Quick Guide to AssessORFData

Deepank Korandla

25 October 2018

Package

AssessORFData 0.99.1

Contents

1 Introduction

AssessORFData is an accompaniment to the AssessORF package, providing access to mapping and results objects generated by AssessORF as well as the genome sequences for the strains corresponding to those objects. Briefly, a mapping object stores the mapping of proteomics evidence and evolutionary conservation evidence to a particular strain's genome, and a results object stores how much evidence there is supporting or against each gene in a set of predicted genes for a particular strain's genome. Detailed descriptions of the structure and content of those two types of objects can be found in the documentation for the AssessORF package.

2 Strains and Gene Sources

AssessORFData has data for 20 strains and their IDs (within the package) are listed below:

```
AssessORFData::GetStrainIDs()
##  [1] "ATCC11842"  "ATCC13032"  "ATCC17978"  "ATCC700084" "CCMP1375"
##  [6] "CECT5344"   "CNRZ327"    "COH1"       "D_UW_3_CX"  "EGDe"
## [11] "H37Rv"      "Houston1"   "Il1403"     "K12_MG1655" "LAL14_1"
## [16] "MGAS5005"   "PAO1"       "SL1344"     "Strain168"  "TCH1516"
```

For each strain, there are 5 objects 1 mapping object and 4 results objects. The 4 results objects per strain differ in that for each one, the set of predicted came from a different program or database. The same 4 gene sources were used for all 20 strains, and their IDs (within the package) are listed below:

```
AssessORFData::GetGeneSources()
## [1] "Prodigal"   "GeneMarkS2" "Genbank"    "Glimmer"
```

3 Getting the Objects

While the `data` function can be used to pull the desired object from the package into the user's workspace, using the the `data` function may be inconvenient for some users because there are 100 mapping and results objects, each of which has a long name. For this reason, AssessORFData has alternative functions, `GetDataMapObj` and `GetResultsObj`, to accomplish a similar task. These functions allow the user to get the object of interest and then save it in their workspace under a different name. Examples on how to use the two functions are described below:

```
library(AssessORFData)

## Can replace the character string specifying the strain ID (first
## parameter) with any of the other 19 strain IDs listed above
mapObj <- GetDataMapObj("MGAS5005")
resObj1 <- GetResultsObj("MGAS5005", "Prodigal")
resObj2 <- GetResultsObj("MGAS5005", "Genbank")
resObj3 <- GetResultsObj("MGAS5005", "GeneMarkS2")
resObj4 <- GetResultsObj("MGAS5005", "Glimmer")
```

4 Saving the Genome

The `SaveGenomeToPath` function allows the user to save the genome for a strain of their choosing to a file path on their local machine in situations where the user wants to run their own analyses on their strain's genome, e.g. predict genes for the genome using a different gene finding program. An example of how to use the function is provided below:

```
library(AssessORFData)

## A path to a temporary file is used in this example.
tmpFile <- paste0(tempfile(), ".fasta")

## Replace the second parameter below with a character string specifying
## the desired file path, making sure the file is of type FASTA.
SaveGenomeToPath("MGAS5005", tmpFile)

unlink(tmpFile)
```

5 Session Info

All of the output in this vignette was produced under the following conditions:

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 17134)
##
## Matrix products: default
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] BiocStyle_2.9.6      AssessORFData_0.99.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.19        BiocManager_1.30.3  compiler_3.5.1
##  [4] XVector_0.21.3      zlibbioc_1.27.0     prettyunits_1.0.2
##  [7] base64enc_0.1-3     remotes_2.0.1       tools_3.5.1
## [10] bit_1.1-14          digest_0.6.18       pkgbuild_1.0.2
## [13] pkgload_1.0.1       evaluate_0.12       RSQLite_2.1.1
## [16] memoise_1.1.0       pkgconfig_2.0.2     rlang_0.2.2
## [19] DBI_1.0.0           cli_1.0.1           rstudioapi_0.8
## [22] commonmark_1.6      yaml_2.2.0          parallel_3.5.1
## [25] xfun_0.3            knitr_1.20          withr_2.1.2
## [28] stringr_1.3.1       roxygen2_6.1.0      xml2_1.2.0
## [31] Biostrings_2.49.1   desc_1.2.0          fs_1.2.6
## [34] S4Vectors_0.19.19   devtools_2.0.0      IRanges_2.15.17
## [37] bit64_0.9-7         stats4_3.5.1        rprojroot_1.3-2
## [40] glue_1.3.0          R6_2.3.0            processx_3.2.0
## [43] bookdown_0.7        rmarkdown_1.10      sessioninfo_1.1.0
## [46] blob_1.1.1          callr_3.0.0         purrr_0.2.5
## [49] DECIPHER_2.9.1      magrittr_1.5        htmltools_0.3.6
## [52] backports_1.1.2     ps_1.2.0            usethis_1.4.0
## [55] BiocGenerics_0.27.1 assertthat_0.2.0    stringi_1.1.7
## [58] crayon_1.3.4
```