Gathering Bibliometric Information using Scopus using rscopus

by John Muschelli

Abstract We demonstrate how to download author and affiliation using the `rscopus` package, interacing with the Elsevier Scopus API. We show histograms of the number of citations from an author, as well as calculating citation metrics with the output.

Introduction

We would like to gather information about publications, authors, and institutions with respect to published research. Scopus a repository of information about scientific articles and books, which includes information about authors, citations, and abstracts of these pieces of literature. Scopus claims to have the largest database of this information. Therefore, providing users an interface to this repository should be worthwhile.

One common task for researchers is to keep the curriculum vitae (CV) up to date. That requires having up to date information on the papers published and under submission. Keeping track of these papers can be tedious and automated solutions could exist, but are not widely used (THINK OF SERVICES THAT TRY TO do this). One conern is always to be missing certain crucial papers in your CV. Although **rscopus** does not provide these tools specifically, it can be used to consistently cross-reference information about publications with a CV.

Additionally on CVs, it is useful to present information of the impact of a paper on the CV. This can be done by highlighting certain pieces of information, such is done in NIH biosketches, or ranking them based on some metric. The metric commonly used is the number of citations. Also, information about the journal impact factor may be taken into account. We do not note that these are particularly good metrics or metrics that reflect true impact, but are simply those that we have seen used in practice.

rscopus allows you to interface with Scopus APIs and gather information about authors, affiliations, articles, and abstracts. The **bibliometrix** package provides a level of integration that is useful for using multiple packages that deal with bibliometric data, incorporating functionality from **rscopus**. The **bibliometrix** package also enables users to analyze data from ISI Web of Knowledge (WoK) and PubMed. Web of Knowledge is one competitor to Scopus, but **bibliometrix** does not have an interface to the WoK API; therefore data must be manually exported from the site into `R`. Additional access to the web of Science API would be useful and has been implemented in a GitHub package **rwos** (`https://github.com/juba/rwos`), but is not on CRAN.

Moreover, other packages such as **scholar** and **gcite** can provide interfaces to the Google Scholar citation information. Using these in combination with **rscopus** can more likely guarantee complete information.

As compared to Google Scholar, WoK and Scopus information is **more curated** (is that true?).

As many pieces of academic promotion are related citation metrics or impact, these metrics can have real-life implications.

Here we will present a few examples of how to use the **rscopus** package, and present a simple descriptive analysis of citations and citation indices from people within the same field.

Scopus has a number of APIs available (`https://dev.elsevier.com/sc_apis.html`). We will distinguish between 2 types of APIs: search APIs and retrieval APIs. The retrieval APIs require unique indentifiers, such as an author ID or affiliation ID, to retrieve information. As these identifiers are not commonly known, we will use the search APIs to search on other criteria to obtain these identifiers. We will focus on gathering information about authors, affiliations, and citations.

API Key

Before using the package, one must obtain an access key to the API from `https://dev.elsevier.com/sc_apis.html` with the following steps:

1. Go to `https://dev.elsevier.com/user/login`. Login or create a free account.
2. Click “Create API Key”. Put in a label, such as `rscopus key`. Add a website. `http://example.com` is fine if you do not have a site.
3. **Read** and agree to the terms of service if you do indeed agree.

4. Add `Elsevier_API = "API KEY GOES HERE"` to `~/.Renviron` file, or add `export Elsevier_API=API KEY GOES HERE` to your `~/.bash_profile`.

Alternatively, you you can either set the API key using `rscopus::set_api_key` or by `options("elsevier_api_key" = api_key)`. You can access the API key using `rscopus::get_api_key`.

You should be able to test out the API key using the interactive Scopus APIs.

A note about API keys and IP addresses

The API Key is bound to a set of IP addresses, usually bound to your institution or organization (see `https://dev.elsevier.com/tecdoc_api_authentication.html`). Therefore, if you are using **rscopus** for a Shiny application, you must host the Shiny application from the institution/organization servers in some way. Also, you cannot access the Scopus API with this key if you are offsite and must VPN into the server or use a computing cluster with an institution IP.

Use cases

Processing author names to IDs

Researchers commonly would like to gather information about a set of authors. Most times the authors are the given by first and last names or initials; additional information such as affiliation may be available. Scopus provides unique identifier for authors (`au_id`) or affiliations (`affil_id`). In many cases with the API, you will specify the author identifier (`au_id`) instead of a first and last name, as there may be many authors with the same name. This identifier is unique to this author, though curation errors do happen and someone may have 2 unique identifiers. These identifiers can be merged by request on the Scopus website. In order to get the identifier from Scopus, you can search using a first and last name using the `process_author_name` command. For example, let us try to idnetify the author ID for John Muschelli:

```
library(rscopus)
auth_info = process_author_name(last_name = "Muschelli", first_name = "John")

       auth_name       au_id affil_id
1 John Muschelli 40462056100 60006183
                                       affil_name
1 Johns Hopkins Bloomberg School of Public Health

auth_info

$first_name
[1] "John"

$last_name
[1] "Muschelli"

$au_id
[1] "40462056100"
```

The output is a simple list of first and last name with an author ID. The function chooses the first author found, which may be useful if the author name is somewhat unique.

Retrieving author citation data

In order to get data about author papers and citations, the `author_data` function will retrieve this information:

```
jm = author_data(last_name = "Muschelli", first_name = "John")

       auth_name       au_id affil_id
1 John Muschelli 40462056100 60006183
                                       affil_name
1 Johns Hopkins Bloomberg School of Public Health
list(query = "AU-ID(40462056100)", count = 25, start = 0, view = "COMPLETE",
    facets = "subjarea(sort=fd,count=350)")
```

```
$query
[1] "AU-ID(40462056100)"

$count
[1] 25

$start
[1] 0

$view
[1] "COMPLETE"

$facets
[1] "subjarea(sort=fd,count=350)"

Response [https://api.elsevier.com/content/search/scopus?query=AU-ID%2840462056100%29&count=25&start=0&view=C
  Date: 2018-10-15 18:45
  Status: 200
  Content-Type: application/json;charset=UTF-8
  Size: 257 kB


  |
  |=================================================================| 100%
```

```
names(jm)
```

```
[1] "entries"   "df"        "full_data"
```

We see the output is a list of the converted entries from the Scopus API, a data.frame of the results for citations, and a list named full_data. The data.frame df has the information many users wish to retrieve, which is information about the author's documents and citations:

```
head(jm$df[, c("dc:identifier", "eid", "dc:title", "dc:creator",
"prism:publicationName", "prism:doi", "citedby-count",  "pii", "pubmed-id")])
```

```
          dc:identifier                eid
1 SCOPUS_ID:85053246791 2-s2.0-85053246791
2 SCOPUS_ID:85043338865 2-s2.0-85043338865
3 SCOPUS_ID:85047750078 2-s2.0-85047750078
4 SCOPUS_ID:85028874240 2-s2.0-85028874240
5 SCOPUS_ID:85050271095 2-s2.0-85050271095
6 SCOPUS_ID:85009266881 2-s2.0-85009266881

1                                                        Objective Evaluation of Multiple Sclerosis Lesion Se
2                                                                        MIMoSA: An Automated Method for Inte
3                       Radiomic subtyping improves disease stratification beyond key molecular, clinical, an
4                                    Feasibility of Coping Effectiveness Training for Caregivers of Children 
5                                                                                   Freesurfer: Connecting 
6 Thrombolytic removal of intraventricular haemorrhage in treatment of severe stroke: results of the randomis
          dc:creator         prism:publicationName
1       Commowick O.           Scientific Reports
2       Valcarcel A.       Journal of Neuroimaging
3   Kickingereder P.                Neuro-Oncology
4 Haakonsen Smith C. Journal of Genetic Counseling
5       Muschelli J.                  F1000Research
6          Hanley D.                    The Lancet
                      prism:doi citedby-count               pii pubmed-id
1     10.1038/s41598-018-31911-7            0              <NA>      <NA>
2              10.1111/jon.12506            0              <NA>      <NA>
3          10.1093/neuonc/nox188            3              <NA>      <NA>
4      10.1007/s10897-017-0144-1            2              <NA>      <NA>
5 10.12688/f1000research.14361.1            0              <NA>      <NA>
6  10.1016/S0140-6736(16)32410-2           46 S0140673616324102  28081952
```

We see that the full_data has this df inside it, with other data.frames:

```
names(jm$full_data)
```

```
[1] "df"            "affiliation"    "author"         "openaccessFlag"
```

These additional data.frames can have additional information about co-author affiliations or co-author information. This information may be useful for creating network graphs. For example, to get all authors from all the papers, you can use the author element from full_data:

```
head(jm$full_data$author)
```

```
  @_fa @seq                                                  author-url
1 true    1  https://api.elsevier.com/content/author/author_id/8431704700
2 true    2 https://api.elsevier.com/content/author/author_id/57203861434
3 true    3 https://api.elsevier.com/content/author/author_id/57199507814
4 true    4 https://api.elsevier.com/content/author/author_id/57197801981
5 true    5 https://api.elsevier.com/content/author/author_id/57203867656
6 true    6 https://api.elsevier.com/content/author/author_id/57203864793
       authid     authname   surname given-name initials afid.@_fa
1  8431704700 Commowick O. Commowick    Olivier       O.      true
2 57203861434    Istace A.    Istace     Audrey       A.      true
3 57199507814       Kain M.      Kain    Michaël       M.      true
4 57197801981   Laurent B.   Laurent   Baptiste       B.      true
5 57203867656     Leray F.     Leray    Florent       F.      true
6 57203864793     Simon M.     Simon    Mathieu       M.      true
    afid.$ entry_number
1 60030553            1
2 60001780            1
3 60030553            1
4 60105610            1
5 60030553            1
6 60030553            1
```

The column entry_number should merge with the data.frame of citations, as well as the information about author affiliations, which is located in the affiliation data.frame from full_data:

```
head(jm$full_data$affiliation)
```

```
  @_fa
1 true
2 true
3 true
4 true
5 true
6 true
                                                        affiliation-url
1 https://api.elsevier.com/content/affiliation/affiliation_id/60030553
2 https://api.elsevier.com/content/affiliation/affiliation_id/60001780
3 https://api.elsevier.com/content/affiliation/affiliation_id/60105610
4 https://api.elsevier.com/content/affiliation/affiliation_id/60062760
5 https://api.elsevier.com/content/affiliation/affiliation_id/60028893
6 https://api.elsevier.com/content/affiliation/affiliation_id/60028893
      afid
1 60030553
2 60001780
3 60105610
4 60062760
5 60028893
6 60028893
                                                             affilname
1                                                Universite de Rennes 1
2                                            Centre Hospitalier Lyon-Sud
3                      Laboratoire de Traitement de l'Information Medicale
4 Centre de Recherche en Acquisition et Traitement d'Images pour la Sante
5                               Centre Hospitalier Universitaire de Rennes
6                               Centre Hospitalier Universitaire de Rennes
  affiliation-city affiliation-country entry_number name-variant.@_fa
```

```
1           Rennes               France              1               <NA>
2             Lyon               France              1               <NA>
3            Brest               France              1               <NA>
4      Villeurbanne              France              1               <NA>
5           Rennes               France              1               <NA>
6           Rennes               France              1               <NA>
  name-variant.$
1           <NA>
2           <NA>
3           <NA>
4           <NA>
5           <NA>
6           <NA>
```

This information is rich for understanding information about an author's publication record, how many citations are recorded for a specific article, which journals have been published in, and who has co-authored publications with an author.

Calculating author indices

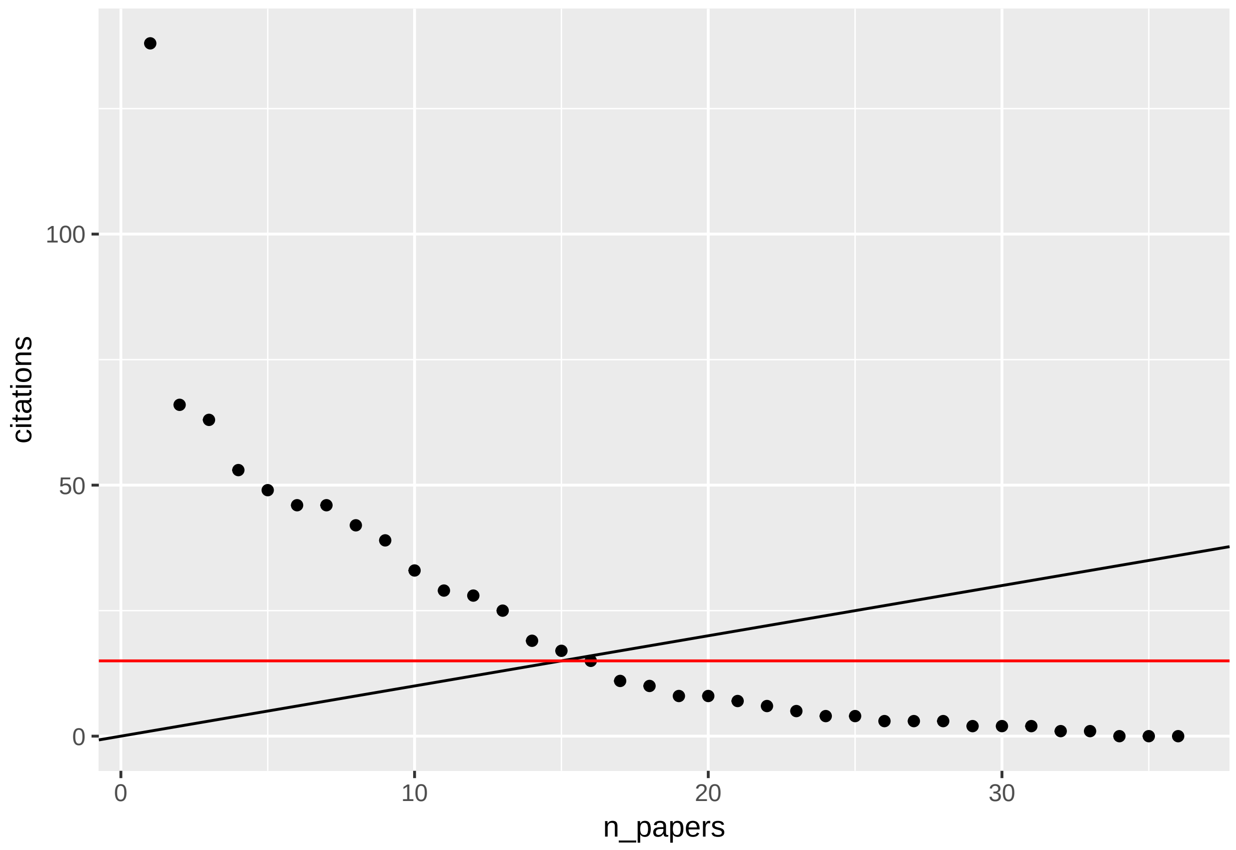
With the data from the `author_data` output, we can calculate the overall H-index (CITE) as follows:

```
library(dplyr)
h_data = jm$df %>%
  mutate(citations = as.numeric(`citedby-count`)) %>%
  arrange(-citations) %>%
  mutate(paper = 1,
         n_papers = cumsum(paper))
h_index = max(which(h_data$citations >= h_data$n_papers))
h_index
```

```
[1] 15
```

Using **ggplot2**, we can also visually show the H-index computation, where we plot the number of citations versus the number of papers (cumulatively) along with the X-Y line:

```
library(ggplot2)
h_data %>%
  ggplot(aes(x = n_papers, y = citations)) +
  geom_point() + geom_abline(slope = 1, intercept = 0) +
  geom_hline(yintercept = h_index, color = "red")
```



Additional indices can be created from the dat.

Retrieving information about an author

In `process_author_name`, we demonstrated how to get information form an author with a relatively unique name. If this is not the case, the `get_complete_author_info`, which powers `process_author_name`, can present more results. In order to retrieve author IDs from first and last names, the `get_complete_author_info` can be used. Here we search for authors with the last name West and first initial M:

```
last_name = "West"
first_name = "M"
auth_info_list = get_complete_author_info(
  last_name = last_name,
  first_name = first_name)
class(auth_info_list)
```

```
[1] "list"
```

```
names(auth_info_list)
```

```
[1] "get_statement" "content"
```

We see here, which is common in in some low-level functions returned from the API, the output is a list with elements `get_statement`, which returns an object of class `response` (from the **pttr** package), and `content`, which is the content from the response. Most times, the `content` is of interest, but failed requrests may be explored with the `get_statement` output.

In many `content` elements returned from the API, there are elements of the list named `entries` or `entry`. The low-level function `gen_entries_to_df` attempts to coerce this list into a standard `data.frame` for more usability, but may not perform perfectly as lists from JSON cannot always be directly coerced into a rectangular format. For example, here we will convert that output into a `data.frame`:

```
coerced = gen_entries_to_df(auth_info_list$content$`search-results`$entry)
names(coerced)
```

```
[1] "df"           "name-variant" "subject-area"
```

```
head(coerced$df)
```

```
  @_fa                                                      prism:url
1 true https://api.elsevier.com/content/author/author_id/35480328200
2 true https://api.elsevier.com/content/author/author_id/35419377800
3 true  https://api.elsevier.com/content/author/author_id/7003392768
4 true  https://api.elsevier.com/content/author/author_id/7402395730
5 true  https://api.elsevier.com/content/author/author_id/7402068812
6 true  https://api.elsevier.com/content/author/author_id/7401998578
          dc:identifier                eid                orcid
1 AUTHOR_ID:35480328200 9-s2.0-35480328200 0000-0002-0839-3449
2 AUTHOR_ID:35419377800 9-s2.0-35419377800                 <NA>
3  AUTHOR_ID:7003392768  9-s2.0-7003392768                 <NA>
4  AUTHOR_ID:7402395730  9-s2.0-7402395730                 <NA>
5  AUTHOR_ID:7402068812  9-s2.0-7402068812                 <NA>
6  AUTHOR_ID:7401998578  9-s2.0-7401998578                 <NA>
  preferred-name.surname preferred-name.given-name preferred-name.initials
1                   West                 Catharine                      C.
2                   West                Malcolm J.                    M.J.
3            Diener-West                     Marie                      M.
4                   West                 Robert M.                    R.M.
5                   West            Michael Abigail                   M.A.
6                   West                 David M.                     D.M.
  document-count
1            269
2            191
3            186
4            166
5            161
6            157
                                     affiliation-current.affiliation-url
1          https://api.elsevier.com/content/affiliation/affiliation_id/60003771
2          https://api.elsevier.com/content/affiliation/affiliation_id/60019870
3          https://api.elsevier.com/content/affiliation/affiliation_id/60006183
4          https://api.elsevier.com/content/affiliation/affiliation_id/60012070
5 https://api.elsevier.com/content/affiliation/affiliation_id/60012018 60023691
6          https://api.elsevier.com/content/affiliation/affiliation_id/60008221
  affiliation-current.affiliation-id
1                           60003771
2                           60019870
3                           60006183
4                           60012070
5                  60012018 60023691
6                           60008221
           affiliation-current.affiliation-name
1                      University of Manchester
2              James Cook University, Australia
3 Johns Hopkins Bloomberg School of Public Health
4                           University of Leeds
5        University of Pittsburgh Medical Center
6                              Massey University
  affiliation-current.affiliation-city
1                           Manchester
2                           Townsville
3                            Baltimore
4                                Leeds
5             Pittsburgh San Francisco
6                     Palmerston North
  affiliation-current.affiliation-country entry_number
1                          United Kingdom            1
2                               Australia            2
3                           United States            3
4                          United Kingdom            4
```

```
5             United States United States            5
6                             New Zealand            6
```

We see this has information about the multiple authors returned, along with names, variations on those names, number of documents, and affiliations. We can then extract the author ID we want from this data.frame. This process is wrapped in the get_author_info:

```
auth_info_df = get_author_info(last_name = last_name,
                              first_name = first_name)
head(auth_info_df)
```

```
             auth_name       au_id          affil_id
1       Catharine West 35480328200          60003771
2      Malcolm J. West 35419377800          60019870
3    Marie Diener-West  7003392768          60006183
4       Robert M. West  7402395730          60012070
5 Michael Abigail West  7402068812 60012018 60023691
6        David M. West  7401998578          60008221
                                     affil_name
1                       University of Manchester
2               James Cook University, Australia
3 Johns Hopkins Bloomberg School of Public Health
4                             University of Leeds
5         University of Pittsburgh Medical Center
6                               Massey University
```

but we should note this information is condensed and a subset that is available from get_complete_author_info.

If we now have an affiliation ID, such as 60006183 for the Johns Hopkins Bloomberg School of Public Health, we can pass this to get_author_info or process_author_name:

```
spec_affil = get_author_info(
  last_name = last_name,
  first_name = first_name,
  affil_id = 60006183)
spec_affil
```

```
          auth_name      au_id affil_id
1 Marie Diener-West 7003392768 60006183
                                     affil_name
1 Johns Hopkins Bloomberg School of Public Health
```

Retrieving summary information about an author

The author_retrieval function can gather summary information about an author using the author identifier or name.

```
author_info = author_retrieval(last_name = "Muschelli", first_name = "J")
```

```
       auth_name       au_id affil_id
1 John Muschelli 40462056100 60006183
                                     affil_name
1 Johns Hopkins Bloomberg School of Public Health
```

```
names(author_info$content)
```

```
[1] "author-retrieval-response"
```

```
class(author_info$content$`author-retrieval-response`)
```

```
[1] "list"
```

We can use gen_entries_to_df to convert this response into a data.frame

```
gen_entries_to_df(author_info$content$`author-retrieval-response`)$df
```

```
  @status @_fa
1   found true
                                        coredata.prism:url
1 http://api.elsevier.com/content/author/author_id/40462056100
  coredata.dc:identifier       coredata.eid      coredata.orcid
1  AUTHOR_ID:40462056100 9-s2.0-40462056100 0000-0001-6469-1750
  coredata.document-count coredata.cited-by-count coredata.citation-count
1                      36                     671                     788
                                                                       coredata.link.@href
1 https://www.scopus.com/authid/detail.uri?partnerID=HzOxMe3b&authorId=40462056100&origin=inward
  coredata.link.@rel coredata.link.@_fa preferred-name.surname
1     scopus-author               true              Muschelli
  preferred-name.given-name preferred-name.initials
1                      John                      J.
            affiliation-current.affiliation-name
1 Johns Hopkins Bloomberg School of Public Health
  affiliation-current.affiliation-city
1                            Baltimore
  affiliation-current.affiliation-country publication-range.start
1                           United States                    2011
  publication-range.end entry_number
1                  2018            1
```

but this list typically only has one element, and may be easily referenced using $ as a list.

Retrieving information about multiple authors

In order to get information from multiple authors, one could loop over author information, but this is inefficient for code and API calls. The complete_multi_author_info function can perform this operation. One caveat is that it requires author identifiers and not names. We can take the author IDs from auth_info_df to retrieve information for all these authors:

```
all_author_info = complete_multi_author_info(au_id = auth_info_df$au_id)
names(all_author_info)
```

```
[1] "get_statement" "content"       "au_id"
```

This result is again a low-level output from the API. We can use the process_complete_multi_author_info function to process this into a more amenable solution:

```
processed = process_complete_multi_author_info(all_author_info)
head(names(processed))
```

```
[1] "35480328200" "35419377800" "7003392768"  "7402395730"  "7402068812"
[6] "7401998578"
```

Now, each element is the author ID, which contains a list of data.frames. The multi_author_info will perform both of these operations together. This result is still not “tidy” in many respects, but parts can be combined using dplyr or purrr:

```
journals = purrr:::map_df(processed, function(x) {
  x$journals
  }, .id = "au_id")
head(journals)
```

```
        au_id type                                    sourcetitle
1 35480328200    j                                 Cancer Letters
2 35480328200    j   Journal of Cancer Research and Clinical Oncology
3 35480328200    j                           Nature Reviews Cancer
4 35480328200    j Annals of the Royal College of Surgeons of England
5 35480328200    j                                 Cancer Letters
6 35480328200    j                     PLoS Computational Biology
           sourcetitle-abbrev     issn
1               Cancer Lett. 03043835
2 J. Cancer Res. Clin. Oncol. 01715216
3           Nat. Rev. Cancer 1474175X
```

```
4   Ann. R. Coll. Surg. Engl. 00358843
5               Cancer Lett. 18727980
6         PLoS Comput. Biol. 1553734X
```

Citations over time

Some APIs from Elsevier are disabled by default (see https://dev.elsevier.com/api_key_settings.html). Notably, the Citations Overview API is disabled, which allows users to access information about citations over time for articles of authors. This information is particularly useful for creating bibliometric indices, such as the *h*-index (CITE). The `rscopus` package interfaces with these APIs, but the API must be enabled for that specific API key. On the Scopus website one can searching for authors, select up to 15 authors, and then create a "Citation Overview", which will give this citation information, which is in a CSV format. The `rscopus` package provides a `read_cto` function to read in this data.

We also provide an example export from a single author:

```
file = system.file("extdata", "CTOExport.csv", package = "rscopus")
citations_over_time = rscopus::read_cto(file)
names(citations_over_time)
```

```
[1] "data"               "year_columns"       "author_information"
```

The real information is in the `data` element of this list.
Here we present `short_title`, first 3 (relevant) words of the title, instead of the full document title for viewing purposes as titles can be quite long.

```
yr_cols = citations_over_time$year_columns
citations_over_time = citations_over_time$data
citations_over_time = citations_over_time %>%
  mutate(short_title = unique_title(`Document Title`))
head(citations_over_time[, c("short_title", yr_cols[1:5])])
```

```
                           short_title <2008 2008 2009 2010 2011
1         Objective Evaluation Multiple     0    0    0    0    0
2              MIMoSA: Automated Method     0    0    0    0    0
3           Radiomic subtyping improves     0    0    0    0    0
4      Feasibility Coping Effectiveness     0    0    0    0    0
5     Freesurfer: Connecting Freesurfer     0    0    0    0    0
6 Thrombolytic removal intraventricular     0    0    0    0    0
```

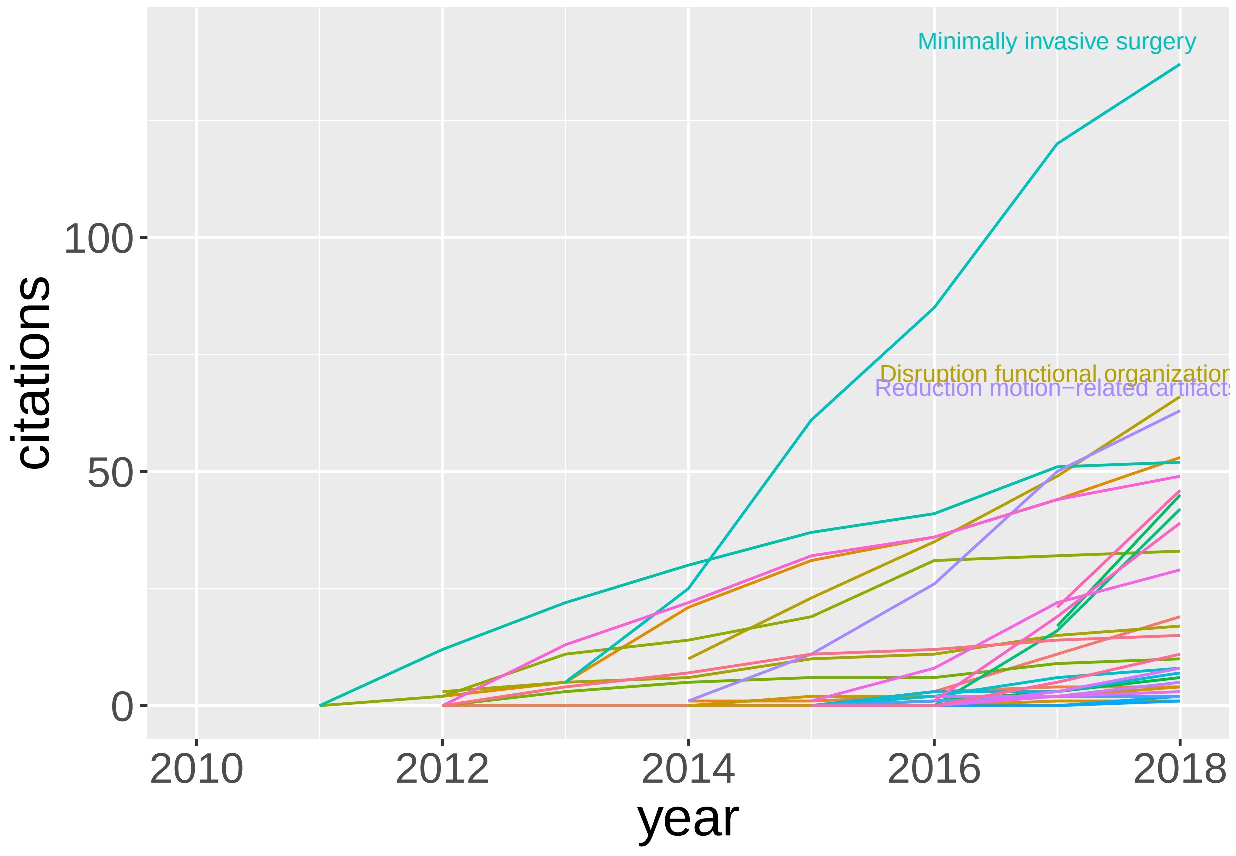
In the citation overview, you must specify a range of years on Scopus, with a maximum of 15 years. As many times this wide format is not what you want to plot, a helper function `read_cto_long` will read the data in long format, done by **tidyr** (CTIE). Here we use **dplyr** to arrange the data by maximum number of citations:

```
library(dplyr)
long_cite = rscopus::read_cto_long(file)
long_cite = long_cite$data %>%
  group_by(`Document Title`, year) %>%
  summarize(citations = sum(citations),
          `Publication Year` = unique(`Publication Year`)) %>%
  mutate(short_title = unique_title(`Document Title`))
long_cite = long_cite %>% arrange(-citations, year, short_title)
head(long_cite[, c("short_title", "year", "citations")])
```

```
# A tibble: 6 x 3
  short_title                           year  citations
  <chr>                                 <fct>     <int>
1 Minimally invasive surgery            2015         36
2 Minimally invasive surgery            2017         35
3 ISLES 2015 public                     2018         28
4 Large-scale radiomic profiling        2018         26
5 Thrombolytic removal intraventricular 2018         25
6 Minimally invasive surgery            2016         24
```

Thus, we have one record per year and article. Here we will plot the cumulative citations per each paper over the years of publication and label the top 3 cited papers:

```
# get cumulative sum
csum = long_cite %>%
  mutate(citations = ifelse(is.na(citations), 0, citations)) %>%
  arrange(`Document Title`, year) %>%
  group_by(`Document Title`) %>%
  mutate(citations = cumsum(citations))
# remove past and future with as.integer
csum = csum %>%
  mutate(year = as.integer(as.character(year))) %>%
  filter(!is.na(year)) %>%
  filter(year >= `Publication Year`)
# grab last citations and top 3 papers
last_year = csum %>%
  arrange(`Document Title`, year) %>%
  group_by(`Document Title`) %>%
  slice(n()) %>%
  ungroup %>% arrange(-citations) %>%
  head(3)
g = ggplot(csum,
         aes(x = year, y = citations, color = short_title  )) +
  xlim(c(2010, 2018)) + geom_line() +
  # label the titles numbers for top 3
  geom_text(data = last_year, size = 3, aes(label = short_title),
          nudge_x = -1, nudge_y = 5)
# don't want label for document title
g + guides(color = FALSE) + theme(text = element_text(size = 20))
```



Retrieving Affiliation Information

In order to get information about an affiliation, the `get_affiliation_info` can be used. Here we will look for the pattern `Johns Hopkins`:

```
jhu_info = get_affiliation_info(affil_name = "Johns Hopkins")
head(jhu_info[, c("affil_id", "affil_name")])
```

```
  affil_id                                              affil_name
1 60005248                                Johns Hopkins University
2 60001117                      The Johns Hopkins School of Medicine
3 60006183         Johns Hopkins Bloomberg School of Public Health
4 60001555                                  Johns Hopkins Hospital
5 60003443                        Johns Hopkins Medical Institutions
6 60022054 The Johns Hopkins University Applied Physics Laboratory
```

This function implicitly calls `affil_search`, which searches the affiliation information from Scopus. Additional information can be extracted using `affil_search`, but this typically includes a large number of records as it searches all the documents. This affiliation ID can be used to be more specific when searching authors or documents.

```
eid_info = rscopus::article_retrieval(id = jm$df$eid[1],
                                      identifier = "eid")
scopus_id = jm$df$`dc:identifier`[1]
scopus_id = sub("SCOPUS_ID:", "", scopus_id)
sc_info = rscopus::article_retrieval(id = scopus_id, identifier = "scopus_id")
doi_info = rscopus::article_retrieval(id = jm$df$`prism:doi`[1], identifier = "doi")
em_ret = embase_retrieval(id = jm$df$`prism:doi`[1], identifier = "doi")
em_content = em_ret$content$`abstracts-retrieval-response`
df = gen_entries_to_df(em_content)
res = generic_elsevier_api()
```

In some cases, one may have an article in mind and would like information about the authors of that paper. In order to get the author IDs from the paper identifier, one can use the `abstract_retrieval` function:

```
sc_id = sub("SCOPUS_ID:", "", jm$df$`dc:identifier`[1])
res = abstract_retrieval(id = sc_id, identifier = "scopus_id")
sc_info = res$content$`abstracts-retrieval-response`
sc_df = purrr::map_df(
  sc_info$authors[[1]],
  as.data.frame,
  stringsAsFactors = FALSE,
  make.names = FALSE)
head(sc_df[, c("ce.given.name", "ce.initials", "X.auid")])
```

```
  ce.given.name ce.initials      X.auid
1       Olivier          O.  8431704700
2        Audrey          A. 57203861434
3       Michaël          M. 57199507814
4      Baptiste          B. 57197801981
5       Florent          F. 57203867656
6       Mathieu          M. 57203864793
```

This information is located within the `author` `data.frame` from the `full_data` as well. As we took the first entry from the Scopus identifier, we will subset the author data by `entry_number` 1 from the `author` `data.frame`:

```
paper_author_info = jm$full_data$author
head(paper_author_info[paper_author_info$entry_number == 1,])
```

```
  @_fa @seq                                                  author-url
1 true    1  https://api.elsevier.com/content/author/author_id/8431704700
2 true    2 https://api.elsevier.com/content/author/author_id/57203861434
3 true    3 https://api.elsevier.com/content/author/author_id/57199507814
4 true    4 https://api.elsevier.com/content/author/author_id/57197801981
```

```
5 true    5 https://api.elsevier.com/content/author/author_id/57203867656
6 true    6 https://api.elsevier.com/content/author/author_id/57203864793
       authid     authname   surname given-name initials afid.@_fa
1  8431704700 Commowick O. Commowick    Olivier       O.      true
2 57203861434    Istace A.    Istace     Audrey       A.      true
3 57199507814      Kain M.      Kain    Michaël       M.      true
4 57197801981   Laurent B.   Laurent   Baptiste       B.      true
5 57203867656     Leray F.     Leray    Florent       F.      true
6 57203864793     Simon M.     Simon    Mathieu       M.      true
    afid.$ entry_number
1 60030553            1
2 60001780            1
3 60030553            1
4 60105610            1
5 60030553            1
6 60030553            1
```

Thus, if we retrieve a single author's information, we can gather other author IDs from this directly. If we have a specific paper, we can retrieve author IDs from that paper information as well.

John Muschelli
Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health
615 N Wolfe St Baltimore, MD 21205
jmuschel@jhsph.edu