Package ‘envirem’

January 29, 2019

Type Package

Title Generation of ENVIREM Variables

Version 1.4

Date 2019-01-29

Author Pascal O. Title, Jordan B. Bemmels

Maintainer Pascal Title <ptitle@umich.edu>

Depends raster

Imports RSAGA

Suggests rgdal

Description Generation of bioclimatic rasters that will be particularly useful for species distribution modeling.

License GPL (>= 2)

URL http://envirem.github.io

BugReports https://github.com/ptitle/envirem/issues

NeedsCompilation yes

Encoding UTF-8

RoxygenNote 6.0.1

ByteCompile true

Repository CRAN

Date/Publication 2019-01-29 18:50:03 UTC

R topics documented:

`aridityIndexThornthwaite`

aridityIndexThornthwaite

Description

Generates thornthwaite aridity index raster.

Usage

```
aridityIndexThornthwaite(precipStack, PETstack)
```

Arguments

`precipStack`	rasterStack of monthly precipitation.
`PETstack`	rasterStack of monthly potential evapotranspiration.

Details

`Thornthwaite aridity index = 100d / n` where d = sum of monthly differences between precipitation and PET for months where precip < PET where n = sum of monthly PET for those months

Value

RasterLayer, unitless

Author(s)

Pascal Title

References

Thornthwaite, C.W. (1948). An approach toward a rational classification of climate. *Geographical Review*, **38**, 55-94.

See Also

Requires rasters created with `monthlyPET`.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
solar <- grep('solrad', names(env), value=TRUE)
maxtemp <- grep('tmax', names(env), value=TRUE)
mintemp <- grep('tmin', names(env), value=TRUE)

# read them in as rasterStacks
meantemp <- stack(env[[meantemp]])
solar <- stack(env[[solar]])
maxtemp <- stack(env[[maxtemp]])
mintemp <- stack(env[[mintemp]])
tempRange <- abs(maxtemp - mintemp)

# get monthly PET
pet <- monthlyPET(meantemp, solar, tempRange)

precip <- grep('prec', names(env), value=TRUE)
precip <- stack(env[[precip]])

aridityIndexThornthwaite(precip, pet)
```

`climaticMoistureIndex` *Climatic Moisture Index*

Description

Generate climatic moisture index.

Usage

```
climaticMoistureIndex(annualPrecip, PET)
```

Arguments

`annualPrecip`	rasterLayer of annual precipitation
`PET`	rasterLayer of annual potential evapotranspiration

Details

`P/PET - 1` when `P < PET`
`1 - PET/P` when `P >= PET`

Value

rasterLayer ranging from -1 to +1.

Author(s)

Pascal Title

References

Willmott, C. & Feddema, J. (1992). A More Rational Climatic Moisture Index. *The Professional Geographer*, **44**, 84-88.

Vörösmarty, C.J., Douglas, E.M., Green, P.A. & Revenga, C. (2005). Geospatial Indicators of Emerging Water Stress: An Application to Africa. *AMBIO: A Journal of the Human Environment*, **34**, 230-236.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
solar <- grep('solrad', names(env), value=TRUE)
maxtemp <- grep('tmax', names(env), value=TRUE)
mintemp <- grep('tmin', names(env), value=TRUE)

# read them in as rasterStacks
meantemp <- stack(env[[meantemp]])
solar <- stack(env[[solar]])
maxtemp <- stack(env[[maxtemp]])
mintemp <- stack(env[[mintemp]])
tempRange <- abs(maxtemp - mintemp)

# get monthly PET
pet <- monthlyPET(meantemp, solar, tempRange)

# get mean annual PET
annualPET <- sum(pet)
```

```
climaticMoistureIndex(env[['bio_12']], annualPET)
```

continentality *Continentality*

Description

Generate Continentality index.

Usage

```
continentality(tmax, tmin, tempScale = 1)
```

Arguments

tmax	rasterLayer of average temperature of the warmest month
tmin	rasterLayer of average temperature of the coldest month
tempScale	integer; scaling factor for the temperature data, see envirem for additional details.

Details

```
continentality index = tmax - tmin
```

Value

rasterLayer in units of degrees C.

Author(s)

Pascal Title

References

Rivas-Martínez, S. & Rivas-Sáenz, S. "Synoptical Worldwide Bioclimatic Classification System". Available online at http://www.globalbioclimatics.org/ [accessed 15 February 2016]

Sayre, R., Comer, P., Warner, H. & Cress, J. (2009) *A new map of standardized terrestrial ecosystems of the conterminous United States: US Geological Survey Professional Paper 1768*. Reston, VA.

See Also

thermicityIndex

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify appropriate layers
tmean <- grep('tmean', names(env))
tmin <- grep('tmin', names(env))
tmax <- grep('tmax', names(env))

tmean <- env[[tmean]]
tmin <- env[[tmin]]
tmax <- env[[tmax]]

# calculate temperature extremes
temp <- otherTempExtremes(tmean, tmin, tmax)

meantempWarmest <- temp[['meanTempWarmest']]
meantempColdest <- temp[['meanTempColdest']]

continentality(meantempWarmest, meantempColdest, tempScale = 10)
```

`dataTypeCheck` *Data Type Check*

Description

Determines the best data type to implement when writing the raster to file

Usage

```
dataTypeCheck(r)
```

Arguments

r	raster object

Details

Function to determine the most memory efficient data type given whether or not the raster contains integer or non-integer values, and the range of those values, based on the definitions described in `dataType`.

Author(s)

Pascal Title

Examples

```
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
r <- raster(rasterFiles[1])
dataTypeCheck(r)
```

---

embergerQ *Emberger's pluviometric quotient*

Description

Calculate Emberger's pluviometric quotient.

Usage

```
embergerQ(P, M, m, tempScale = 1)
```

Arguments

`P`	rasterLayer, mean annual precipitation
`M`	rasterLayer, mean max temperature of the warmest month
`m`	rasterLayer, mean min temperature of the coldest month
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Details

```
Q = 2000 P / [(M + m + 546.4) * (M - m)]
```

Value

rasterLayer in mm / degrees C

Author(s)

Pascal Title

References

Daget, P. (1977) Le bioclimat méditerranéen: analyse des formes climatiques par le système d'Emberger. *Vegetatio*, **34**, 87–103.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

embergerQ(env[['bio_12']], env[['bio_5']], env[['bio_6']], tempScale = 10)
```

envirem *envirem*

Description

Generation of bioclimatic rasters that are complementary to the WorldClim dataset.

Details

Package: envirem
Type: Package
Version: 1.2
Date: 2018-03-06
License: GPL-2 | GPL-3

NOTE: Temperature rasters are now assumed by default to be in degrees C. However, different input datasets have different units. For example, worldclim v1 has temperature rasters in degrees C * 10. Worldclim v2 uses degrees C. CHELSA has several options, depending on whether rasters are downloaded as floating point or integer. Therefore, there is an argument `tempScale` to specify the units of temperature:

For example:
If using worldclim v1 data where temperature is in degrees C * 10, specify `tempScale = 10`.
If using worldclim v2 where temperature is in degrees C, specify `tempScale = 1`.
If using CHELSA with floating point accuracy where temperature is in degrees C, specify `tempScale = 1`.
If using CHELSA as integers where temperature is in degrees C * 1000, specify `tempScale = 1000`.

If a function does not have the `tempScale` argument, then the function is not sensitive to the units of the input temperature rasters.

Precipitation is in mm across both WorldClim and CHELSA, therefore no equivalent control is provided.

Of course, it is also perfectly acceptable to leave `tempScale = 1` and modify the input rasters yourself.

The main function for generating ENVIREM rasters is `generateRasters`. A complete tutorial of this R package can be found at `http://envirem.github.io`.

Author(s)

Pascal O. Title, Jordan B. Bemmels

References

`http://envirem.github.io`

Title, P.O., Bemmels, J.B. 2017. ENVIREM: An expanded set of bioclimatic and topographic variables increases flexibility and improves performance of ecological niche modeling. Ecography doi: 10.1111/ecog.02880.

generateRasters *Execute Layer Creation*

Description

Main function to generate specified ENVIREM layers. If requested, this function will split input rasters into tiles, generate desired variables, and reassemble the results. Results are named according to specified resName and timeName. For the distinction between this function and `layerCreation`, see `Details`.

Usage

```
generateRasters(var, maindir, resName, timeName, outputDir,
  rasterExt = ".tif", nTiles = 1, tempScale = 1,
  overwriteResults = TRUE, outputFormat = "GTiff", tempDir = "~/temp",
  gdalinfoPath = NULL, gdal_translatePath = NULL)
```

Arguments

`var`	a vector of variable names to generate, see Details.
`maindir`	path to directory of input rasters
`resName`	output nametag for the resolution
`timeName`	output nametag for the time period
`outputDir`	output directory. A directory will be generated according to the resName and timeName, so this is the output location for the directory that will be generated.
`rasterExt`	the file extension of the input rasters
`nTiles`	the number of tiles to split the rasters when tiling is requested, must be a perfect square
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.
`overwriteResults`	logical, should existing rasters be overwritten
`outputFormat`	output format for rasters, see `writeRaster` for options
`tempDir`	temporary directory that will be created and then removed
`gdalinfoPath`	path to gdalinfo binary, leave as `NULL` if it is in the default search path.
`gdal_translatePath`	path to gdal_translate binary, leave as `NULL` if it is in the default search path.

Details

The function `layerCreation` will generate envirem rasters from input R objects (rasterStacks) and will return the result as an R object. In contrast, the function `generateRasters` reads in input rasters from a specified directory, splits input rasters into tiles if necessary, internally calls `layerCreation` and writes the result to file.

Possible variables to generate include:

annualPET
aridityIndexThornthwaite
climaticMoistureIndex
continentality
embergerQ
growingDegDays0
growingDegDays5
maxTempColdest
minTempWarmest
monthCountByTemp10
PETColdestQuarter
PETDriestQuarter
PETseasonality
PETWarmestQuarter
PETWettestQuarter
thermicityIndex

If `var = 'all'`, then all of the variables will be generated.

`resName` and `timeName` are only used for naming the output directory.

Rasters in `mainDir` should be named appropriately (see `verifyFileStructure`) and with identical resolution, origin and extent.

Output rasters are written with the most appropriate `dataType`, as inferred with `dataTypeCheck`. This will reduce the file size of these rasters.

If the goal is to use these rasters with the standalone Maxent program, we recommend `outputFormat = 'EHdr'`.

Value

The requested set of rasterLayers will be written to `outputDir`.

Author(s)

Pascal Title

See Also

Naming of rasters in inputDir will be checked with `verifyFileStructure`.

`growingDegDays` *Growing degree days*

Description

Growing degree days above some base temperature.

Usage

```
growingDegDays(meantempstack, baseTemp, tempScale = 1)
```

Arguments

`meantempstack`	rasterStack of mean monthly temperature in deg C * 10
`baseTemp`	base temperature in degrees C.
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Details

growing degree days = sum of all monthly temps greater than baseTemp, multiplied by total number of days

Value

rasterLayer in degrees C * days.

Author(s)

Pascal Title

References

Metzger, M.J., Bunce, R.G.H., Jongman, R.H.G., Sayre, R., Trabucco, A. & Zomer, R. (2013). A high-resolution bioclimate map of the world: a unifying framework for global biodiversity research and monitoring. *Global Ecology and Biogeography*, **22**, 630-638.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

meantemp <- env[[grep('tmean', names(env), value=TRUE)]]
growingDegDays(meantemp, 10, tempScale = 10)
```

`layerCreation` *Creates all layers*

Description

Generates all rasterLayers for one particular input dataset. For the distinction between this function and `generateRasters`, see `Details`.

Usage

```
layerCreation(masterstack, solradstack, var, tempScale = 1)
```

Arguments

`masterstack`	rasterStack containing all precipitation, min temperature, max temperature and bioclimatic variables
`solradstack`	rasterStack of monthly solar radiation
`var`	vector of names of variables to generate, see Details.
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Details

The function `verifyFileStructure` should be used to verify that the appropriate rasters are present in `masterstack`.

This function is called internally by `generateRasters`.

The function `layerCreation` will generate envirem rasters from input R objects (rasterStacks) and will return the result as an R object. In contrast, the function `generateRasters` reads in input rasters from a specified directory, splits input rasters into tiles if necessary, internally calls `layerCreation` and writes the result to file.

Possible variables to generate include:

annualPET
aridityIndexThornthwaite
climaticMoistureIndex
continentality
embergerQ
growingDegDays0
growingDegDays5
maxTempColdest
minTempWarmest
monthCountByTemp10
PETColdestQuarter
PETDriestQuarter
PETseasonality
PETWarmestQuarter
PETWettestQuarter
thermicityIndex

If `var = 'all'`, then all of the variables will be generated.

Value

rasterStack

Author(s)

Pascal Title

See Also

This function is called internally by generateRasters.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)

# create stack of temperature and precipitation rasters
# and stack of solar radiation rasters
solradFiles <- grep('solrad', rasterFiles, value=TRUE)
worldclim <- stack(setdiff(rasterFiles, solradFiles))
solar <- stack(solradFiles)

# generate all possible envirem variables
layerCreation(worldclim, solar, var='all', tempScale = 10)
```

`monthCountByTemp` *Month count by temperature*

Description

Number of months with mean temperature greater than some base temp.

Usage

```
monthCountByTemp(tempStack, minTemp = 10, tempScale = 1)
```

Arguments

`tempStack`	rasterStack of monthly mean temperature in degrees C
`minTemp`	reference temperature in degrees C
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Value

rasterLayer with values representing counts of months.

Author(s)

Pascal Title

References

Metzger, M.J., Bunce, R.G.H., Jongman, R.H.G., Sayre, R., Trabucco, A. & Zomer, R. (2013). A high-resolution bioclimate map of the world: a unifying framework for global biodiversity research and monitoring. *Global Ecology and Biogeography*, **22**, 630-638.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
meantemp <- env[[meantemp]]
monthCountByTemp(meantemp, 10, tempScale = 10)
```

`monthlyPET` *monthly PET*

Description

Monthly potential evapotranspiration

Usage

```
monthlyPET(Tmean, RA, TD, tempScale = 1)
```

Arguments

`Tmean`	rasterStack of monthly mean temperature
`RA`	rasterStack of monthly extraterrestrial solar radiation
`TD`	rasterStack of monthly temperature range
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Details

```
PET = 0.0023 * RA * (Tmean + 17.8) * TD ^ 0.5
```

Value

rasterStack of monthly PET in mm / month

Author(s)

Pascal Title

References

Hargreaves, G. L., Hargreaves, G. H., & Riley, J. P. (1985). Irrigation water requirements for Senegal River basin. *Journal of Irrigation and Drainage Engineering*, **111**, 265-275.

Zomer, R.J., Trabucco, A., Bossio, D.A. & Verchot, L.V. (2008). Climate change mitigation: A spatial analysis of global land suitability for clean development mechanism afforestation and reforestation. *Agriculture, Ecosystems and Environment*, **126**, 67-80.

Zomer, R.J., Trabucco, A., Van Straaten, O. & Bossio, D.A. (2006) *Carbon, Land and Water: A Global Analysis of the Hydrologic Dimensions of Climate Change Mitigation through Afforestation/Reforestation. International Water Management Institute Research Report 101*. Colombo, Sri Lanka.

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
solar <- grep('solrad', names(env), value=TRUE)
maxtemp <- grep('tmax', names(env), value=TRUE)
mintemp <- grep('tmin', names(env), value=TRUE)

# read them in as rasterStacks
meantemp <- stack(env[[meantemp]])
solar <- stack(env[[solar]])
maxtemp <- stack(env[[maxtemp]])
mintemp <- stack(env[[mintemp]])
tempRange <- abs(maxtemp - mintemp)

monthlyPET(meantemp, solar, tempRange, tempScale = 10)
```

`otherTempExtremes` *Temperature Extremes*

Description

Generates max temp of the coldest month, min temp of the warmest month, mean temp of the coldest month, mean temp of the warmest month.

Usage

```
otherTempExtremes(meantempStack, mintempStack, maxtempStack)
```

Arguments

`meantempStack`	rasterStack of monthly mean temperature
`mintempStack`	rasterStack of monthly min temperature
`maxtempStack`	rasterStack of monthly max temperature

Value

rasterStack of maxTempColdest, minTempWarmest, meanTempColdest, meanTempWarmest, in same units as input rasters.

Author(s)

Pascal Title

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify appropriate layers
tmean <- grep('tmean', names(env))
tmin <- grep('tmin', names(env))
tmax <- grep('tmax', names(env))

tmean <- env[[tmean]]
tmin <- env[[tmin]]
tmax <- env[[tmax]]

# calculate temperature extremes
otherTempExtremes(tmean, tmin, tmax)
```

`pacificCentric` *Center raster on the Pacific*

Description

Takes a raster that is centered on 0 longitude (default) and recenters it on the Pacific

Usage

```
pacificCentric(r, crop = TRUE)
```

Arguments

`r`	rasterLayer or rasterStack in unprojected geographic coordinates
`crop`	logical, should raster then be cropped to longitude [100, 300]

Details

Cropping to [100, 300] is equivalent to [100, -60]

Value

rasterLayer or rasterStack

Author(s)

Pascal Title

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
bio1 <- raster(grep('bio_1\\.', rasterFiles, value=TRUE))

pacificCentric(bio1, crop = TRUE)
```

`petExtremes` *PET Extremes*

Description

Calculates mean PET of the coldest, warmest, wettest and driest quarters.

Usage

```
petExtremes(PETstack, precipStack, meantempStack)
```

Arguments

`PETstack`	rasterStack of monthly PET
`precipStack`	rasterStack of monthly precipitation
`meantempStack`	rasterStack of monthly mean temperature

Details

Generates mean monthly PET for the warmest, coldest, wettest and driest 3 consecutive months.

Value

rasterStack of PETColdestQuarter, PETWarmestQuarter, PETWettestQuarter, PETDriestQuarter in mm / month.

Author(s)

Pascal Title

References

Metzger, M.J., Bunce, R.G.H., Jongman, R.H.G., Sayre, R., Trabucco, A. & Zomer, R. (2013). A high-resolution bioclimate map of the world: a unifying framework for global biodiversity research and monitoring. *Global Ecology and Biogeography*, **22**, 630-638.

See Also

monthlyPET

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
solar <- grep('solrad', names(env), value=TRUE)
maxtemp <- grep('tmax', names(env), value=TRUE)
mintemp <- grep('tmin', names(env), value=TRUE)
precip <- grep('prec', names(env), value=TRUE)

# read them in as rasterStacks
meantemp <- stack(env[[meantemp]])
solar <- stack(env[[solar]])
maxtemp <- stack(env[[maxtemp]])
mintemp <- stack(env[[mintemp]])
tempRange <- abs(maxtemp - mintemp)
precip <- stack(env[[precip]])

# get monthly PET
pet <- monthlyPET(meantemp, solar, tempRange)

petExtremes(pet, precip, meantemp)
```

PETseasonality *PET seasonality*

Description

Seasonality of potential evapotranspiration

Usage

```
PETseasonality(PETstack)
```

Arguments

`PETstack` rasterStack of monthly PET rasters

Details

PET seasonality = 100 * standard deviation of monthly PET.

Value

rasterLayer in mm / month

Author(s)

Pascal Title

References

Metzger, M.J., Bunce, R.G.H., Jongman, R.H.G., Sayre, R., Trabucco, A. & Zomer, R. (2013). A high-resolution bioclimate map of the world: a unifying framework for global biodiversity research and monitoring. *Global Ecology and Biogeography*, **22**, 630-638.

See Also

monthlyPET

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify the appropriate layers
meantemp <- grep('mean', names(env), value=TRUE)
solar <- grep('solrad', names(env), value=TRUE)
maxtemp <- grep('tmax', names(env), value=TRUE)
mintemp <- grep('tmin', names(env), value=TRUE)

# read them in as rasterStacks
meantemp <- stack(env[[meantemp]])
solar <- stack(env[[solar]])
maxtemp <- stack(env[[maxtemp]])
mintemp <- stack(env[[mintemp]])
tempRange <- abs(maxtemp - mintemp)

# get monthly PET
pet <- monthlyPET(meantemp, solar, tempRange)

PETseasonality(pet)
```

split_raster — *Split raster into tiles*

Description

Splits a rasterLayer into tiles

Usage

```
split_raster(file, s = 2, outputDir, gdalinfoPath = NULL,
  gdal_translatePath = NULL)
```

Arguments

`file`	full path to a raster
`s`	division applied to each side of the raster (s=2 -> 4 tiles)
`outputDir`	path and directory name for output
`gdalinfoPath`	path to gdalinfo binary. Set to `NULL` if default search path is sufficient.
`gdal_translatePath`	path to gdal_translate binary. Set to `NULL` if default search path is sufficient.

Details

GDAL must be installed for this function to work. To determine if the default search paths are sufficient, you can type in R `Sys.which('gdalinfo')` and `Sys.which('gdal_translate')`. If a path is returned, then you can leave those arguments as `NULL`.

Value

Rasters are written to the output directory.

Author(s)

Pascal Title

References

GDAL. 2015. GDAL - Geospatial Data Abstraction Library: Version 1.11.3, Open Source Geospatial Foundation, http://gdal.org

Examples

```
## Not run:
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
bio1file <- grep('bio_1\\.', rasterFiles, value=TRUE)

We will split this raster into 4 tiles, that will be written to disk.
split_raster(bio1file, s = 2, outputDir = '~/temp/', gdalinfoPath = NULL, gdal_translatePath = NULL)

## End(Not run)
```

`thermicityIndex` *Compensated Thermicity index*

Description

Compensated Thermicity index

Usage

```
thermicityIndex(annualTemp, minTemp, maxTemp, continentality,
  returnCompensated = TRUE, tempScale = 1)
```

Arguments

`annualTemp`	rasterLayer, mean annual temperature
`minTemp`	rasterLayer, min temp of the coldest month
`maxTemp`	rasterLayer, max temp of the coldest month
`continentality`	rasterLayer, continentality index
`returnCompensated`	logical: if `FALSE`, regular thermicity index is returned.
`tempScale`	integer; scaling factor for the temperature data, see envirem for additional details.

Details

thermicity index = tempRange + minTemp + maxTemp

The compensated thermicity index incorporates corrections designed to make this metric more appropriately comparable across the globe.

Value

rasterLayer in degrees C

Author(s)

Pascal Title

References

Rivas-Martínez, S. & Rivas-Sáenz, S. "Synoptical Worldwide Bioclimatic Classification System". Available online at http://www.globalbioclimatics.org/ [accessed 15 February 2016]

Sayre, R., Comer, P., Warner, H. & Cress, J. (2009) *A new map of standardized terrestrial ecosystems of the conterminous United States: US Geological Survey Professional Paper 1768*. Reston, VA.

See Also

continentality

Examples

```
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
env <- stack(rasterFiles)

# identify appropriate layers
tmean <- grep('tmean', names(env))
tmin <- grep('tmin', names(env))
tmax <- grep('tmax', names(env))

tmean <- env[[tmean]]
tmin <- env[[tmin]]
tmax <- env[[tmax]]

# calculate temperature extremes
temp <- otherTempExtremes(tmean, tmin, tmax)

ci <- continentality(temp[['meanTempWarmest']], temp[['meanTempColdest']], tempScale = 10)

# compensated thermicity index
thermicityIndex(env[['bio_1']], env[['bio_6']], temp[['maxTempColdest']], ci, tempScale = 10)
```

`topoWetnessIndex` *Topographic Wetness Index*

Description

SAGA-GIS topographic wetness index

Usage

```
topoWetnessIndex(dem, sagaEnv)
```

Arguments

`dem`	elevation rasterLayer, with defined proj4string.
`sagaEnv`	list object returned from `RSAGA::rsaga.env`, which supplies appropriate SAGA paths, and parallelization information.

Details

If this function returns an error, there may be a conflict with the version of SAGA-GIS installed on your machine, and the version of SAGA-GIS that the RSAGA package is designed to work with.

From a DEM, this function will write an appropriate raster to disk, run an RSAGA function to calculate the topographic wetness index, and will then read it back in and return it.

This function requires that SAGA-GIS be installed on your system. SAGA-GIS can be found at `www.saga-gis.org`.

See the documentation for `RSAGA::rsaga.env` for specifying appropriate paths and parallelization details.

Value

rasterLayer, unitless

Author(s)

Pascal Title

References

Boehner, J., Koethe, R. Conrad, O., Gross, J., Ringeler, A. & Selige, T. (2002) Soil regionalization by means of terrain analysis and process parameterization. *Soil Classification 2001 European Soil Bureau, Research Report No. 7* (eds Micheli, E., Nachtergaele, F. & Montanarella, L.), pp. 213-222. Luxembourg.

Conrad, O., Bechtel, B., Bock, M., Dietrich, H., Fischer, E., Gerlitz, L., Wehberg, J., Wichmann, V. & Boehner, J. (2015) System for automated geoscientific analyses (SAGA) v. 2.1.4. *Geoscientific Model Development*, **8**, 1991-2007.

Examples

```
## Not run:
# Find example rasters
rasterFiles <- list.files(system.file('extdata', package='envirem'), full.names=TRUE)
elev <- raster(grep('elev', rasterFiles, value=TRUE))

# setting up appropriate RSAGA environment
sagaEnv <- RSAGA::rsaga.env(modules = '/usr/lib/x86_64-linux-gnu/saga/', cores = 2,
parallel = TRUE, version = "2.2.0")
topoWetnessIndex(elev, sagaEnv)

## End(Not run)
```

verifyFileStructure *Verify File Structure*

Description

Ensures that the necessary files are present for other functions to work properly.

Usage

```
verifyFileStructure(path, returnFileNames = TRUE, rasterExt = ".tif")
```

Arguments

`path`	path to directory of rasters
`returnFileNames`	logical, should file paths and names be returned
`rasterExt`	file extension of rasters

Details

This function searches for the following in the directory specified by `path`:

19 bioclimatic variables named as bio_1.tif

12 precipitation rasters named as prec_1.tif

12 min temperature rasters named as tmin_1.tif

12 max temperature rasters named as tmax_1.tif

12 mean temperature rasters named as tmean_1.tif [optional]

12 solar radiation rasters named as et_solrad_1.tif

If mean temperature rasters are not detected, the raster creation functions will create mean temperature by taking the mean of the min and max.

Value

Prints messages to the console if problems are found. If `returnFileNames == TRUE`, then a vector of filenames is returned.

Author(s)

Pascal Title

Examples

```
# As there are no problems with these files, the list of files
# will be returned.
verifyFileStructure(system.file('extdata', package='envirem'))
```

Index