Package ‘orthogene’

April 10, 2022

Type Package

Title Interspecies gene mapping

Version 1.0.2

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

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, DelayedMatrixStats, Matrix.utils, grr, repmis, GenomeInfoDbData, tools

Suggests remotes, knitr, BiocStyle, covr, markdown, rmarkdown, here, testthat (>= 3.0.0), piggyback, badger, magick

RoxygenNote 7.1.2

Encoding UTF-8

Config/testthat/edition 3

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

git_branch RELEASE_3_14

git_last_commit be43cf5

git_last_commit_date 2022-04-08

Date/Publication 2022-04-10

Author Brian Schilder [cre] (<`https://orcid.org/0000-0001-5949-2191`>)

Maintainer Brian Schilder <`brian_schilder@alumni.brown.edu`>

R topics documented:

`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 a gene matrix by gene symbols*

Description

Map matrix rownames to standardised gene symbols, and then aggregate many-to-one rows into a new matrix.

Usage

```
aggregate_mapped_genes(
  gene_df,
  species = "human",
  FUN = "sum",
  method = c("monocle3", "stats", "delayedarray"),
  transpose = FALSE,
  gene_map = NULL,
  gene_map_col = "name",
  non121_strategy = "drop_output_species",
  as_sparse = TRUE,
  as_DelayedArray = FALSE,
  dropNA = TRUE,
  sort_rows = FALSE,
  verbose = TRUE
)
```

Arguments

`gene_df`	Input matrix where row names are genes.
`species`	Species to map against.
`FUN`	Aggregation function (*DEFAULT:* `"sum"`).
`method`	Aggregation method.
`transpose`	Transpose `gene_df` before mapping genes.
`gene_map`	A user-supplied `gene_map`. If `NULL` (*DEFAULT*)), map_genes will be used to create a `gene_map`.
`gene_map_col`	Column in `gene_map` to aggregate `gene_df` by.

`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`.
`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

```
data("exp_mouse")
X_agg <- aggregate_mapped_genes(gene_df = exp_mouse, species = "mouse")
```

`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 to use for gene mapping: `"gprofiler"` (slower but more species and genes) or `"homologene"` (faster but fewer species and genes).
`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 gconvert when `method="gprofiler"`.

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")
```

`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",
  mthreshold = Inf,
  as_sparse = FALSE,
  sort_rows = FALSE,
  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 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`.
`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"`).
`sort_rows`	Sort `gene_df` rows alphanumerically.
`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 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(ex`
`= 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)`

`gprofiler_orgs` *Reference organisms*

Description

Organism for which gene references are available via gProfiler API.

Used as a backup if API is not available.

Usage

```
gprofiler_orgs
```

Format

`data.frame URL <-'https://biit.cs.ut.ee/gprofiler/api/util/organisms_list' gprofiler_orgs <-jsonlite::fromJSON(URL) gprofiler_orgs <-dplyr::arrange(gprofiler_orgs,scientific_name) usethis::use_data(gprofiler_orgs,overwrite = TRUE,internal=TRUE)`

Source

gProfiler site

`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 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 <- 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`	Run map_species on `species` first.
`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"),
  mthreshold = Inf,
  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 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`).
`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 <- orthogene::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"),
  method = c("homologene", "gprofiler", "babelgene"),
  use_local = TRUE,
  verbose = TRUE
)
```

Arguments

`species`	Species query (e.g. "human", "homo sapiens", "hapiens", 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 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.
`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"
))
```

`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("gprofiler", "homologene"),
  method_convert_orthologs = c("gprofiler", "homologene", "babelgene"),
  drop_nonorths = TRUE,
  non121_strategy = "drop_both_species",
  round_digits = 2,
  return_report = TRUE,
```

```
  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) or `"homologene"` (faster but fewer species and genes).
`method_convert_orthologs`	R package to to use in convert_orthologs step: `"gprofiler"` (slower but more species and genes) or `"homologene"` (faster but fewer species and genes).
`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`).

`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

List of ortholog report statistics

Examples

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

Index