Package ‘angstroms’

October 12, 2022

Title Tools for 'ROMS' the Regional Ocean Modeling System

Version 0.0.1

Description Helper functions for working with Regional Ocean Modeling System 'ROMS' output. See <https://www.myroms.org/> for more information about 'ROMS'.

Depends R (>= 3.3.0), raster

License GPL-3

LazyData true

Imports nabor, ncdf4, proj4, sp, spbabel

RoxygenNote 6.0.1

Suggests testthat, covr

URL https://github.com/mdsumner/angstroms

BugReports https://github.com/mdsumner/angstroms/issues

NeedsCompilation no

Author Michael D. Sumner [aut, cre]

Maintainer Michael D. Sumner <mdsumner@gmail.com>

Repository CRAN

Date/Publication 2017-05-01 10:35:22 UTC

R topics documented:

angstroms *Tools for ROMS model output.*

Description

Facilities for easy access to Regional Ocean Modeling System (ROMS) output.

Details

coords_points	produce sp::SpatialPoints from ROMS coordinate arrays
croproms	create a raster::extent by cropping a ROMS data layer
rawdata	read raw NetCDF variables by name
romsdata read data layers from 4D variables by arbitrary slice	
romsboundary	generate the spatial boundary of the ROMS data set in x-y co
romscoords	extract the x-y (long-lat) coordinates
romshcoords	extract the multi-layer 'h'eight grid with S-coordinate stretch
romsmap	re-map a spatial layer (polygons, lines, points) into ROMS gr

antarctica *Antarctica simple coastline.*

Description

Taken from "rnaturalearth::countries110"

coords_points *Create SpatialPoints.*

Description

Convenience wrapper around `SpatialPoints` for a two layer brick with longitude and latitude coordinate arrays.

Usage

```
coords_points(x, ...)
```

Arguments

x two layer RasterBrick with longitude and latitude values

... ignored

Value

SpatialPoints

Examples

```
## library(raadtools)
##coords_points(romscoords(cpolarfiles()$fullname[1]))

pts <- coords_points(ice_coords)
```

croproms *Crop a ROMS layer*

Description

Crop a ROMS data layer from romsdata with a raster extent.

Usage

```
croproms(x, ext, ...)
```

Arguments

x ROMS xy- coordinates, see romscoords

ext raster::extent in the coordinate system of x

... ignored

Details

The spatial crop is performed in the coordinate space of roms data.

Examples

```
## notice that extent is in long-lat, but ice_local is in the grid
## space of ice_coords
ice_local <- croproms(ice_coords, extent(100, 120, -75, -60))
plot(ice_coords[[2]], col = grey(seq(0, 1, length  = 20)))
plot(crop(ice_fake, ice_local), add = TRUE)
```

ice_fake *Fake model data.*

Description

`ice_coords` and `ice_fake` are generated from a projected map of southern Ocean sea ice data.

Details

The coords layer is the longitude and latitude values for the centres of the polar cells. This is veyr loosely analogous to the coordinate arrays used by ROMS data, included here for working examples, illustration and code tests.

The proper metadata for these layers is "-3950000, 3950000, -3950000, 4350000 (xmin, xmax, ymin, ymax)"

"+proj=stere +lat_0=-90 +lat_ts=-70 +lon_0=0 +k=1 +x_0=0 +y_0=0 +a=6378273 +b=6356889.449 +units=m +no_defs"

rawdata *Read the variable as is*

Description

Read the variable as is

Usage

```
rawdata(x, varname)
```

Arguments

`x`	netcdf file path
`varname`	variable name

`romsboundary` *Boundary polygon from raster of coordinates.*

Description

Create a boundary polygon by tracking around coordinates stored in a RasterStack.

Usage

```
romsboundary(cds)
```

Arguments

`cds`	two-layer Raster

Details

The first layer in the stack is treated as the X coordinate, second as Y.

Examples

```
ice_grid_boundary <- romsboundary(ice_coords)
plot(antarctica)
## does not make sense in this space
plot(ice_grid_boundary, add = TRUE, border = "grey")

## ok in this one
#library(rgdal)
#   proj4string(ice_grid_boundary) <- CRS("+init=epsg:4326")
# pweird <- "+proj=laea +lon_0=147 +lat_0=-42 +ellps=WGS84"
#  laea_world <- spTransform(antarctica, pweird)
#  plot(extent(laea_world) + 8e6, type = "n", asp = 1)
#  plot(laea_world, add = TRUE)
#  plot(spTransform(ice_grid_boundary, pweird), add  = TRUE, border = "darkgrey")
```

`romscoords` *Extract coordinate arrays from ROMS.*

Description

Returns a RasterStack of the given variable names.

Usage

```
romscoords(x, spatial = c("lon_u", "lat_u"), ncdf = TRUE,
  transpose = FALSE, ...)
```

Arguments

`x`	ROMS file name
`spatial`	names of coordinate variables (e.g. lon_u, lat_u)
`ncdf`	default to NetCDF no matter what file name
`transpose`	the extents (ROMS is FALSE, Access is TRUE)
`...`	unused

Details

The two layers from the model output are used to define the real-world space. This is used to create a boundary `romsboundary`, to map real-world objects into grid space `romscoords` and to generate graticules for mapping into the grid space with `graphics::contour`.

Value

RasterStack with two layers of the 2D-variables

Examples

```
## Not run: 
  coord <- romscoord("roms.nc")

## End(Not run)
## with in-built fake data
plot(ice_fake, asp = 0.5)
contour(ice_coords[[1]], add = TRUE, levels = seq(-165, 165, by = 15))
contour(ice_coords[[2]], add = TRUE)
```

`romsdata` *ROMS single slice 2D layer Extract a data layer from ROMS by name and 4-D slice.*

Description

`romsdata` always works in the first two dimensions (x-y), the more specialist functions will work in the space indicated by their name `roms_xy`, `roms_xt` and so on.

Usage

```
roms_xy(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)

roms_xz(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)

roms_xt(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)

roms_yz(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)
```

```
roms_yt(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)

roms_zt(x, varname, slice = c(1L, 1L), transpose = TRUE, ...)

romsdata(x, varname, slice = c(1L, 1L), ncdf = TRUE, transpose = TRUE,
  ...)
```

Arguments

`x`	ROMS file name
`varname`	name of ROMS variable
`slice`	index in w and t (depth and time), defaults to first encountered
`transpose`	the extents (ROMS is FALSE, Access is TRUE)
`...`	unused
`ncdf`	default to `TRUE`, set to `FALSE` to allow raster format detection brick

Value

RasterLayer

Examples

```
#x <- raadtools:::cpolarfiles()$fullname[1]
#plot(roms_xy(x, "u"))
#plot(roms_xz(x, "u", slice = c(392L,1L)), asp = NA)
#plot(roms_xt(x, "u", slice = c(392L,1L)), asp = NA)

#plot(roms_yz(x, "u"))
#plot(roms_yt(x, "u", slice = c(1L,1L)), asp = NA)
#plot(roms_zt(x, "u", slice = c(1L, 392L)), asp = NA)
```

romshcoords *Coordinates at depth*

Description

Extract the multi-layer 'h'eight grid with S-coordinate stretching applied

Usage

```
romshcoords(x, S = "Cs_r", depth = "h")
```

Arguments

`x`	ROMS file name
`S`	of S-coordinate stretching curve at RHO-points
`depth`	depth thing

Details

S and h are the names of the appropriate variables

Value

RasterStack with a layer for every depth

romsmap *Remap an object to the ROMS grid.*

Description

Find the nearest-neighbour coordinates of x in the coordinate arrays of coords.

Usage

```
romsmap(x, ...)

## S3 method for class 'SpatialPolygonsDataFrame'
romsmap(x, coords, crop = FALSE,
  lonlat = TRUE, ...)

## S3 method for class 'SpatialLinesDataFrame'
romsmap(x, coords, crop = FALSE,
  lonlat = TRUE, ...)

## S3 method for class 'SpatialPointsDataFrame'
romsmap(x, coords, crop = FALSE,
  lonlat = TRUE, ...)
```

Arguments

x	object to transform to the grid space, e.g. a Spatial object
...	unused
coords	romscoords RasterStack
crop	logical, if TRUE crop x to the extent of the boundary of the values in coords
lonlat	logical, if TRUE check for need to back-transform to longitude/latitude and do it

Details

The input coords is a assumed to be a 2-layer RasterStack or RasterBrick and using nabor::knn the nearest matching position of the coordinates of x is found in the grid space of coords. The motivating use-case is the curvilinear longitude and latitude arrays of ROMS model output.

No account is made for the details of a ROMS cell, though this may be included in future. We tested only with the "lon_u" and "lat_u" arrays.

Value

input object with coordinates transformed to space of the coords

Note

Do not use this for extraction purposes without checking the output, this is best used for exploration and visualization. Re-mapping ROMS data is better done by looking up the `coords_points` within spatial objects, and transferring via the grid index.

Examples

```
ant_ice_coords <- romsmap(antarctica, ice_coords)
plot(ice_fake, main = "sea ice in pure grid space")
plot(ant_ice_coords, add = TRUE)
```

Index