Package 'dragon'

April 8, 2022

Title Deep Time Redox Analysis of the Geobiology Ontology Network

Version 1.2.1

Description Create, visualize, manipulate, and analyze bipartite mineral-chemistry networks for set of focal element(s) across deep-time on Earth. The method is described in Spielman and Moore (2020) <doi:10.3389/feart.2020.585087>.

License MIT + file LICENSE

Imports config, golem (>= 0.2.1), shiny, DT (>= 0.14), ggplot2 (>= 3.3.5), readr, openxlsx, dplyr (>= 1.0.0), RColorBrewer, stringr, tidyr (>= 1.0.0), purrr, tibble, broom (>= 0.5.6), cowplot (>= 1.0.0), ggforce, magrittr, shinydashboard, shinyWidgets, colourpicker (>= 1.0), colorspace (>= 1.4), visNetwork (>= 2.0.9), igraph (>= 1.3.0), rlang, htmltools, stats, promises, future, lubridate, xml2, rvest, curl, tidyselect

Encoding UTF-8

RoxygenNote 7.1.2

Suggests testthat (>= 2.1.0), processx, knitr, zip, rmarkdown

VignetteBuilder knitr

URL `https://github.com/sjspielman/dragon`

BugReports `https://github.com/sjspielman/dragon/issues`

Depends R (>= 3.5.0)

Collate 'utils_definitions.R' 'utils_globals.R' 'utils_names.R' 'utils_style-network.R' 'utils_ui-choices.R' 'fct_prepare-med-data.R' 'fct_build-legend.R' 'fct_build-linear-models.R' 'fct_build-network.R' 'fct_build-shiny-tables.R' 'fct_calculate-network-info.R' 'fct_export-network.R' 'fct_style-network.R' 'fct_timeline.R' 'mod_choose-color-palette.R' 'mod_choose-custom-element-colors.R' 'app_config.R' 'app_server.R' 'app_ui.R' 'run_app.R'

NeedsCompilation no

Author Stephanie J. Spielman [aut, cre]
(<https://orcid.org/0000-0002-9090-4788>)

Maintainer Stephanie J. Spielman <stephanie.spielman@gmail.com>

Repository CRAN

Date/Publication 2022-04-08 08:42:33 UTC

R topics documented:

initialize_network 2
run_app 4
run_dragon 4

Index **5**

`initialize_network` *Initialize a mineral-chemistry network as stand-alone network rather than for embedding into the Shiny App.*

Description

Initialize a mineral-chemistry network as stand-alone network rather than for embedding into the Shiny App.

Usage

```
initialize_network(
  elements_of_interest,
  force_all_elements = FALSE,
  elements_by_redox = FALSE,
  restrict_to_elements = FALSE,
  ignore_na_redox = FALSE,
  age_range = c(0, 5),
  max_age_type = "Maximum",
  cluster_algorithm = "Louvain",
  cluster_seed = NULL,
  use_data_cache = TRUE
)
```

Arguments

`elements_of_interest`
: An array of specified elements whose minerals should be included in the network. For all elements, specify "all".

`force_all_elements`
: A logical. If FALSE (default), minerals containing any of ‘elements_of_interest‘ will be included in network. If TRUE, only minerals with full intersection of all specified elements will be included in network.

`elements_by_redox`	A logical. If FALSE (default), element nodes will be constructed regardless of redox state. If TRUE, creates separate node for each element's redox state, e.g. Fe2+ and Fe3+ would be separate nodes.
`restrict_to_elements`	A logical. If FALSE (default), constructed network will _only_ contain the specified focal element(s)
`ignore_na_redox`	A logical. If TRUE and elements_by_redox is TRUE, element nodes without redox states will be removed from the network.
`age_range`	A array of two numbers giving inclusive range of mineral ages in Ga to include in network.
`max_age_type`	A string indicating how mineral ages should be assessed. If "Maximum" (default), filters minerals using maximum known ages at localities. If "Minimum", filters minerals using minimum known ages at localities.
`cluster_algorithm`	A string giving community clustering algorithm, one of "Louvain" (default) or "Leading eigenvector".
`cluster_seed`	An integer giving a random seed for reproducible clustering. Default is NULL. "Louvain" (default) or "Leading eigenvector".
`use_data_cache`	A logical. If TRUE (default) cached Mineral Evolution Database will be used to build the network. If FALSE, data will be fetched from MED here. CAUTION: Requires internet connection and will take several minutes to update.

Value

Named list containing an igraph-formatted network (‘network’), an igraph::communities object giving node cluster memberships (‘clustering’), a tibble of nodes associated metadata (‘nodes’), and a tibble of edges and associated metadata (‘edges’), and a tibble of mineral locality information (‘locality_info‘)

Examples

```
## Not run:
# Include all Iron minerals whose maximum known age is between 1-2 Gya, and apply
#   Louvain community clustering
initialize_network("Fe", age_range = c(1,2))

# Include all minerals containing either Iron and Oxygen whose maximum known age
#   is between 1-2 Gya
initialize_network(c("Fe", "O"), age_range = c(1,2))

# Include all minerals containing both Iron and Oxygen whose maximum known age is
#   between 1-2 Gya
initialize_network(c("Fe", "O"), force_all_elements = TRUE, age_range = c(1,2))

# Build the full mineral network
initialize_network("all")
```

```
## End(Not run)
```

run_app *Run the "dragon" Shiny Application*

Description

Run the "dragon" Shiny Application

Usage

```
run_app()
```

Examples

```
## Not run:
library(dragon)
dragon::run_app()

## End(Not run)
```

run_dragon *Run the "dragon" Shiny Application. Wrapper for dragon::run_app().*

Description

Run the "dragon" Shiny Application. Wrapper for dragon::run_app().

Usage

```
run_dragon()
```

Examples

```
## Not run:
library(dragon)
dragon::run_dragon()

## End(Not run)
```

Index

initialize_network, 2

run_app, 4
run_dragon, 4