Package 'gprofiler2'

May 24, 2019

Type Package

Title Interface to the 'g:Profiler' Toolset

Version 0.1.4

Author Liis Kolberg <liis.kolberg@ut.ee>, Uku Raudvere <uku.raudvere@ut.ee>

Maintainer Liis Kolberg <liis.kolberg@ut.ee>

Description A toolset for functional enrichment analysis and visualization, gene/protein/SNP identifier conversion and mapping orthologous genes across species via 'g:Profiler' (<https://biit.cs.ut.ee/gprofiler>).
The main tools are:
(1) 'g:GOSt' - functional enrichment analysis and visualization of gene lists;
(2) 'g:Convert' - gene/protein/transcript identifier conversion across various namespaces;
(3) 'g:Orth' - orthology search across species;
(4) 'g:SNPense' - mapping SNP rs identifiers to chromosome positions, genes and variant effects
This package is an R interface corresponding to the 2019 update of 'g:Profiler' and provides access to 'g:Profiler' for versions 'e94_eg41_p11' and higher. See the package 'gProfileR' for accessing older versions from the 'g:Profiler' toolset.

BugReports https://biit.cs.ut.ee/gprofiler/page/contact

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports jsonlite, RCurl, ggplot2, plotly, tidyr, crosstalk, plyr, grDevices, gridExtra, grid, viridisLite, dplyr

Depends R (>= 3.5)

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-24 12:00:03 UTC

R topics documented:

`gconvert` *Gene ID conversion.*

Description

Interface to the g:Profiler tool g:Convert that uses the information in Ensembl databases to handle hundreds of types of identifiers for genes, proteins, transcripts, microarray probesets, etc, for many species, experimental platforms and biological databases. The input is flexible: it accepts a mixed list of IDs and recognises their types automatically. It can also serve as a service to get all genes belonging to a particular functional category.

Usage

```
gconvert(query, organism = "hsapiens", target = "ENSG",
  numeric_ns = "", mthreshold = Inf, filter_na = TRUE)
```

Arguments

`query`	vector that can consist of mixed types of gene IDs (proteins, transcripts, microarray IDs, etc), SNP IDs, chromosomal intervals or term IDs.
`organism`	organism name. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.
`target`	target namespace.
`numeric_ns`	namespace to use for fully numeric IDs.
`mthreshold`	maximum number of results per initial alias to show. Shows all by default.
`filter_na`	logical indicating whether to filter out results without a corresponding target.

Value

The output is a data.frame which is a table closely corresponding to the web interface output.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>, Uku Raudvere <uku.raudvere@ut.ee>

Examples

```
gconvert(c("POU5F1", "SOX2", "NANOG"), organism = "hsapiens", target="AFFY_HG_U133_PLUS_2")
```

---

get_base_url *Get the current base URL.*

Description

Get the current base URL.

Usage

```
get_base_url()
```

---

get_tls_version *Get the TLS version for SSL*

Description

Get the TLS version for SSL

Usage

```
get_tls_version()
```

---

get_user_agent *Get current user agent string.*

Description

Get the HTTP User-Agent string.

Usage

```
get_user_agent()
```

gorth *Orthology search.*

Description

Interface to the g:Profiler tool g:Orth that, given a target organism, retrieves the genes of the target organism that are similar in sequence to the source organism genes in the input.

Usage

```
gorth(query, source_organism = "hsapiens",
  target_organism = "mmusculus", numeric_ns = "", mthreshold = Inf,
  filter_na = TRUE)
```

Arguments

query	vector of gene IDs to be translated.
source_organism	name of the source organism. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.
target_organism	name of the target organism. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.
numeric_ns	namespace to use for fully numeric IDs.
mthreshold	maximum number of ortholog names per gene to show.
filter_na	logical indicating whether to filter out results without a corresponding target name.

Value

The output is a data.frame which is a table closely corresponding to the web interface output.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>, Uku Raudvere <uku.raudvere@ut.ee>

Examples

```
gorth(c("Klf4","Pax5","Sox2","Nanog"), source_organism="mmusculus", target_organism="hsapiens")
```

gost *Gene list functional enrichment.*

Description

Interface to the g:Profiler tool g:GOSt for functional enrichments analysis of gene lists. In case the input 'query' is a list of gene vectors, results for multiple queries will be returned in the same data frame with column 'query' indicating the corresponding query name. If 'multi_query' is selected, the result is a data frame for comparing multiple input lists, just as in the web tool.

Usage

```
gost(query, organism = "hsapiens", ordered_query = FALSE,
  multi_query = FALSE, significant = TRUE, exclude_iea = TRUE,
  measure_underrepresentation = FALSE, evcodes = FALSE,
  user_threshold = 0.05, correction_method = c("g_SCS", "bonferroni",
  "fdr", "false_discovery_rate", "gSCS", "analytical"),
  domain_scope = c("annotated", "known", "custom"), custom_bg = NULL,
  numeric_ns = "", sources = NULL)
```

Arguments

query	vector, or a (named) list of vectors for multiple queries, that can consist of mixed types of gene IDs (proteins, transcripts, microarray IDs, etc), SNP IDs, chromosomal intervals or term IDs.
organism	organism name. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.
ordered_query	in case input gene lists are ranked this option may be used to get GSEA style p-values.
multi_query	in case of multiple gene lists, returns comparison table of these lists. If enabled, the result data frame has columns named 'p_values', 'query_sizes', 'intersection_sizes' with vectors showing values in the order of input queries. To get the results in a long format set 'multi_query' to FALSE and just input query list of multiple gene vectors.
significant	whether all or only statistically significant results should be returned.
exclude_iea	exclude GO electronic annotations (IEA).
measure_underrepresentation	measure underrepresentation.
evcodes	include evidence codes to the results. Note that this can decrease performance and make the query slower. In addition, a column 'intersection' is created that contains the gene id-s that intersect between the query and term. This parameter does not work if 'multi_query' is set to TRUE.
user_threshold	custom p-value threshold, results with a larger p-value are excluded.

correction_method
: the algorithm used for multiple testing correction, one of "gSCS" (synonyms: "analytical", "g_SCS"), "fdr" (synonyms: "false_discovery_rate"), "bonferroni".

domain_scope
: how to define statistical domain, one of "annotated", "known" or "custom".

custom_bg
: vector of gene names to use as a statistical background. If given, the domain_scope is set to 'custom'.

numeric_ns
: namespace to use for fully numeric IDs.

sources
: a vector of data sources to use. Currently, these include GO (GO:BP, GO:MF, GO:CC to select a particular GO branch), KEGG, REAC, TF, MIRNA, CORUM, HP, HPA, WP. Please see the g:GOSt web tool for the comprehensive list and details on incorporated data sources.

Value

A named list where 'result' contains data.frame with the enrichment analysis results and 'meta' contains metadata needed for Manhattan plot. If the input consisted of several lists the corresponding list is indicated with a variable 'query'. When requesting a 'multi_query', either TRUE or FALSE, the columns of the resulting data frame differ. If 'evcodes' is set, the return value includes columns 'evidence_codes' and 'intersection'. The latter conveys info about the intersecting genes between the corresponding query and term.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>, Uku Raudvere <uku.raudvere@ut.ee>

Examples

```
gostres <- gost(c("X:1000:1000000", "rs17396340", "GO:0005005", "ENSG00000156103", "NLRP1"))
```

---

gostplot *Manhattan plot of functional enrichment results.*

Description

This function creates a Manhattan plot out of the results from gprofiler2::gost(). The plot is very similar to the one shown in the g:GOSt web tool.

Usage

```
gostplot(gostres, capped = TRUE, interactive = TRUE, pal = c(`GO:MF`
  = "#dc3912", `GO:BP` = "#ff9900", `GO:CC` = "#109618", KEGG = "#dd4477",
  REAC = "#3366cc", WP = "#0099c6", TF = "#5574a6", MIRNA = "#22aa99", HPA
  = "#6633cc", CORUM = "#66aa00", HP = "#990099"))
```

Arguments

gostres	named list from gost() function (with names 'result' and 'meta')
capped	whether the -log10(p-values) would be capped if >= 16, just as in the web options.
interactive	if enabled, returns interactive plot using 'plotly'. If disabled, static 'ggplot()' object is returned.
pal	values mapped to relevant colors for data sources.

Value

The output is either a plotly object (if interactive = TRUE) or a ggplot object (if interactive = FALSE).

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>

Examples

```
gostres <- gost(c("Klf4", "Pax5", "Sox2", "Nanog"), organism = "mmusculus")
gostplot(gostres)
```

---

gsnpense *Convert SNP rs numbers to genes.*

Description

Interface to the g:Profiler tool g:SNPense that maps SNP rs identifiers to chromosome positions, genes and variant effects. Available only for human SNPs.

Usage

```
gsnpense(query, filter_na = TRUE)
```

Arguments

query	vector of SNP IDs to be translated (should start with prefix 'rs').
filter_na	logical indicating whether to filter out results without a corresponding target name.

Value

The output is a data.frame which is a table closely corresponding to the web interface output. Columns 'ensgs' and 'gene_names' can contain list of multiple values.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>, Uku Raudvere <uku.raudvere@ut.ee>

Examples

```
gsnpense(c("rs11734132", "rs7961894", "rs4305276", "rs17396340", "rs3184504"))
```

---

mapViridis *Map vector of numeric values to Viridis color scale.*

Description

Map vector of numeric values to Viridis color scale.

Usage

```
mapViridis(values, domain_min = 0, domain_max = 50, n = 256)
```

Arguments

values	vector of numeric values (mostly -log10(p-values))
domain_min	numeric value that corresponds to the 'yellow' in the color scale
domain_max	numeric value that corresponds to the 'dark blue' in the color scale
n	number of bins to generate from the color scale

Value

The output is a corresponding vector of colors from the Viridis color scale with domain in range(domain_min, domain_max).

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>

publish_gostplot *Create and save an annotated Manhattan plot of enrichment results.*

Description

This function allows to highlight a list of selected terms on the Manhattan plot created with the gprofiler2::gostplot() function. The resulting plot is saved to a publication ready image if 'filename' is specified. The plot is very similar to the one shown in the g:GOSt web tool after clicking on circles.

Usage

```
publish_gostplot(p, highlight_terms = NULL, filename = NULL)
```

Arguments

p	ggplot object from gostplot(gostres, interactive = FALSE) function
highlight_terms	vector of selected term IDs from the analysis or a (subset) data.frame that has a column called 'term_id'. No annotation is added if set to NULL.
filename	file name to create on disk and save the annotated plot. Filename extension should be from c("png", "pdf", "jpeg", "tiff", "bmp").

Value

The output is a ggplot object.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>

Examples

```
gostres <- gost(c("Klf4", "Pax5", "Sox2", "Nanog"), organism = "mmusculus")
p <- gostplot(gostres, interactive = FALSE)
publish_gostplot(p, highlight_terms = c("GO:0001010", "WP:WP1763"))
```

publish_gosttable *Create and save a table with the functional enrichment analysis results.*

Description

This function creates a table mainly for the results from gost() function. However, if the input at least contains columns named 'term_id' and 'p_value' then any enrichment results data frame can be visualised in a table with this function.

Usage

```
publish_gosttable(gostres, highlight_terms = NULL, use_colors = TRUE,
  show_columns = c("source", "term_name", "term_size",
  "intersection_size"), filename = NULL)
```

Arguments

gostres	named list from gost() function (with names 'result' and 'meta') or a data frame that has columns named "term_id" and "p_value(s)".
highlight_terms	vector of selected term IDs from the analysis or a (subset) data.frame that has a column called 'term_id'. All data is shown if set to NULL.
use_colors	if enabled, the p-values are highlighted in the viridis colorscale just as in g:Profiler, otherwise the table has no background colors.
show_columns	names of additional columns to show besides term_id and p_value. By default the output table shows additional columns named "source", "term_name", "term_size", "intersection_size"
filename	file name to create on disk and save the annotated plot. Filename extension should be from c("png", "pdf", "jpeg", "tiff", "bmp").

Details

The output table is very similar to the one shown under the Manhattan plot.

Value

The output is a ggplot object.

Author(s)

Liis Kolberg <liis.kolberg@ut.ee>

Examples

```
gostres <- gost(c("Klf4", "Pax5", "Sox2", "Nanog"), organism = "mmusculus")
publish_gosttable(gostres, highlight_terms = c("GO:0001010", "WP:WP1763"))
```

set_base_url *Set the base URL.*

Description

Function to change the g:Profiler base URL. Useful for overriding the default URL (http://biit.cs.ut.ee/gprofiler) with the beta (http://biit.cs.ut.ee/gprofiler_beta) or an archived version (available starting from the version e94_eg41_p11, e.g. http://biit.cs.ut.ee/gprofiler_archive3/e94_eg41_p11).

Usage

```
set_base_url(url)
```

Arguments

url	the base URL.

set_tls_version *Set the TLS version to use for SSL*

Description

Set the TLS version. Could be useful at environments where SSL was built without TLS 1.2 support

Usage

```
set_tls_version(v)
```

Arguments

v	version: "1.2" (default), "1.1" (fallback)

set_user_agent *Set custom user agent string.*

Description

Set the HTTP User-Agent string. Useful for overriding the default user agent for packages that depend on gprofiler2 functionality.

Usage

```
set_user_agent(ua, append = F)
```

Arguments

ua	the user agent string.
append	logical indicating whether to append the passed string to the default user agent string.

Index