the maximum miRBase version included in the package
```

```
## [1] 21
```

```
validVersions(nc);            # Receive all valid versions
```

```
##  [1] 21.0 20.0 19.0 18.0 17.0 16.0 15.0 14.0 13.0 12.0 11.0 10.1 10.0  9.2
## [15]  9.1  9.0  8.2  8.1  8.0  7.1  6.0
```

```
nOrganisms(nc);               # Number of organisms
```

```
## [1] 228
```

```
validOrganisms(nc);          # Valid organisms
```

```
##   [1] "aae"   "aau"   "aca"   "aga"   "age"   "ahy"   "aja"   "aly"
##   [9] "ama"   "ame"   "amg"   "api"   "aqc"   "aqu"   "asu"   "ata"
##  [17] "ath"   "atr"   "bbe"   "bcy"   "bdi"   "bfl"   "bfv"   "bgy"
##  [25] "bhv1"  "bhv5"  "bkv"   "blv"   "bma"   "bmo"   "bna"   "bol"
##  [33] "bpcv1" "bpcv2" "bra"   "bta"   "cbn"   "cbr"   "cca"   "ccl"
##  [41] "ccr"   "cel"   "cfa"   "cgr"   "chi"   "cin"   "cla"   "cln"
##  [49] "cme"   "cpa"   "cqu"   "cre"   "crm"   "crt"   "csa"   "csi"
##  [57] "cte"   "ctr"   "dan"   "ddi"   "der"   "dev"   "dgr"   "dme"
##  [65] "dmo"   "dpe"   "dpr"   "dps"   "dpu"   "dre"   "dse"   "dsi"
##  [73] "dvi"   "dwi"   "dya"   "ebv"   "eca"   "efu"   "egr"   "egu"
##  [81] "emu"   "esi"   "far"   "fru"   "gar"   "gga"   "ggo"   "ghb"
##  [89] "ghr"   "gma"   "gpy"   "gra"   "gsa"   "gso"   "han"   "har"
##  [97] "hbr"   "hbv"   "hci"   "hcmv"  "hco"   "hex"   "hhi"   "hhv6b"
## [105] "hiv1"  "hma"   "hme"   "hpa"   "hpe"   "hru"   "hsa"   "hsv1"
## [113] "hsv2"  "htu"   "hvsa"  "hvt"   "hvu"   "iltv"  "ipu"   "isc"
## [121] "jcv"   "kshv"  "lca"   "lco"   "lgi"   "lja"   "lla"   "lmi"
## [129] "lus"   "lva"   "mcmv"  "mcv"   "mdm"   "mdo"   "mdv1"  "mdv2"
## [137] "mes"   "meu"   "mghv"  "mja"   "mml"   "mmu"   "mne"   "mse"
## [145] "mtr"   "ngi"   "nlo"   "nta"   "nve"   "nvi"   "oan"   "oar"
## [153] "ocu"   "odi"   "oha"   "ola"   "osa"   "pab"   "pbi"   "pde"
## [161] "peu"   "pgi"   "pin"   "pma"   "pmi"   "pol"   "ppa"   "ppc"
## [169] "ppe"   "ppt"   "ppy"   "pra"   "prd"   "prv"   "psj"   "pta"
## [177] "ptc"   "pti"   "ptr"   "pvu"   "pxy"   "rco"   "rgl"   "rlcv"
## [185] "rmi"   "rno"   "rrv"   "sbi"   "sci"   "sha"   "sja"   "sko"
## [193] "sla"   "sly"   "sma"   "sme"   "smo"   "smr"   "sof"   "spu"
## [201] "ssa"   "ssc"   "ssl"   "ssp"   "ssy"   "str"   "stu"   "sv40"
## [209] "tae"   "tca"   "tcc"   "tch"   "tgu"   "tni"   "tre"   "ttu"
## [217] "tur"   "vun"   "vvi"   "xbo"   "xla"   "xtr"   "zma"
```

Additional information

Packages loaded via namespace

The following packages are used in the `miRNAmeConverter` package:

- DBI_0.3.1 (Databases, 2014)
- tools_3.2.2 (R Core Team, 2015)
- RSQLite_1.0.0 (Wickham, James, & Falcon, 2014)
- miRBaseVersions.db annotation package (Haunsberger, 2016)

Future Aspects

This tool can only offer good functionality if it will be kept up to date. Therefore we plan to include new miRBase releases as soon as possible.

References

AMBROS, V., BARTEL, B., BARTEL, D. P., BURGE, C. B., CARRINGTON, J. C., CHEN, X., ... TUSCHL, T. (2003). A uniform system for microRNA annotation. *RNA*, *9*(3), 277–279. http://doi.org/10.

1261/rna.2183803

Boettiger, C. (2015). *Knitcitations: Citations for 'knitr' markdown files*. Retrieved from http://CRAN.R-project.org/package=knitcitations

Databases, R. S. I. G. on. (2014). *DBI: R database interface*. Retrieved from http://CRAN.R-project.org/package=DBI

Francois, R. (2014). *Bibtex: Bibtex parser*. Retrieved from http://CRAN.R-project.org/package=bibtex

Griffiths-Jones, S. (2004). The microRNA registry. *Nucleic Acids Research*, *32*(suppl 1), D109–D111. http://doi.org/10.1093/nar/gkh023

Griffiths-Jones, S., Grocock, R. J., Dongen, S. van, Bateman, A., & Enright, A. J. (2006). MiRBase: MicroRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, *34*(suppl 1), D140–D144. http://doi.org/10.1093/nar/gkj112

Griffiths-Jones, S., Saini, H. K., Dongen, S. van, & Enright, A. J. (2008). MiRBase: Tools for microRNA genomics. *Nucleic Acids Research*, *36*(suppl 1), D154–D158. http://doi.org/10.1093/nar/gkm952

Haunsberger, S. (2016). *MiRBaseVersions.db: Collection of mature miRNA names of 21 different miRBase release versions*.

Kozomara, A., & Griffiths-Jones, S. (2011). MiRBase: Integrating microRNA annotation and deep-sequencing data. *Nucleic Acids Research*, *39*(suppl 1), D152–D157. http://doi.org/10.1093/nar/gkq1027

Kozomara, A., & Griffiths-Jones, S. (2014). MiRBase: Annotating high confidence microRNAs using deep sequencing data. *Nucleic Acids Research*, *42*(D1), D68–D73. http://doi.org/10.1093/nar/gkt1181

Meyers, B. C., Axtell, M. J., Bartel, B., Bartel, D. P., Baulcombe, D., Bowman, J. L., ... others. (2008). Criteria for annotation of plant microRNAs. *The Plant Cell*, *20*(12), 3186–3190.

R Core Team. (2015). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Wickham, H., James, D. A., & Falcon, S. (2014). *RSQLite: SQLite interface for r*. Retrieved from http://CRAN.R-project.org/package=RSQLite

Xie, Y. (2014). Knitr: A comprehensive tool for reproducible research in R. In V. Stodden, F. Leisch, & R. D. Peng (Eds.), *Implementing reproducible computational research*. Chapman; Hall/CRC. Retrieved from http://www.crcpress.com/product/isbn/9781466561595

Xie, Y. (2015a). *Dynamic documents with R and knitr* (2nd ed.). Boca Raton, Florida: Chapman; Hall/CRC. Retrieved from http://yihui.name/knitr/

Xie, Y. (2015b). *Knitr: A general-purpose package for dynamic report generation in r*. Retrieved from http://yihui.name/knitr/