Package ‘HiClimR’

December 10, 2019

Version 2.1.5

Title Hierarchical Climate Regionalization

Author Hamada S. Badr [aut, cre], Benjamin F. Zaitchik [aut], Amin K. Dezfuli [aut]

Description
A tool for Hierarchical Climate Regionalization applicable to any correlation-based clustering. It adds several features and a new clustering method (called, 'regional' linkage) to hierarchical clustering in R ('hclust' function in 'stats' library): data regridding, coarsening spatial resolution, geographic masking, contiguity-constrained clustering, data filtering by mean and/or variance thresholds, data preprocessing (detrending, standardization, and PCA), faster correlation function with preliminary big data support, different clustering methods, hybrid hierarchical clustering, multivariate clustering (MVC), cluster validation, visualization of regionalization results, and exporting region map and mean timeseries into NetCDF-4 file.

License GPL (>= 2)

URL https://github.com/hsbadr/HiClimR

BugReports https://github.com/hsbadr/HiClimR/issues

Maintainer Hamada S. Badr <badr@jhu.edu>

Depends R (>= 2.10)

Imports graphics, grDevices, stats, utils, ncdf4

Suggests devtools, knitr, rmarkdown, roxygen2

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 6.1.1

LazyData yes

LazyLoad yes

ByteCompile yes

NeedsCompilation yes

SystemRequirements NetCDF (>= 4.1)

Repository CRAN

Date/Publication 2019-12-10 17:20:02 UTC

R topics documented:

coarseR *Coarsening spatial resolution for gridded data*

Description

coarseR is a helper function that helps coarsening spatial resolution of the input matrix for the HiClimR function.

Usage

```
coarseR(x=x, lon=lon, lat=lat, lonStep=1, latStep=1, verbose = TRUE)
```

Arguments

`x` an (N rows by M columns) matrix of 'double' values: N objects (spatial points or stations) to be clustered by M observations (temporal points or years). For gridded data, the N objects should be created from the original matrix x0 using as.vector(t(x0)), where x0 is an (n rows by m columns) matrix, n = length(unique(lon)) and m = length(unique(lat)).

`lon` a vector of longitudes with length N. For gridded data, the length may have the value (n) provided that n * m = N where n = length(unique(lon)) and m = length(unique(lat)).

`lat` a vector of latitudes with length N or m. See lon.

`lonStep` an integer greater than or equal to 1 for longitude step to coarsen gridded data in the longitudinal direction. If lonStep = 1, gridded data will not be coarsened in the longitudinal direction (the default). If lonStep = 2, every other grid in longitudinal direction will be retained.

`latStep` an integer greater than or equal to 1 for latitude step to coarsen gridded data in the latitudinal direction. If latStep = 1, gridded data will not be coarsened in the latitudinal direction (the default). If latStep = 2, every other grid in latitudinal direction will be retained. lonStep and latStep are independent so that user can optionally apply different coarsening level to each dimension.

`verbose` logical to print processing information if verbose = TRUE.

Details

For high-resolution data, the computational and memory requirements may not be met on old machines. This function enables the user to use coarser data in any spatial dimension:longitude, latitude, or both. It is available for testing or running `HiClimR` package on old computers or machines with small memory resources. The rows of output matrix (`x` component) will be also named by longitude and latitude coordinates. If `lonStep = 1` and `latStep = 1`, `coarseR` function will just rename rows of matrix `x`.

Value

A list with the following components:

`lon`	longitude mesh vector for the coarsened data.
`lat`	latitude mesh vector for the coarsened data.
`rownum`	original row numbers for the coarsened data.
`x`	coarsened data of the input data matrix `x`.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, `https://doi.org/10.1007/s12145-015-0221-7`.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, `https://cran.r-project.org/package=HiClimR`.

See Also

`HiClimR`, `HiClimR2nc`, `validClimR`, `geogMask`, `coarseR`, `fastCor`, `grid2D` and `minSigCor`.

Examples

```
require(HiClimR)

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Coarsening spatial resolution
xc <- coarseR(x = x, lon = lon, lat = lat, lonStep = 2, latStep = 2)
lon <- xc$lon
lat <- xc$lat
x <- xc$x
```

fastCor *Fast correlation for large matrices*

Description

fastCor is a helper function that compute Pearson correlation matrix for HiClimR and validClimR functions. It is similar to cor function in R but uses a faster implementation on 64-bit machines (an optimized BLAS library is highly recommended). fastCor also uses a memory-efficient algorithm that allows for splitting the data matrix and only compute the upper-triangular part of the correlation matrix. It can be used to compute correlation matrix for the columns of any data matrix.

Usage

```
fastCor(xt, nSplit = 1, upperTri = FALSE, optBLAS = FALSE, verbose = TRUE)
```

Arguments

xt
: an (M rows by N columns) matrix of 'double' values: N objects (spatial points or stations) to be clustered by M observations (temporal points or years). It is the transpose of the input matrix x required for HiClimR and validClimR functions.

nSplit
: integer number greater than or equal to one, to split the data matrix into nSplit splits of the total number of columns ncol(xt). If nSplit = 1, the default method will be used to compute correlation matrix for the full data matrix (no splits). If nSplit > 1, the correlation matrix (or the upper-triangular part if upperTri = TRUE) will be allocated and filled with the computed correlation sub-matrix for each split. the first n-1 splits have equal size while the last split may include any remaining columns. This is used with upperTri = TRUE to compute only the upper-triangular part of the correlation matrix. The maximum number of splits nSplitMax = floor(N / 2) makes splits with 2 columns; if nSplit > nSplitMax, nSplitMax will be used. Very large number of splits nSplit makes computation slower but it could handle big data or if the available memory is not enough to allocate the correlation matrix, which helps in solving the “Error: cannot allocate vector of size...” memory limitation problem. It is recommended to start with a small number of splits. If the data is very large compared to the physical memory, it is highly recommended to use a 64-Bit machine with enough memory resources and/or use coarsening feature for gridded data by setting lonStep > 1 and latStep > 1.

upperTri
: logical to compute only the upper-triangular half of the correlation matrix if upperTri = TRUE and nSplit > 1., which includes all required info since the correlation/dissimilarity matrix is symmetric. This almost halves memory use, which can be very important for big data.

optBLAS
: logical to use optimized BLAS library if installed and optBLAS = TRUE only on 64-bit machines.

verbose
: logical to print processing information if verbose = TRUE.

Details

The fastCor function computes the correlation matrix by calling the cross product function in the Basic Linear Algebra Subroutines (BLAS) library used by R. A significant performance improvement can be achieved when building R on 64-bit machines with an optimized BLAS library, such as *ATLAS*, *OpenBLAS*, or the commercial *Intel MKL*. For big data, the memory required to allocate the square matrix of correlations may exceed the total amount of physical memory available resulting in "Error: cannot allocate vector of size...". fastCor allows for splitting the data matrix into nSplit splits and only computes the upper-triangular part of the correlation matrix with upperTri = TRUE. This almost halves memory use, which can be very important for big data. If nSplit > 1, the correlation matrix (or the upper-triangular part if upperTri = TRUE) will be allocated and filled with computed correlation sub-matrix for each split. the first n-1 splits have equal size while the last split may include any remaining columns.

Value

An (N rows by N columns) correlation matrix.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, https://doi.org/10.1007/s12145-015-0221-7.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, https://cran.r-project.org/package=HiClimR.

See Also

HiClimR, HiClimR2nc, validClimR, geogMask, coarseR, fastCor, grid2D and minSigCor.

Examples

```
require(HiClimR)

## Load test case data
x <- TestCase$x

## Use fastCor function to compute the correlation matrix
t0 <- proc.time() ; xcor <- fastCor(t(x)) ; proc.time() - t0
## compare with cor function
t0 <- proc.time() ; xcor0 <- cor(t(x)) ; proc.time() - t0

## Not run:

## Split the data into 10 splits and return upper-triangular half only
xcor10 <- fastCor(t(x), nSplit = 10, upperTri = TRUE)
```

```
## End(Not run)
```

geogMask *Geographic mask from longitude and latitude*

Description

geogMask is a helper function that preprocess input for the HiClimR via geogMask parameter.

Usage

```
geogMask(continent=NULL, region=NULL, country=NULL, lon=NULL, lat=NULL,
  InDispute=TRUE, verbose = TRUE, plot=FALSE, colPalette=NULL, pch = 15, cex = 1)
```

Arguments

continent	NULL or a string (or array of strings) to specify continent name(s): only one of continent, region, or country should be specified. Valid list of continent names can be obtained by running geogMask().
region	NULL or a string (or array of strings) to specify region name(s): only one of continent, region, or country should be specified. Valid list of region names can be obtained by running geogMask().
country	NULL or a string (or array of strings) to specify country ISO3 character code(s): only one of continent, region, or country should be specified. Valid list of country ISO3 character code(s) can be obtained by running geogMask().
lon	a vector of longitudes with length N. Longitudes takes values from -180 to 180 (not 0 to 360). For gridded data, the length may have the value (n) provided that n * m = N where n = length(unique(lon)) and m = length(unique(lat)).
lat	a vector of latitudes with length N or m. See lon.
InDispute	a logical: should the areas in dispute be considered for geographic masking by country? If InDispute = TRUE (the default), areas in dispute will be considered as a part of the country.
verbose	logical to print processing information if verbose = TRUE.
plot	logical to call the plotting method if plot = TRUE.
colPalette	a color palette or a list of colors such as that generated by rainbow, heat.colors, topo.colors, terrain.colors or similar functions.
pch	Either an integer specifying a symbol or a single character to be used as the default in plotting points. See points for possible values.
cex	A numerical value giving the amount by which plotting symbols should be magnified relative to the default = 1.

Details

In some applications, a user may want to focus on an area that is a mask-defined subset of the full dataset. For instance, the NASA Tropical Rainfall Measuring Mission (TRMM) data covers ocean and land, while a researcher might be interested in the precipitation variability only over land, a country, or a list of countries (e.g., Nile Basin countries). This masking capability is supported by the `geogMask` helper function. It requires the longitude (`lon`) and latitude (`lat`) vectors together with a string (or array of strings) to specify `continent` name(s), `region` name(s), or `country` ISO3 character code(s) via either `continent`, `region`, or `country` parameters. Valid values for them can be obtained by running `geogMask()`. World mask data is based on the HIU Large Scale International Boundaries (LSIB) data (https://hiu.state.gov/data).

Value

A vector of indices for the spatial elements to be masked, as required by `HiClimR`.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, https://doi.org/10.1007/s12145-015-0221-7.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, https://cran.r-project.org/package=HiClimR.

See Also

`HiClimR`, `HiClimR2nc`, `validClimR`, `geogMask`, `coarseR`, `fastCor`, `grid2D` and `minSigCor`.

Examples

```
require(HiClimR)

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Check the valid options for geographic masking
geogMask()

## geographic mask for Africa
gMask <- geogMask(continent = "Africa", lon = lon, lat = lat, plot = TRUE)
```

grid2D *Generate longitude and latitude grid matrices*

Description

grid2D is a helper function that generates longitude and latitude rectangular mesh from short longitude and latitude vectors in gridded data.

Usage

```
grid2D(lon=lon, lat=lat)
```

Arguments

lon	a vector of longitudes with length N. Longitudes takes values from -180 to 180 (not 0 to 360). For gridded data, the length may have the value (n) provided that n * m = N where n = length(unique(lon)) and m = length(unique(lat)).
lat	a vector of latitudes with length N or m. See lon.

Details

grid2D function convert the long latitude and longitude vectors to a rectangular two-dimensional grid for visualization and geographic masking purposes for gridded data in HiClimR package.

Value

A list with the following components:

lon	an (n rows by m columns) matrix of 'double' values for longitude mesh grid, or a vector with length n * m.
lat	an (n rows by m columns) matrix of 'double' values for latitude mesh grid, or a vector with length n * m.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, https://doi.org/10.1007/s12145-015-0221-7.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, https://cran.r-project.org/package=HiClimR.

See Also

HiClimR, HiClimR2nc, validClimR, geogMask, coarseR, fastCor, grid2D and minSigCor.

Examples

```
require(HiClimR)

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)
```

`HiClimR` *Hierarchical Climate Regionalization*

Description

HiClimR is a tool for **Hi**erarchical **Clim**ate **R**egionalization applicable to any correlation-based clustering. Climate regionalization is the process of dividing an area into smaller regions that are homogeneous with respect to a specified climatic metric. Several features are added to facilitate the applications of climate regionalization (or spatiotemporal analysis in general) and to implement cluster validation with an objective tree cutting to find an optimal number of clusters for a user-specified confidence level. These include options for preprocessing and postprocessing as well as efficient code execution for large datasets and options for splitting big data and computing only the upper-triangular half of the correlation/dissimilarity matrix to overcome memory limitations. Hybrid hierarchical clustering reconstructs the upper part of the tree above a cut to get the best of the available methods. Multivariate clustering (MVC) provides options for filtering all variables before preprocessing, detrending and standardization of each variable, and applying weights for the preprocessed variables. The correlation distance for MVC represents the (weighted) average of distances between all variables.

`HiClimR` is the main function that calls all helper functions. It adds several features and a new clustering method (called, *regional* linkage) to hierarchical clustering in R (`hclust` function in *stats* library): data regridding (`grid2D` function), coarsening spatial resolution (`coarseR` function), geographic masking (`geogMask` function), contiguity-constrained clustering, data filtering by mean and/or variance thresholds, data preprocessing (detrending, standardization, and PCA), faster correlation function (`fastCor` function), hybrid hierarchical clustering, multivariate clustering (MVC), cluster validation (`validClimR` and `minSigCor` functions), and visualization of regionalization results, and exporting region map and mean timeseries into NetCDF-4 file.

Badr et al. (2015) describes the regionalization algorithms, features, and data processing tools included in the package and presents a demonstration application in which the package is used to regionalize Africa on the basis of interannual precipitation variability.

HiClimR *is applicable to any correlation-based clustering.*

Usage

```
HiClimR(
# Input data matrix (N spatial elements x M observations)
x = list(),

# Geographic coordinates
lon=NULL, lat=NULL,

# Coarsening spatial resolution
lonStep=1, latStep=1,

# Geographic masking:
geogMask=FALSE, gMask=NULL, continent=NULL, region=NULL, country=NULL,

# Contiguity constraint:
contigConst = 0,

# Data thresholds:
meanThresh = if(inherits(x, ”list”)) vector(”list”, length(x)) else list(NULL),
varThresh = if(inherits(x, ”list”)) as.list(rep(0, length(x))) else list(0),

# Data preprocessing:
detrend = if(inherits(x, ”list”)) as.list(rep(FALSE, length(x))) else list(FALSE),
standardize = if(inherits(x, ”list”)) as.list(rep(FALSE, length(x))) else list(FALSE),
weightMVC = if(inherits(x, ”list”)) as.list(rep(1, length(x))) else list(1),
nPC=NULL,

# Clustering options:
method=”ward”, hybrid=FALSE, kH=NULL, members=NULL,

# Big data support:
nSplit = 1, upperTri = TRUE, verbose = TRUE,

# Cluster validation:
validClimR=TRUE, rawStats=TRUE, k=NULL, minSize=1, alpha=0.05,

# Graphical options:
plot=TRUE, dendrogram = TRUE, colPalette=NULL, hang=-1, labels=FALSE, pch = 15, cex = 1

)
```

Arguments

`x` — an (`N` rows by `M` columns) matrix of ’double’ values: `N` objects (spatial points or stations) to be clustered by `M` observations (temporal points or years). For gridded data, the `N` objects should be created from the original matrix `x0` using `as.vector(t(x0))`, where `x0` is an (`n` rows by `m` columns) matrix, `n = length(unique(lon))` and `m = length(unique(lat))`. Zero-variance rows (e.g., stations with zero variability) and/or missing values (e.g., years with missing observations) are allowed. The zero-variance rows and the columns with missing values will be removed. However, it is recommended to take care of both zero-variance rows and missing values before clustering. For Multivariate Clustering (MVC), x can be a list of nvar matrices for the nvar variables (one matrix for each variable). The matrixes in x list should have the same number of rows (objects: spatial points or stations) Data preprocessing is specified by lists of meanThresh, varThresh, detrend, and standardize with the same length of x where length(x) = nvar. Each variable is separately preprocessed to allow for all possible options. However, it is strongly recommended to standardize all variables since their magnitude range could be different. Note that: for gridded data, the rows of input data matrix for each variable is ordered by longitudes (check TestCase$x for more details).

lon — a vector of longitudes with length N. Longitudes takes values from -180 to 180 (not 0 to 360). For gridded data, the length may have the value (n) provided that n * m = N where n = length(unique(lon)) and m = length(unique(lat)).

lat — a vector of latitudes with length N or m. See lon.

lonStep — an integer greater than or equal to 1 for longitude step to coarsen gridded data in the longitudinal direction. If lonStep = 1, gridded data will not be coarsened in the longitudinal direction (the default). If lonStep = 2, every other grid in longitudinal direction will be retained.

latStep — an integer greater than or equal to 1 for latitude step to coarsen gridded data in the latitudinal direction. If latStep = 1, gridded data will not be coarsened in the latitudinal direction (the default). If latStep = 2, every other grid in latitudinal direction will be retained. lonStep and latStep are independent so that user can optionally apply different coarsening level to each dimension.

geogMask — a logical: if geogMask = TRUE, geogMask function will be called. Additional arguments are required. It requires the longitude and latitude vector together with a string (or array of strings) to specify continent, region name(s), or country ISO3 character code(s). If gMask != NULL, the provided gmask vector will be used for geographic masking without calling geogMask.

gMask — A vector of indices for the spatial elements to be masked, as required by HiClimR. This is typically an output vector from geogMask function. This helps in saving time when the same geographic mask will be used many times.

continent — NULL or a string (or array of strings) to specify continent name(s): only one of continent, region, or country should be specified. Valid list of continent names can be obtained by running geogMask().

region — NULL or a string (or array of strings) to specify region name(s): only one of continent, region, or country should be specified. Valid list of region names can be obtained by running geogMask().

country — NULL or a string (or array of strings) to specify country ISO3 character code(s): only one of continent, region, or country should be specified. Valid list of country ISO3 character code(s) can be obtained by running geogMask().

contigConst — 0 or a positive value for contiguity constraint: This is used to apply weighted geographic distances on the initial dissimilarity matrix between the variable(s) in the data matrix x. It should be a positive real value (contigConst > 0, recommended between 0 and 1) where contigConst = 0 means that clustering is completely based on the data with no constraints for contiguity while contigConst

`> 0` gives priority to merging contiguous points. Geographic coordinates `lon` and `lat` are required to compute the geographic distances. `regional` linkage method does not support contiguity constraints.

`meanThresh` `NULL` or a threshold for the temporal mean: This is used with `varThresh` to mask zero- and near-zero-variance data, Observations with mean less than or equal to `meanThresh` will be removed. If `meanThresh = NULL`, then the `varThresh` will be used either to mask zero-variance data by default or by increased variance threshold to mask near-zero-variance data. For Multivariate Clustering (MVC), `meanThresh` is a list of thresholds with the same length of `x` where `length(x) = nvar`. Each variable is separately preprocessed to allow for all possible options. However, it is strongly recommended to standardize all variables since their magnitude range could be different.

`varThresh` zero or a threshold for the temporal variance: This is used with `meanThresh` to mask zero- and near-zero-variance data, Observations with variance less than or equal to `varThresh` will be removed. If `varThresh = 0`, then the zero-variance data will masked (default). For Multivariate Clustering (MVC), `varThresh` is a list of thresholds with the same length of `x` where `length(x) = nvar`. Each variable is separately preprocessed to allow for all possible options. However, it is strongly recommended to standardize all variables since their magnitude range could be different.

`detrend` a logical: should the data be detrended before clustering? Detrending (removing the linear trend) is important when variations from temporal point to another is of interest (e.g., interannual variability). The columns of the data matrix `x` should be temporally ordered (constant step size) or have appropriate names (e.g., `colnames(x) = years[1:M]`). For Multivariate Clustering (MVC), `detrend` is a list of thresholds with the same length of `x` where `length(x) = nvar`. Each variable is separately preprocessed to allow for all possible options. However, it is strongly recommended to standardize all variables since their magnitude range could be different.

`standardize` a logical: should the data be standardized before clustering? The standardized data makes use of the mean of equally-weighted objects within each cluster (cluster mean = mean of standardized variables within the cluster). Otherwise, the mean of raw data will be used (cluster mean = mean of raw variables within the cluster). The variance of the mean is updated at each agglomeration step. For Multivariate Clustering (MVC), `standardize` is a list of thresholds with the same length of `x` where `length(x) = nvar`. Each variable is separately preprocessed to allow for all possible options. However, it is strongly recommended to standardize all variables since their magnitude range could be different.

`weightMVC` a list of positive wights (`weightMVC > 0`) for Multivariate Clustering (MVC) with the same length of `x` where `length(x) = number of variables`. The filtered variables are weighted and combined by column (for each object: spatial points or stations) after preprocessing (detrending and standardization) and before PCA (if requested) and computing the correlation/dissimilarity matrix. The default weight is `weightMVC = 1` for all variables.

`nPC` `NULL` or number of principal components (PCs) to be retained. If `nPC = NULL`, then the raw data will be used for clustering. Otherwise, the data will be filtered and reconstructed using `nPC` PCs obtained from SVD-based PCA. The `detrend` and/or `standardize` options will be applied, if requested, before PCA.

`method`
the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of `"regional"`, `"ward"`, `"single"`, `"complete"`, `"average"`, `"mcquitty"`, `"median"` or `"centroid"`. The default is `"ward"` method.

`hybrid`
a logical: should the upper part of the tree be reconstructed using `regional` linkage clustering method? This adds hybrid hierarchical clustering feature to get the best of the available methods. It utilizes the pros of available methods, especially the better overall homogeneity in *ward*'s method and the separation, contiguity, and objective tree cut of *regional* linkage method. If `hybrid = FALSE`, only the default clustering using the selected `method` will be used (i.e., no hybrid clustering).Otherwise, the upper part of the tree will be reconstructed above a cut of `kH` clusters using `regional` linkage method. Note: `hybrid` option is redundant when using `regional` linkage as the main clustering method.

`kH`
`NULL` or an integer for the number of regions/clusters in the upper part of the tree to be reconstructed with `regional` linkage method, if `hybrid = TRUE`. If `kH = NULL`, the tree will be reconstructed for the upper part with the first merging cost larger than the mean merging cost for the entire tree. If hybrid clustering is requested, the updated upper part of the tree will be used for cluster validation, and so `kH` should be greater than the final number of clusters `k`, if selected.

`members`
`NULL` or a vector with length size of `d`. See the 'Details' section.

`nSplit`
integer number greater than or equal to one, to split the data matrix into `nSplit` splits of the total number of columns `ncol(xt)`. If `nSplit = 1`, the default method will be used to compute correlation matrix for the full data matrix (no splits). If `nSplit > 1`, the correlation matrix (or the upper-triangular part if `upperTri = TRUE`) will be allocated and filled with the computed correlation sub-matrix for each split. the first `n-1` splits have equal size while the last split may include any remaining columns. This is used with `upperTri = TRUE` to compute only the upper-triangular part of the correlation matrix. The maximum number of splits `nSplitMax = floor(N / 2)` makes splits with 2 columns; if `nSplit > nSplitMax`, `nSplitMax` will be used. Very large number of splits `nSplit` makes computation slower but it could handle big data or if the available memory is not enough to allocate the correlation matrix, which helps in solving the "Error: cannot allocate vector of size..." memory limitation problem. It is recommended to start with a small number of splits. If the data is very large compared to the physical memory, it is highly recommended to use a 64-Bit machine with enough memory resources and/or use coarsening feature for gridded data by setting `lonStep > 1` and `latStep > 1`.

`upperTri`
logical to compute only the upper-triangular half of the correlation matrix if `upperTri = TRUE` and `nSplit > 1`., which includes all required info since the correlation/dissimilarity matrix is symmetric. This almost halves memory use, which can be very important for big data.

`verbose`
logical to print processing information if `verbose = TRUE`.

`validClimR`
a logical: If `validClimR = TRUE`, `validClimR` will be called to compute validation indices including statistical summary for inter- and intra-cluster correlations. This is computationally expensive. It can also objectively cut the dendrogram tree for `regional` clustering method, if `k = NULL`.

rawStats — a logical: should validation indices be computed based on the raw data or PCA-filtered data?

k — NULL or an integer k > 1 for the number of regions/clusters. Only for regional linkage method, k = NULL is supported, where the "optimal" number of regions will be used at a user specified significance level alpha. It is required to specify number of clusters k for the other methods, since they are not based on inter-cluster correlation. If k = NULL for these methods (except regional) linkage, the validClimR with be aborted. One can use validClimR function to compute inter-cluster correlation at different number of clusters to objectively cut the tree for the other methods, which could be computationally expensive to cover the entire merging history for large number of spatial elements.

minSize — minimum cluster size. The regional linkage method tend to isolate noisy data in small clusters. The minSize can be used to exclude these very small clusters from the statSum statistical summary, because they are most likely noisy data that need to be checked in a quality control step. The analysis may be then repeated.

alpha — confidence level: the default is alpha = 0.05 for 95% confidence level.

plot — logical to call the plotting method if plot = TRUE.

dendrogram — logical to enable or disable dendrogram plotting.

colPalette — a color palette or a list of colors such as that generated by rainbow, heat.colors, topo.colors, terrain.colors or similar functions.

hang — The fraction of the plot height by which labels should hang below the rest of the plot. A negative value will cause the labels to hang down from 0.

labels — A character vector of labels for the leaves of the tree. By default the row names or row numbers of the original data are used. If labels = FALSE no labels at all are plotted.

pch — Either an integer specifying a symbol or a single character to be used as the default in plotting points. See points for possible values.

cex — A numerical value giving the amount by which plotting symbols should be magnified relative to the default = 1.

Details

HiClimR function is based on hclust, which now uses an optimized algorithm to deal with only the upper/lower triangular-half of the symmetric dissimilarity matrix instead of the old algorithm that uses the full matrix in the merging steps. It performs a hierarchical cluster analysis using Pearson correlation distance dissimilarity for the N objects being clustered. Initially, each object is assigned to its own cluster and then the algorithm proceeds iteratively, at each stage joining the two most similar clusters, continuing until there is just a single cluster. At each stage distances between clusters are recomputed by a dissimilarity update formula according to the particular clustering method being used.

All clustering methods in hclust are included. The *regional* linkage method minimizes inter-cluster correlations between cluster means (see Badr et al. 2015). *Ward's* minimum variance method aims at finding compact, spherical clusters. The *complete linkage* method finds similar clusters. The *single linkage* method (which is closely related to the minimal spanning tree) adopts a ‘friends of friends' clustering strategy. The other methods can be regarded as aiming for clusters with characteristics somewhere between the single and complete link methods. Note however, that methods `"median"` and `"centroid"` are *not* leading to a *monotone distance* measure, or equivalently the resulting dendrograms can have so called *inversions* (which are hard to interpret).

The `regional` linkage method is explained in the context of a spatiotemporal problem, in which `N` spatial elements (e.g., weather stations) are divided into `k` regions, given that each element has a time series of length `M`. It is based on inter-regional correlation distance between the temporal means of different regions (or elements at the first merging step). It modifies the update formulae of `average` linkage method by incorporating the standard deviation of the merged region timeseries, which is a function of the correlation between the individual regions, and their standard deviations before merging. It is equal to the average of their standard deviations if and only if the correlation between the two merged regions is `100%`. In this special case, the `regional` linkage method is reduced to the classic `average` linkage clustering method.

If `members != NULL`, then `d` is taken to be a dissimilarity matrix between clusters instead of dissimilarities between singletons and `members` gives the number of observations per cluster. This way the hierarchical cluster algorithm can be 'started in the middle of the dendrogram', e.g., in order to reconstruct the part of the tree above a cut (see examples). Dissimilarities between clusters can be efficiently computed (i.e., without `hclust` itself) only for a limited number of distance/linkage combinations, the simplest one being squared Euclidean distance and centroid linkage. In this case the dissimilarities between the clusters are the squared Euclidean distances between cluster means.

In hierarchical cluster displays, a decision is needed at each merge to specify which subtree should go on the left and which on the right. Since, for n observations there are $n - 1$ merges, there are $2^{(n-1)}$ possible orderings for the leaves in a cluster tree, or dendrogram. The algorithm used in `hclust` is to order the subtree so that the tighter cluster is on the left (the last, i.e., most recent, merge of the left subtree is at a lower value than the last merge of the right subtree). Single observations are the tightest clusters possible, and merges involving two observations place them in order by their observation sequence number.

Value

An object of class `HiClimR` and `hclust`, which describes the tree produced by the clustering process. The object is a list with the following components:

`merge` an $n - 1$ by 2 matrix. Row i of `merge` describes the merging of clusters at step i of the clustering. If an element j in the row is negative, then observation $-j$ was merged at this stage. If j is positive then the merge was with the cluster formed at the (earlier) stage j of the algorithm. Thus negative entries in `merge` indicate agglomerations of singletons, and positive entries indicate agglomerations of non-singletons.

`height` a set of $n - 1$ real values (non-decreasing for ultrametric trees). The clustering *height*: that is, the value of the criterion associated with the clustering `method` for the particular agglomeration.

`order` a vector giving the permutation of the original observations suitable for plotting, in the sense that a cluster plot using this ordering and matrix `merge` will not have crossings of the branches.

`labels` labels for each of the objects being clustered.

`method` the cluster method that has been used.

`call` the call which produced the result.

`dist.method` the distance that has been used to create `d` (only returned if the distance object has a `"method"` attribute).

`skip` a vector of `lonStep` and `latStep` if `coarseR = TRUE`.

`PCA` if `nPC != NULL`, the eigenvalues, explained variance, and accumulated variance will be stored here.

`coords` an (`Nc` rows by `2` columns) matrix of 'double' values: longitude and latitude coordinates for the preprocessed data used for clustering, where `NC` is the number of spatial elements to be clustered after coarsening spatial resolution by `lonStep` and `latStep`. It will be returned only if `lon` and `lat` vectors were defined. It will be similar to the provided `lon` and `lat` if `lonStep = 1` and `latStep = 1`.

`nvars` number of variables used for multivariate clustering (MVC).

`ncols` number of columns for each variable.

`data` the preprocessed data used for clustering will be stored here. If `rawData = TRUE` and `nPC != NULL`, the preprocessed data without filtering (PCA) will be returned here.

`mask` a vector of the indices of masked spatial points by both geographic masking and data thresholds.

`treeH` An object of class `hclust`, which describes the upper part of the tree reconstructed by the hybrid clustering process if `hybrid = TRUE`.

If `validClimR = TRUE`, an object of class `HiClimR`, which produces indices for validating the tree produced by the clustering process, will be merged in the dendrogram tree above. This object is a list with the following components:

`cutLevel` the minimum significant correlation used for objective tree cut together with the corresponding confidence level.

`clustMean` the cluster means which are the region's mean timeseries for all selected regions.

`clustSize` cluster sizes for all selected regions.

`clustFlag` a flag `0 or 1` to indicate the cluster used in `statSum` validation indices (`interCor`, `intraCor`, `diffCor`, and `statSum`), based on `minSize` minimum cluster size. If `clustFlag = 0`, the cluster has been excluded because its size is less than the `minSize` minimum cluster size. The sum of `clustFlag` elements represents the selected number clusters.

`interCor` inter-cluster correlations for all selected regions. It is the inter-cluster correlations between cluster means. The maximum inter-cluster correlation is a measure for separation or contiguity, and it is used for objective tree cut (to find the "optimal" number of clusters).

`intraCor` intra-cluster correlations for all selected regions. It is the intra-cluster correlations between the mean of each cluster and its members. The average intra-cluster correlation is a weighted average for all clusters, and it is a measure for homogeneity.

`diffCor` difference between intra-cluster correlation and maximum inter-cluster correlation for all selected regions.

`statSum` overall statistical summary for `interCluster`, `intraCor`, and `diffCor`.

`region` ordered regions vector of size `N` number of spatial elements for the selected number of clusters, after excluding the small clusters defined by `minSize` argument.

`regionID` ordered regions ID vector of length equals the selected number of clusters, after excluding the small clusters defined by `minSize` argument. It helps in mapping ordered regions and their actual names before ordering. Only the `region` component uses ordered ID, while other components use the names used during the clustering process.

There are `print`, `plot` and `identify` (see `identify.hclust`) methods and `rect.hclust()` functions for `hclust` objects.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>. The `HiClimR` is a modification of `hclust` function, which is based on Fortran code contributed to STATLIB by F. Murtagh.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, `https://doi.org/10.1007/s12145-015-0221-7`.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, `https://cran.r-project.org/package=HiClimR`.

Wilks, D. S. (2011): *Statistical methods in the atmospheric sciences*, Academic press.

Gordon, A. D. (1999): *Classification*. Second Edition. London: Chapman and Hall / CRC

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988): *The New S Language*. Wadsworth & Brooks/Cole. (S version.)

Murtagh, F. (1985): “Multidimensional Clustering Algorithms”, in *COMPSTAT Lectures 4*. Wuerzburg: Physica-Verlag (for algorithmic details of algorithms used).

Hartigan, J. A. (1975): *Clustering Algorithms*. New York: Wiley.

Everitt, B. (1974): *Cluster Analysis*. London: Heinemann Educ. Books.

Anderberg, M. R. (1973): *Cluster Analysis for Applications*. Academic Press: New York.

Sneath, P. H. A. and R. R. Sokal (1973): *Numerical Taxonomy*. San Francisco: Freeman.

McQuitty, L.L. (1966): Similarity Analysis by Reciprocal Pairs for Discrete and Continuous Data. *Educational and Psychological Measurement*, **26**, 825-831.

Source Code: `https://github.com/hsbadr/HiClimR`

See Also

`HiClimR`, `HiClimR2nc`, `validClimR`, `geogMask`, `coarseR`, `fastCor`, `grid2D`, `minSigCor`. `identify.hclust`, `rect.hclust`, `cutree`, `dendrogram`, and `kmeans`.

Examples

```
require(HiClimR)

#----------------------------------------------------------------------------------#
# Typical use of HiClimR for single-variate clustering:                            #
#----------------------------------------------------------------------------------#

## Load the test data included/loaded in the package (1 degree resolution)
x <- TestCase$x
lon <- TestCase$lon
lat <- TestCase$lat

## Generate/check longitude and latitude mesh vectors for gridded data
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Single-Variate Hierarchical Climate Regionalization
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## For more examples: https://github.com/hsbadr/HiClimR#examples

## Not run:

#----------------------------------------------------------------------------------#
# Additional Examples:                                                             #
#----------------------------------------------------------------------------------#

## Use Ward's method
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Use data splitting for big data
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = TRUE, kH = NULL,
    members = NULL, nSplit = 10, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Use hybrid Ward-Regional method
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = TRUE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)
## Check senitivity to kH for the hybrid method above


#----------------------------------------------------------------------------------#
# Typical use of HiClimR for multivariate clustering:                              #
#----------------------------------------------------------------------------------#

## Load the test data included/loaded in the package (1 degree resolution)
x1 <- TestCase$x
lon <- TestCase$lon
lat <- TestCase$lat

## Generate/check longitude and latitude mesh vectors for gridded data
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Test if we can replicate single-variate region map with repeated variable
y <- HiClimR(x=list(x1, x1), lon = lon, lat = lat, lonStep = 1, latStep = 1,
    geogMask = FALSE, continent = "Africa", meanThresh = list(10, 10),
    varThresh = list(0, 0), detrend = list(TRUE, TRUE), standardize = list(TRUE, TRUE),
    nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Generate a random matrix with the same number of rows
x2 <- matrix(rnorm(nrow(x1) * 100, mean=0, sd=1), nrow(x1), 100)

## Multivariate Hierarchical Climate Regionalization
y <- HiClimR(x=list(x1, x2), lon = lon, lat = lat, lonStep = 1, latStep = 1,
    geogMask = FALSE, continent = "Africa", meanThresh = list(10, NULL),
    varThresh = list(0, 0), detrend = list(TRUE, FALSE), standardize = list(TRUE, TRUE),
    weightMVC = list(1, 1), nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)
## You can apply all clustering methods and options

#----------------------------------------------------------------------------------#
# Miscellaneous examples to provide more information about functionality and usage #
# of the helper functions that can be used separately or for other applications.   #
#----------------------------------------------------------------------------------#

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
```

```
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Coarsening spatial resolution
xc <- coarseR(x = x, lon = lon, lat = lat, lonStep = 2, latStep = 2)
lon <- xc$lon
lat <- xc$lat
x <- xc$x

## Use fastCor function to compute the correlation matrix
t0 <- proc.time(); xcor <- fastCor(t(x)); proc.time() - t0
## compare with cor function
t0 <- proc.time(); xcor0 <- cor(t(x)); proc.time() - t0

## Check the valid options for geographic masking
geogMask()

## geographic mask for Africa
gMask <- geogMask(continent = "Africa", lon = lon, lat = lat, plot = TRUE,
    colPalette = NULL)

## Hierarchical Climate Regionalization Without geographic masking
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## With geographic masking (you may specify the mask produced above to save time)
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = TRUE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## With geographic masking and contiguity contraint
## Change contigConst as appropriate
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = TRUE,
    continent = "Africa", contigConst = 1, meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE, kH = NULL,
    members = NULL, nSplit = 1, upperTri = TRUE, verbose = TRUE,
    validClimR = TRUE, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Find minimum significant correlation at 95
rMin <- minSigCor(n = nrow(x), alpha = 0.05, r = seq(0, 1, by = 1e-06))

## Validtion of Hierarchical Climate Regionalization
z <- validClimR(y, k = 12, minSize = 1, alpha = 0.01,
    plot = TRUE, colPalette = NULL)
```

```
## Apply minimum cluster size (minSize = 25)
z <- validClimR(y, k = 12, minSize = 25, alpha = 0.01,
    plot = TRUE, colPalette = NULL)

## The optimal number of clusters, including small clusters
k <- length(z$clustFlag)

## The selected number of clusters, after excluding small clusters (if minSize > 1)
ks <- sum(z$clustFlag)

## Dendrogram plot
plot(y, hang = -1, labels = FALSE)

## Tree cut
cutTree <- cutree(y, k = k)
table(cutTree)

## Visualization for gridded data
RegionsMap <- matrix(y$region, nrow = length(unique(y$coords[, 1])), byrow = TRUE)
colPalette <- colorRampPalette(c("#00007F", "blue", "#007FFF", "cyan",
    "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000"))
image(unique(y$coords[, 1]), unique(y$coords[, 2]), RegionsMap, col = colPalette(ks))

## Visualization for gridded or ungridded data
plot(y$coords[, 1], y$coords[, 2], col = colPalette(max(Regions, na.rm = TRUE))[y$region],
    pch = 15, cex = 1)

## Export region map and mean timeseries into NetCDF-4 file
y.nc <- HiClimR2nc(y=y, ncfile="HiClimR.nc", timeunit="years", dataunit="mm")
## The NetCDF-4 file is still open to add other variables or close it
nc_close(y.nc)


## End(Not run)
```

HiClimR2nc *Export NetCDF-4 file for Hierarchical Climate Regionalization*

Description

HiClimR2nc is a helper function that exports region map and mean timeseries into NetCDF-4 file, using the ncdf4 package.

Usage

```
HiClimR2nc(y=NULL, ncfile="HiClimR.nc", timeunit="", dataunit="")
```

Arguments

`y` a dendrogram tree produced by `HiClimR`.

`ncfile` Path and name of the NetCDF-4 file to be created.

`timeunit` an optional character string for the time units, A zero-length string (default: `timeunit=""`) removes units attribute.

`dataunit` an optional character string for the data units, A zero-length string (default: `timeunit=""`) removes units attribute.

Details

`HiClimR2nc` function exports region map and mean timeseries from `HiClimR` tree into NetCDF-4 file, using the `ncdf4` package. The NetCDF-4 file will be open to add other variables, if needed. It is important to close the created file using `nc_close`, which flushes any unwritten data to disk.

Value

An object of class `ncdf4`, which has the fields described in `nc_open`.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, `https://doi.org/10.1007/s12145-015-0221-7`.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, `https://cran.r-project.org/package=HiClimR`.

See Also

`HiClimR`, `HiClimR2nc`, `validClimR`, `geogMask`, `coarseR`, `fastCor`, `grid2D` and `minSigCor`.

Examples

```
require(HiClimR)
require(ncdf4)

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Hierarchical Climate Regionalization
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
        standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE,
        kH = NULL, members = NULL, validClimR = TRUE, k = 12, minSize = 1,
        alpha = 0.01, plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Not run:

## Export region map and mean timeseries into NetCDF-4 file
y.nc <- HiClimR2nc(y=y, ncfile="HiClimR.nc", timeunit="years", dataunit="mm")
## The NetCDF-4 file is still open to add other variables or close it
nc_close(y.nc)


## End(Not run)
```

minSigCor — *Minimum significant correlation for a sample size*

Description

minSigCor is a helper function that estimates the minimum significant correlation for a sample size n at a confidence level defined by the argument alpha.

Usage

```
minSigCor(n=41, alpha=0.05, r=seq(0, 1, by=1e-6))
```

Arguments

n	sample size or the length of a timeseries vector.
alpha	confidence level: the default is alpha = 0.05 for 95% confidence level.
r	a vector of values from 0 to 1 to search for the minimum significant correlation for the user-specified sample size n at confidence level alpha. This should be a subset of the valid positive correlation range 0-1. The default is to search for the minimum significant correlation in the complete range 0-1 with a very fine step of 1e-6. For faster computations, the user may set a shorter range with larger step (e.g., seq(0.1, 0.5, by=1e-3)).

Details

minSigCor function estimates the minimum significant correlation for a sample size (number of observations or temporal points in a timeseries) at a certain confidence level selected by the argument alpha and an optional search range r. It is called by validClimR function objective tree cut based on the specified confidence level.

Value

A positive value between 0 and 1 for the estimated the minimum significant correlation.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, https://doi.org/10.1007/s12145-015-0221-7.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, https://cran.r-project.org/package=HiClimR.

See Also

HiClimR, HiClimR2nc, validClimR, geogMask, coarseR, fastCor, grid2D and minSigCor.

Examples

```
require(HiClimR)

## Find minimum significant correlation at 95% confidence level
rMin <- minSigCor(n = 41, alpha = 0.05, r = seq(0, 1, by = 1e-06))
```

TestCase *Test Data for Functionality Demonstration of* HiClimR *Package*

Description

This data is a subset of University of East Anglia Climatic Research Unit (CRU) TS (timeseries) precipitation dataset version 3.2.

Usage

```
data(TestCase)
```

Format

`TestCase` is a list of three components: `x`, `lon`, and `lat`. `x` is an (`6400` rows by `41` columns) matrix as required for `HiClimR` function. The rows represent spatial points (or stations), while the columns represent observations (temporal points or years). `lon` and `lat` are vectors of length `80` for unique longitudes and latitudes coordinates, where `80 * 80 = 6400` for this gridded data.

Details

CRU TS 3.21 data (1901-2012) is monthly gridded precipitation with `0.5` degree resolution. This test data is a subset with 1 degree resolution for African precipitation in January, 1949-1989.

Source

Climatic Research Unit (CRU) time-series datasets of variations in climate with variations in other phenomena.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, https://doi.org/10.1007/s12145-015-0221-7.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, https://cran.r-project.org/package=HiClimR.

Harris, I., Jones, P. D., Osborn, T. J., and Lister, D. H. (2014): Updated high-resolution grids of monthly climatic observations - the CRU TS3.10 Dataset, *International journal of climatology*, **34**, 623-642, https://doi.org/10.1002/joc.3711.

Examples

```
require(HiClimR)

x <- TestCase$x
dim(x)
colnames(x)
```

---

validClimR *Validation of Hierarchical Climate Regionalization*

Description

validClimR computes indices for cluster validation, and an objective tree cut for regional linkage clustering method.

Usage

```
validClimR(y=NULL, k=NULL, minSize=1, alpha=0.05, verbose = TRUE,
    plot=FALSE, colPalette=NULL, pch = 15, cex = 1)
```

Arguments

y	a dendrogram tree produced by HiClimR.
k	NULL or a n integer k > 1 for the number of regions/clusters. Only for regional linkage method, k = NULL is supported, where the "optimal" number of regions will be used at a user specified significance level alpha. It is required to specify number of clusters k for the other methods, since they are not based on inter-cluster correlation. If k = NULL for these methods (except regional) linkage, the validClimR with be aborted. One can use validClimR function to compute inter-cluster correlation at different number of clusters to objectively cut the tree for the other methods, which could be computationally expensive to cover the entire merging history for large number of spatial elements.
minSize	minimum cluster size. The regional linkage method tend to isolate noisy data in small clusters. The minSize can be used to exclude these very small clusters from the statSum statistical summary, because they are most likely noisy data that need to be checked in a quality control step. The analysis may be then repeated.

alpha — confidence level: the default is alpha = 0.05 for 95% confidence level.

verbose — logical to print processing information if verbose = TRUE.

plot — logical to call the plotting method if plot = TRUE.

colPalette — a color palette or a list of colors such as that generated by rainbow, heat.colors, topo.colors, terrain.colors or similar functions.

pch — Either an integer specifying a symbol or a single character to be used as the default in plotting points. See points for possible values.

cex — A numerical value giving the amount by which plotting symbols should be magnified relative to the default = 1.

Details

The validClimR function is used for validation of a dendrogram tree produced by HiClimR, by computing detailed statistical information for each cluster about cluster means, sizes, intra- and inter-cluster correlations, and overall summary. It requires the preprocessed data matrix and the tree from HiClimR function as inputs. An optional parameter can be used to validate clustering for a selected number of clusters k. If k = NULL, the default which supports only the regional linkage method, objective cutting of the tree to find the optimal number of clusters will be applied based on a user specified significance level (alpha parameter). In regional linkage method, noisy spatial elements are isolated in very small-size clusters or individuals since they do not correlate well with any other elements. They can be excluded from the validation indices (interCor, intraCor, diffCor, and statSum), based on minSize minimum cluster size. The excluded clusters are identified in the output of validClimR in clustFlag, which takes a value of 1 for selected clusters or 0 for excluded clusters. The sum of clustFlag elements represents the selected number clusters.This should be followed by a quality control step before repeating the analysis.

Value

An object of class HiClimR which produces indices for validating the tree produced by the clustering process. The object is a list with the following components:

cutLevel — the minimum significant correlation used for objective tree cut together with the corresponding confidence level.

clustMean — the cluster means which are the region's mean timeseries for all selected regions.

clustSize — cluster sizes for all selected regions.

clustFlag — a flag 0 or 1 to indicate the cluster used in statSum validation indices (interCor, intraCor, diffCor, and statSum), based on minSize minimum cluster size. If clustFlag = 0, the cluster has been excluded because its size is less than the minSize minimum cluster size. The sum of clustFlag elements represents the selected number clusters.

interCor — inter-cluster correlations for all selected regions. It is the inter-cluster correlations between cluster means. The maximum inter-cluster correlation is a measure for separation or contiguity, and it is used for objective tree cut (to find the "optimal" number of clusters).

`intraCor` intra-cluster correlations for all selected regions. It is the intra-cluster correlations between the mean of each cluster and its members. The average intra-cluster correlation is a weighted average for all clusters, and it is a measure for homogeneity.

`diffCor` difference between intra-cluster correlation and maximum inter-cluster correlation for all selected regions.

`statSum` overall statistical summary for `interCluster`, `intraCor`, and `diffCor`.

`region` ordered regions vector of size `N` number of spatial elements for the selected number of clusters, after excluding the small clusters defined by `minSize` argument.

`regionID` ordered regions ID vector of length equals the selected number of clusters, after excluding the small clusters defined by `minSize` argument. It helps in mapping ordered regions and their actual names before ordering. Only the `region` component uses ordered ID, while other components use the names used during the clustering process.

Author(s)

Hamada Badr <badr@jhu.edu>, Ben Zaitchik <zaitchik@jhu.edu>, and Amin Dezfuli <dez@jhu.edu>. The `HiClimR` is a modification of `hclust` function, which is based on Fortran code contributed to STATLIB by F. Murtagh.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, `https://doi.org/10.1007/s12145-015-0221-7`.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, `https://cran.r-project.org/package=HiClimR`.

See Also

`HiClimR`, `HiClimR2nc`, `validClimR`, `geogMask`, `coarseR`, `fastCor`, `grid2D` and `minSigCor`.

Examples

```
require(HiClimR)

## Load test case data
x <- TestCase$x

## Generate longitude and latitude mesh vectors
xGrid <- grid2D(lon = unique(TestCase$lon), lat = unique(TestCase$lat))
lon <- c(xGrid$lon)
lat <- c(xGrid$lat)

## Hierarchical Climate Regionalization
y <- HiClimR(x, lon = lon, lat = lat, lonStep = 1, latStep = 1, geogMask = FALSE,
    continent = "Africa", meanThresh = 10, varThresh = 0, detrend = TRUE,
    standardize = TRUE, nPC = NULL, method = "ward", hybrid = FALSE,
    kH = NULL, members = NULL, validClimR = TRUE, k = 12, minSize = 1,
    alpha = 0.01, plot = TRUE, colPalette = NULL, hang = -1, labels = FALSE)

## Validtion of Hierarchical Climate Regionalization
z <- validClimR(y, k = 12, minSize = 1, alpha = 0.01, plot = TRUE)

## Use a specified number of clusters (k = 12)
z <- validClimR(y, k = 12, minSize = 1, alpha = 0.01, plot = TRUE)

## Apply minimum cluster size (minSize = 25)
z <- validClimR(y, k = 12, minSize = 25, alpha = 0.01, plot = TRUE)

## The optimal number of clusters, including small clusters
k <- length(z$clustFlag)

## The selected number of clusters, after excluding small clusters (if minSize > 1)
ks <- sum(z$clustFlag)
```

WorldMask *World Mask for Geographic Masking in HiClimR*

Description

This data is used for geographic masking by `geogMask` function in `HiClimR` package.

Usage

```
data(WorldMask)
```

Format

`WorldMask` is a list with two components: `info` and `mask`. `info` is an (`284` rows by `10` columns) matrix. The rows are for areas or countries while the columns are for codes required by `geogMask`. `mask` is an (`3601` rows by `1801` columns) matrix with integer values from `1` to `284` for the areas defined in `info`.

Details

This data is used internally by `geogMask` function for geographic masking in `HiClimR` package. The user is advised to refer to the function manual for more details. The world mask is available in `0.1` degree (`10` km) resolution. The `info` data provides information for continents, regions, and country codes).

Source

The data are based on the Humanitarian Information Unit (HIU) Large Scale International Boundaries (LSIB) dataset.

References

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2015): A Tool for Hierarchical Climate Regionalization, *Earth Science Informatics*, **8**(4), 949-958, `https://doi.org/10.1007/s12145-015-0221-7`.

Hamada S. Badr, Zaitchik, B. F. and Dezfuli, A. K. (2014): Hierarchical Climate Regionalization, *Comprehensive R Archive Network (CRAN*, `https://cran.r-project.org/package=HiClimR`.

LSIB Data: `https://hiu.state.gov/data/`.

Examples

```
require(HiClimR)

geogMask()
```

Index