

# Package: ClimMobTools (via r-universe)

October 22, 2024

**Type** Package

**Title** API Client for the 'ClimMob' Platform

**Version** 1.2

**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

**LazyData** false

**Depends** R (>= 3.5.0)

**Imports** httr, jsonlite, lpSolve, Matrix, methods, RSpecra, stats, utils

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

**Language** en-US

**RoxygenNote** 7.2.1

**VignetteBuilder** knitr

**Repository** <https://agrdatasci.r-universe.dev>

**RemoteUrl** <https://github.com/agrdatasci/climmobtools>

**RemoteRef** HEAD

**RemoteSha** e43e1c2411dc6a73bda515ea446bceca79bb9b1d

## Contents

ClimMobTools . . . . .	2
getDataCM . . . . .	3
getProjectProgress . . . . .	4
getProjectsCM . . . . .	6
getTraitList . . . . .	7
randomize . . . . .	8
rankTricot . . . . .	9
rmGeoIdentity . . . . .	11
<b>Index</b>	<b>13</b>

---

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>
- ClimMob Platform: <https://climmob.net>
- 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	a character for the user's application programming interface (API) key
project	a character for the project id
userowner	a character for user name of project's owner
as.data.frame	logical, to return a data frame
as.text	logical, to return a text file that can be parsed to json
server	optional, a character to select from which server the data will be retrieved. See details
...	additional arguments passed to methods
x	an object of class <code>CM_list</code>
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: the default server is "climmob" used for clients of <https://climmob.net/climmob3/>, other options are:

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

**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 Also**

ClimMob website <https://climmob.net/>

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

**Examples**

```
# This function only works with an API key
# the API key can be obtained once a free ClimMob account
# is created via https://climmob.net/
```

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

```
getDataCM(key = my_key,
           project = "beanaru23",
           userowner = "student",
           server = "1000farms")
```

```
# get in the wide format
```

```
getDataCM(key = my_key,
           project = "beanaru23",
           userowner = "student",
           server = "1000farms",
           pivot.wider = TRUE)
```

---

getProjectProgress      *Get project progress*

---

**Description**

Fetch the progress of a ClimMob project

**Usage**

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

**Arguments**

key	a character for the user's application programming interface (API) key
project	a character for the project id
userowner	a character for user name of project's owner
server	optional, a character to select from which server the data will be retrieved. See details

**Details**

server: the default server is "climmob" used for clients of <https://climmob.net/climmob3/>, other options are:

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

**Value**

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

**Author(s)**

Kauê de Sousa

**See Also**

ClimMob website <https://climmob.net/>

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

**Examples**

```
# This function only works with an API key
# the API key can be obtained once a free ClimMob account
# is created via https://climmob.net/

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

getProjectProgress(key = my_key,
                   project = "beanaru23",
                   userowner = "student",
                   server = "1000FARMS")
```

---

`getProjectsCM`*Get ClimMob projects*

---

**Description**

Fetch the status of ClimMob projects

**Usage**

```
getProjectsCM(key, server = "climmob3", ...)
```

**Arguments**

<code>key</code>	a character for the user's application programming interface (API) key
<code>server</code>	optional, a character to select from which server the data will be retrieved. See details
<code>...</code>	additional arguments passed to methods. See details

**Details**

`server`: the default server is "climmob" used for clients of <https://climmob.net/climmob3/>, other options are:

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

**Value**

A data.frame with the variables:

<code>project_id</code>	the project's id
<code>project_name</code>	the project's name
<code>user_owner</code>	the account name that owns the project
<code>country</code>	the country of project's implementation
<code>status</code>	the current status
<code>creation_date</code>	date where the project was created

**Author(s)**

Kauê de Sousa

**See Also**

ClimMob website <https://climmob.net/>

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

**Examples**

```
# This function only works with an API key
# the API key can be obtained once a free ClimMob account
# is created via https://climob.net/

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

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

---

`getTraitList`*Organise 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>	a data.frame with the ClimMob data
<code>pattern</code>	a character with 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

```

**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

**References**

van Etten J., et al. (2019). Experimental Agriculture, 55(S1), 275–296. doi:10.1017/S0014479716000739

**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

*Remove geographical identity*


---

**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 its agro-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 <i>sf</i> methods

**Value**

A data frame with the random coordinates long lat within a pre-defined 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, 6

as.data.frame.CM\_list (getDataCM), 3

ClimMobTools, 2

ClimMobTools-package (ClimMobTools), 2

getDataCM, 3, 5, 6

getProjectProgress, 4, 4, 6

getProjectsCM, 4, 5, 6

getTraitList, 7

randomise (randomize), 8

randomize, 8

rankTricot, 9

rmGeoIdentity, 11

sf, 11