Package 'grec'

February 19, 2020

Type Package

Title Gradient-Based Recognition of Spatial Patterns in Environmental Data

Version 1.4.1

Date 2020-02-10

URL https://github.com/LuisLauM/grec

BugReports https://github.com/LuisLauM/grec/issues

Maintainer Wencheng Lau-Medrano <luis.laum@gmail.com>

Description Provides algorithms for detection of spatial patterns from oceanographic data using image processing methods based on Gradient Recognition.

License GPL (>= 3)

Depends R (>= 3.2.0), imagine (>= 1.5.2), raster

Imports utils

LazyData true

RoxygenNote 7.0.2

Encoding UTF-8

NeedsCompilation no

Author Wencheng Lau-Medrano [aut, cre]

Repository CRAN

Date/Publication 2020-02-19 15:20:02 UTC

R topics documented:

grec-package — *GRadient-based RECognition of spatial patterns in environmental data*

Description

Provides algorithms for detection of spatial patterns from oceanographic data using image processing methods based on Gradient Recognition.

Author(s)

Wencheng Lau-Medrano, <luis.laum@gmail.com>

chl — *Sea Surface Chlorophyll Data*

Description

Surface chlorophyll maps downloaded from ERDDAP for running examples with `grec` functions.

Usage

```
chl
```

Format

A `list` with chlorophyll information from February to April of Aqua MODIS source.

References

ERDDAP website: https://coastwatch.pfeg.noaa.gov/erddap/index.html

colPalette — *Default color palette most using on environmental representations.*

Description

Vector with 2000 colors generated from `tim.colors` function.

Usage

```
colPalette
```

Format

A vector of 2000 colors in RGB format.

References

`tim.colors` from **fields** package

`detectFronts.RasterLayer`

Apply gradient-based methodologies to environmental data

Description

This function takes a environmental map (as a numeric `matrix`, `array`, `XYZlist` or `RasterLayer`) and allows the users to apply methodologies based on gradient-searching.

Usage

```
## S3 method for class 'RasterLayer'
detectFronts(x, method = "BelkinOReilly2009", intermediate = FALSE, ...)

## S3 method for class 'array'
detectFronts(x, method = "BelkinOReilly2009", intermediate = FALSE, ...)

## Default S3 method:
detectFronts(
  x,
  method = "BelkinOReilly2009",
  intermediate = FALSE,
  ConvolNormalization = TRUE,
  ...
)

detectFronts(
  x,
  method = "BelkinOReilly2009",
  intermediate = FALSE,
  ConvolNormalization = TRUE,
  ...
)

## S3 method for class 'list'
detectFronts(x, method = "BelkinOReilly2009", intermediate = FALSE, ...)

## S3 method for class 'matrix'
detectFronts(x, method = "BelkinOReilly2009", intermediate = FALSE, ...)
```

Arguments

x	Main input of class matrix, array, XYZ list or RasterLayer. See 'Details.'
method	character string indicating the method that will be used. See 'Details'.
intermediate	logical indicating whether to get the intermediate matrices (TRUE) or just the final one (FALSE).
...	Extra arguments that will depend on the selected method. See Details.
ConvolNormalization	logical indicating if convolutions will make a normalization (TRUE by default). See Details.

Details

Version 1.3.x performs two methods:

1. BelkinOReilly2009 (default): Based on Belkin & O'Reilly (2009) paper, it uses a Contextual Median Filter (CMF) for smoothing the original data.
2. median_filter: it uses a typical median filter (MF) for smoothing the original data. It also allows the user to change the window size for median filter (3 as default).

x could be given as a single numeric matrix from an environmental map. Othersiwe it also can be set as a three-dimension XYZ list: 'x' (a vector of longitudes), 'y' (vector of latitudes) and 'z' as a matrix of dimensions length(x$x)xxlength(x$y). You can also specify x as a RasterLayer or array object. If x is an array, it must have 3 dimensions: lon, lat and time. It is not required to specify the dimnames. The output will preserve all the attributes of input.

... allows the (advanced) users to modify some aspects of filter application. Depending on the selected methodology, some parameters can be modified:

times numeric. How many times do you want to apply the method?

kernelValues numeric. Vector with which are going to be used in convolution to identify Vertical and Horizontal gradients. By default, it will be the typical Sobel kernels.

radius numeric. If median filter method was selected, it allows to change the window size of the filter.

Normalization is a common practice in convolution in order to ensure that outputs are weighted within original range of values. It is achieved dividing outputs of convolution by sum(abs(kernel)). It is hardly recomended to use normalization in order to have always coherent values in regards of the original inputs; however, it can be deactivated by using ConvolNormalization argument.

Finally, Belkin & O'Reilly work proposed a log transformation after the gradient calculation. However, this step has not been considered as default in the function due to its application is focused on Chlorophyll values (maps).

Value

The output will preserve the input class (matrix, array, list or RasterLayer).

References

Belkin, I. M., & O'Reilly, J. E. (2009). An algorithm for oceanic front detection in chlorophyll and SST satellite imagery. Journal of Marine Systems, 78(3), 319-326 (http://dx.doi.org/10.1016/j.jmarsys.2008.11.018).

Examples

```
data(sst)
exampleSSTData <- list(x = sst$longitude,
                       y = sst$latitude,
                       z = sst$sst[,,1])

data(chl)
exampleChlData <- list(x = chl$longitude,
                       y = chl$latitude,
                       z = chl$chlorophyll[,,1])

# Simple application (over a XYZ list)
out_sst <- detectFronts(x = exampleSSTData)
out_chl <- detectFronts(x = exampleChlData)

# External transformation for chl data
out_chl$z <- log10(out_chl$z)

par(mfrow = c(2, 2), mar = rep(0, 4), oma = rep(0, 4))

image(exampleSSTData, col = colPalette, axes = FALSE)
mtext(text = "Original SST", side = 3, line = -2, adj = 0.99, cex = 1.2)

image(out_sst, col = colPalette, axes = FALSE)
mtext(text = "SST gradient", side = 3, line = -2, adj = 0.99, cex = 1.2)

image(exampleChlData, col = colPalette, axes = FALSE)
mtext(text = "Original Chlorophyll", side = 3, line = -2, adj = 0.99, cex = 1.2)

image(out_chl, col = colPalette, axes = FALSE)
mtext(text = "Chlorophyll gradient\n(log scale)", side = 3, line = -4, adj = 0.99,
      cex = 1.2)
```

---

sst *Sea Surface Temperature Data*

Description

SST maps downloaded from ERDDAP for running examples with `grec` functions.

Usage

```
sst
```

Format

A `list` with SST information from February to April of Aqua MODIS source.

References

ERDDAP website: https://coastwatch.pfeg.noaa.gov/erddap/index.html

Index