Package ‘ncdf.tools’

May 29, 2015

Title Easier 'NetCDF' File Handling

Version 0.7.1.295

Date 2015-05-20

Author Jannis v. Buttlar

Maintainer Jannis v. Buttlar <jbuttlar@bgc-jena.mpg.de>

Imports RNetCDF, chron, parallel, abind, plotrix, raster,
RColorBrewer, JBTools

Description Set of tools to simplify the handling of 'NetCDF' files with the 'RNetCDF' package.
Most functions are wrappers of basic functions from the 'RNetCDF' package to easily run combinations of these
functions for frequently encountered tasks.

License GPL-2

LazyLoad yesf

Depends R (>= 2.10.0)

Repository CRAN

Repository/R-Forge/Project jbtools

Repository/R-Forge/Revision 30

Repository/R-Forge/DateTimeStamp 2015-05-29 12:34:21

Date/Publication 2015-05-29 15:39:30

NeedsCompilation no

R topics documented:

`ncdf.tools-package` *Easier 'NetCDF' File Handling*

Description

Set of tools to simplify the handling of 'NetCDF' files with the 'RNetCDF' package. Most functions are wrappers of basic functions from the 'RNetCDF' package to easily run combinations of these functions for frequently encountered tasks.

Details

Package:	ncdf.tools
Title:	Easier 'NetCDF' File Handling
Version:	0.7.1.295
Date:	2015-05-20
Author:	Jannis v. Buttlar
Maintainer:	Jannis v. Buttlar <jbuttlar@bgc-jena.mpg.de>
Imports:	RNetCDF, chron, parallel, abind, plotrix, raster, RColorBrewer, JBTools
License:	GPL-2

LazyLoad:	yesf
Depends:	R (>= 2.10.0)

Author(s)

Jannis v. Buttlar

`aggregateNcdf` *Aggregate data in netCDF files*

Description

This function aggregates time periods in netCDF files. Basically it is just a wrapper around the respective cdo function.

Usage

```
aggregateNcdf(fileName, path.out = getwd(), period)
```

Arguments

`fileName`	character vector: names of the files to aggregate.
`path.out`	character: path to save the results files to.
`period`	integer or one of hour, day, month or year: period to aggregate to. In case of an integer value, the unit is time steps.

Value

character string: name of the file created.

Author(s)

Jannis v. Buttlar

checkNcdfFile — *check netCDF file for consistency with CF/COARDS/BGI netCDF conventions*

Description

This function checks whether a netCDF file is consistent with the parts of the COARDS/CF netCDF conventions used in the BGI department (MPI for Biogeochemistry, Jena, Germany).

Usage

```
checkNcdfFile(file.name, dims = c("longitude", "latitude", "time"),
    type = "strict", var.check = "single")
```

Arguments

`file.name`	character string: file name to check
`dims`	vector of strings: names of the dimensions which need to be in the file.
`type`	character string: if 'strict', then all aspects are checked. If this is any other value, only aspects relevant for the processing of decomp.ncdf are checked.
`var.check`	character string: If 'single', then readNcdfVarName is used to infer the name of the variable in the target file which will then be checked,

Value

logical: (invisible) whether the file passed the check

Author(s)

Jannis v. Buttlar

classR2Ncdf — *transfers R classes to netCDF classes*

Description

Crudely determines the netCDF class from R classes. Only integer, character and double are implemented yet.

Usage

```
classR2Ncdf(object)
```

Arguments

`object`	object which class should be determined

Value

character string: netCDF class used in the RNetCDF package.

Author(s)

Jannis v. Buttlar

See Also

`RNetCDF`

`closeAllNcfiles` *Close all open RnetCDF file connections*

Description

closeAllNcfiles is a convenience function to close all netCDF connections that are currently open.

Usage

```
closeAllNcfiles()
```

Value

nothing is returned.

Author(s)

Jannis v. Buttlar

`convertBinary2Ncdf` *transform binary file to netCDF file*

Description

This function transforms a binary data file to a netCDF file formatted in a standardized way.

Usage

```
convertBinary2Ncdf(file.input, date.vec, length = 1, type = numeric(),
    type.ncdf = "NC_DOUBLE", dimensions, dimension.values, signed = TRUE,
    var.name, long_name = var.name, var.units = "[]", scale.factor.in = 1,
    scale.factor.out = scale.factor.in, na.value.in = -9999,
    na.value.out = na.value.in, offset.in = 0, offset.out = offset.in)
```

Arguments

`file.input`	character string: name of the input file.
`date.vec`	R date object: time vector for the time coordinate
`length`	integer: Length in bytes of each entry in the input file.
`type`	R data type of the data in the input file.
`type.ncdf`	character string: Desired data type in the netCDF file.
`dimensions`	character vector: Names of the dimensions in the binary file.
`dimension.values`	list: Each list element has to contain the coordinate values for the respective dimension.
`signed`	logical: Whether the binary file contains signed integer values.
`var.name`	character string: Short name of the variable in the binary file (used for the meta data in the NetCDF file).
`long_name`	character string: long name of the variable in binary file (used for the meta data in the NetCDF file).
`var.units`	character string: units of the variable (used for the meta data in the NetCDF file).
`scale.factor.in`	numeric: factor to multiply the binary input data with.
`scale.factor.out`	numeric: desired scale factor of the data in the netCDF file.
`na.value.in`	numeric: missing value for input data.
`na.value.out`	numeric: missing value for output data.
`offset.in`	numeric: offset for input data.
`offset.out`	numeric: offset for output data.

Value

Nothing is returned but a netCDF file with a standardized name is written in the working directory.

Author(s)

Jannis v. Buttlar

`convertDateNcdf2R` *Convert netCDF time vector to POSIXct R date object*

Description

This function converts a time vector from a netCDF file or a vector of Julian days (or seconds, minutes, hours) since a specified origin into a POSIXct R vector.

Usage

```
convertDateNcdf2R(time.source, units = "days", origin = as.POSIXct("1800-01-01",
    tz = "UTC"), time.format = c("%Y-%m-%d", "%Y-%m-%d %H:%M:%S",
    "%Y-%m-%d %H:%M", "%Y-%m-%d %Z %H:%M", "%Y-%m-%d %Z %H:%M:%S"))
```

Arguments

`time.source`	numeric vector or netCDF connection: either a number of time units since origin or a netCDF file connection, In the latter case, the time vector is extracted from the netCDF file, This file, and especially the time variable, has to follow the CF netCDF conventions.
`units`	character string: units of the time source. If the source is a netCDF file, this value is ignored and is read from that file.
`origin`	POSIXct object: Origin or day/hour zero of the time source. If the source is a netCDF file, this value is ignored and is read from that file.
`time.format`	

Value

POSIXct vector: time vector in native R format

Author(s)

Jannis v. Buttlar

`convertDateR2Ncdf` *Convert time vectors in netCDF files to Julian days since a certain origin*

Description

This function automatically converts time vectors in netCDF files to a standardized Gregorian calendar

Usage

```
convertDateR2Ncdf(ncdf.obj, date.vec = "auto", origin = "1800-01-01",
    write.to.ncdf = TRUE)
```

Arguments

`ncdf.obj`	character string or netCDF connection: netCDF file for which to convert the dates
`date.vec`	POSIXct vector: date vectors for the time dimension. If set to 'auto', this is tried to be extracted from the netCDF file
`origin`	character string: origin to be used for the time vector. This start of the Gregorian calendar should be kept to avoid possible mistakes due to flawed conversions.
`write.to.ncdf`	logical: whether to write the time vector to the netCDF file.

Details

This function sets a time vector in a netCDF file to a standardized format which is readable by most software. It transfers the time vector to days since the start of the Gregorian calendar.

Value

(invisibly): the time vector. Additionally the time vector is written to the respective file.

Author(s)

Jannis v. Buttlar

`convertFilename2Date` *Convert file name patterns to R date object*

Description

This function converts parts of netCDF file names to date strings (in case the file name contains date information). This is used, e.g. in transNcdfCutFiles.

Usage

```
convertFilename2Date(file.names, fun.extr.string, fun.conv.string)
```

Arguments

`file.names`	character vector: names of the files
`fun.extr.string`	function
`fun.conv.string`	function

Value

POSIXct object with the date.

Author(s)

Jannis v. Buttlar

See Also

`transNcdfCutFiles`

`createLatLongTime` *Create empty lat/lon/time netCDF file*

Description

this function creates an empty standardized latitude/longitude/time netCDF file.

Usage

```
createLatLongTime(file.name, var.names = sub("[.]nc", "", file.name),
    lat.values = c(), long.values = c(), time.values = c(), add.dims = list(),
    lat.length = length(lat.values), long.length = length(long.values),
    time.length = length(time.values), scale_factor = 1, add_offset = 0,
    type.var = "NC_DOUBLE", missing_value = -9999, units = "[]")
```

Arguments

`file.name`	character string: name of the target file.
`var.names`	character vector: names of the variables in the target file.
`lat.values`	numeric values: coordinate values for the latitude positions.
`long.values`	numeric values: coordinate values for the latitude positions.
`time.values`	POSIXct vector: time values for the time dimension
`add.dims`	
`lat.length`	integer: length of the latitude dimension
`long.length`	integer: length of the longitude dimension
`time.length`	integer: length of the time dimension
`scale_factor`	numeric: scale factor
`add_offset`	numeric: offset
`type.var`	character string: type of the data
`missing_value`	numeric: missing data value
`units`	character string: units of the variables in target file.

Value

Nothing is returned but a file is created. TODO: units has to work with more than one variable

Author(s)

Jannis v. Buttlar

`createStdNcdfFile` *Create an empty netCDF file with standardized attributes and dimensions*

Description

This function writes an empty netCDF file with variable names, dimensions and attributes formatted in a standardized way.

Usage

```
createStdNcdfFile(var.names, file.name = c(), units = "[]", lat.values = c(),
    long.values = c(), time.values = c(), add.dims = list(),
    lat.length = length(lat.values), long.length = length(long.values),
    time.length = length(time.values), year.start.end = c(),
    scale_factor = 1, add_offset = 0, type.var = "NC_DOUBLE",
    missing_value = -9999, con.atts = c(), data = c())
```

Arguments

`var.names`	character string: name of the target variable in the file
`file.name`	character string: name of the file. If not given, this is determined automatically in a standardized way from the variable name and the dimension extends.
`units`	character string: units of variable (should be compatible with udunits)
`lat.values`	numeric values: coordinate values for the latitude positions.
`long.values`	numeric values: coordinate values for the latitude positions.
`time.values`	POSIXct vector: time values for the time dimension
`add.dims`	
`lat.length`	integer: length of the latitude dimension
`long.length`	integer: length of the longitude dimension
`time.length`	integer: length of the time dimension
`year.start.end`	integer vector (length two): start and end year. If not given, this is determined from the time vector.
`scale_factor`	numeric: scale factor
`add_offset`	numeric: offset
`type.var`	character string: type of the data

`missing_value`	numeric: missing data value
`con.atts`	RNetCDF file connection: Possible file to use as source for copying attributes to the new file.
`data`	

Value

character string: name off the file created.

Author(s)

Jannis v. Buttlar

`indexDatacube` *Create logical index matrices for multidimensional datacubes*

Description

This function facilitates supplying logical index array for only some but not all of the dimensions of a data array. This mimics Matlabs indexing scheme. The indexing mechanisms of R only allow supplying logical indices for all dimensions.

Usage

```
indexDatacube(datacube = c(), logical.ind, dims = "auto", dims.datacube = dim(datacube))
```

Arguments

`datacube`	array: datacube from which to extract the sub-parts datacube and dims.datacube should be supplied.
`logical.ind`	logical array: TRUE/FALSE index matrix for a subset of the dimensions of the datacube. The size of logical.ind‘s dimensions has to match the sizes of the corresponding dimensions in datacube.
`dims`	integer vector or 'auto' : indices of the dimensions in datacube corresponding to the dimensions of logical.ind. If set to 'auto' this matching is tried to be accomplished by comparing the sizes of the dimensions of the two objects.
`dims.datacube`	integer vector: dimensions of the datacube. Only one of dims.datacube or datacube should be supplied!

Value

integer index matrix which can be used to index datacube

Author(s)

Jannis v. Buttlar

`infoNcdfAtts` *Print a summary of all netCDF variable attributes*

Description

This function returns a summary of all attributes of a single variable in a netCDF file.

Usage

```
infoNcdfAtts(file.con, var.id = "NC_GLOBAL")
```

Arguments

`file.con`	a NetCDF object pointing to the respective netCDF file.
`var.id`	the name or id of the variable for which to display attributes.

Details

If an id or variable name is given for 'var.id ', attributes from one variable are returned. Global attributes are returned if 'NC_GLOBAL' is given.

Value

A matrix containing the name, value and type (columns) of all attributes (rows)

Author(s)

Jannis v. Buttlar

See Also

`infoNcdfDims`, `infoNcdfVars`, `att.inq.nc`

`infoNcdfDims` *Show info about all dimensions in a netCDF file*

Description

This function displays summary information about all dimensions in a netCDF file.

Usage

```
infoNcdfDims(file.con, extended = TRUE)
```

Arguments

`file.con`	a NetCDF object pointing to the respective netCDF file.
`extended`	logical: if TRUE, some extended dimension info that may take time to compute for large files is computed.

Value

A matrix containing the id, name, length, range and step (columns) of all dimensions (rows)

Author(s)

Jannis v. Buttlar

See Also

`infoNcdfVars`, `dim.inq.nc`

`infoNcdfVars` *Display information about all variables in netCDF file*

Description

This function returns different summary information about all variables in a netCDF file.

Usage

```
infoNcdfVars(file.con, order.var = c("id", "name")[2], info.ext = FALSE,
    dimvars = FALSE)
```

Arguments

`file.con`	a NetCDF object pointing to the respective netCDF file.
`order.var`	character vector: Whether to sort the variables according to their name (default) or id.
`info.ext`	logical: whether to compute ranges/means etc. for the variables. Setting this to TRUE may take a while to compute with large files.
`dimvars`	logical: whether to include the coordinate variables in the output.

Value

a dataframe with the different information in its columns and each variable in one row.

Author(s)

Jannis v. Buttlar

See Also

infoNcdfDims, infoNcdfAtts

modifyNcdfAddDim — *Add a new dimension to one or more variables in a netCDF file*

Description

Adds another dimension to specified variables in a netCDF file and saves the results in another netCDF file.

Usage

```
modifyNcdfAddDim(file.con.orig, file.con.copy, var.name = "Default",
    dim.name = "new.dim", dim.values = c(), dim.length = length(dim.values),
    dim.pos.copy = 1)
```

Arguments

`file.con.orig`	a NetCDF object pointing to the respective netCDF file FROM which to copy
`file.con.copy`	a NetCDF object pointing to the respective netCDF file TO which to copy
`var.name`	character vector: names of the variables to which a dimension should be added. Defaults to all except those with identical names as dimensions in file.con.orig (coordinate variables)
`dim.name`	character string: name of the dimension to add
`dim.values`	numeric/character vector with the values for the dimension (coordinate values)
`dim.length`	integer: length of the dimension to add
`dim.pos.copy`	integer: position in the new dimension where to copy the original data. If set to 0, no values are copied and the variable in the new file will be empty. Setting to 1 (default) results in the original values to be filled in the first value of the new dimension and the remaining values left empty (NaN).

Author(s)

Jannis v. Buttlar

See Also

modifyNcdfCopyMetadata, att.copy.nc, modifyNcdfCopyVar

`modifyNcdfAppendHistory` *Append a string to netCDF history*

Description

Convenience function to append a string together with the date and the user to the history attribute of an NetCDF file.

Usage

```
modifyNcdfAppendHistory(file, string)
```

Arguments

`file`	character sting or RNetCDF file connection: file to write to.
`string`	character string: string to append to the history

Value

nothing is returned

Author(s)

Jannis v. Buttlar

`modifyNcdfCopyAtts` *Copy all attributes between different netCDF variables*

Description

modifyNcdfcopyAtts copies all attributes from one variable in a netCDF file to another variable (possibly in another file).

Usage

```
modifyNcdfCopyAtts(file.con.orig, var.orig, var.copy, file.con.copy = file.con.orig)
```

Arguments

`file.con.orig`	a NetCDF object pointing to the original netCDF file from which to copy the attributes
`var.orig`	the name or id of the variable FROM which to copy all attributes
`var.copy`	the name or id of the variable TO which to copy all attributes
`file.con.copy`	a NetCDF object pointing to the netCDF file to which to copy the attributes (same as file.con.orig by default)

Value

nothing is returned.

Author(s)

Jannis v. Buttlar

See Also

`modifyNcdfCopyMetadata`, `modifyNcdfCopyVar`

---

modifyNcdfCopyMetadata

Copy attributes and dimensions between netCDF files

Description

This function copies all global attributes and/or all dimensions from one netCDF file to another.

Usage

```
modifyNcdfCopyMetadata(file.con.orig, file.con.copy, glob.atts = TRUE,
    dimensions = TRUE)
```

Arguments

`file.con.orig`	a NetCDF object pointing to the respective netCDF file from which to copy
`file.con.copy`	a NetCDF object pointing to the respective netCDF file to which to copy
`glob.atts`	logical: whether to copy all global attributes
`dimensions`	logical: whether to copy all dimensions

Value

nothing is returned.

Author(s)

Jannis v. Buttlar

See Also

`modifyNcdfCopyVar`, `att.copy.nc`

`modifyNcdfCopyVar` *Copy variable values between netCDF files*

Description

modifyNcdfCopyVar copies all values of one variable from one netCDF file to another netCDF file and takes care of dimensions etc. .

Usage

```
modifyNcdfCopyVar(file.con.orig, file.con.copy = file.con.orig,
    var.id.orig, var.id.copy = var.id.orig)
```

Arguments

`file.con.orig`	a NetCDF object pointing to the original netCDF file FROM which to copy the variable.
`file.con.copy`	a NetCDF object pointing to the netCDF file TO which to copy the variable.
`var.id.orig`	character string or netCDF variable id: The name or id of the variable to copy from.
`var.id.copy`	character string or netCDF variable id: The name or id of the variable to copy to.

Details

Two cases are implemented:

Case 1: copy of one variable and attributes from one file to another file: The dimensions of the variable to copy have to be also existent (i.e. dimensions with the same name (not necessarily id)) in the netCDF file to which the variable should be copied. In addition these dimensions have to have the same sizes.

Case 2: copy of one variable to another one (of different name) in the same file.

Value

Nothing is returned

Author(s)

Jannis v. Buttlar

See Also

`modifyNcdfCopyMetadata`, `att.copy.nc`

modifyNcdfDefAtts *Define a set netCDF attributes at once*

Description

Easily define a couple of attributes for a single netCDF variable in one step.

Usage

```
modifyNcdfDefAtts(file.con, var.id, atts)
```

Arguments

`file.con`	a NetCDF object pointing to the respective netCDF file.
`var.id`	the variable id (integer) or name (string) for which to define attributes.
`atts`	list: the attributes to define (see details or an example).

Details

The atts attribute should be a list with as many elements as attributes should be added to the variable in the netCDF file. The names of the attributes are taken from the names of the elements of this list and the attribute values are defined by the values of the list elements. The type/class of the attribute (values) is determined automatically.

Author(s)

Jannis v. Buttlar

See Also

`att.put.nc`

Examples

```
## needs an open connection to a valid netCDF file pointed to by file.con
attributes.define <- list(LongName = 'This is the long name',
                          missingValue = -99999,
                          units = 'm/s')
library(RNetCDF)
file.con   <- create.nc('test.nc')
dim.def.nc(file.con, 'testdim')
var.def.nc(file.con, 'test', 'NC_CHAR', 'testdim')
modifyNcdfDefAtts(file.con, 'test', atts = attributes.define)

## show all attributes
infoNcdfAtts(file.con, 'test')
```

modifyNcdfSetMissing *Set missing value attribute to a netCDF file*

Description

This function sets the missing_value and the _Fill_value of a variable in a netCDF file to a given value (-9999 by default).

Usage

```
modifyNcdfSetMissing(con, var, value = -9999)
```

Arguments

`con`	file connection to modify
`var`	variable name (or index) of the variable to modify
`value`	value of the missing value attribute

Author(s)

Jannis v. Buttlar

modifyNcdfStdFile *Standardize file name and missing value attribute of a ncdf file.*

Description

Wrapper function around modifyNcdfStdFile and modifyNcdfSetMissing to standardize the name and the missing value attributes of a ncdf file.

Usage

```
modifyNcdfStdFile(file.con)
```

Arguments

`file.con`	character string (i.e. file name) or file connection of the file to change.

Author(s)

Jannis v. Buttlar

`modifyNcdfStdNames` *Modify non standard longitude and latitude names*

Description

This function modifies dimension names like 'lat', 'lon' and 'long' to 'latitude' and 'longitude'.

Usage

```
modifyNcdfStdNames(fileCon)
```

Arguments

`fileCon`	file in which to modify the names. can be supplied as a character vector or as a netCDF file connection.

Value

Nothing is returned

Author(s)

Jannis v. Buttlar

`plotDatacube` *Visualize/plot an overview of a netCDF file*

Description

This function plots some overview statistics of a netCDF file.

Usage

```
plotDatacube(data.object, data = c(), fourth.dim = 0, var.name = "auto",
    parallel = FALSE, max.cores = 16, n.series = 16, lwd = 2,
    ...)
```

Arguments

`data.object`	object to plot: file name or file.con object linking to a netCDF file
`data`	array: data to plot. Can be passed to the function to prevent the repeated loading of huge netCDF data.
`fourth.dim`	position in possible forth dimension (height, spectral band etc) to plot
`var.name`	character string: name of the variable to plot
`parallel`	logical: Whether to parallelize the computations.

`max.cores`	integer: maximum amount of cores to use for the parallelized computations.
`n.series`	integer: amount of example series to plot
`lwd`	integer: graphical parameter, see ?par
`...`	

Value

some overview statistics of the different datacubes.

Author(s)

Jannis v. Buttlar

`readFLUXNETNcdf` *read data from FLUXNET NetCDF file.*

Description

This function reads data from standard (BGI - Jena) FLUXNET netCDF files and returns it in an R object.

Usage

```
readFLUXNETNcdf(path = getwd(), sites, pars = "all", time.ends,
    dim.borders = list(1))
```

Arguments

`path`	character string: path to the input file(s)
`sites`	character string: ids of the sites to extract
`pars`	character string: names of the variables to extract
`time.ends`	POSIXct object: start and end date of the period to extract.
`dim.borders`	list: indices for other dimensions

Value

array: FLUXNET data

Author(s)

Jannis v. Buttlar

`readNcdf` *Easy reading of netCDF data*

Description

Convenience function to automatically read in data from a netCDF file without specifying variable names and opening and closing file connections.

Usage

```
readNcdf(file.name, var.name = c())
```

Arguments

`file.name`	character string: name of the netCDF file file to read the data from.
`var.name`	character string: name of the variable to extract. If not supplied, this is tried to be determined with readNcdfVarName().

Value

(multidimensional) array: data from the netCDF file.

Author(s)

Jannis v. Buttlar

`readNcdfCoordinates` *Read coordinate or dimension values from netCDF file*

Description

This function reads the coordinate values from a netCDF file.

Usage

```
readNcdfCoordinates(fileCon)
```

Arguments

`fileCon`	netCDF file connection or character string: Connection to the netCDF file or its file name. In the latter case, the connection is created and closed automatically.

Value

A list with the coordinate values (if available) for all dimensions.

Author(s)

Jannis v. Buttlar

See Also

`infoNcdfDims`

`readNcdfVarName` *Get name of variable in netCDF file*

Description

readNcdfVarName tries to automatically detect the name of the "main" variable in a netCDF file. The name returned is the name of a non coordinate variable. If more than one of such variables are existent, the name of the variable which spans all available dimensions or with a name appearing as a pattern in the file name is used.

Usage

```
readNcdfVarName(file)
```

Arguments

`file`	connection to the netCDF file.

Value

character string: name of the variable.

Author(s)

Jannis v. Buttlar

See Also

`RNetCDF`, `infoNcdfVars`

`transNcdfCutFiles` *Cut margins of netCDF files*

Description

Convenience wrapper around cdo to cut outer (time) margins of NetCDF files.

Usage

```
transNcdfCutFiles(file.names, time.range.out = c(), time.range.file = c(),
   fun.start = c(), fun.end = c(), format = "", convert = function(x) chron(paste(x,
       "15", sep = ""), format = "ymd", out.format = "d-m-y"))
```

Arguments

`file.names`	vector of character stings: file names to process.
`time.range.out`	
`time.range.file`	POSIXct vector of length two or 'auto': time range of the original files. If not supplied, this is determined automatically from the file name via convertFilename2Date and fun.start/fun.end.
`fun.start`	see time.range.file
`fun.end`	see time.range.file
`format`	
`convert`	

Value

character string: names of the file names after cutting.

Author(s)

Jannis v. Buttlar

`transNcdfMerge` *Merge several netCDF files*

Description

transNcdfMerge is a convenience wrapper around cdo to merge several netCDF files containing subsequent time steps into one continuous file.

Usage

```
transNcdfMerge(file.names, name.change = function(x) return(x),
    time.diff = NULL, fun.start = NULL, fun.end = NULL, time.range.out = c(),
    format = "%Y%m", convert = function(x) chron(paste(x, "15",
       sep = ""), format = "ymd", out.format = "d-m-y"), path.target = getwd(),
    target.name = "")
```

Arguments

`file.names`	character vector: names of the files to merge.
`name.change`	
`time.diff`	maximum time difference to be allowed between two subsequent input files.
`fun.start`	function: function to retrieve the start date from the file name e.g. function(x) substr(x, nchar(x)-15, nchar(x)-10)
`fun.end`	see fun.start
`time.range.out`	POSIXct vector: (start date, end date): start and end dates of the final file. If not supplied, all available data are used.
`format`	character string: see ?transNcdfCutFile
`convert`	
`path.target`	file path: path where to copy to the results files.
`target.name`	

Value

list: name of the file created and its time range.

Author(s)

Jannis v. Buttlar

`transNcdfRotate` *Transpose a NetCDF datacube*

Description

transNcdfRotate is a convenience function to transpose a datacube arranged in an arbitrary dimension order into a datacube with dimensions [latitude (decreasing), longitude (increasing), time (increasing)].

Usage

```
transNcdfRotate(data.object, file.name.out = "none", file.con = c(),
    var.name = c(), reverse.dim = TRUE)
```

Arguments

`data.object`	RNetCDF file connection or R array: data object to be transposed.
`file.name.out`	character string: name of the netCDF file created for the results. Default 'none' means that no results file is created.
`file.con`	RNetCDF file connection: link to the data object to be transposed. Supplying both data.object and file.con only makes sense if data.object is an array which saves time as the data does not have to be loaded again.
`var.name`	character string: name of the variable to transpose. If not gives, this name is tried to be inferred by using readNcdfVarName.
`reverse.dim`	logical: whether to reverse the sequence of the dimensions in case they are not in the following order : latitude: descending, longitude: ascending, time: ascending.

Value

list: data:transposed datacube and aperm.reverse: an index vector that can be supplied to aperm to reverse the transformation.

Author(s)

Jannis v. Buttlar

`transNcdfSubset` *Cut and save a subset of a netCDF file*

Description

This function reads a subset of lat/lon/time values out of a netCDF file and creates a new netCDF file with the results.

Usage

```
transNcdfSubset(file.input, dim.values = list(latitude = c(),
    longitude = c(), time = c()), values.type = c("range", "indices",
    "values")[2], file.output = sub("[.]nc", "_subs.nc", file.input),
    var.name = readNcdfVarName(file.input))
```

Arguments

`file.input`	character string: name of the input ncdf file.
`dim.values`	
`values.type`	character string: type of the dim.values supplied. 'range' means that the lower an upper border are supplied, 'indices' means that 1:n indices are supplied, 'values' would imply actual coordinate values.
`file.output`	character string: name of the results file.
`var.name`	

Value

character string: name of the file created.

Author(s)

Jannis v. Buttlar

Index