Package 'chronosphere'

October 12, 2022

Type Package

Title Earth System History Variables

Version 0.4.1

Author Adam T. Kocsis, Nussaibah B. Raja

Collate zzz.R classes.R data.R colours.R reconstruction.R utility.R platemodel.R fetching.R info.R XArray-base.R XArray-combine.R XArray-xattrib.R XArray-subset.R XArray-apply.R RasterArray-base.R RasterArray-cast.R RasterArray-combine.R RasterArray-groupgen.R RasterArray-raster.R RasterArray-subset.R RasterArray-xattrib.R SpatialStack-base.R SpatialStack-sp.R SpatialStack-subset.R SpatialArray-base.R SpatialArray-sp.R SpatialArray-cast.R SpatialArray-combine.R SpatialArray-subset.R inter.R plotting.R

Maintainer Adam T. Kocsis <adam.t.kocsis@gmail.com>

Description The purpose of the 'chronosphere' project is to facilitate spatially explicit analyses of (paleo)environmental/ecological research. The package serves as a gateway to plate tectonic reconstructions, deep time global climate model results as well as fossil occurrence datasets such as the Paleobiology Database <https://paleobiodb.org/> and the Paleo-Reefs Database <https://www.paleo-reefs.pal.uni-erlangen.de/>. Environmental data stored on a remote server can be downloaded and imported directly to the R environment. Query functions to the GPlates <https://www.gplates.org/> desktop application or the GPlates Web Service <https://gws.gplates.org/> allow users to reconstruct coordinates, static plates, and Spatial objects. A wrapper class 'RasterArray' is implemented around the 'RasterLayer' class, allowing the organization of spatially explicit raster data in n-dimensional arrays. The project is developed under the umbrella of the DFG (Deutsche Forschungsgemeinschaft) Research Unit TERSANE2 (For 2332, TEmperature Related Stressors in ANcient Extinctions).

License CC BY 4.0

Date 2021-04-16

BugReports https://github.com/chronosphere-portal/r_package/issues

Encoding UTF-8

LazyData false

Depends R (>= 3.5.0), raster, sp

Imports graphics, methods, grDevices, utils

NeedsCompilation no

RoxygenNote 7.1.1

VignetteBuilder knitr

Suggests jpeg, knitr, ncdf4, rgdal, rmarkdown

Repository CRAN

Date/Publication 2021-04-18 21:50:07 UTC

R topics documented:

`aggregate` *Aggregate raster cells in a* `RasterArray` *object*

Description

The method is inherited from the `RasterStack` class.

Usage

```
## S4 method for signature 'RasterArray'
aggregate(x, ...)
```

Arguments

x	a RasterArray-class object.
...	arguments passed to the aggregate function.

Value

An aggregated RasterArray class object.

Examples

```
data(dems)
agg <- aggregate(dems, 5)
```

---

apply *Apply-type iterator for RasterArrays and SpatialArrays*

Description

The function implements the apply-type iterators for the RasterArray class. Output values are constrained to RasterArrays, whenever possible. Not yet implemented for multidimensional MARGINs.

Arguments

X	an array, including matrices and RasterArrays.
MARGIN	a vector giving the subscripts which the function will be applied over. E.g., for a matrix 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns. Where X has named dimnames, it can be a character vector selecting dimension names. For RasterArrays only single dimension margins are implemented. If it is NULL then the function is applied to every item.
FUN	the function to be applied: see 'Details' of apply.
...	optional arguments passed to FUN.

Format

An object of class standardGeneric of length 1.

Value

Depending on the on the output of FUN, a list, a vector or RasterArray or SpatialArray object.

Examples

```
# Null dimensional margin
data(coasts)
# apply function to every element manually
# memory taken by every layer
apply(coasts, MARGIN=NULL, object.size)
# double of itself
data(dems)
a<- cbind(dems, dems)
same <- apply(a, 1, sum)
```

---

as *Coerce RasterLayer, RasterStack and RasterBrick object to a RasterArray*

Description

The function coerces RasterLayer, RasterStack and RasterBrick object to a RasterArray.

Arguments

from	Object to be coerced.

Value

A RasterArray class object.

---

as.data.frame.RasterArray *S3-type method for* RasterArray *and* SpatialArray

---

Description

Convert RasterArray class objects to data.frames allowing View(), head() and tail() to work.

Usage

```
## S3 method for class 'RasterArray'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S3 method for class 'SpatialArray'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	a RasterArray and SpatialArray class object.
row.names	Argument to define the rownames of the resulting data.frame.
optional	logical Flag to reset the rownames and colnaems attributes.
...	additional arguments passed to and from methods.

Details

Formal conversion method transforming the proxy object to a data.frame.

Value

A data.frame class object.

Examples

```
data(dems)
df <- as.data.frame(dems)
```

---

as.list,RasterArray-method

Coerce a RasterArray class object to a list

Description

The function will return the items of the RasterArray as a list (conserving the names of the elements in the stack).

Usage

```
## S4 method for signature 'RasterArray'
as.list(x, ...)
```

Arguments

x	A RasterArray class object.
...	arguments passed to or from methods.

Value

A list of RasterLayers.

as.list,SpatialArray-method

Coerce a SpatialArray class object to a list

Description

The function will return the items of the SpatialArray as a list (conserving the names of the elements in the stack).

Usage

```
## S4 method for signature 'SpatialArray'
as.list(x, ...)
```

Arguments

x	A \codeSpatialArray class object.
...	arguments passed to or from methods.

Value

A list of RasterLayers.

as.RasterArray

Convert Raster objects to RasterArrays*

Description

The function converts RasterLayer, RasterStack and RasterBrick objects to RasterArray class objects.

Usage

```
as.RasterArray(from)

## S4 method for signature 'RasterLayer'
as.RasterArray(from)

## S4 method for signature 'RasterStack'
as.RasterArray(from)

## S4 method for signature 'RasterBrick'
as.RasterArray(from)

## S4 method for signature 'SpatialPoints'
```

```
as.SpatialArray(from)

## S4 method for signature 'SpatialPointsDataFrame'
as.SpatialArray(from)

## S4 method for signature 'SpatialLines'
as.SpatialArray(from)

## S4 method for signature 'SpatialLinesDataFrame'
as.SpatialArray(from)

## S4 method for signature 'SpatialPolygons'
as.SpatialArray(from)

## S4 method for signature 'SpatialPolygonsDataFrame'
as.SpatialArray(from)
```

Arguments

from	Object to be converted.

Value

A RasterArray class object.

Examples

```
data(dems)
recent <- dems[1]
# convert RasterLayer to RasterArray
ra <- as.RasterArray(recent)
```

---

as.SpatialArray *Convert Spatial* objects to SpatialArrays*

Description

The function converts Spatial*objects to SpatialArray-class objects.

Usage

```
as.SpatialArray(from)
```

Arguments

from	Object to be converted.

Value

A SpatialArray class object.

Examples

```
# data(coasts)
# recent <- coasts[1]
# # convert Spatial* to as.SpatialArray
# sa <- as.SpatialArray(recent)
```

---

calc,RasterArray,function-method

Calculate method for the RasterArray object

Description

Calculate values for a new RasterLayer/RasterArray object from another RasterArray object, using a formula.

Usage

```
## S4 method for signature 'RasterArray,`function`'
calc(x, fun, margin = NULL, na.rm = NULL, forcefun = FALSE, forceapply = FALSE)
```

Arguments

x	A RasterArray class object.
fun	function to be applied.
margin	The MARGIN parameter of the apply function. If set to NULL then the fun will be applied to the entire stack, producing a single layer.
na.rm	Remove NA values, if supported by 'fun' (only relevant when summarizing a multilayer Raster object into a RasterLayer)
forcefun	logical. Force calc to not use fun with apply; for use with ambiguous functions and for debugging (see Details)
forceapply	logical. Force calc to use fun with apply; for use with ambiguous functions and for debugging (see Details)

Details

The method is an extension of the calc function. The strucuture expressed as the RasterArray's dimensions allows the calculations to be iterated for different margins of the array, similarly to the apply function, controlled by the margin argument.

Value

A RasterLayer or RasterArray class object.

Examples

```
data(dems)

d2 <- cbind(dems, dems)
double <- calc(d2, margin=1, fun=sum)
```

---

cbind.RasterArray *Combine Raster and Vector Spatial data to* RasterArray *or* SpatialArray *objects by rows or columns*

Description

The function takes a sequence of RasterLayer or RasterArray class objects and combines them to two dimensional RasterArrays, Alternatively, the function can be used to combine vector Spatial* data to SpatialArrays. Named objects will be forced together based on names, colnames or rownames attributes, via insertion of NAs.

Usage

```
## S3 method for class 'RasterArray'
cbind(...)

## S3 method for class 'RasterArray'
rbind(...)

## S3 method for class 'SpatialArray'
cbind(...)

## S3 method for class 'SpatialArray'
rbind(...)
```

Arguments

...	RasterLayer or RasterArray class objects to be combined.

Value

A RasterArray or SpatialArray class object.

Examples

```
data(dems)
# create matrices out of vectors
colb <- cbind(dems, dems)
rowb <- rbind(dems, dems)
# automatic name matching
dems2 <- dems[c(1:4, 6:10)]
matched <- suppressWarnings(cbind(dems, dems2))
```

cellStats,RasterArray-method

Statistics across cells in a RasterArray object

Description

The method is inherited from the RasterStack class. Postitions of layers are conserved in the output. (including missing layers)

Usage

```
## S4 method for signature 'RasterArray'
cellStats(x, stat, ...)
```

Arguments

x	a RasterArray class object.
stat	A function to be applied.
...	arguments passed to the cellStats function.

Value

A set of the values matching the output of stat, organized the same way as the RasterArray.

Examples

```
data(dems)
cellStats(dems, stat=mean, na.rm=TRUE)
```

chronosphere

Earth System History Variables

Description

The purpose of the 'chronosphere' project is to facilitate spatially explicit analyses of (paleo)environmental/ecological research. The package serves as a gateway to plate tectonic reconstructions, deep time global climate model results as well as fossil occurrence datasets such as the Paleobiology Database https://paleobiodb.org/ and the PaleoReefs Database https://www.paleo-reefs.pal.uni-erlangen.de/. Environmental data stored on a remote server can be downloaded and imported directly to the R environment. Query functions to the GPlates (https://www.gplates.org/) desktop application or the GPlates Web Service (https://gws.gplates.org/) allow users to reconstruct coordinates, static plates, and Spatial objects. A wrapper class RasterArray is implemented around the RasterLayer class, allowing the organization of spatially explicit raster data in n-dimensional arrays. The project is developed under the umbrella of the DFG (Deutsche Forschungsgemeinschaft) Research Unit TERSANE2 (For 2332, TEmperature Related Stressors in ANcient Extinctions).

Details

This is still the Beta version. As is R, this is free software and comes with ABSOLUTELY NO WARRANTY. Nevertheless, notes about found bugs and suggestions are more than welcome.

Author(s)

Adam T. Kocsis (adam.t.kocsis@gmail.com) and Nussaibah B. Raja

---

coasts *PaleoMAP PaleoCoastlines (demo)*

Description

A dataset containing the coastline reconstructions based on the PaleoMAP PaleoDEMS (dems) and the Paleobiology Database https://paleobiodb.org for the time interval 0 - 25 Ma.

Usage

```
data(coasts)
```

Format

A SpatialArray with 5 continental margin and 5 paleocoastline layers.

Details

This is version v7. The article describing the entire set is under review. Once that is published, the entire dataset will be available.

Source

Kocsis, A. T., & Scotese, C. R. (2020). PaleoMAP PaleoCoastlines data [Data set]. Zenodo. https://doi.org/10.5281/zenodo.3903164

---

colnames,XArray-method

Column names of two-dimensional RasterArray *or* SpatialArray

---

Description

Get or set the column names of two-dimensional codeRasterArray or SpatialArray objects

Usage

```
## S4 method for signature 'XArray'
colnames(x)

## S4 replacement method for signature 'XArray'
colnames(x) <- value
```

Arguments

x	RasterArray or SpatialArray object.
value	character vector.

Value

A character vector of column names or NULL.

Examples

```
data(coasts)
colnames(coasts)
colnames(coasts) <- c("a", "b")
```

---

combine — *Combine RasterLayers and one-dimensional RasterArrays and Spatial* objects with one-dimensinoal SpatialArrays*

Description

Methods sequences that start with an NA do not yet work.

Usage

```
combine(x, ...)

## S4 method for signature 'XArray'
combine(x, ...)

## S4 method for signature 'RasterLayer'
combine(x, ...)

## S4 method for signature 'VectorSpatialClasses'
combine(x, ...)
```

Arguments

x	RasterLayer or RasterArray objects or Spatial* and SpatialArray objects to combine.
...	additional objects to combine.

Value

A RasterArray or SpatialArrayclass object.

Examples

```
data(dems)
a <- combine(dems[1], dems[2])
```

---

crop,RasterArray-method

Crop a RasterArray object

Description

The method is inherited from the RasterStack class.

Usage

```
## S4 method for signature 'RasterArray'
crop(x, y, ...)
```

Arguments

x	a RasterArray class object.
y	an xtent object, or any object from which an Extent object can be extracted (see Details)
...	arguments passed to the crop function.

Value

A cropped RasterArray class object.

Examples

```
data(dems)
# crop to Australia
ext <- extent(c(
  xmin = 106.58,
  xmax = 157.82,
  ymin = -45.23,
  ymax = 1.14
))
# cropping all DEMS (Australia drifted in)
au<- crop(dems, ext)
```

`datasets` *Download a database extract from* `chronosphere` *remote server*

Description

The function will download a list of available data from the data repository

Usage

```
datasets(
  dat = NULL,
  datadir = NULL,
  verbose = FALSE,
  master = FALSE,
  greetings = TRUE
)
```

Arguments

`dat`	`character`. Database ID. If this is set to `NULL`, then a simplified list of availables variables will be downloaded, including all `dat` and `var` combinations. If `dat` is a valid database ID, then all accessible resolutions and version of a dataset are shown.
`datadir`	`character` Directory where the downloaded files are kept. Individual entries will be looked up from the directory if this is given, and will be downloaded if they are not found. The default `NULL` option will download data to a temporary directory that exists only until the R session ends.
`verbose`	`logical` Should console feedback during download be displayed?
`master`	`logical` When `dat` is `NULL`, should the function download the master records file?
`greetings`	`logical` When the function is invoked without arguments, it displays a message to keep new users informed about different versions and resolutions (even with `verbose=FALSE`). This argument turns this message off on demand.

Details

The function will download a single .csv file and attach it as a `data.frame`.

Value

A `data.frame` class object.

Examples

```
# available datasets and variables - proper
# ind <- datasets()
# available datasets and variables
# just one example archive is available locally
ind <- datasets(
  datadir=system.file("extdata", package="chronosphere"))
# all available versions and resolutions in database 'paleomap'
# oneDat <- datasets("paleomap")
```

---

dems *PaleoDEM rasters from the reconstructions Chris Scotese.*

Description

A dataset containing the paleoDEM reconstructions of Chris Scotese for the time interval 0 - 45 Ma.

Usage

```
data(dems)
```

Format

A RasterArray with 10 layers.

Details

This is a subset of the total dataset, use `fetch(dat="paleomap", var="dem", res=1, ver="20190719")` to get the whole set.

Source

Scotese, C. R. Wright, N. (2018). PALEOMAP Paleodigital Elevation Models (PaleoDEMS) for the Phanerozoic. URL: https://www.earthbyte.org/paleodem-resource-scotese-and-wright-2018/

dim,XArray-method *Dimensions of* SpatialArray *or* RasterArray *objects*

Description

The function returns the dimensions of the array in which RasterLayers or Spatial* objects are organized.

Usage

```
## S4 method for signature 'XArray'
dim(x)
```

Arguments

x	A SpatialArray or RasterArray class object.

Value

A numeric vector.

Examples

```
data(dems)
dim(dems)
data(coasts)
dim(coasts)
```

dimlayer *Dimensions of RasterLayers in a RasterArray object*

Description

The funcion will return the dimensions RasterLayers

Usage

```
dimlayer(x, ...)

## S4 method for signature 'RasterArray'
dimlayer(x)
```

Arguments

x	A RasterArray class object.
...	additional arguments passed to class-specific methods.

Value

A numeric vector with the number of rows and columns in the RasterLayers.

---

dimnames,XArray-method

Names of multidimensional RasterArray *or* SpatialArray *objects.*

Description

Get or set the dimnames of multidimensional RasterArray or SpatialArray objects

Usage

```
## S4 method for signature 'XArray'
dimnames(x)

## S4 replacement method for signature 'XArray'
dimnames(x) <- value
```

Arguments

x	RasterArray or SpatialArray object.
value	character vector.

Value

A list of character vectors or NULL.

Examples

```
data(dems)
dimnames(dems)
data(coasts)
dimnames(coasts)
dimnames(coasts)[[2]] <- c("first", "second")
```

disaggregate *Disaggregate raster cells in a RasterArray object*

Description

The method is inherited from the RasterStack class.

Usage

```
## S4 method for signature 'RasterArray'
disaggregate(x, ...)
```

Arguments

x	a RasterArray class object.
...	arguments passed to the disaggregate function.

Value

A disaggregated RasterArray class object.

Examples

```
data(dems)
disagg <- disaggregate(dems, 3)
```

extent *Extent of a RasterArray object*

Description

The method is inherited from the RasterStack class.

Usage

```
## S4 method for signature 'RasterArray'
extent(x, ...)
```

Arguments

x	a RasterArray-class object.
...	arguments passed to the extent function.

Value

An aggregated RasterArray class object.

Examples

```
data(dems)
agg <- extent(dems)
```

`extract` *Extraction of values from multiple RasterLayers in a RasterArray object*

Description

The function takes a set of time-dependent coordinates and extracts the value they point to from associted RasterLayers in a RasterArray.

Usage

```
extract

## S4 method for signature 'RasterArray,matrix'
extract(x, y)

## S4 method for signature 'RasterArray,data.frame'
extract(x, y, by = NULL, margin = 1, lng = "plng", lat = "plat", force = NULL)
```

Arguments

`x`	(`RasterArray`). A set of `RasterLayers` that are associated with entries (one dimension) or the rows of `x`.
`y`	(`matrix` or `data.frame`). The data table containing the coordinates and (optionally) the indices or names of the associated `RasterLayers` in `x`.
`by`	(`character` or `vector`) In case of a `data.frame` input, the link between `x` and `y`. If `by` is a `character` string then it is expected to be column of `x` and should contain the `names` or the indices of the associated `RasterLayers` in `x`. If it is a `vector` its length should match the number of rows in `x` and it will be used as if it were a column of `x`.
`margin`	(`numeric`) A single value describing which margin (dimension of `x`) `by` is referring to (1: rows, 2: columns, etc.).
`lng`	(`character`) A column of `x` that includes the paleolongitudes.
`lat`	(`character`) A column of `x` that includes the paleolatitudes.
`force`	(`character`) If set to `"numeric"` the `by` argument or the column it points to will be converted to numeric values, and `x` will be subsetted with `numeric` subscripts of the `x` `RasterArray`. If set to `"character"`, the by column (or vector) will be forced to `character` values and will be used as character subscripts.

Format

An object of class `standardGeneric` of length 1.

Value

A numeric vector, matrix or array of values.

Examples

```
# one pair of random coordinates from Africa
mat <- matrix(c(
  -1.34, 42.96
), ncol=2, byrow=TRUE)

# repeat four times
mat<- mat[rep(1,4), ]

# assign default names and age
df<- data.frame(plng=mat[, 1],plat=mat[, 2], age=c(1,3,5, 1))
rownames(df) <- paste("point", 1:nrow(df))

# first coordinate pair will be extrated from RasterLayer 1 ["0"]
# second coordinate pair will be extracted from RasterLayer 3 ["10"]
# thrid coordinate pair will be extracted from RasterLayer 5 ["20"]
# fourth coordinate pair will be extracted from RasterLayer 1 ["0"]
data(dems)
extract(dems, df, by="age")

# by=NULL will be implemented in the next update
# (all coordinates extracted from all layers)
```

---

fetch *Data fetching*

Description

Function to download and attach variables in the chronosphere package

Usage

```
fetch(
  dat,
  var = NULL,
  ver = NULL,
  res = NULL,
  datadir = NULL,
  verbose = TRUE,
  call = FALSE,
  call.expr = FALSE,
  ...
)
```

Arguments

dat	(character) The dataset to get variables from.
var	(character) Vector of variable names to get.
ver	(character) The version of the variable. Defaults to NULL, which will download the latest available version. We have to create a data table, which should be part of the package. This has to be searched for valid argument combinations. Right this is just a folder with a date.
res	(character or numeric) The resolution of raster layers. This has to be the same for all RasterLayers that make up the variable.
datadir	(character) Directory where downloaded files are kept. Individual layers will be looked up from the directory if this is given, and will be downloaded if they are not found. The default NULL option will download data to a temporary directory that exists only until the R session ends.
verbose	(logical) Should console feedback during download be displayed?
call	(logical) If set to TRUE the function call is returned instead of the object.
call.expr	(logical) If call is set to TRUE, then should the call be returned as an expression (TRUE) or a message (FALSE)?
...	Arguments passed to variable-specific loading functions.

Details

Use the function datasets to find available variables.

Value

An object that matches the 'type' field of the varibles in the output of the datasets function.

Examples

```
# An actual download call
# a <- fetch(dat="paleomap", var="dem")
# call repetition
fetch(dat="paleomap", var="dem", call=TRUE)
# A locally-present object, in package's directory
a <- fetch(dat="paleomap", var="model",
  datadir=system.file("extdata", package="chronosphere"))
```

---

info *Documentation page of a variable*

Description

This is a temporary function that takes the user to the Evolv-ED blog.

Usage

```
info(dat, var)
```

Arguments

dat	(characater) Object downloaded with fetch or the database identifier string.
var	(character) In case dat is character, the variable identifier.

`is.na.RasterArray` *Positions of missing values in a RasterArray object*

Description

The function behaves similar to the regular `is.na()` function applied to the proxy object of a `RasterArray`.

Usage

```
## S3 method for class 'RasterArray'
is.na(x)
```

Arguments

x	A RasterArray class object.

Value

A `logical` `vector`, `matrix` or `array` matching the structure of the `RasterArray`.

Examples

```
data(dems)
dems[2] <- NA
is.na(dems)
```

is.na.SpatialArray *Positions of missing values in a SpatialArray object*

Description

The function behaves similar to the regular `is.na()` function applied to the proxy object of a `RasterArray`.

Usage

```
## S3 method for class 'SpatialArray'
is.na(x)
```

Arguments

`x`	A `RasterArray` class object.

Value

A `logical vector`, `matrix` or `array` matching the structure of the `RasterArray`.

Examples

```
data(coasts)
coasts[2,1] <- NA
is.na(coasts)
```

layers *Names of RasterArray or SpatialArray Layers in the stack*

Description

Names of `RasterArray` or `SpatialArray` Layers in the stack

Usage

```
layers(x, ...)

## S4 method for signature 'XArray'
layers(x)
```

Arguments

`x`	A `RasterArray` or `SpatialArray` class object.
`...`	additional arguments passed to class-specific methods.

Value

A character vector of names.

Examples

```
# names of layers in the stack
data(dems)
layers(dems)
```

---

length,XArray-method *Number of elements or layers in a RasterArray or SpatialArray class object*

Description

Function to return the length of the array in which RasterLayers are organized.

Usage

```
## S4 method for signature 'XArray'
length(x)

## S4 method for signature 'XArray'
nlayers(x)

## S4 method for signature 'SpatialStack'
nlayers(x)
```

Arguments

x	a RasterArray or SpatialArray class object.

Details

The length() function returns the number elements that should be present based on the array structure itself, and not the total number of values stored in the object (such as the length method of RasterStacks). As the object can contain missing values, the number of actual layers can be queried with nlayers.

Value

A numeric value.

Examples

```
data(dems)
# omit third element
dems[3] <- NA
# number of elements in the RasterArray
length(dems)
# remaining number values in the stack
length(dems@stack)
# the number of remaining layers in the RasterArray
nlayers(dems)
```

---

mapedge *Function to quickly draft the edge of the equirectangular projection*

Description

Function to plot the edge of a map with different projections.

Usage

```
mapedge(x = 360, y = 180, xmin = -180, xmax = 180, ymin = -90, ymax = 90)
```

Arguments

x	(numeric) Number of segments in the x (longitude) dimension.
y	(numeric) Number of segments in the y (latitude) dimension.
xmin	(numeric) Minimum value of x (longitude).
xmax	(numeric) Minimum value of x (longitude).
ymin	(numeric) Maximum value of y (latitude).
ymax	(numeric) Maximum value of y (latitude).

Value

A SpatialPolygons class object.

Examples

```
# requires rgdal
edge <- mapedge()
molledge <- spTransform(edge, CRS("+proj=moll"))
```

mapplot *Wrapper function to plot maps of different classes*

Description

This function plots Raster and sp-type objects.

Usage

```
mapplot(x, ...)

## S4 method for signature 'RasterLayer'
mapplot(
  x,
  col = "gradinv",
  axes = FALSE,
  box = FALSE,
  legend = FALSE,
  legend.title = NULL,
  ...
)

## S4 method for signature 'RasterStack'
mapplot(x, col = gradinv(255), ...)

## S4 method for signature 'RasterArray'
mapplot(
  x,
  col = "gradinv",
  rgb = FALSE,
  legend = FALSE,
  axes = FALSE,
  box = FALSE,
  ncol = 3,
  legend.title = NULL,
  plot.title = NULL,
  rowlabels = rownames(x),
  multi = FALSE,
  ask = FALSE,
  ...
)

## S4 method for signature 'SpatialPolygonsDataFrame'
mapplot(x, col = "lightgrey", ...)

## S4 method for signature 'SpatialPolygons'
mapplot(x, col = "lightgrey", ...)
```

Arguments

x	Object to be plotted
...	arguments passed to class-specific methods.
col	(character) Color or color scheme of the plot. See ramps for available palettes (ipccLine and ipccRCP are not available).
axes	(logical) Should axes be displayed?
box	(logical) Should bounding boxes be displayed?
legend	(logical) Triggers whether the legend of a RasterLayer would be plotted.
legend.title	(character) Title for the legend, if legend = TRUE.
rgb	set to (TRUE) to make a red-green-blue plot based on three layers or bands.
ncol	numeric) Set number of columns in a multi-plot for a single variable. For a RasterArray with multiple variables, this number is automatically set to the number of variables.
plot.title	(character) The title for each individual plot. Only available for a single variable at the moment.
rowlabels	(character) label for each row of the overall plot. Uses the rownames of the RasterArray by default. Only availble for multivariate RasterArrays.
multi	(logical) Should the plots be printed on multiple pages?
ask	(logical or NULL) If multi.page = TRUE and ask = TRUE, then the user will be prompted before a new page of output is started

Value

None.

Examples

```
 #single variable
data(dems)
mapplot(dems, ncol=4)
```

---

mask,RasterArray,RasterLayer-method

Methods to mask RasterArray objects, or to mask with them

Description

Additional functions to mask generic function involving the RasterArray class. The following methods are implemented:

Usage

```
## S4 method for signature 'RasterArray,RasterLayer'
mask(
  x,
  mask,
  inverse = FALSE,
  maskvalue = NA,
  updatevalue = NA,
  updateNA = FALSE,
  ...
)

## S4 method for signature 'RasterArray,RasterArray'
mask(
  x,
  mask,
  inverse = FALSE,
  maskvalue = NA,
  updatevalue = NA,
  updateNA = FALSE,
  ...
)

## S4 method for signature 'RasterArray,Spatial'
mask(x, mask, inverse = FALSE, updatevalue = NA, updateNA = FALSE, ...)

## S4 method for signature 'RasterArray,RasterStackBrick'
mask(
  x,
  mask,
  inverse = FALSE,
  maskvalue = NA,
  updatevalue = NA,
  updateNA = FALSE,
  ...
)

## S4 method for signature 'RasterLayer,RasterArray'
mask(
  x,
  mask,
  filename = "",
  inverse = FALSE,
  maskvalue = NA,
  updatevalue = NA,
  updateNA = FALSE,
  ...
)
```

```
## S4 method for signature 'RasterStackBrick,RasterArray'
mask(
  x,
  mask,
  filename = "",
  inverse = FALSE,
  maskvalue = NA,
  updatevalue = NA,
  updateNA = FALSE,
  ...
)
```

Arguments

x	Raster* object
mask	Raster* object or a Spatial* object
inverse	logical. If TRUE, areas on mask that are _not_ the maskvalue are masked
maskvalue	numeric. The value in mask that indicates the cells of x that should become updatevalue (default = NA)
updatevalue	numeric. The value that cells of x should become if they are not covered by mask (and not NA)
updateNA	logical. If TRUE, NA values outside the masked area are also updated to the updatevalue (only relevant if the updatevalue is not NA.
...	additional arguments as in writeRaster.
filename	character. Optional output filename (only if x is a RasterLayer and RasterStack-Brick)

Details

RasterArray masked with RasterLayer: every RasterLayer in the stack masked.

RasterArray masked with another RasterArray: one-to-one match between RasterLayers.

RasterArray masked with RasterStack: one-to-one match between RasterLayers.

RasterArray masked with Spatial: all layers masked with an Sp object

RasterArray masked with Spatial: all layers masked with an Sp object

RasterLayer masked with RasterArray: layer is masked out iteratively with every member of Raster-Array.

Value

A RasterArray or RasterLayer class object (see detaisl above).

Examples

```
data(dems)

# land
lands <- dems
for(i in 1:length(lands)){
  values(lands[i])[values(lands[i])<0] <- NA
  values(lands[i])[!is.na(values(lands[i]))] <- 1
}

# land topographies
landTopo<- mask(dems, lands)
```

`matchtime` *Match the dates of a time-dependent variable with a predefined vector*

Description

The function takes a variable `x` (e.g. a vector or a `RasterArray` object), and reorders it to best match the dates provided in a vector `y`.

Usage

```
matchtime(x, y, ...)

## S4 method for signature 'numeric'
matchtime(x, y, index = FALSE, ...)

## S4 method for signature 'character'
matchtime(x, y, index = FALSE, ...)

## S4 method for signature 'RasterArray'
matchtime(x, y, index = FALSE, time = 1, ...)
```

Arguments

`x`	Object to be reordered to match `y`.
`y`	(`numeric`) The vector of dates (numeric values) to order to.
`...`	Additional arguments passed to class-specific methods.
`index`	(`logical`) If this argument is `TRUE`, only the indices will be returned that refer to the new order, rather than the reordered `x` variable.
`time`	`numeric`. Single value referring to that dimension of `x` where the time-coding names are (`time=1` is the default for RasterArrays in `chronosphere`).

Value

An object of the class as x or a numeric vector.

Examples

```
# original vector
orig <- 1:10
# target values
targ <- c(5.1,4.2, 3.4, 2.7, 2.3)
# how do the two series match the best?
matchtime(orig, targ)
```

---

minValue,RasterArray-method

Minimum and maximum values in a RasterArray object

Description

The method is inherited from the RasterStack class. Postitions of layers are conserved in the output. (including missing layers)

Usage

```
## S4 method for signature 'RasterArray'
minValue(x, vec = FALSE)

## S4 method for signature 'RasterArray'
maxValue(x, vec = FALSE)
```

Arguments

x	a RasterArray class object.
vec	Should the dimensions of the RasterArray be omitted?

Value

A numeric vector.

Examples

```
data(dems)
rangeVals <- cbind(
  minValue(dems),
  maxValue(dems)
)
```

names,XArray-method *Names of one-dimensional* RasterArray, SpatialStack *or* SpatialArray *objects.*

Description

Get or set the names of one-dimensional RasterArray or SpatialArray objects

Usage

```
## S4 method for signature 'XArray'
names(x)

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

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

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

Arguments

x	RasterArray, SpatialStack or SpatialArray object.
value	character vector.

Value

A character vector of names or NULL.

Examples

```
data(dems)
names(dems)
names(dems)[4] <- "weirdo"
# NULL
data(coasts)
names(coasts)
```

ncell,RasterArray-method

Number of cells in a RasterArray object

Description

The method is inherited from the RasterStack class.

Usage

```
## S4 method for signature 'RasterArray'
ncell(x)
```

Arguments

x	a RasterArray class object.

Value

A numeric value.

Examples

```
data(dems)
ncell(dems)
```

ncol,XArray-method

Number of columns and rows of a RasterArray *or* SpatialArray

Description

Unlike the ncol and nrow functions of the raster package (ncell), this function returns the number of columns and rows of the RasterArray container, rather than the dimensions of the contained RasterLayers.

Usage

```
## S4 method for signature 'XArray'
ncol(x)

## S4 method for signature 'XArray'
nrow(x)
```

Arguments

x	A RasterArray or SpatialArray class object.

Value

A numeric value of the number of columns and rows.

Examples

```
data(coasts)
ncol(coasts)
nrow(coasts)
```

---

newbounds *Redefine bounds of a named matrix*

Description

The function restructures a matrix and extends its current limits to a range defined by a names attribute

Usage

```
newbounds(x, cols = NULL, rows = NULL)
```

Arguments

x	The matrix to be restructured.
cols	Column names guiding the restructuring.
rows	Row names guiding the restructuring.

Details

This is essentially a subsetting function that allows you to subset even when the rownames or colnames vector extends beyond the bounds of a matrix and traditional subsetting methods result in the notorious 'out of bounds' error.

Value

A matrix with extended bounds.

Examples

```
a<-matrix(1:9, ncol=3)
rownames(a) <- c("a", "c", "d")
newbounds(a, rows=letters[1:5])
```

nums *Names as numerics*

Description

The set of functions return names of objects directly transformed to numeric values.

Usage

```
nums(x)

colnums(x)

rownums(x)
```

Arguments

x	Object with names, colnames or rownames attributes.

Value

Numeric vector.

Examples

```
data(dems)
# ages as numerics
nums(dems)
# younger than 20Ma
dems[nums(dems)<20]
```

nvalues *The total number of values in a RasterArray object*

Description

The total number of values in a RasterArray object

Usage

```
nvalues(x, ...)

## S4 method for signature 'RasterArray'
nvalues(x)
```

Arguments

x	A RasterArray class object.
...	additional arguments passed to class-specific methods.

Value

A numeric value.

Examples

```
data(dems)
nvalues(dems)
```

---

platemodel-class *Class of objects representing plate tectonic models*

Description

Meta-object containing paths to a unique plate tectonic model

Usage

```
## S4 method for signature 'platemodel'
initialize(.Object, path = NULL, rotation = NULL, polygons = NULL)
```

Arguments

.Object	Constructor argument (not needed).
path	(character) Path to a .mod unique plate model object.
rotation	(character) If path is NULL, the path to the rotation file-part of the model.
polygons	(character) If path is NULL, the path to the plate polygon file-part of the model.

Value

A platemodel class object.

Examples

```
# extract provided archive
a <- fetch(dat="paleomap", var="model",
  datadir=system.file("extdata", package="chronosphere"))
# manually attach
model <- platemodel(file.path(tempdir(),
  "paleomap_model_v19o_r1c/paleomap_model_v19o_r1c.mod"))
model
```

---

plot,RasterArray,missing-method
Shorthand for the plotting RasterArray *and* SpatialArray *objects*

Description

This plot, method executes the mapplot function on the RasterArray or SpatialArrayobject.

Usage

```
## S4 method for signature 'RasterArray,missing'
plot(x, y, ...)

## S4 method for signature 'SpatialArray,missing'
plot(x, y, ...)
```

Arguments

x	A (RasterArray or SpatialArray) Object to be plotted.
y	Not implemented yet.
...	Arguments passed to the mapplot function.

Value

None.

Examples

```
data(dems)
plot(dems)
```

---

projectRaster *Project a RasterArray object*

Description

The method implemets the projectRaster function for RasterArray class objects.

Usage

```
projectRaster(
  from,
  to,
  res,
  crs,
  method = "bilinear",
  alignOnly = FALSE,
  over = FALSE,
  filename = "",
  ...
)

## S4 method for signature 'RasterArray'
projectRaster(
  from,
  to,
  res,
  crs,
  method = "bilinear",
  alignOnly = FALSE,
  over = FALSE
)
```

Arguments

from	A Raster* RasterArray object to project.
to	Raster* object with the parameters to which 'from' should be projected
res	single or (vector of) two numerics. To, optionally, set the output resolution if 'to' is missing
crs	character or object of class 'CRS'. PROJ.4 description of the coordinate reference system. In projectRaster this is used to set the output CRS if 'to' is missing, or if 'to' has no valid CRS
method	method used to compute values for the new RasterLayer. Either 'ngb' (nearest neighbor), which is useful for categorical variables, or 'bilinear' (bilinear interpolation; the default value), which is appropriate for continuous variables.
alignOnly	logical. Use to or other parameters only to align the output (i.e. same origin and resolution), but use the projected extent from from
over	logical. If TRUE wrapping around the date-line is turned off. This can be desirable for global data (to avoid mapping the same areas twice) but it is not desireable in other cases
filename	character output filname. Not applicable for RasterArray class objects.
...	additional arguments as for writeRaster.

Value

A projected RasterArray class object.

Examples

```
# project first three to mollweide
data(dems)
suppressWarnings(
  mollDem <- projectRaster(dems[1:3], crs=CRS("+proj=moll"))
)
```

---

proxy *The proxy of a RasterArray or SpatialArray object*

Description

This function returns an object that symbolizes the structure of layers in the RasterArray and SpatialArray.

Usage

```
proxy(x, ...)

## S4 method for signature 'XArray'
proxy(x)
```

Arguments

x	RasterArray or SpatialArray ocal object.
...	additional arguments passed to class-specific methods.

Details

The proxy method wraps the names of layers in the stack using the index slot of the RasterArray.

Value

A vector, matrix or array of characters representing the RasterArray and SpatialArray structure.

Examples

```
data(dems)
proxy(dems)

data(coasts)
proxy(coasts)
```

ramps *Colour gradient ramps*

Description

The object contains functions produced by the colorRampPalette function.

Usage

```
gradinv(n)

ocean(n)

terra(n)

ipccTemp(n, force = 11)

ipccPrec(n, force = 11)

wet(n)

ipccLine(n = 6)

ipccRCP(n = 4)

showPal(pal = "all")
```

Arguments

n	(numeric) Number of different colors to generate from the palette
force	(logical) Specify pal when multiple are available. More details to come.
pal	(character) A palette name from the lists below

Details

showPal can be used to display the available palettes. You can use pal = "all" or pal="" if you want to look at all the available palettes. You can also view single palettes individually. The following colour palettes are implemented:

- gradinv(): inverse heatmap.
- ocean(): ocean bathymetrical colours.
- terra(): terrestrial topographical colours.
- ipccTemp(): gradient from blue to red according to the official IPCC AR6 WG2 colour palette.
- ipccPrec(): gradient from brown to green according to the official IPCC AR6 WG2 colour palette.

- `wet()`: gradient from white to green to blue.
- `ipccLine()`: discrete colours for line graphs according to the official IPCC AR6 WG2 colour palette.
- `ipccRCP()`: discrete colours for climate scenarios according to the official IPCC AR6 WG2 colour palette.

Value

A function producing a colour gradient ramp.

`RasterArray-class` *Virtual Array of RasterLayers*

Description

Array template for RasterLayers

Arguments

`stack`	A `RasterStack` class object.
`index`	A `vector`, `matrix` or `array` type object. Includes either the indices of layers in the stack, or their names.
`dim`	A `numeric` vector. Same as for `array`, creates `proxy` procedurally.

Details

The class implements structures to organize RasterLayers that have the same dimensions. Subsetting rules were defined using the proxy object in the `index` slot. See examples for implementations.

The class has two slots: stack: RasterStack, the actual data. index: A proxy object that represents the organization of the layers.

Value

A `RasterArray` class object.

Examples

```
# data import
  data(dems)
  st <-dems@stack
  ind <- 1:nlayers(st)
  names(ind) <- letters[1:length(ind)]
  ra<- RasterArray(stack=st, index=ind)
```

reconstruct *Reconstruct geographic features*

Description

Reconstruct the geographic locations from present day coordinates and spatial objects back to their paleo-positions. Each location will be assigned a plate id and moved back in time using the chosen reconstruction model.

Usage

```
reconstruct(x, ...)

## S4 method for signature 'matrix'
reconstruct(
  x,
  age,
  model = "PALEOMAP",
  listout = TRUE,
  verbose = FALSE,
  enumerate = TRUE,
  chunk = 200,
  reverse = FALSE,
  path.gplates = NULL,
  cleanup = TRUE,
  dir = NULL,
  plateperiod = FALSE
)

## S4 method for signature 'data.frame'
reconstruct(x, ...)

## S4 method for signature 'numeric'
reconstruct(x, ...)

## S4 method for signature 'character'
reconstruct(
  x,
  age,
  model = "PALEOMAP",
  listout = TRUE,
  verbose = FALSE,
  path.gplates = NULL,
  cleanup = TRUE,
  dir = NULL,
  plateperiod = FALSE
)
```

```
## S4 method for signature 'SpatialPolygonsDataFrame'
reconstruct(
  x,
  age,
  model = "PALEOMAP",
  listout = TRUE,
  verbose = FALSE,
  path.gplates = NULL,
  cleanup = TRUE,
  dir = NULL,
  plateperiod = FALSE
)

## S4 method for signature 'SpatialLinesDataFrame'
reconstruct(
  x,
  age,
  model = "PALEOMAP",
  listout = TRUE,
  verbose = FALSE,
  path.gplates = NULL,
  cleanup = TRUE,
  dir = NULL,
  plateperiod = FALSE
)
```

Arguments

x	are the features to be reconstructed. Can be a vector with longitude and latitude representing a single point or a matrix/dataframe with the first column as longitude and second column as latitude, or a SpatialPolygonsDataFrame class object. The character strings "plates" and "coastlines" return static plates and rotated present-day coastlines, respectively.
...	arguments passed to class-specific methods.
age	(numeric)is the age in Ma at which the points will be reconstructed
model	(character or platemodel) The reconstruction model. The class of this argument selects the submodule used for reconstruction, a character value will invoke the remote reconstruction submodule and will submit x to the GPlates Web Service. A platemodel class object will call the local-reconstruction submodule. The default is "PALEOMAP". See details for available models.
listout	(logical)If multiple ages are given, the output can be returned as a list if listout = TRUE.
verbose	(logical) Should call URLs (remote submodule) or console feedback (local-submodule) be printed?
enumerate	(logical) Should be all coordinate/age combinations be enumerated and reconstructed (set to TRUE by default)? FALSE is applicable only if the number of rows

in x is equal to the number elementes in age. Then a point will be reconstructed to the age that has the same index in age as the row of the coordinates in x. List output is not available in this case.

chunk (numeric) Argument of the remote reconstruction submodule. Single integer, the number of coordinates that will be queried from the GPlates in a single go.

reverse (logical) Argument of the remote reconstruction submodule. The flag to control the direction of reconstruction. If reverse = TRUE, the function will calculate the present-day coordinates of the given paleo-coordinates.

path.gplates (character) Argument of the local reconstruction submodule. In case the GPlates executable file is not found at the coded default location, the full path to the executable (gplates-<ver>.exe on Windows) can be entered here.

cleanup (logical) Argument of the local reconstruction submodule. Should the temporary files be deleted immediately after reconstructions?

dir (character) Argument of the local reconstruction submodule. Directory where the temporary files of the reconstruction are stored (defaults to a temporary directory created by R). Remember to toggle cleanup if you want to see the files.

plateperiod (logical) Argument of the local reconstruction submodule. Should the durations of the plates be forced on the partitioned feature? If these are set to TRUE and the plate duration estimates are long, then you might lose some data.

Details

The function implements two reconstruction submodules, which are selected with the model argument:

If model is a character entry, then the reconstruct() function uses the GPlates Web Service (https://gws.gplates.org/, remote reconstruction submodule). The available reconstruction models for this submodule are:

- "SETON2012" (Seton et al., 2012) for coastlines and plate polygons.
- "MULLER2016" (Muller et al., 2016) for coastlines and plate polygons.
- "GOLONKA" (Wright et al. 2013) for coastlines only.
- "PALEOMAP" (Scotese and Wright, 2018) for coastlines and plate polygons.
- "MATTHEWS2016" (Matthews et al., 2016) for coastlines and plate polygons.

If model is a platemodel class object, then the function will try to use the GPLates desktop application (https://www.gplates.org/) to reconstruct the coordinates (local reconstruction submodule). Plate models are available in chronosphere with the fetch function. See datasets for the available models. The function will try to find the main GPlates executable in its default installation directory. If this does not succeed, use path.gplates to enter the full path to the GPlates executable as a character string.

Value

A numeric matrix if x is a numeric, matrix or data.frame, or Spatial* class objects, depending on input.

References

Matthews, K. J., Maloney, K. T., Zahirovic, S., Williams, S. E., Seton, M., & Müller, R. D. (2016). Global plate boundary evolution and kinematics since the late Paleozoic. Global and Planetary Change, 146, 226–250. https://doi.org/10.1016/j.gloplacha.2016.10.002

Müller, R. D., Seton, M., Zahirovic, S., Williams, S. E., Matthews, K. J., Wright, N. M., ... Cannon, J. (2016). Ocean Basin Evolution and Global-Scale Plate Reorganization Events Since Pangea Breakup. Annual Review of Earth and Planetary Sciences, 44(1), 107–138. https://doi.org/10.1146/annurev-earth-060115-012211

Scotese, C., & Wright, N. M. (2018). PALEOMAP Paleodigital Elevation Models (PaleoDEMS) for the Phanerozoic PALEOMAP Project. Retrieved from https://www.earthbyte.org/paleodem-resource-scotese-and-wright-2018/

Seton, M., Müller, R. D., Zahirovic, S., Gaina, C., Torsvik, T., Shephard, G., ... Chandler, M. (2012). Global continental and ocean basin reconstructions since 200Ma. Earth-Science Reviews, 113(3–4), 212–270. https://doi.org/10.1016/j.earscirev.2012.03.002

Wright, N., Zahirovic, S., Müller, R. D., & Seton, M. (2013). Towards community-driven paleogeographic reconstructions: integrating open-access paleogeographic and paleobiology data with plate tectonics. Biogeosciences, 10(3), 1529–1541. https://doi.org/10.5194/bg-10-1529-2013

Examples

```
# With the web service (GPlates Web Service was offline at submission)
# simple matrices
# reconstruct(matrix(c(95, 54), nrow=1), 140)

# points reconstruction
xy <-cbind(long=c(95,142), lat=c(54, -33))
# reconstruct(xy, 140)

# coastlines/plates
# coast <- reconstruct("coastlines", 140)
# plate <- reconstruct("plates", 139)
```

---

reference *Retrieve citation of data object*

Description

The function prints or returns the citation string of a chosen object/item.

Usage

```
reference(dat, var = NULL, ver = NULL, print = TRUE, prefix = "")
```

Arguments

dat	(characater) Object downloaded with fetch or the database identifier string.
var	(character) In case dat is character, the variable identifier.
ver	(character) In case dat is character, the version identifier.
print	(logical) Should the citations be printed to the console, or returned as a character vector.
prefix	(characater) In case the output is printed on the console. Use this to include a prefix before every entry.

Details

The function is intended to be updated to handle BibTEX entries.

---

resample,RasterArray,ANY-method

Resample a RasterArray object

Description

The method is inherited from the RasterStack class.

Usage

```
## S4 method for signature 'RasterArray,ANY'
resample(x, y, ...)
```

Arguments

x	a RasterArray class object.
y	The y argument of the resample function.
...	arguments passed to the resample function.

Value

A resampled RasterArray class object.

Examples

```
data(dems)
template <- raster(res=5)
resampled <- resample(dems, template)
```

`rotate` *Rotate a* `RasterArray` *object*

Description

The method is inherited from the `RasterStack` class.

Usage

```
## S4 method for signature 'RasterArray'
rotate(x, ...)
```

Arguments

`x`	(`RasterArray`) Object.
`...`	Additional arguments passed to the `rotate` function.

Value

A `RasterArray`-class object.

`rownames,XArray-method`

Row names of two-dimensional `RasterArray` *or* `SpatialArray` *objects*

Description

Get or set the row names of two-dimensional `RasterArray` or `SpatialArray`objects

Usage

```
## S4 method for signature 'XArray'
rownames(x)

## S4 replacement method for signature 'XArray'
rownames(x) <- value
```

Arguments

`x`	`RasterArray` or `SpatialArray` object.
`value`	`character` vector.

Value

A `character` vector of row names or `NULL`.

Examples

```
data(coasts)
rownames(coasts)
rownames(coasts) <- paste(rownames(coasts), "Ma")
```

---

shaper *Code snippets defining ranges based on points located on a plot*

Description

The function returns snippets of code that you can paste in your script after you select points on a plot. Useful for defining areas on a map. The default methods assume that you will first click in the bottom left and then in the bottom right corner.

Usage

```
shaper(f = "p", n = 2, round = 2, ...)
```

Arguments

f	(character) A single letter value specifying for which function's arugment format you want to get parameters. "p" is for plot, "r" is for rect, "s" is for segments. "e" returns a call to create an extent class object from the package raster. "m" will return code to define a 2 column matrix.
n	(integer) The number of points to request.
round	(integer) Number of digits to round to, can be two values, first is for x second for y.
...	arguments passed to the locator function

Value

For certain methods ("m" and "e") the function returns a matrix or extent class object if the function output is assigned to a name.

Examples

```
# plot something
data(dems)
mapplot(dems[1], col="earth")
# click 5 times to get the long-lat coords of 5 points
# shaper("m",5)
# example output:
mat <- matrix(c(
  -2.89, 31.55,
  3.32, 26.99,
  21.17, 17.87,
```

```
  33.6, 11.03,
  5.65, 19.39
), ncol=2, byrow=TRUE)
#plot them
points(mat)
```

SpatialArray-class *Virtual Array of Vector Spatial data*

Description

Array template for Spatial object

Arguments

stack	A SpatialStack class object.
index	A vector, matrix or array type object. Includes either the indices of layers in the stack, or their names.
dim	A numeric vector. Same as for array, creates proxy procedurally.

Details

The class implements structures to organize Spatial objects that have the same CRS. Subsetting rules were defined using the proxy object in the index slot. See examples for implementations.

The class has two slots: stack: SpatialStack, the actual data. index: A proxy object that represents the organization of the layers.

Value

A SpatialStack class object.

Examples

```
# data import
  data(dems)
  st <-dems@stack
  ind <- 1:nlayers(st)
  names(ind) <- letters[1:length(ind)]
  ra<- RasterArray(stack=st, index=ind)
```

SpatialStack-class *Stack of Spatial Objects*

Description

Vector data in the same CRS organized into a vector.

Arguments

Spatials	A list of Spatial objects or a character vector of file names identifying items for readOGR, in case the rgdal package is installed..
proj4string	A CRS-class object.
verbose	A logical value. Same as for array, creates proxy procedurally.

Details

The class implements a stack of vector data that mimic RasterStack-class objects, only with vector data. Classes, such as link[sp]{SpatialPoints}, link[sp]{SpatialPointsDataFrame}, link[sp]{SpatialLines}, link[sp]{SpatialLinesDataFrame}, link[sp]{SpatialPolygons} and link[sp]{SpatialPolygonsDataFrame} can be concatenated to a vector/list, where elements can be accessed using list-type subsetting. The only restriction is that the items must share the same CRS.

The class has two slots: Spatials: List of Spatial items. CRS: The coordinate reference system (CRS). bbox: The bounding box of all items.

Value

A SpatialStack class object.

spTransform,SpatialStack,ANY-method *Function to transform the coordinate reference system of an entire SpatialStack or SpatialArray*

Description

Joint reprojection of entire sets of vector data.

Usage

```
## S4 method for signature 'SpatialStack,ANY'
spTransform(x, CRSobj, ...)

## S4 method for signature 'SpatialArray,ANY'
spTransform(x, CRSobj, ...)
```

Arguments

x	The SpatialStack or SpatialArray object.
CRSobj	A CRS class or character object defining a coordinate reference system.
...	Additional arguments.

Details

The function requires the rgdal package to run.

Value

A SpatialStack or SpatialArray object.

Examples

```
# load example data
data(coasts)
mollCoast <- spTransform(coasts, "+proj=moll")
```

---

stack,VectorSpatialClasses-method

Stacking method for the vector Spatial SpatialStack objects*

Description

The function alls a RasterArray-like stacking of Spatial* objects and SpatialStacks .

Usage

```
## S4 method for signature 'VectorSpatialClasses'
stack(x, ...)

## S4 method for signature 'SpatialStack'
stack(x, ...)
```

Arguments

x	SpatialPoints,SpatialPointsDataFrame,SpatialLines,SpatialLinesDataFrame,SpatialPolygor object.
...	Additional Spatial* objects.

Value

A RasterArray class object.

A RasterArray class object.

Examples

```
data(coasts)
one <- coasts[1]
two <- coasts[2]
three <- coasts[3]
# create a SpatialStack similar to a RasterStack
spStack <- stack(one, two, three)
```

`subset,SpatialStack-method`
Subset a SpatialStack object

Description

Extract subsets of SpatialStack class object similarly to a RasterStack

Usage

```
## S4 method for signature 'SpatialStack'
subset(x, i, drop = TRUE)
```

Arguments

x	SpatialStack object.
i	subscript ofvector-like subsetting.
drop	logical in case the result of subsetting is a single element, should the SpatialStack wrapper be dropped and the element be reduced to a single RasterLayer?

Value

A Spatial or SpatialStack class object.

Examples

```
# stack of the paleomap paleocoastlines
data(coasts)
spstack <- coasts@stack
subset(spstack, "X5Ma_CS_v7")
```

subset,XArray-method *Subset a* RasterArray *or* SpatialArray *object*

Description

Extract subsets of RasterArray or SpatialArray class object similarly to a regular array.

Usage

```
## S4 method for signature 'XArray'
subset(x, i, j, ..., oneDim = FALSE, drop = TRUE)
```

Arguments

x	RasterArray or SpatialArray object.
i	subscript of the first dimension(rows) or vector-like subsetting.
j	subscript of the second dimension (columns).
...	subscript of additional dimensions.
oneDim	logical In case of multidimensional RasterArrays or SpatialArrays, setting oneDim to TRUE allows the application of one dimensional subscripts.
drop	logical in case the result of subsetting is a single element, should the \codeRasterArray or SpatialArray wrapper be dropped and the element be reduced to a single RasterLayer/ Spatial* object?

Value

A RasterLayer, RasterArray, Spatial* or SpatialArray class object.

Examples

```
data(dems)
# first 4
subset(dems, i=1:4)
# missing at the end
subset(dems, i=1:12)
# character subscript
subset(dems, i=c("5", "25"))
# logical subscript
subs <- rep(TRUE, length(dems))
subs[1] <- FALSE # remove first
subset(dems, i= subs)
# no drop
subset(dems, i=1, drop=FALSE)
data(coasts)
subset(coasts, i=2, j=1:2)
```

t *Transpose a* RasterArray *or* SpatialArray *object*

Description

Transpose a RasterArray or SpatialArray object

Usage

```
t

## S4 method for signature 'XArray'
t(x)
```

Arguments

x	A RasterArray or SpatialArray class object.

Format

An object of class standardGeneric of length 1.

Value

A RasterArray or SpatialArray class object.

Examples

```
data(dems)
t(dems)
data(coasts)
t(coasts)
```

types *Return types of objects in a* SpatialStack *or* SpatialArray *object*

Description

Methods sequences that start with an NA do not yet work.

Usage

```
types(x)

## S4 method for signature 'SpatialStack'
types(x)

## S4 method for signature 'SpatialArray'
types(x)
```

Arguments

x	SpatialStack or SpatialArray object.

Value

A character class object.

Examples

```
data(coasts)
types(coasts)
```

---

xres,RasterArray-method

Resolution of a RasterArray object

Description

The methods are inherited from the RasterStack class, see resolution. Replacement is not allowed.

Usage

```
## S4 method for signature 'RasterArray'
xres(x)

## S4 method for signature 'RasterArray'
yres(x)

## S4 method for signature 'RasterArray'
res(x)
```

Arguments

x	a RasterArray class object.

Value

A numeric vector.

Examples

```
data(dems)
res(dems)
yres(dems)
xres(dems)
```

[,SpatialStack,ANY,ANY-method

Indexing to extract subsets of a SpatialStack object

Description

The single and double bracket subsetting is identical in the case of SpatiaStacks.

Usage

```
## S4 method for signature 'SpatialStack,ANY,ANY'
x[i, drop = TRUE]
```

Arguments

x	SpatialStack object.
i	subscript of vector-like subsetting.
drop	logical in case the result of subsetting is a single element, should the SpatialStack wrapper be dropped and the element be reduced to a single Spatial?

Value

A SpatialStack or Spatial class object.

Examples

```
# stack of the paleomap paleocoastlines
data(coasts)
spstack <- coasts@stack
spstack[1]
```

[,XArray,ANY,ANY-method

*Indexing to extract subsets of a code*RasterArray *or* SpatialArray *object*

Description

Single bracket '[' refers to indices and names within the RasterArray . Use double brackets to extract layers based on their names (in the stack).

Usage

```
## S4 method for signature 'XArray,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

x	RasterArray or SpatialArray object.
i	subscript of the first dimension(rows) or vector-like subsetting.
j	subscript of the second dimension (columns).
...	subscript of additional dimensions.
drop	logical in case the result of subsetting is a single element, should the RasterArray or SpatialArray wrapper be dropped and the element be reduced to a single RasterLayer or Spatial*?

Value

A RasterLayer, RasterArray, Spatial* or SpatialArray class object.

Examples

```
data(dems)
# numeric subsetting
firstThree <- dems[1:3]
# character subsetting
second <- dems["10"]
# logical subsetting
subscript <- rep(FALSE, length(dems))
subscript[2] <- TRUE
second2 <- dems[subscript]
data(coasts)
present<- coasts["0", ]
allMargin <- coasts[, "margin"]
```

[<-,SpatialStack,character,ANY,VectorSpatialClasses-method

Replace layers of a SpatialStack object

Description

The single and double bracket subsetting is identical in the case of SpatialStacks.

Usage

```
## S4 replacement method for signature 'SpatialStack,character,ANY,VectorSpatialClasses'
x[i, j = NULL, ...] <- value

## S4 replacement method for signature 'SpatialStack,logical,ANY,VectorSpatialClasses'
x[i, j = NULL, ...] <- value

## S4 replacement method for signature 'SpatialStack,numeric,ANY,VectorSpatialClasses'
x[i, j = NULL, ...] <- value
```

```
## S4 replacement method for signature 'SpatialStack,character,ANY,SpatialStack'
x[i, j = NULL, ...] <- value

## S4 replacement method for signature 'SpatialStack,logical,ANY,SpatialStack'
x[i, j = NULL, ...] <- value

## S4 replacement method for signature 'SpatialStack,numeric,ANY,SpatialStack'
x[i, j = NULL, ...] <- value

## S4 replacement method for signature 'SpatialStack,character,ANY,ANY'
x[[i, j, ...]] <- value

## S4 replacement method for signature 'SpatialStack,logical,ANY,ANY'
x[[i, j, ...]] <- value

## S4 replacement method for signature 'SpatialStack,numeric,ANY,VectorSpatialClasses'
x[[i, j = NULL, ...]] <- value
```

Arguments

x	SpatialStack object.
i	subscript of vector-like subsetting.
j	unused.
...	unused..
value	A single Spatial object.

Value

None.

Examples

```
data(coasts)
spstack <- coasts@stack[1:2]
spstack[1] <- mapedge()
```

[<-,XArray,ANY,ANY,logical-method

Replace layers of a RasterArray *or* SpatialArray *object*

Description

Single bracket '[' refers to indices and names within the \codeRasterArray or SpatialArray. Use double brackets to replace layers based on their names (in the stack). RasterLayers and RasterArray entries can be used to replace values in RasterArrays. Spatial* objects and SpatialArrays can be used with SpatialArrays.

Usage

```
## S4 replacement method for signature 'XArray,ANY,ANY,logical'
x[i, j, ...] <- value

## S4 replacement method for signature 'RasterArray,ANY,ANY,RasterLayer'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialPoints'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialPointsDataFrame'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialLines'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialLinesDataFrame'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialPolygons'
x[i, j, ...] <- value

## S4 replacement method for signature 'SpatialArray,ANY,ANY,SpatialPolygonsDataFrame'
x[i, j, ...] <- value
```

Arguments

x	RasterArray or SpatialArray object.
i	subscript of the first dimension(rows) or vector-like subsetting.
j	subscript of the second dimension (columns).
...	subscript of additional dimensions.
value	A same class object as x.

Value

None.

Examples

```
data(dems)
# replace third element with missing value
dems[3] <- NA
# duplicate first element and make it the second too
dems[2] <-dems[1]
```

[[,SpatialStack,ANY,ANY-method

Indexing to extract Spatial items from a SpatialStack object

Description

The single and double bracket subsetting is identical in the case of SpatialStacks.#'

Usage

```
## S4 method for signature 'SpatialStack,ANY,ANY'
x[[i, drop = TRUE]]
```

Arguments

x	SpatialStack object.
i	subscript of vector-like subsetting.
drop	logical should the SpatialStack be dropped and the element be reduced to a single Spatial object?

Value

A Spatial or SpatialStack class object.

[[,XArray,ANY,ANY-method

Indexing to extract RasterLayer*s of a* RasterArray *or* Spatial* *of a* SpatialArray *object*

Description

Double bracket '[[' refers to layers' name in the RasterStack of the RasterArray or the SpatialStack of the SpatialArray. Use single brackets to extract elements based on their position in the RasterArray or SpatialArray

Usage

```
## S4 method for signature 'XArray,ANY,ANY'
x[[i, drop = TRUE]]
```

Arguments

x	RasterArray or SpatialArray object.
i	subscript of the first dimension(rows) or vector-like subsetting.
drop	logical should the RasterStack be dropped and the element be reduced to a single RasterLayer?

Value

A RasterLayer or RasterArray class object.

Examples

```
data(dems)
# finds a layer
dems[["dem_30"]]
# returns a stack
dems[[c("dem_0", "dem_15")]]
# replaces a layervalues, but not the attributes of the layer
dem2 <- dems
dem2[["dem_0"]] <- dem2[["dem_5"]]
# compare every value in the 0 and 5 ma maps, they are all the same
mean(values(dem2[["dem_0"]]==dem2[["dem_5"]]))
```

[[<-,XArray,ANY,ANY,ANY-method

Replace RasterLayer*s in a* RasterArray *object and* Spatial* *objects in a* SpatialArray *object.*

Description

Double bracket '[[' refers to layers' name in the RasterStack of the RasterArray and the SpatialStack of the SpatialArray. Use single brackets to replace elements based on their position in the RasterArray/SpatialArray.

Usage

```
## S4 replacement method for signature 'XArray,ANY,ANY,ANY'
x[[i]] <- value

## S4 replacement method for signature 'SpatialArray,ANY,missing,SpatialStack'
x[[i]] <- value
```

Arguments

x	RasterArray or SpatialArray object.
i	subscript of layers to replace.
value	character vector.

Value

None.

Index