

Package ‘ClimMobTools’

January 20, 2025

Type Package

Title API Client for the 'ClimMob' Platform

Version 1.4

URL <https://agrdatasci.github.io/ClimMobTools/>

BugReports <https://github.com/agrdatasci/ClimMobTools/issues>

Description API client for 'ClimMob', an open source software for decentralized large-N trials with the 'tricot' approach <<https://climmob.net/>>. Developed by van Etten et al. (2019) <[doi:10.1017/S0014479716000739](https://doi.org/10.1017/S0014479716000739)>, it turns the research paradigm on its head; instead of a few researchers designing complicated trials to compare several technologies in search of the best solutions for the target environment, it enables many participants to carry out reasonably simple experiments that taken together can offer even more information. 'ClimMobTools' enables project managers to deep explore and analyse their 'ClimMob' data in R.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5.0)

Imports httr, jsonlite, lpSolve, Matrix, methods, RSpectra, stats, utils

Suggests climatrends, gosset, knitr, nasapower, rmarkdown, sf, PlackettLuce, testthat (>= 2.1.0)

Language en-US

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Author Kauê de Sousa [aut, cre] (<<https://orcid.org/0000-0002-7571-7845>>),
Jacob van Etten [aut] (<<https://orcid.org/0000-0001-7554-2558>>),
Brandon Madriz [aut]

Maintainer Kauê de Sousa <desousa.kaue@gmail.com>

Repository CRAN

Date/Publication 2025-01-16 11:20:02 UTC

Contents

| | |
|--------------------|-----------|
| ClimMobTools | 2 |
| getDataCM | 3 |
| getProjectProgress | 4 |
| getProjectsCM | 5 |
| getTraitList | 6 |
| randomize | 7 |
| rankTricot | 9 |
| rmGeoIdentity | 11 |
| Index | 12 |

ClimMobTools

API Client for the 'ClimMob' platform in R

Description

API client for 'ClimMob', an open source software for decentralized large-N trials with the 'tricot' approach <https://climmob.net/>. Developed by van Etten et al. (2019) [doi:10.1017/S0014479716000739](https://doi.org/10.1017/S0014479716000739), it turns the research paradigm on its head; instead of a few researchers designing complicated trials to compare several technologies in search of the best solutions for the target environment, it enables many participants to carry out reasonably simple experiments that taken together can offer even more information. 'ClimMobTools' enables project managers to deep explore and analyse their 'ClimMob' data in R.

Author(s)

Kauê de Sousa and Jacob van Etten and Brandon Madriz

See Also

Useful links:

- Development repository: <https://github.com/agrdatasci/ClimMobTools>
- Static documentation: <https://agrdatasci.github.io/ClimMobTools/>
- Report bugs: <https://github.com/agrdatasci/ClimMobTools/issues>
- The tricot user guide: <https://hdl.handle.net/10568/109942>

getDataCM

Get ClimMob data

Description

Fetch the data from a ClimMob project using an application programming interface (API) key

Usage

```

getDataCM(
  key,
  project,
  userowner,
  as.data.frame = TRUE,
  as.text = FALSE,
  server = "climmob3",
  ...
)

## S3 method for class 'CM_list'
as.data.frame(x, ..., tidynames = TRUE, pivot.wider = FALSE)

```

Arguments

| | |
|---------------|--|
| key | character, the user's API key |
| project | character, the project id |
| userowner | character, username of project's owner |
| as.data.frame | logical, to return a data frame, as.data.frame = FALSE returns a list |
| as.text | logical, to return a text file that can be parsed to json |
| server | character, to indicate from which server the data will be retrieved. See details |
| ... | additional arguments passed to methods |
| x | an object of class CM_list |
| tidynames | logical, TRUE make clean column names |
| pivot.wider | logical, if TRUE return a wider object where each tricot package is a row |

Details

server: options are: "climmob" or "1000farms"

Value

An object of class 'CM_list' or a text file or a data.frame with class "CM_df" with the variables:

| | |
|----------|------------------------------|
| id | the participant's package id |
| moment | the data collection moment |
| variable | the variable name |
| value | the value for each variable |

Author(s)

Kauê de Sousa

See AlsoOther GET functions: [getProjectProgress\(\)](#), [getProjectsCM\(\)](#)**Examples**

```
# This function only works with an API key
# the API key can be obtained from your ClimMob account

library("ClimMobTools")
my_key = "d39a3c66-5822-4930-a9d4-50e7da041e77"

getDataCM(key = my_key,
           project = "breadwheat",
           userowner = "gosset",
           server = "climmob3")

# get in the wide format

getDataCM(key = my_key,
           project = "breadwheat",
           userowner = "gosset",
           server = "climmob3",
           pivot.wider = TRUE)
```

| | |
|--------------------|-----------------------------|
| getProjectProgress | <i>Get project progress</i> |
|--------------------|-----------------------------|

Description

Fetch the progress of a ClimMob project

Usage

```
getProjectProgress(key, project, userowner, server = "climmob3")
```

Arguments

| | |
|-----------|--|
| key | character, the user's API key |
| project | character, the project id |
| userowner | character, username of project's owner |
| server | character, to indicate from which server the data will be retrieved. See details |

Details

server: the default server is "climmbob" used for clients of <https://climmbob.net/climmbob3/>, other options are:

"1000farms" for clients of <https://1000farms.climmbob.net/>

Value

A list with number of submissions per assessment and submissions per assessment per enumerator

Author(s)

Kauê de Sousa

See Also

Other GET functions: [getDataCM\(\)](#), [getProjectsCM\(\)](#)

Examples

```
# This function only works with an API key
# the API key can be obtained from your ClimMob account

library("ClimMobTools")
my_key = "ff05a174-28d0-4a40-ab5a-35dc486133a6"

getProjectProgress(key = my_key,
                  project = "gina2024",
                  userowner = "student",
                  server = "1000FARMS")
```

getProjectsCM

Get ClimMob projects

Description

Fetch the status of ClimMob projects

Usage

```
getProjectsCM(key, server = "climmbob3", ...)
```

Arguments

| | |
|--------|--|
| key | character, the user's API key |
| server | character, to indicate from which server the data will be retrieved. See details |
| ... | additional arguments passed to methods. See details |

Details

server: the default server is "climmbob" used for clients of <https://climmbob.net/climmbob3/>, other options are:

"1000farms" for clients of <https://1000farms.climmob.net/>

Value

A data.frame with the variables:

| | |
|---------------|---|
| project_id | the project's id |
| project_name | the project's name |
| user_owner | the account name that owns the project |
| country | the country of project's implementation |
| status | the current status |
| creation_date | date where the project was created |

Author(s)

Kauê de Sousa

See Also

Other GET functions: [getDataCM\(\)](#), [getProjectProgress\(\)](#)

Examples

```
# This function only works with an API key
# the API key can be obtained from your ClimMob account

my_key = "ff05a174-28d0-4a40-ab5a-35dc486133a6"

getProjectsCM(key = my_key, server = "1000FARMS")
```

getTraitList

Organize trait ranks in a ClimMob data

Description

This function helps in identifying the traits assessed in the tricot project and validates the data returning a list with logical vectors to support the transformation of tricot rankings into a PlackettLuce ranking

Usage

```
getTraitList(data, pattern, trait.labels = NULL, ...)
```

Arguments

| | |
|---------------------------|---|
| <code>data</code> | data.frame, the ClimMob data |
| <code>pattern</code> | character, the tricot ranking pattern |
| <code>trait.labels</code> | an optional character with clean trait labels |
| <code>...</code> | additional arguments, not implemented yet |

Value

a list with trait validation data

Examples

```
require("gosset")  
  
data("breadwheat", package = "gosset")  
  
getTraitList(breadwheat, c("_best", "_worst"))
```

randomize

Set an experimental incomplete block design

Description

Generate an incomplete block A-optimal design. The function is optimized for incomplete blocks of three, but it will also work with comparisons of any other number of options. The design strives for approximate A optimality, this means that it is robust to missing observations. It also strives for balance for positions of each option. Options are equally divided between first, second, third, etc. position. The strategy is to create a "pool" of combinations that does not repeat combinations and is A-optimal. Then this pool is ordered to make subsets of consecutive combinations also relatively balanced and A-optimal

Usage

```
randomize(  
  npackages,  
  itemnames,  
  ncomp = 3,  
  availability = NULL,  
  props = NULL,  
  ...  
)
```

Arguments

| | |
|---------------------------|---|
| <code>npackages</code> | an integer for the number of incomplete blocks to be generated |
| <code>itemnames</code> | a character for the name of items tested in the experiment |
| <code>ncomp</code> | an integer for the number of items to be assigned to each incomplete block |
| <code>availability</code> | optional, a vector with integers indicating the number of plots available for each <i>itemnames</i> |
| <code>props</code> | optional, a numeric vector with the desired proportions for each <i>itemnames</i> |
| <code>...</code> | additional arguments passed to methods |

Value

A dataframe with the randomized design

Author(s)

Jacob van Etten

References

Bailey and Cameron (2004). Combinations of optimal designs. <https://webpace.maths.qmul.ac.uk/l.h.soicher/designtheory.org/library/preprints/optimal.pdf>

Examples

```
ncomp = 3
npackages = 20
itemnames = c("apple", "banana", "grape", "mango", "orange")
availability = c(5, 8, 50, 50, 50)

randomize(ncomp = ncomp,
          npackages = npackages,
          itemnames = itemnames)

randomize(ncomp = ncomp,
          npackages = npackages,
          itemnames = itemnames,
          availability = availability)

# run diagnostics to certify that randomization is balanced
# the number of interactions should have the lower sd as possible
# this verification may not work well when technologies are
# tested in different proportions
design = randomize(ncomp = ncomp,
                 npackages = npackages,
                 itemnames = itemnames)

design$best = "A"
```



```

design$worst = "C"

# number of times each item is tested in the
# trial design
ntest = table(unlist(design[,c(1:3)]))

ntest

# put into the PlackettLuce structure to check
# number of interactions between items
r = gosset::rank_tricot(design, c(1:3), c(4:5))

bn = gosset::set_binomialfreq(r)

bn$interactions = bn$win1 + bn$win2

bn = bn[,c(1,2,5)]

bn

```

rankTricot

Build Plackett-Luce rankings from tricot dataset

Description

Create an object of class "rankings" from tricot data

Usage

```

rankTricot(
  data,
  items,
  input,
  group = FALSE,
  validate.rankings = FALSE,
  additional.rank = NULL,
  ...
)

```

Arguments

| | |
|-------|---|
| data | a data.frame with columns specified by items and input values |
| items | a character or numerical vector for indexing the column(s) containing the item names in data |
| input | a character or numerical vector for indexing the column(s) containing the values in data to be ranked |
| group | logical, if TRUE return an object of class "grouped_rankings" |

```

validate.rankings      logical, if TRUE implements a check on ranking consistency looking for possible
                       ties, NA or letters other than A, B, C. These entries are set to 0
additional.rank        optional, a data frame for the comparisons between tricot items and the local
                       item
...                   additional arguments passed to methods. See details

```

Details

full.output: logical, to return a list with a "rankings", a "grouped_rankings" and the ordered items

Value

a PlackettLuce "rankings" or "grouped_rankings" object

Author(s)

Kauê de Sousa and Jacob van Etten, with ideas from Heather Turner

Examples

```

# beans data where each observer compares 3 varieties randomly distributed
# from a list of 11 and additionally compares these 3 varieties
# with their local variety
if (require("PlackettLuce")){
  data("beans", package = "PlackettLuce")

  # first build rankings with only tricot items
  # and return an object of class 'rankings'
  R = rankTricot(data = beans,
                 items = c(1:3),
                 input = c(4:5))

  head(R)

#####

# pass the comparison with local item as an additional rankings, then
# each of the 3 varieties are compared separately with the local item
# and return an object of class grouped_rankings
G = rankTricot(data = beans,
               items = c(1:3),
               input = c(4:5),
               group = TRUE,
               additional.rank = beans[c(6:8)])

  head(G)
}

```

| | |
|---------------|-------------------------------------|
| rmGeoIdentity | <i>Remove geographical identity</i> |
|---------------|-------------------------------------|

Description

Build a buffer around the a set of geographical coordinates and take a random point around the buffer. The function is used to omit the precise location of tricot participants but keeping a close distance to the trial environment.

Usage

```
rmGeoIdentity(longlat, dist = 0.015, nQuadSegs = 2L, ...)
```

Arguments

| | |
|-----------|---|
| longlat | a data.frame or matrix with geographical coordinates long lat |
| dist | numeric, buffer distance for all <i>lonlat</i> |
| nQuadSegs | integer, number of segments per quadrant |
| ... | further arguments passed to sf methods |

Value

A data frame with the random coordinates long lat within the buffer

Examples

```
xy = matrix(c(11.097799, 60.801090,  
             11.161298, 60.804199,  
             11.254428, 60.822457),  
           nrow = 3, ncol = 2, byrow = TRUE)
```

```
rmGeoIdentity(xy)
```

```
#' the function also handles NAs
```

```
xy2 = matrix(c(11.097799, 60.801090,  
              NA, NA,  
              11.161298, 60.804199,  
              11.254428, 60.822457,  
              11.254428, NA),  
            nrow = 5, ncol = 2, byrow = TRUE)
```

```
rmGeoIdentity(xy2)
```

Index

* GET functions

getDataCM, [3](#)

getProjectProgress, [4](#)

getProjectsCM, [5](#)

as.data.frame.CM_list (getDataCM), [3](#)

ClimMobTools, [2](#)

ClimMobTools-package (ClimMobTools), [2](#)

getDataCM, [3](#), [5](#), [6](#)

getProjectProgress, [4](#), [4](#), [6](#)

getProjectsCM, [4](#), [5](#), [5](#)

getTraitList, [6](#)

randomise (randomize), [7](#)

randomize, [7](#)

rankTricot, [9](#)

rmGeoIdentity, [11](#)

sf, [11](#)