Package 'rgeomorphon'

September 16, 2025

Type Package

Title A Lightweight Implementation of the 'Geomorphon' Algorithm

Version 0.3.0

Maintainer Andrew Brown <brown.andrewg@gmail.com>

Description A lightweight implementation of the geomorphon terrain form classification algorithm of Jasiewicz and Stepinski (2013) <doi:10.1016/j.geomorph.2012.11.005> based largely on the 'GRASS GIS' 'r.geomorphon' module. This implementation employs a novel algorithm written in C++ and 'RcppParallel'.

License GPL (>= 3)

Depends R (>= 3.6.2)

Imports Rcpp, RcppParallel

LinkingTo Rcpp, RcppArmadillo, RcppParallel

Suggests terra, future.apply, litedown, tinytest

Enhances future, parallel

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

URL https://github.com/brownag/rgeomorphon/, https://humus.rocks/rgeomorphon/

BugReports https://github.com/brownag/rgeomorphon/issues

LazyData true

VignetteBuilder litedown

NeedsCompilation yes

Author Andrew Brown [aut, cre] (ORCID: <https://orcid.org/0000-0002-4565-533X>)

Repository CRAN

Date/Publication 2025-09-16 07:20:02 UTC

Contents

`forms_matrix` *Create a* `forms_matrix` *object*

Description

This constructor function wraps a 9x9 integer matrix and associates it with a set of levels, creating a 'forms_matrix' object.

Usage

```
forms_matrix(x, levels = get_forms_grass_enum())
```

Arguments

`x`	Integer. A 9x9 matrix.
`levels`	Named integer vector. Map of integer values to their string names. Default: `get_forms_grass_enum()`

Details

This function is intended for custom classification matrix based on positive and negative overlooks. See `forms_matrix_get()` for a convenient accessor for the standard classification systems with 4, 5, 6 or 10 forms.

Value

An object of class `c("forms_matrix", "matrix", "array")`.

Examples

```
library(terra)
library(rgeomorphon)

# default values
x <- forms_matrix_get(num_forms = 10, levels = get_forms_grass_enum())
```

```
# inspect
x

# create a 9-class system where PEAK is combined with RIDGE
x[x == 2] <- 3
a <- get_forms_grass_enum()
a <- a[!names(a) == "G_PK"]

# create a forms matrix with custom levels
fm <- forms_matrix(x, a)

# run geomorphon algorithm
SEARCH = 7       # outer search radius (cells)
SKIP = 1         # inner skip radius (cells)
DIST = 0         # flatness distance (cells)
FLAT = 1         # flat angle threshold
MODE = "anglev1" # comparison mode

## classic volcano
data("volcano", package = "datasets")
dem <- terra::rast(volcano)
terra::crs(dem) <- terra::crs("EPSG:2193")
terra::ext(dem) <- c(1756968, 1757578, 5917000, 5917870)
names(dem) <- "elevation"

# include original forms, positive, and negative output
res <- geomorphons(
    dem,
    search = SEARCH,
    skip = SKIP,
    dist = DIST,
    flat = FLAT,
    comparison_mode = MODE,
    forms = TRUE,
    positive = TRUE,
    negative = TRUE
)

 # apply custom classification to positive and negative
 res2 <- geomorphon_theme(
   forms_matrix_apply(
       x = res[[c("positive", "negative")]],
       rcl = fm
   )
 )

 # compare with default
 terra::plot(terra::rast(c(`10 form`=res$forms, `9 form`=res2)))
```

forms_matrix_apply *Apply a* forms_matrix *to Positive and Negative Overlooks*

Description

This function applies a forms_matrix to reclassify a SpatRaster object with 2 layers containing positive and negative overlooks.

Usage

```
forms_matrix_apply(
  x,
  rcl = forms_matrix_get(),
  positive = "positive",
  negative = "negative",
  ...
)
```

Arguments

x	SpatRaster containing two layers with names specified in positive and negative.
rcl	forms_matrix. Matrix to use for classification of x. Rows are "negative" and columns are "positive".
positive	Character. Layer name of positive count. Default: "positive".
negative	Character. Layer name of negative count. Default: "negative".
...	Additional arguments passed to terra::classify().

Value

A SpatRaster containing the classification result.

See Also

forms_matrix()

Examples

```
library(terra)
library(rgeomorphon)

SEARCH = 7       # outer search radius (cells)
SKIP = 1         # inner skip radius (cells)
DIST = 0         # flatness distance (cells)
FLAT = 1         # flat angle threshold
MODE = "anglev1" # comparison mode

## classic volcano
```

```
data("volcano", package = "datasets")
dem <- terra::rast(volcano)
terra::crs(dem) <- terra::crs("EPSG:2193")
terra::ext(dem) <- c(1756968, 1757578, 5917000, 5917870)
names(dem) <- "elevation"

res <- geomorphons(
    dem,
    search = SEARCH,
    skip = SKIP,
    dist = DIST,
    flat = FLAT,
    comparison_mode = MODE,
    forms = TRUE,
    ternary = TRUE,
    positive = TRUE,
    negative = TRUE
)

res2 <- terra::rast(lapply(c(4, 5, 6), function(n) {
  geomorphon_theme(
    forms_matrix_apply(
        x = res[[c("positive", "negative")]],
        rcl = forms_matrix_get(n)
    )
  )
}))
names(res2) <- c("forms4", "forms5", "forms6")

terra::plot(c(res, res2))
```

`forms_matrix_get` *Get a* `forms_matrix` *for Geomorphon Classification*

Description

Gets one of the internally defined forms matrices. A form matrix is defined for the classic 10-form output (default; Jasiewicz & Stepinski, 2013) as well as three simplified classes: 4-form, 5-form, and 6-form (Masetti et al., 2018)

Usage

```
forms_matrix_get(num_forms = 10, levels = get_forms_grass_enum())
```

Arguments

`num_forms`	Integer. The number of forms to classify, one of `4`, `5`, `6`, or `10` (default).
`levels`	Named integer with values between 0 and 10 corresponding to form class labels. Default: `get_forms_grass_enum()`

Details

For creating custom classification systems see the `forms_matrix()` constructor.

Value

An object of class `forms_matrix`

References

Stepinski, T., Jasiewicz, J., 2011, Geomorphons - a new approach to classification of landform, in : Eds: Hengl, T., Evans, I.S., Wilson, J.P., and Gould, M., Proceedings of Geomorphometry 2011, Redlands, 109-112. Available online: https://www.geomorphometry.org/uploads/pdf/pdf2011/StepinskiJasiewicz2011geomorphometry.pdf

Jasiewicz, J., Stepinski, T., 2013, Geomorphons - a pattern recognition approach to classification and mapping of landforms, Geomorphology, vol. 182, 147-156. (doi:10.1016/j.geomorph.2012.11.005)

Masetti, G., Mayer, L. A., & Ward, L. G. 2018, A Bathymetry- and Reflectivity-Based Approach for Seafloor Segmentation. Geosciences, 8(1), 14. (doi:10.3390/geosciences8010014)

See Also

`forms_matrix()`

Examples

```
forms_matrix_get()
```

`geomorphons` *Calculate Geomorphons*

Description

'Rcpp' implementation of the 'geomorphon' terrain classification system based on 'r.geomorphon' algorithm of Jasiewicz and Stepinski (2013) from 'GRASS GIS'.

Usage

```
geomorphons(
  elevation,
  filename = NULL,
  search = 3,
  skip = 0,
  flat_angle_deg = 1,
  dist = 0,
  comparison_mode = "anglev1",
  tdist = 0,
  forms = TRUE,
```

```
  ternary = FALSE,
  positive = FALSE,
  negative = FALSE,
  use_meters = FALSE,
  nodata_val = NA_integer_,
  xres = NULL,
  yres = xres,
  simplify = FALSE,
  LAPPLY.FUN = lapply,
  nchunk = geomorphon_chunks_needed(elevation)
)
```

Arguments

elevation	matrix or SpatRaster object. Digital Elevation Model values. It is **STRONGLY** recommended to use a grid in a projected coordinate system.
filename	character. Output filename. Default NULL creates a temporary file.
search	numeric. User input for search radius (default: 3). Units depend on use_meters.
skip	numeric. User input for skip radius (default: 0). Units depend on use_meters.
flat_angle_deg	numeric. Flatness angle threshold in **degrees**. Default: 1.0.
dist	numeric. Flatness distance (default: 0). Units depend on use_meters.
comparison_mode	Character. One of "anglev1", "anglev2", "anglev2_distance". Default: "anglev1".
tdist	numeric. Terrain distance factor. When greater than 0, overrides Z tolerance from angular logic. Default: 0.0.
forms	character. Number of geomorphon forms to identify. One of "forms10 (default), "forms6", "forms5", or "forms4.
ternary	logical. Include "ternary" output? Default: FALSE
positive	logical. Include "positive" output? Default: FALSE
negative	logical. Include "negative" output? Default: FALSE
use_meters	Logical. Default: FALSE uses cell units. Set to TRUE to specify search, skip, and dist in units of meters.
nodata_val	numeric. NODATA value. Default: NA_integer_.
xres	numeric. X grid resolution (used only when elevation is a matrix). Default: NULL.
yres	numeric. Y grid resolution (used only when elevation is a matrix). Default: xres.
simplify	logical. If result is length 1 list, the first element is returned. Default: FALSE
LAPPLY.FUN	An lapply()-like function such as future.apply::future_lapply(). Default: lapply().
nchunk	Number of tile chunks to use. Default: geomorphon_chunks_needed(elevation).

Value

List of SpatRaster or matrix of geomorphon algorithm outputs. When more than one of `forms`, `ternary`, `positive`, `negative` are set the result is a list. For one result type, and default `simplify` argument, the result is the first (and only) element of the list.

Distance Calculation and Coordinate Reference Systems

The algorithm assumes planar distances and angles are calculated based on cell resolutions, so it is strongly recommended that elevation data be in a projected coordinate system.

Buffer Around Area of Interest

For reliable geomorphon classification, especially near study area boundaries, it is recommended to use a raster that includes a buffer of at least `search + 1` cells around the area of interest. This implementation utilizes all available DEM data up to the specified search radius.

A buffer of `search + skip + 1` cells is automatically applied when processing SpatRaster input, as this is necessary to avoid edge effects when processing large rasters in tiles. Matrix input is not altered.

Tiled Processing for Large Rasters

For Digital Elevation Models (DEMs) that are too large to fit into available memory, `rgeomorphon` employs an automatic tiled processing workflow. This method breaks the large raster into a grid of smaller, manageable chunks that are processed sequentially.

The premise of this approach is the use of buffered tiles. To ensure seamless results and avoid edge artifacts, a buffer of surrounding data is added to each chunk before the geomorphon calculation is performed. This provides the necessary neighborhood of cells for the algorithm to work correctly. After each tile is processed, the buffer region is removed from the result. Finally, the clean, processed tiles are mosaicked back together into a single, complete output raster that perfectly matches the extent of the original input DEM.

This entire workflow is handled internally by the main `geomorphons()` function, which can also leverage parallel processing to speed up the operation on multi-core systems. See the vignette on parallel processing with 'future' package.

The number of chunks needed can be controlled by setting several environment variables. These variables are read by the function at runtime.

Default Behavior:

By default, the function assumes a single worker, scales the estimated memory needed by a factor of 10, and applies the square root to the total number of chunks. This can be replicated with the following settings:

```
Sys.setenv(R_RGEOMORPHON_N_WORKERS = 1)
Sys.setenv(R_RGEOMORPHON_MEM_SCALE_NEED = 10)
Sys.setenv(R_RGEOMORPHON_MEM_SCALE_WORKERS = 1)
Sys.setenv(R_RGEOMORPHON_MEM_POWER = 0.5)
```

Customized Behavior:

You can customize the tiling behavior by setting the environment variables to different values. For example, to use four workers, scale memory needs by a factor of five, apply a worker scaling factor of two, and a power of 1.5 to the total, you would set the following:

```
Sys.setenv(R_RGEOMORPHON_N_WORKERS = 4)
Sys.setenv(R_RGEOMORPHON_MEM_SCALE_NEED = 5)
Sys.setenv(R_RGEOMORPHON_MEM_SCALE_WORKERS = 2)
Sys.setenv(R_RGEOMORPHON_MEM_POWER = 1.5)
```

Comparison with GRASS 'r.geomorphon'

This implementation achieves very high agreement with the classification logic of GRASS GIS 'r.geomorphon' when using equivalent parameters and data in a projected coordinate system.

'r.geomorphon' employs a row buffering strategy which can, for cells near the edges of a raster, result in a truncated line-of-sight compared to the full raster extent. This may lead GRASS to classify edge-region cells differently or as NODATA where this implementation may produce a more 'valid' geomorphon form given the available data.

More information about the 'r.geomorphon' module can be found in the GRASS GIS manual: https://grass.osgeo.org/grass-stable/manuals/r.geomorphon.html

References

Stepinski, T., Jasiewicz, J., 2011, Geomorphons - a new approach to classification of landform, in : Eds: Hengl, T., Evans, I.S., Wilson, J.P., and Gould, M., Proceedings of Geomorphometry 2011, Redlands, 109-112. Available online: https://www.geomorphometry.org/uploads/pdf/pdf2011/StepinskiJasiewicz2011geomorphometry.pdf

Jasiewicz, J., Stepinski, T., 2013, Geomorphons - a pattern recognition approach to classification and mapping of landforms, Geomorphology, vol. 182, 147-156. (doi:10.1016/j.geomorph.2012.11.005)

See Also

geomorphon_theme() geomorphon_chunks_needed()

Examples

```
library(terra)
library(rgeomorphon)

SEARCH = 7       # outer search radius (cells)
SKIP = 1         # inner skip radius (cells)
DIST = 0         # flatness distance (cells)
FLAT = 1         # flat angle threshold
MODE = "anglev1" # comparison mode

## classic volcano
data("volcano", package = "datasets")
dem <- terra::rast(volcano)
terra::crs(dem) <- terra::crs("EPSG:2193")
terra::ext(dem) <- c(1756968, 1757578, 5917000, 5917870)
names(dem) <- "elevation"
```

```
system.time({
    rg <- geomorphons(
        dem,
        search = SEARCH,
        skip = SKIP,
        dist = DIST,
        flat = FLAT,
        comparison_mode = MODE
    )
})

plot(c(dem, rg))
```

`geomorphon_categories` *Apply Geomorphon Theme to Result Object*

Description

Applies standard class names and colors to a SpatRaster, or creates a factor matrix. Input values should be integers between 1 and 10.

Usage

```
geomorphon_categories()

geomorphon_colors()

geomorphon_theme(x, forms = "forms10")
```

Arguments

`x`	A SpatRaster or matrix object.
`forms`	character. One of: `"forms10"` (default), `"forms6"`, `"forms5"`, `"forms4"`. These are themes corresponding to the built-in 10-form, 6-form, 5-form, and 4-form `"forms"` outputs from `geomorphons()`.

Details

When `x` is a matrix the result is a factor using `geomorphon_categories()`. Values are integers 1 to 10 and labels are the geomorphon form names.

Value

A SpatRaster or matrix object with geomorphon class names (and colors for SpatRaster) applied.

Examples

```
geomorphon_theme(1:10)
```

`geomorphon_chunks_needed` *Estimate Tile Processing Needs*

Description

`geomorphon_chunks_needed()` is a heuristic for number of tiles needed to calculate geomorphons on larger-than-memory rasters. Allows for scaling by number of parallel workers, a multiplicative factor for the memory needs, and a multiplicative factor for worker needs.

Usage

```
geomorphon_chunks_needed(
  x,
  workers = Sys.getenv("R_RGEOMORPHON_N_WORKERS", unset = 1),
  scl_need = Sys.getenv("R_RGEOMORPHON_MEM_SCALE_NEED", unset = 10),
  scl_workers = Sys.getenv("R_RGEOMORPHON_MEM_SCALE_WORKERS", unset = 1),
  pow_total = Sys.getenv("R_RGEOMORPHON_MEM_POWER", unset = 0.5)
)
```

Arguments

`x`	A *SpatRaster* object.
`workers`	*integer*. Number of parallel workers. Default uses value of environment variable `R_RGEOMORPHON_N_WORKERS`. If unset, `1`
`scl_need`	*numeric*. Scaling factor for memory needs. Default uses value of environment variable `R_RGEOMORPHON_MEM_SCALE_NEED`. If unset, `10`.
`scl_workers`	*numeric*. Scaling factor for each worker. Default uses value of environment variable `R_RGEOMORPHON_MEM_SCALE_WORKERS`. If unset, `1`.
`pow_total`	*numeric*. Exponent for scaling total number of chunks. Default uses value of environment variable `R_RGEOMORPHON_MEM_POWER`. If unset, `1`.

Value

integer. Number of tile chunks to divide `x` into.

Examples

```
data("salton", package = "rgeomorphon")

x <- terra::rast(salton)
terra::ext(x) <- attr(salton, "extent")
terra::crs(x) <- attr(salton, "crs")

geomorphon_chunks_needed(x)
```

`print.forms_matrix` *Print method for a forms_matrix object*

Description

Controls how the 'forms_matrix' object is displayed in the console.

Usage

```
## S3 method for class 'forms_matrix'
print(x, show_values = FALSE, ...)
```

Arguments

`x`	The `forms_matrix` object to print.
`show_values`	A logical value. If `FALSE` (default), prints enum names. If `TRUE`, prints the underlying integer values.
`...`	Additional arguments passed to print (not used here).

Value

Invisibly returns the original object `x`.

Examples

```
print(forms_matrix_get(num_forms = 4))
```

salton *Bathymetric Information on California's Salton Sea*

Description

Matrix derived from one foot contours of the Salton Sea floor. This data was created with the vertical datum NGVD29 and NAD83 California Teale Albers (EPSG:3110) projection. Each value in the matrix represents the elevation, in meters, of a 300 m x 300 m cell. Cell values are interpolated using a thin plate spline fit to an exhaustive sample of contour line vertices.

Usage

```
salton
```

Format

matrix, with cells representing X, Y grid locations, and attributes `"crs"` (containing WKT2019 string with coordinate reference system information) and `"extent"` (named numeric of length 4, containing xmin, xmax, ymin, ymax)

Source

California Division of Fish and Wildlife. 2007. Bathymetric Contours (1 foot) - Salton Sea (ds426). Available online: https://map.dfg.ca.gov/metadata/ds0426.html

Examples

```
str(salton)

# construct and georeference a SpatRaster object
dem <- terra::rast(salton)
terra::crs(dem) <- attr(salton, "crs")
terra::ext(dem) <- attr(salton, "extent")
names(dem) <- "Elevation (feet)"

dem
```

Index