Package: rTARDIS (via r-universe)

October 22, 2026

Title Terrains and Routes Directed in Spacetime

Version 2.1.2

Description Implements classes and methods for constructing and analysing graph representations of complex geographic spaces at local to global scales, particularly where accessible spaces are discontinuously distributed but retain connectivity (e.g., island systems where isolation is prevalent but not necessarily absolute), and where that distribution also changes through time (e.g., habitat patch fragmentation or plate tectonic drift). The package (originally named TARDIS) was first described and published in: Flannery-Sutherland, J., Elsler, A., Farnsworth, A., Lunt, D., Benton, M. (2025) <doi:10.1038/s41559-025-02739-y>.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 2.10)

LazyData true

Imports cppRouting, exactextractr, geosphere, h3jsr, h3r, igraph, Matrix, methods, palaeoverse, rgl, rlemon, scales, sf, stats, terra

Suggests chronosphere, knitr, rmarkdown, testthat (>= 3.0.0), via

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

Config/pak/sysreqs libabsl-dev cmake libfreetype6-dev libgdal-dev gdal-bin libgeos-dev libglpk-dev libglu1-mesa-dev make texlive libicu-dev libpng-dev libuv1-dev libxml2-dev libgl1-mesa-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libnode-dev zlib1g-dev

Repository https://jf15558.r-universe.dev

Date/Publication 2026-10-21 19:14:13 UTC

RemoteUrl https://github.com/jf15558/rtardis

RemoteRef HEAD

RemoteSha dab9a90239704668f55642f1fde87cc726b452b7

Contents

build_tardis . . . 3
cell_to_coords . . . 4
chelonoides . . . 5
click_iso . . . 6
click_lcp . . . 7
click_msa . . . 8
click_optim . . . 9
click_points . . . 11
click_to_link . . . 12
click_to_snip . . . 13
cost_surface . . . 14
cretaceous . . . 15
cumulative_cost . . . 15
domains . . . 17
extract_from_geoglist . . . 18
galapagos . . . 19
get_cost . . . 20
get_grid . . . 21
get_rotations . . . 22
instantiate_tardis . . . 23
isochrone . . . 24
least_cost . . . 25
link_delaunay . . . 26
link_islands . . . 27
map_to_geoglist . . . 29
min_span . . . 30
optim_route . . . 31
plot . . . 33
plot_globe . . . 35
point_check . . . 36
project_geoglist . . . 38
rand_walk . . . 39
rast_to_geoglist . . . 40
read_geoglist . . . 42
regions . . . 43
rotation_path . . . 44
save_geoglist . . . 45
slice_geoglist . . . 45
slice_tardis . . . 46
tardis_to_sparse . . . 47
time_shift . . . 49
weight_tardis . . . 50

Index 54

`build_tardis` *build_tardis*

Description

Generate a tardis graph from a `geoglist`, where graph weights represent the geographic distances between cells. The function assumes that the input `geoglist` contains topographic and/or bathymetric data measured in metres.

Usage

```
build_tardis(geog, tlink = 1, rotations = NULL, verbose = TRUE)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`tlink`	`integer`. The linking mode through time if multiple layers are are present in geog: either `1` (forwards-in-time), `2` (backwards- in-time) or `3` (bidirectional). Defaults to `1`.
`rotations`	`list` or `NULL`. By default `NULL`, indicating that temporal links are spatially constant. Otherwise, a list with `nlayers(geog) - 1` elements recording the shift in cell locations between layers.
`verbose`	`logical`. Should function progress be reported to the user?

Details

A `tardis` object is an S3 class. It is a list with elements `edges`, `gdat`, `tdat` and `tlink`. The first two are taken from `geog`, and `tlink` from the corresponding function argument. `edges` is a graph edgelist. The graph can be envisaged as a 3D lattice, where layers in the lattice correspond to time layers in `geog`. Graph weights for horizontal edges within layers record the great circle distances between adjacent cells in metres, adjusted for differences in cell elevation using Pythagoras's theorem. For raster inputs, 8-degree (Queen's case) adjacency is used, while for hexagonally resampled inputs, 6-degree adjacency is used. Single layer cases (i.e. no time element) are allowed. Otherwise vertical edges between time layers are assigned weights of zero so that they do not affect downstream distance calculations.

In many cases, the positions of cells will remain constant within the extent of each layer in `geog` and the `rotations` arguments can be left as `NULL`. For global landscapes over geological timescales, however, the positions of landmasses chance noticeably due to continental drift. In these cases, rotations can be a list of two-column numeric matrices containing the IDs of geographically homologous cells between successive pairs of layers in `geog`. No list element can be blank as there must be at least one edge between each pair of layers. Otherwise, any number of edges between layers can be specified, although it makes sense for there to be maximally as many edges as cells within a layer. As this is quite a specific format, the function `get_rotations` can be used to expedite its construction. Examples of the rotations object are available in its documentation.

Value

An object of class `tardis`.

Examples

```
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
byrow = TRUE), right = FALSE)

# create a geoglist with hexagonal resampling and mask the sea
hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)

# build a tardis with hexagonal cells
htd <- build_tardis(hexes)

# create a single layer geoglist in raster format and mask the sea
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])

# build a tardis from raster cells
rtd <- build_tardis(rasts)
```

`cell_to_coords` *cell_to_coords*

Description

Convert a cell integer ID to its lon-lat coordinates, spatial cell ID and layer, based on based on the general gridding parameters used by `rTARDIS`. This will work for almost any integer (see details) and does not relate to any specific landscape structure. See `point_check()` instead if you are trying to resolve numeric coordinates to cells within a specific landscape.

Usage

```
cell_to_coords(cellid, gdat, modulo.warning = TRUE)
```

Arguments

`cellid`	`vector`. A vector of cell integer IDs, either numeric or character.
`gdat`	Either a vector in gdat format (see examples), or an object containing this information (i.e., `tardis` or `geoglist`).
`modulo.warning`	`logical`. Should the function warn the user if zero modulo cells are present?

Details

Spatial cell position is retrieved as the modulo of layer cell count: `cellid %% gdat[5]`. Temporal position is retrieved by integer division of the layer cell count: `(cellid %/% gdat[5]) + 1`.

In very rare cases (one in tens of thousands of cell), the modulo operation results in a spatial cell index of zero and will not work. This behavior is accounted for in all `rTARDIS` functions by masking such cells from the landscape. However, lon-lat coordinates for these rare cells cannot be derived, hence why they are returned as `NA` in the `data.frame` output.

Value

A `data.frame` recording the lon-lat coordinates, spatial cell IDs and layer numbers of the input cell IDs. Integers which modulo to zero will return `NA` for their lon-lat coordinates (see details).

Examples

```
library(terra)
library(rTARDIS)

# using an rTARDIS object
gal <- galapagos()
rasts <- rast_to_geoglist(gal[[1]])
cell_to_coords(1012, rasts)

# using a manually-built gdat
gvec <- setNames(
            c(-92.0048120, -88.0048120, -2.0010895, 0.9989105, 3609, NA, 6),
            c("xmin", "xmax", "ymin", "ymax", "ncell", "ncol", "hex")
            )
cell_to_coords(c(1, 5, 233, 18), gvec)
```

---

chelonoides *chelonoides*

Description

A `phylo` class object, as defined by the `ape` R packakge, containing a time-scaled phylogenetic tree for the 13 known species of Galapagos giant tortoises (*Chelonoides* spp.). In addition to the branch lengths, the tree object also contains the root age (`$root.age`) and a `data.frame` recording the node ages in millions-of-years ago and modern endemic distributions (islands, coordinates) of each tip (`$biogeography`).

Usage

```
chelonoides
```

Format

An object of class `phylo` of length 6.

Details

Tree topology and branch lengths were inferred based on mitochondrial DNA, then origin islands inferred using the `BioGeoBEARS` R package.

Galapagos giant tortoises show highly endemic geographic distributions, even where multiple species co-exist on one island. The longitude-latitude coordinates correspond to the population epicentres for each species. For the internal nodes of the tree, the ancestral island was inferred, but not the precise coordinates, hence why the coordinates are NA for these entries.

Island abbreviations: Isa = Isabella, Santi = Santiago, SantaC = Santa Cruz, SanC = San Cristobal, Pin = Pinta, Esp = Espagola, Flo = Floreana

Source

Time-scaled tree and node-origin islands from Poulakakis et al. (2020). Colonization history of Galapagos giant tortoises: Insights from mitogenomes support the progression rule. Journal of Zoological Systematics and Evolutionary Research, 58, 1262-1275

Modern species locations from Jensen et al (2022). The Galapagos giant tortoise Chelonoidis phantasticus is not extinct. Communications Biology, 5, 546

`click_iso` *click_iso*

Description

Interactively calculate and display an isochrone around a clicked point on the landscape plotted by the function. Clicked points falling in masked regions are automatically resolved to the nearest available cell.

Usage

```
click_iso(
  tardis,
  weights = "gdist",
  geog,
  time = NULL,
  cost = 1e+06,
  col = "gold",
  ...
)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.

geog	geoglist. The geoglist used to build tardis.
time	integer. The tardis time slice to plot and interact with. Defaults to NULL, in which case the first slice is used.
cost	numeric. The maximum cost for calculating the isochrone.
col	character. The colour to use for plotting the isochrone.
...	Additional arguments passed to plot.geoglist().

Value

No return value.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)
hexes <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0), as.hex = TRUE, hex = 7)
hlink <- link_islands(hexes)
htd <- build_tardis(hexes)

# click a point on the map
click_iso(tardis = htd, geog = hexes, time = 2, cost = 1e5)
```

---

click_lcp *click_lcp*

Description

Interactively calculate and display least cost paths by clicking their start and end points on the landscape plotted by the function. Clicked points falling in masked regions are automatically resolved to the nearest available cell.

Usage

```
click_lcp(
  tardis,
  weights = "gdist",
  geog,
  time = NULL,
  n = 1,
  col = "gold",
  ...
)
```

Arguments

tardis	tardis. The output of build_tardis().
weights	character. The name of the weighting scheme in tardis to use for distance calculation. By default these are true geographic distances ("gdist"). Alternatively, the name of another weighting scheme added to tardis using weight_tardis().
geog	geoglist. The geoglist used to build tardis.
time	integer. The tardis time slice to plot and interact with. Defaults to NULL, in which case the first slice is used.
n	integer. the number of point pairs to run (only for click_lcp)
col	character. The colour to use for plotting least cost paths.
...	Additional arguments passed to plot.geoglist().

Value

No return value.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)
hexes <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0), as.hex = TRUE, hex = 7)
hlink <- link_islands(hexes)
htd <- build_tardis(hexes)

# click two start and end points on the map
click_iso(tardis = htd, geog = hexes, time = 2, n = 2)
```

---

click_msa *click_msa*

Description

Interactively calculate and display a minimum spanning arboresence for a set of points by clicking on the landscape plotted by the function. Clicked points falling in masked regions are automatically resolved to the nearest available cell.

Usage

```
click_msa(
  tardis,
  weights = "gdist",
  geog,
```

```
  time = NULL,
  n = 1,
  col = "gold",
  ...
)
```

Arguments

tardis	tardis. The output of build_tardis().
weights	character. The name of the weighting scheme in tardis to use for distance calculation. By default these are true geographic distances ("gdist"). Alternatively, the name of another weighting scheme added to tardis using weight_tardis().
geog	geoglist. The geoglist used to build tardis.
time	integer. The tardis time slice to plot and interact with. Defaults to NULL, in which case the first slice is used
n	integer. The number of points to link with a minimum spanning arborescence.
col	character. The colour to use for plotting the minimum spanning arborescence.
...	Additional arguments passed to plot.geoglist()

Value

No return value.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)
hexes <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0), as.hex = TRUE, hex = 7)
hlink <- link_islands(hexes)
htd <- build_tardis(hexes)

# click a point on the map
click_iso(tardis = htd, geog = hexes, time = 2, cost = 1e5)
```

---

click_optim *click_optim*

Description

Interactively calculate and display the optimal travel path linking a set of points by clicking on the landscape plotted by the function. Clicked points falling in masked regions are automatically resolved to the nearest available cell.

Usage

```
click_optim(
  tardis,
  weights = "gdist",
  geog,
  time = NULL,
  n = 3,
  loop = FALSE,
  col = "gold",
  ...
)
```

Arguments

tardis	tardis. The output of build_tardis().
weights	character. The name of the weighting scheme in tardis to use for distance calculation. By default these are true geographic distances ("gdist"). Alternatively, the name of another weighting scheme added to tardis using weight_tardis().
geog	geoglist. The geoglist used to build tardis.
time	integer. The tardis time slice to plot and interact with. Defaults to NULL, in which case the first slice is used.
n	integer. The number of points to link with an optimal path. This should be greater than 2.
loop	logical. Should the optimal route additionally be closed from end point to start point to return a polygon? Defaults to FALSE.
col	character. The colour to use for plotting the optimal path.
...	Additional arguments passed to plot.geoglist()

Value

No return value.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)
hexes <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0), as.hex = TRUE, hex = 7)
hlink <- link_islands(hexes)
htd <- build_tardis(hexes)

# click a point on the map
click_iso(tardis = htd, geog = hexes, time = 2, cost = 1e5)
```

click_points *click_points*

Description

Click locations on a `geoglist` layer and return the coordinates and cell IDs of those locations. All locations are resolved to the nearest available cell, including any which fall outside of unmasked areas of the layer.

Usage

```
click_points(geog, layer = 1, points = 1, print.only = TRUE, ...)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`layer`	`numeric`. The layer in the `geoglist` to be plotted, along with its links.
`points`	`numeric`. The number of points you wish to generate.
`print.only`	`logical`. Defaults to `TRUE`, printing cell indices and values to the console only. If `FALSE`, then those points are returned as a `SpatVector` object.
`...`	Additional arguments passed to `plot.geoglist()`

Value

Either `NULL` (default) or a `SpatVector` of the clicked points.

Examples

```
library(terra)
library(rTARDIS)

# load data
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)

# make geoglist
rasts <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0))

# click to get two points
click_points(rasts, points = 2)
```

click_to_link *click_to_link*

Description

Interactively add links to a geoglist by clicking their start and end points in the plotting window. Clicked locations are automatically resolved to the nearest cell bordering masked space in the geoglist.

Usage

```
click_to_link(geog, layer = 1, nlinks = 1, ...)
```

Arguments

geog	geoglist. The output of rast_to_geoglist().
layer	numeric. The layer in the geoglist to be plotted, along with its links.
nlinks	numeric. The number of links you wish to generate. Simply rerun the function if you need to add more lines.
...	Additional arguments passed to plot.geoglist()

Value

The input geoglist with added links.

Examples

```
library(terra)
library(rTARDIS)

# load data
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)

# make geoglist
rasts <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0))

# click the start and end points for two links
click_to_link(rasts, nlinks = 2)
```

click_to_snip *click_to_snip*

Description

Interactively remove links from a `geoglist` by drawing lines through them in the plotting window. Lines are drawn by clicking their start and end points in the plotting window.

Usage

```
click_to_snip(geog, layer = 1, nsnips = 1, ...)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`layer`	`numeric`. The layer in the `geoglist` to be plotted, along with its links.
`nsnips`	`numeric`. The number of snipping lines you wish to generate. Simply rerun the function if you need to remove more lines.
`...`	Additional arguments passed to `plot.geoglist()`

Value

The input `geoglist` with the snipped links removed.

Examples

```
library(terra)
library(rTARDIS)

# load data
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)

# make geoglist and add links
rasts <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0))
rasts <- link_islands(rasts)

# click to either side of a link to remove it
click_to_snip(rasts)
```

cost_surface *cost_surface*

Description

Visualise the weighting scheme for the layers in a `tardis` object. Weights are calculated as the mean of inbound edges for a given cell (i.e., the average cost to reach that cell from its neighbours).

Usage

```
cost_surface(tardis, weights = "gdist", exclude.links = FALSE, verbose = TRUE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`exclude.links`	`logical`. Should the weights of island links be excluded from the cost calculation? Defaults to `FALSE`.
`verbose`	`logical`. Should function progress be reported to the user?

Value

A `geoglist` with layers corresponding to the landscape layers in the graph, and cell values calculated as the mean of their inbound edge weights.

Examples

```
library(terra)
library(rTARDIS)

# load data
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)

# make geoglist and add links
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])
rlink <- link_islands(rasts)

rtd <- build_tardis(rasts)
cs <- cost_surface(rtd)
plot(cs)
```

cretaceous *cretaceous*

Description

Load a `terra` R package `SpatRaster` object containing two palaeogeographic digital elevation models of the Earth during the Cretaceous. Models are in geographic coordinates with elevations in metres relative to sea level.

Usage

```
cretaceous()
```

Details

Each DEM records the estimated topography and bathymetry of the Earth at 115 and 110 million years in the past. As such, each time slice is taken to represent the 2.5 million years either side of these points (i.e., the 110 Ma reconstruction spans 112.5-107.5 Ma). Both reconstructions come from the PALEOMAP plate rotation model series by Chris Scotese and colleagues, and were obtained using the `fetch()` function of the `chronosphere` R package.

Value

`SpatRaster`.

Source

Kocsis, A. Scotese, C., Raja, N. (2013) chronosphere: Evolving Earth System Variables. Zenodo. https://doi.org/10.5281/zenodo.3525481.

Scotese, C., Verard, C., Burgener, L., Elling, R., Kocsis, A. (2024). Phanerozoic-scope supplementary material to 'The Cretaceous World: Plate Tectonics, Paleogeography, and Paleoclimate' from the PALEOMAP project (Version v24221). Zenodo. https://doi.org/10.5281/zenodo.10659112.

Scotese, C., Verard, C., Burgener, L., Elling, R., Kocsis, A. (2025). The Cretaceous World: Plate Tectonics, Paleogeography, and Paleoclimate. Geological Society, London, Special Publications, 544, SP544b

cumulative_cost *cumulative_cost*

Description

From a given starting cell, calculate the cost of reaching every other location in the same time slice in a `tardis` graph.

Usage

```
cumulative_cost(tardis, weights = "gdist", origin, verbose = TRUE)
```

Arguments

tardis	tardis. The output of build_tardis().
weights	character. The name of the weighting scheme in tardis to use for distance calculation. By default these are true geographic distances ("gdist"). Alternatively, the name of another weighting scheme added to tardis using weight_tardis().
origin	SpatVector. The output of point_check(), denoting the point from which to calculate cumulative costs.
verbose	logical Should function progress be reported to the user?

Value

A geoglist recording the costs required to reach, from the origin point, each cell in the same time slice as that point.

Examples

```
library(terra)
library(rTARDIS)

# load data
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# build a geoglist and add links
hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

# build a tardis graph
htd <- build_tardis(hexes)

# resolve a destination point to that graph
org <- rbind(c(-89.78873, -1.420627, 2))
hpts <- point_check(htd, org)

# get the cumulative cost around that point
cumulative_cost(htd, origin = hpts)
```

domains *domains*

Description

Calculate Voronoi domains around islands in a `geoglist`. These polygonal domains are the basis for how connections are determined across masked areas.

Usage

```
domains(geog)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.

Value

A `SpatVectorCollection` with each element containing the Voronoi polygons for the corresponding layer in the input `geoglist`.

Examples

```
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist with hexagonal resampling and mask the sea
hexes <- rast_to_geoglist(gal[[1]], gal_m[[1]], as.hex = TRUE, hex = 6)

dms <- domains(hexes)
plot(hexes, 1)
plot(dms, add = TRUE)
```

extract_from_geoglist *extract_from_geoglist*

Description

Extract values from the layers in a `geoglist` using a terra R package `SpatVector` containing one or several geometries. This will typically come from another `rTARDIS` function (e.g, `least_cost()`, `isochrone()`, ect), but could instead be user-designed.

Usage

```
extract_from_geoglist(geog, geom, layer = NULL)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`geom`	`SpatVector`. The geometry or geometries which will be used to extract values from `geog`.
`layer`	`numeric`. If not `NULL`, then an integer specifying the layer in `geog` from which values are to be extracted. This argument is primarily intended for use with user-designed `geom` objects which do not contain layer assignments, unlike returns from other `rTARDIS` functions.

Value

`SpatVector`. A `SpatVector` of points corresponding to the centroids of all cells in `geog` intersected by each geometry in `geom` (denoted by `$feature`), the layers of each intersected cell (`$layer`), and the cell values (`$value`).

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

rasts <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
rasts <- link_islands(rasts)
rtd <- build_tardis(rasts)
org <- rbind(c(-89.78873, -1.420627, 2),
             c(-89.58525, -1.473917, 2))
dst <- rbind(c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

rpts <- point_check(rtd, rbind(org, dst))
```

```
rlcp <- least_cost(rtd, origin = rpts[1,], dest = rpts[3,])
vals <- extract_from_geoglist(rasts, rlcp)
```

---

galapagos *galapagos*

Description

Load a terra 'SpatRaster' object from a source tiff containing four palaeogeographic digital elevation models and one modern digital elevation model of the Galapagos archipelago. Models are in geographic coordinates with elevations in metres above sea level.

Usage

```
galapagos()
```

Details

Each DEM records the topography and bathymetry of the archipelago at 0.5 million-year intervals from 2 million years ago to the present. As such, each time slice is taken to represent the 0.25 million years either side of these average points (i.e., the 2 Ma reconstruction spans 2.25-1.75 Ma).

The palaeogeographic reconstructions were interpolated from an irregular grid and additionally contain artefacts. As such, they should only be treated as toy datasets, rather than wholly accurate, and later versions are expected to be released in the future.

The present day reconstruction was sourced from the Pacific Marine Environmental Laboratory from data comppiled by William Chadwick, Oregon State University, although it has been downscaled to palaeogeographic resolution and so should again be treated as an example dataset, rather than the basis for a scientifically robust analysis.

Value

SpatRaster

Source

Past DEM: supplied by the original authors from Karnauskas et al (2017). Paleoceanography of the eastern equatorial Pacific over the past 4 million years and the geologic origins of modern Galápagos upwelling. Earth and Planetary Science Letters, 460, 22-28

Modern DEM: documented at https://www.pmel.noaa.gov/eoi/staff/chadwick/galapagos.html, downloaded from the embedded link in the page: ftp://ftp.pmel.noaa.gov/newport/chadwick/galap_bathy/gala

get_cost *get_cost*

Description

Get the cost of movement between cells in a `tardis` graph. Costs can be calculated between pairs of cell, from one cell to many, or pairwise between a set of cells.

Usage

```
get_cost(tardis, weights = "gdist", origin, dest)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`origin`	`character`. A vector of `tardis` cell IDs.
`dest`	`character`. A vector of `tardis` cell IDs.

Details

Getting the costs for a set of cells effectively creates a distance matrix. This can very quickly create large objects. For example, a `tardis` graph with ~20,000 accessible cells will give a 20k x 20k matrix approaching a gigabyte in size. You may wish to optimise the resolution and masking of your landscapes first if costing a large set of cells.

Value

For a pair of cells, a single value is returned. For one cell to many, a vector of cost values is returned. For a set of cells, a square matrix of costs is returned

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1]], gal_m[[1]], as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)
pairs <- get_cost(htd, origin = "19", dest = "1806")
one_to_many <- get_cost(htd, origin = "19", dest = c("19", "23", "26", "31"))
```

```
many_to_many <- get_cost(htd, origin = c("19", "23", "26", "31"),
                              dest = c("19", "23", "26", "31"))
```

---

get_grid *get_grid*

Description

Return all H3 cell IDs at a specified resolution within a rectangular bounding box.

Usage

```
get_grid(bbox, hex)
```

Arguments

bbox	`numeric`. A vector of four numbers denoting the xmin, xmax, ymin and ymax coordinates of the bounding box within which cells are to be retrieved.
hex	`integer`. The H3 resolution (1 to 15) at which to retrieve cells. Note that retrieving high resolution grids (`hex` > 4) for large (i.e., approaching global) extents will take a long time and may become prohibitive.

Details

The function can be viewed as the functional equivalent to a vector of raster grid cell IDs at a specified resolution within that same bounding box. The main difference is that the number of hexagonal cells in a given bounding box cannot be easily calculated, unlike for a raster grid. The function additionally controls for misbehaviour of bounding boxes approaching global extents, avoiding subsequent complications for H3 cell retrieval.

Value

A character vector of H3 cell IDs.

Examples

```
library(rTARDIS)
get_grid(bbox = c(-10, 13.2, -0.5, 30), hex = 3)
```

get_rotations *get_rotations*

Description

Generate a rotation list from commonly-used plate rotation models for use with `build_tardis()`. Ensure that the same plate rotation model on which the input `geoglist` is based is requested for function call.

Usage

```
get_rotations(geog, times, model, method = "grid", verbose = TRUE, ...)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`. This should contain at least two layers, representing two palaeogeographic time slices where there is appreciable intervening plate rotation.
`times`	`numeric`. A vector with `nlayers(geog) + 1` positive elements, expressing the temporal boundaries of each layer as millions of years in the past. The vector need not end in the present (i.e. `0`), but time must flow from oldest to youngest. Reconstructions will be made using the midpoints of these intervals
`model`	`character`. The desired plate reconstruction model. See `palaeoverse::palaeorotate()` for details.
`method`	`character`. The reconstruction method to use. See `palaeoverse::palaeorotate()` for details.
`verbose`	`logical` Should function progress be reported to the user?
`...`	Other arguments passed to `palaeoverse::palaeorotate()`.

Details

Cell reconstruction from layer to layer is performed by `palaeoverse::palaeorotate()`. By default this uses the `"grid"` method, by retrieving reconstructed coordinates from pre-rotated H3 grids at resolution 3 (~119 km). This is almost certainly sufficient for any global scale analyses given that global palaeogeographic models of topography and bathymetry rarely exceed 1 degree resolution (~111 km), while models which do exceed this resolution can become prohibitively computationally expensive to work with.

Value

A list with as many elements as layers in `geog$layers`. Each element is a two-column matrix containing the IDs of geographically homologous cells between successive landscape layers.

Examples

```
library(terra)
library(rTARDIS)

rast <- rast(nrows = 2, ncols = 2, vals = 1)
geog <- rast_to_geoglist(c(rast, rast))
ages <- c(440, 430, 420)
foo <- get_rotations(geog, times = ages, model = "MERDITH2021")
```

---

instantiate_tardis *instantiate_tardis*

Description

Take a `tardis` object and generate a graph compatible with functions from the `rcppRouting` package. This function is primarily called internally by other `rTARDIS` package functions, but it may be useful for the user to be able to directly interact with this graph themselves, or to supply a pre-instantiated `tardis` object within a loop to avoid repeated calls to this function.

Usage

```
instantiate_tardis(tardis, weights = "gdist")
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.

Value

. The input `tardis` object with the additional element, `$tgraph`.

Examples

```
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist in raster format and mask the sea
```

```
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])
rasts <- link_islands(rasts)

# build a tardis from raster cells
rtd <- build_tardis(rasts)

# instantiate
rtd <- instantiate_tardis(rtd)
```

`isochrone` *isochrone*

Description

Calculate a cost isochrone around a point, that is all cells in the same time layer which fall within a certain cost of access of that point.

Usage

```
isochrone(tardis, weights = "gdist", origin, cost = 1e+05, verbose = TRUE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`origin`	`SpatVector`. The output of `point_check()`, denoting the points around which to calculate isochrones.
`cost`	The maximum cumulative cost up to which cells will be included within the isochrone around a point. Either a single number or a vector with as many elements as points in `origin` to enable different costs for each point. By default, this value assumed to be in metres, corresponding to the `rTARDIS` default weighting scheme.
`verbose`	`logical` Should function progress be reported to the user?

Value

A `SpatVector` containing isochrone polygons, recording to which input point in `origin` they correspond to (`$feature`) and their time layer (`$layer`).

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist and link islands
hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)

org <- rbind(c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

hpts <- point_check(htd, org)
foo <- isochrone(htd, origin = hpts, cost = 100000)
```

---

least_cost *least_cost*

Description

Determine least cost paths between pairs of origin and destination coordinates using Djikstra's algorithm. Costs are either geographic distances or a custom weighting scheme supplied by the user.

Usage

```
least_cost(tardis, weights = "gdist", origin, dest, verbose = TRUE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`origin`	`SpatVector`. The output of `point_check()`, denoting the starting points of the least cost paths.
`dest`	`SpatVector`. The output of `point_check()`, denoting the destination points of the least cost paths. The time ordering of point pairs must match the time linking mode if it is unidirectional (i.e., points in older layers cannot be accessed from points in younger layers if the linking mode is forwards in time (`tlink = 1`).
`verbose`	`logical`. Should function progress be reported to the user?

Value

A SpatVector of time-discrete lines representing the least cost paths between each point pair, recording which overall path they belong to ($feature), the layer the path segment occupies ($layer), the costs along each segment ($cost) and their geographic distances ($distance, identical if weights = "gdist").

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

rasts <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
rasts <- link_islands(rasts)
rtd <- build_tardis(rasts)
pts <- rbind(c(-89.78873, -1.420627, 2),
             c(-88.70836, -0.2627832, 2))

rpts <- point_check(rtd, pts)
rlcp <- least_cost(rtd, origin = rpts[1,], dest = rpts[2,])
```

---

link_delaunay *link_delaunay*

Description

Calculate links between boundary cells between isolated regions within a geoglist layer using Delaunay triangulation. Links can be discarded if they fall above a threshold distance.

Usage

```
link_delaunay(geog, max.dist = NULL, replace = FALSE, verbose = TRUE)
```

Arguments

geog	geoglist. The output of rast_to_geoglist().
max.dist	numeric or NULL. If not NULL, then the maximum link length permitted in metres. Only links below this threshold will be returned.
replace	logical. Should any existing links in geog be replaced? Defaults to FALSE, in which case links are appended to any existing ones.
verbose	logical. Should function progress be to the user? This may be useful when dealing with large rasters (high resolution and/or many layers).

Details

Linkage is a key feature of rTARDIS to permit movement otherwise disconnected regions in geographic space. Delaunay triangulation is used to calculate line-of-site linkage between cells at the edges of Voronoi-adjacent islands (see domains()). Links which intersect unmasked regions other than at their start and end points are discarded, as are those which start and terminate on the same region. The result of this procedure is denser network of links compared to those generated by link_mask(), where single links are generated between Voronoi-adjacent islands.

Currently, links are calculated using terra::delaunay(). While terra generally uses spherical geometry, this function appears to be a rare exception where spherical geometry is not yet supported. As such, Delaunay linkage on global scales is approximate, although it still appears to return reasonable results.

Value

The input geoglist with the $links element filled with a SpatVector of lines representing the island links, and recording their start ($srt) and end ($end) cell IDs, the landscape layer to which they belong ($layer) and their lengths in metres ($distance). If no landscape layers contained islands, then $links will remain NULL.

Examples

```
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist with hexagonal resampling and mask the sea
hexes <- rast_to_geoglist(gal[[1]], gal_m[[1]], as.hex = TRUE, hex = 6)
hexes <- link_delaunay(hexes)

# plot the first layer of the geoglist with island links
plot(hexes)
```

---

link_islands *link_islands*

Description

Calculate links between the nearest cells of isolated regions within a geoglist layer, based on their Voronoi adjacency.

Usage

```
link_islands(geog, klink = NULL, replace = FALSE, verbose = TRUE)
```

Arguments

geog	geoglist. The output of rast_to_geoglist().
klink	integer or NULL. The k-nearest Voronoi neighbours to be linked for each island. Defaults to NULL, in which case the full Voronoi neighbourhood of each island will be linked.
replace	logical. Should any existing links in geog be replaced? Defaults to FALSE, in which case links are appended to any existing ones.
verbose	logical. Should function progress be to the user?

Details

Linkage is a key feature of rTARDIS to permit movement otherwise disconnected regions in geographic space. Linkage between islands is based on their Voronoi neighbourhoods (see domains()). This can be thought of as detecting line-of-site between all islands in a layer, then ranking those line-of-site neighbours by the shortest distances required to reach them. Links which intersect other islands or their Voronoi neighbourhoods are considered invalid and discarded internally. The number of links between neighbours is then determined by klink.

For klink = 1, the returned links will form a minimum spanning tree, ensuring that every island can be reached by some route from any other island. This is a conservative method of linkage and islands may instead have many neighbours of similar distances. Setting klink to higher values will therefore create an increasingly densely connected neighbourhood of islands. klink can be set as high as the user likes, but the Voronoi neighbourhood of an island will set the maximum number of links which the function will return.

Currently, Voronoi neighbourhoods are calculated using terra::voronoi(). While terra generally uses spherical geometry, this function appears to be a rare exception where spherical geometry is not yet supported. As such, Voronoi neighbourhoods on global scales are approximate, although they still appear to return reasonable results.

Value

The input geoglist with the $links element filled with a SpatVector of lines representing the island links, and recording their start ($srt) and end ($end) cell IDs, the landscape layer to which they belong ($layer) and their lengths in metres ($distance). If no landscape layers contained islands, then $links will remain NULL.

Examples

```
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
```

```
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist with hexagonal resampling and mask the sea
hexes <- rast_to_geoglist(gal[[1]], gal_m[[1]], as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

# plot geoglist with links
plot(hexes)
```

`map_to_geoglist` *map_to_geoglist*

Description

Convert a vector of values to a `geoglist`, using an existing `geoglist` as a scaffold. This function is primarily intended to support mapping of outputs from functions involving `tardis` objects and their derivatives, unlike `rast_to_geoglist()` which creates their structure from scratch.

Usage

```
map_to_geoglist(geog, values, name = "value")
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`values`	`numeric`. A named vector of numeric values. Missing values like `NA` or `Inf` are permitted, although not necessarily encouraged. Element names must be numeric IDs which correspond to non-masked cells from a single layer in `geog`. Duplicate IDs are not permitted, i.e., only one value to map per cell ID.
`name`	`character`. The field name under which values will be stored.

Value

A single layer `geoglist` with the mapped `values` present in the `name` field.

Examples

```
library(terra)
library(rTARDIS)

# load galapagos dataset
gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# hex gridded tardis
rasts <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
```

```
                          as.hex = TRUE, hex = 6)
rtd <- build_tardis(rasts)

# map two random values to cell
nt <- c(1034, 983)
names(nt) <- c("19", "987")

foo <- map_to_geoglist(rasts, nt)
```

`min_span` *min_span*

Description

Determine the least cost minimum spanning arborescence between sets of points in a spatiotemporal landscape graph. Least cost paths are identified between pairs of points using Djikstra's algorithm, then the minimum spanning arborescence (see details) selected from these sets of pairwise distances. Multiple sets of points may be analysed in one function call.

Usage

```
min_span(
  tardis,
  weights = "gdist",
  points,
  groups = NULL,
  merge = TRUE,
  verbose = TRUE
)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`points`	`SpatVector`. The output of `point_check()`, denoting the points to be linked, and potentially with a grouping variable added
`groups`	`character`. `NULL` by default, in which case all entries in `points` will be linked in a single arborescence. Otherwise a column name in `points` denoting subsets of points to link. This column can be `character` or `numeric`, but must contain at least two points per group.
`merge`	`logical`. By default, all branches in an arborescence will be collapsed into a single geometry. Set to `FALSE` to keep each branch as as a separate geometry
`verbose`	`logical` Should function progress be reported to the user?

Details

If geographic distances are used (default), then there will be a single least cost path between any two points and the minimum spanning arborescence will be equivalent to a minimum spanning tree. However, as `rTARDIS` graphs are directed, it is possible that the shortest path from A to B may not be the same as the shortest path from B to A depending on the weighting scheme, hence why minimum spanning arborescence is used as the more general case. Note that the time directionality in `tardis` will also affect the minimum spanning arborescence structure.

Value

A `SpatVector` of geometries representing the time-discrete portions of the minimum spanning arborescence(s), recording their overall point group (`$feature`) and time layer (`$layer`) the total cost of each portion (`$cost`), and the total geographic distance summed in each portion (`$distance`, identical if `weights = "gdist"`). If `merge = FALSE`, then individual branches are returned, rather than layer-wise branch aggregates.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)

org <- rbind(c(-89.78873, -1.420627, 2),
             c(-89.58525, -1.473917, 2),
             c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

hpts <- point_check(htd, org)
hmca <- min_span(htd, points = hpts)

oldpar <- par(no.readonly = TRUE)
plot(hexes, par.reset = FALSE)
plot(hmca, add = TRUE, col = "gold", lwd = 2)
par(oldpar)
```

`optim_route` *optim_route*

Description

Determine the optimal sequence in which to visit a set of points (the travelling salesman problem). Least cost paths are identified between all pairs of input points using Djikstra's algorithm, then the optimal route identified from these sets of pairwise distances. Multiple sets of points may be analysed in one function call.

Usage

```
optim_route(
  tardis,
  weights = "gdist",
  points,
  groups = NULL,
  loop = FALSE,
  verbose = TRUE
)
```

Arguments

tardis	tardis. The output of build_tardis().
weights	character. The name of the weighting scheme in tardis to use for distance calculation. By default these are true geographic distances ("gdist"). Alternatively, the name of another weighting scheme added to tardis using weight_tardis().
points	SpatVector. The output of point_check(), denoting the points to be linked.
groups	character. NULL by default, in which case all entries in points will be linked in a single route. Otherwise a column name in points denoting subsets of points to link. This column can be character or numeric, but must contain at least two points per group.
loop	logical. Should the optimal route additionally be closed from end point to start point to return a polygon? Defaults to FALSE.
verbose	logical. Should function progress be reported to the user?

Value

A SpatVector of time-discrete lines representing the optimal route for each point group ($group), recording which overall path they belong to ($feature), the layer the path segment occupies ($layer), the costs along each segment ($cost) and their geographic distances ($distance, identical if weights = "gdist").

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3, byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
```

```
                            as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)

org <- rbind(c(-89.78873, -1.420627, 2),
             c(-89.58525, -1.473917, 2),
             c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

hpts <- point_check(htd, org)
hopt <- optim_route(htd, points = hpts)

oldpar <- par(no.readonly = TRUE)
plot(hexes, par.reset = FALSE)
plot(hopt, add = TRUE)
par(oldpar)
```

---

plot *plot.geoglist*

Description

Plotting method for a `geoglist`. If the `geoglist` contains multiple layers, then the default behaviour is to plot the first one.

Usage

```
## S3 method for class 'geoglist'
plot(
  x,
  y = 1,
  pal = sf.colors(10),
  links = TRUE,
  lcol = "grey",
  lwd = 1,
  lty = 1,
  hex.border = NA,
  legend = TRUE,
  axes = TRUE,
  bg = NA,
  add = FALSE,
  xlim = NULL,
  ylim = NULL,
  ...
)
```

Arguments

x	geoglist. The output of rast_to_geoglist().
y	numeric. The layer in x to be plotted, along with its links. Defaults to 1 (the first layer).
pal	vector. A vector of colours to be used for plotting layer values, such as those returned by an R colour palette. The length of this colour palette will also correspond to the number of breaks used when plotting, if x contains continuous values.
links	logical. Should mask links be plotted, if available?
lcol	character or integer. The colour to be used for plotting links.
lwd	integer. The line width to be used for plotting links.
lty	integer. The line type to be used for plotting links.
hex.border	character or integer. The colour to be used for plotting hexagonal grids. By default none.
legend	logical. Should a legend be added to the plot? Defaults to TRUE. Legends are set to appear on the right hand side of the plot.
axes	logical. Should axes be added to the plot? These will look sensible for lon-lat geoglists, but may look odd for other projection systems.
bg	character or integer. The colour to use for the map background. NA (no colour) by default.
add	logical. Should x be added to an existing plot? FALSE by default.
xlim	numeric. If not NULL, then a vector of two numbers to set the minimum and maximum x extent of the plot in terms of the projection system in x.
ylim	numeric. If not NULL, then a vector of two numbers to set the minimum and maximum y extent of the plot in terms of the projection system in x.
...	Other arguments passed to terra::plot(). Note that some of these have default settings already specified in this function, e.g., add = FALSE, but can be safely changed if desired.

Value

No return value.

Examples

```
library(rTARDIS)
library(terra)

gal <- cretaceous()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)
rasts <- rast_to_geoglist(gal, gal_m, times = c(112.5, 110, 107.5))
rasts <- link_islands(rasts, klink = 1)

plot(rasts)
```

plot_globe *plot_globe*

Description

Plot a geoglist on a 3D interactive globe. A geoglist of any spatial scale can be plotted, but the function is primarily intended for those which cover the total surface of the Earth or at least a decent portion of one of its hemispheres.

Usage

```
plot_globe(
  x,
  y = 1,
  range = NULL,
  pal = sf.colors(10),
  links = TRUE,
  lcol = "steelblue",
  lwd = 1,
  lty = 1,
  bg = "aliceblue",
  graticule = TRUE,
  grat.col = "grey95",
  add = FALSE,
  ...
)
```

Arguments

x	geoglist. The output of rast_to_geoglist().
y	numeric. The layer in the x to be plotted, along with its links. Defaults to 1 (the first layer).
range	vector. If not NULL, the minimum and maximum values to be plotted.
pal	vector. A vector of colours to be used for plotting layer values, such as those returned by an R colour palette. The length of this colour palette will also correspond to the number of breaks used when plotting, if x contains continuous values.
links	logical. Should mask links be plotted, if available?
lcol	character or integer. The colour to be used for plotting links.
lwd	integer. The line width to be used for plotting links.
lty	integer. The line type to be used for plotting links.
bg	character or integer. The colour to use for the globe background.
graticule	logical. Should a lon-lat graticule be added to the sphere? Defaults to TRUE.
grat.col	character or integer. The colour to be used for the graticule lines.

add	logical. Should x be added to an existing rgl window?
...	Other arguments passed to rgl primitive plotting functions. Note that some of these are set internally and so using this argument may cause errors. Defaults to FALSE, which will initiate plotting on a blank sphere in a new rgl window.

Details

Longitude-latitude coordinates are converted to spherical coordinates internally as follows, using an authalic Earth radius of 6371.007 km.

`radius <- 6371.007` `x <- radius * cos(lat) * cos(long)` `y <- radius * cos(lat) * sin(long)` `z <- radius * sin(lat)`

The background sphere is produced using a slightly smaller radius to prevent occlusion and artefacts in features plotted onto its surface.

Value

No return value.

Examples

```
library(terra)
library(rTARDIS)

gal <- cretaceous()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)
rasts <- rast_to_geoglist(gal, gal_m, times = c(112.5, 110, 107.5),
                          as.hex = TRUE, hex = 3)
rasts <- link_islands(rasts, klink = NULL)
plot_globe(rasts, 1)
```

---

point_check *point_check*

Description

Resolve a set of coordinates to accessible cells in a tardis object. Discrepant points are adjusted to their nearest accessible cell based on great circle distance, with the option to cap the distance tolerated. Checks are performed on unique coordinate sets internally for efficiency.

Usage

```
point_check(tardis, points, max.dist = NULL, round = NULL, verbose = TRUE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()` or `weight_tardis()`.
`points`	`matrix` or `data.frame`. A two or three column `matrix` or `data.frame` of coordinates. Column ordering is assumed to be longitude (decimal degrees), latitude (decimal degrees) and time (positive, time before present). The time column is only required if `tardis` contains multiple layers.
`max.dist`	`numeric`. The maximum distance in metres permitted for adjusting points to accessible cells. Points with adjustments above this threshold will be discarded. `NULL` by default, meaning that no points are discarded.
`round`	`integer`. If not `NULL`, then the number of decimal places to which longitudes and latitudes will be rounded. This can speed up function progress when dealing with large numbers of input points and should be selected according to the resolution of the `tardis` grid.
`verbose`	`logical` Should function progress be reported to the user?

Value

A `SpatVector` of points, recording which input point they correspond to (`$feature`), the ID of the `tardis` cell they fall into (`$cell`), the layer to which they belong (`$layer`) and their adjusted distance (`$adj`).

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)

org <- rbind(c(-89.78873, -1.420627, 2),
             c(-89.58525, -1.473917, 2),
             c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

hpts <- point_check(htd, org)
```

project_geoglist *project_geoglist*

Description

Transform a `geoglist` and its links to a coordinate reference system (CRS) other than lon-lat. This may be desirable for plotting purposes, but will render the `geoglist` incompatible with most `rTARDIS` functions.

Usage

```
project_geoglist(geog, crs, ...)
```

Arguments

`geog`	`geoglist`. The output of `rast_to_geoglist()`.
`crs`	`character`. A valid CRS character string to which geog will be transformed.
`...`	Additional arguments passed to `terra::project` to control how the grid is resampled to the new CRS geometry.

Value

`geoglist`. The input `geoglist` transformed to the target CRS.

Examples

```
library(terra)
library(rTARDIS)

gal <- cretaceous()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])
rasts <- link_islands(rasts, klink = 1)

regs <- project_geoglist(geog = rasts, crs = "+proj=eqearth")
plot(regs)
```

rand_walk *rand_walk*

Description

Conduct random walks within a `tardis` graph. Walk lengths can be specified by the number of steps or a target distance to traverse. In the case of the latter, the walk is terminated at the closest distance to the target as the walks will not necessarily be able to precisely hit the target distance. Full traverses in time and space are possible, but as walks in both these dimensions not are not necessarily easy to interpret, so walks are restricted to the time layer in which they originate as a default.

Usage

```
rand_walk(
  tardis,
  weights = "gdist",
  origin,
  mode = "steps",
  rwlen = 1000,
  restrict = TRUE,
  trials = 100,
  verbose = TRUE
)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`origin`	`SpatVector`. The output of `point_check()`, denoting the starting points for the random walk(s).
`mode`	`character`. The nature of the random walk, either the number of `"steps"` to take or the `"cost"` in terms of the cumulative weight of the edges traversed.
`rwlen`	`numeric`. The length of the random walk as selected by `mode`. Either a single number or a vector with as many elements as origin points, enabling different walk lengths for each point. When `weights = NULL` and `mode = "cost"`, this can be directly interpreted as metres covered by the random walk, but may also be expressed in relation to a user-supplied weighting scheme.
`restrict`	`logical`. Should the random walk should be constrained to the same time layer as its origin. `TRUE` by default.
`trials`	`integer`. The number of random walk trials to conduct before quitting, if `mode = "cost"` and the random walk length proves insufficient to reach the cost value.
`verbose`	`logical`. Should function progress be reported to the user?

Value

If `restrict = TRUE` (default), then a `SpatRaster` with each layer recording the frequencies of cell visits. If `restrict = FALSE`, a list of `SpatRaster` objects, each recording the frequencies of cells visited by each random walk in each landscape layer through time.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)
org <- rbind(c(-89, -1.05, 2))
pts <- point_check(htd, org)

foo <- rand_walk(htd, origin = pts, rwlen = 1e4)
```

---

rast_to_geoglist *rast_to_geoglist*

Description

Convert a set of rasters to an S3 class `geoglist` object compatible with downstream `rTARDIS` functions. Typically, these rasters will record topography and/or bathymetry measured in metres, but could record other geographic properties instead. If the rasters represent the same area through time, then the first raster in the set must be the oldest.

Usage

```
rast_to_geoglist(
  geog,
  mask = NULL,
  times = NULL,
  as.hex = FALSE,
  hex = "auto",
  method = "mean",
  verbose = TRUE,
  ...
)
```

Arguments

geog	SpatRaster. A set of geographic rasters. These must be in longitude-latitude projection. Missing values are not permitted and should be replaced (e.g. with the average of the surrounding cells, or a dummy value).
mask	SpatRaster or NULL. If not NULL, this will be used to designate non-accessible areas in geog. It must be fully contiguous with geog (i.e., share the same resolution, extent and number of layers) and contain only 1 (non-masked, accessible) or NA (masked, non-accessible) values.
times	numeric or NULL. If the layers do not relate to each other, temporally, or only a single raster layer is present, then times can be left as NULL. Otherwise a numeric vector with nlayers(geog) + 1 positive elements, with successive pairs expressing the temporal extent of each layer as time in the past, such that: t(n) > x >= t(n + 1). For temporal ordering, the vector need not end in the present (i.e. 0), but time must flow from oldest to youngest.
as.hex	logical. Should geog be resampled to the hexagonal grid system defined by Uber's H3 library? Defaults to FALSE.
hex	"auto" or integer. The desired H3 resolution to be used for resampling rasters. Defaults to "auto", resulting in the function selecting the H3 resolution closest to the resolution of geog. Otherwise an integer in the range 1 - 15.
method	character. The function to be used for resampling the raster grid. This must be compatible with exactextractr::exact_extract(). Defaults to "mean".
verbose	logical. Should function progress be reported to the user?
...	Additional arguments passed internally to exactextractr::exact_extract() for resampling of raster grids.

Details

Masking is a key feature of rTARDIS. Besides providing more realistic representation of accessible geographic space, it can dramatically reduce the number of cells and so memory required for landscape representation, improving computational efficiency.

Resampling to a hexagonal grid is implemented using exactextractr::exact_extract() with weighting by cell area, as this is currently much faster compared to the weighted resampling in the terra R package. Weighting is not meaningful for all functions, e.g., "max".

Resampling to a hexagonal grid may be desirable when working with landscapes approaching global extents in reduce the number of cells required to represent polar latitudes, again helping to improve computational efficiency. The trade off is that landscape features may be altered or lost depending on the grid resolution used, although this a risk of any resampling procedure. In addition, a large rectangular grid layer may be noticeably faster to plot than a hexagonal grid of similar extent and resolution due to their handling by terra, affecting performance in some downstream functions.

Value

A geoglist with four list elements. $gdat records spatial properties of the input rasters used throughout downstream rTARDIS functions. $tdat records the temporal extent of each layer, or is NULL if this information was omitted in the function run. $layers is a set of geographic layers, either as a standard SpatRaster, or a SpatVectorCollection of hexagonal polygons if resampling

was implemented. $links is a NULL placeholder slot for storing the output of downstream linking functions.

Examples

```
# load libraries
library(terra)
library(rTARDIS)

# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist from a single raster layer
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])

# create a multi-layer geoglist with hexagonal resampling
hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
```

---

read_geoglist *read_geoglist*

Description

Read a geoglist from a file produced by save_geoglist(). These functions may be convenient to for storing and retrieving large geoglist objects which took significant time to create.

Usage

```
read_geoglist(path)
```

Arguments

path	character. A character string specifying the path to the geoglist file. This should end in ".RData".

Value

A geoglist, reconstructed from the data saved in path.

regions *regions*

Description

Classify all cells in the layers of a geoglist by which region they belong to. Optionally, only the boundary cells of these regions may be returned.

Usage

```
regions(geog, bounds.only = TRUE, use.links = FALSE)
```

Arguments

geog	geoglist. The output of rast_to_geoglist().
bounds.only	logical. Should only the boundary cells of regions be returned? Defaults to TRUE.
use.links	logical. Should links in geog be accounted for when determining cell connectivity and so region membership? Defaults to TRUE. This argument will have no effect if geog does not contain any links.

Value

A geoglist recording the region affinity of the cells in each layer.

Examples

```
library(terra)
library(rTARDIS)

gal <- cretaceous()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])

regs <- regions(rasts)
plot(regs)
```

rotation_path *rotation_path*

Description

For a given starting cell and destination layer, get its position in all intervening layers. Points can be shifted forwards or backwards in time regardless of the time directionality in the `tardis` graph, but will only produce meaningful results when locations are not homologous between layers (i.e., where locations shift due to plate tectonic rotation or some other process). If the cell location is lost in an intervening layer, the path will terminate at its last available location, rather than in the destination layer.

Usage

```
rotation_path(tardis, points, layer, time = NULL, as.lines = FALSE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`points`	`numeric` or `SpatVector`. Either a numeric vector of cell IDs a `SpatVector` of points with a cell ID variable named 'cell'.
`layer`	`numeric`. The layer number to shift points to. If a single number, then all points (regardless of their starting layer) will be shifted to that layer. Alternatively a vector of numbers with as many elements as points to permit different destination layers for each point.
`time`	`numeric`. Instead of a layer number, numeric ages can be supplied. These are resolved to layer numbers internally, then treated like the `layer` argument.
`as.lines`	`logical`. Should the point-to-point shifts in each layer be returned as line objects? These may be convenient for plotting purposes. `FALSE` by default.

Value

A `SpatVector` of points or lines. Both record the input point from which they are derived (`$feature`). For points, the corresponding cell ID (`$cell`) and graph layer (`$layer`) are recorded for each rowwise shift. For lines, their start cell (`$from`), end (`$to`) cell, start layer (`$srt`) and end layer (`$end`) are recorded instead.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
```

```
                            as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)
org <- rbind(c(-89.78873, -1.420627, 2.2),
             c(-89.58525, -1.473917, 2.2))
hpts <- point_check(htd, org)

hpts2 <- rotation_path(htd, hpts, time = 1.8)
```

---

save_geoglist *save_geoglist*

Description

Save a `geoglist` in R's .RData format. As the raster or vector components of a `geoglist` cannot be saved directly, this function handles restructuring of that data to a format retrievable by the complementary function `read_geoglist()`. These functions may be convenient to for storing and retrieving large `geoglist` objects which took significant time to create.

Usage

```
save_geoglist(geog, path)
```

Arguments

`geog`	`geoglist`. A `geoglist` produced by `rast_to_geoglist()`, optionally with island links.
`path`	`character`. A character string specifying the directory and file name to be used to store or retrieve the `geoglist`. This should end in ".RData"

Value

No return value.

---

slice_geoglist *slice_geoglist*

Description

Subset a `geoglist` to a given set of layers, along with any links if present. If temporal. A discontiguous set of layers can be selected, unlike with `slice_tardis()`, but any temporal information will be discarded if so.

Usage

```
slice_geoglist(geog, times = NULL, layers = NULL)
```

Arguments

geog	geoglist. The output of rast_to_geoglist().
times	numeric. A vector of numbers denoting the desired times of layers to subset from geog. One of times or layers must be specified. Times corresponding to duplicated layer numbers will be ignored.
layers	numeric. A vector of numbers denoting the layers to subset from geog. One of times or layers must be specified. Duplicated layer numbers will be ignored.

Value

A geoglist of the requested subset of layers.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)
rasts <- rast_to_geoglist(gal, gal_m, times = c(seq(2.25, 0, -0.5), 0))
rasts1 <- slice_geoglist(rasts, layers = 1)
rasts2 <- slice_geoglist(rasts, layers = c(1, 3))
```

---

slice_tardis *slice_tardis*

Description

Subset a tardis graph to a contiguous range of layers. Unlike with slice_geoglist(), the range must be contiguous to retain full temporal connectivity between layers.

Usage

```
slice_tardis(tardis, times = NULL, layers = NULL, adjust.id = TRUE)
```

Arguments

tardis	tardis. The output of build_tardis().
times	numeric. A single positive number giving the age of the layer to subset from tardis, or vector of two positive numbers denoting layer age range to subset. One of times or layers must be specified.

`layers`	`numeric`. A single positive integer giving the layer to subset from `tardis` or a vector of two positive integers denoting the layer range to subset. One of `times` or `layers` must be specified.
`adjust.id`	`logical`. Should cell IDs be adjusted so that they correspond to the new layer numbering of the subsetted `tardis` object? Defaults to `TRUE`. Otherwise, this argument may be useful when working with single layers from a `tardis` object, but where retaining original spatiotemporal indexing is desirable.

Details

This function was developed for instances where successive analyses do not require the entire `tardis` graph, in which case it is more efficient to weight and analyse subsets, rather than operate on the entire graph or create subsets from scratch. Note that layers are counted in decreasing age order, so the oldest time layer will be 1 and so forth.

When adjusting cell IDs (default behaviour), all cells in the first selected layer will be adjusted to the range `1:tardis$gdat["ncell"]`. Cells in the second layer start from `tardis$gdat["ncell"] + 1` and so forth.

Value

A `tardis` graph of the requested subset of layers.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

rasts <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75))
rtd <- build_tardis(rasts)

gts <- slice_tardis(rtd, times = c(2.2, 1.8))
gts <- slice_tardis(rtd, layers = 2)
```

`tardis_to_sparse` *tardis_to_sparse*

Description

Create either a sparse weighted adjacency or transition probability matrix from a layer of a `tardis` graph.

Usage

```
tardis_to_sparse(
  tardis,
  weights = "gdist",
  mode = "adjacency",
  raw.weight = TRUE,
  jump.dir = "auto"
)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`. This can only contain a single time slice, so you may need to use `slice_tardis()` first.
`weights`	`character`. The name of the weighting scheme in `tardis` to use for distance calculation. By default these are true geographic distances (`"gdist"`). Alternatively, the name of another weighting scheme added to `tardis` using `weight_tardis()`.
`mode`	`character`. One of `"adjacency"` or `"transition"`.
`raw.weight`	`logical`. If creating a transition matrix, should island link probabilities be calculated from their raw weights (default) or instead adjusted to approximate a jump-diffusion process (see details).
`jump.dir`	`numeric` or `"auto"`. The stepwise cardinality if approximating jump-diffusion probabilities. By default `"auto"`, in which case the cardinality is selected based on the `tardis` landscape type. Otherwise a suitable integer. Only relevant when `mode = "transition` and `raw.weight = F` (see details for guidance).

Details

This function produces a matrix representation of the edgelist in `tardis`, hence why it is sparse (see `get_cost()` for creating dense matrices). For the adjacency matrix, the weights in the desired weighting scheme are taken directly.

For the probability matrix, by default, the reciprocals of weights are normalised so that they sum to one. As such, a huge weight for one edge versus a tiny weight for another, unrelated edge could end up with similar proportional probabilities. This behaviour is important for island links, which will naturally be longer than links between adjacent cells.

Island link probabilities will be proportionally smaller than other transition probabilities from their starting cell, but can often display higher probabilities for that single jump due to stepwise multiplication of probabilities along alternative routes through unmasked space. This can be accounted for by setting `jump = T`. Instead of using the proportional weight directly, that weight is instead expressed as a multiple of the mean weights for non-jump links from the focal cell (e.g., a link is 8.3 times the mean non-link weight). A naive transition probability is then raised to the power of this multiplier (m). By default this transition probability is selected based on the structure of `tardis`. For a `tardis` built from a rectangular raster grid, each grid cell has 8 neighbours, so the naive cost of moving in any direction is 1/8 (= 0.125), raised to the power of m equivalent steps along the link. For a hexagonally resampled `tardis`, unmasked cells have 6 neighbours. In this way, movement along the link is approximated as though it were a series of direct steps through continuous space, where other movement directions are probable but unrealised.

8 and 6 are sensible defaults for rectangular and hexagonal grids respectively, but other values may be justified. For example `jump.dir = 2` would approximate a scenario where a particle may move forwards or backwards along a link with equal probability, but not deviate from the line of the link itself, unlike with higher cardinalities.

Value

`matrix` A `Matrix::sparseMatrix` adjacency or transition probability matrix.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1]], gal_m[[1]], as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)
tr <- tardis_to_sparse(htd, mode = "transition")
aj <- tardis_to_sparse(htd, mode = "adjacency")
```

`time_shift` *time_shift*

Description

Take a set of points in a `tardis` graph and shift them to other time layers, subject to available inter-layer connections. Points can be shifted forwards or backwards in time regardless of the time directionality in the `tardis` graph. Points without cells in their target layer will return `NA`.

Usage

```
time_shift(tardis, points, layer, time = NULL)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`points`	`numeric` or `SpatVector`. Either a numeric vector of cell IDs or a `SpatVector` of points with a cell ID variable named 'cell'.
`layer`	`numeric`. The layer number to shift points to. If a single number, then all points (regardless of their starting layer) will be shifted to that layer. Alternatively a vector of numbers with as many elements as points to permit different destination layers for each point.
`time`	`numeric`. Instead of a layer number, numeric ages can be supplied. These are resolved to layer numbers internally, then treated like the `layer` argument.

Value

If `points` is a vector, then a vector of cell IDs, with `NA` entries for points without valid starting or ending locations. If `points` is a `SpatVector` then a `SpatVector` of updated point geometries, with cell ID (`$cell`) and time layer fields (`$layer`). For `NA` cells, a blank geometry is returned.

Examples

```
library(terra)
library(rTARDIS)

gal <- galapagos()
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

hexes <- rast_to_geoglist(gal[[1:2]], gal_m[[1:2]], times = c(2.25, 2, 1.75),
                          as.hex = TRUE, hex = 6)
hexes <- link_islands(hexes)

htd <- build_tardis(hexes)
org <- rbind(c(-89.78873, -1.420627, 2),
             c(-89.58525, -1.473917, 2),
             c(-88.70836, -0.2627832, 2),
             c(-90.44276,  0.2943382, 2))

hpts <- point_check(htd, org)

hpts2 <- time_shift(htd, hpts, time = 1.8)
hpts2 <- time_shift(htd, hpts, layer = 2)
```

---

weight_tardis *weight_tardis*

Description

Generate a custom weighting scheme for a `tardis` graph This may be based on the properties in `tardis` itself or derived from `geoglist` objects recording alternative properties of the landscape.

Usage

```
weight_tardis(tardis, name, vars = NULL, wfun, verbose = TRUE)
```

Arguments

`tardis`	`tardis`. The output of `build_tardis()`.
`name`	`character`. The name for weighting scheme to be generated. The names `"cell"`, `"type"`, `"layer"`, `"bearing"`, `"hdist"`, `"vdist"` and `"gdist"` are reserved for internal use.

vars	list. A named list of geoglist objects, each recording an alternative geographic property of the landscape represented in tardis. As such, these must bear the same resolution, extent, number of layers and layer order as the geoglist used to create tardis. The list names "cell", "type", "layer", "bearing", "hdist", "vdist" and "gdist" are reserved for internal use.
wfun	function(origin, dest) A function to calculate the cost of traversal for the edges in each graph layer. See @details for the required function signature.
verbose	logical. Should function progress be to the user?

Details

The weight_tardis function is heavily inspired by the weighting function used in the gen3sis R package by Oskar Hagen. This function only affects weights for links within landscape layers. Temporal link weights are always zero and cannot be altered using this function.

Internally, weight_tardis() generates two data.frames, origin and dest. These minimally record the core properties for each pair of origin and destination cells comprising the edges in a graph layer: the cell ID ($cell), the type of edge it forms ($type), its time layer ($layer), the bearing ($bearing) and horizontal distance to its partner cell in degrees and metresrespectively, the vertical distance ($vdist) to its partner cell in metres, and the total geographic distance ($gdist) based on those horizontal and vertical values. Horizontal distance will be the same in origin and dest, but the vertical distance will be positive or negative depending on whether elevation is gained or lost along an edge. If vars is supplied, then subsequent columns in origin and dest record the cell characteristics in each geoglist in vars.

The core of the weighting function can have as many steps as the user likes, but must consist of vectorised calculations that call on the column names in origin and dest. The argument default exemplifies this and will return identical geographic distances to those in within x, using Pythagoras's theorem on the horizontal (origin$hdist) and vertical (origin$vdist) intercell distances. The user function can use any combination of origin and dest columns in any order. Weights are iteratively calculated for each landscape layer in tardis, with landscape layer column enabling weighting rules to vary through time. Similarly, the $type can be used to weight mask links differently to links in unmasked space.

Crucially, all returned weights should be finite and >= 0, as negative weights are not permitted for downstream functions. NA values are permitted to allow the designation of impermeable edges (for example to restrict movement above a certain threshold cost). Such values, however, may introduce inaccessible islands into a landscape even if island linkage has already been performed.

Value

The input tardis object with the new weighting scheme added to tardis$edges under the column name given in weights.

Examples

```
library(terra)
library(rTARDIS)

# defaults for CRAN checks (safely ignore)
origin <- dest <- NULL
```

```
# load a dataset of the Galapagos archipelago through geological time
gal <- galapagos()
gal <- crop(gal, ext(-92, -88, -2, 1))

# create a land-sea mask from the archipelago raster set
gal_m <- classify(gal, matrix(c(-Inf, 0, NA, 0, Inf, 1), ncol = 3,
                  byrow = TRUE), right = FALSE)

# create a geoglist, masking the sea
rasts <- rast_to_geoglist(gal[[1]], gal_m[[1]])
rasts <- link_islands(rasts)
rtd <- build_tardis(rasts)

# define a function where all geographic distances are doubled
times_ten <- function(origin, dest) {
  (origin$hdist^2 + abs(origin$vdist)^2) * 2
}

# add the weighting scheme
rtd <- weight_tardis(rtd, name = "tentimes", wfun = times_ten)


# define a function where mask links are upweighed by a factor of 100
high_mask <- function(origin, dest) {

  # calculate true geographic distance
  gdist <- sqrt(origin$hdist^2 + abs(origin$vdist)^2)

  # get the indices of the mask links
  mlink <- which(origin$type == 1)

  # multiply mask link costs by 100
  gdist[mlink] <- gdist[mlink] * 100

  # return costs
  return(gdist)
}

# add the weighting scheme
rtd <- weight_tardis(rtd, name = "highmask", wfun = high_mask)


# create a dummy alternative raster dataset with random values
gal_v <- rast(gal_m, vals = abs(rnorm(ncell(gal_m) * nlyr(gal_m))))

# convert the dummy data to a geoglist with the same masking
gvars <- rast_to_geoglist(gal_v, gal_m)

# place the alternative geoglist data into a list
vrs <- list(rand = gvars)
```

```
# define a function which uses the means of the alternative data
alt_func <- function(origin, dest) {
  (origin$rand + dest$rand) / 2
}

# add the weighting scheme
rtd <- weight_tardis(rtd, name = "altfunc", vars = vrs, wfun = alt_func)
```

Index

∗ **datasets**
chelonoides, 5

build_tardis, 3

cell_to_coords, 4
chelonoides, 5
click_iso, 6
click_lcp, 7
click_msa, 8
click_optim, 9
click_points, 11
click_to_link, 12
click_to_snip, 13
cost_surface, 14
cretaceous, 15
cumulative_cost, 15

domains, 17

extract_from_geoglist, 18

galapagos, 19
get_cost, 20
get_grid, 21
get_rotations, 22

instantiate_tardis, 23
isochrone, 24

least_cost, 25
link_delaunay, 26
link_islands, 27

map_to_geoglist, 29
min_span, 30

optim_route, 31

plot, 33
plot_globe, 35
point_check, 36
project_geoglist, 38

rand_walk, 39
rast_to_geoglist, 40
read_geoglist, 42
regions, 43
rotation_path, 44

save_geoglist, 45
slice_geoglist, 45
slice_tardis, 46

tardis_to_sparse, 47
time_shift, 49

weight_tardis, 50