PharmaSUG 2023 - Paper SD-334

Making Publication Metric Tracking Easy: Using a Reproducible, Integrated System of R and Microsoft Power BI® to Ease the Pain of Assessing Publication Metrics

Joshua J. Cook, Jessica Truett
Andrews Research & Education Foundation (AREF)

ABSTRACT

Publication metrics are increasingly being used to measure individual- and organization-level productivity and impact within academia and industry. Additionally, these metrics, including publication counts over time, are being used to make business decision. Historically, publication metrics have not been the easiest thing to quantify and manage. This is primarily due to the manual nature of querying PubMed from the web, which is difficult to do for multiple highly published authors with non-unique names. Instead of manually obtaining this data, it is much more feasible to leverage the R programming language and various reporting systems to manage this data. Importantly, many of these tools are free! This text demonstrates how R, a few R packages, and reporting systems such as Quarto® and Microsoft Power BI® can be integrated into a reproducible system capable of streamlining the collection, wrangling, and reporting process of publication data for stakeholders.

INTRODUCTION

Publications in peer-reviewed journals serve as a primary source of knowledge for many different areas of study, including data science and medicine. In addition to serving as a resource, publications are also quantified in the form of publication data metrics. These metrics include publication and citation count, affiliation spread, and journal impact factor. A combination of these metrics is often equated to productivity and impact of the published author or affiliated institution. As a result, publication metrics have been used to make business decisions regarding promotions, tenure, awards, benchmarking, grant funding, and even clinical study sponsorship.

Despite their widespread use, publication metrics are not the easiest thing to quantify and manage. The simplest way to obtain information regarding these metrics is to use a database search engine such as that of PubMed. However, manually searching PubMed[1] quickly becomes an issue when authors have multiple aliases and affiliations, or common names. This problem is further exacerbated when an institution is comparing metrics across many different authors on a repetitive schedule, requiring quick refreshing of data and associated reports.

By leveraging the tools of a data scientist, primarily the R programming language (4.2.3)[2] and Microsoft Power BI®[3], it is possible and relatively simple to make an integrated system that can streamline this arduous process for any organization. Customizable queries will be written and executed in R to extract publication metric data from a PubMed interface using the easyPubMed package[4]. easyPubMed interfaces with the National Center for Biotechnology Information (NCBI) Entrez Programming Utilities (E-utilities), which allows for easy programmatic access to PubMed[1]. The data frame within R will be prepared for reporting using the XML (for reading and writing XML files) and tidyverse (for data wrangling) packages[5,6]. The resulting final data frames will be connected to Quarto®[7] and Microsoft Power BI®[3] for reporting. Quarto® is an open-source scientific and technical publishing system that is often described as "the next generation of R Markdown."[7] Quarto® is built into RStudio® and offers a simple solution to reporting by being able to execute R code using Knitr which is thus able to create data-rich and visually appealing reports[7,8]. Microsoft Power BI® is a proprietary business intelligence application developed by Microsoft[3].

GETTING STARTED

SETUP

This demonstration utilized RStudio® as the primary Integrated Development Environment (IDE). As with any project in RStudio®, a primary folder should be setup that will be used as the working directory. This is where the R project (Rproj) file will be housed, as well as any required subfolders. These examples use the following:

```
C:\publications\
```

Create subordinate folders for resources that will be required in reports, such as data and images:

```
C:\publications\data\
C:\publications\images\
```

Within RStudio®, only three libraries are installed and imported-`tidyverse`, which is used for data wrangling, `easyPubMed`, which is an R interface to the Entrez Programming Utilities, and `XML,` which is used to read and create XML documents:

```
install.packages("tidyverse")

install.packages("easyPubMed")

install.packages(“XML”)

library(tidyverse)

library(easyPubMed)

library(XML)
```

The R script can then be saved to the primary folder with any filename.

UNDERSTANDING THE QUERY

Queries are the foundation of PubMed searching, and thus are also the foundation of the easyPubMed package. Before jumping headfirst into writing complex queries, it is important to understand that easyPubMed functions via a two-step process. First, queries must be entered using into a string (single or double quotes) with any combination of PubMed field tags, a full listing of which is available on PubMed (Help - PubMed (nih.gov))

Searching by author (AU) is utilized in this example. Field tags may be combined using “AND” or “OR” syntax:

```
“Adam W Anz[AU]”

“Adam W Anz[AU] OR Adam Anz[AU]”
```

Importantly, individual search terms are each listed with their field tags in brackets and are all inside the same query string. Related statements, such as the inclusion of two different options for journal names, (TA) are surrounded by parenthesis:

```
   “Adam W Anz[AU] AND (American Journal of Sports Medicine[TA] OR
Arthroscopy[TA])”
```

If the query contains a special character, such as an apostrophe, these characters should be commented out using standard R syntax:

```
“Christopher O\'Grady[AU]”
```

This string is correctly read by R as “Christopher O’Grady.”

For longer queries, such as with full titles (TI), it may be better to utilize separate queries and field tags:

```
   “Bone Marrow Aspirate Concentrate Is Equivalent to Platelet-Rich Plasma
for the Treatment of Knee Osteoarthritis at 1 Year: A Prospective, Randomized
Trial”

   “[TI]”
```

Once a query has been finalized, it should be saved as an R object with a concise name so it can be used in many of the easyPubMed functions:

```
   AnzQuery <- "Adam W Anz[AU]"

   AnzJournalQuery <- "Adam W Anz[AU] AND (American Journal of Sports
Medicine[TA] OR Arthroscopy[TA])"
```

For the longer queries, two objects should be saved to define both the query and the field tag:

```
   ArticleTitle <- "Bone Marrow Aspirate Concentrate Is Equivalent to
Platelet-Rich Plasma for the Treatment of Knee Osteoarthritis at 1 Year: A
Prospective, Randomized Trial"

   TitleField <- "[TI]"
```

Building queries is just the first step to utilizing easyPubMed. The second step involves using user-defined queries in easyPubMed functions to retrieve data from the PubMed database.

BASIC DATA RETRIEVAL

The simplest use of a query is to retrieve the PubMed IDs of any entries matching the query criteria using the `fetch_pubmed_data()` function:

```
   AnzQuery <- "Adam W Anz[AU]"

   AnzIDs <- get_pubmed_ids(AnzQuery)
```

The results of the query (PubMed IDs) are posted on the Entrez History Server and can be used to fetch article information using the `fetch_pubmed_data()` function. The `format` argument specifies the formats supported by Entrez and easyPubMed. The abstract option will download data from PubMed in the same format as the abstract page, including the publication citation, title, author, author information, etc:

```
   Anz_abstracts <- fetch_pubmed_data(

      pubmed_id_list = AnzIDs,

      format="abstract"

      )

      print(Anz_abstracts[1:16])
```

The `print` function writes the first 16 lines of the abstract data stored in `Anz_abstracts`, which includes information on his most recent entry in Entrez (i.e., his most recent publication at the time of this paper) as shown in Output 1.

```
[1] "1. Arthroscopy. 2023 Mar;39(3):728-729. doi:
10.1016/j.arthro.2022.11.030."
[2] ""
[3] "Editorial Commentary: Elbow Injury Results When Pediatric and
Adolescent "
[4] "Throwing Athletes Throw as Hard as Possible, and Weighted Baseball
Training "
[5] "Should Be Banned for Youth Athletes."
[6] ""
[7] "Anz AW(1)."
[8] ""
[9] "Author information:"
[10] "(1)Andrews Research & Education Foundation and Andrews Institute for
"
[11] "Orthopaedics & Sports Medicine."
[12] ""
```

```
[13] "Comment on"
[14] "    Arthroscopy. 2023 Mar;39(3):719-727."
[15] ""
[16] "We are in the middle of an epidemic involving pediatric and
adolescent throwing "
```

Output 1. Print Output Showing the First 16 Lines of the Abstract Data

The `format` argument may also be set to “xml,” which makes it easier to extract specific xml-tagged information, such as article titles (in this case, the most recent 16 article titles):

```
 Anz_xml <- fetch_pubmed_data(
      pubmed_id_list = AnzIDs,
      format="xml"
      )
   Anz_titles <- custom_grep(
      Anz_xml,
      "ArticleTitle",
"char"
)
   print(Anz_titles[1:16])
```

The `print` function writes the first 16 lines of the xml-tagged article title data stored in `Anz_xml`, which includes information on the most recent 16 entries in Entrez (i.e., the most recent 16 publications at the time of this paper) as shown in Output 2.

```
[1] "Editorial Commentary: Elbow Injury Results When Pediatric and
Adolescent Throwing Athletes Throw as Hard as Possible, and Weighted
Baseball Training Should Be Banned for Youth Athletes."
[2] "Blood Flow Restriction Using a Pneumatic Tourniquet Is Not Associated
With a Cellular Systemic Response."
[3] "Bone Marrow Aspirate Concentrate Is Equivalent to Platelet-Rich Plasma
for the Treatment of Knee Osteoarthritis at 2 Years: A Prospective
Randomized Trial."
[4] "The safety and efficacy of 2 anterior-inferior portals for
arthroscopic repair of anterior humeral avulsion of the glenohumeral
ligament: cadaveric comparison."
[5] "Association Between Passive Hip Range of Motion and Pitching
Kinematics in High School Baseball Pitchers."
[6] "Platelet-Rich Plasma: Fundamentals and Clinical Applications."
[7] "Arthroscopic Subchondral Drilling Followed by Injection of Peripheral
Blood Stem Cells and Hyaluronic Acid Showed Improved Outcome Compared to
Hyaluronic Acid and Physiotherapy for Massive Knee Chondral Defects: A
Randomized Controlled Trial."
[8] "Biologic Association Annual Summit: 2020 Report."
[9] "Elevation of Peripheral Blood CD34+ and Platelet Levels After Exercise
With Cooling and Compression."
[10] "Mobilized Peripheral Blood Stem Cells are Pluripotent and Can Be
Safely Harvested and Stored for Cartilage&#xa0;Repair."
[11] "Blood Flow Restriction Training Using the Delfi System Is Associated
With a Cellular Systemic Response."
[12] "Chondral Lesions of the Knee: An Evidence-Based Approach."
[13] "The Effects of Body Mass Index on Softball Pitchers' Hip and Shoulder
Range of Motion."
```

```
[14] "Lower Extremity Pain and Pitching Kinematics and Kinetics in
Collegiate Softball Pitchers."
[15] "Bone Marrow Aspirate Concentrate Is Equivalent to PRP for the
Treatment of Knee OA at 1 Year: Response."
[16] "Autologous thrombin preparations: Biocompatibility and growth factor
release."
```

Output 2. Print Output Showing the Most Recent 16 Title Entries in Entrez

The `fetch_pubmed_data()` function may be useful for glancing at data, but the `batch_pubmed_download()`function makes it possible to download all records returned by a query to the local machine as TXT or XML files (in the case below as XML files). Importantly, the primary folder is the save destination unless specified:

```
Anz_download <- batch_pubmed_download(
   pubmed_query_string = AnzQuery,
   format = "xml",
   batch_size = 1000,
   encoding = "UTF8"
   )
```

The `articles_to_list()` function takes the downloaded XML files and converts them into a list of strings that correspond to individual PubMed articles using a "/PubmedArticle" XML tag:

```
Anz_list <- articles_to_list(pubmed_data = Anz_download)
```

Utilizing `lapply`, the `article_to_df()` function can be applied over the list of strings to extract publication-specific information from each record and store it as a list of data frames where each row corresponds to a dataframe for each of the authors of the article. The `autofill` argument imputes missing affiliations for the multiple row entries corresponding to each author of an article. The `getAuthors` argument specifies whether or not to extract all the author names and should be utilized in cases where there are an abnormally large amounts of authors associated with a publication (for example, publications coming from CERN's Large Hadron Collider or international clinical trials):

```
Anz_df_list <- lapply(Anz_list, article_to_df, autofill = TRUE)
```

Finally, the list of dataframes can be combined into one full dataframe that lists repeating entries of publication information as rows that represent each individual author:

```
Anz_full_list <- do.call(rbind, Anz_df_list)
```

The resulting, full dataframe can be filtered, sorted, and reported using standard R and tidyverse syntax. This full data frame will include data for the variables listed in Table 1.

Variable	**Meaning**
pmid	The unique PubMed identifier assigned to the article.
doi	The unique Digital Object Identifier assigned to the article.
title	The full title of the article.
abstract	The full abstract of the article.
year	The article publication year (electronic).
month	The article publication month (electronic).
day	The article publication day (electronic).

jabbrv	The abbreviated name of the publishing journal.
journal	The full name of the publishing journal.
keywords	Keywords associated with the article.
lastname	The author's last name.
firstname	The author's first name.
address	The address affiliated with the author.
email	The email affiliated with the author.

Table 1. All Variables Included in Dataframes Generated with easyPubMed

COMPLEX QUERY EXECUTION AND DATA WRANGLING

There are several obstacles in publication data that need to be considered before complex queries are executed. First, publishing aliases are often inconsistent throughout journal formats and throughout time-related database changes. Second, names are rarely unique and often there are multiple people with the same name publishing in the literature. Finally, the methods mentioned thus far produce a list with entries for each author, which may not be ideal for tracking publication metrics. Each of these obstacles are addressed below.

ALIASES AND NAMES OF AUTHORS

To address the alias problem, queries should include every possible format of the author's alias. Importantly, PubMed allows for the listing of first, middle, and last names in several formats. However, the generated dataframe will store the middle initial in the "firstname" column. Additional post-query cleaning with `tidyverse` using the specified aliases is utilized to ensure the quality of the final dataframe:

```
AnzQuery <- 'Adam Anz[AU] OR Adam W Anz[AU] OR AW Anz[AU]'
Anz_download <- batch_pubmed_download(
   pubmed_query_string = AnzQuery,
   format = "xml",
   batch_size = 1000,
   encoding = "UTF8")
Anz_list <- articles_to_list(pubmed_data = Anz_download)
Anz_df_list <- lapply(Anz_list, article_to_df, autofill = TRUE)
Anz_full_list <- do.call(rbind, Anz_df_list)
```

To deal with non-unique names, a differentiating variable must be chosen. In many cases, the instances of unique journal names is the simplest differentiating variable which can be counted and printed:

```
n_distinct(Anz_full_list$journal)
journals <- distinct(Anz_full_list, journal)
print(journals)
```

In this case, only 22 unique journal names are present. Information about the author is needed to determine which journals are applicable to their field and are thus likely true publications from that author. Dr. Adam Anz is a board-certified orthopaedic sports medicine surgery specialist in Gulf Breeze, FL. Therefore, only entries connected to journals relating to orthopaedic sports medicine surgery are retained in the dataframe (which are all entries in this case, requiring no further filtering).

If a list of only publications with Dr. Anz confirmed as the author is needed (i.e., dropping all co-authors), the list can be wrangled with `tidyverse`,

```
Anz_clean_list <- dplyr::filter(
      Anz_full_list,
      lastname == "Anz"
      )
Anz_clean_list <- dplyr::filter(
      Anz_clean_list,
      firstname == "Adam"
      | firstname == "Adam W"
      )
```

INCLUDING MULTIPLE AUTHORS

Once a publication dataframe has been wrangled to fit the description of a single author, the process can be quickly repeated for additional authors (which is a primary use-case for large organizations with subgroups of specialties or research foci):

```
   JordanQuery <- 'Steve Jordan[AU] OR Steven Jordan[AU] OR SE Jordan[AU] OR
Steve E Jordan[AU] OR Steven E Jordan[AU]'
   Jordan_download <- batch_pubmed_download(
      pubmed_query_string = JordanQuery,
      format = "xml",
      batch_size = 1000,
      encoding = "UTF8")
   Jordan_list <- articles_to_list(pubmed_data = Jordan_download)
   Jordan_df_list <- lapply(Jordan_list, article_to_df, autofill = TRUE)
   Jordan_full_list <- do.call(rbind, Jordan_df_list)
   Jordan_clean_list <- dplyr::filter(
      Jordan,
      lastname == "Jordan"
      )
   Jordan_clean_list <- dplyr::filter(
  Jordan_clean_list,
      firstname == "Steve"
      | firstname == "Steven"
      | firstname == "Steve E"
      | firstname == "Steven E"
      | firstname == "SE"
      )
```

In the case of this author, the simplest differentiating variable was “address.” Thus, the dataframe was filtered to only include addresses relevant to Dr. Steve Jordan, who is also a board-certified orthopaedic sports medicine surgery specialist in Gulf Breeze, FL. The original full list `Jordan_full_list` was filtered using the new `Jordan_clean_list` via the `semi_join()` function to ensure that only publications truly from Dr. Jordan are included in both final dataframes:

```
   Jordan_clean_list <- dplyr::filter(
      Jordan_clean_list,
      str_detect(Jordan_clean_list $address,
"Andrews|Orthopedic|Orthopaedic")
      )
Jordan_full_list <- semi_join(
      Jordan_full_list,
      Jordan_clean_list,
      by="title",
      copy=FALSE)
```

AGGREGATING THE DATA

Once queries are written for all authors, dataframes can be aggregated using `tidyverse`.

```
   All_full_list <- Dplyr::bind_rows(Anz_full_list, Jordan_full_list)
   All_clean_list <- Dplyr::bind_rows(Anz_clean_list, Jordan_clean_list)
```

WRANGLING THE DATA

Prior to reporting data, NA values should be replaced with the “Missing” string:

```
   All_full_list <- mutate(All_full_list, across(where(is.numeric),
~replace_na(.x, "Missing")))
   All_full_list <- mutate(All_full_list, across(where(is.character),
~replace_na(.x, "Missing")))
   All_clean_list <- mutate(All_clean_list, across(where(is.numeric),
~replace_na(.x, "Missing")))
   All_clean_list <- mutate(All_clean_list, across(where(is.character),
~replace_na(.x, "Missing")))
```

Additional data wrangling is accomplished with `tidyverse.`

ASSESSING PUBLICATION METRICS

The Andrews Institute (AI) and Andrews Research & Education Foundation (AREF) currently have 34 actively publishing orthopaedic clinicians that include surgeons, physical therapists, and primary care sports medicine specialists. For demonstration purposes, the remaining 32 clinicians not mentioned previously have been added to both `All_full_list` and `all_clean_list`.

At this point, queries have been written for 34 clinicians and have subsequently been used to extract publication metric data from the Entrez database that is now stored in two final dataframes. The final dataframes have been wrangled and are ready for reporting to stakeholders. There are many ways to approach reporting, but this demonstration will focus on utilizing Quarto® and Microsoft Power BI®.

QUICK AND REPRODUCIBLE REPORTS WITH QUARTO®

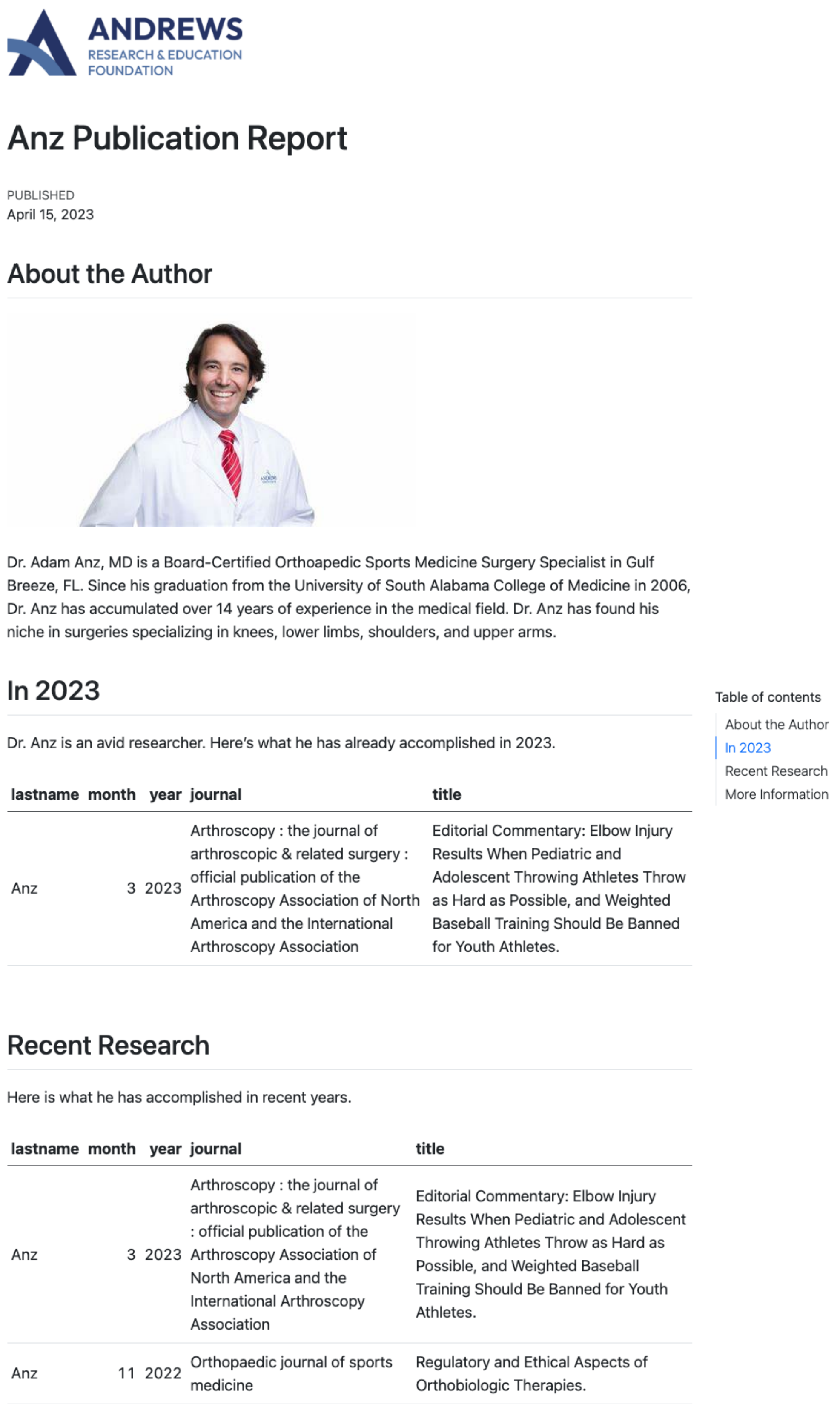
A full tutorial on Quarto® is outside the scope of this paper, but the following steps can be taken to generate a simple report. Importantly, these reports can be reproduced as the underlying data changes by simply re-rendering within RStudio®:

1. In RStudio®, create a new Quarto® Document with a title, output format (HTML, PDF, or Word), and optionally, the name of the author.

2. Edit the YAML header to fit the needs of the report. In this case, the report will include the organization logo in the title, be output in HTML with all resources embedded, and include a floating table of contents:

   ```
   ---
   title: |
    ![](images/0.%20AndrewsREF_Horizontal05-01.png){fig-align="left"
   width="273"}  \
     \
     Anz Publication Report
   date: "today"
   format: html
   embed-resources: true
   documentclass: report
   editor: visual
   toc: TRUE
   number-sections: FALSE
   colorlinks: TRUE
   ---
   ```

3. Create an R code block for setup and data wrangling. The output of this block should not be included in the report. Importantly, actively querying the Entrez database every time a report is generated will significantly slow down the rendering process. Instead, it is recommended to aggregate and wrangle data in a separate script, and then save the data in-memory as an RData file that can be loaded at the time of report generation without repeatedly querying Entrez (saved here as "publications.RData").
4. Create additional R code blocks to display tables and figures that tell the story of the data (using packages such as `gt` and `ggplot2`).
5. Add context using external text, images, and links.
6. Render the document, which will generate the complete HTML file within the working directory. The final HTML file may be distributed to stakeholders so long as the `embed-resources` argument is set to `TRUE` in the YAML header. If false, the HTML file cannot be loaded outside the development environment. An example of a generated report for Dr. Anz is included in Figure 1.



ANDREWS
RESEARCH & EDUCATION
FOUNDATION

Anz Publication Report

PUBLISHED
April 15, 2023

About the Author

Dr. Adam Anz, MD is a Board-Certified Orthoapedic Sports Medicine Surgery Specialist in Gulf Breeze, FL. Since his graduation from the University of South Alabama College of Medicine in 2006, Dr. Anz has accumulated over 14 years of experience in the medical field. Dr. Anz has found his niche in surgeries specializing in knees, lower limbs, shoulders, and upper arms.

In 2023

Dr. Anz is an avid researcher. Here's what he has already accomplished in 2023.

lastname	month	year	journal	title
Anz	3	2023	Arthroscopy : the journal of arthroscopic & related surgery : official publication of the Arthroscopy Association of North America and the International Arthroscopy Association	Editorial Commentary: Elbow Injury Results When Pediatric and Adolescent Throwing Athletes Throw as Hard as Possible, and Weighted Baseball Training Should Be Banned for Youth Athletes.

Recent Research

Here is what he has accomplished in recent years.

lastname	month	year	journal	title
Anz	3	2023	Arthroscopy : the journal of arthroscopic & related surgery : official publication of the Arthroscopy Association of North America and the International Arthroscopy Association	Editorial Commentary: Elbow Injury Results When Pediatric and Adolescent Throwing Athletes Throw as Hard as Possible, and Weighted Baseball Training Should Be Banned for Youth Athletes.
Anz	11	2022	Orthopaedic journal of sports medicine	Regulatory and Ethical Aspects of Orthobiologic Therapies.

Table of contents

Figure 1. A Simple Quarto® Report Generated for Dr. Adam Anz

INTEGRATING R DATA WITH MICROSOFT POWER BI®

There are three options for integrating R data with Microsoft Power BI®, each with their own benefits and limitations. First, the data within the final dataframes can be exported to Microsoft Excel® files using the `writexl` package, which can then be directly imported:

```
write_xlsx(All_full_list, "All_full_list.xlsx")
write_xlsx(All_clean_list, "All_clean_list.xlsx")
```

This option is only feasible for smaller dataframes that do not exceed the limits of spreadsheets.

A second option for larger dataframes is to save the in-memory data as an RData file that can then be loaded within an R script that is integrated within Microsoft Power BI®. Importantly, R must be installed on the same local machine as the Power BI client, and there is no need to include installation scripts for packages so long as they were already installed via an external IDE. The working directory must be specified at the start of the script to specific where the RData file is located:

```
Setwd("dir")
Load("data.RData")
```

An example of this process is included in Figure 2 and Figure 3.

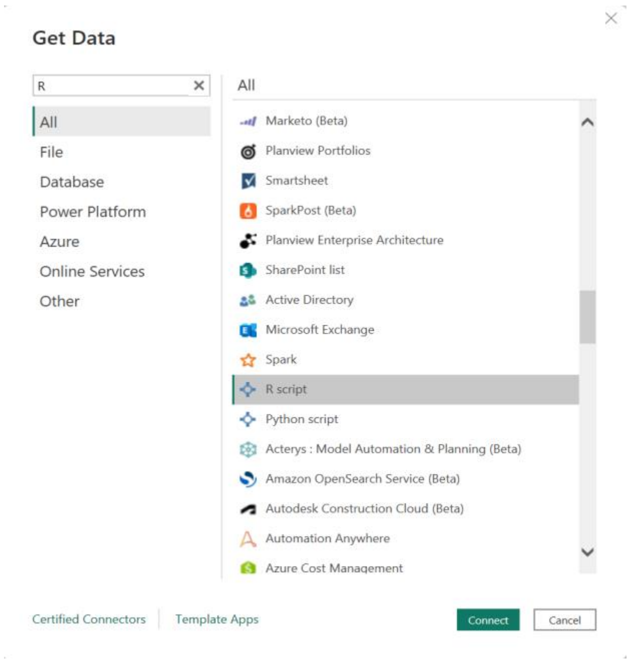
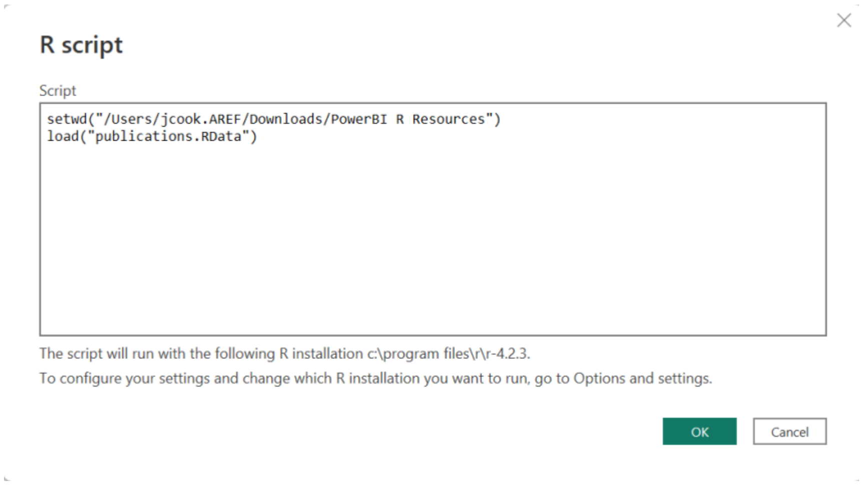


Figure 2. Getting Data via an R Script in Microsoft Power BI®.

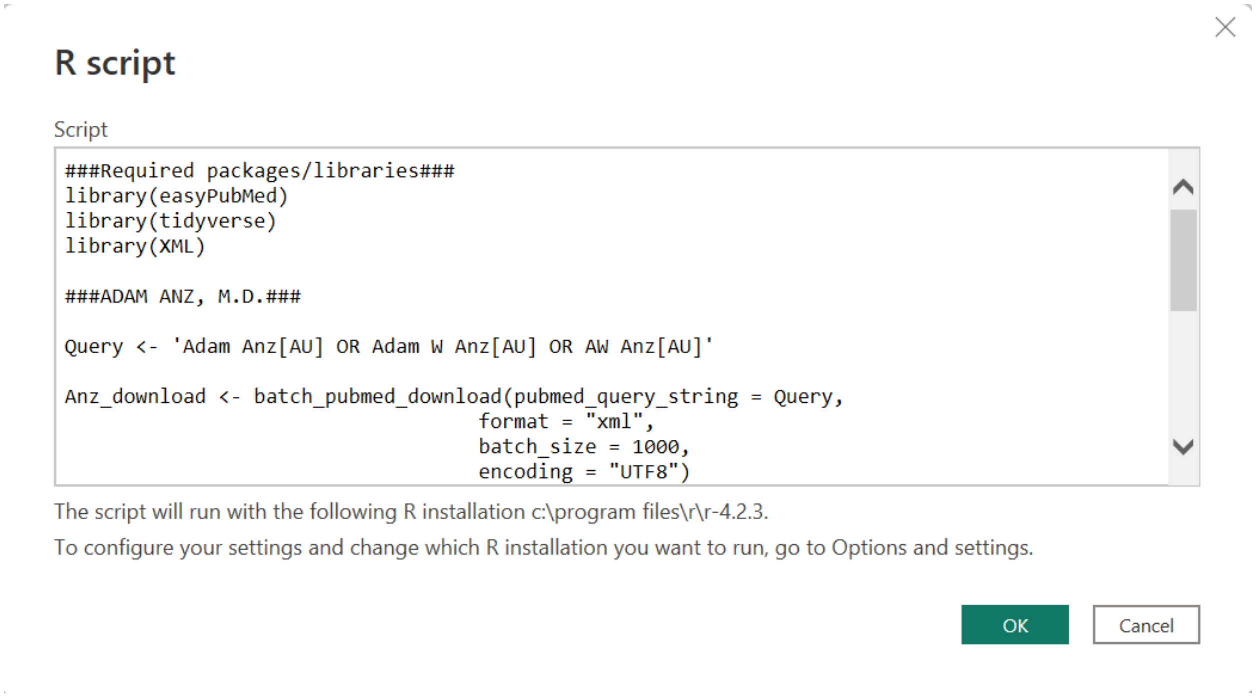


```
setwd("/Users/jcook.AREF/Downloads/PowerBI R Resources")
load("publications.RData")
```

The script will run with the following R installation c:\program files\r\r-4.2.3.
To configure your settings and change which R installation you want to run, go to Options and settings.

Figure 3. Setting the R Working Directory and Loading RData Files in Microsoft Power BI®.

Finally, the entire R script can be loaded within Microsoft Power BI® in the same way as option two. The benefit of this method is that there is no need to open a dedicated IDE such as RStudio®, and instead, the data is regenerated from the Entrez database at the time of import. However, libraries need to be loaded within the script and running the full script will take a much longer time to render as noted in the Quarto® tutorial above (Figure 4).



```
###Required packages/libraries###
library(easyPubMed)
library(tidyverse)
library(XML)

###ADAM ANZ, M.D.###

Query <- 'Adam Anz[AU] OR Adam W Anz[AU] OR AW Anz[AU]'

Anz_download <- batch_pubmed_download(pubmed_query_string = Query,
                                   format = "xml",
                                   batch_size = 1000,
                                   encoding = "UTF8")
```

The script will run with the following R installation c:\program files\r\r-4.2.3.
To configure your settings and change which R installation you want to run, go to Options and settings.

Figure 4. Running a Full R Script in Microsoft Power BI®.

All three import options will place the dataframes into Microsoft Power BI® as imported data that can be refreshed and with all column headers displayed as manipulatable fields.

BASIC, INTERACTABLE REPORTS WITH MICROSOFT POWER BI®

A full tutorial on Microsoft Power BI® is outside the scope of this paper, but the following steps can be taken to generate a simple report. Figures within these reports are dynamic, filterable, and can be cross-referenced. Importantly, these reports can be regenerated as the underlying data changes by simply refreshing the data using any of the above methods.

1. After data import, transform the data using standard Microsoft Power Query® syntax (outside the scope of this paper).
2. From the visualizations tab, edit report page settings to define the canvas size, background, and any other customizations.
3. Choose from available visualizations within Microsoft Power BI® or external visualizations from the visualization store, and other scripting options (such as R or Python).
4. Once a visualization is chosen, drag fields (data columns) of interest from the data tab to the visualization or filter tabs. The example in Figure 5 is an internal clustered bar chart with Investigator on the y-axis and Count of title on the x-axis.
5. Continue adding visualizations to tell the story of the data.
6. Add context using external text, images, and links.
7. Publish the final report (Figure 6) to the Microsoft Power BI Online Service® to distribute to stakeholders.

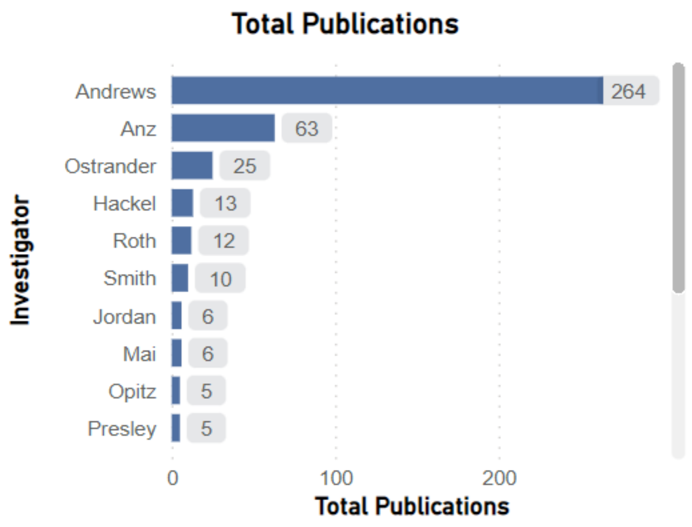


Figure 5. A Simple Cluster Bar Chart Generated Using Microsoft Power BI®.

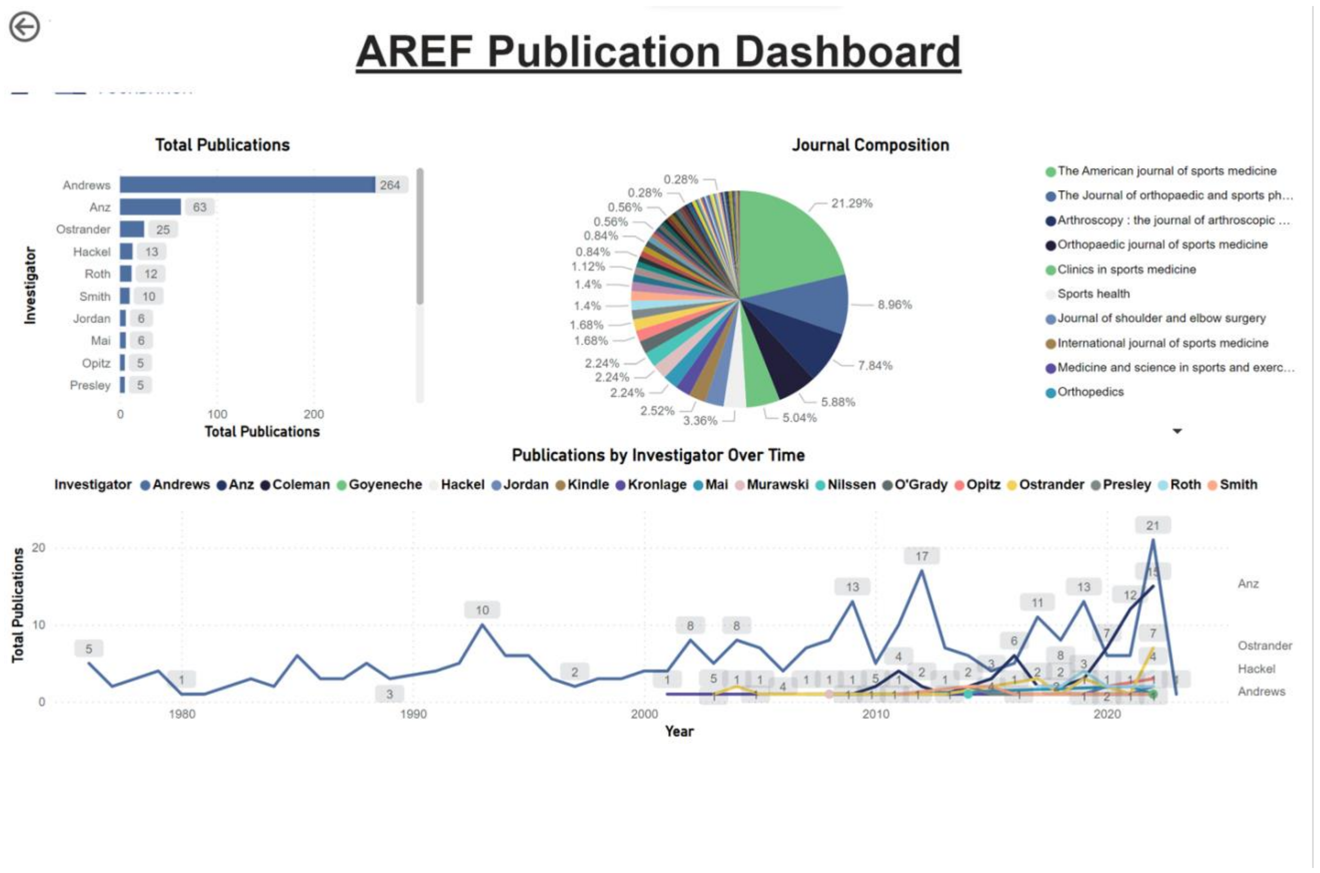

AREF Publication Dashboard
Total Publications
Investigator
Andrews 264
Anz 63
Ostrander 25
Hackel 13
Roth 12
Smith 10
Jordan 6
Mai 6
Opitz 5
Presley 5
0
100
200
Total Publications
Journal Composition
21.29%
8.96%
7.84%
5.88%
5.04%
3.36%
2.52%
2.24%
2.24%
2.24%
1.68%
1.68%
1.4%
1.4%
1.12%
0.84%
0.84%
0.56%
0.56%
0.28%
0.28%
The American journal of sports medicine
The Journal of orthopaedic and sports ph...
Arthroscopy : the journal of arthroscopic ...
Orthopaedic journal of sports medicine
Clinics in sports medicine
Sports health
Journal of shoulder and elbow surgery
International journal of sports medicine
Medicine and science in sports and exerc...
Orthopedics
Publications by Investigator Over Time
Investigator Andrews Anz Coleman Goyeneche Hackel Jordan Kindle Kronlage Mai Murawski Nilssen O'Grady Opitz Ostrander Presley Roth Smith
Total Publications
Year
1980
1990
2000
2010
2020
Anz
Ostrander
Hackel
Andrews

Figure 6. The Final Published Microsoft Power BI® Report.

BRANDING REPORTS WITH MICROSOFT POWER BI®

After generating simple figures, brand-specific colors, logos, and formatting guidelines can be applied through a variety of methods inside of the Microsoft Power BI® client. If needed, there are also options to generate figures using R scripts with packages such as `ggplot2` or `plotly`. A full, brand-specific publication report for the Andrews Research & Education Foundation (AREF) is included in Figure 7.

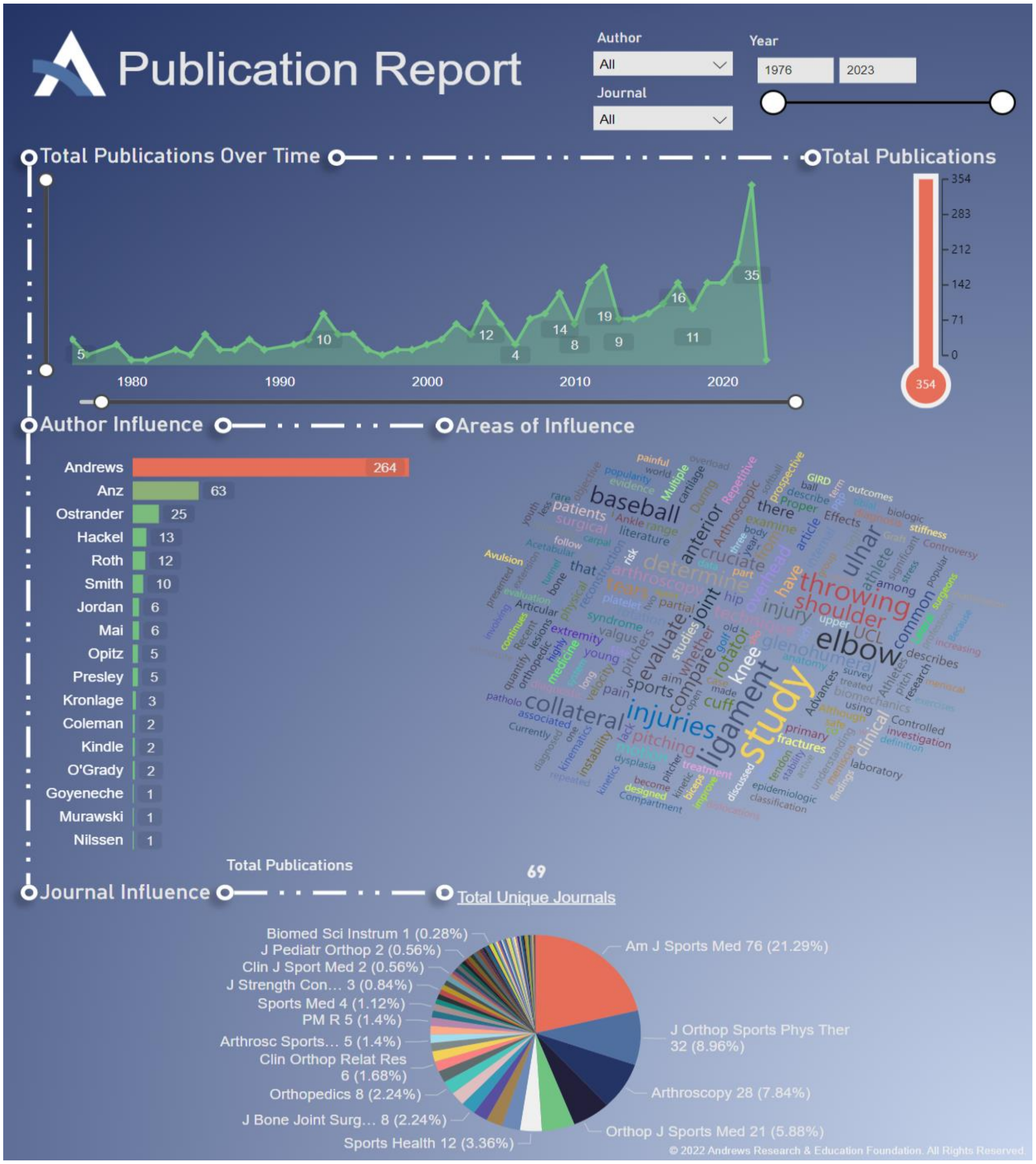


Figure 7. A complete, on-brand publication report generated for the Andrews Research & Education Foundation (AREF).

CONCLUSION

Publication metrics are increasingly being used to measure individual- and organization-level productivity and impact within academia and industry. Additionally, these metrics, including publication counts over time, are being used to make business decision. Historically, publication metrics have not been the easiest thing to quantify and manage. This is primarily due to the manual nature of querying PubMed from the web, which is difficult to do for multiple highly published authors with non-unique names. Instead of manually obtaining this data, it is much more feasible to leverage the R programming language and various reporting systems to manage this data. This text has demonstrated how R, a few R packages, and reporting systems such as Quarto® and Microsoft Power BI® can be integrated into a reproducible system capable of streamlining the collection, wrangling, and reporting process of publication data for stakeholders. In the future, this system should also capture article citation counts and journal impact factors to add more context to these metrics. Automation of this system (i.e., automated, timed data refreshes) using a third-party program should investigated as well.

REFERENCES

1. PubMed. National Center for Biotechnology Information (NCBI). 2023. https://pubmed.ncbi.nlm.nih.gov/
2. *R: A language and environment for statistical computing*. Version 4.2.3. R Foundation for Statistical Computing; 2023. https://www.R-project.org/
3. *Microsoft Power BI*. Microsoft; 2023. https://powerbi.microsoft.com/en-us/
4. *easyPubMed: Search and Retrieve Scientific Publication Records from PubMed*. Version 2.22. 2022. https://www.data-pulse.com/dev_site/easypubmed/
5. *XML: Tools for Parsing and Generating XML Within R and S-Plus*. Version 3.99-0.14. 2023. https://CRAN.R-project.org/package=XML
6. *Welcome to the tidyverse*. Journal of Open Source Software; 2019. https://www.tidyverse.org/
7. *Quarto*. Version 1.2. 2022. https://quarto.org/
8. *RStudio: Integrated Development Environment for R*. Version 2023.3.0.386. Posit Software, PBC; 2023. http://www.posit.co/

ACKNOWLEDGMENTS

The authors thank PharmaSUG for their support through the PharmaSUG Student Scholarship Program. We also thank Richann Watson, Louise Hadden, Troy Martin Hughes, Jim Box, Ellie Hartsfield, Ashton Hayward, and Adam Anz for their support with this project, and for their general guidance and mentorship.

RECOMMENDED READING

- *Retrieving and Processing PubMed Records using easyPubMed* (Retrieving and Processing PubMed Records using easyPubMed (data-pulse.com))
- *Advanced Features for Analysis of PubMed Records using easyPubMed* (Advanced Features for Analysis of PubMed Records using easyPubMed (data-pulse.com))
- *easyPubMed: working with retstart and retmax* (easyPubMed: working with retstart and retmax (data-pulse.com))
- *Using the SAS System® to interface with the Entrez Programming Utilities (E-utilities) of the National Center for Biotechnology Information (NCBI)*(Using the SAS System® to interface with the Entrez Programming Utilities (E-utilities) of the National Center for Biotechnology Information (NCBI). (pharmasug.org))

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Joshua J. Cook, B.S.
(850)736-1801
Jcook0312@outlook.com