Package 'slopes'

June 17, 2026

Title Calculate Slopes of Roads, Rivers and Trajectories

Version 1.0.2

Description Calculates the slope (longitudinal gradient or steepness) of linear geographic features such as roads (for more details, see Ariza-López et al. (2019) <doi:10.1038/s41597-019-0147-x>) and rivers (for more details, see Cohen et al. (2018) <doi:10.1016/j.jhydrol.2018.06.066>). It can use local Digital Elevation Model (DEM) data or download DEM data via the 'ceramic' package. The package also provides functions to add elevation data to linestrings and visualize elevation profiles.

License GPL-3

URL https://github.com/ropensci/slopes/, https://docs.ropensci.org/slopes/

BugReports https://github.com/ropensci/slopes/issues

Encoding UTF-8

LazyData true

Imports sf, terra, methods, pbapply, geodist, colorspace

Depends R (>= 2.10)

Suggests knitr, rmarkdown, ceramic, bookdown, covr, testthat, osmextract, stplanr, dplyr, tmap, leaflet, bench, rlang

Config/Needs/website generics

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Robin Lovelace [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-5679-6536>), Rosa Félix [aut] (ORCID: <https://orcid.org/0000-0002-5642-6006>), Joey Talbot [aut] (ORCID: <https://orcid.org/0000-0002-6520-4560>), Gonçalo Matos [aut] (ORCID: <https://orcid.org/0009-0001-3489-1732>), Dan Olner [rev] (Dan reviewed the package for rOpenSci, see https://github.com/ropensci/software-review/issues/420#issuecomment-857662657),

Andy Teucher [rev] (Andy reviewed the package for rOpenSci, see
https://github.com/ropensci/software-review/issues/420#issuecomment-858231647)

Maintainer Robin Lovelace <rob00x@gmail.com>

Repository CRAN

Date/Publication 2026-06-17 07:20:08 UTC

Contents

cyclestreets_route *Cycle route data*

Description

Cycle route data

Usage

```
cyclestreets_route
```

Format

An sf object

dem_lisbon *Read the bundled Lisbon DEM as a SpatRaster*

Description

Returns a `SpatRaster` (terra package) of the Digital Elevation Model for central Lisbon, Portugal, bundled with the slopes package.

Usage

```
dem_lisbon()
```

Value

A `SpatRaster` object with 133 rows, 200 columns, and 1 elevation layer.

Examples

```
dem_lisbon()
```

distance_z *Extract distance and elevation data from route*

Description

Extracts cumulative distance and elevation vectors from route XYZ coordinates.

Usage

```
distance_z(route_xyz, lonlat)
```

Arguments

route_xyz	An sf object with XYZ coordinates
lonlat	Logical, whether coordinates are longitude/latitude

Value

List with components d (distances) and z (elevations)

elevation_add *Add elevation data to route linestrings*

Description

Adds elevation (Z) coordinates to linestring geometries using DEM data.

Usage

```
elevation_add(routes, dem = NULL, method = "bilinear", terra = NULL)
```

Arguments

routes	An sf object containing linestring geometries
dem	A SpatRaster object containing elevation data (default: NULL for automatic download)
method	Method for raster extraction (default: "bilinear")
terra	Deprecated. Ignored; terra is always used.

Value

An sf object with XYZ linestring geometries

Examples

```
library(sf)
routes = lisbon_road_network[204, ]
dem = dem_lisbon()
(r3d = elevation_add(routes, dem))
st_z_range(routes)
st_z_range(r3d)
plot(st_coordinates(r3d)[, 3])
plot_slope(r3d)
## Not run:
# Get elevation data (requires internet connection, ceramic pkg, and API key):
if (requireNamespace("ceramic", quietly = TRUE)) {
  r3d_get = elevation_add(cyclestreets_route)
  plot_slope(r3d_get)
}

## End(Not run)
```

---

elevation_extract *Extract elevation values from coordinates*

Description

Extracts elevation values from a DEM raster at specified coordinate locations. Accepts both `SpatRaster` (terra) and legacy `Raster*` (raster) objects; legacy objects are automatically converted to `SpatRaster`.

Usage

```
elevation_extract(m, dem, method = "bilinear", terra = NULL)
```

Arguments

`m`	Matrix or sf object with coordinates
`dem`	A SpatRaster (or legacy RasterLayer) containing elevation data
`method`	Method for raster extraction (default: "bilinear")
`terra`	Deprecated. Ignored; terra is always used.

Value

Numeric vector of elevation values

elevation_get *Get elevation data for routes*

Description

Downloads elevation data using the ceramic package for given routes. Returns a `SpatRaster` object (terra package).

Usage

```
elevation_get(routes, ...)
```

Arguments

`routes`	An sf object containing linestring geometries
`...`	Additional arguments passed to ceramic::cc_elevation

Value

A SpatRaster covering the routes

lisbon_road_network *Lisbon road network*

Description

Lisbon road network

Usage

```
lisbon_road_network
```

Format

An sf object

lisbon_road_segment *Lisbon road segment*

Description

Lisbon road segment

Usage

```
lisbon_road_segment
```

Format

An sf object

lisbon_road_segment_3d *Lisbon road segment 3D*

Description

Lisbon road segment 3D

Usage

```
lisbon_road_segment_3d
```

Format

An sf object

lisbon_road_segment_xyz_mapbox *Lisbon road segment XYZ*

Description

Lisbon road segment XYZ

Usage

```
lisbon_road_segment_xyz_mapbox
```

Format

An sf object

lisbon_route *Lisbon route data*

Description

Lisbon route data

Usage

```
lisbon_route
```

Format

An sf object

lisbon_route_3d *Lisbon route 3D*

Description

Lisbon route 3D

Usage

```
lisbon_route_3d
```

Format

An sf object

lisbon_route_xyz_mapbox *Lisbon route XYZ*

Description

Lisbon route XYZ

Usage

```
lisbon_route_xyz_mapbox
```

Format

An sf object

magnolia_xy *Magnolia coordinates*

Description

Magnolia coordinates

Usage

```
magnolia_xy
```

Format

A data frame

make_breaks *Create slope breaks for color mapping*

Description

Creates symmetric slope breaks around zero for color classification.

Usage

```
make_breaks(brks)
```

Arguments

brks	Vector of positive slope break values (as percentages)

Value

Vector of slope breaks including negative values and zero

make_colz *Assign colors to slope values*

Description

Maps slope gradient values to colors based on break points.

Usage

```
make_colz(g, b, pal)
```

Arguments

g	Vector of slope gradient values
b	Vector of break points
pal	Vector of colors corresponding to breaks

Value

Character vector of colors for each slope value

make_pal *Create color palette for slope visualization*

Description

Creates or processes color palettes for slope gradient visualization.

Usage

```
make_pal(pal, b)
```

Arguments

pal	Color palette (function or character vector)
b	Vector of breaks for color mapping

Value

Character vector of colors

plot_dz *Plot distance-elevation profile with slope coloring*

Description

Creates a distance-elevation plot with segments colored by slope gradient.

Usage

```
plot_dz(
  d,
  z,
  fill = TRUE,
  horiz = FALSE,
  pal = NULL,
  ...,
  legend_position = "top",
  col = "black",
  cex = 0.9,
  bg = grDevices::rgb(1, 1, 1, 0.8),
  title = "Slope colors (percentage gradient)",
  brks = c(3, 6, 10, 20, 40, 100),
  seq_brks = NULL,
  ncol = 4
)
```

Arguments

d	Vector of cumulative distances
z	Vector of elevation values
fill	Logical, whether to fill segments with slope colors (default: TRUE)
horiz	Logical, whether legend should be horizontal (default: FALSE)
pal	Color palette for slope visualization (default: NULL, uses slopes_palette)
...	Additional arguments passed to graphics functions
legend_position	Position of legend (default: "top")
col	Color of the elevation profile line (default: "black")
cex	Character expansion factor for legend text (default: 0.9)
bg	Background color for legend (default: semi-transparent white)
title	Title for the legend (default: "Slope colors (percentage gradient)")
brks	Vector of slope break points for coloring (default: c(3, 6, 10, 20, 40, 100))
seq_brks	Sequence of breaks to show in legend (default: NULL, auto-generated)
ncol	Number of columns in legend (default: 4)

Value

NULL (creates plot as side effect)

`plot_slope` *Plot elevation profile with slope coloring*

Description

Creates an elevation profile plot from route geometries with XYZ coordinates, with segments colored according to slope gradient.

Usage

```
plot_slope(
  route_xyz,
  lonlat = sf::st_is_longlat(route_xyz),
  fill = TRUE,
  horiz = FALSE,
  pal = NULL,
  legend_position = "top",
  col = "black",
  cex = 0.9,
  bg = grDevices::rgb(1, 1, 1, 0.8),
  title = "Slope colors (percentage gradient)",
  brks = c(3, 6, 10, 20, 40, 100),
  seq_brks = seq(from = 3, to = length(brks) * 2 - 2),
  ncol = 4,
  ...
)
```

Arguments

`route_xyz`	An sf object containing linestring geometries with XYZ coordinates
`lonlat`	Logical, whether coordinates are longitude/latitude (default: auto-detected)
`fill`	Logical, whether to fill segments with slope colors (default: TRUE)
`horiz`	Logical, whether legend should be horizontal (default: FALSE)
`pal`	Color palette for slope visualization (default: NULL, uses slopes_palette)
`legend_position`	Position of legend (default: "top")
`col`	Color of the elevation profile line (default: "black")
`cex`	Character expansion factor for legend text (default: 0.9)
`bg`	Background color for legend (default: semi-transparent white)
`title`	Title for the legend (default: "Slope colors (percentage gradient)")
`brks`	Vector of slope break points for coloring (default: c(3, 6, 10, 20, 40, 100))

seq_brks	Sequence of breaks to show in legend (default: auto-generated)
ncol	Number of columns in legend (default: 4)
...	Additional arguments passed to plot_dz

Value

NULL (creates plot as side effect)

`route_to_segments` *Split a route into vertex-to-vertex segments*

Description

Splits a linestring with XYZ coordinates into individual 2-point segments, one per consecutive vertex pair. Useful for computing per-segment slopes with `slope_xyz()`.

Usage

```
route_to_segments(route_xyz)
```

Arguments

route_xyz	An sf object with a single LINESTRING geometry with Z coordinates, as returned by `elevation_add()`.

Value

An sf object with one LINESTRING feature per vertex-to-vertex segment.

Examples

```
route_xyz = elevation_add(lisbon_route, dem = dem_lisbon())
segs = route_to_segments(route_xyz)
segs$slope = slope_xyz(segs)
summary(segs$slope)
```

sequential_dist *Calculate sequential distances between points*

Description

Calculates distances between consecutive points in a coordinate matrix.

Usage

```
sequential_dist(m, lonlat = TRUE)
```

Arguments

m	Matrix of coordinates (x, y)
lonlat	Logical, whether coordinates are longitude/latitude (default: TRUE)

Value

Numeric vector of distances between consecutive points

sf_mid_ext_lonlat *Extract midpoint and extent from routes in lonlat*

Description

Internal helper function to get midpoint and extent of routes in lon/lat coordinates.

Usage

```
sf_mid_ext_lonlat(routes)
```

Arguments

routes	An sf object containing linestring geometries

Value

A list with midpoint coordinates and width/height dimensions

slopes_palette *Get color palette for slopes visualization*

Description

Returns a color palette suitable for visualizing slope data, with options for different color schemes.

Usage

```
slopes_palette(n = 6, palette = "Green-Brown")
```

Arguments

n	Number of colors to return (default: 6)
palette	Name of the color palette to use (default: "Green-Brown")

Value

A character vector of color codes

Examples

```
# Get default Green-Brown palette with 6 colors
slopes_palette()

# Get 4 colors from Green-Brown palette
slopes_palette(n = 4)

# Use a different palette
slopes_palette(n = 5, palette = "Blue-Red")
```

slope_distance *Calculate slopes using distance data*

Description

Calculates slope gradients from distance and elevation vectors.

Usage

```
slope_distance(d, elevations)
```

Arguments

d	Vector of distance values between points
elevations	Vector of elevation values

Value

Numeric vector of slope values

---

slope_distance_mean *Calculate mean slope using distance weighting*

Description

Computes the mean slope across segments using distance-weighted averaging.

Usage

```
slope_distance_mean(d, elevations, directed = FALSE)
```

Arguments

d	Vector of distance values between points
elevations	Vector of elevation values
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric value representing the mean slope

---

slope_distance_weighted *Calculate distance-weighted slopes*

Description

Applies distance-based weighting to slope calculations for more accurate results.

Usage

```
slope_distance_weighted(d, elevations, directed = FALSE)
```

Arguments

d	Vector of distance values between points
elevations	Vector of elevation values
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric value representing the weighted slope

slope_matrices *Calculate slopes for multiple coordinate matrices*

Description

Applies slope calculation function to a list of coordinate matrices.

Usage

```
slope_matrices(m_xyz_split, fun = slope_matrix_weighted, ...)
```

Arguments

m_xyz_split	List of coordinate matrices with elevation data
fun	Function to apply for slope calculation (default: slope_matrix_weighted)
...	Additional arguments passed to the slope function

Value

Numeric vector of slope values for all matrices

slope_matrix *Calculate slopes from coordinate matrix*

Description

Calculates slope gradients from a matrix of coordinates and elevation data.

Usage

```
slope_matrix(m, elevations = m[, 3], lonlat = TRUE)
```

Arguments

m	Matrix of coordinates (x, y, z)
elevations	Vector of elevation values (default: third column of m)
lonlat	Logical, whether coordinates are longitude/latitude (default: TRUE)

Value

Numeric vector of slope values

slope_matrix_mean *Calculate mean slope from coordinate matrix*

Description

Computes the mean slope from a matrix of coordinates with elevation data.

Usage

```
slope_matrix_mean(m, elevations = m[, 3], lonlat = TRUE, directed = FALSE)
```

Arguments

m	Matrix of coordinates (x, y, z)
elevations	Vector of elevation values (default: third column of m)
lonlat	Logical, whether coordinates are longitude/latitude (default: TRUE)
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric value representing the mean slope

slope_matrix_to_raster *Convert slope matrix to SpatRaster*

Description

Converts a slope matrix or a legacy RasterLayer to a SpatRaster (terra). Accepts `SpatRaster`, legacy `Raster*`, or a plain matrix.

Usage

```
slope_matrix_to_raster(x)
```

Arguments

x	A matrix, SpatRaster, or legacy RasterLayer object

Value

A SpatRaster object

slope_matrix_weighted *Calculate weighted slopes from coordinate matrix*

Description

Applies distance-based weighting to slope calculations from coordinate matrix.

Usage

```
slope_matrix_weighted(m, elevations = m[, 3], lonlat = TRUE, directed = FALSE)
```

Arguments

m	Matrix of coordinates (x, y, z)
elevations	Vector of elevation values (default: third column of m)
lonlat	Logical, whether coordinates are longitude/latitude (default: TRUE)
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric value representing the weighted slope

slope_raster *Calculate slopes using raster elevation data*

Description

Calculates slope gradients for routes using digital elevation model (DEM) raster data.

Usage

```
slope_raster(
  routes,
  dem,
  lonlat = sf::st_is_longlat(routes),
  method = "bilinear",
  fun = slope_matrix_weighted,
  terra = NULL,
  directed = FALSE
)
```

Arguments

routes	An sf object containing linestring geometries
dem	A SpatRaster object (terra package) containing elevation data
lonlat	Logical, whether coordinates are longitude/latitude (default: auto-detected)
method	Method for raster extraction (default: "bilinear")
fun	Function for slope calculation (default: slope_matrix_weighted)
terra	Deprecated. Ignored; terra is always used.
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric vector of slope values

---

slope_vector *Calculate slopes from vector data*

Description

Calculates slope gradients using vector distance and elevation data.

Usage

```
slope_vector(x, elevations)
```

Arguments

x	Vector of distance values
elevations	Vector of elevation values

Value

Numeric vector of slope values

slope_xyz *Calculate slopes from XYZ coordinate data*

Description

Calculates slope gradients from linestring geometries with XYZ coordinates.

Usage

```
slope_xyz(
  route_xyz,
  fun = slope_matrix_weighted,
  lonlat = TRUE,
  directed = FALSE
)
```

Arguments

route_xyz	An sf object or data frame with XYZ coordinates
fun	Function for slope calculation (default: slope_matrix_weighted)
lonlat	Logical, whether coordinates are longitude/latitude (default: TRUE)
directed	Logical, whether to calculate directed slopes (default: FALSE)

Value

Numeric vector of slope values

slope_xyz_simple *Extract XYZ coordinates from SpatRaster or matrix*

Description

Simplifies raster or matrix data to XYZ coordinate format. Accepts SpatRaster (terra), legacy Raster* objects, or a plain matrix.

Usage

```
slope_xyz_simple(x)
```

Arguments

x	A SpatRaster, legacy RasterLayer, or matrix object

Value

A data frame with x, y, z coordinates

Index