

# Package ‘orthogene’

July 3, 2022

**Type** Package

**Title** Interspecies gene mapping

**Version** 1.2.0

## 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, ggtree, tools

**Suggests** remotes, knitr, BiocStyle, covr, markdown, rmarkdown, here, testthat (>= 3.0.0), piggyback, badger, magick, desc, hrbrthemes, Cairo, yulab.utils, haven, GenomeInfoDbData, ape, phytools, rphylopic, TreeTools, RColorBrewer, ggimage

**RoxygenNote** 7.1.2

**Encoding** UTF-8

**Config/testthat/edition** 3

**Config/rmdcheck/\_R\_CHECK\_FORCE\_SUGGESTS\_** false

**git\_url** <https://git.bioconductor.org/packages/orthogene>

**git\_branch** RELEASE\_3\_15

**git\_last\_commit** e67840a

**git\_last\_commit\_date** 2022-06-14

**Date/Publication** 2022-07-03

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

**Maintainer** Brian Schilder <brian\_schilder@alumni.brown.edu>

## R topics documented:

orthogene-package . . . . .	2
aggregate_mapped_genes . . . . .	3
all_genes . . . . .	4
convert_orthologs . . . . .	5
create_background . . . . .	9
exp_mouse . . . . .	10
exp_mouse_enst . . . . .	11
gprofiler_orgs . . . . .	12
infer_species . . . . .	12
map_genes . . . . .	14
map_orthologs . . . . .	15
map_species . . . . .	16
plot_orthotree . . . . .	17
prepare_tree . . . . .	19
report_orthologs . . . . .	20

<b>Index</b>	<b>23</b>
--------------	-----------

---

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](https://orcid.org/0000-0001-5949-2191))

## 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"),  
  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 ( <i>DEFAULT</i> : "sum").
method	Aggregation method.
transpose	Transpose gene_df before mapping genes.
gene_map	A user-supplied gene_map. If NULL ( <i>DEFAULT</i> ), <a href="#">map_genes</a> 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 <code>map_species</code> .
method	R package 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 NAs of any kind.
run_map_species	Standardise species names with <code>map_species</code> first (Default: TRUE).
verbose	Print messages.
...	Additional arguments to be passed to <code>gconvert</code> when <code>method="gprofiler"</code> .

**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	<i>Map genes from one species to another</i>
-------------------	--

---

**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 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 <a href="#">gorth</a> . Only used when method="gprofiler" ( <i>DEFAULT</i> : Inf).
as_sparse	Convert gene_df to a sparse matrix. Only works if gene_df is one of the following classes: <ul style="list-style-type: none"> <li>• matrix</li> <li>• Matrix</li> <li>• data.frame</li> <li>• data.table</li> <li>• tibble</li> </ul> <p>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").</p>
sort_rows	Sort gene_df rows alphanumerically.
verbose	Print messages.
...	Additional arguments to be passed to <a href="#">gorth</a> or <a href="#">homologene</a> .

*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 <b>orthogene</b> , 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 <a href="#">map_orthologs</a> or <a href="#">map_genes</a> .
method	R package to use for gene mapping: <ul style="list-style-type: none"> <li>• "gprofiler" : Slower but more species and genes.</li> <li>• "homologene" : Faster but fewer species and genes.</li> </ul>

- "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	<i>Transcript expression data: mouse</i>
----------------	--

---

**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	<i>Reference organisms</i>
----------------	----------------------------

---

### 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	<i>Infer species from gene names</i>
---------------	--------------------------------------

---

### 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	<p>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:</p> <ul style="list-style-type: none"> <li>• <code>matrix</code> : A sparse or dense matrix.</li> <li>• <code>data.frame</code> : A <code>data.frame</code>, <code>data.table</code>, or <code>tibble</code>.</li> <li>• <code>codelist</code> : A list or character vector.</li> </ul> <p>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 <code>...</code> arguments. <i>Note</i>: If you set <code>method="homologene"</code>, you must either supply genes in gene symbol format (e.g. "Sox2") OR set <code>standardise_genes=TRUE</code>.</p>
gene_input	<p>Which aspect of <code>gene_df</code> to get gene names from:</p> <ul style="list-style-type: none"> <li>• <code>"rownames"</code> : From row names of <code>data.frame/matrix</code>.</li> <li>• <code>"colnames"</code> : From column names of <code>data.frame/matrix</code>.</li> <li>• <code>&lt;column name&gt;</code> : From a column in <code>gene_df</code>, e.g. <code>"gene_names"</code>.</li> </ul>
test_species	<p>Which species to test for matches with. If set to <code>NULL</code>, will default to a list of humans and 5 common model organisms. If <code>test_species</code> is set to one of the following options, it will automatically pull all species from that respective package and test against each of them:</p> <ul style="list-style-type: none"> <li>• <code>"homologene"</code> 20+ species (default)</li> <li>• <code>"gprofiler"</code> 700+ species</li> <li>• <code>"babelgene"</code> 19 species</li> </ul>
method	<p>R package to use for gene mapping:</p> <ul style="list-style-type: none"> <li>• <code>"gprofiler"</code> : Slower but more species and genes.</li> <li>• <code>"homologene"</code> : Faster but fewer species and genes.</li> <li>• <code>"babelgene"</code> : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.</li> </ul>
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 <code>gprofiler2::gconvert(filter_na=)</code> as well an additional round of more comprehensive NA filtering by <b>orthogene</b> .
numeric_ns	namespace to use for fully numeric IDs ( <a href="#">list of available namespaces</a> ).
run_map_species	Standardise species names with <a href="#">map_species</a> 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	<i>Map orthologs</i>
---------------	----------------------

---

**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 <a href="#">gorth</a> .
input_species	Name of the input species (e.g., "mouse", "fly"). Use <a href="#">map_species</a> to return a full list of available species.
output_species	Name of the output species (e.g. "human", "chicken"). Use <a href="#">map_species</a> to return a full list of available species.
method	R package to use for gene mapping: <ul style="list-style-type: none"> <li>• "gprofiler" : Slower but more species and genes.</li> <li>• "homologene" : Faster but fewer species and genes.</li> </ul>

- "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 <- 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 <a href="#">API</a> .
output_format	Which column to return.
method	R package to to use for gene mapping: <ul style="list-style-type: none"> <li>• "gprofiler" : Slower but more species and genes.</li> <li>• "homologene" : Faster but fewer species and genes.</li> <li>• "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.</li> </ul>
use_local	If TRUE <i>default</i> , <a href="#">map_species</a> 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("homologene", "gprofiler", "babelgene"),
  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")),
show_plot = TRUE,
save_paths = c(tempfile(fileext = ".ggtree.pdf"), tempfile(fileext = ".ggtree.png")),
width = 10,
height = 10,
mc.cores = 1,
verbose = TRUE
)

```

### Arguments

tree	A phylogenetic tree of class <a href="#">phylo</a> . If no tree is provided (NULL) a 100-way multiz tree will be imported from <a href="#">UCSC Genome Browser</a> .
orth_report	An ortholog report from one or more species generated by <a href="#">report_orthologs</a> .
species	Species to include in the final plot. If NULL, then all species from the given database (method) will be included (via <a href="#">map_species</a> ), so long as they also exist in the tree.
method	R package to use for gene mapping: <ul style="list-style-type: none"> <li>• "gprofiler" : Slower but more species and genes.</li> <li>• "homologene" : Faster but fewer species and genes.</li> <li>• "babelgene" : Faster but fewer species and genes. Also gives consensus scores for each gene mapping based on a several different data sources.</li> </ul>
reference_species	Reference species.
clades	A named list of clades each containing list fo species to define the respective clade using <a href="#">MRCA</a> .
show_plot	Whether to print the final tree plot.
save_paths	Paths to save plot to.
width	Plot size in units ("in", "cm", "mm", or "px"). If not supplied, uses the size of current graphics device.
height	Plot size in units ("in", "cm", "mm", or "px"). If not supplied, uses the size of current graphics device.
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 <- orthogene::plot_orthotree(species = c("human","monkey","mouse"))
```

---

prepare_tree	<i>Prepare a phylogenetic tree</i>
--------------	------------------------------------

---

### 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_path = file.path("http://hgdownload.soe.ucsc.edu/goldenPath",
    "hg38/multiz100way", "hg38.100way.scientificNames.nh"),
  species = NULL,
  output_format = "scientific_name",
  run_map_species = c(TRUE, TRUE),
  method = c("gprofiler", "homologene", "babelgene"),
  force_ultrametric = TRUE,
  age_max = NULL,
  show_plot = TRUE,
  verbose = TRUE,
  ...
)
```

### Arguments

<code>tree_path</code>	Local path or URL to tree to import with <a href="#">read.tree</a> .
<code>species</code>	Species names to subset the tree by (after <code>standardise_species</code> step).
<code>output_format</code>	Which column to return.
<code>run_map_species</code>	Whether to first standardise species names with <a href="#">map_species</a> .
<code>method</code>	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 <a href="#">force.ultrametric</a> .
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 <a href="#">makeChronosCalib</a> and <a href="#">chronos</a> .
show_plot	Show a basic plot of the resulting tree.
verbose	Print messages.
...	Additional arguments passed to <a href="#">makeChronosCalib</a> .

### Value

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

### Examples

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

---

report_orthologs	<i>Report orthologs</i>
------------------	-------------------------

---

### 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,
  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).
mc.cores	Number of cores to parallelise each target_species with.
verbose	Print messages.
...	Additional arguments to be passed to <a href="#">gorth</a> or <a href="#">homologene</a> .

*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

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

# Index

## \* datasets

- exp\_mouse, [10](#)
- exp\_mouse\_enst, [11](#)
- gprofiler\_orgs, [12](#)

aggregate\_mapped\_genes, [3](#)  
all\_genes, [4](#), [21](#)

chronos, [20](#)  
convert\_orthologs, [5](#), [21](#)  
create\_background, [9](#)

DelayedArray, [4](#)

exp\_mouse, [10](#)  
exp\_mouse\_enst, [11](#)

force.ultrametric, [20](#)

gconvert, [5](#), [14](#)  
gorth, [7](#), [8](#), [15](#), [16](#), [21](#), [22](#)  
gprofiler\_orgs, [12](#)

homologene, [8](#), [16](#), [22](#)  
homologeneData, [5](#)

infer\_species, [12](#)

makeChronosCalib, [20](#)  
map\_genes, [3](#), [9](#), [14](#)  
map\_orthologs, [9](#), [15](#)  
map\_species, [5](#), [7](#), [14](#), [15](#), [16](#), [17–19](#)  
MRCA, [18](#)

orthogene (orthogene-package), [2](#)  
orthogene-package, [2](#)

phylo, [18](#)  
plot\_orthotree, [17](#)  
prepare\_tree, [19](#)

read.tree, [19](#)  
report\_orthologs, [18](#), [20](#)