

# Package ‘gcite’

October 13, 2022

**Type** Package

**Title** Google Citation Parser

**Version** 0.10.1

**Date** 2019-03-05

**Description** Scrapes Google Citation pages and creates data frames of citations over time.

**License** GPL-3

**LazyData** true

**LazyLoad** yes

**Imports** xml2, httr, rvest, stats, pbapply, data.table, wordcloud, tm, graphics

**RoxygenNote** 6.1.1

**Encoding** UTF-8

**Suggests** covr, testthat, spelling

**Language** en-US

**NeedsCompilation** no

**Author** John Muschelli [aut, cre] (<<https://orcid.org/0000-0001-6469-1750>>)

**Maintainer** John Muschelli <muschelli1j2@gmail.com>

**Repository** CRAN

**Date/Publication** 2019-03-06 23:52:52 UTC

## R topics documented:

author_cloud . . . . .	2
gcite . . . . .	3
gcite_author_info . . . . .	4
gcite_citation_index . . . . .	5
gcite_citation_page . . . . .	6
gcite_cite_over_time . . . . .	7
gcite_graph . . . . .	8

gcite_main_graph . . . . .	9
gcite_papers . . . . .	9
gcite_paper_df . . . . .	10
gcite_stopwords . . . . .	11
gcite_url . . . . .	11
gcite_username . . . . .	12
gcite_user_info . . . . .	13
gcite_wordcloud . . . . .	14
gcite_wordcloud_spec . . . . .	14
is_travis . . . . .	15
set_cookies_txt . . . . .	15
title_cloud . . . . .	16

<b>Index</b>	<b>17</b>
--------------	-----------

---

author_cloud	<i>Make Wordcloud of authors from Papers</i>
--------------	--

---

## Description

Takes a vector of authors and then creates a frequency table of those words and plots a wordcloud

## Usage

```
author_cloud(authors, addstopwords = gcite_stopwords(),
             author_pattern = NULL, split = ",", verbose = TRUE,
             colors = c("#66C2A4", "#41AE76", "#238B45", "#006D2C", "#00441B"), ...)
```

```
author_frequency(authors, author_pattern = NULL, split = ",",
                 addstopwords = gcite_stopwords(), verbose = TRUE)
```

## Arguments

authors	Vector of authors of papers
addstopwords	Additional words to remove from wordcloud
author_pattern	regular expression for patterns to exclude from individual authors
split	split author names (default ", "), passed to <code>strsplit</code>
verbose	Print diagnostic messages
colors	color words from least to most frequent. Passed to <code>gcite_wordcloud_spec</code>
...	additional options passed to <code>gcite_wordcloud_spec</code>

## Value

A data.frame of the words and the frequencies of the authors

**Examples**

```
## Not run:
L = gcite_author_info("John Muschelli")
paper_df = L$paper_df
authors = paper_df$authors
author_cloud(authors)

## End(Not run)
```

gcite

*Google Citations Information***Description**

Wraps getting the information from Google Citations and plotting the wordcloud

**Usage**

```
gcite(author, user, plot_wordcloud = TRUE, author_args = list(),
      title_args = list(), warn = FALSE, force = FALSE, sleeptime = 0,
      ...)
```

**Arguments**

author	author name separated by spaces
user	user ID for google Citations
plot_wordcloud	should the wordcloud be plotted
author_args	Arguments to pass to <a href="#">author_cloud</a>
title_args	Arguments to pass to <a href="#">title_cloud</a>
warn	should warnings be printed from wordcloud?
force	If passing a URL and there is a failure, should the program return NULL, passed to <a href="#">gcite_citation_page</a>
sleeptime	time in seconds between http requests, to avoid Google Scholar rate limit
...	additional options passed to <a href="#">gcite_user_info</a> and therefore <a href="#">GET</a>

**Value**

List from either [gcite\\_user\\_info](#) or [gcite\\_author\\_info](#)

**Examples**

```
if (!is_travis() & !is_cran()) {
  res = gcite(author = "John Muschelli")
  paper_df = res$paper_df
  gcite_wordcloud(paper_df)
  author_cloud(paper_df$authors)
}
```

---

gcite\_author\_info      *Getting User Information from name*

---

### Description

Calls [gcite\\_user\\_info](#) after getting the user identifier

### Usage

```
gcite_author_info(author, ask = TRUE, pagesize = 100, verbose = TRUE,
  secure = TRUE, force = FALSE, read_citations = TRUE,
  sleeptime = 0, ...)
```

### Arguments

author	author name separated by spaces
ask	If multiple authors are found, should a menu be given
pagesize	Size of pages, max 100, passed to <a href="#">gcite_url</a>
verbose	Print diagnostic messages
secure	use https vs. http
force	If passing a URL and there is a failure, should the program return NULL, passed to <a href="#">gcite_citation_page</a>
read_citations	Should all citation pages be read?
sleeptime	time in seconds between http requests, to avoid Google Scholar rate limit
...	Additional arguments passed to <a href="#">GET</a>

### Value

A list of citations, citation indices, and a data.frame of authors, journal, and citations, and a data.frame of the links to all paper URLs.

### Examples

```
## Not run:
if (!is_travis()) {
df = gcite_author_info(author = "John Muschelli", secure = FALSE)
}

## End(Not run)
if (is_travis() & !is_cran()) {
df = gcite_author_info(author = "Jiawei Bai", secure = FALSE)
}
```

---

gcite\_citation\_index *Parse Google Citation Index*

---

## Description

Parses a google citation indices (h-index, etc.) from main page

## Usage

```
gcite_citation_index(doc, ...)  
  
## S3 method for class 'xml_node'  
gcite_citation_index(doc, ...)  
  
## S3 method for class 'xml_document'  
gcite_citation_index(doc, ...)  
  
## S3 method for class 'character'  
gcite_citation_index(doc, ...)
```

## Arguments

doc	A xml_document or the url for the main page
...	Additional arguments passed to <a href="#">GET</a> if doc is a URL

## Value

A matrix of indices

## Examples

```
library(httr)  
library(rvest)  
library(gcite)  
url = "https://scholar.google.com/citations?user=T9eqZgMAAAAJ"  
url = gcite_url(url = url, pagesize = 10, cstart = 0)  
if (!is_travis() & !is_cran()) {  
  ind = gcite_citation_index(url)  
  doc = content(httr::GET(url))  
  ind = gcite_citation_index(doc)  
  ind_nodes = rvest::html_nodes(doc, "#gsc_rsb_st")[[1]]  
  ind = gcite_citation_index(ind_nodes)  
}
```

---

gcite\_citation\_page *Parse Google Citation Index*

---

## Description

Parses a google citation indices (h-index, etc.) from main page

## Usage

```
gcite_citation_page(doc, title = NULL, force = FALSE, ...)
```

```
## S3 method for class 'xml_nodeset'  
gcite_citation_page(doc, title = NULL,  
  force = FALSE, ...)
```

```
## S3 method for class 'xml_document'  
gcite_citation_page(doc, title = NULL,  
  force = FALSE, ...)
```

```
## S3 method for class 'character'  
gcite_citation_page(doc, title = NULL,  
  force = FALSE, ...)
```

```
## S3 method for class 'list'  
gcite_citation_page(doc, title = NULL, force = FALSE,  
  ...)
```

```
## Default S3 method:  
gcite_citation_page(doc, title = NULL, force = FALSE,  
  ...)
```

## Arguments

doc	A xml_document or the url for the main page
title	title of the article
force	If passing a URL and there is a failure, should the program return NULL?
...	arguments passed to <a href="#">GET</a>

## Value

A matrix of indices

## Examples

```
library(httr)  
library(rvest)  
url = paste0("https://scholar.google.com/citations?view_op=view_citation&"),
```

```

"hl=en&oe=ASCII&user=T9eqZgMAAAAJ&pagesize=100&",
"citation_for_view=T9eqZgMAAAAJ:W70EmFMy1HYC")
url = gcite_url(url = url, pagesize = 10, cstart = 0)
if (!is_travis() & !is_cran()) {
  ind = gcite_citation_page(url)
  doc = content(httr::GET(url))
  ind = gcite_citation_page(doc)
  ind_nodes = html_nodes(doc, "#gsc_vcd_table div")
  ind_nodes = html_nodes(ind_nodes, xpath = '//div[@class = "gs_scl"]')
  ind = gcite_citation_page(ind_nodes)
}

```

---

gcite\_cite\_over\_time *Parse Google Citations Over Time*

---

## Description

Parses a google citations over time from the main Citation page

## Usage

```

gcite_cite_over_time(doc, ...)

## S3 method for class 'xml_node'
gcite_cite_over_time(doc, ...)

## S3 method for class 'xml_document'
gcite_cite_over_time(doc, ...)

## S3 method for class 'character'
gcite_cite_over_time(doc, ...)

## Default S3 method:
gcite_cite_over_time(doc, ...)

```

## Arguments

doc	A xml_document or the url for the main page
...	arguments passed to <a href="#">GET</a>

## Value

A matrix of citations

**Examples**

```

library(httr)
library(rvest)
url = "https://scholar.google.com/citations?user=T9eqZgMAAAAJ"
url = gcite_url(url = url, pagesize = 10, cstart = 0)
if (!is_travis() & !is_cran()) {
#' ind = gcite_cite_over_time(url)
doc = content(httr::GET(url))
ind = gcite_cite_over_time(doc)
ind_nodes = rvest::html_nodes(doc, ".gsc_md_hist_b")
ind = gcite_cite_over_time(ind_nodes)
}

```

---

gcite\_graph

*Parse Google Citation Graph*


---

**Description**

Parses a google citation bar graph from html

**Usage**

```

gcite_graph(citations, ...)

## S3 method for class 'xml_node'
gcite_graph(citations, ...)

## S3 method for class 'xml_document'
gcite_graph(citations, ...)

## S3 method for class 'character'
gcite_graph(citations, ...)

## Default S3 method:
gcite_graph(citations, ...)

```

**Arguments**

citations      A list of nodes or xml\_node  
...             arguments passed to [GET](#)

**Value**

A matrix of citations and years



---

gcite_main_graph	<i>Parse Google Citation Graph</i>
------------------	------------------------------------

---

**Description**

Parses a google citation bar graph from html

**Usage**

```
gcite_main_graph(citations, ...)  
  
## S3 method for class 'xml_document'  
gcite_main_graph(citations, ...)  
  
## S3 method for class 'character'  
gcite_main_graph(citations, ...)  
  
## Default S3 method:  
gcite_main_graph(citations, ...)
```

**Arguments**

citations	A list of nodes or xml_node
...	arguments passed to <a href="#">GET</a>

**Value**

A matrix of citations and years

---

gcite_papers	<i>Parse Google Citation Index</i>
--------------	------------------------------------

---

**Description**

Parses a google citation indices (h-index, etc.) from main page

**Usage**

```
gcite_papers(doc, ...)  
  
## S3 method for class 'xml_nodeset'  
gcite_papers(doc, ...)  
  
## S3 method for class 'xml_document'  
gcite_papers(doc, ...)
```

```
## S3 method for class 'character'
gcite_papers(doc, ...)

## Default S3 method:
gcite_papers(doc, ...)
```

### Arguments

doc            A xml\_document or the url for the main page  
 ...            Additional arguments passed to [GET](#) if doc is a URL

### Value

A matrix of indices

### Examples

```
library(httr)
library(rvest)
url = "https://scholar.google.com/citations?user=T9eqZgMAAAAJ"
url = gcite_url(url = url, pagesize = 10, cstart = 0)
if (!is_travis() & !is_cran()) {
  ind = gcite_papers(url)
  doc = content(httr::GET(url))
  ind = gcite_papers(doc)
  ind_nodes = rvest::html_nodes(doc, "#gsc_a_b")
  ind = gcite_papers(ind_nodes)
}
```

---

gcite_paper_df	<i>Get Paper Data Frame from Title URLs</i>
----------------	---

---

### Description

Get Paper Data Frame from Title URLs

### Usage

```
gcite_paper_df(urls, verbose = TRUE, force = FALSE, sleeptime = 0,
  ...)
```

### Arguments

urls            A character vector of urls, from all\_papers\$title\_link  
 verbose        Print diagnostic messages  
 force          If passing a URL and there is a failure, should the program return NULL, passed to [gcite\\_citation\\_page](#)

sleeptime      time in seconds between http requests, to avoid Google Scholar rate limit  
 ...              Additional arguments passed to [GET](#)

**Value**

A data.frame of authors, journal, and citations

**Examples**

```
if (!is_travis() & !is_cran()) {
  L = gcite_user_info(user = "uERvKpYAAAAJ",
    read_citations = FALSE)
  urls = L$all_papers$title_link
  paper_df = gcite_paper_df(urls = urls, force = TRUE)
}
```

---

gcite_stopwords	<i>Google Cite Stopwords</i>
-----------------	------------------------------

---

**Description**

Additional stopwords to remove from Google Cite results

**Usage**

```
gcite_stopwords()
```

**Value**

Character Vector

**Examples**

```
gcite_stopwords()
```

---

gcite_url	<i>Google Citations URL</i>
-----------	-----------------------------

---

**Description**

Simple wrapper for adding in pagesize and start values for the page

**Usage**

```
gcite_url(url, cstart = 0, pagesize = 100)
```

```
gcite_base_url(secure = TRUE)
```

```
gcite_user_url(user, secure = TRUE)
```

**Arguments**

url	URL of the google citations page
cstart	Starting value for the citation page
pagesize	number of citations to return, max is 100
secure	should https be used (default), instead of http
user	Username/user ID for Google Scholar Citations

**Value**

A character string

**Examples**

```
url = "https://scholar.google.com/citations?user=T9eqZgMAAAAJ"
gcite_url(url = url, pagesize = 100, cstart = 5)
```

---

gcite_username	<i>Google Citation Username Searcher</i>
----------------	--

---

**Description**

Search Google Citation for an author username

**Usage**

```
gcite_username(author, verbose = TRUE, ask = TRUE, secure = TRUE,
  ...)
```

**Arguments**

author	author name separated by spaces
verbose	Verbose diagnostic printing
ask	If multiple authors are found, should a menu be given
secure	use https vs. http
...	arguments passed to <a href="#">GET</a>

**Value**

A character vector of the username of the author

**Examples**

```
if (!is_travis() & !is_cran()) {
  gcite_username("John Muschelli")
}
```

---

gcite\_user\_info      *Getting User Information of papers*

---

**Description**

Loops through pages for all information on Google Citations

**Usage**

```
gcite_user_info(user, pagesize = 100, verbose = TRUE, secure = TRUE,
  force = FALSE, read_citations = TRUE, sleeptime = 0, ...)
```

**Arguments**

user	user ID for google Citations
pagesize	Size of pages, max 100, passed to <a href="#">gcite_url</a>
verbose	Print diagnostic messages
secure	use https vs. http
force	If passing a URL and there is a failure, should the program return NULL, passed to <a href="#">gcite_citation_page</a>
read_citations	Should all citation pages be read?
sleeptime	time in seconds between http requests, to avoid Google Scholar rate limit
...	Additional arguments passed to <a href="#">GET</a>

**Value**

A list of citations, citation indices, and a data.frame of authors, journal, and citations, and a data.frame of the links to all paper URLs and the character string of the user name.

**Examples**

```
## Not run:
if (!is_travis() & !is_cran()) {
  df = gcite_user_info(user = "uERvKpYAAAAJ")
}

## End(Not run)
```

---

gcite\_wordcloud      *Wordcloud of Google Citations Information*

---

### Description

Simple wrapper for [author\\_cloud](#) and [title\\_cloud](#)

### Usage

```
gcite_wordcloud(paper_df, author_args = list(), title_args = list(),
                warn = FALSE)
```

### Arguments

paper_df	A data.frame with columns of authors and titles
author_args	Arguments to pass to <a href="#">author_cloud</a>
title_args	Arguments to pass to <a href="#">title_cloud</a>
warn	should warnings be printed from wordcloud?

---

gcite\_wordcloud\_spec      *gcite Wordcloud default*

---

### Description

Simple wrapper for [wordcloud](#) with different defaults

### Usage

```
gcite_wordcloud_spec(words, freq, min.freq = 1, max.words = Inf,
                    random.order = FALSE, colors = c("#F768A1", "#DD3497", "#AE017E",
                    "#7A0177", "#49006A"), vfont = c("sans serif", "plain"), ...)
```

### Arguments

words	words to be plotted
freq	the frequency of those words
min.freq	words with frequency below min.freq will not be plotted
max.words	Maximum number of words to be plotted. least frequent terms dropped
random.order	plot words in random order. If false, they will be plotted in decreasing frequency
colors	color words from least to most frequent
vfont	passed to text for the font
...	additional options passed to <a href="#">wordcloud</a>

### Value

Nothing

---

is_travis	<i>Check if on Travis CI</i>
-----------	------------------------------

---

**Description**

Simple check for Travis CI for examples

**Usage**

```
is_travis()
```

```
is_cran()
```

**Value**

Logical if user is named travis

**Examples**

```
is_travis()
is_cran()
```

---

set_cookies_txt	<i>Set Cookies from Text file</i>
-----------------	-----------------------------------

---

**Description**

Set Cookies from Text file

**Usage**

```
set_cookies_txt(file)
```

**Arguments**

file	tab-delimited text file of cookies, to be read in using <a href="#">readLines</a> . Comments should start the line with the pound symbol
------	--

**Value**

Either NULL if no domains contain the word "scholar", or an object of class request from [set\\_cookies](#)

**Note**

This function searches for domains that contain the word "scholar"

---

title_cloud	<i>Make Wordcloud of Titles from Papers</i>
-------------	---

---

**Description**

Takes a vector of titles and then creates a frequency table of those words and plots a wordcloud

**Usage**

```
title_cloud(titles, addstopwords = gcite_stopwords(), ...)
```

```
paper_cloud(...)
```

```
title_word_frequency(titles, addstopwords = NULL)
```

**Arguments**

titles	Vector of titles of papers
addstopwords	Additional words to remove from wordcloud
...	additional options passed to <a href="#">gcite_wordcloud_spec</a>

**Value**

A data.frame of the words and the frequencies of the title words

**Examples**

```
## Not run:  
L = gcite_author_info("John Muschelli")  
paper_df = L$paper_df  
titles = paper_df$title  
title_cloud(titles)  
  
## End(Not run)
```



# Index

author\_cloud, [2](#), [3](#), [14](#)  
author\_frequency (author\_cloud), [2](#)

gcite, [3](#)  
gcite\_author\_info, [3](#), [4](#)  
gcite\_base\_url (gcite\_url), [11](#)  
gcite\_citation\_index, [5](#)  
gcite\_citation\_page, [3](#), [4](#), [6](#), [10](#), [13](#)  
gcite\_cite\_over\_time, [7](#)  
gcite\_graph, [8](#)  
gcite\_main\_graph, [9](#)  
gcite\_paper\_df, [10](#)  
gcite\_papers, [9](#)  
gcite\_stopwords, [11](#)  
gcite\_url, [4](#), [11](#), [13](#)  
gcite\_user\_info, [3](#), [4](#), [13](#)  
gcite\_user\_url (gcite\_url), [11](#)  
gcite\_username, [12](#)  
gcite\_wordcloud, [14](#)  
gcite\_wordcloud\_spec, [2](#), [14](#), [16](#)  
GET, [3–13](#)

is\_cran (is\_travis), [15](#)  
is\_travis, [15](#)

paper\_cloud (title\_cloud), [16](#)

readLines, [15](#)

set\_cookies, [15](#)  
set\_cookies\_txt, [15](#)  
strsplit, [2](#)

title\_cloud, [3](#), [14](#), [16](#)  
title\_word\_frequency (title\_cloud), [16](#)

wordcloud, [14](#)