Package 'CTDquerier'

July 3, 2025

Title Package for CTDbase data query, visualization and downstream analysis

Version 2.17.0

Description Package to retrieve and visualize data from the Comparative Toxicogenomics Database (http://ctdbase.org/). The downloaded data is formated as DataFrames for further downstream analyses.

Depends R (>= 4.1)

License MIT + file LICENSE

Encoding UTF-8

LazyData false

Imports RCurl, stringr, S4Vectors, stringdist, ggplot2, igraph, utils, grid, gridExtra, methods, stats, BiocFileCache

VignetteBuilder rmarkdown

Suggests BiocStyle, knitr, rmarkdown

RoxygenNote 7.2.1

biocViews Software, BiomedicalInformatics, Infrastructure, DataImport, DataRepresentation, GeneSetEnrichment, NetworkEnrichment, Pathways, Network, GO, KEGG

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

git_branch devel

git_last_commit 14628b1

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-07-03

Author Carles Hernandez-Ferrer [aut], Juan R. Gonzalez [aut], Xavier Escribà-Montagut [cre]

Maintainer Xavier Escribà-Montagut <xavier.escriba@isglobal.org>

Contents

`CTDdata` *Class CTDdata*

Description

Class resulting of `query_ctd_gene`, `query_ctd_chem` and `query_ctd_dise`. It is used to encapsulate all the information in *CTDbase* for given set of genes, chemicals or diseases.

Usage

```
## S4 method for signature 'CTDdata'
enrich(x, y, universe, use = "curated", warnings = TRUE, ...)

## S4 method for signature 'CTDdata'
get_table(object, index_name, ...)

## S4 method for signature 'CTDdata'
get_terms(object)

## S4 method for signature 'CTDdata,ANY'
plot(x, y, index_name = "base", representation = "heatmap", ...)
```

Arguments

`x`	Object of class `CTDdata`.
`y`	NOT USED
`universe`	String vector of genes used as universe. If not provided, all genes in CTDbase are used.
`use`	Select if all or only curated relations are used.
`warnings`	Shows or hiddes warnings.
`...`	NOT USED
`object`	Object of class `CTDdata`.
`index_name`	Name of the plot to be draw. See `CTDdata`'s detail section for more information (`?`CTDdata-class``). representation.
`representation`	Can take values `"heatmap"` or `"network"`.

Details

CTDdata objects provides with a summarized representation of the downloaded data obtained from the standard show method. For instance, a CTDdata created using query_ctd_chem shows:

```
Object of class 'CTDdata'
-------------------------
. Type: CHEMICAL
. Creation (timestamp): 2018-03-13 13:11:50
. Length: 2
. Items: IRON, ..., AIR POLLUTANTS
. Diseases: 1755 ( 203 / 3322 )
. Chemical-gene interactions: 2070 ( 2799 )
. KEGG pathways: 637 ( 637 )
. GO terms: 3641 ( 3641 )
```

The information shows corresponds to:

- Type: Indicates the source (chemical, gene or disease) used to create the object.
- Creation: Shows the time-stamp from the creation time.
- Length: Shows the number of terms used to create the object.
- Items: Shows some of the terms used to create the object.
- Diseases: Corresponds to the unique number of diseases obtained in the query. In parenthesis: number of curated chemical-diseases and total number of chemical-diseases association.
- Chemical-gene interactions: Indicates the unique number of chemical-gene interactions. In parenthesis the total number of chemical-gene interactions.
- KEGG pathways: Shows the unique number of KEGG pathway versus chemical associations. In parenthesis the total number of associations.
- GO terms: Shows the unique number of GO terms versus chemical associations. In parenthesis the total number of associations.

CTDdata objects allows many types of representation according to the different sources (chemical, gene or disease). The used method is plot, matching the argument x with a CTDdata object. The argument index_name indicates the type of plot to be draw. The default value of index_name is "base".

- "base": shows a bar-plot indicating the number of lost & found terms for the given object.

For *gene* queries, index_name can take values:

- "disease": (argument representation must be "heatmap") shows shows the inference score that associates the given genes with diseases according to CTDbase.
- "chemical interactions": (argument representation must be "heatmap") shows the number of reference that cites the association between the given genes and chemicals.
- "gene-gene interaction": (argument representation can be "network" and "heatmap") in the network representation teh original genes are dark-colored while the other genes are light-colored. Both plots allows to to explore the gene-gene interactions.
- "kegg pathways": (argument representation must be "network") shows the linked between genes and KEGG pathways.
- "go terms": (argument representation must be "network") shows the links between genes and GO terms.

For *chemical* queries, index_name can take values:

- "gene interactions": (argument representation can be "network" and "heatmap") shows the gene-chemical interactions. Network representation includes the "mechanism" of the interactions.
- "disease": (argument representation can be "network" and "heatmap") shows the inference score of the link between chemicals and diseases.
- "kegg pathways": (argument representation must be "network") shows the P-Value of relation between KEGG pathways and chemicals.
- "go terms": (argument representation must be "network") shows the P-Value of relation between GO terms and chemicals.

For *disease* queries, index_name can take values:

- "gene": (argument representation must be "heatmap") shows the number of references linking a set of genes with a set of diseases.
- "chemical": (argument representation must be "heatmap"): shows the inference-score linking diseases with chemicals.
- "kegg pathways": (argument representation must be "network") shows the pathways linked to a set of diseases.

The other arguments of plot functions follows:

- subset.chemical: filters the chemicals to be include into the plot.
- subset.gene: filters the genes to be include into the plot.
- subset.pathway: filters the KEGG pathways or GO terms included into the plot.
- subset.source: filters the origin in gene-gene interaction network.
- subset.target: filters the end in gene-gene interaction network.
- field.score: can take values "Inference" or "Reference" depending of the used source and representation.
- filter.score: allows to filter the relations to be included into the plot in base of the set of field.score.
- max.length: indicates the maximum number of characters of the names of each "item" in the plot.
- ontology: for the KEGG pathways, allows to filter the pathways in base of their ontology. By default: c("Biological Process", "Cellular Component", "Molecular Function").
- main: title to be displayed in network representations. For heatmap representations use: + ggtitle("TITLE").

Value

An object of class CTDdata

Methods (by generic)

- enrich(CTDdata): Method to perform enrichment analysis given two object of class CTDdata.
- get_table(CTDdata): Method to obtain a specific inner table from an CTDdata object.
- get_terms(CTDdata): Return a list with the terms found to create the object.
- plot(x = CTDdata, y = ANY): Generates a basic plot showing the number of terms that can be used to query CTDbase.

Slots

`timestamp` Character with the timestamp.

`type` Character saving `"GENE"`, `"CHEMICAL"` or `"DISEASE"` depending if it was created using `query_ctd_gene`, `query_ctd_chem` or `query_ctd_dise`

`terms` `DataFrame` with the genes, chemicals or diseases used to create the object.

`losts` Character with the terms used to create the object but that were nor present in CTDbase.

`gene_interactions` (Only for chemicals) Table with a relation of the genes interacting with the given chemicals.

`chemicals_interactions` (Only for genes) Table with a relation of the chemicals interacting with the given genes.

`diseases` Table with a relation of the diseases associated with given genes or chemicals.

`gene_gene_interactions` (Only for genes) Table with a relation of the genes interacting with the given genes.

`kegg` Table with a relation of the KEGG pathways affected by the given chemicals or where the given genes play a role.

`go` Table with a relation of the GO terms affected by the given chemicals or where the given genes play a role.

See Also

`query_ctd_gene` to create a `CTDdata` from a set of genes, `query_ctd_chem` to create a `CTDdata` from a set of chemicals, `query_ctd_dise` to create a `CTDdata` from a set of diseases, `get_table` to retrieve encapsulated data and `plot` to get nice plots from stored data.

`CTDquerier` *Package for CTDbase data query, data visualization and data analysis.*

Description

It can retrieve information related to genes, chemicals and diseases.

Data Download

`CTDquerier` offers two functions to query CTDbase (http://ctdbase.org): `query_ctd_gene` to query CTDbase given a set of genes; and `query_ctd_chem` to query CTDbase given a set of chemicals. Both functions returns `CTDdata` objects. Raw downloaded information can be retrieved from `CTDdata` using method `get_table`.

Data Visualization

`CTDdata` objects offers basic visualization of the downloaded information using standard `plot` method.

`download_ctd_chem` *Function to download checmicals available in CTDbase*

Description

This function download the "Chemical vocabulary" file (`CTD_chemicals.tsv.gz`) from `http://ctdbase.org/downloa`

Usage

```
download_ctd_chem(verbose = FALSE, ask = TRUE)
```

Arguments

`verbose`	(default `FALSE`) If set to `TRUE` is shows relevant information of each step.
`ask`	(default `TRUE`) If `TRUE` it asks the the persistent location must be used.

Details

The field included in the file (`CTD_chemicals.tsv.gz`) are:

1. ChemicalName
2. ChemicalID (MeSH identifier)
3. CasRN (CAS Registry Number, if available)
4. Definition
5. ParentIDs (identifiers of the parent terms; '|'-delimited list),
6. TreeNumbers (identifiers of the chemical's nodes; '|'-delimited list),
7. ParentTreeNumbers (identifiers of the parent nodes; '|'-delimited list),
8. Synonyms ('|'-delimited list)
9. DrugBankIDs ('|'-delimited list)

Value

Passed name into `filename` argument if it could be download `1` otherwise.

Examples

```
download_ctd_chem()
file.exists( "CTD_chemicals.tsv.gz" )
```

download_ctd_dise *Function to download diseases available in CTDbase*

Description

This function download the "Disease vocabulary" file (CTD_diseases.tsv.gz) from http://ctdbase.org/downloads.

Usage

```
download_ctd_dise(verbose = FALSE, ask = TRUE)
```

Arguments

verbose	(default FALSE) If set to TRUE is shows relevant information of each step.
ask	(default TRUE) If TRUE it asks the the persistent location must be used.

Details

The field included in the file (CTD_diseases.tsv.gz) are:

1. DiseaseName
2. DiseaseID (MeSH or OMIM identifier)
3. Definition
4. AltDiseaseIDs (alternative identifiers; '|'-delimited list)
5. ParentIDs (identifiers of the parent terms; '|'-delimited list)
6. TreeNumbers (identifiers of the disease's nodes; '|'-delimited list)
7. ParentTreeNumbers (identifiers of the parent nodes; '|'-delimited list)
8. Synonyms ('|'-delimited list)
9. SlimMappings (MEDIC-Slim mappings; '|'-delimited list)

Value

Passed name into filename argument if it could be download 1 otherwise.

Examples

```
download_ctd_dise()
file.exists( "CTD_diseases.tsv.gz" )
```

download_ctd_genes *Function to download genes available in CTDbase*

Description

This function download the "Gene vocabulary" file (CTD_genes.tsv.gz) from http://ctdbase.org/downloads.

Usage

```
download_ctd_genes(verbose = FALSE, ask = TRUE)
```

Arguments

verbose	(default FALSE) If set to TRUE is shows relevant information of each step.
ask	(default TRUE) If TRUE it asks the the persistent location must be used.

Details

The field included in the file (CTD_genes.tsv.gz) are:

1. GeneSymbol
2. GeneName
3. GeneID (NCBI Gene identifier)
4. AltGeneIDs (alternative NCBI Gene identifiers; '|'-delimited list)
5. Synonyms ('|'-delimited list)
6. BioGRIDIDs ('|'-delimited list)
7. PharmGKBIDs ('|'-delimited list)
8. UniprotIDs ('|'-delimited list)

Value

Passed name into filename argument if it could be download 1 otherwise.

Examples

```
download_ctd_genes()
```

enrich *Method to perform enrichment analysis using two* CTDdata *objects*

Description

This methods performs a fisher test using the genes in two objects of class CTDdata. The object in 'x' is used as source while the object on 'y' is used as universe. When object 'x' corresponds to an object created with query_ctd_gene, the used genes are the found terms in CTDbase. In the other cases (chemicals and disease CTDdata), the genes from the 'gene interactions' table are used. If universe is missing, all genes in CTDbase are used as universe.

Usage

```
enrich(x, y, universe, use = "curated", warnings = TRUE, ...)
```

Arguments

x	Object of class CTDdata.
y	Object of class CTDdata.
universe	Vector of strings corresponding to the genes to be used as universe.
use	(default: "curated") It can take values "curated" or "all" to filter or not filter for curated the genes into the CTDdata objects.
warnings	(default: TRUE).
...	NOT USED

Value

A list with class htest. Check fisher.test for more information.

Examples

```
# Example in a tryCatch, since we are performing a connection to a server we might
# get a refused connection due to a server rejection. Evaluate the recieved HTTP
# message to understand if the server is not available or if your IP adress is temporarly restricted
tryCatch({
data("gala")
air <- query_ctd_chem( terms = "Air Pollutants" )
hgnc_universe <- readRDS(paste0(path.package("CTDquerier"),"/extdata/universe.RDS"))
enrich(gala, air, hgnc_universe)
}, error = function(w){NULL})
```

gala CTDdata *for ilustrative purpouses*

Description

CTDdata with information of 258 genes downloaded from CTDbase. The object was created from from the genes obtained from the scientific article entitleed Case-control admixture mapping in Latino populations enriches for known asthma-associated genes" (Table E1) by Torgerson et. al. The genes were used to query CTDbase using query_ctd_genes function.

Usage

```
data("gala")
```

Format

An object of class CTDdata of length 1.

Value

An CTDdata object.

Examples

```
data("gala")
gala
```

get_table *Method to obtain a specific inner table from a* CTDdata *object.*

Description

Obtain the raw data from a CTDdata object, result from a query to CTDbase.

Usage

```
get_table(object, index_name, ...)
```

Arguments

object	Object of class CTDdata.
index_name	String indicating the type of data to obtain.
...	NOT USED

Details

Available tables are (index_name):

1. "gene interactions": (Only for chemicals) Table with a relation of the genes interacting with the given chemicals.
2. "chemical interactions": (Only for genes) Table with a relation of the chemicals interacting with the given genes.
3. "diseases": Table with a relation of the diseases associated with given genes or chemicals.
4. "gene-gene interactions": (Only for genes) Table with a relation of the genes interacting with the given genes.
5. "kegg pathways": Table with a relation of the KEGG pathways affected by the given chemicals or where the given genes play a role.
6. "go terms": Table with a relation of the GO terms affected by the given chemicals or where the given genes play a role.

Value

A DataFrame containing the raw result from CTDdata.

Examples

```
data("gala")
get_table(gala, "diseases")[1:3, ]
```

---

get_terms *Getter to obtain the terms used to perform a query into CTDbase*

Description

Getter to obtain the terms used to perform a query into CTDbase

Usage

```
get_terms(object)
```

Arguments

object	Object of class CTDdata.

Value

A list with two accessors: "used" for the terms that exists in CTDbase, and "lost" with the terms that do not exists in CTDbase.

Examples

```
data("gala")
get_terms(gala)[["lost"]]
```

leaf_plot *Function to create a leaf plot*

Description

This functions taked a `data.frame` and returns a `gtable` with three plots. The left-leafes, the axis names and the right-leafes.

Usage

```
leaf_plot(
  dta,
  label = "name",
  valueLeft = "var1",
  valueRight = "var2",
  titleLeft = NULL,
  titleRight = NULL,
  colorLeft = "#FF7F50",
  colorRight = "#20B2AA"
)
```

Arguments

`dta`	`data.frame` with, at last, three columns corresponding to the axis labels, the left values and the right values.
`label`	(default `"name"`) Name of the column in `dta` with the labels.
`valueLeft`	(default `"var1"`) Name of the column with the values for the left plot.
`valueRight`	(default `"var2"`) Name of the column with the values for the right plot.
`titleLeft`	(default `NULL`) Character used as a title for the left plot.
`titleRight`	(default `NULL`) Character used as a title for the right plot.
`colorLeft`	(default `"#FF7F50"`) Color for left plot bars.
`colorRight`	(default `"#20B2AA"`) Color for right plot bars.

Value

A ggplo2 object.

Examples

```
data <- data.frame(
   labels = LETTERS[1:15],
   right = runif(n = 15) * 11,
   left = runif(n = 15) * 9
)
leaf_plot( data, "labels", "left", "right", "runif09", "runif11")
```

load_ctd_chem *Function to load the* .tsv.gz *file for chemicals*

Description

Function to load the .tsv.gz file for chemicals

Usage

```
load_ctd_chem(verbose = FALSE)
```

Arguments

verbose	(default FALSE) If set to TRUE messages and warnings are raised.

Details

The field included in the file (CTD_chemicals.tsv.gz) are:

1. ChemicalName
2. ChemicalID (MeSH identifier)
3. CasRN (CAS Registry Number, if available)
4. Definition
5. ParentIDs (identifiers of the parent terms; '|'-delimited list),
6. TreeNumbers (identifiers of the chemical's nodes; '|'-delimited list),
7. ParentTreeNumbers (identifiers of the parent nodes; '|'-delimited list),
8. Synonyms ('|'-delimited list)
9. DrugBankIDs ('|'-delimited list)

Value

A data.frame with the content of the file "CTD_genes.tsv.gz"

Examples

```
if(download_ctd_chem()){
fdl <- load_ctd_chem()
dim( fdl )
}
```

load_ctd_dise *Function to load the* .tsv.gz *file for disease*

Description

Function to load the .tsv.gz file for disease

Usage

```
load_ctd_dise(verbose = FALSE)
```

Arguments

verbose	(default FALSE) If set to TRUE messages and warnings are raised.

Details

The field included in the file (CTD_diseases.tsv.gz) are:

1. DiseaseName
2. DiseaseID (MeSH or OMIM identifier)
3. Definition
4. AltDiseaseIDs (alternative identifiers; '|'-delimited list)
5. ParentIDs (identifiers of the parent terms; '|'-delimited list)
6. TreeNumbers (identifiers of the disease's nodes; '|'-delimited list)
7. ParentTreeNumbers (identifiers of the parent nodes; '|'-delimited list)
8. Synonyms ('|'-delimited list)
9. SlimMappings (MEDIC-Slim mappings; '|'-delimited list)

Value

A data.frame with the content of the file "CTD_genes.tsv.gz"

Examples

```
if(download_ctd_dise()){
fdl <- load_ctd_dise()
dim( fdl )
}
```

load_ctd_gene *Function to load the* .tsv.gz *file for genes*

Description

This function works in pair with download_ctd_genes. This function loads into the R session the downloaded "CTD_genes.tsv.gz" file.

Usage

```
load_ctd_gene(verbose = FALSE)
```

Arguments

verbose	(default FALSE) If set to TRUE messages and warnings are raised.

Details

The field included in the file (CTD_genes.tsv.gz) are:

1. GeneSymbol
2. GeneName
3. GeneID (NCBI Gene identifier)
4. AltGeneIDs (alternative NCBI Gene identifiers; '|'-delimited list)
5. Synonyms ('|'-delimited list)
6. BioGRIDIDs ('|'-delimited list)
7. PharmGKBIDs ('|'-delimited list)
8. UniprotIDs ('|'-delimited list)

Value

A data.frame with the content of the file "CTD_genes.tsv.gz"

Examples

```
if(download_ctd_genes()){
fdl <- load_ctd_gene()
dim( fdl )
}
```

query_ctd_chem *Function to query CTDbase using chemical terminology (Chemical Name)*

Description

This function checks for CTDbase gene vocabulary and query CTDbase for each one, downloading chemical-genes interactions, associated diseases, associated KEGG pathways and associated GO terms.

Usage

```
query_ctd_chem(terms, max.distance = 10, ask = FALSE, verbose = FALSE)
```

Arguments

terms	Character vector with the chemicals used in the query.
max.distance	(default 10) Maximum distance allowed between a given element in terms argument and a possible match in CTDbase.
ask	(default TRUE) If TRUE it asks the the persistent location must be used to save the vocabulary if it was not downloaded previously.
verbose	(default FALSE) If set to TRUE is shows relevant information of each step.

Value

An object of class CTDdata.

Examples

```
# Example in a tryCatch, since we are performing a connection to a server we might
# get a refused connection due to a server rejection. Evaluate the recieved HTTP
# message to understand if the server is not available or if your IP adress is temporarly restricted
rst <- tryCatch({query_ctd_chem( terms = c( "Iron", "Air Pollutants" ), verbose = TRUE )}, error = function(w){N
```

query_ctd_dise *Function to query CTDbase using disease terminology*

Description

This function checks for CTDbase disease vocabulary and query CTDbase for each one, downloading disease-gene interactions, chemicals interactions, associated diseases, associated KEGG pathways and associated GO terms.

Usage

```
query_ctd_dise(terms, ask = TRUE, verbose = FALSE)
```

Arguments

terms	Character vector with the diseases used in the query.
ask	(default TRUE) If TRUE it asks the the persistent location must be used to save the vocabulary if it was not downloaded previously.
verbose	(default FALSE) If set to TRUE is shows relevant information of each step.

Value

An object of class CTDdata.

Examples

```
# Example in a tryCatch, since we are performing a connection to a server we might
# get a refused connection due to a server rejection. Evaluate the recieved HTTP
# message to understand if the server is not available or if your IP adress is temporarly restricted
rst <- tryCatch({query_ctd_dise( terms = "Asthma", verbose = TRUE )}, error = function(w){NULL})
```

---

query_ctd_gene *Function to query CTDbase using gene terminology (Gene Symbol)*

Description

This function checks for CTDbase gene vocabulary and query CTDbase for each one, downloading gene-gene interactions, chemicals interactions, associated disease, associated KEGG pathways and associated GO terms.

Usage

```
query_ctd_gene(terms, ask = TRUE, verbose = FALSE)
```

Arguments

terms	Character vector with the genes used in the query.
ask	(default TRUE) If TRUE it asks the the persistent location must be used to save the vocabulary if it was not downloaded previously.
verbose	(default FALSE) If set to TRUE is shows relevant information of each step.

Value

An object of class CTDdata.

Examples

```
# Example in a tryCatch, since we are performing a connection to a server we might
# get a refused connection due to a server rejection. Evaluate the recieved HTTP
# message to understand if the server is not available or if your IP adress is temporarly restricted
rst <- tryCatch({query_ctd_gene( terms = c( "APP", "HMOX1A", "hmox1" ), verbose = TRUE )}, error = function(w){N
```

Index