Package: orthogene (via r-universe)

September 28, 2024

Type Package

Title Interspecies gene mapping

Version 1.11.0

Description `orthogene` is an R package for easy mapping of orthologous genes across hundreds of species. It pulls up-to-date gene ortholog mappings across **700+ organisms**. It also provides various utility functions to aggregate/expand common objects (e.g. data.frames, gene expression matrices, lists) using **1:1**, **many:1**, **1:many** or **many:many** gene mappings, both within- and between-species.

URL https://github.com/neurogenomics/orthogene

BugReports https://github.com/neurogenomics/orthogene/issues

License GPL-3

Depends R (>= 4.1)

VignetteBuilder knitr

biocViews Genetics, ComparativeGenomics, Preprocessing, Phylogenetics, Transcriptomics, GeneExpression

Imports dplyr, methods, stats, utils, Matrix, jsonlite, homologene, gprofiler2, babelgene, data.table, parallel, ggplot2, ggpubr, patchwork, DelayedArray, grr, repmis, ggtree, tools

Suggests rworkflows, remotes, knitr, BiocStyle, markdown, rmarkdown, testthat (>= 3.0.0), piggyback, magick, GenomeInfoDbData, ape, phytools, rphylopic (>= 1.0.0), TreeTools, ggimage, OmaDB

RoxygenNote 7.2.3

Encoding UTF-8

Config/testthat/edition 3

Config/rcmdcheck/_R_CHECK_FORCE_SUGGESTS_ false

Repository https://bioc.r-universe.dev

RemoteUrl https://github.com/bioc/orthogene

RemoteRef HEAD

RemoteSha 37712f29ccbd328283eda5bb9b61d3c66e5e62e6

Contents

`orthogene-package` **orthogene**: *Interspecies gene mapping*

Description

orthogene is an R package for easy mapping of orthologous genes across hundreds of species.

Details

It pulls up-to-date interspecies gene ortholog mappings across 700+ organisms.
It also provides various utility functions to map common objects (e.g. data.frames, gene expression matrices, lists) onto 1:1 gene orthologs from any other species.

Author(s)

Maintainer: Brian Schilder <`brian_schilder@alumni.brown.edu`> (ORCID)

Source

- GitHub : Source code and Issues submission.
- Author Site : `orthogene` was created by Brian M. Schilder.

See Also

Useful links:

- https://github.com/neurogenomics/orthogene
- Report bugs at https://github.com/neurogenomics/orthogene/issues

---

`aggregate_mapped_genes` *Aggregate/expand a gene matrix by gene mappings*

Description

Aggregate/expand a gene matrix (`gene_df`) using a gene mapping data.frame (`gene_map`). Importantly, mappings can be performed across a variety of scenarios that can occur during within-species and between-species gene mapping:

- 1 gene : 1 gene
- many genes : 1 gene
- 1 gene : many genes
- many genes : many genes

For more details on how aggregation/expansion is performed, please see: many2many_rows.

Usage

```
aggregate_mapped_genes(
  gene_df,
  gene_map = NULL,
  input_col = "input_gene",
  output_col = "ortholog_gene",
  input_species = "human",
  output_species = input_species,
  method = c("gprofiler", "homologene", "babelgene"),
  agg_fun = "sum",
  agg_method = c("monocle3", "stats"),
  aggregate_orthologs = TRUE,
  transpose = FALSE,
  mthreshold = 1,
  target = "ENSG",
  numeric_ns = "",
  as_integers = FALSE,
  as_sparse = TRUE,
  as_DelayedArray = FALSE,
  dropNA = TRUE,
  sort_rows = FALSE,
  verbose = TRUE
)
```

Arguments

gene_df	Input matrix where row names are genes.
gene_map	A data.frame that maps the current gene names to new gene names. This function's behaviour will adapt to different situations as follows: • gene_map=<data.frame> : When a data.frame containing the gene key:value columns (specified by input_col and output_col, respectively) is provided, this will be used to perform aggregation/expansion. • gene_map=NULL and input_species!=output_species : A gene_map is automatically generated by map_orthologs to perform inter-species gene aggregation/expansion. • gene_map=NULL and input_species==output_species : A gene_map is automatically generated by map_genes to perform within-species gene gene symbol standardization and aggregation/expansion.
input_col	Column name within gene_map with gene names matching the row names of X.
output_col	Column name within gene_map with gene names that you wish you map the row names of X onto.
input_species	Name of the input species (e.g., "mouse","fly"). Use map_species to return a full list of available species.
output_species	Name of the output species (e.g. "human","chicken"). Use map_species to return a full list of available species.
method	R package to use for gene mapping: • "gprofiler" : Slower but more species and genes. • "homologene" : Faster but fewer species and genes. • "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.
agg_fun	Aggregation function.
agg_method	Aggregation method.
aggregate_orthologs	[Optional] After performing an initial round of many:many aggregation/expansion with many2many_rows, ensure each orthologous gene only appears in one row by using the aggregate_rows function (default: TRUE).
transpose	Transpose gene_df before mapping genes.
mthreshold	maximum number of results per initial alias to show. Shows all by default.
target	target namespace.
numeric_ns	namespace to use for fully numeric IDs (list of available namespaces).
as_integers	Force all values in the matrix to become integers, by applying floor (default: FALSE).
as_sparse	Convert aggregated matrix to sparse matrix.
as_DelayedArray	Convert aggregated matrix to DelayedArray.
dropNA	Drop genes assigned to NA in groupings.
sort_rows	Sort gene_df rows alphanumerically.
verbose	Print messages.

Value

Aggregated matrix

Examples

```
#### Aggregate within species: gene synonyms ####
data("exp_mouse_enst")
X_agg <- aggregate_mapped_genes(gene_df = exp_mouse_enst,
                                input_species = "mouse")

#### Aggregate across species: gene orthologs ####
data("exp_mouse")
X_agg2 <- aggregate_mapped_genes(gene_df = exp_mouse,
                                 input_species = "mouse",
                                 output_species = "human",
                                 method="homologene")
```

`all_genes` *Get all genes*

Description

Return all known genes from a given species.

Usage

```
all_genes(
  species,
  method = c("gprofiler", "homologene", "babelgene"),
  ensure_filter_nas = FALSE,
  run_map_species = TRUE,
  verbose = TRUE,
  ...
)
```

Arguments

`species`	Species to get all genes for. Will first be standardised with `map_species`.
`method`	R package to use for gene mapping: • `"gprofiler"` : Slower but more species and genes. • `"homologene"` : Faster but fewer species and genes. • `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.
`ensure_filter_nas`	Perform an extra check to remove genes that are `NA`s of any kind.
`run_map_species`	Standardise `species` names with map_species first (Default: `TRUE`).

verbose — Print messages.

... — Additional arguments to be passed to gorth or homologene.

NOTE: To return only the most "popular" interspecies ortholog mappings, supply `mthreshold=1` here AND set `method="gprofiler"` above. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.

For more details, please see here.

Details

References homologeneData or gconvert.

Value

Table with all gene symbols from the given `species`.

Examples

```
genome_mouse <- all_genes(species = "mouse")
genome_human <- all_genes(species = "human")
```

---

all_species *All species*

Description

List all species currently supported by **orthogene**. Wrapper function for map_species. When `method=NULL`, all species from all available methods will be returned.

Usage

```
all_species(method = NULL, verbose = TRUE)
```

Arguments

method — R package to use for gene mapping:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

verbose — Print messages.

Value

data.table of species names, provided in multiple formats.

Examples

```
species_dt <- all_species()
```

`convert_orthologs` *Map genes from one species to another*

Description

Currently supports ortholog mapping between any pair of 700+ species.
Use map_species to return a full list of available organisms.

Usage

```
convert_orthologs(
  gene_df,
  gene_input = "rownames",
  gene_output = "rownames",
  standardise_genes = FALSE,
  input_species,
  output_species = "human",
  method = c("gprofiler", "homologene", "babelgene"),
  drop_nonorths = TRUE,
  non121_strategy = "drop_both_species",
  agg_fun = NULL,
  mthreshold = Inf,
  as_sparse = FALSE,
  as_DelayedArray = FALSE,
  sort_rows = FALSE,
  gene_map = NULL,
  input_col = "input_gene",
  output_col = "ortholog_gene",
  verbose = TRUE,
  ...
)
```

Arguments

`gene_df` Data object containing the genes (see `gene_input` for options on how the genes can be stored within the object).
Can be one of the following formats:

- `matrix` :
  A sparse or dense matrix.
- `data.frame` :
  A `data.frame`, `data.table`. or `tibble`.

- codelist :
  A `list` or character `vector`.

Genes, transcripts, proteins, SNPs, or genomic ranges can be provided in any format (HGNC, Ensembl, RefSeq, UniProt, etc.) and will be automatically converted to gene symbols unless specified otherwise with the ... arguments.
Note: If you set `method="homologene"`, you must either supply genes in gene symbol format (e.g. "Sox2") OR set `standardise_genes=TRUE`.

`gene_input` Which aspect of `gene_df` to get gene names from:

- `"rownames"` :
  From row names of data.frame/matrix.
- `"colnames"` :
  From column names of data.frame/matrix.
- `<column name>` :
  From a column in `gene_df`, e.g. `"gene_names"`.

`gene_output` How to return genes. Options include:

- `"rownames"` :
  As row names of `gene_df`.
- `"colnames"` :
  As column names of `gene_df`.
- `"columns"` :
  As new columns "input_gene", "ortholog_gene" (and "input_gene_standard" if `standardise_genes=TRUE`) in `gene_df`.
- `"dict"` :
  As a dictionary (named list) where the names are input_gene and the values are ortholog_gene.
- `"dict_rev"` :
  As a reversed dictionary (named list) where the names are ortholog_gene and the values are input_gene.

`standardise_genes`
If `TRUE` AND `gene_output="columns"`, a new column "input_gene_standard" will be added to `gene_df` containing standardised HGNC symbols identified by gorth.

`input_species` Name of the input species (e.g., "mouse","fly"). Use map_species to return a full list of available species.

`output_species` Name of the output species (e.g. "human","chicken"). Use map_species to return a full list of available species.

`method` R package to use for gene mapping:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`drop_nonorths` Drop genes that don't have an ortholog in the `output_species`.

non121_strategy

How to handle genes that don't have 1:1 mappings between input_species:output_species. Options include:

- "drop_both_species" or "dbs" or 1 :
  Drop genes that have duplicate mappings in either the input_species or output_species
  (*DEFAULT*).
- "drop_input_species" or "dis" or 2 :
  Only drop genes that have duplicate mappings in the input_species.
- "drop_output_species" or "dos" or 3 :
  Only drop genes that have duplicate mappings in the output_species.
- "keep_both_species" or "kbs" or 4 :
  Keep all genes regardless of whether they have duplicate mappings in either species.
- "keep_popular" or "kp" or 5 :
  Return only the most "popular" interspecies ortholog mappings. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.
- "sum","mean","median","min" or "max" :
  When gene_df is a matrix and gene_output="rownames", these options will aggregate many-to-one gene mappings (input_species-to-output_species) after dropping any duplicate genes in the output_species.

agg_fun — Aggregation function passed to aggregate_mapped_genes. Set to NULL to skip aggregation step (default).

mthreshold — Maximum number of ortholog names per gene to show. Passed to gorth. Only used when method="gprofiler" (*DEFAULT :* Inf).

as_sparse — Convert gene_df to a sparse matrix. Only works if gene_df is one of the following classes:

- matrix
- Matrix
- data.frame
- data.table
- tibble

If gene_df is a sparse matrix to begin with, it will be returned as a sparse matrix (so long as gene_output= "rownames" or "colnames").

as_DelayedArray

Convert aggregated matrix to DelayedArray.

sort_rows — Sort gene_df rows alphanumerically.

gene_map — A data.frame that maps the current gene names to new gene names. This function's behaviour will adapt to different situations as follows:

- gene_map=<data.frame> :
  When a data.frame containing the gene key:value columns (specified by input_col and output_col, respectively) is provided, this will be used to perform aggregation/expansion.

- gene_map=NULL and input_species!=output_species :
  A gene_map is automatically generated by map_orthologs to perform inter-species gene aggregation/expansion.
- gene_map=NULL and input_species==output_species :
  A gene_map is automatically generated by map_genes to perform within-species gene gene symbol standardization and aggregation/expansion.

input_col — Column name within gene_map with gene names matching the row names of X.

output_col — Column name within gene_map with gene names that you wish you map the row names of X onto.

verbose — Print messages.

... — Additional arguments to be passed to gorth or homologene.

NOTE: To return only the most "popular" interspecies ortholog mappings, supply mthreshold=1 here AND set method="gprofiler" above. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.

For more details, please see here.

Value

gene_df with orthologs converted to the output_species.
Instead returned as a dictionary (named list) if gene_output="dict" or "dict_rev".

Examples

```
data("exp_mouse")
gene_df <- convert_orthologs(
    gene_df = exp_mouse,
    input_species = "mouse"
)
```

---

create_background *Create gene background*

Description

Create a gene background as the union/intersect of all orthologs between input species (species1 and species2), and the output_species. This can be useful when generating random lists of background genes to test against in analyses with data from multiple species (e.g. enrichment of mouse cell-type markers gene sets in human GWAS-derived gene sets).

Usage

```
create_background(
  species1,
  species2,
  output_species = "human",
  as_output_species = TRUE,
  use_intersect = TRUE,
  bg = NULL,
  gene_map = NULL,
  method = "homologene",
  non121_strategy = "drop_both_species",
  verbose = TRUE
)
```

Arguments

species1	First species.
species2	Second species.
output_species	Species to convert all genes from species1 and species2 to first. Default="human", but can be to either any species supported by **orthogene**, including species1 or species2.
as_output_species	Return background gene list as output_species orthologs, instead of the gene names of the original input species.
use_intersect	When species1 and species2 are both different from output_species, this argument will determine whether to use the intersect (TRUE) or union (FALSE) of all genes from species1 and species2.
bg	User supplied background list that will be returned to the user after removing duplicate genes.
gene_map	User-supplied gene_map data table from map_orthologs or map_genes.
method	R package to use for gene mapping: • "gprofiler" : Slower but more species and genes. • "homologene" : Faster but fewer species and genes. • "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.
non121_strategy	How to handle genes that don't have 1:1 mappings between input_species:output_species. Options include: • "drop_both_species" or "dbs" or 1 : Drop genes that have duplicate mappings in either the input_species or output_species (*DEFAULT*). • "drop_input_species" or "dis" or 2 : Only drop genes that have duplicate mappings in the input_species.

- `"drop_output_species"` or `"dos"` or `3` :
  Only drop genes that have duplicate mappings in the `output_species`.
- `"keep_both_species"` or `"kbs"` or `4` :
  Keep all genes regardless of whether they have duplicate mappings in either species.
- `"keep_popular"` or `"kp"` or `5` :
  Return only the most "popular" interspecies ortholog mappings. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.
- `"sum"`,`"mean"`,`"median"`,`"min"` or `"max"` :
  When `gene_df` is a matrix and `gene_output="rownames"`, these options will aggregate many-to-one gene mappings (`input_species`-to-`output_species`) after dropping any duplicate genes in the `output_species`.

`verbose` Print messages.

Value

Background gene list.

Examples

```
bg <- orthogene::create_background(species1 = "mouse",
                                   species2 = "rat",
                                   output_species = "human")
```

---

exp_mouse *Gene expression data: mouse*

Description

Mean pseudobulk single-cell RNA-seq gene expression matrix.

Data originally comes from Zeisel et al., 2018 (Cell).

Usage

```
data("exp_mouse")
```

Format

sparse matrix

Source

Publication `ctd <- ewceData::ctd()` `exp_mouse <- as(ctd[[1]]$mean_exp, "sparseMatrix")`
`usethis::use_data(exp_mouse, overwrite = TRUE)`

exp_mouse_enst *Transcript expression data: mouse*

Description

Mean pseudobulk single-cell RNA-seq Transcript expression matrix.

Data originally comes from Zeisel et al., 2018 (Cell).

Usage

```
data("exp_mouse_enst")
```

Format

sparse matrix

Source

Publication `data("exp_mouse")` `mapped_genes <- map_genes(genes = rownames(exp_mouse)[seq(1,100)], target = "ENST", species = "mouse", drop_na = FALSE)` `exp_mouse_enst <- exp_mouse[mapped_genes$input,]` `rownames(exp_mouse_enst) <- mapped_genes$target` `all_nas <- orthogene:::find_all_nas(rownames(exp_mouse` `exp_mouse_enst <- exp_mouse_enst[!all_nas,]` `exp_mouse_enst <- phenomix::add_noise(exp_mouse_enst)` `usethis::use_data(exp_mouse_enst, overwrite = TRUE)`

format_species *Format species names*

Description

Format scientific species names into a standardised manner.

Usage

```
format_species(
  species,
  remove_parentheses = TRUE,
  abbrev = FALSE,
  remove_subspecies = FALSE,
  remove_subspecies_exceptions = c("Canis lupus familiaris"),
  split_char = " ",
  collapse = " ",
  remove_chars = c(" ", ".", "(", ")", "[", "]"),
  replace_char = "",
  lowercase = FALSE,
  trim = "'",
  standardise_scientific = FALSE
)
```

Arguments

species	Species query (e.g. "human", "homo sapiens", "hsapiens", or 9606). If given a list, will iterate queries for each item. Set to NULL to return all species.
remove_parentheses	Remove substring within parentheses: e.g. "Xenopus (Silurana) tropicalis" –> "Xenopus tropicalis"
abbrev	Abbreviate all taxonomic levels except the last one: e.g. "Canis lupus familiaris" ==> "C l familiaris"
remove_subspecies	Only keep the first two taxonomic levels: e.g. "Canis lupus familiaris" –> "Canis lupus"
remove_subspecies_exceptions	Selected species to ignore when remove_subspecies=TRUE. e.g. "Canis lupus familiaris" –> "Canis lupus familiaris"
split_char	Character to split species names by.
collapse	Character to re-collapse species names with after splitting with split_char.
remove_chars	Characters to remove.
replace_char	Character to replace remove_chars with.
lowercase	Make species names all lowercase.
trim	Characters to trim from the beginning/end of each species name.
standardise_scientific	Automatically sets multiple arguments at once to create standardised scientific names for each species. Assumes that species is provided in some version of scientific species names: e.g. "Xenopus (Silurana) tropicalis" –> "Xenopus tropicalis"

Value

A named vector where the values are the standardised species names and the names are the original input species names.

Examples

```
species <- c("Xenopus (Silurana) tropicalis","Canis lupus familiaris")
species2 <- format_species(species = species, abbrev=TRUE)
species3 <- format_species(species = species,
                           standardise_scientific=TRUE,
                           remove_subspecies_exceptions=NULL)
```

`get_silhouettes` *Get silhouettes*

Description

Get silhouette images of each species from phylopic.

Usage

```
get_silhouettes(
  species,
  which = rep(1, length(species)),
  run_format_species = TRUE,
  include_image_data = FALSE,
  mc.cores = 1,
  add_png = FALSE,
  remove_bg = FALSE,
  verbose = TRUE
)
```

Arguments

`species`	A character vector of species names to query phylopic for.
`which`	An integer vector of the same length as `species`. Lets you choose which image you want to use for each species (1st, 2nd 3rd, etc.).
`run_format_species`	Standardise species names with format_species before querying phylopic (default: `TRUE`).
`include_image_data`	Include the image data itself (not just the image UID) in the results.
`mc.cores`	Accelerate multiple species queries by parallelising across multiple cores.
`add_png`	Return URLs for both the SVG and PNG versions of the image.
`remove_bg`	Remove image background.
`verbose`	Print messages.

Value

data.frame with:

- input_species : Species name (input).
- species : Species name (standardised).
- uid : Species UID.
- url : Image URL.

Source

Related function: ggimage::geom_phylopic

phylopic/rphylopic API changes

ggimage: Issue with finding valid PNGs

Examples

```
species <- c("Mus_musculus","Pan_troglodytes","Homo_sapiens")
uids <- get_silhouettes(species = species)
```

---

gprofiler_namespace *gconvert namespaces*

Description

Available namespaces used by link[gprofiler2]gconvert.

Format

data.frame

Source

gProfiler site

Manually-prepared CSV #### path <- "inst/extdata/gprofiler_namespace.csv.gz" gprofiler_namespace <- data.table::fread(path)

---

gprofiler_orgs *Reference organisms*

Description

Organism for which gene references are available via gProfiler API. Used as a backup if API is not available.

Format

data.frame

Source

gProfiler site

```
# NOTE!: Must run usethis::use_data for all internal data at once. # otherwise, the prior
internal data will be overwritten. #### Internal data 1: gprofiler_namespace #### ####
Manually-prepared CSV #### path <- "inst/extdata/gprofiler_namespace.csv.gz" gprofiler_namespace
<- data.table::fread(path) #### Internal data 2: gprofiler_orgs gprofiler_orgs <- orthogene:::get_orgdb_
#### Save #### usethis::use_data(gprofiler_orgs,gprofiler_namespace, overwrite = TRUE,
internal=TRUE)
```

---

infer_species *Infer species from gene names*

Description

Infers which species the genes within `gene_df` is from. Iteratively test the percentage of `gene_df` genes that match with the genes from each `test_species`.

Usage

```
infer_species(
  gene_df,
  gene_input = "rownames",
  test_species = c("human", "monkey", "rat", "mouse", "zebrafish", "fly"),
  method = c("homologene", "gprofiler", "babelgene"),
  make_plot = TRUE,
  show_plot = TRUE,
  verbose = TRUE
)
```

Arguments

`gene_df` Data object containing the genes (see `gene_input` for options on how the genes can be stored within the object).
Can be one of the following formats:

- `matrix` :
  A sparse or dense matrix.
- `data.frame` :
  A `data.frame`, `data.table`. or `tibble`.
- codelist :
  A `list` or character `vector`.

Genes, transcripts, proteins, SNPs, or genomic ranges can be provided in any format (HGNC, Ensembl, RefSeq, UniProt, etc.) and will be automatically converted to gene symbols unless specified otherwise with the ... arguments.
Note: If you set `method="homologene"`, you must either supply genes in gene symbol format (e.g. "Sox2") OR set `standardise_genes=TRUE`.

gene_input — Which aspect of gene_df to get gene names from:

- "rownames" :
  From row names of data.frame/matrix.
- "colnames" :
  From column names of data.frame/matrix.
- <column name> :
  From a column in gene_df, e.g. "gene_names".

test_species — Which species to test for matches with. If set to NULL, will default to a list of humans and 5 common model organisms. If test_species is set to one of the following options, it will automatically pull all species from that respective package and test against each of them:

- "homologene" : 20+ species (default)
- "gprofiler" : 700+ species
- "babelgene" : 19 species

method — R package to use for gene mapping:

- "gprofiler" : Slower but more species and genes.
- "homologene" : Faster but fewer species and genes.
- "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

make_plot — Make a plot of the results.

show_plot — Print the plot of the results.

verbose — Print messages.

Value

An ordered dataframe of test_species from best to worst matches.

Examples

```
data("exp_mouse")
matches <- orthogene::infer_species(gene_df = exp_mouse[1:200,])
```

---

map_genes *Map genes*

Description

Input a list of genes, transcripts, proteins, SNPs, or genomic ranges in any format (HGNC, Ensembl, RefSeq, UniProt, etc.) and return a table with standardised gene symbols (the "names" column).

Usage

```
map_genes(
  genes,
  species = "hsapiens",
  target = "ENSG",
  mthreshold = Inf,
  drop_na = FALSE,
  numeric_ns = "",
  run_map_species = TRUE,
  verbose = TRUE
)
```

Arguments

genes	Gene list.
species	Species to map against.
target	target namespace.
mthreshold	maximum number of results per initial alias to show. Shows all by default.
drop_na	Drop all genes without mappings. Sets `gprofiler2::gconvert(filter_na=)` as well an additional round of more comprehensive NA filtering by **orthogene**.
numeric_ns	namespace to use for fully numeric IDs (list of available namespaces).
run_map_species	Standardise species names with map_species first (Default: TRUE).
verbose	Print messages.

Details

Uses gconvert. The exact contents of the output table will depend on `target` parameter. See `?gprofiler2::gconvert` for more details.

Value

Table with standardised genes.

Examples

```
genes <- c(
    "Klf4", "Sox2", "TSPAN12", "NM_173007", "Q8BKT6",
    "ENSMUSG00000012396", "ENSMUSG00000074637"
)
mapped_genes <- map_genes(
    genes = genes,
    species = "mouse"
)
```

`map_orthologs` *Map orthologs*

Description

Map orthologs from one species to another.

Usage

```
map_orthologs(
  genes,
  standardise_genes = FALSE,
  input_species,
  output_species = "human",
  method = c("gprofiler", "homologene", "babelgene"),
  mthreshold = Inf,
  gene_map = NULL,
  input_col = "input_gene",
  output_col = "ortholog_gene",
  verbose = TRUE,
  ...
)
```

Arguments

`genes`	can be a mixture of any format (HGNC, Ensembl, RefSeq, UniProt, etc.) and will be automatically converted to standardised HGNC symbol format.
`standardise_genes`	If `TRUE` AND `gene_output="columns"`, a new column "input_gene_standard" will be added to `gene_df` containing standardised HGNC symbols identified by gorth.
`input_species`	Name of the input species (e.g., "mouse","fly"). Use map_species to return a full list of available species.
`output_species`	Name of the output species (e.g. "human","chicken"). Use map_species to return a full list of available species.
`method`	R package to use for gene mapping: • `"gprofiler"` : Slower but more species and genes. • `"homologene"` : Faster but fewer species and genes. • `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.
`mthreshold`	Maximum number of ortholog names per gene to show. Passed to gorth. Only used when `method="gprofiler"` (*DEFAULT :* `Inf`).
`gene_map`	A data.frame that maps the current gene names to new gene names. This function's behaviour will adapt to different situations as follows:

- gene_map=<data.frame> :
  When a data.frame containing the gene key:value columns (specified by input_col and output_col, respectively) is provided, this will be used to perform aggregation/expansion.
- gene_map=NULL and input_species!=output_species :
  A gene_map is automatically generated by map_orthologs to perform inter-species gene aggregation/expansion.
- gene_map=NULL and input_species==output_species :
  A gene_map is automatically generated by map_genes to perform within-species gene gene symbol standardization and aggregation/expansion.

input_col — Column name within gene_map with gene names matching the row names of X.

output_col — Column name within gene_map with gene names that you wish you map the row names of X onto.

verbose — Print messages.

... — Additional arguments to be passed to gorth or homologene.

NOTE: To return only the most "popular" interspecies ortholog mappings, supply mthreshold=1 here AND set method="gprofiler" above. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.

For more details, please see here.

Details

map_orthologs() is a core function within convert_orthologs(), but does not have many of the extra checks, such as non121_strategy) and drop_nonorths.

Value

Ortholog map data.frame with at least the columns "input_gene" and "ortholog_gene".

Examples

```
data("exp_mouse")
gene_map <- map_orthologs(
    genes = rownames(exp_mouse),
    input_species = "mouse")
```

---

map_species *Standardise species names*

Description

Search gprofiler database for species that match the input text string. Then translate to a standardised species ID.

Usage

```
map_species(
  species = NULL,
  search_cols = c("display_name", "id", "scientific_name", "taxonomy_id"),
 output_format = c("scientific_name", "id", "display_name", "taxonomy_id", "version",
    "scientific_name_formatted"),
  method = c("homologene", "gprofiler", "babelgene"),
  remove_subspecies = TRUE,
  remove_subspecies_exceptions = c("Canis lupus familiaris"),
  use_local = TRUE,
  verbose = TRUE
)
```

Arguments

species	Species query (e.g. "human", "homo sapiens", "hsapiens", or 9606). If given a list, will iterate queries for each item. Set to NULL to return all species.
search_cols	Which columns to search for species substring in metadata API.
output_format	Which column to return.
method	R package to use for gene mapping: • "gprofiler" : Slower but more species and genes. • "homologene" : Faster but fewer species and genes. • "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.
remove_subspecies	Only keep the first two taxonomic levels: e.g. "Canis lupus familiaris" –> "Canis lupus"
remove_subspecies_exceptions	Selected species to ignore when remove_subspecies=TRUE. e.g. "Canis lupus familiaris" –> "Canis lupus familiaris"
use_local	If TRUE *default*, map_species uses a locally stored version of the species metadata table instead of pulling directly from the gprofiler API. Local version may not be fully up to date, but should suffice for most use cases.
verbose	Print messages.

Value

Species ID of type output_format

Examples

```
ids <- map_species(species = c(
    "human", 9606, "mus musculus",
    "fly", "C elegans"
))
```

plot_orthotree *Create a phylogenetic tree of shared orthologs*

Description

Automatically creates a phylogenetic tree plot annotated with metadata describing how many orthologous genes each species shares with the `reference_species` ("human" by default).

Usage

```
plot_orthotree(
  tree = NULL,
  orth_report = NULL,
  species = NULL,
  method = c("babelgene", "homologene", "gprofiler"),
  tree_source = "timetree",
  non121_strategy = "drop_both_species",
  reference_species = "human",
  clades = list(Primates = c("Homo sapiens", "Macaca mulatta"), Eutherians =
    c("Homo sapiens", "Mus musculus", "Bos taurus"), Mammals = c("Homo sapiens",
   "Mus musculus", "Bos taurus", "Ornithorhynchus anatinus", "Monodelphis domestica"),
   Tetrapods = c("Homo sapiens", "Mus musculus", "Gallus gallus", "Anolis carolinensis",
     "Xenopus tropicalis"), Vertebrates = c("Homo sapiens", "Mus musculus",
    "Gallus gallus", "Anolis carolinensis", "Xenopus tropicalis", "Danio rerio"),
     Invertebrates = c("Drosophila melanogaster",
      "Caenorhabditis elegans")),
  clades_rotate = list(),
  scaling_factor = NULL,
  show_plot = TRUE,
 save_paths = c(tempfile(fileext = ".ggtree.pdf"), tempfile(fileext = ".ggtree.png")),
  width = 15,
  height = width,
  mc.cores = 1,
  verbose = TRUE
)
```

Arguments

`tree`	A phylogenetic tree of class phylo. If no tree is provided (`NULL`) a 100-way multiz tree will be imported from UCSC Genome Browser.
`orth_report`	An ortholog report from one or more species generated by report_orthologs.
`species`	Species to include in the final plot. If `NULL`, then all species from the given database (`method`) will be included (via map_species), so long as they also exist in the `tree`.
`method`	R package to use for gene mapping: • `"gprofiler"` : Slower but more species and genes.

- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`tree_source` Can be one of the following:

- "timetree2022":
  Import and prune the TimeTree >147k species phylogenetic tree. Can also simply type "timetree".
- "timetree2015":
  Import and prune the TimeTree >50k species phylogenetic tree.
- "OmaDB":
  Construct a tree from OMA (Orthologous Matrix browser) via the getTaxonomy function. *NOTE:* Does not contain branch lengths, and therefore may have limited utility.
- "UCSC":
  Import and prune the UCSC 100-way alignment phylogenetic tree (hg38 version).
- "<path>":
  Read a tree from a newick text file from a local or remote URL using read.tree.

`non121_strategy`
How to handle genes that don't have 1:1 mappings between `input_species`:`output_species`. Options include:

- `"drop_both_species"` or `"dbs"` or `1` :
  Drop genes that have duplicate mappings in either the `input_species` or `output_species`
  (*DEFAULT*).
- `"drop_input_species"` or `"dis"` or `2` :
  Only drop genes that have duplicate mappings in the `input_species`.
- `"drop_output_species"` or `"dos"` or `3` :
  Only drop genes that have duplicate mappings in the `output_species`.
- `"keep_both_species"` or `"kbs"` or `4` :
  Keep all genes regardless of whether they have duplicate mappings in either species.
- `"keep_popular"` or `"kp"` or `5` :
  Return only the most "popular" interspecies ortholog mappings. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.
- `"sum"`,`"mean"`,`"median"`,`"min"` or `"max"` :
  When `gene_df` is a matrix and `gene_output="rownames"`, these options will aggregate many-to-one gene mappings (`input_species`-to-`output_species`) after dropping any duplicate genes in the `output_species`.

`reference_species`
Reference species.

`clades` A named list of clades each containing a character vector of species used to define the respective clade using MRCA.

clades_rotate	A list of clades to rotate (via rotate), each containing a character vector of species used to define the respective clade using MRCA.
scaling_factor	How much to scale y-axis parameters (e.g. offset) by.
show_plot	Whether to print the final tree plot.
save_paths	Paths to save plot to.
width	Saved plot width.
height	Saved plot height.
mc.cores	Number of cores to parallelise different steps with.
verbose	Print messages.

Value

A list containing:

- plot : Annotated ggtree object.
- tree : The pruned, standardised phylogenetic tree used in the plot.
- orth_report : Ortholog reports for each `species` against the `reference_species`.
- metadata : Metadata used in the plot, including silhouette PNG ids from phylopic.
- clades : Metadata used for highlighting clades.
- method : `method` used.
- reference_species : `reference_species` used.
- save_paths : `save_paths` to plot.

Source

ggtree tutorial

Examples

```
orthotree <- plot_orthotree(species = c("human","monkey","mouse"))
```

`prepare_tree` *Prepare a phylogenetic tree*

Description

Import a phylogenetic tree and then conduct a series of optional standardisation steps. Optionally, if `output_format` is not `NULL`, species names from both the tree and the `species` argument will first be standardised using map_species.

Usage

```
prepare_tree(
  tree_source = "timetree",
  species = NULL,
  output_format = "scientific_name_formatted",
  run_map_species = c(TRUE, TRUE),
  method = c("homologene", "gprofiler", "babelgene"),
  force_ultrametric = TRUE,
  age_max = NULL,
  show_plot = TRUE,
  save_dir = tools::R_user_dir("orthogene", which = "cache"),
  verbose = TRUE,
  ...
)
```

Arguments

`tree_source` Can be one of the following:

- "timetree2022":
  Import and prune the TimeTree >147k species phylogenetic tree. Can also simply type "timetree".
- "timetree2015":
  Import and prune the TimeTree >50k species phylogenetic tree.
- "OmaDB":
  Construct a tree from OMA (Orthologous Matrix browser) via the getTaxonomy function. *NOTE:* Does not contain branch lengths, and therefore may have limited utility.
- "UCSC":
  Import and prune the UCSC 100-way alignment phylogenetic tree (hg38 version).
- "<path>":
  Read a tree from a newick text file from a local or remote URL using read.tree.

`species` Species names to subset the tree by (after `standardise_species` step).

`output_format` Which column to return.

`run_map_species`
Whether to first standardise species names with map_species.

`method` R package to use for gene mapping:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`force_ultrametric`
Whether to force the tree to be ultrametric (i.e. make all tips the same date) using force.ultrametric.

age_max	Rescale the edges of the tree into units of millions of years (MY) instead than evolutionary rates (e.g. dN/dS ratios). Only used if age_max, the max number , is numeric. Times are computed using makeChronosCalib and chronos.
show_plot	Show a basic plot of the resulting tree.
save_dir	Directory to cache full tree in. Set to NULL to avoid using cache.
verbose	Print messages.
...	Additional arguments passed to makeChronosCalib.

Value

A filtered tree of class "phylo" (with standardised species names).

Source

TimeTree 5: An Expanded Resource for Species Divergence Times

Examples

```
species <- c("human","chimp","mouse")
tr <- orthogene::prepare_tree(species = species)
```

---

report_orthologs *Report orthologs*

Description

Identify the number of orthologous genes between two species.

Usage

```
report_orthologs(
  target_species = "mouse",
  reference_species = "human",
  standardise_genes = FALSE,
  method_all_genes = c("homologene", "gprofiler", "babelgene"),
  method_convert_orthologs = method_all_genes,
  drop_nonorths = TRUE,
  non121_strategy = "drop_both_species",
  round_digits = 2,
  return_report = TRUE,
  ref_genes = NULL,
  mc.cores = 1,
  verbose = TRUE,
  ...
)
```

Arguments

`target_species` Target species.

`reference_species`
Reference species.

`standardise_genes`
If `TRUE` AND `gene_output="columns"`, a new column "input_gene_standard" will be added to `gene_df` containing standardised HGNC symbols identified by gorth.

`method_all_genes`
R package to to use in all_genes step:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`method_convert_orthologs`
R package to to use in convert_orthologs step:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`drop_nonorths` Drop genes that don't have an ortholog in the `output_species`.

`non121_strategy`
How to handle genes that don't have 1:1 mappings between `input_species`:`output_species`. Options include:

- `"drop_both_species"` or `"dbs"` or `1` :
  Drop genes that have duplicate mappings in either the `input_species` or `output_species`
  (*DEFAULT*).
- `"drop_input_species"` or `"dis"` or `2` :
  Only drop genes that have duplicate mappings in the `input_species`.
- `"drop_output_species"` or `"dos"` or `3` :
  Only drop genes that have duplicate mappings in the `output_species`.
- `"keep_both_species"` or `"kbs"` or `4` :
  Keep all genes regardless of whether they have duplicate mappings in either species.
- `"keep_popular"` or `"kp"` or `5` :
  Return only the most "popular" interspecies ortholog mappings. This procedure tends to yield a greater number of returned genes but at the cost of many of them not being true biological 1:1 orthologs.
- `"sum"`,`"mean"`,`"median"`,`"min"` or `"max"` :
  When `gene_df` is a matrix and `gene_output="rownames"`, these options will aggregate many-to-one gene mappings (`input_species`-to-`output_species`) after dropping any duplicate genes in the `output_species`.

round_digits Number of digits to round to when printing percentages.

return_report Return just the ortholog mapping between two species (FALSE) or return both the ortholog mapping as well a data.frame of the report statistics (TRUE).

ref_genes A table of all genes for the reference_species. If NULL (default), this will automatically be created using all_genes.

mc.cores Number of cores to parallelise each target_species with.

verbose Print messages.

... Arguments passed on to convert_orthologs

gene_df Data object containing the genes (see gene_input for options on how the genes can be stored within the object).
Can be one of the following formats:

- matrix :
  A sparse or dense matrix.
- data.frame :
  A data.frame, data.table. or tibble.
- codelist :
  A list or character vector.

Genes, transcripts, proteins, SNPs, or genomic ranges can be provided in any format (HGNC, Ensembl, RefSeq, UniProt, etc.) and will be automatically converted to gene symbols unless specified otherwise with the ... arguments.
Note: If you set method="homologene", you must either supply genes in gene symbol format (e.g. "Sox2") OR set standardise_genes=TRUE.

gene_input Which aspect of gene_df to get gene names from:

- "rownames" :
  From row names of data.frame/matrix.
- "colnames" :
  From column names of data.frame/matrix.
- <column name> :
  From a column in gene_df, e.g. "gene_names".

gene_output How to return genes. Options include:

- "rownames" :
  As row names of gene_df.
- "colnames" :
  As column names of gene_df.
- "columns" :
  As new columns "input_gene", "ortholog_gene" (and "input_gene_standard" if standardise_genes=TRUE) in gene_df.
- "dict" :
  As a dictionary (named list) where the names are input_gene and the values are ortholog_gene.

- `"dict_rev"` :
  As a reversed dictionary (named list) where the names are ortholog_gene and the values are input_gene.

`input_species` Name of the input species (e.g., "mouse","fly"). Use map_species to return a full list of available species.

`output_species` Name of the output species (e.g. "human","chicken"). Use map_species to return a full list of available species.

`agg_fun` Aggregation function passed to aggregate_mapped_genes. Set to `NULL` to skip aggregation step (default).

`mthreshold` Maximum number of ortholog names per gene to show. Passed to gorth. Only used when `method="gprofiler"` (*DEFAULT :* `Inf`).

`method` R package to use for gene mapping:

- `"gprofiler"` : Slower but more species and genes.
- `"homologene"` : Faster but fewer species and genes.
- `"babelgene"` : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.

`as_sparse` Convert `gene_df` to a sparse matrix. Only works if `gene_df` is one of the following classes:

- `matrix`
- `Matrix`
- `data.frame`
- `data.table`
- `tibble`

If `gene_df` is a sparse matrix to begin with, it will be returned as a sparse matrix (so long as `gene_output= "rownames"` or `"colnames"`).

`sort_rows` Sort `gene_df` rows alphanumerically.

`gene_map` A data.frame that maps the current gene names to new gene names. This function's behaviour will adapt to different situations as follows:

- `gene_map=<data.frame>` :
  When a data.frame containing the gene key:value columns (specified by `input_col` and `output_col`, respectively) is provided, this will be used to perform aggregation/expansion.
- `gene_map=NULL` and `input_species!=output_species` :
  A `gene_map` is automatically generated by map_orthologs to perform inter-species gene aggregation/expansion.
- `gene_map=NULL` and `input_species==output_species` :
  A `gene_map` is automatically generated by map_genes to perform within-species gene gene symbol standardization and aggregation/expansion.

`as_DelayedArray` Convert aggregated matrix to DelayedArray.

`input_col` Column name within `gene_map` with gene names matching the row names of `X`.

`output_col` Column name within `gene_map` with gene names that you wish you map the row names of `X` onto.

Value

A list containing:

- map : A table of inter-species gene mappings.
- report : A list of aggregate orthology report statistics.

If >1 `target_species` are provided, then a table of aggregated `report` statistics concatenated across species will be returned instead.

Examples

```
orth_fly <- report_orthologs(
    target_species = "fly",
    reference_species = "human")
```

Index