Package 'GAPGOM'

October 16, 2019

Type Package

Title GAPGOM (novel Gene Annotation Prediction and other GO Metrics)

Version 1.0.0

Description Collection of various measures and tools for lncRNA annotation prediction put inside a redistributable R package. The package contains two main algorithms; lncRNA2GOA and TopoICSim. lncRNA2GOA tries to annotate novel genes (in this specific case lncRNAs) by using various correlation/geometric scoring methods on correlated expression data. After correlating/scoring, the results are annotated and enriched. TopoICSim is a topologically based method, that compares gene similarity based on the topology of the GO DAG by information content (IC) between GO terms.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports stats, utils, methods, Matrix, fastmatch, plyr, dplyr, magrittr, data.table, igraph, graph, RBGL, GO.db, org.Hs.eg.db, org.Mm.eg.db, GOSemSim, GEOquery, AnnotationDbi, Biobase, BiocFileCache, matrixStats

Suggests org.Dm.eg.db, org.Rn.eg.db, org.Sc.sgd.db, org.Dr.eg.db, org.Ce.eg.db, org.At.tair.db, org.EcK12.eg.db, org.Bt.eg.db, org.Cf.eg.db, org.Ag.eg.db, org.EcSakai.eg.db, org.Gg.eg.db, org.Pt.eg.db, org.Pf.plasmo.db, org.Mmu.eg.db, org.Ss.eg.db, org.Xl.eg.db, testthat, pryr, knitr, rmarkdown, prettydoc, ggplot2, kableExtra, profvis, reshape2

Depends R (>= 3.6.0)

biocViews GO, GeneExpression, GenePrediction

BugReports https://github.com/Berghopper/GAPGOM/issues/

URL https://github.com/Berghopper/GAPGOM/

RoxygenNote 6.1.1

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/GAPGOM

git_branch RELEASE_3_9

git_last_commit 35e5642

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

Author Casper Peters [aut, cre],
Finn Drabløs [aut],
Rezvan Ehsani [aut]

Maintainer Casper Peters <cp100u@hotmail.com>

R topics documented:

`expression_prediction` *GAPGOM - expression_prediction()*

Description

Predicts annotation of un-annotated genes based on existing Gene Ontology annotation data and correlated expression patterns.

Usage

```
expression_prediction(gene_id, expression_set, organism, ontology,
  enrichment_cutoff = 250, method = "combine", significance = 0.05,
  go_amount = 5, filter_pvals = FALSE, idtype = "ENTREZID",
  verbose = FALSE, id_select_vector = NULL, id_translation_df = NULL,
  go_data = NULL)
```

Arguments

`gene_id`	gene rowname to be compared to the other GO terms.
`expression_set`	ExpressionSet class containing expression values and other useful information, see GAPGOM::f5_example_data documentation for further explanation of this type. If you want a custom ExpressionSet you have to define one yourself.
`organism`	where to be scanned genes reside in, this option is neccesary to select the correct GO DAG. Options are based on the org.db bioconductor package; http://www.bioconductor.org/packa Following options are available: "fly", "mouse", "rat", "yeast", "zebrafish", "worm", "arabidopsis", "ecolik12", "bovine", "canine", "anopheles", "ecsakai", "chicken", "chimp", "malaria", "rhesus", "pig", "xenopus". Fantom5 data only has "human" and "mouse" available depending on the dataset.
`ontology`	desired ontology to use for prediction. One of three; "BP" (Biological process), "MF" (Molecular function) or "CC" (Cellular Component). Cellular Component is not included with the package's standard data and will thus yield no results.

enrichment_cutoff
: cutoff number for the amount of genes to be enriched in the enrichment analysis. (default is 250)

method
: which statistical method to use for the prediction, currently there are 5 available; "pearson", "spearman", "kendall", "fisher", "sobolev" and "combine".

significance
: normalized p-values (fdr) that are below this number will be kept. has to be a float/double between 0-1. Default is 0.05

go_amount
: minimal amount of gos that a result needs to have to be considered similar enough.

filter_pvals
: filters pvalues that are equal to 0 (Default=FALSE).

idtype
: idtype of the expression_data. If not correctly specified, error will specify available IDs. default="ENTREZID"

verbose
: set to true for more informative/elaborate output.

id_select_vector
: gene rowname(s) that you want to keep in the dataset. For example, let's say you need to only include protein coding genes. You then make a vector including only ids that are protein coding. Most importantly, this is used in the GO term enrichment. Meaning that this vector should only contain genes that are annotated in the GO databases.

id_translation_df
: df with translations between ID and GOID. col1 = ID, col2 = GOID. (this may be generated with ".generate_translation_df()" but this is not officially supported. It might be useful for running anylyses on the same expressionset because it improves performance.)

go_data
: from set_go_data function. A GoSemSim go_data object.

Details

This function is specifically made for predicting lncRNA annotation by assuming "guilt by association". For instance, the expression data in this package is actually based on mRNA expression data, but correlated with lncRNA. This expression data is the used in combination with mRNA GO annotation to calculate similarity scores between GO terms,

Value

The resulting dataframe with prediction of similar GO terms. These are ordered with respect to FDR values. The following columns will be in the dataframe; GOID - Gene Ontology ID, Ontology - Ontology type (MF or BP), FDR - False Positive Rate, Term - description of GOID, used_method - the used method to determine the ontology term similarity

Examples

```
# Example with default dataset, take a look at the data documentation
# to fully grasp what's going on with making of the filter etc. (Biobase
# ExpressionSet)

library(Biobase)

# keep everything that is a protein coding gene (for annotation)
filter_vector <- pData(featureData(GAPGOM::expset))[(
pData(featureData(GAPGOM::expset))$GeneType=="protein_coding"),]$GeneID
# set gid and run.
```

```
gid <- "ENSG00000228630"

result <- GAPGOM::expression_prediction(gid,
                                        GAPGOM::expset,
                                        "human",
                                        "BP",
                                        id_translation_df =
                                          GAPGOM::id_translation_df,
                                        id_select_vector = filter_vector,
                                        method = "combine", verbose = TRUE,
                                        filter_pvals = TRUE
)
```

`expression_semantic_scoring` *GAPGOM - expression_prediction()*

Description

Predicts annotation of un-annotated genes based on existing Gene Ontology annotation data and correlated expression patterns.

Usage

```
expression_semantic_scoring(gene_id, expression_set, method = "combine")
```

Arguments

`gene_id`	gene rowname to be compared to the other GO terms.
`expression_set`	ExpressionSet class containing expression values and other useful information, see GAPGOM::f5_example_data documentation for further explanation of this type. If you want a custom ExpressionSet you have to define one yourself.
`method`	which statistical method to use for the prediction, currently there are 5 available; "pearson", "spearman", "kendall", "fisher", "sobolev" and "combine".

Details

This function is specifically made for predicting lncRNA annotation by assuming "guilt by association". For instance, the expression data in this package is actually based on mRNA expression data, but correlated with lncRNA. This expression data is the used in combination with mRNA GO annotation to calculate similarity scores between GO terms,

Value

The resulting dataframe with prediction of similar GO terms. These are ordered with respect to FDR values. The following columns will be in the dataframe; GOID - Gene Ontology ID, Ontology - Ontology type (MF or BP), FDR - False Positive Rate, Term - description of GOID, used_method - the used method to determine the ontology term similarity

Examples

```
# Example with default dataset, take a look at the data documentation
# to fully grasp what's going on with making of the filter etc. (Biobase
# ExpressionSet)

# set an artbitrary gene you want to find similarities for. (5th row in this
# case)
gid <- "ENSG00000228630"
result <- GAPGOM::expression_semantic_scoring(gid,
                                              GAPGOM::expset)
```

`expset` *GAPGOM - Expression Data.*

Description

A Bioconductor ExpressionSet object containing a subset from the lncRNA2Function data.[1] Because the original lncRNA2Function data was unavailable for some time, this subset is selected with expression data from (this is based on the lncRNA2Function however): https://tare.medisin.ntnu.no/pred_lncRNA/ . The original lncRNA2Function data is now online on a new domain: http://bio-annotation.cn/lncrna2function/

Usage

```
expset
```

Format

An ExpressionSet containing 1001 rows of the lncRNA2Function expression dataset. Annotation data can be found under; pData(featureData(GAPGOM::expset)) and expression values under; assayData(GAPGOM::expset)[["exprs"]]

Source

http://bio-annotation.cn/lncrna2function/

References

[1]. Jiang Q et al.: **LncRNA2Function: a comprehensive resource for functional investigation of human lncRNAs based on RNA-seq data.** in: *BMC Genomics*, 2015, doi: 10.1186/1471-2164-16-S3-S2.

fantom_download *GAPGOM - fantom_download()*

Description

Downloads and unpacks fantom5 data of either human or mouse.

Usage

```
fantom_download(organism = "human", unpack = TRUE, noprompt = FALSE)
```

Arguments

organism	Either "mouse" or "human". FANTOM5 only has these two as fully annotated+tpm normalized datasets.
unpack	Default=TRUE, if set to TRUE, file will be unpacked automatically.
noprompt	Default=FALSE, user prompt, if you are 100 download and have enough space, set this to TRUE.

Details

This function downloads the whole fantom5 dataset and unpacks it. automatic unpacking can be turned off. The file is downloaded to a special caching directory, which will be returned on exit.

Value

The resulting filename/location of the file or NULL if cancelled.

Examples

```
fantom_file <- fantom_download(organism = "mouse", noprompt = TRUE)
```

fantom_load_raw *GAPGOM - fantom_load_raw()*

Description

Loads raw fantom5 data from file.

Usage

```
fantom_load_raw(filepath, verbose = FALSE, example = FALSE)
```

Arguments

filepath	filename of fantom5 file.
verbose	Switch to TRUE for extra messages. Default=FALSE
example	Boolean switch for R CMD Check (NOT MEANT TO BE TURNED ON FOR END-USERS).

Details

This function loads raw fantom5 data and returns the resulting data.table/ data.frame object.

Value

The resulting datatable containing raw fantom5 data. (Most of the time very large!)

Examples

```
fantom_file <- fantom_download(organism = "mouse", noprompt = TRUE)
ft5 <- fantom_load_raw(fantom_file, verbose = TRUE, example = TRUE)
```

---

fantom_to_expset *GAPGOM - fantom_to_expset()*

Description

Convert raw data.table/data.frame fantom5 object to a proper ExpressionSet.

Usage

```
fantom_to_expset(fanraw, species, filter = TRUE, verbose = FALSE)
```

Arguments

fanraw	raw data.table object from the fantom_load_raw() function.
species	either "human" or "mouse". This is important because both datasets have different metadata/stats
filter	Filter, this causes only entries to be added that have an entrez ID. Normally this should be left on default (TRUE) because all algorithms in this library need the entrez IDs for translation.
verbose	Switch to TRUE for extra messages. Default=FALSE

Details

This function converts fantom5 data and converts it into an ExpresionSet. This ExpressionSet is then returned. This function only accepts the RLE normalized data!

Value

The resulting ExpressionSet contains the original data. The expressiondata can be found under assayData(ExpressionSet)[["exprs"]] Other information (first 6 info columns) can be found under; pData(featureData(ExpressionSet))

Examples

```
fantom_file <- fantom_download(organism = "mouse", noprompt = TRUE)
ft5 <- fantom_load_raw(fantom_file, verbose = TRUE, example = TRUE)
expset <- fantom_to_expset(ft5, "mouse", verbose = TRUE)
```

id_translation_df *GAPGOM - id_translation_df*

Description

An translation dataframe between ensembl ids and goids. This dataframe belongs with the expset data.

Usage

```
id_translation_df
```

Format

An translation dataframe between ensembl ids and goids.

ORIGID original id as in the expset, ensembl.

GO GOID term

Details

https://tare.medisin.ntnu.no/pred_lncRNA/

Source

https://tare.medisin.ntnu.no/pred_lncRNA/

set_go_data *GAPGOM - set_go_data()*

Description

Sets GO data like GOSemSim (this function purely makes choosing datasets a little easier and prints available keytypes if specified incorrectly.)

Usage

```
set_go_data(organism, ontology, computeIC = TRUE, keytype = "ENTREZID")
```

Arguments

organism	where to be scanned genes reside in, this option is neccesary to select the correct GO DAG. Options are based on the org.db bioconductor package; http://www.bioconductor.org/packa Following options are available: "fly", "mouse", "rat", "yeast", "zebrafish", "worm", "arabidopsis", "ecolik12", "bovine", "canine", "anopheles", "ecsakai", "chicken", "chimp", "malaria", "rhesus", "pig", "xenopus". Fantom5 data only has "human" and "mouse" available depending on the dataset.
ontology	desired ontology to use for prediction. One of three; "BP" (Biological process), "MF" (Molecular function) or "CC" (Cellular Component). Cellular Component is not included with the package's standard data and will thus yield no results.
computeIC	whether to compute Information Content.
keytype	keytype used in querying of godata

Value

return godata as from GoSemSim

Notes

Internal function used in multiple functions of topoICSim.

Examples

```
# set go data for human, MF ontology.
go_data <- GAPGOM::set_go_data("human", "MF")
```

---

topo_ic_sim_genes *GAPGOM - topo_ic_sim_genes()*

Description

Algorithm to calculate similarity between GO terms of two genes/genelists.

Usage

```
topo_ic_sim_genes(organism, ontology, genes1, genes2,
  custom_genes1 = NULL, custom_genes2 = NULL, verbose = FALSE,
  debug = FALSE, progress_bar = TRUE, garbage_collection = FALSE,
  use_precalculation = FALSE, drop = NULL, all_go_pairs = NULL,
  idtype = "ENTREZID", go_data = NULL)
```

Arguments

organism	organism where to be scanned genes reside in, this option is neccesary to select the correct GO DAG. Options are based on the org.db bioconductor package; http://www.bioconductor.org/packages/release/BiocViews.html#___OrgDb Following options are available: "fly", "mouse", "rat", "yeast", "zebrafish", "worm", "arabidopsis", "ecolik12", "bovine", "canine", "anopheles", "ecsakai", "chicken", "chimp", "malaria", "rhesus", "pig", "xenopus".
ontology	desired ontology to use for similarity calculations. One of three; "BP" (Biological process), "MF" (Molecular function) or "CC" (Cellular Component).
genes1	Gene ID(s) of the first Gene (vector).
genes2	Gene ID(s) of the second Gene (vector).
custom_genes1	Custom genes added to the first list, needs to be a named list with the name being the arbitrary ID and the value being a vector of GO terms.
custom_genes2	same as custom_genes1 but added to second gene list.
verbose	set to true for more informative/elaborate output.
debug	verbosity for debugging.
progress_bar	Whether to show the progress of the calculation (default = FALSE)

garbage_collection
: whether to do R garbage collection. This is useful for very large calculations/datasets, as it might decrease ram usage. This option might however increase calculation time slightly.

use_precalculation
: wheter to use precalculated score matrix or not. This speeds up calculation for the most frequent GO terms. Only available for human, mouse with ids entrez/ensembl. Default is False because this is the safest and most accurate option. Every update of org.Db libraries makes this matrix outdated, so use at your own risk.

drop
: vector of evidences in go data structure you want to skip (see set_go_data).

all_go_pairs
: dataframe of GO Term pairs with a column representing similarity between the two. You can add the dataframe from previous runs to improve performance (only works if the last result has at least part of the genes of the current run). You can also use it for pre-calculation and getting the results back in a fast manner.

idtype
: id type of the genes you specified. default="ENTREZID". To see other options, enter empty string.

go_data
: prepared go_data, from the set_go_data function. It is practically the same as in GOSemSim, but with a slightly nicer interface.

Details

This function is made for calculating topological similarity between two gene vectors of which each gene has its GO terms in the GO DAG structure. The topological similarity is based on edge weights and information content (IC). The output it a nxn matrix depending on the vector lengths. Intraset similarity can be calculated by comparing the same gene vector to itself and using mean() on the output. The same can be done for Interset similarity, but between two **different** gene lists (IntraSet and InterSet similarities are only applicable to gene sets). [1]

Value

List containing the following; $GeneSim; similarity between genes taken from the mean of all term similarities (single gene). Or a nxn matrix of gene similarities. Intraset similarity can be calculated by comparing the same gene vector to itself and using mean() on the output. The same can be done for Interset similarity, but between two **different** gene vectors (gene vector). ; $AllGoPairs; All possible GO combinations with their semantic distances (matrix). NAs might be present in the matrix, these are GO pairs that didn't occur.

References

[1] Ehsani R, Drablos F: **TopoICSim: a new semantic similarity measure based on gene ontology.** *BMC Bioinformatics* 2016, **17**(1):296)

Examples

```
# single gene mode
result <- GAPGOM::topo_ic_sim_genes("human", "MF", "218", "501")

# genelist mode
list1 <- c("126133","221","218","216","8854","220","219","160428","224",
"222","8659","501","64577","223","217","4329","10840","7915","5832")
# ONLY A PART OF THE GENELIST IS USED BECAUSE OF R CHECK TIME CONTRAINTS
```

```
result <- GAPGOM::topo_ic_sim_genes("human", "MF", list1[1:2],
                                                    list1[1:2])

# with custom gene
custom <- list(cus1=c("GO:0016787", "GO:0042802", "GO:0005524"))
result <- GAPGOM::topo_ic_sim_genes("human", "MF", "218", "501",
                                    custom_genes1 = custom)
```

---

topo_ic_sim_term *GAPGOM - topo_ic_sim_term()*

Description

Algorithm to calculate similarity between two GO terms.

Usage

```
topo_ic_sim_term(organism, ontology, go1, go2, go_data = NULL)
```

Arguments

organism	where to be scanned genes reside in, this option is neccesary to select the correct GO DAG. Options are based on the org.db bioconductor package; http://www.bioconductor.org/packa Following options are available: "fly", "mouse", "rat", "yeast", "zebrafish", "worm", "arabidopsis", "ecolik12", "bovine", "canine", "anopheles", "ecsakai", "chicken", "chimp", "malaria", "rhesus", "pig", "xenopus".
ontology	desired ontology to use for similarity calculations. One of three; "BP" (Biological process), "MF" (Molecular function) or "CC" (Cellular Component).
go1	GO term of first term.
go2	GO term of second term.
go_data	prepared go_data, from the set_go_data function. It is practically the same as in GOSemSim, but with a slightly nicer interface.

Details

This function is made for calculating topological similarity of two GO terms in the GO DAG structure. The topological similarity is based on edge weights and information content (IC). [1]

Value

TopoICSim score between the two terms.

References

[1] Ehsani R, Drablos F: **TopoICSim: a new semantic similarity measure based on gene ontology.** *BMC Bioinformatics* 2016, **17**(1):296)

Examples

```
result <- topo_ic_sim_term("human", "MF", "GO:0018478", "GO:0047105")
```

Index