Package ‘nctools’

November 19, 2026

Version 0.2.0

Title Tools for Manipulating netCDF Files

Description Provides helper functions for creating, editing, subsetting, concatenating, and transforming netCDF files. The package includes utilities for writing arrays to netCDF, renaming variables and dimensions, changing longitude conventions, and applying summary functions over dimensions through 'ncdf4'.

Depends R (>= 4.0)

Imports ncdf4, stats, utils

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

License GPL-3

URL https://roliveros-ramos.github.io/nctools/

BugReports https://github.com/roliveros-ramos/nctools/issues

ByteCompile TRUE

Encoding UTF-8

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Ricardo Oliveros-Ramos [aut, cre] (ORCID: <https://orcid.org/0000-0002-8069-2101>)

Maintainer Ricardo Oliveros-Ramos <ricardo.oliveros@gmail.com>

Repository CRAN

Date/Publication 2026-11-19 15:50:02 UTC

Contents

checkLongitude 2
findPrimeMeridian 3
ncatt_get_all 4

ncatt_put_all 5
ncdim_isUnlim 6
ncdim_size 7
ncvar_change_compression 7
ncvar_compression 8
ncvar_dim 9
ncvar_size 10
nc_apply 11
nc_changePrimeMeridian 13
nc_extract 14
nc_loc 15
nc_mean 17
nc_quantile 19
nc_rcat 21
nc_rename 22
nc_subset 23
nc_unlim 24
write_ncdf 26

Index **30**

`checkLongitude` *Standardise longitude values to a selected prime meridian convention*

Description

Converts longitude values to either the `[-180, 180]` convention or the `[0, 360]` convention.

Usage

```
checkLongitude(x, primeMeridian = "center", sort = FALSE, ...)
```

Arguments

`x`	Numeric vector of longitude values.
`primeMeridian`	Character string specifying the target longitude convention. Use `"center"` for longitudes in the `[-180, 180]` range and `"left"` for longitudes in the `[0, 360]` range.
`sort`	Logical. If `TRUE`, sort the output values after conversion.
`...`	Additional arguments. Currently unused.

Details

Longitude values are modified only when needed. For `primeMeridian = "center"`, values greater than `180` are shifted by subtracting `360`. For `primeMeridian = "left"`, negative values are shifted by adding `360`.

Value

A numeric vector of longitude values expressed in the requested convention.

See Also

findPrimeMeridian()

Examples

```
local({
  checkLongitude(c(-80, -70, -60), primeMeridian = "left")
})
```

---

findPrimeMeridian *Identify the prime meridian convention from longitude values*

Description

Infers whether a longitude vector is expressed using the `[-180, 180]` convention or the `[0, 360]` convention.

Usage

```
findPrimeMeridian(x)
```

Arguments

x	Numeric vector of longitude values.

Details

The function returns `"center"` if any longitude is negative, and `"left"` if any longitude is greater than `180`. If neither condition is met, the convention cannot be determined unambiguously and the function returns `NULL` with a warning.

Value

A character string, either `"center"` or `"left"`, or `NULL` if the convention cannot be determined.

See Also

checkLongitude()

Examples

```
local({
  findPrimeMeridian(c(-80, -70, -60))
})
```

`ncatt_get_all` *Get all attributes from variables or dimensions*

Description

Returns all attributes associated with variables or dimensions in an open netCDF file.

Usage

```
ncatt_get_all(nc, type = c("var", "dim"))
```

Arguments

`nc`	An open netCDF connection created with `ncdf4::nc_open()`.
`type`	Character string indicating whether to return attributes for variables (`"var"`) or dimensions (`"dim"`).

Value

A named list of attribute lists.

See Also

`ncatt_put_all()`, `ncdf4::ncatt_get()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncatt_get_all(nc)
})
```

ncatt_put_all *Write multiple attributes to a netCDF variable or file*

Description

Writes several attributes to a variable or to the global file metadata of an open netCDF file.

Usage

```
ncatt_put_all(
  nc,
  varid,
  attname,
  attval,
  prec = NA,
  verbose = FALSE,
  definemode = FALSE
)
```

Arguments

nc	An open netCDF connection created with ncdf4::nc_open().
varid	Variable identifier. This can be a character string giving the variable name, an object of class ncvar4, or an integer id. As a special case, varid = 0 writes global attributes.
attname	Names of the attributes to write. Alternatively, this may be a named vector or list of attribute values when attval is omitted.
attval	Values of the attributes to write. If omitted, attname must be a named vector or list whose names are used as attribute names.
prec	Optional precision used when writing the attributes. Passed to ncdf4::ncatt_put().
verbose	Logical. If TRUE, print additional information while writing attributes.
definemode	Logical. Passed to ncdf4::ncatt_put(). See that function for details.

Details

Attributes can be supplied either as parallel attname and attval arguments, or as a single named vector or list. Missing attribute names are not allowed.

Value

NULL, invisibly.

See Also

ncatt_get_all(), ncdf4::ncatt_put()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncatt_put_all(nc, 0, attval = list(title = "Edited copy"))
})
```

---

ncdim_isUnlim *Test whether dimensions are unlimited*

Description

Returns whether each dimension in an open netCDF file is unlimited.

Usage

```
ncdim_isUnlim(nc)
```

Arguments

nc	An open netCDF connection created with `ncdf4::nc_open()`.

Value

A named logical vector indicating whether each dimension is unlimited.

See Also

`ncdim_size()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncdim_isUnlim(nc)
})
```

ncdim_size *Get dimension lengths from a netCDF file*

Description

Returns the lengths of all dimensions in an open netCDF file.

Usage

```
ncdim_size(nc)
```

Arguments

nc	An open netCDF connection created with `ncdf4::nc_open()`.

Value

A named list giving the length of each dimension.

See Also

`ncvar_dim()`, `ncdim_isUnlim()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncdim_size(nc)
})
```

ncvar_change_compression *Change variable compression settings in a netCDF object*

Description

Modifies the compression setting of one or more variables in an open netCDF object.

Usage

```
ncvar_change_compression(nc, varid = NA, compression)
```

Arguments

nc	An open netCDF connection created with ncdf4::nc_open().
varid	Variable identifier. This can be a character string giving the variable name, or an object of class ncvar4. If NA, the compression setting is changed for all variables in the file.
compression	Numeric compression level to assign.

Value

The modified netCDF object.

See Also

ncvar_compression()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  nc = ncvar_change_compression(nc, compression = 4)
  ncvar_compression(nc)
  # This changes the in-memory definition, not the stored data.
})
```

---

ncvar_compression *Get variable compression settings from a netCDF file*

Description

Returns the compression setting of one variable, or of all variables, in an open netCDF file.

Usage

```
ncvar_compression(nc, varid = NA)
```

Arguments

nc	An open netCDF connection created with ncdf4::nc_open().
varid	Variable identifier. This can be a character string giving the variable name, or an object of class ncvar4. If NA, compression settings are returned for all variables in the file.

Value

If `varid` is `NA`, a named vector or list with the compression setting of each variable. Otherwise, the compression setting of the selected variable.

See Also

`ncvar_change_compression()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncvar_compression(nc)
})
```

---

ncvar_dim *Get variable dimensions from a netCDF file*

Description

Returns the dimensions associated with one variable, or with all variables in an open netCDF file.

Usage

```
ncvar_dim(nc, varid = NULL, value = FALSE)
```

Arguments

`nc`	An open netCDF connection created with `ncdf4::nc_open()`.
`varid`	Variable identifier. This can be a character string giving the variable name, or an object of class `ncvar4`. If `NULL`, dimensions are returned for all variables in the file.
`value`	Logical. If `TRUE`, return the coordinate values of each dimension instead of only the dimension names.

Value

If `varid` is `NULL`, a named list with one element per variable. If `varid` is supplied, the dimensions of the selected variable only. When `value = FALSE`, dimensions are returned as character vectors of dimension names. When `value = TRUE`, dimensions are returned as named lists of coordinate values.

See Also

ncvar_size(), ncdim_size(), ncdf4::nc_open()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncvar_dim(nc)
})
```

---

ncvar_size *Get variable sizes from a netCDF file*

Description

Returns the dimension sizes of a selected variable in an open netCDF file.

Usage

```
ncvar_size(nc, varid = NULL)
```

Arguments

nc	An open netCDF connection created with ncdf4::nc_open().
varid	Variable identifier. This can be a character string giving the variable name, or an object of class ncvar4. If NULL, dimensions are returned for all variables in the file.

Details

If the file contains more than one variable, varid must be supplied.

Value

A numeric vector giving the dimension sizes of the selected variable. If the file contains a single variable, that variable is used by default.

See Also

ncvar_dim(), ncdim_size()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  copy = tempfile(fileext = ".nc")
  file.copy(input, copy)
  nc = ncdf4::nc_open(copy, write = TRUE)
  on.exit({ ncdf4::nc_close(nc); unlink(copy) }, add = TRUE)
  ncvar_size(nc)
})
```

---

nc_apply *Apply a function over margins of a netCDF variable*

Description

Reads a variable from a netCDF file, applies a function over one or more of its dimensions using `base::apply()`, and writes the result to a new netCDF file.

Usage

```
nc_apply(
  filename,
  varid,
  MARGIN,
  FUN,
  ...,
  output = NULL,
  drop = FALSE,
  newdim = NULL,
  name = NULL,
  longname = NULL,
  units = NULL,
  compression = NA,
  verbose = FALSE,
  force_v4 = TRUE,
  ignore.case = FALSE
)
```

Arguments

`filename`	Character string giving the path to the input netCDF file.
`varid`	Character string giving the name of the variable to process. If missing and the file contains a single variable, that variable is used.

MARGIN	Integer or character vector specifying the dimensions to retain in the output, as in base::apply(). Dimension names may be used.
FUN	Function to apply.
...	Additional arguments passed to FUN.
output	Character string giving the path to the output netCDF file to create.
drop	Logical. Currently not implemented.
newdim	Optional numeric values for the dimension created when FUN returns a vector of length greater than one.
name	Optional character string giving the name of the output variable. By default, the original variable name is used.
longname	Optional character string giving the long name of the output variable. By default, a name based on FUN and the original variable long name is generated.
units	Optional character string giving the units of the output variable. By default, the units of the input variable are used.
compression	Optional numeric compression level for the output variable.
verbose	Logical. Currently unused.
force_v4	Logical. Currently unused internally.
ignore.case	Logical. If TRUE, ignore case when matching dimension names supplied in MARGIN.

Details

The selected variable is read into memory and processed with base::apply(). The output retains the dimensions specified in MARGIN. If FUN returns a vector of length greater than one, an additional dimension is appended to the result; its coordinate values are taken from newdim when provided, or generated automatically otherwise.

Value

Invisibly returns output.

See Also

nc_subset(), write_ncdf(), base::apply()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
 nc_apply(input, "temperature", MARGIN = "time", FUN = mean, na.rm = TRUE, output = output)
})
```

nc_changePrimeMeridian *Change the prime meridian of a variable in a netCDF file*

Description

Reorders the longitude dimension of a variable so that longitudes follow either the `[-180, 180]` convention or the `[0, 360]` convention, and writes the result to a netCDF file.

Usage

```
nc_changePrimeMeridian(
  filename,
  output,
  varid = NA,
  MARGIN = 1,
  primeMeridian = "center",
  verbose = FALSE,
  overwrite = FALSE,
  compression = NA,
  mem.limit = 3072,
  ignore.case = FALSE
)
```

Arguments

`filename`	Character string giving the path to the input netCDF file.
`output`	Optional character string giving the path to the output netCDF file. If omitted, `overwrite = TRUE` must be used and the input file is replaced.
`varid`	Variable identifier. This can be a character string giving the variable name. If omitted and the file contains a single variable, that variable is used.
`MARGIN`	Integer or character string of length one specifying the longitude dimension. This can be either the dimension index or its name.
`primeMeridian`	Character string specifying the target longitude convention. Use `"center"` for longitudes in the `[-180, 180]` range and `"left"` for longitudes in the `[0, 360]` range.
`verbose`	Logical. If `TRUE`, report progress while processing the variable.
`overwrite`	Logical. If `TRUE`, allow overwriting an existing output file, including the input file itself.
`compression`	Optional numeric compression level for the output variable. Supplying this typically requires netCDF4 output.
`mem.limit`	Numeric. Approximate memory limit, in MiB, used when processing large variables. If the variable exceeds this limit, it is reordered iteratively in chunks.
`ignore.case`	Logical. If `TRUE`, ignore case when matching `varid` or a character `MARGIN`.

Details

The function updates the longitude coordinate values of the selected variable, reorders the data accordingly, and writes the result to a new netCDF file. If the variable is small enough, the full array is processed in memory. Otherwise, the variable is processed iteratively in chunks to reduce memory use.

If the longitude values already follow the requested convention, the function returns with a warning. When `output` differs from `filename`, the input file is copied to `output` unchanged.

Value

Invisibly returns `output`.

See Also

`nc_subset()`, `nc_apply()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_changePrimeMeridian(input, output, varid = "temperature", primeMeridian = "left")
})
```

---

nc_extract *Extract a variable to a new netCDF file*

Description

Creates a new netCDF file containing a single variable copied from an existing file.

Usage

```
nc_extract(filename, varid, output)
```

Arguments

`filename`	Character string giving the path to the input netCDF file.
`varid`	Character string giving the name of the variable to extract. If missing and the file contains a single variable, that variable is used.
`output`	Character string giving the path to the output netCDF file to create.

Value

Invisibly returns `output`.

See Also

`write_ncdf()`, `nc_subset()`, `nc_apply()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_extract(input, "temperature", output)
})
```

---

nc_loc *Locate the depth where a variable reaches a given value*

Description

Finds the position along one dimension of a netCDF variable where the variable reaches a target value, and writes the result to a new netCDF file. In the typical use case, this is used to estimate the depth at which a variable equals `loc`, using linear interpolation between adjacent depth levels when needed.

Usage

```
nc_loc(
  filename,
  varid,
  MARGIN,
  loc,
  output = NULL,
  drop = TRUE,
  newdim = NULL,
  name = NULL,
  longname = NULL,
  units = NULL,
  compression = NA,
  verbose = FALSE,
  force_v4 = TRUE,
  ignore.case = FALSE
)
```

Arguments

filename	Character string. Path to the input netCDF file.
varid	Character string. Name of the variable to analyse. If missing and the file contains only one variable, that variable is used.
MARGIN	Integer or character string of length one. Dimension along which the target value is searched. This can be either the dimension index or its name (for example "depth").
loc	Numeric. Target value to locate in the variable.
output	Character string. Path to the output netCDF file to create.
drop	Logical. Should degenerate dimensions be dropped? Currently not used internally.
newdim	Reserved for future use. Currently not used internally.
name	Character string. Name of the output variable. By default, the name of the searched dimension is used.
longname	Character string. Long name for the output variable. By default, a descriptive name is generated automatically.
units	Character string. Units for the output variable. Currently not used internally; by default, the units of the searched dimension are used.
compression	Numeric. Compression level for the output variable. If not NA, compression is applied and netCDF4 output is required.
verbose	Logical. Should extra information be printed? Currently not used internally.
force_v4	Logical. Should the output file be forced to netCDF4? Currently not used internally.
ignore.case	Logical. Ignore case when matching the dimension name in MARGIN?

Details

The function searches for the first sign change in `variable - loc` along the selected dimension, then estimates the exact position by linear interpolation between the two surrounding grid points.

The output file contains a new variable with the searched dimension removed. For example, if MARGIN corresponds to depth, the output is a field giving the estimated depth at which varid reaches loc for each remaining combination of dimensions.

Value

Invisibly returns the path to the created output file.

Examples

```
local({
  input = system.file("extdata",
    "temperature_profile.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
```

```
  on.exit(unlink(output))
  # Locate the 20 degree isotherm using linear interpolation in depth
  nc_loc(input, "temperature", MARGIN = "depth", loc = 20, output = output)
  nc = ncdf4::nc_open(output)
  result = tryCatch({
    list(lon = ncdf4::ncvar_get(nc, "lon"),
         lat = ncdf4::ncvar_get(nc, "lat"),
         depth = ncdf4::ncvar_get(nc, "depth"))
  }, finally = ncdf4::nc_close(nc))

  # The input profiles follow T(z) = 8 + (surface - 8) * exp(-z / scale)
  surface = outer(result$lon, result$lat, function(x, y)
    24 + 3 * exp(-((x + 72)^2 / 16 + (y + 15)^2 / 36)) + (y + 25) / 19)
  scale = outer(result$lon, result$lat, function(x, y)
    50 + 25 * ((x + 80) / 15 + (y + 25) / 19))
  analytical = scale * log((surface - 8) / (20 - 8))
  analytical[1:2, 1:2] = NA_real_
  # Interpolation on 10 m depth intervals approximates the exact solution
  error = result$depth - analytical
  print(range(error, na.rm = TRUE)) # signed interpolation error in metres
  print(max(abs(error), na.rm = TRUE))

  image(result$lon, result$lat, result$depth,
    xlab = "Longitude", ylab = "Latitude",
    main = "Depth of the 20 degree isotherm (m)")
  contour(result$lon, result$lat, analytical, add = TRUE)
  # Contours show the analytical depth; the missing corner remains blank
})
```

---

nc_mean *Compute summary statistics of a netCDF variable*

Description

Applies a summary function to a variable in a netCDF file over the dimensions not listed in `MARGIN`, and writes the result to a new netCDF file.

Usage

```
nc_mean(
  filename,
  varid,
  MARGIN = c(1, 2),
  na.rm = TRUE,
  trim = 0,
  output = NULL,
  drop = TRUE,
  compression = NA,
  verbose = FALSE,
```

```
  force_v4 = TRUE,
  ignore.case = FALSE
)

nc_min(
  filename,
  varid,
  MARGIN = c(1, 2),
  na.rm = TRUE,
  output = NULL,
  drop = TRUE,
  compression = NA,
  verbose = FALSE,
  force_v4 = TRUE,
  ignore.case = FALSE
)

nc_max(
  filename,
  varid,
  MARGIN = c(1, 2),
  na.rm = TRUE,
  output = NULL,
  drop = TRUE,
  compression = NA,
  verbose = FALSE,
  force_v4 = TRUE,
  ignore.case = FALSE
)
```

Arguments

filename	Character string giving the path to the input netCDF file.
varid	Character string giving the name of the variable to process. If missing and the file contains a single variable, that variable is used.
MARGIN	Integer or character vector specifying the dimensions to retain in the output, as in base::apply(). Dimension names may be used.
na.rm	Logical. If TRUE, missing values are removed before computing the statistic.
trim	Numeric scalar giving the fraction of observations to be trimmed from each end before computing the mean. Passed to base::mean(). Used only by nc_mean().
output	Character string giving the path to the output netCDF file to create.
drop	Logical. Currently not implemented.
compression	Optional numeric compression level for the output variable.
verbose	Logical. Currently unused.
force_v4	Logical. Currently unused internally.
ignore.case	Logical. If TRUE, ignore case when matching dimension names supplied in MARGIN.

Details

nc_mean() applies base::mean(), nc_min() applies base::min(), and nc_max() applies base::max().

The output retains the dimensions specified in MARGIN.

Value

Invisibly returns output.

See Also

nc_apply(), base::mean(), base::min(), base::max()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_mean(input, "temperature", MARGIN = c("lon", "lat"), output = output)
})
```

---

nc_quantile *Compute sample quantiles of a netCDF variable*

Description

Applies stats::quantile() to a variable in a netCDF file over the dimensions not listed in MARGIN, and writes the result to a new netCDF file.

Usage

```
nc_quantile(
  filename,
  varid,
  MARGIN = c(1, 2),
  na.rm = TRUE,
  probs = c(0, 0.5, 1),
  output = NULL,
  drop = TRUE,
  compression = NA,
  verbose = FALSE,
  force_v4 = TRUE,
  ignore.case = FALSE
)
```

Arguments

filename	Character string giving the path to the input netCDF file.
varid	Character string giving the name of the variable to process. If missing and the file contains a single variable, that variable is used.
MARGIN	Integer or character vector specifying the dimensions to retain in the output, as in base::apply(). Dimension names may be used.
na.rm	Logical. If TRUE, missing values are removed before computing the quantiles.
probs	Numeric vector of probabilities in [0, 1] passed to stats::quantile().
output	Character string giving the path to the output netCDF file to create.
drop	Logical. Currently not implemented.
compression	Optional numeric compression level for the output variable.
verbose	Logical. Currently unused.
force_v4	Logical. Currently unused internally.
ignore.case	Logical. If TRUE, ignore case when matching dimension names supplied in MARGIN.

Details

The output retains the dimensions specified in MARGIN. An additional dimension is appended to store the quantiles defined by probs.

Value

Invisibly returns output.

See Also

nc_apply(), stats::quantile()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_quantile(input, "temperature", MARGIN = c("lon", "lat"), output = output)
})
```

nc_rcat *Concatenate records of a variable across netCDF files*

Description

Concatenates the records of the same variable from multiple netCDF files along the unlimited dimension and writes the result to a new file.

Usage

```
nc_rcat(filenames, varid, output)
```

Arguments

filenames	Character vector with the paths to the input netCDF files.
varid	Character string giving the name of the variable to concatenate. If missing, the variable is inferred from the first file when possible.
output	Character string giving the path to the output netCDF file to create.

Details

All files must contain the selected variable and must be compatible in all non-unlimited dimensions. The unlimited dimension is appended in the order given by filenames.

Value

Invisibly returns output.

See Also

nc_unlim(), write_ncdf()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  files = system.file("extdata",
    sprintf("temperature_2001_%02d.nc",
    1:3),
    package = "nctools",
    mustWork = TRUE)
  first = tempfile(fileext = ".nc")
  on.exit(unlink(first), add = TRUE)
```

```
  nc_rcat(files[1:2], "temperature", first)
  nc_rcat(c(first, files[3]), "temperature", output)
})
```

`nc_rename` *Rename variables and dimensions in a netCDF file*

Description

Renames one or more variables and/or dimensions in an existing netCDF file, optionally writing the result to a new file.

Usage

```
nc_rename(
  filename,
  oldnames,
  newnames,
  output,
  verbose = FALSE,
  overwrite = FALSE
)
```

Arguments

`filename`	Character string giving the path to the input netCDF file.
`oldnames`	Character vector with the current names of variables or dimensions to rename.
`newnames`	Character vector with the replacement names. Must be in the same order and of the same length as `oldnames`.
`output`	Optional character string giving the path to the output file. If omitted, `overwrite = TRUE` must be used and the original file is replaced.
`verbose`	Logical. If `TRUE`, print the renaming operations performed.
`overwrite`	Logical. If `TRUE`, allow overwriting an existing output file. Use this when modifying the original file in place.

Details

Names in `oldnames` that do not match either a variable or a dimension in the file are ignored with a warning. If none of the requested names are found, the function exits without making changes.

Value

Invisibly returns the path to the modified file, or `NULL` invisibly if nothing was changed.

See Also

`nc_extract()`, `write_ncdf()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_rename(input, "temperature", "temp", output = output)
})
```

`nc_subset` *Subset a variable in a netCDF file*

Description

Extracts a subset of a variable from a netCDF file using coordinate bounds supplied for one or more dimensions, and writes the result to a new file.

Usage

```
nc_subset(
  filename,
  varid,
  output,
  newvarid,
  compression,
  force_v4 = FALSE,
  ...,
  ignore.case = FALSE,
  drop = FALSE
)
```

Arguments

`filename`	Character string giving the path to the input netCDF file.
`varid`	Character string giving the name of the variable to subset. If missing and the file contains a single variable, that variable is used.
`output`	Character string giving the path to the output netCDF file to create.
`newvarid`	Optional character string giving the name of the variable in the output file. By default, the original variable name is used.
`compression`	Optional numeric compression level for the output variable. Supplying this typically requires netCDF4 output.
`force_v4`	Logical. If `TRUE`, force creation of a netCDF4 output file.

...	Named bounds for dimensions to subset. Each argument name must match a dimension name. Values should be numeric vectors of length two giving lower and upper bounds. A length-one value is treated as an exact coordinate and expanded internally to `c(x, x)`.
`ignore.case`	Logical. If `TRUE`, ignore case when matching dimension names supplied in `...`.
`drop`	Logical. If `TRUE`, drop dimensions of length one in the output.

Details

Only dimensions named in `...` are subset; all others are kept in full. The output file contains the subsetted variable, with updated dimension values. Global attributes from the source file are copied to the output file, and the global `history` attribute is updated with the function call.

Value

Invisibly returns `output`, or `NULL` invisibly if none of the supplied dimension names match the variable dimensions.

See Also

`nc_extract()`, `write_ncdf()`

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc_subset(input, "temperature", output, lon = c(-75, -70), lat = c(-20, -10))
})
```

---

nc_unlim *Set a dimension as unlimited*

Description

Recreates a netCDF file with the selected dimension marked as unlimited.

Usage

```
nc_unlim(filename, unlim, output = NULL)
```

Arguments

filename	Character string giving the path to the input netCDF file.
unlim	Character string giving the name of the dimension to set as unlimited.
output	Optional character string giving the path to the output file. If NULL, the input file is replaced.

Details

The function rebuilds the file definition with the requested dimension marked as unlimited in every variable where that dimension is present, then copies the variable values to the new file.

Value

Invisibly returns the list of variable definitions used to create the new file.

See Also

nc_rcat(), write_ncdf()

Examples

```
local({
  input = system.file("extdata",
    "temperature_2001_01.nc",
    package = "nctools",
    mustWork = TRUE)
  output = tempfile(fileext = ".nc")
  on.exit(unlink(output))
  nc = ncdf4::nc_open(input)
  x = tryCatch(ncdf4::ncvar_get(nc, "temperature"), finally = ncdf4::nc_close(nc))
  fixed = tempfile(fileext = ".nc")
  on.exit(unlink(fixed), add = TRUE)
  write_ncdf(x,
    fixed,
    "temperature",
    dim = list(lon = seq(-80,
    -65,
    length.out = 15),
    lat = seq(-25,
    -6,
    length.out = 20),
    time = 0:30),
    compression = NA)
  nc_unlim(fixed, "time", output)
})
```

write_ncdf *Write data to a netCDF file*

Description

write_ncdf() is an S3 generic for writing R objects to a netCDF file. Methods are provided for writing a single array-like object (write_ncdf.default()) and a list of arrays (write_ncdf.list()).

Usage

```
write_ncdf(x, filename, ...)

## Default S3 method:
write_ncdf(
  x,
  filename,
  varid,
  dim,
  longname,
  units,
  prec = "float",
  missval = NA,
  compression = 9,
  chunksizes = NA,
  verbose = FALSE,
  dim.names,
  dim.units,
  dim.longname,
  unlim = FALSE,
  global = list(),
  force_v4 = FALSE,
  ...
)

## S3 method for class 'list'
write_ncdf(
  x,
  filename,
  varid,
  dim,
  longname,
  units,
  prec = "float",
  missval = NA,
  compression = 9,
  chunksizes = NA,
  verbose = FALSE,
```

```
  dim.names,
  dim.units,
  dim.longname,
  unlim = FALSE,
  global = list(),
  force_v4 = FALSE,
  ...
)
```

Arguments

x	Object to write. Supported methods currently accept either a single array-like object or a list of array-like objects.
filename	Character string giving the path to the netCDF file to create.
...	Additional arguments passed to methods.
varid	Character string giving the name of the variable to create in the output file.
dim	A named or unnamed list defining the coordinates of each dimension. Each element contains the coordinate values for one dimension. If omitted, dimensions are generated as `seq_len()` for each dimension of `x`.
longname	Character string giving the long name of the output variable. Defaults to `""`.
units	Character string giving the units of the output variable. Defaults to `""`.
prec	Character string giving the storage precision passed to `ncdf4::ncvar_def()`, for example `"short"`, `"integer"`, `"float"` or `"double"`.
missval	Missing value used in the netCDF variable definition.
compression	Numeric compression level passed to `ncdf4::ncvar_def()`. Compression usually requires netCDF4 output.
chunksizes	Optional chunk sizes for compressed variables.
verbose	Logical. Should `ncdf4` report progress while creating and filling the file?
dim.names	Character vector with names for the dimensions when `dim` is unnamed. Ignored when `dim` already has names.
dim.units	Character vector giving the units of the dimensions. If omitted, empty strings are used.
dim.longname	Character vector giving the long names of the dimensions. If omitted, empty strings are used.
unlim	Character string naming the unlimited dimension. Use `FALSE` for no unlimited dimension.
global	Named list of global attributes to add to the output file. A `history` attribute documenting the function call is added automatically.
force_v4	Logical. Should the output file be forced to netCDF4 format?

Details

`write_ncdf.default()` writes a single object `x` as one variable in a new netCDF file. The dimensions of the variable are defined by `dim`.

If `dim` is unnamed, dimension names are taken from `dim.names`, or generated automatically as `"dim1"`, `"dim2"`, and so on. Dimension units and long names can be supplied through `dim.units` and `dim.longname`.

`write_ncdf.list()` writes each element of `x` as a separate variable in the same netCDF file.

If `varid` is omitted, variable names are taken from `names(x)`. Arguments such as `longname`, `units`, and `prec` can be supplied either as length-one values, to be recycled to all variables, or as one value per variable.

The object `dim` defines the full set of dimensions available in the file. Individual variables may use all or a subset of those dimensions, provided their sizes are consistent with the declared dimension lengths.

Value

Invisibly returns `filename`.

Examples

```
local({
  # Synthetic daily temperature for January 2001, as in the bundled data
  lon = seq(-80, -65, length.out = 15)
  lat = seq(-25, -6, length.out = 20)
  days = 0:30
  spatial = outer(lon, lat, function(x, y)
    16 + 6 * exp(-((x + 72)^2 / 16 + (y + 15)^2 / 36)) - 0.15 * (y + 15))
  temperature = outer(spatial, days, function(field, day)
    field + 2 * sin(2 * pi * day / 365) + 0.2 * sin(2 * pi * day / 7))
  temperature[1:2, 1:2, ] = NA_real_

  files = vapply(seq_len(3), function(i) tempfile(fileext = ".nc"), "")
  on.exit(unlink(files))

  # Quick defaults: dimensions are named dim1, dim2, dim3, with index values
  write_ncdf(temperature, filename = files[1], varid = "temperature")

  # Supply geographic coordinates, time coordinates, and variable metadata
  coordinates = list(lon = lon, lat = lat, time = days)
  coordinate_units = c("degrees_east", "degrees_north", "days since 2001-01-01")
  write_ncdf(temperature, filename = files[2], varid = "temperature",
    dim = coordinates, dim.units = coordinate_units,
    longname = "Synthetic daily temperature", units = "degree_C")

  # Add descriptive attributes, unlimited time, and explicit storage settings
  write_ncdf(temperature, filename = files[3], varid = "temperature",
    dim = coordinates, dim.units = coordinate_units,
    dim.longname = c("Longitude", "Latitude", "Time"),
    longname = "Synthetic daily temperature", units = "degree_C",
    unlim = "time", prec = "float", missval = -9999,
```

```
    compression = 4, force_v4 = TRUE,
    global = list(
      title = "Synthetic temperature for January 2001",
      source = "Spatial gradient and Gaussian centre with annual and weekly cycles",
      comment = "Synthetic data, not observations; a fixed corner patch is missing."
    ))

  nc = ncdf4::nc_open(files[3])
  tryCatch({
    print(ncvar_dim(nc, "temperature"))
    print(ncdf4::ncatt_get(nc, 0, "title"))
  }, finally = ncdf4::nc_close(nc))
})
```

Index

base::apply(), *11, 12*, *18*, *20*
base::max(), *19*
base::mean(), *18, 19*
base::min(), *19*

checkLongitude, 2
checkLongitude(), *3*

findPrimeMeridian, 3
findPrimeMeridian(), *3*

nc_apply, 11
nc_apply(), *14, 15*, *19, 20*
nc_changePrimeMeridian, 13
nc_extract, 14
nc_extract(), *22*, *24*
nc_loc, 15
nc_max *(*nc_mean*)*, 17
nc_mean, 17
nc_min *(*nc_mean*)*, 17
nc_quantile, 19
nc_rcat, 21
nc_rcat(), *25*
nc_rename, 22
nc_subset, 23
nc_subset(), *12*, *14, 15*
nc_unlim, 24
nc_unlim(), *21*
ncatt_get_all, 4
ncatt_get_all(), *5*
ncatt_put_all, 5
ncatt_put_all(), *4*
ncdf4::nc_open(), *4–10*
ncdf4::ncatt_get(), *4*
ncdf4::ncatt_put(), *5*
ncdf4::ncvar_def(), *27*
ncdim_isUnlim, 6
ncdim_isUnlim(), *7*
ncdim_size, 7
ncdim_size(), *6*, *10*
ncvar_change_compression, 7
ncvar_change_compression(), *9*
ncvar_compression, 8
ncvar_compression(), *8*
ncvar_dim, 9
ncvar_dim(), *7*, *10*
ncvar_size, 10
ncvar_size(), *10*

stats::quantile(), *19, 20*

write_ncdf, 26
write_ncdf(), *12*, *15*, *21, 22*, *24, 25*