Package ‘gProfileR’

September 7, 2019

Version 0.6.8
License GPL (>= 2)
Description This package has been deprecated and will not be updated.
    New users should use the package ’gprofiler2’ (<https://CRAN.R-project.org/package=gprofiler2>)
    for up-to-date data and improved functionality.
    Functional enrichment analysis, gene identifier conversion and
    mapping homologous genes across related organisms via the ’g:Profiler’ toolkit
Title Interface to the ’g:Profiler’ Toolkit
Author Juri Reimand <juri.reimand@ut.ee>, Raivo Kolde
    <rkolde@gmail.com>, Tambet Arak <tambet.arak@gmail.com>
Maintainer Ivan Kuzmin <ivan.kuzmin@ut.ee>
BugReports http://biit.cs.ut.ee/gprofiler/contact.cgi
Depends R (>= 2.10)
Imports RCurl, plyr, utils
Collate ’gProfileR.R’
RoxygenNote 6.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2019-09-07 15:20:02 UTC

R topics documented:
gconvert ................................................................. 2
get_base_url .......................................................... 3
get_tls_version ....................................................... 3
get_user_agent ....................................................... 3
gorth ................................................................. 4
 gprofiler ............................................................ 5
set_base_url ........................................................ 7
gconvert

Description

Interface to the g:Convert tool. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.

Usage

```r
gconvert(query, organism = "hsapiens", target = "ENSG",
         region_query = F, numeric_ns = "", mthreshold = Inf,
         filter_na = T, df = T)
```

Arguments

- **query**: list of gene IDs.
- **organism**: organism name.
- **target**: target namespace.
- **region_query**: interpret query as chromosomal ranges.
- **numeric_ns**: namespace to use for fully numeric IDs.
- **mthreshold**: maximum number of results per initial alias to show.
- **filter_na**: logical indicating whether to filter out results without a corresponding target.
- **df**: logical indicating whether the output will be a data.frame or list.

Value

The output can be either a list or a data.frame. The list has an entry for every input gene. The data frame is a table closely corresponding to the web interface output.

Author(s)

Juri Reimand <jyri.reimand@ut.ee>, Raivo Kolde <rkolde@gmail.com>, Tambet Arak <tambet.arak@gmail.com>

References

**get_base_url**

Examples

```r
## Not run:
gconvert(c("POU5F1", "SOX2", "NANOG"), organism = "hsapiens", target="AFFY_HG_U133_PLUS_2")
```

## End(Not run)

---

**get_base_url**

*Get the base URL.*

**Description**

Get the base URL.

**Usage**

```r
get_base_url()
```

---

**get_tls_version**

*Get the TLS version for SSL*

**Description**

Get the TLS version for SSL.

**Usage**

```r
get_tls_version()
```

---

**get_user_agent**

*Get current user agent string.*

**Description**

Get the HTTP User-Agent string.

**Usage**

```r
get_user_agent()
```
**Description**

Interface to the g:Orth tool. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.

**Usage**

```r
gorth(query, source_organism = "hsapiens",
      target_organism = "mmusculus", region_query = F, numeric_ns = "",
      mthreshold = Inf, filter_na = T, df = T)
```

**Arguments**

- `query` : list of gene IDs to be translated.
- `source_organism` : name of the source organism.
- `target_organism` : name of the target organism.
- `region_query` : interpret query as chromosomal ranges.
- `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.
- `df` : logical indicating whether the output will be a data.frame or list.

**Details**

To alleviate the problem of having many orthologs per gene (most of them uninformative) one can set a threshold for the number of results. The program tries to find the most informative by selecting the most popular ones.

**Value**

The output can be either a list or a data.frame. The list has an entry for every input gene. The data frame is a table closely corresponding to the web interface output.

**Author(s)**

Raivo Kolde <rkolde@gmail.com>, Juri Reimand <juri.reimand@ut.ee>, Tambet Arak <tambet arak@gmail.com>

**References**

gprofiler

Examples

```
## Not run:
gorth(c("Klf4","Pax5","Sox2","Nanog"), source_organism="mmusculus", target_organism="hsapiens")

## End(Not run)
```

Description

Interface to the g:Profiler tool for finding enrichments in gene lists. Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'. If requesting PNG output, the request is directed to the g:GOST tool in case 'query' is a vector and the g:Cocoa (compact view of multiple queries) tool in case 'query' is a list. PNG output can fail (return FALSE) in case the input query is too large. In such case, it is advisable to fall back to a non-image request.

Usage

```
gprofiler(query, organism = "hsapiens", sort_by_structure = T, ordered_query = F, significant = T, exclude_iea = F, underrep = F, evcodes = F, region_query = F, max_p_value = 1, min_set_size = 0, max_set_size = 0, min_isect_size = 0, correction_method = "analytical", hier_filtering = "none", domain_size = "annotated", custom_bg = "", numeric_ns = "", png_fn = NULL, include_graph = F, src_filter = NULL)
```

Arguments

- **query**: vector of gene IDs or a list of such vectors. In the latter case, the query is directed to g:Cocoa, which yields a different graphical output if requested with the `png_fn` parameter.
- **organism**: organism name.
- **sort_by_structure**: whether hierarchical sorting is enabled or disabled.
- **ordered_query**: in case output gene lists are ranked this option may be used to get GSEA style p-values.
- **significant**: whether all or only statistically significant results should be returned.
- **exclude_iea**: exclude electronic annotations (IEA).
- **underrep**: measure underrepresentation.
- **evcodes**: include GO evidence codes as the final column of output. Note that this can decrease performance and make the query slower.
- **region_query**: interpret query as chromosomal ranges.
max_p_value: custom p-value threshold, results with a larger p-value are excluded.

min_set_size: minimum size of functional category, smaller categories are excluded.

max_set_size: maximum size of functional category, larger categories are excluded.

min_isect_size: minimum size of the overlap (intersection) between query and functional category, smaller intersections are excluded.

correction_method: the algorithm used for determining the significance threshold, one of "gSCS", "fdr", "bonferroni".

hier_filtering: hierarchical filtering strength, one of "none", "moderate", "strong".

domain_size: statistical domain size, one of "annotated", "known".

custom_bg: vector of gene names to use as a statistical background.

numeric_ns: namespace to use for fully numeric IDs.

png_fn: request the result as PNG image and write it to png_fn.

include_graph: request inclusion of network data with the result.

src_filter: 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, MI, CORUM, HP, HPA, OMIM. Please see the g:GOSt web tool for the comprehensive list and details on incorporated data sources.

Value

A data frame with the enrichment analysis results. If the input consisted of several lists the corresponding list is indicated with a variable 'query number'. When requesting a PNG image, either TRUE or FALSE, depending on whether a non-empty result was received and a file written or not, respectively. If 'include_graph' is set, the return value may include the attribute 'networks', containing a list of all network sources, each in turn containing a list of graph edges. The edge structure is a list containing the two interacting symbols and two boolean values (in that order), indicating whether the first or second interactor is part of the input query (core nodes).

Author(s)

Juri Reimand <jyri.reimand@ut.ee>, Raivo Kolde <rkolde@gmail.com>, Tambet Arak <tambet.arak@gmail.com>

References


Examples

## Not run:
gprofiler(c("Klf4", "Pax5", "Sox2", "Nanog"), organism = "mmusculus")

## End(Not run)
Description

Set the base URL. Useful for overriding the default URL (http://biit.cs.ut.ee/gprofiler) with the bleeding-edge beta or an archived version.

Usage

```r
set_base_url(url)
```

Arguments

- `url` the base URL.

---

Description

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

Usage

```r
set_tls_version(v)
```

Arguments

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

---

Description

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

Usage

```r
set_user_agent(ua, append = F)
```
### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ua</code></td>
<td>the user agent string.</td>
</tr>
<tr>
<td><code>append</code></td>
<td>logical indicating whether to append the passed string to the default user agent string.</td>
</tr>
</tbody>
</table>
Index

gconvert, 2
get_base_url, 3
get_tls_version, 3
get_user_agent, 3
gorth, 4
gprofiler, 5

set_base_url, 7
set_tls_version, 7
set_user_agent, 7