

Package: pubDashboard (via r-universe)

March 21, 2025

Title Creating Publication Data Visualization Dashboards

Version 0.0.1

Description Package to facilitate the creation of data visualization dashboards through the flexdashboard and openalexR packages.

License GPL (>= 3)

Imports openalexR, rempsyc, insight, dplyr, rlang, stringr, countrycode, lubridate, tidyr, progress,

Suggests ggplot2, RColorBrewer, DT, waffle (>= 1.0.1), ggflags (>= 0.0.3), plotly, ggtext, dygraphs, xts, rstudioapi, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

LazyData true

URL <https://github.com/rempsys/pubDashboard>,
<https://rempsys.github.io/pubDashboard/>

BugReports <https://github.com/rempsys/pubDashboard/issues>

Remotes jimjam-slam/ggflags, hrbrmstr/waffle

Config/pak/sysreqs libicu-dev libssl-dev

Repository <https://rempsys.r-universe.dev>

RemoteUrl <https://github.com/rempsys/pubDashboard>

RemoteRef HEAD

RemoteSha 60b80fc01cc5730546179e7d43c9cd0d0a63e023

Contents

add_region	2
clean_journals_continents	3
countries	4
detect_missing_journals	4
dygraph_year	4
fetch_openalex_pubs	5
get_journal_id	6
journal_field	7
lapply_fetch_openalex_pubs	7
read_bind_all_data	9
render_dashboard	9
scatter_continent_year	11
scatter_country_year	12
scatter_figure1	13
scatter_journal_year	14
table_continent	15
table_continent_journal	15
table_continent_year	16
table_country	16
table_country_journal	17
table_country_year	18
table_journal_count	18
table_journal_year	19
table_missing_country	19
universities	20
us_states	21
waffle_continent	21
waffle_continent_journal	22
waffle_country	23
waffle_country_journal	23
world_capitals	24
Index	26

add_region	<i>Add regions to pubDashboard dataframe</i>
------------	--

Description

Add regions to pubDashboard dataframe

Usage

```
add_region(data, progress_bar = FALSE)
```

Arguments

`data` The dataframe on which to add region.
`progress_bar` Logical, whether to print a progress bar.

Examples

```
## Not run:  
x <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1, per_page = 1)  
x <- add_region(x)  
names(x)  
  
## End(Not run)
```

clean_journals_continents

Clean dataframe, for names of journals and continents

Description

Clean dataframe, for names of journals and continents

Usage

```
clean_journals_continents(data, progress_bar = FALSE)
```

Arguments

`data` The processed dataframe of data
`progress_bar` Logical, whether to print a progress bar.

Examples

```
## Not run:  
x <- fetch_openalex_pubs(journal_name = "Collabra",  
  pages = 1, per_page = 1, publication_year = 2024)  
x <- clean_journals_continents(x)  
names(x)  
  
## End(Not run)
```

countries	<i>List of countries taken from the package countrycode</i>
-----------	---

Description

List of countries taken from the package countrycode

Usage

```
countries
```

Format

A vector of class character containing country names.

detect_missing_journals	<i>Detect missing journals</i>
-------------------------	--------------------------------

Description

Detect missing journals

Usage

```
detect_missing_journals(data)
```

Arguments

data	The processed dataframe of data
------	---------------------------------

dygraph_year	<i>Generate a dygraph of journal paper percentages, by country and year</i>
--------------	---

Description

Generate a dygraph of journal paper percentages, by country and year

Usage

```
dygraph_year(data, level = "continent")
```

Arguments

data	The processed dataframe of data
level	Level of analysis, either country or continent

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
dygraph_year(data)
dygraph_year(data, "country")

## End(Not run)
```

fetch_openalex_pubs *Downloads relevant publication data using openalexR*

Description

Downloads relevant publication data using openalexR

Usage

```
fetch_openalex_pubs(
  journal_name = NULL,
  journal_id = NULL,
  clean_journals_continents = FALSE,
  progress_bar = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

journal_name	The list of desired journals (by journal name).
journal_id	The list of desired journals (by OpenAlex ID).
clean_journals_continents	Logical, whether to also process the dataframe with the clean_journals_continents function. It is set to FALSE by default because on large datasets it can be very time consuming.
progress_bar	Logical, whether to print a progress bar.
verbose	Passed to openalexR::oa_fetch() and defaults to TRUE.
...	Arguments passed to openalexR::oa_fetch()

Details

As recommended by the authors of the openalexR package,

Before we go any further, we highly recommend you set openalexR.mailto option so that your requests go to the polite pool for faster response times. If you have OpenAlex Premium, you can add your API key to the openalexR.apikey option as well. These lines best go into .Rprofile with file.edit("~/Rprofile").

```
options(openalexR.mailto = "example@email.com")
options(openalexR.apikey = "EXAMPLE_APIKEY")"
```

Examples

```
## Not run:
x <- fetch_openalex_pubs(journal_name = "Collabra",
  pages = 1, per_page = 1, publication_year = 2024)
names(x)
# Same as:
x <- fetch_openalex_pubs(journal_id = "S4210175756",
  pages = 1, per_page = 1, publication_year = 2024)
names(x)

## End(Not run)
```

get_journal_id	<i>Extract openalexR journal ID from journal names</i>
----------------	--

Description

Extract openalexR journal ID from journal names

Usage

```
get_journal_id(journal_name, verbose = TRUE)
```

Arguments

journal_name	The list of desired journals (by journal name).
verbose	Passed to <code>openalexR::oa_fetch()</code> and defaults to TRUE.

Details

As recommended by the authors of the openalexR package,

Before we go any further, we highly recommend you set openalexR.mailto option so that your requests go to the polite pool for faster response times. If you have OpenAlex Premium, you can add your API key to the openalexR.apikey option as well. These lines best go into .Rprofile with file.edit("~/Rprofile").

```
options(openalexR.mailto = "example@email.com")
options(openalexR.apikey = "EXAMPLE_APIKEY")"
```

Examples

```
## Not run:
x <- get_journal_id(journal_name = "Collabra")
x
x <- get_journal_id(journal_name = c(
  "Social Psychological and Personality Science", "Nature Human Behaviour"))
x$id

## End(Not run)
```

journal_field

List of academic journals and corresponding fields

Description

List of academic journals and corresponding fields

Usage

```
journal_field
```

Format

A data frame with 60 rows and 5 variables:

journal academic journal

journal_abbrev abbreviation of the journal name

openalex_id the OpenAlex ID

field the field of research

original_journal Whether it is part of one of the six original journals ...

lapply_fetch_openalex_pubs

Loop fetch_openalex_pubs over journals or years

Description

Loop fetch_openalex_pubs over journals or years

Usage

```
lapply_fetch_openalex_pubs(
  years = 1987:2023,
  journal_id,
  over = "year",
  from_publication_date = "1987-01-01",
  save = TRUE,
  file_suffix = "",
  verbose = TRUE,
  ...
)
```

Arguments

years	Desired list to loop over (for option over = "year").
journal_id	The list of desired journals (by OpenAlex ID).
over	Looping over what. Options are "year" or "journal".
from_publication_date	Start date (for option over = "journal").
save	Whether to save the data to disk.
file_suffix	What suffix to add to the file (e.g., "_new").
verbose	Passed to <code>openalexR::oa_fetch()</code> and defaults to TRUE.
...	Arguments passed to <code>openalexR::oa_fetch()</code>

Details

As recommended by the authors of the openalexR package,

Before we go any further, we highly recommend you set openalexR.mailto option so that your requests go to the polite pool for faster response times. If you have OpenAlex Premium, you can add your API key to the openalexR.apikey option as well. These lines best go into .Rprofile with file.edit("~/Rprofile").

```
options(openalexR.mailto = "example@email.com")
options(openalexR.apikey = "EXAMPLE_APIKEY")
```

Examples

```
## Not run:
lapply_fetch_openalex_pubs(
  over = "journal",
  journal_id = c("https://openalex.org/S90392387",
                "https://openalex.org/S33443600"),
  from_publication_date = "2024-01-01",
  to_publication_date = "2024-10-15",
  save = FALSE)

## End(Not run)
```

read_bind_all_data	<i>Read local pubDashboard data files and bind them in a single dataframe</i>
--------------------	---

Description

Read local pubDashboard data files and bind them in a single dataframe

Usage

```
read_bind_all_data(data_folder = "data", check_duplicate = FALSE)
```

Arguments

data_folder	The folder in which the data lives
check_duplicate	whether to check article ids with rempsyc::best_duplicate

render_dashboard	<i>Render complete pubDashboard dashboard</i>
------------------	---

Description

Render complete pubDashboard dashboard

Usage

```
render_dashboard(  
  file_name = "dashboard",  
  title = "title",  
  author = "author",  
  journal_name = NULL,  
  journal_id = NULL,  
  data_folder = "data",  
  tab_continent = TRUE,  
  tab_continent_year = TRUE,  
  tab_continent_journal = TRUE,  
  tab_country = TRUE,  
  tab_country_journal = TRUE,  
  tab_psychology = FALSE,  
  tab_economics = FALSE,  
  tab_general = FALSE,  
  tab_figure1 = FALSE,  
  tab_missing = TRUE,  
  ...  
)
```

Arguments

file_name	Desired file name.
title	Desired dashboard title.
author	Desired displayed dashboard author.
journal_name	The list of desired journals (by journal name).
journal_id	The list of desired journals (by OpenAlex ID).
data_folder	Folder where to save the data.
tab_continent	Whether to render the "Continent" tab.
tab_continent_year	Whether to render the "Continent by year" tab.
tab_continent_journal	Whether to render the "Continent by journal" tab.
tab_country	Whether to render the "Country" tab.
tab_country_journal	Whether to render the "Country by journal" tab.
tab_psychology	Whether to render the "Psychology" tab.
tab_economics	Whether to render the "Economics" tab.
tab_general	Whether to render the "General" tab.
tab_figure1	Whether to render the "Figure 1" tab.
tab_missing	Whether to render the "Missing" tab.
...	Arguments passed to <code>openalexR::oa_fetch()</code>

Examples

```
## Not run:
render_dashboard(
  file_name = "my_dashboard",
  title = "Wonderful Dashboard",
  author = "Rémi Thériault",
  journal_name = c("Journal of Personality and Social Psychology", "Health Psychology"),
  from_publication_date = "2024-01-01",
  tab_figure1 = TRUE
)

## End(Not run)
```

 scatter_continent_year

Generate table of journal paper percentages, by continent and year

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_continent_year(
  data,
  method = "lm",
  ymin = 0,
  ymax = 100,
  yby = 20,
  plotly = TRUE,
  citation = NULL,
  citation_size = 15,
  text_size = NULL,
  height = NULL,
  data_formatted = NULL,
  ...
)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess"
ymin	Minimum value for y-axis
ymax	Maximum value for y-axis
yby	Tick increments for y-axis
plotly	Logical, whether to use plotly for dynamic data visualization
citation	Optionally, a citation to add as a footer
citation_size	Font size of the citation
text_size	Size of the element_text ggplot2 element
height	Height argument of plotly::ggplotly
data_formatted	To save time, it is possible to reuse the dataframe used for this function that is generated by table_continent_year() .
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra Psychology", pages = 1)
data <- clean_journals_continents(data)
scatter_continent_year(data)

## End(Not run)
```

scatter_country_year *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_country_year(
  data,
  method = "lm",
  ymin = 0,
  ymax = 100,
  yby = 20,
  plotly = TRUE,
  citation = NULL,
  citation_size = 15,
  text_size = NULL,
  height = NULL,
  data_formatted = NULL,
  ...
)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess".
ymin	Minimum value for y-axis
ymax	Maximum value for y-axis
yby	Tick increments for y-axis
plotly	Logical, whether to use plotly for dynamic data visualization.
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.
text_size	Size of the element_text ggplot2 element
height	Height argument of plotly::ggplotly
data_formatted	To save time, it is possible to reuse the dataframe used for this function that is generated by table_country_year() .
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
suppressWarnings(scatter_country_year(data))

## End(Not run)
```

scatter_figure1	<i>Generate table of journal paper percentages, by continent and year</i>
-----------------	---

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_figure1(data, method = "lm", original = TRUE, plotly = TRUE, ...)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess".
original	Logical; if TRUE, attempts to mimic Arnett's (2008) Figure 1 in style.
plotly	Logical, whether to use plotly for dynamic data visualization.
...	Further arguments passed to rempsys::nice_scatter

Examples

```
## Not run:
journals <- c("Developmental Psychology",
             "Journal of Personality and Social Psychology",
             "Journal of Abnormal Psychology",
             "Journal of Family Psychology",
             "Health Psychology",
             "Journal of Educational Psychology"
            )

data <- fetch_openalex_pubs(journal_name = journals, pages = 10)
data <- clean_journals_continents(data)
scatter_figure1(data)

## End(Not run)
```

scatter_journal_year *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
scatter_journal_year(  
  data,  
  method = "lm",  
  ymin = 0,  
  ymax = 100,  
  yby = 20,  
  plotly = TRUE,  
  citation = NULL,  
  citation_size = 15,  
  ncol = 4,  
  ...  
)
```

Arguments

data	The processed dataframe of data
method	Which method to use for the regression line, either "lm" (default) or "loess"
ymin	Minimum value for y-axis
ymax	Maximum value for y-axis
yby	Tick increments for y-axis
plotly	Logical, whether to use plotly for dynamic data visualization
citation	Optionally, a citation to add as a footer
citation_size	Font size of the citation
ncol	How many columns for ggplot2::facet_wrap
...	Further arguments passed to rempsyc::nice_scatter

Examples

```
## Not run:  
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)  
data <- clean_journals_continents(data)  
scatter_journal_year(data)  
  
## End(Not run)
```

table_continent	<i>Generate table of journal paper percentages, by continent</i>
-----------------	--

Description

Generate table of journal paper percentages, by continent

Usage

```
table_continent(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
table_continent(data)

## End(Not run)
```

table_continent_journal	<i>Generate table of journal paper percentages, by continent and journals</i>
-------------------------	---

Description

Generate table of journal paper percentages, by continent and journals

Usage

```
table_continent_journal(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
table_continent_journal(data)

## End(Not run)
```

table_continent_year *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
table_continent_year(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a <code>DT::datatable</code> HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
table_continent_year(data)

## End(Not run)
```

table_country *Generate table of journal paper percentages, by country*

Description

Generate table of journal paper percentages, by country

Usage

```
table_country(data, datatable = TRUE)
```


Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:  
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)  
data <- clean_journals_continents(data)  
table_country(data)  
  
## End(Not run)
```

table_country_journal *Generate table of journal paper percentages, by continent and year*

Description

Generate table of journal paper percentages, by continent and year

Usage

```
table_country_journal(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:  
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)  
data <- clean_journals_continents(data)  
table_country_journal(data)  
  
## End(Not run)
```

table_country_year	<i>Generate table of journal paper percentages, by country and year</i>
--------------------	---

Description

Generate table of journal paper percentages, by country and year

Usage

```
table_country_year(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:  
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)  
data <- clean_journals_continents(data)  
table_country_year(data)  
  
## End(Not run)
```

table_journal_count	<i>Count number of papers per journal, with year range</i>
---------------------	--

Description

Count number of papers per journal, with year range

Usage

```
table_journal_count(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
table_journal_count(data)

## End(Not run)
```

table_journal_year	<i>Generate table of journal paper percentages, by journal, continent and year</i>
--------------------	--

Description

Generate table of journal paper percentages, by journal, continent and year

Usage

```
table_journal_year(data, datatable = TRUE)
```

Arguments

data	The processed dataframe of data
datatable	Whether to output a DT::datatable HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
table_journal_year(data)

## End(Not run)
```

table_missing_country	<i>Generate table of journal paper percentages, by country</i>
-----------------------	--

Description

Generate table of journal paper percentages, by country

Usage

```
table_missing_country(data, datatable = TRUE)
```

Arguments

<code>data</code>	The processed dataframe of data
<code>datatable</code>	Whether to output a <code>DT::datatable</code> HTML table widget instead of a regular dataframe (defaults to TRUE).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
data[1, c(4, 6)] <- NA
table_missing_country(data)

## End(Not run)
```

<code>universities</code>	<i>A data frame of university and corresponding country</i>
---------------------------	---

Description

Obtained from GitHub, and then modified with minor improvements and more universities.

Usage

```
universities
```

Format

A data frame with 9420 rows and 2 variables:

country_code the country code

university the university ...

Source

<https://raw.githubusercontent.com/endSly/world-universities-csv/master/world-universities.csv>

us_states	<i>List of US states taken from the package countrycode</i>
-----------	---

Description

List of US states taken from the package countrycode

Usage

```
us_states
```

Format

A data frame with 50 rows and 3 variables:

state.name the name of the state

state.abb the name of the abbreviation

state.regex the regex for that state ...

waffle_continent	<i>Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)</i>
------------------	--

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```
waffle_continent(  
  data,  
  citation = NULL,  
  citation_size = NULL,  
  data_formatted = NULL  
)
```

Arguments

data The processed dataframe of data

citation Optionally, a citation to add as a footer.

citation_size Font size of the citation.

data_formatted To save time, it is possible to reuse the dataframe used for this function that is generated by [table_continent\(\)](#).

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Journal of Economic Psychology", pages = 1)
data <- clean_journals_continents(data)
waffle_continent(data)

## End(Not run)
```

```
waffle_continent_journal
```

*Generate a waffle chart of journal paper percentages, by continent
(each square = 1% of data)*

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```
waffle_continent_journal(
  data,
  citation = NULL,
  citation_size = NULL,
  journal_abbreviation = TRUE,
  data_formatted = NULL
)
```

Arguments

<code>data</code>	The processed dataframe of data
<code>citation</code>	Optionally, a citation to add as a footer.
<code>citation_size</code>	Font size of the citation.
<code>journal_abbreviation</code>	Logical, whether to use the journal abbreviation to fit the entire plot, otherwise some journal names can be quite long and accordingly be cropped.
<code>data_formatted</code>	To save time, it is possible to reuse the dataframe used for this function that is generated by table_continent_journal() .

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra Psychology", pages = 1)
data <- clean_journals_continents(data)
waffle_continent_journal(data)

## End(Not run)
```

waffle_country	<i>Generate a waffle plot made of country flags</i>
----------------	---

Description

Generate a waffle plot made of country flags

Usage

```
waffle_country(
  data,
  citation = NULL,
  citation_size = NULL,
  data_formatted = NULL,
  flag_size = 8.5
)
```

Arguments

data	The processed dataframe of data
citation	Optionally, a citation to add as a footer.
citation_size	Font size of the citation.
data_formatted	To save time, it is possible to reuse the dataframe used for this function that is generated by table_country() .
flag_size	Size of flags.

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
waffle_country(data)

## End(Not run)
```

waffle_country_journal	<i>Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)</i>
------------------------	--

Description

Generate a waffle chart of journal paper percentages, by continent (each square = 1% of data)

Usage

```
waffle_country_journal(
  data,
  citation = NULL,
  citation_size = NULL,
  journal_abbreviation = TRUE,
  data_formatted = NULL
)
```

Arguments

<code>data</code>	The processed dataframe of data
<code>citation</code>	Optionally, a citation to add as a footer.
<code>citation_size</code>	Font size of the citation.
<code>journal_abbreviation</code>	Logical, whether to use the journal abbreviation to fit the entire plot, otherwise some journal names can be quite long and accordingly be cropped.
<code>data_formatted</code>	To save time, it is possible to reuse the dataframe used for this function that is generated by <code>table_country_journal()</code> .

Examples

```
## Not run:
data <- fetch_openalex_pubs(journal_name = "Collabra", pages = 1)
data <- clean_journals_continents(data)
waffle_country_journal(data)

## End(Not run)
```

world_capitals	<i>List of world capitals taken from the package maps</i>
----------------	---

Description

List of world capitals taken from the package maps

Usage

```
world_capitals
```

Format

A data frame with 259 rows and 6 variables:

name the name of the capital

country.etc the country of the capital

pop population of the capital

lat latitude of the capital

long longitude of the capital

capital whether it is a capital ...

Index

- * **datasets**
 - countries, [4](#)
 - journal_field, [7](#)
 - universities, [20](#)
 - us_states, [21](#)
 - world_capitals, [24](#)
- add_region, [2](#)
- clean_journals_continents, [3, 5](#)
- countries, [4](#)
- detect_missing_journals, [4](#)
- DT::datatable, [15–20](#)
- dygraph_year, [4](#)
- fetch_openalex_pubs, [5](#)
- get_journal_id, [6](#)
- ggplot2::facet_wrap, [14](#)
- journal_field, [7](#)
- lapply_fetch_openalex_pubs, [7](#)
- openalexR::oa_fetch(), [5, 6, 8, 10](#)
- plotly::ggplotly, [11, 12](#)
- read_bind_all_data, [9](#)
- rempsysc::best_duplicate, [9](#)
- rempsysc::nice_scatter, [11–14](#)
- render_dashboard, [9](#)
- scatter_continent_year, [11](#)
- scatter_country_year, [12](#)
- scatter_figure1, [13](#)
- scatter_journal_year, [14](#)
- table_continent, [15](#)
- table_continent(), [21](#)
- table_continent_journal, [15](#)
- table_continent_journal(), [22](#)
- table_continent_year, [16](#)
- table_continent_year(), [11](#)
- table_country, [16](#)
- table_country(), [23](#)
- table_country_journal, [17](#)
- table_country_journal(), [24](#)
- table_country_year, [18](#)
- table_country_year(), [12](#)
- table_journal_count, [18](#)
- table_journal_year, [19](#)
- table_missing_country, [19](#)
- universities, [20](#)
- us_states, [21](#)
- waffle_continent, [21](#)
- waffle_continent_journal, [22](#)
- waffle_country, [23](#)
- waffle_country_journal, [23](#)
- world_capitals, [24](#)