

Package ‘mmstat4’

April 29, 2024

Title Access to Teaching Materials from a ZIP File or GitHub

Version 0.2.1

Description Provides access to teaching materials for various statistics courses, including R and Python programs, Shiny apps, data, and PDF/HTML documents. These materials are stored on the Internet as a ZIP file (e.g., in a GitHub repository) and can be downloaded and displayed or run locally. The content of the ZIP file is temporarily or permanently stored. By default, the package uses the GitHub repository 'sigbertklinke/mmstat4.data.' Additionally, the package includes 'association_measures.R' from the archived package 'ryouready' by Mark Heckman and some auxiliary functions.

License GPL-3

Encoding UTF-8

Imports crayon, digest, httr, knitr, tcltk, rappdirs, reticulate, rio, rstudioapi, shiny, stringdist, tools

Suggests rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.1

NeedsCompilation no

Author Sigbert Klinke [aut, cre] (<<https://orcid.org/0000-0003-3337-1863>>),
Jekaterina Zukovska [ctb] (<<https://orcid.org/0000-0002-7753-9210>>)

Maintainer Sigbert Klinke <sigbert@hu-berlin.de>

Repository CRAN

Date/Publication 2024-04-29 12:10:02 UTC

R topics documented:

askUser	2
association	3
cdf	5
checkFiles	6

defaultApp	8
dupFiles	8
getList	9
getMMstat	10
getText	10
gh	11
ghappAddin	12
ghc	12
ghdecompose	13
ghfile	14
ghget	14
ghinstall	15
ghlist	16
ghopenAddin	17
ghpath	18
ghquery	18
ghrepos	19
ghzip	20
isLocal	21
normpathes	21
note	22
openFile	22
pkglist	23
py_env	24
toInt	24
toNum	25
urlExists	25

Index **27**

askUser	<i>askUser</i>
---------	----------------

Description

askUser provides a way to ask the user a yes/no/cancel question (default). A * after a number indicates the default option.

Usage

```
askUser(
  msg,
  choices = c("Yes", "No", "Cancel"),
  default = 1,
  col = crayon::black
)
```

Arguments

<code>msg</code>	character: the prompt message for the user
<code>choices</code>	character: vector of choices (default: <code>c("Yes", "No", "Cancel")</code>)
<code>default</code>	character/integer: default option if only Enter pressed (default: 1)
<code>col</code>	function: a color function (default: <code>crayon::black</code>)

Value

the integer number chosen by the user

Examples

```
if (interactive())
  askUser("Do you want to use askUser?")
```

association

Association measures

Description

Various association coefficients for nominal and ordinal data; the input formats follows `stats::chisq.test()`.

- `concordant` concordant pairs
- `discordant` discordant pairs
- `ties.row` pairs tied on rows
- `ties.col` pairs tied on columns
- `nom.phi` Phi Coefficient
- `nom.cc` Contingency Coefficient (Pearson's C) and Sakoda's Adjusted Pearson's C
- `nom.TT` Tshuprow's T (not meaningful for non-square tables)
- `nom.CV` Cramer's V (for 2 x 2 tables $V = \text{Phi}$)
- `nom.lambda` Goodman and Kruskal's Lambda with
 - `lambda.cr` The row variable is used as independent, the column variable as dependent variable.
 - `lambda.rc` The column variable is used as independent, the row variable as dependent variable.
 - `lambda.symmetric` Symmetric Lambda (the mean of both above).
- `nom.uncertainty` Uncertainty Coefficient (Theil's U) with
 - `ucc.cr` The row variable is used as independent, the column variable as dependent variable.
 - `uc.rc` The column variable is used as independent, the row variable as dependent variable.
 - `uc.symmetric` Symmetric uncertainty coefficient.

- `ord.gamma` Gamma coefficient
- `ord.tau` a vector with Kendall-Stuart Tau's
 - `tau.a` Tau-a (for quadratic tables only)
 - `tau.b` Tau-b
 - `tau.c` Tau-c
- `ord.somers.d` Somers' d
- `eta` Eta coefficient for nominal/interval data

Usage

`concordant(x, y = NULL)`

`discordant(x, y = NULL)`

`ties.row(x, y = NULL)`

`ties.col(x, y = NULL)`

`nom.phi(x, y = NULL)`

`nom.cc(x, y = NULL)`

`nom.TT(x, y = NULL)`

`nom.CV(x, y = NULL)`

`nom.lambda(x, y = NULL)`

`nom.uncertainty(x, y = NULL)`

`ord.gamma(x, y = NULL)`

`ord.tau(x, y = NULL)`

`ord.somers.d(x, y = NULL)`

`eta(x, y, breaks = NULL)`

Arguments

- | | |
|---------------------|---|
| <code>x</code> | a numeric vector, table or matrix. <code>x</code> and <code>y</code> can also both be factors.
For <code>eta</code> the independent nominal variable (factor or numeric). |
| <code>y</code> | a numeric vector; ignored if <code>x</code> is a table or matrix. If <code>x</code> is a factor, <code>y</code> should be a factor of the same length.
For <code>eta</code> the dependent interval variable (numeric). |
| <code>breaks</code> | either a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which <code>x</code> is to be cut (only for <code>eta</code>). |

Value

the association coefficient(s)

Source

From the [archived ryouready package](#) by Mark Heckmann. The code for the calculation of `nom.lambda`, `nom.uncertainty`, `ord.gamma`, `ord.tau`, `ord.somers.d` was supplied by Marc Schwartz (under GPL 2) and checked against SPSS results.

Examples

```
## Nominal data
# remove gender from the table
hec <- apply(HairEyeColor, 1:2, sum)
nom.phi(hec)
nom.cc(hec)
nom.TT(hec)
nom.CV(hec)
nom.lambda(hec)
nom.uncertainty(hec)
## Ordinal data
# create a fake data set
ordx <- sample(5, size=100, replace=TRUE)
ordy <- sample(5, size=100, replace=TRUE)
concordant(ordx, ordy)
discordant(ordx, ordy)
ties.row(ordx, ordy)
ties.col(ordx, ordy)
ord.gamma(ordx, ordy)
ord.tau(ordx, ordy)
ord.somers.d(ordx, ordy)
## Interval/nominal data
eta(iris$Species, iris$Sepal.Length)
```

cdf

Generates and plots a cumulative distribution function.

Description

Generates and plots a cumulative distribution function.

Usage

```
cdf(x, ...)
```

Default S3 method:

```
cdf(x, y, discrete = TRUE, ...)
```

```
## S3 method for class 'cdf'
plot(x, y, ..., col.01line = "black", pch = 19)
```

Arguments

x	numeric: x-values
...	further parameters given to <code>graphics::plot()</code>
y	numeric: y-values
discrete	logical: if distribution is discrete
col.01line	color: color of horizontal lines at 0 and 1 (default: black)
pch	point type: See <code>graphics::points()</code> for possible values and their interpretation (default: 19)

Value

returns a cdf object

Examples

```
# Binomial distribution
x <- cdf(0:10, pbinom(0:10, 10, 0.5))
plot(x)
# Exponential distribution
x <- seq(0, 5, by=0.01)
x <- cdf(x, pexp(x), discrete=FALSE)
plot(x)
```

checkFiles

Checks whether all specified files are valid R or Python files

Description

checkFiles checks whether all specified files are valid source files that can be executed independently of each other. If an error occurs then:

1. If open is a function name or a function with a file parameter, then checkFiles will try to open the faulty source file, otherwise not.
2. The execution of checkFiles is stopped.

If you do not want the faulty source file to be opened immediately, use `open=0`.

Three modes are available for checking a file:

1. exist: Does the source file exist?
2. parse: (default) is `parse(file)` or `python -m "file"` successful?
3. run: is `Rscript "file"` or `python "file"` successful?

If source files has side effects, e.g. generating an image or some other output, and `mode=="parse"` then this will done during the check.

Usage

```
checkFiles(  
  files,  
  index = seq_along(files),  
  path = NULL,  
  open = openFile,  
  mode = c("parse", "run", "exist"),  
  ...  
)  
  
Rsolo(  
  files,  
  index = seq_along(files),  
  path = NULL,  
  open = openFile,  
  mode = c("parse", "run", "exist"),  
  ...  
)
```

Arguments

files	character: file name(s)
index	integer(s): if length(index)==1 the files from index to length(files) are checked (default: seq_along(files)) otherwise the files with values in index are checked.
path	character: path to start from (default: getwd())
open	function: function or function name to call after an error occurs (default: openFile)
mode	character which check to do
...	further parameters given to the function in open

Value

nothing

Examples

```
if (interactive()) {  
  files <- list.files(pattern="*(R|py)$", full.names=TRUE, recursive=TRUE)  
  checkFiles(files)  
}
```

defaultApp	<i>defaultApp</i>
------------	-------------------

Description

Tries to open the given file with the default application of the operating system using `base::system2()`. Only Windows (windows), macOS (darwin), Linux (linux) and FreeBSD (freebsd) is supported.

Usage

```
defaultApp(file, wait = FALSE, ...)
```

Arguments

file	character: file name
wait	logical: indicates whether the R interpreter should wait for the command to finish, or run it asynchronously (default: FALSE)
...	further arguments passed to system2

Value

Result of `try(system2, ...)`, invisibly

See Also

`berryFunctions::openFile()`

Examples

```
if (interactive()) {  
  ghget()  
  defaultApp(ghlist("dataanalysis.pdf", full.names = TRUE))  
}
```

dupFiles	<i>Find duplicate files</i>
----------	-----------------------------

Description

dupFiles computes checksums to find duplicate files.

Usage

```
dupFiles(files, ...)
```

```
Rdups(files, ...)
```


Arguments

`files` character: file name(s)
`...` further parameters given to `digest::digest()`

Value

a list of file names with the same checksum or NULL

Examples

```
if (interactive()) {  
  files <- list.files(pattern="*.R$", full.names=TRUE, recursive=TRUE)  
  dupFiles(files)  
}
```

*getList**getList*

Description

Creates a list with element names replaced by `link{getText}`.

Usage

```
getList(...)
```

Arguments

`...` named elements of a list

Value

renamed list

Examples

```
getList(BOSTON=1, MTCARS=2)
```

`getMMstat``getMMstat`

Description

Allows to access the package internal `mmsstat` environment.

Usage

```
getMMstat(...)
```

Arguments

... elements

Value

the choosen element

Examples

```
getMMstat('version')
```

`getText``getText`

Description

Translates a given message into another language.

Usage

```
getText(msg)
```

Arguments

msg character vector

Value

vector of translated messages

Examples

```
getText('Test')
```

Description

The function `gh` carries out the following operation on a file named `x`. It searches for a match for `x` within the active repository, utilizing fuzzy string matching. If no unique match is identified, an error is thrown along with suggestions for potential "best" matches. Otherwise, the following operation are performed:

- `gh(x, 'open')` or `ghopen(x)`: Opens a file in the local browser if the file extension is `html` or `pdf`, otherwise in the RStudio editor.
- `gh(x, 'load')` or `ghload(x)`: Loads the contents of a file with `import`.
- `gh(x, 'source')` or `ghsource(x)`: Executes the contents of a file with `source`.
- `gh(x, 'app')` or `ghapp(x)`: Tries to open the file with the default application of the OS, see [defaultApp\(\)](#).
- `ghdata(x, pkg)`: Helper function to load data sets from R packages into Python, simulates `pkg::x`.

Usage

```
gh(x, what = c("open", "load", "source", "app"), ..., .call = NULL)
```

```
ghopen(x, ...)
```

```
ghload(x, ...)
```

```
ghsource(x, ...)
```

```
ghapp(x, ...)
```

Arguments

<code>x</code>	character(1): name of the file, app or data set
<code>what</code>	character or function: a name of a predefined function or another function. The function must have a formal parameter <code>file</code> .
<code>...</code>	further parameters used in utils::browseURL() , openFile() , rio::import() , or base::source() .
<code>.call</code>	the original function call (default: <code>NULL</code>)

Value

invisibly the result of [utils::browseURL](#), [openFile\(\)](#), [rio::import\(\)](#), or [base::source\(\)](#).

Examples

```

if (interactive()) {
  x <- ghopen("bank2.SAV")
  x <- ghload("bank2.SAV")
  str(x)
  x <- ghsorce("univariate/example_ecdf.R")
}

```

ghappAddin

ghappAddin

Description

Runs a Shiny app from the downloaded zip file.

Usage

```
ghappAddin()
```

Value

nothing

Examples

```
if (interactive()) ghappAddin()
```

ghc

Creates a ghdecompose pbject

Description

ghc creates from a list of file names using [ghdecompose\(\)](#) and deletes mssing files.

Usage

```
ghc(...)
```

Arguments

... list(s) of filemaes

Value

a ghdecompose pbject

Examples

```
ghc(list.files(system.file(package="mmstat4"), recursive=TRUE))
```

ghdecompose	<i>ghdecompose</i>
-------------	--------------------

Description

Decomposes a path of a set of files (or dirs) in several parts:

Usage

```
ghdecompose(files, dirs = FALSE)
```

Arguments

files	character vector: path of files
dirs	logical: directory or files names (default: FALSE)

Details

- `outpath` the path part which is common to all files (basically the place where the ZIP file was extracted)
- `inpath` the path part which is not necessary for a unique address in the ZIP file
- `minpath` the minimal path part such that all files are addressable in a unique manner,
- `filename` the basename of the file, and
- `source` the input to `shortpath`.

Value

a data frame with five variables

Examples

```
ghget("local")  
pdf <- ghdecompose(ghlist(full.names=TRUE))  
pdf
```

ghfile	<i>ghfile</i>
--------	---------------

Description

Finds either a unique match in the list of files or throws an error with possible candidate files.

Usage

```
ghfile(x, n = 6, silent = FALSE, msg = "%s")
```

Arguments

x	character: a single file name
n	logical: if x can not be found how many best matches should be returned (default: 6)
silent	logical: if no (unique) match is found, then NULL is returned, otherwise an error is thrown (default: FALSE, throw error)
msg	character: error message how to put the file name(s (default: %s)

Value

the full matching file

Examples

```
ghfile("data/BANK2.sav")
if (interactive()) ghfile("data/BANK2.SAV") # throws an error
```

ghget	<i>ghget</i>
-------	--------------

Description

Makes a repository the active repository and downloads it if necessary. The parameter `.tempdir` is TRUE (default) then the repository is stored in the in the temporary directory `tempdir()` else in the application directory `rappdirs::user_data_dir()` for mmstat4. The parameter `.tempdir` is not logical then the value will be used as installation path.

Usage

```
ghget(..., .force = FALSE, .tempdir = TRUE, .quiet = !interactive())
```

Arguments

...	parameters to set and activate a repository
.force	logical: download and unzip in any case? (default: FALSE)
.tempdir	logical or character: store download temporary or permanently (default: <code>getOption("mmstat4.tempdir")</code>)
.quiet	logical: show repository read attempts (default: <code>!interactive()</code>) <ul style="list-style-type: none"> if <code>.tempdir==TRUE</code> then the downloaded zip file will be stored temporarily in <code>tempdir()</code> if <code>.tempdir==FALSE</code> then the downloaded zip file will be stored temporarily in <code>rappdirs::user_data_dir()</code> otherwise it is assumed that you give the name of an existing directory to store the downloaded zip file

Details

Note, the list of repository names, directories and urls is stored in the installation directory, too.

Value

the name of the current key or nothing if unsuccessful

Examples

```
if (interactive()) {
  # get one of the default ZIP file from internet
  ghget("hu.data")
  # get a locally stored zip file
  ghget(dummy2=system.file("zip", "mmstat4.dummy.zip", package="mmstat4"))
  # get from an URL
  ghget(dummy.url="https://github.com/sigbertklinke/mmstat4.dummy/archive/refs/heads/main.zip")
}
```

ghinstall

ghinstall

Description

If the user agrees, it installs additional software necessary for running a script. Currently, only `type=="py"` for Python scripts and `type=="R"` for R scripts are supported. When a repository is downloaded, `ghinstall` is called once. If the user calls `ghinstall` for an update, the parameter `force=TRUE` must be set.

Usage

```
ghinstall(type = c("py", "R"), force = FALSE)
```

Arguments

type character: programm type (default: py)
 force logical: should the installation really done (default: 'NA')

Details

R mmstat4 init_R.R is opened if present in the active repository.

py mmstat4 internally utilizes a virtual environment named mmstat4.xxxx, where xxxx, varies depending on the repository. When installis invoked, it verifies the existence of the virtual environmentmmstat4.xxxx. If it does not exist, it is opened if present in the active repository.

Value

NULL if type is not found, otherwise type

Examples

```
# to delete the virtual environment use
# reticulate::virtualenv_remove('mmstat4')
if (interactive()) ghinstall()
```

 ghlist

ghlist

Description

Returns unique (short) names for accessing each file in the repository according to a regular expression. For details about regular expressions, see [base::regex](#).

Usage

```
ghlist(
  pattern = ".",
  ignore.case = FALSE,
  perl = FALSE,
  fixed = FALSE,
  useBytes = FALSE,
  full.names = FALSE
)
```

Arguments

pattern character string containing a [regular expression](#) (or character string for fixed = TRUE) to be matched in the given character vector. Coerced by [as.character](#) to a character string if possible. If a character vector of length 2 or more is supplied, the first element is used with a warning. Missing values are allowed except for regexpr, gregexpr and regexec.

<code>ignore.case</code>	if FALSE, the pattern matching is <i>case sensitive</i> and if TRUE, case is ignored during matching.
<code>perl</code>	logical. Should Perl-compatible regexps be used?
<code>fixed</code>	logical. If TRUE, <code>pattern</code> is a string to be matched as is. Overrides all conflicting arguments.
<code>useBytes</code>	logical. If TRUE the matching is done byte-by-byte rather than character-by-character. See ‘Details’.
<code>full.names</code>	logical: should full names returned instead of short names (default: FALSE)

Value

character vector of short names

Examples

```
if (interactive()) ghlist()
```

<code>ghopenAddin</code>	<i>ghopenAddin</i>
--------------------------	--------------------

Description

A RStudio addin to open a file from the downloaded zip file.

Usage

```
ghopenAddin()
```

Value

nothing

Examples

```
if (interactive()) ghopenAddin()
```

ghpath	<i>ghpath</i>
--------	---------------

Description

Returns a path for files based on ghdecompose.

Usage

```
ghpath(df, from = c("outpath", "inpath", "minpath", "filename"))
```

Arguments

df	data frame: returned from ghdecompose
from	character: either inpath (default), outpath, minpath, or filename

Value

a character vector with file paths

Examples

```
ghget("dummy")
pdf <- ghdecompose(ghlist(full.names=TRUE))
ghpath(pdf)
ghpath(pdf, 'o') # equals the input to ghdecompose
ghpath(pdf, 'i')
ghpath(pdf, 'm')
ghpath(pdf, 'f')
```

ghquery	<i>ghquery</i>
---------	----------------

Description

Queries the unique (short) names for each file in the repository. Several query methods are available, see Details.

Usage

```
ghquery(
  query,
  n = 6,
  full.names = FALSE,
  method = c("fpdist", "overlap", "tfidf"),
  costs = NULL,
```

```

    counts = FALSE,
    useBytes = FALSE
  )

```

Arguments

query	character: query string
n	integer: maximal number of matches to return
full.names	logical: should full names used instead of short names (default: FALSE)
method	character: method to be used (default: fpdist)
costs	a numeric vector or list with names partially matching ‘insertions’, ‘deletions’ and ‘substitutions’ giving the respective costs for computing the Levenshtein distance, or NULL (default) indicating using unit cost for all three possible transformations.
counts	a logical indicating whether to optionally return the transformation counts (numbers of insertions, deletions and substitutions) as the “counts” attribute of the return value.
useBytes	a logical. If TRUE distance computations are done byte-by-byte rather than character-by-character.

Details

The following query methods are available:

- fpdist uses a partial backward matching distance based on `utils::adist()`
- overlap uses the **overlap distance** for query and file names

Value

character vector of short names fitting best to the query

Examples

```
if (interactive()) ghquery("bank")
```

ghrepos

ghrepos

Description

If key is NULL, then it returns the known repositories and where they are stored. If key is not NULL, then possible addresses for a repository are returned .

Usage

```
ghrepos(key = NULL)
```

Arguments

key character: "name" of the repository to find (default: NULL)

Value

a data frame with the data about the repositories

Examples

```
ghrepos()
```

ghzip	<i>Creates a ZIP file or directory with files</i>
-------	---

Description

ghzip creates a ZIP file (if `dest` has an extension zip) or copies to the destination directory. If `dest` is NULL then a temporary directory will be used. Please note that neither the ZIP file is deleted nor the target directory is cleaned beforehand if it already exists.

Usage

```
ghzip(files, dest = NULL)
```

Arguments

files ghdecompose object or character: list of files to copy
 dest character: ZIP file name of destination directory (default: NULL)

Value

the name of the destination directory or the ZIP file

Examples

```
if (interactive()) {
  zipfile <- tempfile(fileext='.zip')
  files <- list.files(system.file(package="mmstat4"), recursive=TRUE)
  ghzip(files, zipfile)
}
```

<code>isLocal</code>	<i>isLocal</i>
----------------------	----------------

Description

Checks if a Shiny app runs locally or on a server

Usage

```
isLocal()
```

Value

logical

Examples

```
isLocal()
```

<code>normpathes</code>	<i>normpathes</i>
-------------------------	-------------------

Description

Returns a list with normalized pathes.

Usage

```
normpathes(x)
```

Arguments

`x` file pathes

Value

A list of the same length as `x`, the *i*-th element of which contains the vector of splits of `x[i]`.

Examples

```
normpathes("CRAN/./mmstat4/python/./ghdist.R")
```

note	<i>Create and display a note</i>
------	----------------------------------

Description

note internally stores a colored message, while display utilizes `base::cat()` to present them and reset the internal message stack.

Usage

```
note(msg, col = crayon::green)

display()
```

Arguments

msg	character: message
col	function: a color function (default: <code>crayon::green</code>)

Value

note returns invisibly the number of notes

Examples

```
notetest <- function(msg) {
  on.exit({ display() })
  note(msg)
  # do some complex computation
  x <- 1+1
}
notetest("Hello world!")
```

openFile	<i>openFile</i>
----------	-----------------

Description

The function attempts to open a file either in RStudio or in a text editor, depending on the environment. If the session is interactive, it tries to open the file in RStudio using `rstudioapi::navigateToFile()`. If RStudio is not available or the attempt fails, it opens the file in a text editor using `utils::edit()`. If the session is not interactive, it simply returns the contents of the file.

Usage

```
openFile(file, ...)
```

Arguments

file character: name of the file
 ... further parameters give to `rstudioapi::navigateToFile()` or `utils::edit()`

Value

invisibly the result from `try(rstudioapi::navigateToFile(file))` or `try(utils::edit(file))`.

Examples

```
openFile(system.file("rstudio", "addins.dcf", package = "mmstat4"))
```

pkglist	<i>Extract library and require calls in R and import calls from Python</i>
---------	--

Description

`pkglist` counts the number of `library/require/import` calls for R and Python commands in the files. If you set `code==TRUE` then returns R code for installing packages/modules. It checks via `utils::available.packages()` (for R) and via PyPI if a package/module is available. Otherwise a table with the number of library or import is returned.

Usage

```
pkglist(files, code = TRUE, repos = getOption("repos"))
```

```
Rlibs(files, code = TRUE, repos = getOption("repos"))
```

```
modlist(files, code = TRUE, repos = getOption("repos"))
```

Arguments

files character: file name(s)
 code logical: should names given back or code for init scrips? (default: TRUE)
 repos character: the base URL(s) of the repositories to use (default: `getOption("repos")`)

Value

a table how frequently the packages are called or R Code to install them

Examples

```
if (interactive()) {
  files <- list.files(pattern="*(R|py)", full.names=TRUE, recursive=TRUE)
  pkglist(files)
}
```

py_env	<i>py_env</i>
--------	---------------

Description

Name of the currently used virtual environment.

Usage

```
py_env()
```

Value

the name of the virtual Python environment currently used by mmsstat4

Examples

```
py_env()
```

toInt	<i>toInt</i>
-------	--------------

Description

Converts `x` to an integer. If the conversion fails or the integer is outside `min` and `max` then `NA_integer_` is returned

Usage

```
toInt(x, min = -Inf, max = +Inf)
```

Arguments

<code>x</code>	input object
<code>min</code>	numeric: minimal value
<code>max</code>	numeric: maximal value

Value

a single integer value

Examples

```
toInt(3.0)
toInt("3.0")
toInt("test")
```

toNum	<i>toNum</i>
-------	--------------

Description

Converts `x` to a numeric. If the conversion fails or the value is outside `min` and `max` then `NA` is returned

Usage

```
toNum(x, min = -Inf, max = +Inf)
```

Arguments

<code>x</code>	input object
<code>min</code>	numeric: minimal value
<code>max</code>	numeric: maximal value

Value

a single integer value

Examples

```
toNum(3.0)
toNum("3.0")
toNum("test")
```

urlExists	<i>urlExists</i>
-----------	------------------

Description

Verifies whether a provided `url` is downloadable, without detecting redirections in the URL.

Usage

```
urlExists(url)
```

Arguments

<code>url</code>	a vector of text URLs
------------------	-----------------------

Value

TRUE if URL exists otherwise FALSE

Examples

```
urlExists("https://hu-berlin.de/sk")  
urlExists("https://huglawurza.de")
```

Index

as.character, [16](#)
askUser, [2](#)
association, [3](#)

base::cat(), [22](#)
base::regex, [16](#)
base::source(), [11](#)
base::system2(), [8](#)

cdf, [5](#)
checkFiles, [6](#)
concordant (association), [3](#)

defaultApp, [8](#)
defaultApp(), [11](#)
digest::digest(), [9](#)
discordant (association), [3](#)
display (note), [22](#)
dupFiles, [8](#)

eta (association), [3](#)

getList, [9](#)
getMMstat, [10](#)
getText, [10](#)
gh, [11](#)
ghapp (gh), [11](#)
ghappAddin, [12](#)
ghc, [12](#)
ghdata (gh), [11](#)
ghdecompose, [13](#)
ghdecompose(), [12](#)
ghfile, [14](#)
ghget, [14](#)
ghinstall, [15](#)
ghlist, [16](#)
ghload (gh), [11](#)
ghopen (gh), [11](#)
ghopenAddin, [17](#)
ghpath, [18](#)
ghquery, [18](#)

ghrepos, [19](#)
ghsource (gh), [11](#)
ghzip, [20](#)
graphics::plot(), [6](#)
graphics::points(), [6](#)

isLocal, [21](#)

modlist (pkglist), [23](#)
modules (pkglist), [23](#)

nom.cc (association), [3](#)
nom.CV (association), [3](#)
nom.lamba (association), [3](#)
nom.lambda (association), [3](#)
nom.phi (association), [3](#)
nom.TT (association), [3](#)
nom.uncertainty (association), [3](#)
normpaths, [21](#)
note, [22](#)

openFile, [22](#)
openFile(), [11](#)
ord.gamma (association), [3](#)
ord.somers.d (association), [3](#)
ord.tau (association), [3](#)

pkglist, [23](#)
plot.cdf (cdf), [5](#)
py_env, [24](#)

rappdirs::user_data_dir(), [14](#), [15](#)
Rdups (dupFiles), [8](#)
regular expression, [16](#)
rio::import(), [11](#)
Rlibs (pkglist), [23](#)
Rsolo (checkFiles), [6](#)
rstudioapi::navigateToFile(), [22](#), [23](#)

stats::chisq.test(), [3](#)

`tempdir()`, [14](#), [15](#)
`ties.col` (association), [3](#)
`ties.row` (association), [3](#)
`toInt`, [24](#)
`toNum`, [25](#)

`urlExists`, [25](#)
`utils::adist()`, [19](#)
`utils::available.packages()`, [23](#)
`utils::browseURL`, [11](#)
`utils::browseURL()`, [11](#)
`utils::edit()`, [22](#), [23](#)