Package 'icesTAF'

August 3, 2018

Version 1.6-2

Date 2018-08-03

Title Functions to Support the ICES Transparent Assessment Framework

Imports grDevices, stats, tools, utils

Suggests icesAdvice

LazyData yes

Description Functions to support the ICES Transparent Assessment Framework <http://taf.ices.dk> to organize data, methods, and results used in ICES assessments. ICES is an organization facilitating international collaboration in marine science.

License GPL (>= 2)

URL http://taf.ices.dk

RoxygenNote 6.1.0

NeedsCompilation no

Author Arni Magnusson [aut, cre], Colin Millar [aut]

Maintainer Arni Magnusson <arni.magnusson@ices.dk>

Repository CRAN

Date/Publication 2018-08-03 10:10:03 UTC

R topics documented:

icesTAF-package 2
catage.long 4
catage.taf 5
catage.xtab 6
clean 7
cp 7
deps 9
div 10
dos2unix 11

download . . . 12
flr2taf . . . 13
lim . . . 14
long2taf . . . 15
make . . . 15
makeAll . . . 17
makeTAF . . . 18
mkdir . . . 19
msg . . . 20
plus . . . 20
read.dls . . . 21
read.taf . . . 22
rnd . . . 23
sourceAll . . . 24
sourceTAF . . . 25
summary.taf . . . 26
taf.colors . . . 27
taf.skeleton . . . 28
taf2long . . . 29
taf2xtab . . . 30
tafpng . . . 31
tt . . . 32
write.dls . . . 33
write.taf . . . 34
xtab2taf . . . 35

Index **37**

`icesTAF-package` *Functions to Support the ICES Transparent Assessment Framework*

Description

Functions to support the ICES Transparent Assessment Framework, to organize data, methods, and results used in ICES assessments.

Details

Web services:

`download`	download file in binary mode

Read and write files:

`read.dls`	read `DLS3.2` results from file
`read.taf`	read TAF table from file
`write.dls`	write `DLS3.2` results to file
`write.taf`	write TAF table to file

Run scripts:

make	run R script if needed
makeAll	run all TAF scripts as needed
makeTAF	run TAF script if needed
msg	show message
sourceAll	run all TAF scripts
sourceTAF	run TAF script

Other file management:

clean	clean TAF directories
cp	copy files
dos2unix	convert line endings
mkdir	create directory
taf.skeleton	create empty TAF template
unix2dos	convert line endings

Table tools:

div	divide column values
flr2taf	convert FLR to TAF
long2taf	convert long format to TAF
plus	rename plus group column
rnd	round column values
taf2long	convert TAF to long format
taf2xtab	convert TAF to crosstab
tt	transpose TAF table
xtab2taf	convert crosstab to TAF

Plotting tools:

lim	compute axis limits
taf.colors	predefined colors
tafpng	open PNG graphics device

Example tables:

catage.long	long format
catage.taf	TAF format
catage.xtab	crosstab format
summary.taf	summary results

Administrative tools:

deps	list dependencies

Author(s)

Arni Magnusson and Colin Millar.

References

ICES Transparent Assessment Framework: http://taf.ices.dk.

To explore example TAF stock assessments, see the introductory video and tutorial.

`catage.long` *Catch at Age in Long Format*

Description

Small catch-at-age table to describe a long format data frame to store year-age values.

Usage

```
catage.long
```

Format

Data frame containing three columns:

`Year`	year
`Age`	age
`Catch`	catch (millions of individuals)

Details

The data are an excerpt (first years and ages) from the catch-at-age table for North Sea cod from the ICES (2016) assessment.

Source

ICES (2016) Report of the working group on the assessment of demersal stocks in the North Sea and Skagerrak (WGNSSK). *ICES CM 2016/ACOM:14*, p. 656.

See Also

`catage.taf` and `catage.xtab` describe alternative table formats.

`long2taf` converts a long table to TAF format.

`icesTAF-package` gives an overview of the package.

Examples

```
catage.long
long2taf(catage.long)
```

`catage.taf` *Catch at Age in TAF Format*

Description

Small catch-at-age table to describe a TAF format data frame to store year-age values.

Usage

```
catage.taf
```

Format

Data frame containing five columns:

`Year`	year
1	number of one-year-olds in the catch (millions)
2	number of two-year-olds in the catch (millions)
3	number of three-year-olds in the catch (millions)
4	number of four-year-olds in the catch (millions)

Details

The data are an excerpt (first years and ages) from the catch-at-age table for North Sea cod from the ICES (2016) assessment.

Source

ICES (2016) Report of the working group on the assessment of demersal stocks in the North Sea and Skagerrak (WGNSSK). *ICES CM 2016/ACOM:14*, p. 656.

See Also

`catage.long` and `catage.xtab` describe alternative table formats.

`taf2long` and `taf2xtab` convert a TAF table to alternative formats.

`icesTAF-package` gives an overview of the package.

Examples

```
catage.taf
taf2long(catage.taf)
taf2xtab(catage.taf)
```

`catage.xtab` *Catch at Age in Crosstab Format*

Description

Small catch-at-age table to describe a crosstab format data frame to store year-age values.

Usage

```
catage.xtab
```

Format

Data frame with years as row names and containing four columns:

1	number of one-year-olds in the catch (millions)
2	number of two-year-olds in the catch (millions)
3	number of three-year-olds in the catch (millions)
4	number of four-year-olds in the catch (millions)

Details

The data are an excerpt (first years and ages) from the catch-at-age table for North Sea cod from the ICES (2016) assessment.

Source

ICES (2016) Report of the working group on the assessment of demersal stocks in the North Sea and Skagerrak (WGNSSK). *ICES CM 2016/ACOM:14*, p. 656.

See Also

`catage.long` and `catage.taf` describe alternative table formats.

`xtab2taf` converts a crosstab table to TAF format.

`icesTAF-package` gives an overview of the package.

Examples

```
catage.xtab
xtab2taf(catage.xtab)
```

clean *Clean TAF Directories*

Description

Remove TAF directories: data, input, model, output, report.

Usage

```
clean(dirs = c("data", "input", "model", "output", "report"))
```

Arguments

dirs	directories to delete.

Note

The purpose of removing the directories is to make sure that subsequent TAF scripts start by creating new empty directories.

See Also

sourceTAF runs a TAF script.

sourceAll runs all TAF scripts in a directory.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
clean()

## End(Not run)
```

cp *Copy Files*

Description

Copy or move files, overwriting existing files if necessary, and returning the result invisibly.

Usage

```
cp(from, to, move = FALSE)
```

Arguments

from	source filenames, e.g. *.csv.
to	destination filenames, or directory.
move	whether to move instead of copy.

Value

TRUE for success, FALSE for failure, invisibly.

Note

To prevent accidental loss of files, two safeguards are enforced when move = TRUE:

1. When moving files, the to argument must either have a filename extension or be an existing directory.
2. When moving many files to one destination, the to argument must be an existing directory.

If these conditions do not hold, no files are changed and an error is returned.

See Also

file.copy and unlink are the underlying functions used to copy and (if move = TRUE) delete files.

file.rename is the base function to rename files.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
write(pi, "A.txt")
cp("A.txt", "B.txt")
cp("A.txt", "B.txt", move=TRUE)

## Copy directory tree
cp(system.file(package="datasets"), ".")
mkdir("everything")
cp("datasets/*", "everything")

## End(Not run)
```

deps *List Dependencies*

Description

Search R scripts for packages that are required.

Usage

```
deps(path = ".", base = FALSE, installed = TRUE, available = TRUE,
  list = FALSE)
```

Arguments

path	a directory or file containing R scripts.
base	whether to include base packages in the output.
installed	whether to include installed packages in the output.
available	whether to include available packages in the output.
list	whether to return packages in list format.

Value

Names of packages as a vector, or in list format if `list=TRUE`. If no dependencies are found, the return value is `NULL`.

Note

Package names are matched based on four patterns:

```
library(*)
require(*)
*::object
*:::object
```

The search algorithm may return false-positive dependencies if these patterns occur inside if-clauses, strings, comments, etc.

See Also

`installed.packages`, `available.packages`.

`icesTAF-package` gives an overview of the package.

Examples

```
dir <- system.file(package="MASS", "scripts")
script <- system.file(package="MASS", "scripts/ch08.R")

deps(script)                   # dependencies
deps(script, base=TRUE)        # including base packages
deps(script, installed=FALSE)  # not (yet) installed

deps(dir)
deps(dir, list=TRUE)

## Not run:
deps(dir, available=FALSE)  # dependencies that might be unavailable

## End(Not run)
```

---

div *Divide Columns*

Description

Divide column values in a data frame with a common number.

Usage

```
div(x, cols, by = 1000, grep = FALSE, ...)
```

Arguments

x	a data frame.
cols	column names, or column indices.
by	a number to divide with.
grep	whether cols is a regular expression.
...	passed to grep().

Value

A data frame similar to x, after dividing columns cols by the number by.

Note

Provides notation that is convenient for modifying many columns at once.

See Also

transform can also be used to recalculate column values, using a more general and verbose syntax.

grep is the underlying function used to match column names if grep is TRUE.

rnd is a similar function that rounds columns.

icesTAF-package gives an overview of the package.

Examples

```
# These are equivalent:

x1 <- div(summary.taf, c("Rec","Rec_lo","Rec_hi",
                         "TSB","TSB_lo","TSB_hi",
                         "SSB","SSB_lo","SSB_hi",
                         "Removals","Removals_lo","Removals_hi"))

x2 <- div(summary.taf, "Rec|TSB|SSB|Removals", grep=TRUE)

x3 <- div(summary.taf, "Year|Fbar", grep=TRUE, invert=TRUE)

# Less reliable in scripts if columns have been added/deleted/reordered:

x4 <- div(summary.taf, 2:13)
```

---

dos2unix *Convert Line Endings*

Description

Convert line endings in a text file between Dos (CRLF) and Unix (LF) format.

Usage

```
dos2unix(file)

unix2dos(file)
```

Arguments

file	a filename.

Note

TAF uses unix2dos to ensure that text files on the server have Dos line endings.

See Also

file is used to open a binary connection to the text file.

writeLines is used to apply CRLF or LF line endings.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
file <- "test.txt"
write("123", file)

dos2unix(file)
file.size(file)

unix2dos(file)
file.size(file)

file.remove(file)

## End(Not run)
```

---

download *Download File in Binary Mode*

Description

Download a file in binary mode, e.g. a model executable.

Usage

```
download(url, dir = ".", mode = "wb", chmod = file_ext(url) == "",
  quiet = TRUE, ...)
```

Arguments

url	URL of file to download.
dir	directory to download to.
mode	download mode, see details.
chmod	whether to set execute permission (default is TRUE if file has no filename extension).
quiet	whether to suppress messages.
...	passed to download.file.

Details

With the default mode "wb" the file is downloaded in binary mode (see download.file), to prevent R from adding ^M at line ends. This is particularly relevant for Windows model executables, while the chmod switch is useful when downloading Linux executables.

This function can be convenient for downloading any file, including text files. Data files in CSV or other text format can also be read directly into memory using read.table, read.taf or similar functions, without writing to the file system.

See Also

read.taf reads a TAF table into a data frame.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
url <- paste0("https://github.com/ices-taf/2015_had-iceg/raw/master/",
              "begin/model/catageysa.exe")
download(url)

## End(Not run)
```

---

flr2taf *Convert FLR Table to TAF Format*

Description

Convert a table from FLR format to TAF format.

Usage

```
flr2taf(x, colname = "Value")
```

Arguments

x	a table of class FLQuant.
colname	a column name to use if the FLR table contains only one row.

Value

A data frame in TAF format.

Note

FLR uses the FLQuant class to store tables as 6-dimensional arrays, while TAF tables are stored as data frames with a year column.

See Also

catage.taf describes the TAF format.

as.data.frame is a method provided by the **FLCore** package to convert FLQuant tables to a 7-column long format.

icesTAF-package gives an overview of the package.

Examples

```
x <- array(t(catage.xtab), dim=c(4,8,1,1,1,1))
dimnames(x) <- list(age=1:4, year=1963:1970,
                    unit="unique", season="all", area="unique", iter=1)
flr2taf(x)

x1 <- x[1,,,,,,drop=FALSE]
flr2taf(x1)
flr2taf(x1, "Juveniles")
```

---

lim *Axis Limits*

Description

Compute axis limits. The lower limit is 0 and the upper limit is determined by the highest data value, times a multiplier.

Usage

```
lim(x, mult = 1.1)
```

Arguments

x	a vector of data values.
mult	a number to multiply with the highest data value.

Value

A vector of length two, which can be used as axis limits.

See Also

icesTAF-package gives an overview of the package.

Examples

```
plot(rivers, ylim=lim(rivers))
```

long2taf *Convert Long Table to TAF Format*

Description

Convert a table from long format to TAF format.

Usage

```
long2taf(x)
```

Arguments

x	a data frame in long format.

Value

A data frame in TAF format.

Note

TAF stores tables as data frames with a year column, as seen in stock assessment reports. The long format is more convenient for analysis and producing plots.

See Also

catage.long and catage.taf describe the long and TAF formats.

taf2long converts a TAF table to long format.

icesTAF-package gives an overview of the package.

Examples

```
long2taf(catage.long)
```

make *Run R Script If Needed*

Description

Run an R script if underlying files have changed, otherwise do nothing.

Usage

```
make(recipe, prereq, target, include = TRUE, engine = source,
  debug = FALSE, ...)
```

Arguments

recipe	script filename.
prereq	one or more underlying files, required by the script. For example, data files and/or scripts.
target	one or more output files, produced by the script. Directory names can also be used.
include	whether to automatically include the script itself as a prerequisite file.
engine	function to source the script.
debug	whether to show a diagnostic table of files and time last modified.
...	passed to engine.

Value

TRUE or FALSE, indicating whether the script was run.

Note

This function provides functionality similar to makefile rules, to determine whether a script should be (re)run or not.

If any target is missing or older than any prereq, then the script is run.

References

Stallman, R. M. *et al.* An introduction to makefiles. Chapter 2 in the *GNU Make manual*.

See Also

source runs any R script, sourceTAF is more convenient for running a TAF script, and sourceAll runs all TAF scripts.

make, makeTAF, and makeAll are similar to the source functions, except they avoid repeating tasks that have already been run.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
make("model.R", "input/input.dat", "model/results.dat")

## End(Not run)
```

makeAll *Run All TAF Scripts as Needed*

Description

Run core TAF scripts that have changed, or if previous steps were rerun.

Usage

```
makeAll(...)
```

Arguments

...	passed to makeTAF.

Value

Logical vector indicating which scripts were run.

Note

TAF scripts that will be run as needed: data.R, input.R, model.R, output.R, and report.R.

If a begin.R script exists, it is ignored.

See Also

source runs any R script, sourceTAF is more convenient for running a TAF script, and sourceAll runs all TAF scripts.

make, makeTAF, and makeAll are similar to the source functions, except they avoid repeating tasks that have already been run.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
makeAll()

## End(Not run)
```

makeTAF *Run TAF Script If Needed*

Description

Run a TAF script if the target directory is either older than the script, or older than the directory of the previous TAF step.

Usage

```
makeTAF(script, ...)
```

Arguments

script	TAF script filename.
...	passed to make and sourceTAF.

Value

TRUE or FALSE, indicating whether the script was run.

Note

Any underlying scripts are automatically included if they share the same filename prefix, followed by an underscore. For example, when determining whether a script data.R should be run, this function checks whether data_foo.R and data_bar.R have been recently modified.

See Also

source runs any R script, sourceTAF is more convenient for running a TAF script, and sourceAll runs all TAF scripts.

make, makeTAF, and makeAll are similar to the source functions, except they avoid repeating tasks that have already been run.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
makeTAF("model.R")

## End(Not run)
```

mkdir *Create Directory*

Description

Create directory, including parent directories if necessary, without generating a warning if the directory already exists.

Usage

```
mkdir(path)
```

Arguments

path	a directory name.

Value

TRUE for success, FALSE for failure, invisibly.

See Also

dir.create is the base function to create an empty directory.

unlink with recursive = TRUE removes directories.

clean cleans TAF directories.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
mkdir("emptydir")
unlink("emptydir", recursive=TRUE)

mkdir("outer/inner")
unlink("outer/inner", recursive=TRUE)

## End(Not run)
```

msg *Show Message*

Description

Show a message, as well as the current time.

Usage

```
msg(...)
```

Arguments

...	passed to message.

See Also

message is the base function to show messages, without the current time.

sourceTAF reports progress using msg.

icesTAF-package gives an overview of the package.

Examples

```
msg("script.R running...")
```

plus *Rename Plus Group Column*

Description

Rename the last column in a data frame, by appending a "+" character. This is useful if the last column is a plus group.

Usage

```
plus(x)
```

Arguments

x	a data frame.

Value

A data frame similar to x, after renaming the last column.

See Also

names is the underlying function to rename columns.

icesTAF-package gives an overview of the package.

Examples

```
catage <- catage.taf

# Rename last column
catage <- plus(catage)

# Shorter and less error-prone than
names(catage)[names(catage)=="4"] <- "4+"
```

---

read.dls *Read DLS3.2 Results from File*

Description

Read results from the DLS3.2 advisory method from a file into a list.

Usage

```
read.dls(file)
```

Arguments

file	a filename.

Value

A list containing advice and other elements showing intermediate steps in the calculations.

See Also

write.dls writes DLS3.2 results to a file.

DLS3.2 in the **icesAdvice** package can be used to calculate catch advice for data-limited stocks (DLS).

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
survey <- data.frame(year=2001:2010, randu[1:10,])
dls <- icesAdvice::DLS3.2(1000, survey$y)

write.dls(dls, "dls.txt")
read.dls("dls.txt")

file.remove("dls.txt")

## End(Not run)
```

`read.taf` *Read TAF Table from File*

Description

Read a TAF table from a file into a data frame.

Usage

```
read.taf(file, check.names = FALSE, stringsAsFactors = FALSE,
  fileEncoding = "UTF-8", ...)
```

Arguments

`file`	a filename.
`check.names`	whether to enforce regular column names, e.g. convert column name '`"3"`' to '`"X3"`'.
`stringsAsFactors`	whether to import strings as factors.
`fileEncoding`	character encoding of input file.
`...`	passed to `read.csv`.

Details

Alternatively, `file` can be a directory or a vector of filenames, to read many tables in one call.

Value

A data frame in TAF format, or a list of data frames if `file` is a directory or a vector of filenames.

See Also

`read.csv` is the underlying function used to read a table from a file.

`write.taf` writes a TAF table to a file.

`icesTAF-package` gives an overview of the package.

Examples

```
## Not run:
write.taf(catage.taf, "catage.csv")
catage <- read.taf("catage.csv")

write.taf(catage)
file.remove("catage.csv")

## End(Not run)
```

---

rnd *Round Columns*

Description

Round column values in a data frame.

Usage

```
rnd(x, cols, digits = 0, grep = FALSE, ...)
```

Arguments

x	a data frame.
cols	column names, or column indices.
digits	number of decimal places.
grep	whether cols is a regular expression.
...	passed to grep().

Value

A data frame similar to x, after rounding columns cols to the number of digits.

Note

Provides notation that is convenient for modifying many columns at once.

See Also

round is the underlying function used to round numbers.

grep is the underlying function used to match column names if grep is TRUE.

div is a similar function that divides columns with a common number.

icesTAF-package gives an overview of the package.

The **icesAdvice** package provides the icesRound function to round values for ICES advice sheets.

Examples

```
# With rnd() we no longer need to repeat the column names:

m <- mtcars
m[c("mpg","disp","qsec")] <- round(m[c("mpg","disp","qsec")])
m <- rnd(m, c("mpg","disp","qsec"))

# The x1/x2/x3/x4 approaches are equivalent:

x1 <- rnd(summary.taf, c("Rec","Rec_lo","Rec_hi",
                         "TSB","TSB_lo","TSB_hi",
                         "SSB","SSB_lo","SSB_hi",
                         "Removals","Removals_lo","Removals_hi"))
x1 <- rnd(x1, c("Fbar","Fbar_lo","Fbar_hi"), 3)

x2 <- rnd(summary.taf, "Rec|TSB|SSB|Removals", grep=TRUE)
x2 <- rnd(x2, "Fbar", 3, grep=TRUE)

x3 <- rnd(summary.taf, "Fbar", grep=TRUE, invert=TRUE)
x3 <- rnd(x3, "Fbar", 3, grep=TRUE)

# Less reliable in scripts if columns have been added/deleted/reordered:

x4 <- rnd(summary.taf, 2:13)
x4 <- rnd(x4, 14:16, 3)
```

`sourceAll` *Run All TAF Scripts*

Description

Run core TAF scripts in current directory.

Usage

```
sourceAll(...)
```

Arguments

`...`	passed to `sourceTAF`.

Value

Logical vector, indicating which scripts ran without errors.

Note

TAF scripts that will be run if they exist: data.R, input.R, model.R, output.R, and report.R.

If a begin.R script exists, it is ignored.

See Also

sourceTAF runs a TAF script.

makeAll runs all TAF scripts as needed.

clean cleans TAF directories.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
sourceAll()

## End(Not run)
```

---

sourceTAF *Run TAF Script*

Description

Run a TAF script and return to the original directory.

Usage

```
sourceTAF(script, rm = FALSE, clean = TRUE, quiet = FALSE)
```

Arguments

script	script filename.
rm	whether to remove all objects from the global environment before and after the script is run.
clean	whether to remove the target directory before running the script.
quiet	whether to suppress messages reporting progress.

Details

The default value of `rm = FALSE` is to protect users from accidental loss of work, but the TAF server always runs with `rm = TRUE` to make sure that only files, not objects, are carried over between scripts.

Likewise, the TAF server runs with `clean = TRUE` to make sure that the script starts by creating a new empty directory. The target directory of a TAF script has the same filename prefix as the script: data.R creates 'data' etc. An important exception is that a directory called 'begin' will never be deleted.

Value

TRUE or FALSE, indicating whether the script ran without errors.

Note

Commands within a script (such as `setwd`) may change the working directory, but `sourceTAF` guarantees that after running a script, the working directory reported by `getwd()` is the same before and after running a script.

See Also

`source` is the base function to run R scripts.

`makeTAF` runs a TAF script if needed.

`sourceAll` runs all TAF scripts in a directory.

`icesTAF-package` gives an overview of the package.

Examples

```
## Not run:
write("print(pi)", "script.R")
source("script.R")
sourceTAF("script.R")
file.remove("script.R")

## End(Not run)
```

---

summary.taf *Summary Results in TAF Format*

Description

Small summary results table to describe a TAF format data frame to store values by year.

Usage

```
summary.taf
```

Format

Data frame containing 16 columns:

`Year`	year
`Rec`	recruitment, numbers at age 1 in this year (thousands)
`Rec_lo`	lower 95% confidence limit
`Rec_hi`	upper 95% confidence limit
`TSB`	total stock biomass (tonnes)
`TSB_lo`	lower 95% confidence limit
`TSB_hi`	upper 95% confidence limit
`SSB`	spawning stock biomass (tonnes)
`SSB_lo`	lower 95% confidence limit

SSB_hi	upper 95% confidence limit
Removals	total removals, including catches due to unaccounted mortality
Removals_lo	lower 95% confidence limit
Removals_hi	upper 95% confidence limit
Fbar	average fishing mortality (ages 2-4)
Fbar_lo	lower 95% confidence limit
Fbar_hi	upper 95% confidence limit

Details

The data are an excerpt (first years) from the summary results table for North Sea cod from the ICES (2016) assessment.

Source

ICES (2016) Report of the working group on the assessment of demersal stocks in the North Sea and Skagerrak (WGNSSK). *ICES CM 2016/ACOM:14*, p. 673.

See Also

div and rnd can modify a large number of columns.

icesTAF-package gives an overview of the package.

Examples

```
summary.taf
x <- div(summary.taf, "Rec|TSB|SSB|Removals", grep=TRUE)
x <- rnd(x, "Rec|TSB|SSB|Removals", grep=TRUE)
x <- rnd(x, "Fbar", 3, grep=TRUE)
```

---

taf.colors *TAF Colors*

Description

Predefined colors for TAF plots.

Usage

```
taf.green
taf.orange
taf.blue
taf.dark
taf.light
```

See Also

icesTAF-package gives an overview of the package.

Examples

```
taf.green
barplot(5:1, col=c(taf.green, taf.orange, taf.blue, taf.dark, taf.light))
```

`taf.skeleton` *TAF Skeleton*

Description

Create an empty template for a new TAF analysis: initial directories and R scripts with TAF header comments.

Usage

```
taf.skeleton(name = "analysis", path = ".", force = FALSE)
```

Arguments

`name`	main directory name for the analysis.
`path`	path to create the analysis directory in.
`force`	whether to overwrite an existing directory.

Details

Use `name = "."` to create initial directories and scripts inside the current working directory.

Value

Full path to analysis directory.

See Also

`package.skeleton` creates an empty template for a new R package.

`icesTAF-package` gives an overview of the package.

Examples

```
## Not run:
taf.skeleton()

## End(Not run)
```

taf2long *Convert TAF Table to Long Format*

Description

Convert a table from TAF format to long format.

Usage

```
taf2long(x, names = c("Year", "Age", "Value"))
```

Arguments

x	a data frame in TAF format.
names	a vector of three column names for the resulting data frame.

Value

A data frame with three columns.

Note

TAF stores tables as data frames with a year column, as seen in stock assessment reports. The long format is more convenient for analysis and producing plots.

See Also

catage.long and catage.taf describe the long and TAF formats.

long2taf converts a long table to TAF format.

icesTAF-package gives an overview of the package.

Examples

```
taf2long(catage.taf, names=c("Year","Age","Catch"))
```

taf2xtab *Convert TAF Table to Crosstab Format*

Description

Convert a table from TAF format to crosstab format.

Usage

```
taf2xtab(x)
```

Arguments

x	a data frame in TAF format.

Value

A data frame with years as row names.

Note

TAF stores tables as data frames with a year column, as seen in stock assessment reports. The crosstab format can be more convenient for analysis and producing plots.

See Also

catage.taf and catage.xtab describe the TAF and crosstab formats.

tt converts a TAF table to transposed crosstab format.

xtab2taf converts a crosstab table to TAF format.

icesTAF-package gives an overview of the package.

Examples

```
taf2xtab(catage.taf)
```

tafpng *PNG Device*

Description

Open PNG graphics device to create an image file inside the TAF report folder.

Usage

```
tafpng(filename, width = 800, height = 600, pointsize = 22, ...)
```

Arguments

filename	image filename.
width	image width.
height	image height.
pointsize	text size.
...	passed to png.

Details

The filename can be passed without the preceding "report/", and without the ".png" filename extension.

Specifically, the function prepends "report/" to the filename if (1) the filename does not contain a "/" separator, (2) the working directory is not report, and (3) the directory report exists. The function also appends ".png" to the filename if it does not already have that filename extension.

This automatic filename manipulation can be bypassed by using the png function directly.

Note

A simple convenience function to shorten

```
png("report/myplot.png", width=800, height=600, pointsize=22)
```

to

```
tafpng("myplot")
```

To successfully export trellis and ggplot objects to image files in scripts, these objects should be explicitly printed (see examples below).

See Also

png is the underlying function used to open a PNG graphics device.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
tafpng("myplot")
plot(1)
dev.off()

library(lattice)
tafpng("mytrellis")
print(xyplot(1~1))
dev.off()

library(ggplot2)
tafpng("myggplot")
print(qplot(1))
dev.off()

## End(Not run)
```

---

tt *TAF Transpose*

Description

Convert a table from TAF format to transposed crosstab format.

Usage

```
tt(x, column = FALSE)
```

Arguments

x	a data frame in TAF format.
column	a logical indicating whether the group names should be stored in a column called 'Age' instead of in row names. Alternatively, column can be a string supplying another name for that first column.

Value

A data frame with years as column names.

Note

Transposing can be useful when comparing TAF tables to stock assessment reports.

See Also

catage.taf describes the TAF format.

taf2xtab converts a TAF table to crosstab format, without transposing.

icesTAF-package gives an overview of the package.

Examples

```
taf2xtab(catage.taf)
tt(catage.taf)
tt(catage.taf, TRUE)
tt(catage.taf, "Custom")
```

---

write.dls *Write DLS3.2 Results to File*

Description

Write results from the DLS3.2 advisory method to a file.

Usage

```
write.dls(x, file = "")
```

Arguments

x	a list generated by the DLS3.2 function.
file	a filename.

Note

The resulting text file has Dos line endings (CRLF).

See Also

read.dls reads DLS3.2 results from a file back into R.

DLS3.2 in the **icesAdvice** package can be used to calculate catch advice for data-limited stocks (DLS).

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
survey <- data.frame(year=2001:2010, randu[1:10,])
dls <- icesAdvice::DLS3.2(1000, survey$y)

write.dls(dls, "dls.txt")
read.dls("dls.txt")

file.remove("dls.txt")

## End(Not run)
```

`write.taf` *Write TAF Table to File*

Description

Write a TAF table to a file.

Usage

```
write.taf(x, file = NULL, dir = NULL, quote = FALSE,
  row.names = FALSE, fileEncoding = "UTF-8", underscore = TRUE, ...)
```

Arguments

`x`	a data frame in TAF format.
`file`	a filename.
`dir`	an optional directory name.
`quote`	whether to quote strings.
`row.names`	whether to include row names.
`fileEncoding`	character encoding for output file.
`underscore`	whether automatically generated filenames (when `file = NULL`) should use underscore separators instead of dots.
`...`	passed to `write.csv`.

Details

Alternatively, `x` can be a list of data frames or a string vector of object names, to write many tables in one call. The resulting files are named automatically, similar to `file = NULL`.

The default value `file = NULL` uses the name of `x` as a filename, so a data frame called `survey.uk` will be written to a file called 'survey_uk.csv' (when `underscore = TRUE`) or 'survey.uk.csv' (when `underscore = FALSE`).

The special value `file = ""` prints the data frame in the console, similar to `write.csv`.

See Also

write.csv is the underlying function used to write a table to a file.

read.taf reads a TAF table from a file into a data frame.

icesTAF-package gives an overview of the package.

Examples

```
## Not run:
write.taf(catage.taf, "catage.csv")
catage <- read.taf("catage.csv")

write.taf(catage)
file.remove("catage.csv")

## End(Not run)
```

---

xtab2taf *Convert Crosstab Table to TAF Format*

Description

Convert a table from crosstab format to TAF format.

Usage

```
xtab2taf(x)
```

Arguments

x	a data frame in crosstab format.

Value

A data frame in TAF format.

Note

TAF stores tables as data frames with a year column, as seen in stock assessment reports. The crosstab format can be more convenient for analysis and producing plots.

See Also

catage.taf and catage.xtab describe the TAF and crosstab formats.

taf2xtab converts a TAF table to crosstab format.

icesTAF-package gives an overview of the package.

Examples

```
xtab2taf(catage.xtab)
```

Index

as.data.frame, *14*
available.packages, *9*

catage.long, *3*, 4, *5, 6*, *15*, *29*
catage.taf, *3, 4*, 5, *6*, *14, 15*, *29, 30*, *33*, *35*
catage.xtab, *3–5*, 6, *30*, *35*
clean, *3*, 7, *19*, *25*
cp, *3*, 7

deps, *3*, 9
dir.create, *19*
div, *3*, 10, *23*, *27*
DLS3.2, *21*, *33*
dos2unix, *3*, 11
download, *2*, 12
download.file, *13*

file, *12*
file.copy, *8*
file.rename, *8*
flr2taf, *3*, 13

grep, *11*, *23*

icesRound, *23*
icesTAF *(*icesTAF-package*)*, 2
icesTAF-package, 2
installed.packages, *9*

lim, *3*, 14
long2taf, *3, 4*, 15, *29*

make, *3*, 15, *16–18*
makeAll, *3*, *16, 17*, 17, *18*, *25*
makeTAF, *3*, *16–18*, 18, *26*
message, *20*
mkdir, *3*, 19
msg, *3*, 20

names, *21*

package.skeleton, *28*
plus, *3*, 20
png, *31*

read.csv, *22*
read.dls, *2*, 21, *33*
read.taf, *2*, *13*, 22, *35*
rnd, *3*, *11*, 23, *27*
round, *23*

source, *16–18*, *26*
sourceAll, *3*, *7*, *16–18*, 24, *26*
sourceTAF, *3*, *7*, *16–18*, *20*, *24, 25*, 25
summary.taf, *3*, 26

taf.blue *(*taf.colors*)*, 27
taf.colors, *3*, 27
taf.dark *(*taf.colors*)*, 27
taf.green *(*taf.colors*)*, 27
taf.light *(*taf.colors*)*, 27
taf.orange *(*taf.colors*)*, 27
taf.skeleton, *3*, 28
taf2long, *3*, *5*, *15*, 29
taf2xtab, *3*, *5*, 30, *33*, *35*
tafpng, *3*, 31
transform, *11*
tt, *3*, *30*, 32

unix2dos, *3*
unix2dos *(*dos2unix*)*, 11
unlink, *8*, *19*

write.csv, *35*
write.dls, *2*, *21*, 33
write.taf, *2*, *22*, 34
writeLines, *12*

xtab2taf, *3*, *6*, *30*, 35