Package ‘remote’

February 20, 2015

Type Package
Title Empirical Orthogonal Teleconnections in R
Version 0.3.0
Date 2014-08-25
Author Tim Appelhans, Florian Detsch, Thomas Nauss
Maintainer Tim Appelhans <tim.appelhans@gmail.com>
Description 'remote' is short for "R(-based) EMpirical Orthogonal TEleconnections". It implements a collection of functions to facilitate empirical orthogonal teleconnection analysis. Empirical Orthogonal Teleconnections (EOTs) denote a regression based approach to decompose spatio-temporal fields into a set of independent orthogonal patterns. They are quite similar to Empirical Orthogonal Functions (EOFs) with EOTs producing less abstract results. In contrast to EOFs, which are orthogonal in both space and time, EOT analysis produces patterns that are orthogonal in either space or time.
License GPL (>= 3) | file LICENSE
Depends R (>= 2.10), Rcpp (>= 0.10.3), raster, foreach, methods
Suggests maps, lattice, latticeExtra, grid, gridExtra, sp
LinkingTo Rcpp
NeedsCompilation yes
Repository CRAN
Date/Publication 2014-08-25 16:03:36

R topics documented:

remote-package .................................................. 2
anomalize ......................................................... 3
australiaGPCP ................................................... 4
calcVar .......................................................... 4
covWeight ......................................................... 5
cutStack .......................................................... 6
Description

R EMpirical Orthogonal TEleconnections

Details

A collection of functions to facilitate empirical orthogonal teleconnection analysis. Some handy functions for preprocessing, such as deseasoning, denoising, lagging are readily available for ease of usage.

Author(s)

Tim Appelhans, Florian Detsch

Maintainer: Tim Appelhans <tim.appelhans@gmail.com>

References

Empirical Orthogonal Teleconnections
anomalize

Empirical methods in short-term climate prediction
H. M. van den Dool (2007)

See Also

remote is built upon Raster* classes from the raster-package. Please see their documentation for
data preparation etc.

anomalize

Create an anomaly RasterStack

Description

The function creates an anomaly RasterStack either based on the overall mean of the original stack,
or a supplied reference RasterLayer. For the creation of seasonal anomalies use deseason.

Usage

anomalize(x, reference = NULL, ...)

Arguments

x a RasterStack
reference an optional RasterLayer to be used as the reference
... additional arguments passed to calc which is used under the hood

Value

an anomaly RasterStack

See Also
deseason, denoise, calc

Examples

data(australiaGPCP)

aus_anom <- anomalize(australiaGPCP)

opar <- par(mfrow = c(1,2))
plot(australiaGPCP[[10]], main = "original")
plot(aus_anom[[10]], main = "anomalized")
par(opar)
calcVar

Description

The function calculates the (optionally standardised) space-time variance of a RasterStack or RasterBrick.

Usage

calcVar(x, standardised = FALSE, ...)

---

australiaGPCP  Monthly GPCP precipitation data for Australia

Description

Monthly Gridded Precipitation Climatology Project precipitation data for Australia from 1982/01 to 2010/12

Format

a RasterBrick with the following attributes

dimensions : 12, 20, 240, 348 (nrow, ncol, ncell, nlayers)
resolution : 2.5, 2.5 (x, y)
extent : 110, 160, -40, -10 (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +ellps=WGS84 +towgs84=0,0,0,0,0,0,0 +no_defs

Details

Monthly Gridded Precipitation Climatology Project precipitation data for Australia from 1982/01 to 2010/12

References

The Version-2 Global Precipitation Climatology Project (GPCP) Monthly Precipitation Analysis (1979 - Present)
Adler et al. (2003)
Journal of Hydrometeorology, Volume 4, Issue 6, pp. 1147 - 1167
http://dx.doi.org/10.1175/1525-7541(2003)004<1147:TVGPCP>2.0.CO;2

calcVar  Calculate space-time variance of a RasterStack or RasterBrick

Description

The function calculates the (optionally standardised) space-time variance of a RasterStack or RasterBrick.

Usage

calcVar(x, standardised = FALSE, ...)

---
covWeight

Arguments

  x            a RasterStack or RasterBrick
  standardised  logical.
  ...          currently not used

Value

  the mean (optionally standardised) space-time variance.

Examples

  data("pacificSST")
  calcVar(pacificSST)

Description

  Create a weighted covariance matrix

Usage

  covWeight(m, weights, ...)

Arguments

  m            a matrix (e.g. as returned by getValues)
  weights      a numeric vector of weights. For lat/lon data this can be produced with getWeights
  ...          additional arguments passed to cov.wt

Value

  see cov.wt

See Also

  cov.wt
cutStack

**Shorten a RasterStack**

**Description**

The function cuts a specified number of layers off a RasterStack in order to create lagged RasterStacks.

**Usage**

\[
cutStack(x, \text{tail} = \text{TRUE}, n = \text{NULL})
\]

**Arguments**

- \( x \) a RasterStack
- \( \text{tail} \) logical. If \text{TRUE} the layers will be taken off the end of the stack. If \text{FALSE} layers will be taken off the beginning.
- \( n \) the number of layers to take away.

**Value**

a RasterStack shortened by \( n \) layers either from the beginning or the end, depending on the specification of \( \text{tail} \)

**Examples**

```r
data(australiaGCP)

# 6 layers from the beginning
cutStack(australiaGCP, \text{tail} = \text{FALSE}, n = 6)

# 8 layers from the end
cutStack(australiaGCP, \text{tail} = \text{TRUE}, n = 8)
```

---

deg2rad

**Convert degrees to radians**

**Description**

Convert degrees to radians

**Usage**

\[
deg2rad(deg)
\]

**Arguments**

- \( \text{deg} \) vector of degrees to be converted to radians
denoise

Examples

data(vdendool)

## latitude in degrees
degrees <- coordinates(vdendool)[, 2]
head(degrees)

## latitude in radians
radians <- deg2rad(coordinates(vdendool)[, 2])
head(radians)

denoise | Noise filtering through principal components

Description

Filter noise from a RasterStack by decomposing into principal components and subsequent reconstruction using only a subset of components.

Usage

denoise(x, k = NULL, expl.var = NULL, weighted = TRUE, ...)

Arguments

x 
RasterStack to be filtered

k 
number of components to be kept for reconstruction (ignored if expl.var is supplied)

expl.var 
minimum amount of variance to be kept after reconstruction (should be set to NULL or omitted if k is supplied)

weighted 
logical. If TRUE the covariance matrix will be geographically weighted using the cosine of latitude during decomposition (only important for lat/lon data)

... 
additional arguments passed to princomp

Value

a denoised RasterStack

See Also

anomalize, deseaseon
Examples

data("vdendool")
vdendool <- denoise(vdendool, expl.var = 0.8)

opar <- par(mfrow = c(1,2))
plot(vdendool[[1]], main = "original")
plot(vdd_dns[[1]], main = "denoised")
par(opar)

---

deseason  
Create seasonal anomalies

Description

The function calculates anomalies of a RasterStack by supplying a suitable seasonal window. E. g. to create monthly anomalies of a raster stack of 12 layers per year, use `cycle.window = 12`.

Usage

deseason(x, cycle.window = 12, ...)

Arguments

- **x**: a RasterStack
- **cycle.window**: the window for the creation of the anomalies
- ... currently not used

Value

a deseasoned RasterStack

See Also

anomalize, denoise

Examples

data("australiaGPCP")
aus_dsn <- deseason(australiaGPCP, 12)

opar <- par(mfrow = c(1,2))
plot(australiaGPCP[[1]], main = "original")
plot(aus_dsn[[1]], main = "deseasoned")
par(opar)
EOT analysis of a predictor and (optionally) a response RasterStack

Description

Calculate a given number of EOT modes either internally or between RasterStacks.

Usage

```r
## S4 method for signature 'RasterStack'
eot(x, y = NULL, n = 1, standardised = TRUE,
    write.out = FALSE, path.out = ".", prefix = "remote",
    reduce.both = FALSE, type = c("rsq", "ioa"), verbose = TRUE, ...)

## S4 method for signature 'RasterBrick'
eot(x, y = NULL, n = 1, standardised = TRUE,
    write.out = FALSE, path.out = ".", prefix = "remote",
    reduce.both = FALSE, type = c("rsq", "ioa"), verbose = TRUE, ...)
```

Arguments

- `x`: a RasterStack used as predictor
- `y`: a RasterStack used as response. If `y` is `NULL`, `x` is used as `y`
- `n`: the number of EOT modes to calculate
- `standardised`: logical. If `FALSE` the calculated r-squared values will be multiplied by the variance
- `write.out`: logical. If `TRUE` results will be written to disk using `path.out`
- `path.out`: the file path for writing results if `write.out` is `TRUE`. Defaults to current working directory
- `prefix`: optional prefix to be used for naming of results if `write.out` is `TRUE`
- `reduce.both`: logical. If `TRUE` both `x` and `y` are reduced after each iteration. If `FALSE` only `y` is reduced
- `type`: the type of the link function. Defaults to `"rsq"` as in original proposed method from van den Dool 2000. If set to `"ioa"` index of agreement is used instead
- `verbose`: logical. If `TRUE` some details about the calculation process will be output to the console
- `...`: not used at the moment

Details

For a detailed description of the EOT algorithm and the mathematics behind it, see the References section. In brief, the algorithm works as follows: First, the temporal profiles of each pixel `xp` of the predictor domain are regressed against the profiles of all pixels `xr` in the response domain. The calculated coefficients of determination are summed up and the pixel with the highest sum is
identified as the 'base point' of the first/leading mode. The temporal profile at this base point is the first/leading EOT. Then, the residuals from the regression are taken to be the basis for the calculation of the next EOT, thus ensuring orthogonality of the identified teleconnections. This procedure is repeated until a predefined amount of \( n \) EOTS is calculated. In general, \texttt{remote} implements a 'brute force' spatial data mining approach to identify locations of enhanced potential to explain spatio-temporal variability within the same or another geographic field.

**Value**

if \( n = 1 \) an \texttt{EotMode}, if \( n > 1 \) an \texttt{EotStack} of \( n \) \texttt{EotModes}. Each \texttt{EotMode} has the following components:

- \textit{mode} - the number of the identified mode (1 - \( n \))
- \textit{eot} - the EOT (time series) at the identified base point. Note, this is a simple numeric vector, not of class \texttt{ts}
- \textit{coords_bp} - the coordinates of the identified base point
- \textit{cell_bp} - the cell number of the identified base point
- \textit{cum_exp_var} - the (cumulative) explained variance of the considered EOT
- \textit{r_predictor} - the \texttt{RasterLayer} of the correlation coefficients between the base point and each pixel of the predictor domain
- \textit{rsq_predictor} - as above but for the coefficient of determination
- \textit{rsq_sums_predictor} - as above but for the sums of coefficient of determination
- \textit{int_predictor} - the \texttt{RasterLayer} of the intercept of the regression equation for each pixel of the predictor domain
- \textit{slp_predictor} - same as above but for the slope of the regression equation for each pixel of the predictor domain
- \textit{p_predictor} - the \texttt{RasterLayer} of the significance (p-value) of the the regression equation for each pixel of the predictor domain
- \textit{resid_predictor} - the \texttt{RasterBrick} of the reduced data for the predictor domain

Apart from \textit{rsq_sums_predictor}, all \_predictor fields are also returned for the \_response domain, even if predictor and response domain are equal. This is due to that fact, that if not both fields are reduced after the first EOT is found, these \texttt{RasterLayers} will differ.

**Methods (by class)**

- \texttt{RasterBrick}:  

**References**

**Empirical Orthogonal Teleconnections**  
Journal of Climate, Volume 13, Issue 8, pp. 1421-1435  
\url{http://journals.ametsoc.org/doi/abs/10.1175/1520-0442%282000%29013%3C1421%3AEOT%3E2.0.CO%3B2}

**Empirical methods in short-term climate prediction**  
H. M. van den Dool (2007)
EotCycle

Examples

```r
### EXAMPLE I
### a single field
data(vdendool)

## calculate 2 leading modes
nh_modes <- eot(x = vdendool, y = NULL, n = 2,
    reduce.both = FALSE, standardised = FALSE,
    verbose = TRUE)

plot(nh_modes, y = 1, show.bp = TRUE)
plot(nh_modes, y = 2, show.bp = TRUE)
```

EotCycle (Calculate a single EOT)

Description

EotCycle() calculates a single EOT and is controlled by the main eot() function

Usage

```r
EotCycle(x, y, y.eq.x = FALSE, n = 1, standardised, orig.var, write.out,
    path.out, prefix, type, verbose, ...)
```

Arguments

- **x**: a raster stack used as predictor
- **y**: a RasterStack used as response. If y is NULL, x is used as y
- **y.eq.x**: logical. Whether predictor and response stack are the same
- **n**: the number of EOT modes to calculate
- **standardised**: logical. If FALSE the calculated r-squared values will be multiplied by the variance
- **orig.var**: original variance of the response domain
- **write.out**: logical. If TRUE results will be written to disk using path.out
- **path.out**: the file path for writing results if write.out is TRUE. Defaults to current working directory
- **prefix**: optional prefix to be used for naming of results if write.out is TRUE
- **type**: the type of the link function. Defaults to 'rsq' as in original proposed method from Dool2000. If set to 'ioa' index of agreement is used instead
- **verbose**: logical. If TRUE some details about the calculation process will be output to the console
- **...**: not used at the moment
Class EotMode

Description

Class EotMode

Slots

mode  the number of the identified mode
name  the name of the mode
eot   the EOT (time series) at the identified base point. Note, this is a simple numeric vector
coords_bp  the coordinates of the identified base point
cell_bp  the cell number of the identified base point
cum_exp_var  the cumulative explained variance of the considered EOT mode
r_predictor  the RasterLayer of the correlation coefficients between the base point and each pixel of the predictor domain
rsq_predictor  as above but for the coefficient of determination of the predictor domain
rsq_sums_predictor  as above but for the sums of coefficient of determination of the predictor domain
int_predictor  the RasterLayer of the intercept of the regression equation for each pixel of the predictor domain
slp_predictor  same as above but for the slope of the regression equation for each pixel of the predictor domain
p_predictor  the RasterLayer of the significance (p-value) of the regression equation for each pixel of the predictor domain
resid_predictor  the RasterBrick of the reduced data for the predictor domain
r_response  the RasterLayer of the correlation coefficients between the base point and each pixel of the response domain
rsq_response  as above but for the coefficient of determination of the response domain
int_response  the RasterLayer of the intercept of the regression equation for each pixel of the response domain
slp_response  as above but for the slope of the regression equation for each pixel of the response domain
p_response  same the RasterLayer of the significance (p-value) of the regression equation for each pixel of the response domain
resid_response  the RasterBrick of the reduced data for the response domain
Class EotStack

Description

Class EotStack

Slots

- modes a list containing the individual 'EotMode's of the 'EotStack'
- names the names of the modes

geoWeight Geographic weighting

Description

The function performs geographic weighting of non-projected long/lat data. By default it uses the cosine of latitude to compensate for the area distortion, though the user can supply other functions via f.

Usage

geoWeight(x, f = function(x) cos(x), ...)

Arguments

- x a Raster* object
- f a function to be used to the weighting. Defaults to cos(x)
- ... additional arguments to be passed to f

Value

a weighted Raster* object

Examples

data(vdendoool)

wght <- geoWeight(vdendoool)

opar <- par(mfrow = c(1,2))
plot(vdendoool[[1]], main = "original")
plot(wght[[1]], main = "weighted")
par(opar)
### getWeights

*Calculate weights from latitude*

**Description**

Calculate weights using the cosine of latitude to compensate for area distortion of non-projected lat/lon data.

**Usage**

```r
getWeights(x, f = function(x) cos(x), ...)
```

**Arguments**

- **x**
  - a Raster* object
- **f**
  - a function to be used to the weighting. Defaults to `cos(x)`
- **...**
  - additional arguments to be passed to `f`

**Value**

a numeric vector of weights

**Examples**

```r
data("australiaGPCP")
wghts <- getWeights(australiaGPCP)
wghts_rst <- australiaGPCP[[1]]
wghts_rst[] <- wghts

opar <- par(mfrow = c(1,2))
plot(australiaGPCP[[1]], main = "data")
plot(wghts_rst, main = "weights")
par(opar)
```

---

### lagalize

*Create lagged RasterStacks*

**Description**

The function is used to produce two lagged RasterStacks. The second is cut from the beginning, the first from the tail to ensure equal output lengths (provided that input lengths were equal).

**Usage**

```r
lagalize(x, y, lag = NULL, freq = 12, ...)
```
names

Arguments

x       a RasterStack (to be cut from tail)
y       a RasterStack (to be cut from beginning)
lag      the desired lag (in the native frequency of the RasterStack)
freq      the frequency of the RasterStacks
...  currently not used

Value

a list with the two RasterStacks lagged by lag

Examples

data(pacificSST)
data(australiaGPCP)

# lag GPCP by 4 months
lagged <- lagalize(pacificSST, australiaGPCP, lag = 4, freq = 12)
lagged[[1]][[1]] # check names to see date of layer
lagged[[2]][[1]] # check names to see date of layer

---

names          Names of Eot* objects

Description

Get or set names of Eot* objects

Usage

## S4 method for signature 'EotStack'
names(x)

## S4 replacement method for signature 'EotStack'
names(x) <- value

## S4 method for signature 'EotMode'
names(x)

## S4 replacement method for signature 'EotMode'
names(x) <- value

Arguments

x       a EotMode or EotStack
value       a character vector of up to the same length as x, or NULL.
Value

if x is a EotStack, the names of all modes, if x is a EotMode, the name the respective mode

Examples

data(vdendool)

nh_modes <- eot(vdendool, n = 2)

## mode names
names(nh_modes)
names(nh_modes) <- c("vdendool1", "vdendool2")

names(nh_modes)
names(nh_modes[[2]])

<table>
<thead>
<tr>
<th>nmodes</th>
<th>Number of modes of an EotStack</th>
</tr>
</thead>
</table>

Description

Number of modes of an EotStack

Usage

## S4 method for signature 'EotStack'

nmodes(x)

Arguments

x an EotStack

Details

retrieves the number of modes of an EotStack

Value

integer

Examples

data(vdendool)

nh_modes <- eot(vdendool, n = 2)

nmodes(nh_modes)
**Description**

The function identifies the number of modes needed to explain a certain amount of variance within the response field.

**Usage**

```r
## S4 method for signature 'EotStack'
nXplain(x, var = 0.9)
```

**Arguments**

- `x`: an `EotStack`
- `var`: the minimum amount of variance to be explained by the modes

**Value**

An integer denoting the number of EOTs needed to explain `var`

**Note**

This is a post-hoc function. It needs an `EotStack` created as returned by `eot`. Depending on the potency of the identified EOTs, it may be necessary to compute a high number of modes in order to be able to explain a large enough part of the variance.

**Examples**

```r
data(vdendool)

nh_modes <- eot(x = vdendool, y = NULL, n = 5,
reduce.both = FALSE, standardised = FALSE,
verbose = TRUE)

## How many modes are needed to explain 40% of variance?
nXplain(nh_modes, 0.4)
```
pacificSST  

*Monthly SSTs for the tropical Pacific Ocean*

**Description**

Monthly NOAA sea surface temperatures for the tropical Pacific Ocean from 1982/01 to 2010/12

**Format**

a RasterBrick with the following attributes

- dimensions: 30, 140, 4200, 348 (nrow, ncol, ncell, nlayers)
- resolution: 1, 1 (x, y)
- extent: 150, 290, -15, 15 (xmin, xmax, ymin, ymax)
- coord. ref.: +proj=longlat +ellps=WGS84 +towgs84=0,0,0,0,0,0,0 +no_defs

**Details**

Monthly NOAA sea surface temperatures for the tropical Pacific Ocean from 1982/01 to 2010/12

**References**

Daily High-Resolution-Blended Analyses for Sea Surface Temperature
Reynolds et al. (2007)
Journal of Climate, Volume 20, Issue 22, pp. 5473 - 5496
http://dx.doi.org/10.1175/2007JCLI1824.1

---

**plot**  

*Plot an Eot* object

**Description**

This is the standard plotting routine for the results of `eot`. Three panels will be drawn i) the predictor domain, ii) the response domain, iii) the time series at the identified base point

**Usage**

```r
## S4 method for signature 'EotMode,ANY'
plot(x, y, pred.prm = "rsq", resp.prm = "r",
     show.bp = FALSE, anomalies = TRUE, add.map = TRUE, ts.vec = NULL,
     arrange = c("wide", "long"), clr = NULL, locations = FALSE, ...)

## S4 method for signature 'EotStack,ANY'
plot(x, y, pred.prm = "rsq", resp.prm = "r",
     show.bp = FALSE, anomalies = TRUE, add.map = TRUE, ts.vec = NULL,
     arrange = c("wide", "long"), clr = NULL, locations = FALSE, ...)
```
plot

Arguments

- **x**: either an object of EotMode or EotStack as returned by `eot`
- **y**: integer or character of the mode to be plotted (e.g. 2 or "mode_2")
- **pred.prm**: the parameter of the predictor to be plotted.
  - Can be any of "r", "rsq", "rsq.sums", "p", "int" or "slp"
- **resp.prm**: the parameter of the response to be plotted.
  - Can be any of "r", "rsq", "rsq.sums", "p", "int" or "slp"
- **show.bp**: logical. If TRUE a grey circle will be drawn in the predictor image to indicate the location of the base point
- **anomalies**: logical. If TRUE a reference line will be drawn a 0 in the EOT time series
- **add.map**: logical. If TRUE country outlines will be added to the predictor and response images
- **ts.vec**: an (optional) time series vector of the considered EOT calculation to be shown as the x-axis in the time series plot
- **arrange**: whether the final plot should be arranged in "wide" or "long" format
- **clr**: an (optional) color palette for displaying of the predictor and response fields
- **locations**: logical. If x is an EotStack, set this to TRUE to produce a map showing the locations of all modes. Ignored if x is an EotMode
- ... further arguments to be passed to `spplot`

Methods (by class)

- **x = EotStack, y = ANY:**

Examples

```r
data(vdendool)

## Calculate 2 leading modes
nh_modes <- eot(x = vdendool, y = NULL, n = 2,
    reduce.both = FALSE, standardised = FALSE,
    verbose = TRUE)

## Default settings
plot(nh_modes, y = 1) # is equivalent to
plot(nh_modes[[1]])

plot(nh_modes, y = 2) # shows variance explained by mode 2 only
plot(nh_modes[[2]]) # shows cumulative variance explained by modes 1 & 2

## Showing the location of the mode
plot(nh_modes, y = 1, show.bp = TRUE)

## Changing parameters
plot(nh_modes, y = 1, show.bp = TRUE,
    pred.prm = "r", resp.prm = "p")
```
## predict

### EOT based spatial prediction

**Description**

Make spatial predictions using the fitted model returned by `eot`. A (user-defined) set of $n$ modes will be used to model the outcome using the identified link functions of the respective modes which are added together to produce the final prediction.

**Usage**

```r
## S4 method for signature 'EotStack'
predict(object, newdata, n = 1, ...)
```

```r
## S4 method for signature 'EotMode'
predict(object, newdata, n = 1, ...)
```

**Arguments**

- `object` an Eot* object
- `newdata` the data to be used as predictor
- `n` the number of modes to be used for the prediction. See `nXplain` for calculating the number of modes based on their explanatory power.
- `...` further arguments to be passed to `calc`

**Value**

a `RasterStack` of `nlayers(newdata)`

**Methods (by class)**

- `EotMode`:

**Examples**

```r
## not very useful, but highlights the workflow
data(pacificSST)
data(australiaGPCP)

## train data using eot()
train <- eot(x = pacificSST[[1:10]],
             y = australiaGPCP[[1:10]],
```
## subset

n = 1)

```r
## predict using identified model
pred <- predict(train,
               newdata = pacificSST[[11:20]],
               n = 1)
```

```r
## compare results
opar <- par(mfrow = c(1,2))
plot(australiaGPCP[[13]], main = "original", zlim = c(0, 10))
plot(pred[[3]], main = "predicted", zlim = c(0, 10))
par(opar)
```

## Subset modes in EotStacks

### Description

Extract a set of modes from an EotStack

### Usage

```r
## S4 method for signature 'EotStack'
subset(x, subset, drop = FALSE, ...)
```

```r
## S4 method for signature 'EotStack,ANY,ANY'
x[[1]]
```

### Arguments

- `x`  
  EotStack to be subset

- `subset`  
  integer or character. The modes to extract (either by integer or by their names)

- `drop`  
  if TRUE a single mode will be returned as an EotMode

- `...`  
  currently not used

- `i`  
  indices specifying elements to extract or replace. Indices are numeric or character vectors or empty (missing) or NULL. Numeric values are coerced to integer as by `as.integer` (and hence truncated towards zero). Character vectors will be matched to the `names` of the object (or for matrices/arrays, the `dimnames`): see ‘Character indices’ below for further details.

For `[<-`indexing only: `i, j, ...` can be logical vectors, indicating elements/slices to select. Such vectors are recycled if necessary to match the corresponding extent. `i, j, ...` can also be negative integers, indicating elements/slices to leave out of the selection.

When indexing arrays by `[ a single argument `i` can be a matrix with as many columns as there are dimensions of `x`; the result is then a vector with elements corresponding to the sets of indices in each row of `i`.

An index value of NULL is treated as if it were `integer(0)`.
Value

an Eot* object

Methods (by class)

• x = EotStack, i = ANY, j = ANY:

Examples

data(vdendool)

nh_modes <- eot(x = vdendool, y = NULL, n = 3,
    reduce_both = FALSE, standardised = FALSE,
    verbose = TRUE)

subs <- subset(nh_modes, 2:3) # is the same as
subs <- nh_modes[[2:3]]

## effect of 'drop=FALSE' when selecting a single layer
subs <- subset(nh_modes, 2)
class(subs)
subs <- subset(nh_modes, 2, drop = TRUE)
class(subs)

vdendool  Mean seasonal (DJF) 700 mb geopotential heights

Description

NCEP/NCAR reanalysis data of mean seasonal (DJF) 700 mb geopotential heights from 1948 to 1998

Format

a RasterBrick with the following attributes

dimensions : 14, 36, 504, 50 (nrow, ncol, ncell, nlayers)
resolution : 10, 4.931507 (x, y)
extent : -180, 180, 20.9589, 90 (xmin, xmax, ymin, ymax)
coord. ref. : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0

Details

NCEP/NCAR reanalysis data of mean seasonal (DJF) 700 mb geopotential heights from 1948 to 1998
writeEot

Source

http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis.derived.pressure.html

Original Source: NOAA National Center for Environmental Prediction

References

The NCEP/NCAR 40-year reanalysis project
Kalnay et al. (1996)

writeEot

Write Eot* objects to disk

Description

Write Eot* objects to disk. This is merely a wrapper around writeRaster so see respective help section for details.

Usage

## S4 method for signature 'EotMode'
writeEot(x, path.out = ".\", prefix = "remote",
     overwrite = TRUE, ...)

## S4 method for signature 'EotStack'
writeEot(x, path.out = ".\", prefix, ...)

Arguments

x an Eot* object
path.out the path to the folder to write the files to
prefix a prefix to be added to the file names (see Details)
overwrite see writeRaster. Defaults to TRUE in writeEot
... further arguments passed to writeRaster

Details

writeEot will write the results of either an EotMode or an EotStack to disk. For each mode the following files will be written:

- **pred_r** - the RasterLayer of the correlation coefficients between the base point and each pixel of the predictor domain
- **pred_rsq** - as above but for the coefficient of determination
writeEot

- `pred_rsq_sums` - as above but for the sums of coefficient of determination
- `pred_int` - the `RasterLayer` of the intercept of the regression equation for each pixel of the predictor domain
- `pred_slp` - same as above but for the slope of the regression equation for each pixel of the predictor domain
- `pred_p` - the `RasterLayer` of the significance (p-value) of the regression equation for each pixel of the predictor domain
- `pred_resid` - the `RasterBrick` of the reduced data for the predictor domain

Apart from `pred_rsq_sums`, all these files are also created for the response domain as `resp_*`. These will be pasted together with the prefix & the respective mode so that the file names will look like, e.g.:

```
prefix_mode_n_pred_r.grd
```

for the `RasterLayer` of the predictor correlation coefficient of mode n using the standard raster file type (.grd).

Methods (by class)

- `EotStack`:

See Also

`writeRaster`

Examples

data(vdendool)

```r
nh_modes <- eot(x = vdendool, y = NULL, n = 2, reduce.both = FALSE, standardised = FALSE, verbose = TRUE)
```

```r
## write the complete EotStack
writeEot(nh_modes, prefix = "vdendool")
```

```r
## write only one EotMode
writeEot(nh_modes[[2]], prefix = "vdendool")
```
Index

*Topic package
  remote-package. 2
[[,EotStack,ANY,ANY-method (subset), 21
  anomalize, 3, 7, 8
  as.integer, 21
  australiaGPCP, 4
  calc, 3, 20
  calcVar, 4
  cov.wt, 5
  covWeight, 5
  cutStack, 6
  deg2rad, 6
  denoise, 3, 7, 8
  deseason, 3, 7, 8
  dimnames, 21
  eot, 9, 17–19
  eot, RasterBrick-method (eot), 9
  eot, RasterStack-method (eot), 9
  EotCycle, 11
  EotMode-class, 12
  EotStack-class, 13
  geoWeight, 13
  getValues, 5
  getWeights, 5, 14
  lagalize, 14
  names, 15, 21
  names, EotMode-method (names), 15
  names, EotStack-method (names), 15
  names<- (names), 15
  names<- , EotMode-method (names), 15
  names<- , EotStack-method (names), 15
  nmodes, 16
  nmodes, EotStack-method (nmodes), 16
  nXplain, 17, 20
  nXplain, EotStack-method (nXplain), 17
  pacificSST, 18
  plot, 18
  plot, EotMode, ANY-method (plot), 18
  plot, EotStack, ANY-method (plot), 18
  predict, 20
  predict, EotMode-method (predict), 20
  predict, EotStack-method (predict), 20
  princomp, 7
  remote (remote-package), 2
  remote-package, 2
  spplot, 19
  subset, 21
  subset, EotStack-method (subset), 21
  vdendool, 22
  writeEot, 23
  writeEot, EotMode-method (writeEot), 23
  writeEot, EotStack-method (writeEot), 23
  writeRaster, 23, 24