Package 'Rcost'

October 4, 2017

Type Package

Title Cost 733 WLK Classification in R

Version 0.1-0

Date 2017-10-03

Author Reto Stauffer

Maintainer Reto Stauffer <reto.stauffer@uibk.ac.at>

Description A small R interface to the cost733 WLK classification method.

Depends R (>= 3.2.0)

Imports zoo, raster, ncdf4, sp

LazyData false

License GPL-2 + file LICENSE

RoxygenNote 6.0.1

NeedsCompilation yes

R topics documented:

`datetimeinfo` *Getting Time/Forecast Time Information from a NetCDF Connection*

Description

Returns the time or forecast time information from a `ncdf4` connection. Developed for some ECMWF forecasts, time information is stored as hours since `1900-01-01` wherefore `init` is required to compute forecast steps.

Usage

```
## Basic usage
datetimeinfo(nc,init)
```

Arguments

`nc`	Open `ncdf4` connection.
`init`	`Date`, `POSIXt`, or `character` which can be converted into a `POSIXt` object.

Value

Returns a `data.frame` with 3 columns. `init` contains the `init` datetime, `valid` is the time for which the forecast is valid, and `step` is the forecast step or lead time.

Author(s)

Reto Stauffer

Examples

```
## Not run:
# Open NetCDF connection
nc <- nc_open("costgrib/ECMWF_cost_sfc_201111180000.nc")
# Get time information
datetimeinfo(nc,"2011-11-18")
# Close NetCDF connection
nc_close(nc)

## End(Not run)
```

demofiles *Returns Rcost Package Demo File Locations*

Description

This package contains some demo data used for the examples and vignettes. This function simply returns the location of these files on the local disc.

Usage

```
## Basic usage
demofiles( silent = FALSE )
```

Arguments

silent	logical. If TRUE console output is supressed.

Value

Returns a list object with two elements. sfc locates the path to the demo surface data NetCDF file, pl the one to the corresponding pressure level NetCDF file.

Both files contain five subsequent ERA ITERIM reanalysis fields from July 1th, 2000 to July 5th, 2000.

Author(s)

Reto Stauffer

Examples

```
files <- demofiles( )
print(files)
```

getClassification *Getting the cost733 WLK Classification For a Given Date*

Description

This function was the reason to set up this small R package. This package performs the cost733 WLK classification on gridde data sets such as the ECMWF INTERIM or ECMWF HIRES. The classification is performed based on the fortran routine by Andreas Philipp, UNI Augsburg.

http://cost733class.geo.uni-augsburg.de/moin/cost733wiki/data/pages_bak/WLKC733/attachments/cost733_wlk_1.07.f90

The classification requires the variables "tcw" (total column water), "u700" and "v700" (zonal and meridional wind component in 700 hectopascal), as far as "z500" and "z925". These variables have

to be provided by the nc files given as input argument. See details on how the netcdf files should look like.

The defaults of this method use the specificationas as proposed by Thomas Krennert (ZAMG Vienna). The method here is not 100 percent generic as it e.g, expects a specific file format (and file names) was I have only planned to use the script on ECMWF 00UTC deterministic forecasts at the moment.

Usage

```
## Basic usage
getClassification(nc, date, steps, subset, pwclim, weights,
                  nsector=8, maindirthreshold=0.4,
                  zamg=FALSE, verbose=0)
```

Arguments

nc	list containing one or more open netcdf connection opened by ncdf4::nc_open.
date	Date, POSIXt, or character object which can be converted into POSIXt. Date/time when the forecast was initialized. Used to find the NetCDF files with a specific (fixed) name at the moment.
steps	Optional, if not set the classification will be performed on all steps provided by the NetCDF files.
subset	Optional, default NULL. See getdata for more details.
pwclim	Optional. If not set the dry/wet classification cannot be performed. If set it has to be a numeric vector of length 366 containing the climatological "precip. water content" for each day of the year.
weights	Optional, if not set the ZAMG config with c(15,2,1) will be used for the weighting. If set it has to be a numeric vector of length 3 (for core/mid/margin weight).
nsector	Integer, default 8. Number of wind sectors for the classification.
maindirthreshold	numeric, default is 0.4. Used for wind sector classification. If less than maindirthreshold can be classified for one main wind direction the wind sector classification will be "variable" (sector=0).
zamg	Logical flag, default FALSE. If set to TRUE the main wind direction will be computed based on the classification by "Trendanalyse Klien Endbericht" (as suggested by Thomas Kennert, ZAMG).
verbose	Integer. Logical TRUE/FALSE will be converted to 0/1. However, verbose levels up to 4 (highly verbose) can be set.

Details

For subset please see details in getdata.

The netcdf files have to provide the variables u and v (wind components) on 700 hectopascal, z (geopotential height) on 500 and 925 hectopascal, and tcw (total column water). To be able to extract the necessary information the netcdf file(s) have to provide the following information: - dimension longitude (named longitude) - dimension latitude (named latitude) - dimension time (named

time; units something like "hours since 1900-01-01 00:00" or similar) - dimension level (named level) which is used when searching for level variables I typically use two different files, a surface level netcdf (dimensions: longitude, latitude, and time; containing at least tcw), and a pressure level file (dimensions: longitude, latitude, time, and level; containing at least u700, v700, z500, z925) and use both as input arguments. E.g., - `getClassification( list(nc_open('<sfcfile>'),nc_open('<plfile>')), ... )`

Value

Returns a `zoo` object with the following items: - `date`: date/time when the classification is valid - `yday`: day of the year - `sector`: classifier: wind sector - `cyclonic925/cyclonic500`: classifier: wheter it is `0=anticyclonic` or `1=cyclonic` in 925/500 hPa - `wet`: classifier: `-99=missing` (due to lack of pwclim), `0=dry`, and `1=wet`. - `c925/c500`: curvature values used to create the classification. - `pwi`: computed pwi values which will be used for classification

Author(s)

Reto Stauffer

References

Philipp A., C. Beck, R. Huth and J. Jacobeit (2014): Development and comparison of circulation type classifications using the COST 733 dataset and software. International Journal of Climatology. DOI: 10.1002/joc.3920

Examples

```
# Open netcdf connections
require("ncdf4")
files <- demofiles()
nc    <- list(nc_open(files$sfc), nc_open(files$pl)) # Open NetCDF Files

# Extract initial date (hours since 1900-01-01 00:00:0.0)
# Take first date only.
init  <- as.POSIXct( ncvar_get(nc[[1]],"time") * 3600, origin = "1900-01-01" )[1]

# Perform classification
x1 <- getClassification( nc, init)
plot(x1, main = "COST733 Classification Output (x1)")

# Use different areal weights and only 4 wind sectors
x2 <- getClassification( nc, init, weights = c(10,5,2), nsector = 4 )
plot(x2, main = "COST733 Classification Output (x2)")

# Provide pw climatology (in this case simply a constant of
# 24.844 for all 1:365 days of a year): returns additional
# binary classification "wet" (simply pwi > pwiclim).
x3 <- getClassification( nc, init, weights = c(10,5,2), nsector = 8,
                         pwclim = rep(24.844,365) )
plot(x3, main = "COST733 Classification Output (x3)")

# Close NetCDF Files
for ( n in nc ) nc_close(n)
```

getdata *Extracting Data from NetCDF File Connection*

Description

Built to extract certain variables/forecast times from NetCDF files.

Usage

```
## Basic usage
getdata(nc, init, varname, steps, level = NULL, subset = NULL, silent = FALSE )
```

Arguments

nc	Open ncdf4 connection
init	Date, POSIXt, or character object which can be converted into POSIXt. Date/time when the forecast was initialized. Crucial to compute the forecast steps (lead times).
varname	character, name of the variable to load.
steps	Optional. If not set all forecast steps from the nc file will be returned. Single or multiple steps can be defined (numeric).
level	Used when loading variables from pressure level files where the level has to be specified as a numeric value.
subset	Optional, default NULL. If not set the whole grid from the nc file will be returned. Can also be a extent object or a RasterLayer object. See details.
silent	logical, default is FALSE. If TRUE some messages will be suppressed.

Details

subset allows to only load parts of the data from the nc connection. If subset is of class Extent (see extent) then only (a spatial) subset of the data will be taken. If subset is of class RasterLayer (see raster) the extent of the RasterLayer will be used to crop the data, the resolution (or grid specification) of this RasterLayer will be used to re-sample the data. This allows to change the resolution when loading the data.

Value

Returns a RasterLayer (if only one step) is loaded or a RasterStack object where each layer corresponds to a forecast step. Names of the layers contain the initial initial date plus the forecast step.

Author(s)

Reto Stauffer

Examples

```
## Not run:
library('maps'); library("raster")

# Open netcdf connections
init <- as.POSIXct("2011-11-18 00:00")
init_str <- strftime(init,"
ncs <- nc_open(sprintf("costgrib/ECMWF_cost_sfc_
ncp <- nc_open(sprintf("costgrib/ECMWF_cost_pl_

# Loading "tcw" from surface file, step 24, full field
x <- getdata(ncs,init,"tcw",24)
plot(x); map(add=T)

# Using an extent subset and loading three different steps
x <- getdata(ncs,init,"tcw",c(12,24,48),subset=extent(c(10,30,40,50)))
plot(x)

# Loading pressure level data (u on 700 hPa) and resample the data
# onto a coarser resolution.
template <- raster(xmn=9.5,xmx=30.5,ymn=39.5,ymx=50.5,nrows=21,ncols=11,
             crs=crs("+proj=longlat +datum=WGS84 +ellps=WGS84 +no_defs +towgs84=0,0,0"))
x <- getdata(ncp,init,"u",144,level=700,subset=template)
plot(x); map(add=T)

# Close connections
nc_close(ncs)
nc_close(ncp)

## End(Not run)
```

---

getlonlat *Getting Grid Information from a Raster Object*

Description

Returns the grid specification of a `raster` object.

Usage

```
## Basic usage
getlonlat( x )
```

Arguments

x	`RasterLayer` or `RasterStack` object.

Value

Returns a `list` object containing the following information: - `lons`/`lats`: vector of unique longitude and latitude points - `dx`/`dy`: coordinate increments in x/y direction (grid spacing) - `nx`/`ny`: number of grid points in x/y direction

Author(s)

Reto Stauffer

Examples

```
library("raster")
# Create an empty demo RasterLayer object
test <- raster(xmn=9.5,xmx=20.5,ymn=19.5,ymx=30.5,ncols=11,nrows=11)
# Getting grid information
getlonlat( test )
```

`weightmask` *Returns cost733 Style Weighting Mask*

Description

The cost733 WLK weather type classification algorithm uses a weight mask. This allows to "focus" on specific parts of the area of interest. There are three different zones called "core", "mid" and "margin".

By default (cost733) uses a `stripsize` of `5` (cut's the area in five rows and five columns). This creates a rectangular core of one fifth times one fifth of the full area and two surrounding "bands". The inner "band" is the "mid" zone, the outer one (reaching the edges of the zone) the "margin".

Each of these three zones gets a weight, by default `15` for the "core", `2` for "mid", and `1` for "margin". During computation values within these zones are weighted according to the `weights` set.

Usage

```
## Basic usage
weightmask( lon, lat, weights=c(15,2,1), stripsize=5 )
```

Arguments

`lon`	`numeric` vector of longitude values.
`lat`	`numeric` vector of latitude values.
`weights`	`numeric` vector of length three with the weights for the three zones "core", "mid" and "margin".
`stripsize`	`numeric` value which is used to define the zones.

Value

Returns a matrix of size `length(lon)` times `length(lat)` with the corresponding weights.

Author(s)

Reto Stauffer

See Also

weightmaskZAMG, getClassification

Examples

```
lons <- seq( 5,30,by=0.25)
lats <- seq(40,60,by=0.25)
w1 <- weightmask( lons, lats )
w2 <- weightmask( lons, lats, weights=c(3,2,1) )
w3 <- weightmask( lons, lats, stripsize = 10 )
w4 <- weightmask( lons, lats, weights=c(3,2,1), stripsize = 4.5 )

# Pretty ugly plots
par(mfrow=c(2,2))
   image(w1,zlim=c(1,15),col=1:15,main="weight mask c(15,2,1) stripsize 5")
   image(w2,zlim=c(1,15),col=1:15,main="weight mask c(3,2,1) stripsize 10")
   image(w3,zlim=c(1,15),col=1:15,main="weight mask c(15,2,1) stripsize 5")
   image(w4,zlim=c(1,15),col=1:15,main="weight mask c(3,2,1) stripsize 4.5")
```

---

weightmaskZAMG *Returns cost733-ZAMG Style Weighting Mask*

Description

The cost733 WLK weather type classification algorithm uses a weight mask. This allows to "focus" on specific parts of the area of interest. There are three different zones called "core", "mid" and "margin".

getClassification provides an option which is called zamg=TRUE. In case this is set the algorithm used the ZAMG weighting mask as shown in "Trendanalyse von hydro-meteorologischen Extremwerten". The are using a fixed weight-mask specification for Austria.

This function returns the weights with respect to the WLK classification as used by the ZAMG in Vienna.

Usage

```
## Basic usage
weightmaskZAMG( lon, lat, weights=c(15,2,1), ... )
```

Arguments

lon	numeric vector of longitude values.
lat	numeric vector of latitude values.
weights	numeric vector of length three with the weights for the three zones "core", "mid" and "margin".
...	Unused additional arguments.

Value

Returns a matrix of size `length(lon)` times `length(lat)` with the corresponding weights.

Author(s)

Reto Stauffer

See Also

weightmask, getClassification

Examples

```
lons <- seq( 5,30,by=0.25)
lats <- seq(40,60,by=0.25)
w1 <- weightmaskZAMG( lons, lats )
w2 <- weightmaskZAMG( lons, lats, weights=c(3,2,1) )
w3 <- weightmaskZAMG( lons, lats, stripsize = 10 )
w4 <- weightmaskZAMG( lons, lats, weights=c(3,2,1), stripsize = 4.5 )

# Pretty ugly plots
par(mfrow=c(2,2))
   image(w1,zlim=c(1,15),col=1:15,main="weight mask c(15,2,1) stripsize 5")
   image(w2,zlim=c(1,15),col=1:15,main="weight mask c(3,2,1) stripsize 10")
   image(w3,zlim=c(1,15),col=1:15,main="weight mask c(15,2,1) stripsize 5")
   image(w4,zlim=c(1,15),col=1:15,main="weight mask c(3,2,1) stripsize 4.5")
```

Index