

Package: corridoR (via r-universe)

October 22, 2026

Title Migration Corridors and Genetic Range Shifts from Landscape Genetic Data

Version 0.2.0

Description Models genetic connectivity along migration corridors and forecasts how it shifts under climate change, following Maier et al. (2022) <[doi:10.1038/s41437-022-00561-x](https://doi.org/10.1038/s41437-022-00561-x)>. Least cost corridors are built from accumulated cost surfaces and used as weights to summarize raw environmental layers between sites; each group of environmental features can choose its own corridor bandwidth; and pairwise predictions of differentiation and directional migration are mapped back onto the landscape as magnitude and net-direction surfaces. Includes readers for common genotype formats (STRUCTURE, GENEPOP, VCF, GenAlEx, FSTAT and PLINK files, and 'adegenet' objects), pairwise fixation indices, and the directional differentiation of Sundqvist et al. (2016) <[doi:10.1002/ece3.2096](https://doi.org/10.1002/ece3.2096)>.

License MIT + file LICENSE

URL <https://github.com/paulmaier/corridoR>,
<https://paulmaier.github.io/corridoR/>

BugReports <https://github.com/paulmaier/corridoR/issues>

Depends R (>= 4.1)

Imports gdistance, graphics, grDevices, raster, sf, stats, terra,
utils

Suggests adegenet, caret, Cubist, exactextractr, ggplot2, knitr, lme4,
ranger, rmarkdown, testthat (>= 3.0.0), vcfR

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Paul A. Maier [aut, cre, cph] (ORCID:
 <<https://orcid.org/0000-0003-0851-8827>>)
Maintainer Paul A. Maier <maierpa@gmail.com>
Repository <https://cran.r-universe.dev>
Date/Publication 2026-10-22 19:21:23 UTC
RemoteUrl <https://github.com/cran/corridoR>
RemoteRef HEAD
RemoteSha b4e9f3a51275c409ad543989ae74432a6139e97c

Contents

accumulated_cost	2
as_popcounts	3
both_directions	4
corridor_example	5
corridor_extract	6
directional_gst	7
fit_connectivity	8
lcc_weights	9
least_cost_paths	10
lineage_cross	11
make_transition	11
map_pairwise	12
pair_weights	13
pairwise_fst	14
pairwise_table	15
path_buffers	15
plot_shift	16
predict.corridor_model	17
rank_resistance	18
read_genotypes	19
select_bandwidth	21
shift_arrows	22
site_contrast	23
Index	25

accumulated_cost	<i>Accumulated cost surfaces</i>
------------------	----------------------------------

Description

The accumulated least cost from each site to every cell of the landscape. Corridors between two sites are built from the sum of their surfaces, so this is computed once per site and reused for every pair.

Usage

```
accumulated_cost(tr, sites, id = "site", filename = "")
```

Arguments

tr	A transition layer from <code>make_transition()</code> .
sites	An sf object of points or polygons (polygon centroids are used) with an ID column.
id	Name of the ID column in sites.
filename	Optional GeoTIFF to write the surfaces to, which keeps memory use low for large landscapes.

Value

A multi-layer SpatRaster with one layer per site, named by site ID. Unreachable cells are NA.

Examples

```
ex <- corridor_example()
tr <- make_transition(ex$resistance)
acc <- accumulated_cost(tr, ex$sites[1:3, ])
```

as_popcounts	<i>Population allele counts</i>
--------------	---------------------------------

Description

Every genetic statistic in corridoR works from allele counts per population and locus. A popcounts object holds them: a list with one integer matrix per locus (populations in rows, alleles in columns), plus the population names and the number of individuals sampled in each.

Usage

```
as_popcounts(x, ...)

## S3 method for class 'popcounts'
as_popcounts(x, ...)

## S3 method for class 'genind'
as_popcounts(x, pop = NULL, ...)

## S3 method for class 'genpop'
as_popcounts(x, ...)

## S3 method for class 'genlight'
```

```
as_popcounts(x, pop = NULL, ...)

## S3 method for class 'vcfR'
as_popcounts(x, pop = NULL, ...)
```

Arguments

x	An object to convert.
...	Not used.
pop	Population of each individual, for objects that do not carry it (a vcfR object, or a genlight without pop). See read_genotypes() for the accepted forms.

Details

`as_popcounts()` converts objects from other packages: `genind`, `genpop` and `genlight` from **adegenet** and `vcfR` from **vcfR**. To read a file directly, use [read_genotypes\(\)](#).

Value

An object of class `popcounts`.

Examples

```
ex <- corridor_example()
pc <- read_genotypes(ex$files$small)
pc
```

both_directions	<i>Use each pair in both directions</i>
-----------------	---

Description

Corridor features are the same whichever site is the source, so they are measured once per unordered pair. Models of directional statistics such as dM need a row for each direction; this repeats every row with from and to swapped.

Usage

```
both_directions(x)
```

Arguments

x	A data.frame with columns from and to.
---	--

Value

x with the reversed pairs appended.

Examples

```
both_directions(data.frame(from = "a", to = "b", snow = 3))
```

corridor_example	<i>Simulated example landscape</i>
------------------	------------------------------------

Description

Loads the example shipped with the package: a simulated 60 x 40 km mountain range modeled loosely on the central Sierra Nevada, with 80 snowmelt meadows in four lineages. A long, gradual western slope rises to a crest near the eastern edge; east of it a steep escarpment lies in a rain shadow; two steep-walled canyons cut the western slope. Migrants leave water-stressed meadows for wetter neighbors. Under warming (+4.2 C) snowpack shrinks by a share that falls with elevation, from about 90% in the foothills to under 20% at the crest, so the push uphill grows most at low elevation. The script that builds the example is in the package source (`data-raw/make_example.R`).

Usage

```
corridor_example(acc = FALSE)
```

Arguments

`acc` Also compute the accumulated cost surfaces.

Value

A list with

`dem, resistance` Elevation (m) and movement resistance `SpatRasters`; ridgelines have resistance `1e6`.

`env, env_future` Environmental layers now and under warming: snowpack (mm), runoff (mm), a meadow moisture index (log scale), summer temperature (C), slope (degrees) and forest cover (0-1).

`sites` The 80 meadows as `sf` points with site, elevation and lineage.

`lineage_tmrca` Time since each pair of lineages split (thousand years), for `lineage_cross()`.

`resistance_slope, resistance_cover` The two components blended 80:20 into resistance, for comparing path hypotheses.

`acc` Accumulated cost surfaces for every meadow, from `accumulated_cost()`. Only with `acc = TRUE`; they take a few seconds to compute and are cached for the rest of the session.

`files` Paths to the genotypes (about 1,300 SNPs, 8 individuals per meadow) in `STRUCTURE` (`structure`), `GENEPOP` (`genepop`) and `VCF` (`vcf`) format, the `VCF` population map (`popmap`), and a small `GENEPOP` sample of six meadows and 100 SNPs (`small`) for quick tests.

Examples

```
ex <- corridor_example()
terra::plot(ex$dem, col = hcl.colors(100, "viridis"))
plot(sf::st_geometry(ex$sites), add = TRUE, pch = 19)
```

corridor_extract	<i>Summarize environmental layers within corridors</i>
------------------	--

Description

For every pair of sites, summarizes each environmental layer inside the pair's corridor. With least cost corridors (acc and q), each cell counts in proportion to its corridor weight, so likely routes matter more than marginal ones. With path buffers, cells are weighted by the share of the cell inside the buffer.

Usage

```
corridor_extract(
  env,
  pairs,
  acc = NULL,
  q = 0.05,
  buffers = NULL,
  stat = "mean",
  progress = interactive()
)
```

Arguments

env	A <code>SpatRaster</code> of environmental layers. Layers are resampled to the corridor grid as needed (bilinear for continuous layers).
pairs	<code>Data.frame</code> with columns <code>from</code> and <code>to</code> .
acc, q	Accumulated cost surfaces from <code>accumulated_cost()</code> and the corridor threshold for <code>lcc_weights()</code> . Use these for least cost corridors.
buffers	Alternatively, an <code>sf</code> layer of buffered paths from <code>path_buffers()</code> with columns <code>from</code> and <code>to</code> .
stat	How to summarize each layer: "mean" (weighted mean, the default) or "sum" (weighted sum, for counts such as stream or trail crossings). A named vector sets the statistic per layer.
progress	Show a progress bar.

Value

A `data.frame` with `from`, `to` and one column per layer of `env`.

Examples

```
ex <- corridor_example()
tr <- make_transition(ex$resistance)
pr <- data.frame(from = c("M01", "M02"), to = c("M12", "M05"))
acc <- accumulated_cost(tr, ex$sites[ex$sites$site %in% unlist(pr), ])
```

```
corridor_extract(ex$env, pr, acc = acc, q = 0.05)
```

directional_gst	<i>Directional genetic differentiation and net migration (dM)</i>
-----------------	---

Description

Computes the directional GST of Sundqvist et al. (2016), as in `diversity::divMigrate(stat = "gst")`: for each pair of populations, a hypothetical shared gene pool is built from the normalized geometric mean of their allele frequencies, and each population's differentiation from that pool is converted to relative migration. Following Maier et al. (2022), the difference between emigration and immigration gives net migration, dM.

Usage

```
directional_gst(x)
```

Arguments

`x` A [popcounts](#) object, or anything [as_popcounts\(\)](#) accepts.

Value

A list with

`relative` Relative migration, scaled so the largest value is 1. `relative[i, j]` is migration from population `i` into population `j`.

`dM` `relative - t(relative)`: positive when `i` sends more migrants to `j` than it receives from it.

`gst` The directional GST matrix behind `relative`.

References

Sundqvist L, Keenan K, Zackrisson M, Prodohl P, Kleinhans D (2016) Directional genetic differentiation and relative migration. *Ecology and Evolution* 6:3461-3475.

Maier PA, Vandergast AG, Ostoja SM, Aguilar A, Bohonak AJ (2022) Landscape genetics of a sub-alpine toad: climate change predicted to induce upward range shifts via asymmetrical migration corridors. *Heredity* 129:257-272.

Examples

```
ex <- corridor_example()
dg <- directional_gst(read_genotypes(ex$files$small))
round(dg$dM[1:4, 1:4], 3)
```

fit_connectivity	<i>Fit a Cubist model of connectivity</i>
------------------	---

Description

The modeling step of Maier et al. (2022). Features are first reduced by a PCA within each group (groups of more than two features, centered and scaled), which removes most redundancy among related layers. A Cubist model (rule-based trees with a linear model in each leaf, which extrapolates better than random forests) is tuned by cross-validation over committees and neighbors. Its variable importance then decides which feature to keep whenever features from different groups are collinear (VIF above `vif_threshold`), and the model is refit on what remains.

Usage

```
fit_connectivity(
  data,
  response,
  groups,
  committees = c(1, 10, 50, 75, 100),
  neighbors = c(0, 1, 5, 7, 9),
  folds = 10,
  vif_threshold = 10,
  seed = 12345
)
```

Arguments

<code>data</code>	A data.frame of features, one row per directed pair.
<code>response</code>	Numeric response (FST, dM, or another pairwise statistic).
<code>groups</code>	Named list mapping group names to columns of data. Columns not in any group (for example path length or a lineage term) are used as they are and always kept.
<code>committees, neighbors</code>	Values to try.
<code>folds</code>	Number of cross-validation folds. When data has from and to columns, both directions of a pair always fall in the same fold, so a pair is never used to predict its own reverse.
<code>vif_threshold</code>	Collinearity cutoff. Inf skips the VIF step.
<code>seed</code>	Random seed for the folds.

Value

An object of class `corridor_model` with the final Cubist fit, the PCA for each group, the features kept, the cross-validation results and predictions, and variable importance. Use `stats::predict()` for new or future data.

Examples

```

set.seed(1)
n <- 300
d <- data.frame(snow = rnorm(n), runoff = rnorm(n), temp = rnorm(n), dist = runif(n))
d$runoff <- d$snow + rnorm(n, sd = 0.3)
y <- 0.05 + 0.02 * d$snow + 0.03 * d$dist + rnorm(n, sd = 0.01)
m <- fit_connectivity(d, y, groups = list(climate = c("snow", "runoff", "temp")),
                     committees = c(1, 10), neighbors = c(0, 5), folds = 5)

m
future <- transform(d, snow = snow - 1)
summary(predict(m, future) - predict(m, d))

```

lcc_weights

*Least cost corridor weights for a pair of sites***Description**

Sums the accumulated cost surfaces of the two sites, rescales the total to 0-1, keeps the cheapest q share of cells, and turns the kept values into weights: 1 on the least cost route and 0 at the edge of the corridor. Larger q gives a broader corridor that takes in more alternative routes.

Usage

```
lcc_weights(acc, from, to, q = 0.05)
```

Arguments

acc	Accumulated cost surfaces from accumulated_cost() .
from, to	Site IDs (layer names of acc).
q	Share of cells to keep, for example 0.001 (a narrow corridor) to 0.05 (broad). Several values give one layer each.

Value

A SpatRaster of weights, NA outside the corridor, one layer per value of q .

References

Maier PA et al. (2022) Heredity 129:257-272.

Examples

```

ex <- corridor_example()
tr <- make_transition(ex$resistance)
acc <- accumulated_cost(tr, ex$sites[ex$sites$site %in% c("M01", "M12"), ])
w <- lcc_weights(acc, "M01", "M12", q = c(0.005, 0.05))
terra::plot(w, col = hcl.colors(100, "viridis"))

```

least_cost_paths	<i>Least cost paths between sites</i>
------------------	---------------------------------------

Description

Least cost paths between sites

Usage

```
least_cost_paths(
  tr,
  sites,
  id = "site",
  pairs = NULL,
  dem = NULL,
  max_length = Inf
)
```

Arguments

<code>tr</code>	A transition layer from make_transition() .
<code>sites</code>	An sf object of points or polygons (polygon centroids are used) with an ID column.
<code>id</code>	Name of the ID column in sites.
<code>pairs</code>	Optional data.frame with columns from and to. By default every unordered pair of sites.
<code>dem</code>	Optional elevation SpatRaster. If given, path length is measured over the terrain surface (3D), as in Maier et al. (2022).
<code>max_length</code>	Drop paths longer than this (meters).

Value

An sf object of lines with columns from, to, euclidean and length.

Examples

```
ex <- corridor_example()
tr <- make_transition(ex$resistance)
paths <- least_cost_paths(tr, ex$sites, pairs = data.frame(from = "M01", to = "M12"),
  dem = ex$dem)
paths
```

lineage_cross	<i>Lineage effect for pairs of sites</i>
---------------	--

Description

Deep phylogeographic splits add genetic differentiation that has nothing to do with today's landscape. Following Maier et al. (2022), the lineage effect of a pair is the time to the most recent common ancestor of the two sites' lineages, and zero when both belong to the same lineage. Adding it as a predictor lets a model account for that history.

Usage

```
lineage_cross(from, to, lineages, tmrca)
```

Arguments

from, to	Site IDs of each pair.
lineages	A data.frame with site IDs in the first column and lineage names in the second.
tmrca	A data.frame with two lineage columns and the time since they split in the third (any units; only relative values matter to the models). Pairs of lineages not listed get NA.

Value

A numeric vector, one value per pair.

Examples

```
ex <- corridor_example()
lin <- sf::st_drop_geometry(ex$sites)[, c("site", "lineage")]
lineage_cross(c("M01", "M01"), c("M02", "M30"), lin, ex$lineage_tmrca)
```

make_transition	<i>Transition layer from a resistance surface</i>
-----------------	---

Description

Builds the **gdistance** transition (conductance) layer used for least cost paths and corridors. Conductance between neighboring cells is the inverse of their mean resistance, corrected for diagonal steps. Cells at or above barrier (for example ridgelines scored as impassable) and NA cells cannot be crossed.

Usage

```
make_transition(resistance, barrier = NULL, directions = 8)
```

Arguments

resistance	A single-layer SpatRaster (or RasterLayer) of resistance values, in a projected coordinate system with units of meters.
barrier	Resistance at or above which a cell is a barrier. NULL for none.
directions	4, 8 or 16 neighbors.

Value

A TransitionLayer.

Examples

```
ex <- corridor_example()
tr <- make_transition(ex$resistance)
```

map_pairwise	<i>Map pairwise values onto the landscape</i>
--------------	---

Description

Spreads one value per pair of sites (a predicted FST, a projected change in dM, and so on) over the pair's least cost corridor, scaled by the corridor weight so that less likely routes count for less, and averages overlapping corridors cell by cell. With `direction = TRUE`, each directed pair also contributes a vector from source to destination, reversed when its value is negative, and the vectors are summed in every cell to give the net direction of change, as in Fig. 1D of Maier et al. (2022).

Usage

```
map_pairwise(
  values,
  acc,
  sites,
  id = "site",
  q = 0.05,
  direction = TRUE,
  vectors = c("value", "sign"),
  absolute = TRUE,
  progress = interactive()
)
```

Arguments

values	A data.frame with columns from, to and value. For directional maps include both directions of each pair.
acc	Accumulated cost surfaces from accumulated_cost() .

sites	An sf object of sites (used for the direction of each pair).
id	Name of the ID column in sites.
q	Corridor threshold passed to <code>lcc_weights()</code> .
direction	Also compute the net direction of change.
vectors	How each pair contributes to the direction. "value" (the default) adds a unit vector from source to destination scaled by the pair's value, so pairs with larger changes pull harder. "sign" adds the source-to-destination displacement scaled only by the sign of the value, as in Maier et al. (2022).
absolute	Map the absolute value (the magnitude of change), as for change maps. Set to FALSE to map signed values.
progress	Show a progress bar.

Value

A SpatRaster with layer value (mean weighted value per cell) and, with direction = TRUE, vx and vy (mean direction vector, map units) and bearing (degrees clockwise from north).

Examples

```
ex <- corridor_example(acc = TRUE)
# a made-up change for a few pairs: more net movement toward the east
v <- both_directions(data.frame(from = c("M03", "M08", "M14"), to = c("M20", "M25", "M28")))
east <- sf::st_coordinates(ex$sites)[match(v$to, ex$sites$site), 1] >
  sf::st_coordinates(ex$sites)[match(v$from, ex$sites$site), 1]
v$value <- ifelse(east, 0.05, -0.05)
m <- map_pairwise(v, ex$acc, ex$sites)
terra::plot(m$value, col = hcl.colors(100, "viridis"))
```

pair_weights	<i>Case weights for pairwise data</i>
--------------	---------------------------------------

Description

Sites that appear in many pairs would otherwise dominate a model of pairwise data. Each pair is weighted by one minus the mean relative frequency of its two sites, then the weights are shifted so the largest is 1 and normalized to sum to 1, as in Maier et al. (2022).

Usage

```
pair_weights(from, to)
```

Arguments

from, to	Site IDs of each pair.
----------	------------------------

Value

Numeric weights, one per pair.

Examples

```
pair_weights(c("a", "a", "a", "b"), c("b", "c", "d", "c"))
```

pairwise_fst	<i>Pairwise FST between populations</i>
--------------	---

Description

Pairwise FST between populations

Usage

```
pairwise_fst(x, method = c("hudson", "nei"))
```

Arguments

x	A popcounts object, or anything as_popcounts() accepts.
method	"hudson" (default) is Hudson's FST as a ratio of averages across loci (Bhatia et al. 2013), which is robust to unequal sample sizes. "nei" is Nei's GST from pooled allele frequencies.

Value

A symmetric matrix of pairwise FST with population names as dimnames. Any precomputed matrix (for example from **hierfstat** or **Stacks**) works just as well in the rest of the package.

References

Bhatia G, Patterson N, Sankararaman S, Price AL (2013) Estimating and interpreting FST: the impact of rare variants. *Genome Research* 23:1514-1521.

Examples

```
ex <- corridor_example()
fst <- pairwise_fst(read_genotypes(ex$files$small))
round(fst[1:4, 1:4], 3)
```

pairwise_table	<i>Pairwise genetic table</i>
----------------	-------------------------------

Description

Turns matrices of FST and dM into a long table with one row per directed pair, the layout used by the modeling functions.

Usage

```
pairwise_table(fst, dM = NULL)
```

Arguments

fst	Symmetric matrix of pairwise differentiation.
dM	Optional matrix of net migration, dM[i, j] from i to j.

Value

A data.frame with columns from, to, fst and (if given) dM.

Examples

```
fst <- matrix(c(0, 0.1, 0.2, 0.1, 0, 0.15, 0.2, 0.15, 0), 3,
              dimnames = list(c("a", "b", "c"), c("a", "b", "c")))
pairwise_table(fst)
```

path_buffers	<i>Buffers around least cost paths</i>
--------------	--

Description

Simple corridor bandwidths: a fixed distance on either side of each least cost path.

Usage

```
path_buffers(paths, width = c(100, 500))
```

Arguments

paths	Least cost paths from least_cost_paths() .
width	Buffer width(s) in meters.

Value

A named list of sf polygon layers, one per width.

Examples

```
ex <- corridor_example()
tr <- make_transition(ex$resistance)
p <- least_cost_paths(tr, ex$sites, pairs = data.frame(from = "M01", to = "M12"))
b <- path_buffers(p, c(100, 500))
```

plot_shift

Plot a map of projected change

Description

Draws a map from `map_pairwise()` over shaded relief. Change is colored on a three-tone scale: cool blue below threshold, a narrow white band at it, and red above it, so the areas under most pressure stand out. With a directional map, arrows from `shift_arrows()` show the net direction.

Usage

```
plot_shift(
  map,
  dem = NULL,
  sites = NULL,
  col_sites = "white",
  threshold = NULL,
  smooth = 5,
  alpha = 0.5,
  arrows = "vx" %in% names(map),
  cells = 14,
  scale = c("linear", "sqrt"),
  main = "",
  legend_title = "|change|"
)
```

Arguments

map	A map from <code>map_pairwise()</code> .
dem	Optional elevation SpatRaster for the relief.
sites	Optional sf points to draw on top.
col_sites	Fill colors for sites (recycled, or one per site).
threshold	Value drawn in white. Defaults to the median of the map.
smooth	Width (in cells) of a moving-average window applied to the displayed surface only; 1 for none.
alpha	Opacity of the change layer.
arrows	Draw arrows (needs a directional map).
cells, scale	Passed to <code>shift_arrows()</code> .
main, legend_title	Title and legend title.

Value

The map, invisibly.

Examples

```
ex <- corridor_example(acc = TRUE)
v <- both_directions(data.frame(from = c("M03", "M08"), to = c("M40", "M45"), value = 0.05))
v$value[3:4] <- -0.05
plot_shift(map_pairwise(v, ex$acc, ex$sites), dem = ex$dem, sites = ex$sites)
```

predict.corridor_model

Predict from a connectivity model

Description

Projects new feature values (for example future climate) onto the PCA loadings fitted to present-day data, so the components keep their meaning, then predicts with the Cubist model.

Usage

```
## S3 method for class 'corridor_model'
predict(object, newdata, ...)
```

Arguments

object	A corridor_model from fit_connectivity() .
newdata	A data.frame with the same feature columns as the data the model was fit to.
...	Not used.

Value

Numeric predictions.

rank_resistance	<i>Rank resistance surfaces by how well they explain genetic distance</i>
-----------------	---

Description

Compares competing resistance surfaces, each a hypothesis about what slows movement, by how well the least cost distance through it explains a pairwise genetic distance such as FST. For every surface, the cost of the cheapest route between each pair of sites is related to the genetic distance with a mixed model that has source and destination site as random effects (Maier et al. 2022). Models are fit by maximum likelihood, so they can be compared by log-likelihood and AIC. Straight-line distance is added as a baseline: a surface that cannot beat it adds nothing.

Usage

```
rank_resistance(
  surfaces,
  sites,
  genetic,
  id = "site",
  pairs = NULL,
  max_dist = Inf,
  barrier = 1e+06,
  straight = TRUE
)
```

Arguments

<code>surfaces</code>	Resistance surfaces to compare: a named list of single-layer <code>SpatRasters</code> , or one multi-layer <code>SpatRaster</code> with a layer per hypothesis. Names label the hypotheses.
<code>sites</code>	An <code>sf</code> object of points or polygons with an ID column.
<code>genetic</code>	A square matrix of pairwise genetic distance (for example from <code>pairwise_fst()</code>) with site IDs as row and column names.
<code>id</code>	Name of the ID column in <code>sites</code> .
<code>pairs</code>	Optional <code>data.frame</code> with columns <code>from</code> and <code>to</code> . By default every unordered pair of sites closer than <code>max_dist</code> .
<code>max_dist</code>	Only use pairs whose straight-line distance is below this (meters). Ignored when <code>pairs</code> is given.
<code>barrier</code>	Resistance at or above which a cell cannot be crossed, see <code>make_transition()</code> .
<code>straight</code>	Add straight-line distance as a baseline hypothesis.

Details

Distances are scaled to mean 0 and standard deviation 1 before fitting, so the slopes are comparable across surfaces. Pairs that cannot be connected through every surface (for example because a barrier isolates a site) are left out of all models, so that every model is fit to the same pairs.

Value

A data.frame with one row per hypothesis, sorted from best to worst: hypothesis, logLik, AIC, delta_AIC (difference from the best model), R2m (marginal R2, the share of variance explained by distance alone; Nakagawa & Schielzeth 2013) and slope (the effect of one standard deviation of distance). Attribute distances holds the pairs and their distance under each hypothesis.

References

Maier PA, Vandergast AG, Ostoja SM, Aguilar A, Bohonak AJ (2022) Landscape genetics of a sub-alpine toad: climate change predicted to induce upward range shifts via asymmetrical migration corridors. *Heredity* 129:257-272.

Nakagawa S, Schielzeth H (2013) A general and simple method for obtaining R2 from generalized linear mixed-effects models. *Methods in Ecology and Evolution* 4:133-142.

Examples

```
ex <- corridor_example()
fst <- pairwise_fst(read_genotypes(ex$files$structure))
surfaces <- list(slope = ex$resistance_slope, cover = ex$resistance_cover,
                 slope_80_cover_20 = ex$resistance)
rank_resistance(surfaces, ex$sites, fst, max_dist = 15000)
```

read_genotypes	<i>Read genotypes from a file</i>
----------------	-----------------------------------

Description

Reads the common population genetic file formats into a [popcounts](#) object (allele counts per population and locus), which is what [pairwise_fst\(\)](#) and [directional_gst\(\)](#) use.

Usage

```
read_genotypes(
  file,
  format = c("auto", "structure", "genepop", "vcf", "genalex", "fstat", "plink"),
  pop = NULL,
  missing = c("-9"),
  onerowperind = NULL,
  label = NULL,
  popdata = NULL,
  popflag = FALSE,
  locdata = FALSE,
  phenotype = FALSE,
  extracols = 0,
  pop_names = NULL
)
```

Arguments

<code>file</code>	Path to the genotype file.
<code>format</code>	One of "auto" (guess from the extension), "structure", "genepop", "vcf", "genalex", "fstat", "plink".
<code>pop</code>	Optional population assignment that overrides the one in the file: a vector with one value per individual in file order, a two-column data.frame or whitespace-separated file (individual, population), or a function that takes individual names and returns populations (for example <code>function(x) sub("_.*", "", x)</code>).
<code>missing</code>	STRUCTURE only: codes for missing alleles.
<code>onerowperind, label, popdata, popflag, locdata, phenotype, extracols</code>	STRUCTURE only. NULL means detect: the row layout is detected from repeated labels, and the leading columns from the marker-name row when there is one. Otherwise the defaults are a label and a population column, which is the most common layout.
<code>pop_names</code>	GENEPOP only: names for the populations, in file order. By default each population takes the name of its first individual, which is the GENEPOP convention.

Value

A popcounts object.

Supported formats

STRUCTURE (`.str`, `.stru`, `.structure`) Both layouts: one row per individual (two columns per locus) or two rows per individual (one column per locus). Optional leading columns, in STRUCTURE's order: a label, the population (POPDATA), POPFLAG, LOCdata, PHENOTYPE and any extra columns. An optional first row of marker names is detected automatically. Alleles may be any integer codes; missing data are -9 by default. When the file has a marker-name row, the number of leading columns is worked out from it, otherwise set the arguments below.

GENEPOP (`.gen`, `.genepop`) Title line, loci one per line or comma-separated, Pop separators, and name , genotypes rows. Two- or three-digit allele codes (detected from the genotype width), haploid or diploid; 0, 00, 000 and so on are missing.

VCF (`.vcf`, `.vcf.gz`) Any ploidy, phased or unphased, biallelic or multiallelic. VCF has no populations, so supply pop.

GenAlEx (`.csv`) The standard codominant layout: a first row with the number of loci, individuals and populations, a second row with population names, a header row, then sample, population and two columns per locus.

FSTAT (`.dat`) The standard header (populations, loci, maximum allele, digits per allele), locus names, then population and genotype columns.

PLINK (`.ped` **with** `.map`, **or** `.raw` **from** `--recode A`) The family ID is used as the population unless pop is given. Convert binary `.bed` files with `plink --recode A` or to VCF first.

Objects already in R (**adegenet** `genind`, `genpop`, `genlight`, or a **vcfR** object) go through `as_popcounts()` instead.

Examples

```
ex <- corridor_example()
read_genotypes(ex$files$structure)

read_genotypes(ex$files$genepop)
read_genotypes(ex$files$vcf, pop = ex$files$popmap)
```

select_bandwidth	<i>Choose a corridor bandwidth for each group of features</i>
------------------	---

Description

For each feature group, fits a random forest of the response (usually FST) on that group's features, once per candidate bandwidth, and keeps the bandwidth with the lowest cross-validated RMSE. This lets broad-scale influences such as climate pick a wide corridor while local ones such as soil pick a narrow one.

Usage

```
select_bandwidth(
  features,
  response,
  groups,
  weights = NULL,
  folds = 5,
  num.trees = 1000,
  min.node.size = c(1, 5, 10),
  seed = 12345
)
```

Arguments

features	A named list of data.frames, one per bandwidth, each with columns from, to and the features, as returned by corridor_extract() . All must list the same pairs in the same order.
response	Numeric response for each row (each pair).
groups	A named list mapping group names to feature (column) names.
weights	Optional case weights, see pair_weights() .
folds	Number of cross-validation folds. Both directions of a pair (rows A to B and B to A) always fall in the same fold.
num.trees	Trees per forest.
min.node.size	Values to try for the minimum node size. mtry is tuned over every value from 1 to the number of features in the group.
seed	Random seed for the folds and forests.

Value

A data.frame with one row per group and bandwidth (group, bandwidth, mtry, min.node.size, RMSE, Rsquared), sorted by group and RMSE, with attribute best: a named vector of the chosen bandwidth per group.

Examples

```
# Two candidate bandwidths; climate measured in the wide corridor explains
# the response better
set.seed(1)
n <- 150
ids <- data.frame(from = paste0("s", 1:n), to = paste0("t", 1:n))
wide <- data.frame(ids, snow = rnorm(n), slope = rnorm(n))
narrow <- data.frame(ids, snow = wide$snow + rnorm(n), slope = rnorm(n))
fst <- 0.1 - 0.03 * wide$snow + rnorm(n, sd = 0.01)
sel <- select_bandwidth(list(narrow = narrow, wide = wide), fst,
                        groups = list(climate = "snow", terrain = "slope"),
                        num.trees = 100, folds = 3)

attr(sel, "best")
```

shift_arrows

Arrows for a directional map

Description

Summarizes a map from `map_pairwise()` on a coarser grid, for drawing arrows. Direction follows the summed vectors. By default length does too, so arrows are long where corridors agree on a direction and shrink where opposing flows cancel; `length_by = "magnitude"` scales them by the mean size of change instead.

Usage

```
shift_arrows(
  map,
  cells = 25,
  length = NULL,
  length_by = c("net", "magnitude"),
  scale = c("linear", "sqrt"),
  min_quantile = 0,
  min_length = 0
)
```

Arguments

map	A directional map from <code>map_pairwise()</code> .
cells	Grid size, in cells of map.
length	Length of the longest arrow, in map units. Defaults to about 80% of a grid cell.

length_by	"net" (length of the summed vector) or "magnitude" (mean size of change in the grid cell).
scale	"linear" lengths, or "sqrt" to compress the range so a few very large changes do not shrink every other arrow to a dot.
min_quantile	Drop arrows in the weakest cells (below this quantile of magnitude).
min_length	Shortest arrow, as a share of length. With the default of 0, arrow length is proportional to magnitude, so arrows shrink to nothing where nothing changes.

Value

A data.frame with x, y, xend, yend and magnitude, ready for `ggplot2::geom_segment()` or `graphics::arrows()`.

Examples

```
ex <- corridor_example(acc = TRUE)
v <- both_directions(data.frame(from = c("M03", "M08"), to = c("M20", "M25"), value = 0.05))
v$value[3:4] <- -0.05
m <- map_pairwise(v, ex$acc, ex$sites)
a <- shift_arrows(m, cells = 15)
len <- sqrt((a$xend - a$x)^2 + (a$yend - a$y)^2)
a <- a[len > 0.02 * 15 * terra::res(m)[1], ] # too short to draw, as in plot_shift()
terra::plot(m$value, col = hcl.colors(100, "viridis"))
arrows(a$x, a$y, a$xend, a$yend, length = 0.05)
```

site_contrast

Environmental contrasts between sites

Description

At-site features: the value of each environmental layer at the source site minus its value at the destination site. Sites can be points (the cell value is used) or polygons (the area-weighted mean is used). Site-level attributes such as meadow area or network degree can be supplied as a data.frame instead of, or in addition to, rasters.

Usage

```
site_contrast(
  sites,
  pairs,
  env = NULL,
  attributes = NULL,
  id = "site",
  suffix = ".at"
)
```

Arguments

sites	An sf object of sites.
pairs	Data.frame with columns from and to (directed pairs).
env	Optional SpatRaster of environmental layers.
attributes	Optional data.frame with an ID column and one column per site attribute.
id	Name of the ID column in sites (and attributes).
suffix	Appended to the feature names.

Value

A data.frame with from, to and one column per feature, each equal to the source value minus the destination value.

Examples

```
ex <- corridor_example()
pr <- data.frame(from = c("M01", "M12"), to = c("M12", "M01"))
site_contrast(ex$sites, pr, env = ex$env[["snowpack"]])
```

Index

accumulated_cost, 2
accumulated_cost(), 5, 6, 9, 12
as_popcounts, 3
as_popcounts(), 7, 14, 20

both_directions, 4

corridor_example, 5
corridor_extract, 6
corridor_extract(), 21

directional_gst, 7
directional_gst(), 19

fit_connectivity, 8
fit_connectivity(), 17

graphics::arrows(), 23

lcc_weights, 9
lcc_weights(), 6, 13
least_cost_paths, 10
least_cost_paths(), 15
lineage_cross, 11
lineage_cross(), 5

make_transition, 11
make_transition(), 3, 10, 18
map_pairwise, 12
map_pairwise(), 16, 22

pair_weights, 13
pair_weights(), 21
pairwise_fst, 14
pairwise_fst(), 18, 19
pairwise_table, 15
path_buffers, 15
path_buffers(), 6
plot_shift, 16
popcounts, 7, 14, 19
predict.corridor_model, 17

rank_resistance, 18
read_genotypes, 19
read_genotypes(), 4

select_bandwidth, 21
shift_arrows, 22
shift_arrows(), 16
site_contrast, 23
stats::predict(), 8