Package: ResistanceGA2 (via r-universe)

November 12, 2026

Title Optimize Resistance Surfaces Using Genetic Algorithms

Version 1.2.0

Description Optimizes resistance surfaces using genetic algorithms. Continuous and categorical surfaces can be optimized individually or jointly to create composite resistance surfaces. Resistance distances can be calculated using 'Circuitscape' through 'Julia' or with the 'gdistance' package. The framework extends Peterman (2018) <doi:10.1111/2041-210X.12984>.

URL https://github.com/wpeterman/ResistanceGA2

BugReports https://github.com/wpeterman/ResistanceGA2/issues

VignetteBuilder knitr

Depends R (>= 4.1), terra (>= 1.7-0)

Imports ggExtra, ggplot2, methods, plyr, dplyr, gdistance (>= 1.6), raster, reformulas, magrittr, GA (>= 3.2.1), lme4 (>= 1.0), Matrix, MuMIn, spdep, JuliaConnectoR

Suggests doParallel, igraph, knitr, pkgload, rmarkdown, testthat (>= 3.0.0)

License GPL (>= 3)

LazyData true

Encoding UTF-8

Language en-US

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libglpk-dev make libuv1-dev libxml2-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

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

Date/Publication 2026-09-10 02:51:55 UTC

RemoteUrl https://github.com/wpeterman/ResistanceGA2

RemoteRef HEAD

RemoteSha 65a1a428ef13b95fdf50ba691e0433e368af9253

Contents

all_comb . . . 3
Combine_Surfaces . . . 5
Dc_list . . . 7
Diagnostic.Plots . . . 7
GA.prep . . . 9
gdist.prep . . . 14
graph_to_keep . . . 16
jl.prep . . . 18
jl_out . . . 20
lower . . . 21
mlpe . . . 22
MLPE.lmm . . . 23
mlpe_data . . . 25
mlpe_rga . . . 26
MS_optim . . . 28
Plot.trans . . . 29
raster_orig . . . 31
raster_true . . . 32
recommend_commute_settings . . . 32
Resist.boot . . . 34
resist_list . . . 36
Resistance.tran . . . 37
resistance_surfaces . . . 38
ResistanceGA2-result-methods . . . 39
rga2_demo . . . 42
rga2_demo_covariates . . . 43
rga2_demo_genetic . . . 43
Run_CS.ini . . . 44
Run_CS.jl . . . 46
Run_gdistance . . . 48
sample_pops . . . 50
samples . . . 50
SHNe . . . 51
site_pairwise_diff . . . 52
site_pairwise_dist . . . 53
SS_optim . . . 54
vignette_current_map . . . 56
vignette_pairwise_outputs . . . 56

Index **57**

all_comb *Analyze all combinations*

Description

A wrapper function to run both SS_optim and MS_optim to optimize all combinations of resistance surfaces with the Genetic Algorithms package **GA**. Following optimization, Resist.boot is run to conduct a bootstrap analysis.

Usage

```
all_comb(
  gdist.inputs = NULL,
  jl.inputs = NULL,
  GA.inputs,
  results.dir,
  max.combination = 4,
  iters = 1000,
  replicate = 1,
  sample.prop = 0.75,
  nlm = FALSE,
  dist_mod = TRUE,
  null_mod = TRUE,
  n.cores = NULL,
  ...
)
```

Arguments

gdist.inputs	Object created from running gdist.prep. Supply when optimizing with **gdistance**.
jl.inputs	Object created from running jl.prep. Supply when optimizing with Circuitscape in Julia.
GA.inputs	Object created from running GA.prep. Be sure that Results.dir = "all.comb" was used when creating this object.
results.dir	Directory in which to write the analysis. A nonempty directory is used only after interactive confirmation; the user may instead choose a new time-stamped subdirectory.
max.combination	Maximum number of surfaces per candidate model, or a two-element vector giving the minimum and maximum. A minimum greater than one skips single-surface optimization. Values cannot exceed the number of raster layers.
iters	Positive integer giving the number of population-subsampling iterations passed to [Resist.boot()]. Default is 1000.
replicate	Positive integer giving the number of independent GA runs for each candidate set. Default is 1.

sample.prop	Proportion of populations used to fit each bootstrap model. Default is 0.75; see [Resist.boot()].
nlm	Reserved for a possible final nonlinear optimization step. This feature is not implemented and must remain FALSE.
dist_mod	Logical. Include a geographic-distance baseline in the candidate set? Default is TRUE.
null_mod	Logical. Include an intercept-only baseline? Default is TRUE.
n.cores	Number of parallel workers used by [Resist.boot()]. NULL or 1 runs the bootstrap serially. GA fitness parallelism is controlled separately by GA.inputs$parallel.
...	Reserved for backward compatibility. Additional arguments are not currently used.

Details

Candidate-set comparisons are valid only for models fitted to the same response and retained population pairs. GA replication evaluates numerical stability of the optimizer. Population subsampling evaluates how candidate rankings change across sampled population sets. Neither is a substitute for inspecting fitted coefficients, diagnostic plots, or the scientific plausibility of each surface.

Value

An object with class resga_all_comb. For one replicate, the object contains summary.table, boot.results, model metadata (all.k), resistance-distance matrices (all.cd), the genetic distance matrix, and the underlying ss.results and ms.results. With multiple replicates, these elements are nested in named rep_1, rep_2, and subsequent lists. CSV summaries and serialized optimization outputs are also written beneath results.dir.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[SS_optim()], [MS_optim()], [Resist.boot()], [summary.resga_all_comb()]

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")
gd <- lower(Dc_list[[1]])

gdist.inputs <- gdist.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = gd,
  samples = pts
)

GA.inputs <- GA.prep(
  raster = raster_orig,
```

```
  Results.dir = "all.comb",
  quiet = TRUE
)

results_dir <- file.path(tempdir(), "ResistanceGA2-all-comb")

combo.out <- all_comb(
  gdist.inputs = gdist.inputs,
  GA.inputs = GA.inputs,
  results.dir = results_dir,
  max.combination = 2,
  iters = 10,
  replicate = 1,
  n.cores = 2
)

## End(Not run)
```

`Combine_Surfaces` *Combine multiple resistance surfaces into a composite surface*

Description

Combines multiple resistance surfaces by transforming each layer with specified parameters and summing the results.

Usage

```
Combine_Surfaces(
  PARM,
  gdist.inputs = NULL,
  jl.inputs = NULL,
  GA.inputs,
  out = NULL,
  File.name = paste(GA.inputs$parm.type$name, collapse = "."),
  rescale = TRUE,
  p.contribution = FALSE
)
```

Arguments

`PARM`	Parameters to transform continuous or categorical resistance surfaces. Provide a vector with parameters in the order of resistance surfaces as output by `MS_optim`.
`gdist.inputs`	Object from `gdist.prep`. Supply when optimizing with gdistance.
`jl.inputs`	Object from `jl.prep`. Supply when optimizing with Circuitscape via Julia.
`GA.inputs`	Object from `GA.prep`.
`out`	Directory to write the combined `.asc` file. Default = `NULL` (no file written).

File.name	Name for the output .asc file. Default is the surface names joined by ".".
rescale	Logical. If TRUE (default), the combined surface is divided by its minimum value so the minimum resistance equals 1.
p.contribution	Logical. If TRUE, returns a list with (1) the combined raster surface and (2) the mean proportional contribution of each input surface to the combined resistance values.

Details

For continuous surfaces, PARM should specify, per layer, three values: (1) transformation code (1–9), (2) shape, and (3) maximum value. Transformation codes:

1 = "Inverse-Reverse Monomolecular"
2 = "Inverse-Reverse Ricker"
3 = "Monomolecular"
4 = "Ricker"
5 = "Reverse Monomolecular"
6 = "Reverse Ricker"
7 = "Inverse Monomolecular"
8 = "Inverse Ricker"
9 = "Distance"

Value

A SpatRaster of the combined resistance surface (sum of all transformed/reclassified layers), or a list when p.contribution = TRUE.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

Examples

```
## Not run:
# After running MS_optim(), use the best-fit PARM vector:
combined <- Combine_Surfaces(
  PARM        = ms.out$PARM,
  gdist.inputs = gdist.inputs,
  GA.inputs   = GA.inputs
)

## End(Not run)
```

Dc_list *Data file. A list of three matrices*

Description

Sample file to be used with examples in the vignette

Usage

```
data(Dc_list)
```

Format

A list of length 3

Details

Each matrix depicts the pairwise genetic distance (measured as chord distance) between sample locations

- Dc_cat. Pairwise chord distance matrix simulated across the categorical resistance surface
- Dc_cont. Pairwise chord distance matrix simulated across the continuous resistance surface
- Dc_multi. Pairwise chord distance matrix simulated across the multivariate resistance surface

Diagnostic.Plots *Diagnose an MLPE resistance-distance model*

Description

Fits a mixed-effects model with pairwise effects (MLPE) to an optimized resistance-distance predictor and saves a four-panel diagnostic figure.

Usage

```
Diagnostic.Plots(
  resistance.mat,
  genetic.dist,
  XLAB = "Estimated resistance",
  YLAB = "Genetic distance",
  plot.dir,
  type = c("categorical", "continuous"),
  name = NULL,
  ID = NULL,
  ZZ = NULL
)
```

Arguments

resistance.mat	A square resistance-distance matrix, a lower-triangle distance vector, a dist object, or the path to a Circuitscape resistance-output table.
genetic.dist	Numeric vector of pairwise genetic distances in the same lower-triangle order as resistance.mat. This is commonly gdist.inputs$response or jl.inputs$response.
XLAB	Character string used for the resistance-distance axis label.
YLAB	Character string used for the response axis label.
plot.dir	Directory in which to write the TIFF figure. The directory is created when necessary.
type	Surface type, either "continuous" or "categorical". The value is retained in the returned object and for compatibility with optimization workflows. Effective resistance is continuous for either surface type, so both use the same diagnostic panels.
name	Output-file stem. A stem is derived from a Circuitscape file name when possible; otherwise name is required.
ID	Optional pair-identifier data frame from [gdist.prep()] or [jl.prep()]. When omitted, identifiers are constructed from the inferred population count.
ZZ	Optional sparse MLPE design matrix from [gdist.prep()] or [jl.prep()]. This argument is retained for compatibility with legacy workflows. Current pair-identifier metadata is used when available.

Details

The response panel includes an ordinary least-squares line as a visual trend guide only. Inference should come from the fitted MLPE model, which accounts for the nonindependence of pairwise observations. Strong curvature, changing residual spread, or marked departures in the quantile plot indicate that the fitted mean or variance structure may be inadequate. These plots assess in-sample fit and do not demonstrate transferability to new populations or landscapes.

Value

Invisibly returns a list with the fitted model, the model data, normalized output file path, and surface type. The saved TIFF contains observed response against resistance distance, residuals against marginal fitted values, a residual histogram, and a residual normal quantile plot.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[plot.resga_ss_optim()], [plot.resga_ms_optim()], [MLPE.lmm()]

Examples

```
## Not run:
set.seed(4)
xy <- matrix(stats::runif(12), ncol = 2)
genetic <- lower(as.matrix(stats::dist(xy)))
resistance <- lower(as.matrix(stats::dist(cbind(xy[, 1], xy[, 2]^2))))

diagnostics <- Diagnostic.Plots(
  resistance.mat = resistance,
  genetic.dist = genetic,
  plot.dir = tempdir(),
  type = "continuous",
  name = "example"
)
diagnostics$model

## End(Not run)
```

GA.prep *Prepare resistance surfaces and genetic-algorithm settings*

Description

Validates a multilayer resistance raster, classifies its layers as continuous or categorical, defines parameter bounds and starting values, and stores the settings used by [SS_optim()], [MS_optim()], and [all_comb()].

Usage

```
GA.prep(
  raster,
  Results.dir = NULL,
  min.cat = NULL,
  max.cat = 1000,
  max.cont = 1000,
  shape.min = NULL,
  shape.max = NULL,
  cont.shape = NULL,
  select.trans = NULL,
  cat.levels = 15,
  method = "LL",
  k.value = 2,
  pop.mult = 15,
  percent.elite = 5,
  type = "real-valued",
  pcrossover = 0.85,
  pmutation = 0.125,
  maxiter = 1000,
```

```
  run = NULL,
  keepBest = TRUE,
  population = gaControl(type)$population,
  selection = gaControl(type)$selection,
  crossover = "gareal_blxCrossover",
  mutation = gaControl(type)$mutation,
  pop.size = NULL,
  parallel = FALSE,
  gaisl = FALSE,
  island.pop = 20,
  numIslands = NULL,
  migrationRate = NULL,
  migrationInterval = NULL,
  optim = FALSE,
  optim.method = "L-BFGS-B",
  poptim = 0,
  pressel = 1,
  control = list(fnscale = -1, maxit = 100),
  hessian = FALSE,
  opt.digits = NULL,
  seed = NULL,
  monitor = TRUE,
  quiet = FALSE
)
```

Arguments

raster	A SpatRaster (from **terra**) with one or more layers.
Results.dir	Directory to export optimization results. If the directory does not exist, it will be created automatically. If it exists and is not empty, a message will note that existing results may be overwritten. It is critical that there are NO SPACES in the directory path. If using the all_comb function, specify Results.dir as "all_comb".
min.cat	The minimum value to be assessed during optimization of categorical resistance surfaces (Default = 1 / max.cat)
max.cat	The maximum value to be assessed during optimization of categorical resistance surfaces (Default = 1000)
max.cont	The maximum value to be assessed during optimization of continuous resistance surfaces (Default = 1000)
shape.min	The minimum value for the shape parameter used for transforming resistance surfaces. If unspecified, used 0.5
shape.max	The maximum value for the shape parameter used for transforming resistance surfaces. If unspecified, used 14.5
cont.shape	A vector of hypothesized relationships that each continuous resistance surface will have in relation to the genetic distance response (Default = NULL; see details)

select.trans	Option to specify which transformations are applied to continuous surfaces. Must be provided as a list. "A" = All, "M" = Monomolecular only, "R" = Ricker only. Default = "M"; see Details.
cat.levels	Number of unique levels to permit in categorical surface (Default = 15). See Details
method	Objective function to be optimized. Select "AIC", "R2", or "LL" to optimize resistance surfaces based on AIC, variance explained (R2), or log-likelihood. (Default = "LL")
k.value	Specification of how k, the number of parameters in the mixed effects model, is determined. Specify 1, 2, 3, or 4 (Default = 2; see details). 1 –> k = 2; 2 –> k = number of parameters optimized plus the intercept; 3 –> k = the number of parameters optimized plus the intercept and the number of layers optimized; 4 –> k = the number of layers optimized plus the intercept
pop.mult	Value will be multiplied with number of parameters in surface to determine 'popSize' in GA. By default this is set to 15.
percent.elite	An integer percent used to determine the number of best fitness individuals to survive at each generation ('elitism' in GA). By default the top 5% individuals will survive at each iteration.
type	Default is "real-valued"
pcrossover	Probability of crossover. Default = 0.85
pmutation	Probability of mutation. Default = 0.125
maxiter	Maximum number of iterations to run before the GA search is halted. If using standard ga optimizer, the default = 1000. If using gaisl = TRUE, then this is set to 15x the migrationInterval
run	Number of consecutive generations or epochs without any improvement in objective function before the GA is stopped. If using standard ga, the default = 25. If using gaisl = TRUE, then the default run value will be calculated as migrationInterval * 5
keepBest	A logical argument specifying if best solutions at each iteration should be saved (Default = TRUE)
population	Default is "gareal_Population" from GA
selection	Default is "gareal_lsSelection" from GA
crossover	Default = "gareal_blxCrossover". This crossover method greatly improved optimization during preliminary testing
mutation	Default is "gareal_raMutation" from GA
pop.size	Number of individuals to create each generation. If gaisl = TRUE, then this number is automatically calculated as numIslands * island.pop
parallel	Controls parallel fitness evaluation in the GA package. Use FALSE for serial execution, TRUE for the package-selected worker count, or a positive integer for an explicit worker count. PSOCK workers are used on Windows. Parallel evaluation requires the suggested **doParallel** package.

gaisl	Should the genetic algorithm use the islands parallel optimization? (Default = FALSE)
island.pop	The number of individuals to populate each island. (Default = 20)
numIslands	If gaisl = TRUE, an integer value which specifies the number of islands to use in the genetic evolution (by default will be set to 4)
migrationRate	If gaisl = TRUE, a value in the range (0, 1) which gives the proportion of individuals that undergo migration between islands in every exchange (by default equal to 0.10).
migrationInterval	If gaisl = TRUE, an integer value specifying the number of iterations at which exchange of individuals takes place. This interval between migrations is called an epoch, and it is set at 10 by default.
optim	A logical defaulting to FALSE determining whether or not a local search using general-purpose optimization algorithms should be used. See argument optimArgs for further details and finer control. Setting to TRUE has the potential to improve optimization accuracy, but will increase optimization time.
optim.method	The method to be used among those available in optim function. By default, the BFGS algorithm with box constraints is used, where the bounds are those provided in the ga() function call. Further methods are available as described in the Details section in help(optim).
poptim	A value in the range [0,1] specifying the probability of performing a local search at each iteration of GA (default 0.0). Only change if your optimization is relatively fast.
pressel	A value in the range [0,1] specifying the pressure selection (default 1.00). The local search is started from a random solution selected with probability proportional to fitness. High values of pressel tend to select the solutions with the largest fitness, whereas low values of pressel assign quasi-uniform probabilities to any solution.
control	A list of control parameters. See 'Details' section in optim
hessian	Logical. Should a numerically differentiated Hessian matrix be returned? This will allow for the calculation of standard errors on parameter estimates (not yet implemented). Default = FALSE
opt.digits	The number of significant digits that the objective function will be assessed at. By default, no rounding occurs.
seed	Integer random number seed to replicate ga optimization
monitor	Default = TRUE, which prints the average and best fitness values at each iteration.
quiet	Logical. If TRUE, the objective function and step run time will not be printed to the screen after each step. Only ga summary information will be printed following each iteration. (Default = FALSE)

Details

Provide the resistance surfaces directly as a SpatRaster. If multiple surfaces are to be optimized, include them as layers in that object. File-path and directory-based raster inputs are no longer accepted.

The Default for k.value is 2, which sets k equal to the number of parameters optimized, plus 1 for the intercept term. Prior to version 3.0-0, k.value could not be specified by the user and followed setting 2, such that k was equal to the number of parameters optimized plus the intercept term.

cont.shape can take values of "Increase", "Decrease", or "Peaked". If you believe a resistance surface is related to your response in a particular way, specifying this here may decrease the time to optimization. cont.shape is used to generate an initial set of parameter values to test during optimization. If specified, a greater proportion of the starting values will include your believed relationship. If unspecified (the Default), a completely random set of starting values will be generated.

If it is desired that only certain transformations be assessed for continuous surfaces, then this can be specified using select.trans. By default, only monomolecular transformations will be assessed for continuous surfaces unless otherwise specified. This default is also the recommended starting point for most analyses: monomolecular families are usually easier to interpret and generally reduce the risk of overfitting relative to Ricker families. Ricker transformations should usually be explored only when there is a strong biological or ecological justification for a unimodal or quadratic-type relationship. Specific transformations can be specified by providing a vector of values (e.g., c(1,3,5)), with values corresponding to the equation numbers as detailed in Resistance.tran. If multiple rasters are to be optimized from the same SpatRaster object, then a list of transformations must be provided in the order that the raster surfaces will be assessed. For example:
select.trans = list("M", "A", "R", c(5,6))
will result in surface one only being optimized with Monomolecular transformations, surface two with all transformations, surface three with only Ricker transformations, and surface four with Reverse Ricker and Reverse Monomolecular only. If a categorical surface is among the rasters to be optimized, it is necessary to specify NA to accommodate this.

cat.levels defaults to 15. This means that when a raster surface has <= 15 unique levels, it will be treated as a categorical surface in the analysis. This value can be increased, but optimization of surfaces with many levels may take more time. Additionally, depending upon the prevalence and configuration of categorical features and spatial sample locations, some levels are likely to be poorly estimated. This may be evident if estimated resistance values vary substantially between runs of ResistanceGA.

Setting gaisl = TRUE has the potential greatly reduce the optimization run time, potentially with greater accuracy. This is a distributed multiple-population GA, where the population is partitioned into several subpopulations and assigned to separated islands. Independent GAs are executed in each island, and only occasionally sparse exchanges of individuals are performed among the islands.

It is recommended to first run GA optimization with the default settings.

Value

A named list used as GA.inputs. It contains the validated resistance raster (Resistance.stack and raster), layer names and inferred surface types, parameter indices and bounds, starting suggestions, output directories, the objective and parameter-count rules, random seed, and all standard or island-GA control settings. When Results.dir is "all.comb" or "all_comb", directory fields are left NULL and an inputs list is retained so that [all_comb()] can create replicate-specific settings.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[SS_optim()], [MS_optim()], [all_comb()]

Examples

```
ga.inputs <- GA.prep(
  raster = raster_orig,
  Results.dir = file.path(tempdir(), "ResistanceGA2-example"),
  quiet = TRUE,
  monitor = FALSE
)

ga.inputs$layer.names
```

---

gdist.prep *Prepare data for optimization using* gdistance

Description

Creates a necessary input for optimizing resistance surfaces based on pairwise cost distances, implemented using the gdistance library.

Usage

```
gdist.prep(
  n.Pops,
  response = NULL,
  samples,
  covariates = NULL,
  formula = NULL,
  transitionFunction = function(x) 1/mean(x),
  directions = 8,
  longlat = FALSE,
  method = "commuteDistance",
  min.max_dist = NULL,
  keep = NULL,
  commute.approx = c("none", "aggregate"),
  approx.factor = 4L,
  approx.scale = TRUE
)
```

Arguments

n.Pops	The number of populations that are being assessed.
response	Vector of pairwise genetic distances (lower half of pairwise matrix).
samples	A SpatVector of sample point locations.

`covariates` Data frame of additional covariates to include in the MLPE model during optimization.

`formula` If covariates are included in the model, specify the R formula for the fixed effects portion of the MLPE model (e.g., `response ~ covariate`). The `response` term will use the values supplied to the `response` parameter; covariate names must match column names in `covariates`.

`transitionFunction` Function to calculate the `gdistance` TransitionLayer. See `transition`. Default = `function(x) 1 / mean(x)`.

`directions` Directions in which cells are connected (4, 8, 16, or other). Default = 8.

`longlat` Logical. If `TRUE`, a `geoCorrection` will be applied to the transition matrix. Default = `FALSE`.

`method` Pairwise distance method: `'commuteDistance'` (default, equivalent to Circuitscape resistance distance) or `'costDistance'` (least-cost path distance). See `costDistance` and `commuteDistance`.

`min.max_dist` Optional two-element vector `c(min, max)` specifying the inclusive Euclidean distance range for pairwise comparisons. Pairs outside this range are omitted through the same selected-pair workflow used by `keep`.

`keep` Optional vector of length equal to the number of pairwise observations, with 1 to keep an observation and 0 to drop it. Use `graph_to_keep` to derive this from a pruned population graph, adjacency matrix, or edge list. Can be used in conjunction with, or in place of, `min.max_dist`.

`commute.approx` Approximation backend for iterative `commuteDistance` optimization runs. `"none"` (default) keeps exact **gdistance** output. `"aggregate"` first coarsens the resistance raster before calculating commute distances.

`approx.factor` Integer aggregation factor used when `commute.approx = "aggregate"`. Default = 4.

`approx.scale` Logical. If `TRUE`, multiply aggregate approximation distances by `approx.factor^2` to keep values closer to the full-resolution scale. Default = `TRUE`.

Details

Sample locations must be supplied as a point `SpatVector`. When `Run_gdistance` is called, the resistance surface is converted internally to the raster class required by **gdistance**.

Use `recommend_commute_settings` if you want a quick benchmark-based suggestion for `commute.approx` and `approx.factor` before calling `gdist.prep()`.

Use `graph_to_keep` if you want to restrict optimization to a graph-pruned subset of population pairs.

A covariate formula is expanded to include the resistance-distance predictor and MLPE pair effect. Covariates therefore adjust the resistance association; they do not become part of the resistance surface itself.

Value

A named list used as `gdist.inputs`. It contains the response, point locations, optional covariates and expanded MLPE formula, transition and correction settings, pair identifiers (`ID`), MLPE design matrix (`ZZ`), retained-pair indicator (`keep`), population count, distance method, commute-approximation settings, and model data (`df`). The object can be passed directly to [Run_gdistance()], [SS_optim()], or [MS_optim()].

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[Run_gdistance()], [GA.prep()], [recommend_commute_settings()]

Examples

```
pts <- terra::vect(sample_pops[[1]], type = "points")

commute_plan <- recommend_commute_settings(
  raster = terra::subset(resistance_surfaces, "continuous"),
  n.Pops = nrow(sample_pops[[1]])
)

gdist.inputs <- gdist.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = lower(Dc_list[[1]]),
  samples = pts,
  method = "commuteDistance",
  commute.approx = commute_plan$commute.approx,
  approx.factor = commute_plan$approx.factor
)

head(gdist.inputs$df)
```

---

graph_to_keep *Convert graph topology to a pair-selection keep vector*

Description

Translate a pruned population graph, adjacency matrix, or edge list into the integer keep vector used by `gdist.prep` and `jl.prep` to retain only a subset of pairwise observations.

Usage

```
graph_to_keep(graph, sampled_pops = NULL, pop_n = NULL, labels = NULL)
```

Arguments

graph	Graph-like object describing which population pairs should be retained. Supported inputs are: • a square adjacency/incidence matrix; • a two-column edge list supplied as a data frame; • an igraph or popgraph object, when **igraph** is available.
sampled_pops	Integer number of sampled populations. Required when it cannot be inferred from graph or labels.
pop_n	Optional integer vector giving the number of sampled individuals per population. When supplied, the returned keep vector is expanded to the individual-pair scale used by jl.prep(pop2ind = ...).
labels	Optional character vector giving population labels in the same order as the genetic distance matrix and sample coordinates supplied to gdist.prep() or jl.prep(). Use this when graph vertices are named.

Details

ResistanceGA2 assumes pairwise distances are symmetric, so directed graph inputs are treated as undirected. For square matrices without dimnames, row order is assumed to already match the order of populations in your pairwise genetic distance data.

This helper is intended to bridge graph-based pruning workflows from tools such as **popgraph** or **graph4lg** into ResistanceGA2 without requiring those packages as hard dependencies.

Value

Integer vector of 1s and 0s aligned to lower / To.From.ID ordering. Pass this to gdist.prep(keep =) or jl.prep(pairs_to_include =).

Examples

```
adj <- matrix(
  c(0, 1, 0,
    1, 0, 1,
    0, 1, 0),
  nrow = 3,
  byrow = TRUE,
  dimnames = list(c("A", "B", "C"), c("A", "B", "C"))
)

keep <- graph_to_keep(adj, labels = c("A", "B", "C"))
keep

graph_to_keep(adj, labels = c("A", "B", "C"), pop_n = c(1L, 2L, 1L))
```

jl.prep *Prepare inputs for running Circuitscape via Julia*

Description

Creates all inputs required by the optimization functions when using Circuitscape run through Julia (via **JuliaConnectoR**).

Usage

```
jl.prep(
  n.Pops,
  response = NULL,
  CS_Point.File,
  covariates = NULL,
  formula = NULL,
  JULIA_HOME = NULL,
  Neighbor.Connect = 8,
  pairs_to_include = NULL,
  pop2ind = NULL,
  nb = NULL,
  parallel = FALSE,
  cores = NULL,
  cholmod = TRUE,
  precision = FALSE,
  run_test = TRUE,
  write.files = NULL,
  write.criteria = NULL,
  silent = TRUE,
  scratch = NULL,
  rm.files = TRUE
)
```

Arguments

n.Pops	The number of populations being assessed.
response	Vector of pairwise genetic distances (lower half of pairwise matrix). If using pairs_to_include, provide the full lower-half vector; it will be trimmed automatically.
CS_Point.File	A SpatVector of sample point locations.
covariates	Data frame of additional covariates for the MLPE model.
formula	R formula for the fixed effects of the MLPE model (e.g., response ~ covariate). The response term uses the values in response; covariate names must match covariates columns.
JULIA_HOME	Path to the Julia bin directory. If NULL, JULIA_BINDIR must already identify that directory.

`Neighbor.Connect`
: Connectivity scheme for Circuitscape: 4 or 8 (Default = 8).

`pairs_to_include`
: Integer vector of 1 (keep) or 0 (drop) for each pairwise observation. Use `graph_to_keep` to derive this from a pruned population graph, adjacency matrix, or edge list. Default = `NULL` (use all pairs).

`pop2ind`
: Integer vector giving the number of sampled individuals per population. Default = `NULL`.

`nb`
: Maximum distance (in coordinate units) for treating locations as the same neighborhood. Default = `NULL`.

`parallel`
: Logical. Run Circuitscape in parallel? Default = `FALSE`.

`cores`
: Number of cores for parallel processing (only used when `parallel` = `TRUE`).

`cholmod`
: Logical. Use the CHOLMOD direct solver? Default = `TRUE`. Cannot be used with single precision (`precision` = `TRUE`).

`precision`
: Logical. Use experimental single-precision mode? Default = `FALSE`.

`run_test`
: Logical. Test the Julia/Circuitscape connection before continuing? Default = `TRUE`.

`write.files`
: Directory path. If provided, the `.ini` and `.asc` files used in each Circuitscape run are saved there. If the directory does not exist, it will be created automatically.

`write.criteria`
: Minimum run time (seconds) threshold for writing files when `write.files` is set. If `NULL`, all runs are written.

`silent`
: Logical. Suppress Circuitscape progress output? Default = `TRUE`.

`scratch`
: Scratch directory for temporary files when the default `tempdir()` is not writable.

`rm.files`
: Logical. Remove temporary files after each Julia run? Default = `TRUE`.

Details

Julia and Circuitscape.jl must be installed in the selected Julia environment (see https://github.com/Circuitscape/Circuitscape.jl).

Set `JULIA_HOME` to the `bin/` subdirectory of your Julia installation (e.g., `"/usr/local/julia/bin"`).

The CHOLMOD solver (`cholmod` = `TRUE`) uses a direct Cholesky decomposition that can be much faster than the default algebraic multigrid solver for moderate problem sizes, but may consume large amounts of memory for very large problems. It cannot be used with single precision.

Point locations are supplied as a `SpatVector` and written to a temporary Circuitscape point file internally.

The requested Circuitscape parallel settings are stored in the returned object. They control Circuitscape execution and are separate from GA fitness parallelism in [GA.prep()] and bootstrap parallelism in [Resist.boot()].

Value

A named list used as `jl.inputs`. It contains retained and full pair identifiers (`ID` and `ID.all`), MLPE design matrices, response and covariate data, fitted-model formula, sample points, connectivity and solver settings, Julia location, file controls, pair inclusion metadata, and population-to-individual mapping. The object can be passed directly to [Run_CS.jl()], [SS_optim()], or [MS_optim()].

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[Run_CS.jl()], [Run_CS.ini()], [GA.prep()]

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")

jl.inputs <- jl.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = lower(Dc_list[[1]]),
  CS_Point.File = pts,
  JULIA_HOME = Sys.getenv("JULIA_BINDIR"),
  run_test = FALSE
)

## End(Not run)
```

---

jl_out *Data file. Example raster output from CIRCUITSCAPE Julia*

Description

A `SpatRaster` produced by a Circuitscape Julia run.

Usage

```
data(jl_out)
```

Format

`SpatRaster` with 1 layer

`lower` *Extract the lower triangle of a square matrix*

Description

Returns matrix entries below the diagonal in the column-major order used by ResistanceGA2 pairwise responses, predictors, and keep vectors.

Usage

```
lower(matrix)
```

Arguments

`matrix`	Square matrix. Dimension names are permitted but are not retained in the output.

Details

Pairwise vectors supplied to the same analysis must all be created with the same population order. A warning is issued for a vector or a nonsquare input.

Value

A vector of length `choose(nrow(matrix), 2)` containing entries for pairs `(2,1)`, `(3,1)`, `...`, `(n,n - 1)`. This is the same ordering as `matrix[lower.tri(matrix)]` and [To.From.ID()].

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[To.From.ID()], [mlpe_data()]

Examples

```
m <- matrix(1:16, nrow = 4)
lower(m)   # returns 6-element vector
```

`mlpe` *Fit MLPE Models With an lme4-Style Formula*

Description

Fit maximum-likelihood population effects (MLPE) models using standard `lme4`-style formulas while defining dyadic random-intercept terms from explicit endpoint columns.

Usage

```
mlpe(formula, data, pairs, REML = FALSE, keep = NULL, ...)
```

Arguments

`formula`	Mixed-effects model formula.
`data`	Data frame containing the response, covariates, and endpoint columns.
`pairs`	Definition of one or more MLPE dyad terms. Supply either a character vector of length two, which is interpreted as `list(pair = c("from", "to"))`, or a named list mapping random-effect term names in `formula` to the two endpoint columns used to construct each dyadic random intercept.
`REML`	Logical. If TRUE, fit by restricted maximum likelihood for linear mixed models.
`keep`	Optional logical/integer vector used to retain a subset of rows from `data` before fitting.
`...`	Additional arguments passed to `glFormula` when fitting generalized models, such as `family`.

Details

Dyadic random effects are specified in `formula` using standard random intercept syntax, for example `(1 | pair)`. The corresponding endpoint columns are declared through `pairs`, e.g. `pairs = c("from", "to")` or `pairs = list(pair = c("from", "to"))`.

Additional ordinary `lme4` random effects may be included in the same model. The dyadic covariance contribution is rebuilt after any row subsetting so omitted pairwise observations are handled correctly.

Fixed-effect coefficients are conditional associations given the declared dyadic and ordinary random effects. Compare likelihood-based criteria only across models fitted to the same rows and by the same likelihood method.

Value

A fitted `merMod` object. Linear models return an `lmerMod`; generalized models return a `glmerMod`.

See Also

[mlpe_data()], [MLPE.lmm()]

Examples

```
y <- matrix(rnorm(25), 5)
x <- matrix(rnorm(25), 5)

dat <- mlpe_data(response = y, resistance = x)

fit <- mlpe(
  response ~ resistance + (1 | pair),
  data = dat,
  pairs = c("from", "to")
)
```

MLPE.lmm *Legacy convenience wrapper for a single MLPE resistance predictor*

Description

This function accepts a vector, matrix, `dist` object, or file path for a single pairwise resistance-distance predictor and fits a one-predictor MLPE model. It is retained for backward compatibility. For new analyses, prefer `mlpe_data` and `mlpe`, which support explicit endpoint-pair columns, omitted observations, and multiple dyadic random-intercept terms.

Usage

```
MLPE.lmm(
  resistance,
  pairwise.genetic,
  REML = FALSE,
  ID = NULL,
  ZZ = NULL,
  scale = TRUE
)
```

Arguments

`resistance`	Path to a Circuitscape resistance output file, or a vector, matrix, or `dist` object of pairwise resistance distances.
`pairwise.genetic`	Numeric genetic-distance vector in the same lower-triangle order as `resistance`.
`REML`	Logical. Fit using restricted maximum likelihood? Use `FALSE` for likelihood-based comparison of models with different fixed effects and `TRUE` for final coefficient estimation.
`ID`	Optional pair-identifier data frame from [gdist.prep()] or [jl.prep()]. It is constructed automatically for a complete set of pairs.
`ZZ`	Optional sparse MLPE design matrix corresponding to `ID`. Supply this only for a legacy workflow that requires an explicit design.

`scale` Logical. Center and standardize resistance distances before fitting? Default is `TRUE`.

Details

The resistance coefficient estimates the conditional association between genetic and resistance distance after accounting for the MLPE covariance structure. It is not, by itself, a causal effect or evidence of predictive transfer to new landscapes. When `scale = TRUE`, the coefficient is the expected response change for a one-standard-deviation increase in resistance distance.

Value

A fitted `lmerMod` object. Use `summary()` for fixed-effect estimates and uncertainty, `fitted()` and `residuals()` for diagnostics, and `AIC()` only for maximum-likelihood fits compared on the same observations.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

References

Clarke, R. T., P. Rothery, and A. F. Raybould. 2002. Confidence limits for regression relationships between distance matrices: Estimating gene flow with distance. Journal of Agricultural, Biological, and Environmental Statistics 7:361-372. doi:10.1198/108571102320

See Also

[mlpe_data()], [mlpe()], [Diagnostic.Plots()]

Examples

```
set.seed(10)
xy <- matrix(stats::runif(12), ncol = 2)
genetic <- lower(as.matrix(stats::dist(xy)))
resistance <- lower(as.matrix(stats::dist(cbind(xy[, 1], xy[, 2]^2))))

fit <- MLPE.lmm(
  resistance = resistance,
  pairwise.genetic = genetic,
  REML = TRUE
)

summary(fit)
```

mlpe_data *Build Long-Form Data for MLPE Models*

Description

Convert pairwise responses and covariates into a long-form data frame that is ready for `mlpe`. Inputs can be supplied either as symmetric matrices/`dist` objects, or as long vectors paired with an explicit two-column pair index.

Usage

```
mlpe_data(
  response,
  ...,
  pairs = NULL,
  labels = NULL,
  pair_names = c("from", "to"),
  keep = NULL
)
```

Arguments

`response`	Pairwise response as a lower-triangle vector, symmetric matrix, `dist` object, or a long vector aligned with `pairs`.
`...`	Named pairwise covariates supplied in the same format as `response`.
`pairs`	Optional data frame or matrix containing at least two columns that identify the endpoints for each observation. When supplied, `response` and all covariates are treated as long vectors aligned to these rows. Additional columns in `pairs` are preserved.
`labels`	Optional vector of endpoint labels used when converting a symmetric matrix, `dist` object, or lower-triangle vector.
`pair_names`	Names to use for the generated endpoint columns when `pairs` is not supplied.
`keep`	Optional logical/integer vector used to retain a subset of pair observations.

Details

Matrix and `dist` inputs are converted in the ordering used by [lower()]. Every response, covariate, pair row, and `keep` value must describe the same ordered pair. This function restructures data only; it does not test whether a distance response is scientifically appropriate for the intended model.

Value

A data frame with one row per retained pairwise observation. The first two columns identify pair endpoints, named by `pair_names` when generated internally. Supplied covariates retain their names, and the response is stored in `response`.

Examples

```
y <- matrix(rnorm(25), 5)
x <- matrix(rnorm(25), 5)

dat <- mlpe_data(response = y, resistance = x)

head(dat)

id <- data.frame(from = c(1, 1, 2), to = c(2, 3, 3))
mlpe_data(response = c(0.2, 0.4, 0.6), x = c(1, 2, 3), pairs = id)
```

`mlpe_rga` *Legacy MLPE wrapper with an explicit ZZ-matrix interface*

Description

This function is retained for backward compatibility and internal optimization workflows that still pass a precomputed `ZZ` matrix. For new standalone analyses, prefer `mlpe_data` and `mlpe`, which construct dyadic random effects from explicit endpoint columns and avoid manual `To.From.ID()` setup.

Usage

```
mlpe_rga(formula, data, REML = FALSE, ZZ = NULL, keep = NULL, ...)
```

Arguments

`formula`	An `lme4`-style mixed-effects formula.
`data`	Data frame containing aligned pairwise response, predictor, and endpoint variables.
`REML`	Logical. Fit a linear model by restricted maximum likelihood? Default is `FALSE`. Generalized models cannot use REML.
`ZZ`	Optional sparse MLPE design matrix. It is inferred for a complete pairwise data set when neither `ZZ` nor pair metadata is supplied.
`keep`	Optional binary or logical vector selecting rows of `data`.
`...`	Additional arguments passed to `glFormula` (e.g., `family`)

Details

Runs MLPE as detailed by Clarke et al. (2002) using a standard `lme4` formula and either inferred or supplied dyadic random-effect structure.

An AIC value will only be returned if `REML = FALSE`. In the legacy one-dyad workflow, the random effect is usually the population vector `pop1` generated by `To.From.ID`.

A generalized MLPE model can be fit if an appropriate `family`. It is not possible to use REML when fitting a generalized model.

Value

A fitted `lmerMod` or `glmerMod` object, depending on whether a generalized-model family is supplied through ....

Author(s)

Bill Peterman <Peterman.73@osu.edu>

References

Clarke, R. T., P. Rothery, and A. F. Raybould. 2002. Confidence limits for regression relationships between distance matrices: Estimating gene flow with distance. Journal of Agricultural, Biological, and Environmental Statistics 7:361-372. doi:10.1198/108571102320

See Also

[mlpe()], [mlpe_data()], [MLPE.lmm()]

Examples

```
# Prefer mlpe_data() + mlpe() for new analyses.

# Create square 'distance' matrices
y <- matrix(rnorm(25), 5)
y.pois <- matrix(rpois(25, 5), 5)
x <- matrix(rnorm(25), 5)

# Create to-from object (5 populations sampled)
id <- To.From.ID(5)

# Create data frame
df <- data.frame(y = lower(y),
                 y.pois = lower(y.pois),
                 x = lower(x),
                 pop = id$pop1)

# Fit MLPE model
out <- mlpe_rga(formula = y ~ x + (1 | pop),
                REML = TRUE,
                data = df)

# Fit generalized MLPE model
out.pois <- mlpe_rga(formula = y.pois ~ x + (1 | pop),
                     family = poisson,
                     data = df)

# Fit model with only select pairs
 keep <- c(1,1,1,1,1,1,0,1,1,0)

 out.select <- mlpe_rga(formula = y ~ x + (1 | pop),
                        data = df,
                        keep = keep)
```

MS_optim *Simultaneous optimization of multiple resistance surfaces*

Description

Optimize multiple resistance surfaces simultaneously using a genetic algorithm.

Usage

```
MS_optim(
  gdist.inputs = NULL,
  jl.inputs = NULL,
  GA.inputs,
  diagnostic_plots = TRUE
)
```

Arguments

gdist.inputs	Object from gdist.prep. Supply when optimizing with gdistance.
jl.inputs	Object from jl.prep. Supply when optimizing with Circuitscape via Julia.
GA.inputs	Object from GA.prep.
diagnostic_plots	Logical. Generate and save diagnostic plots? Default = TRUE.

Details

All supplied layers are transformed and combined during one GA search. Percent contributions are spatial averages of each transformed layer's fraction of the composite resistance surface. They describe the fitted composite and are not independent regression effect sizes. Inspect the final coefficient, response plot, and contribution plot together. Do not multiply the output raster by the contribution percentages again.

Value

A list with class resga_ms_optim. It contains the fitted GA.summary, the maximum-likelihood and REML MLPE fits (MLPE.model and MLPE.model_REML), a one-row model-fit table (AICc.tab), optimized resistance-distance matrix (cd), mean proportional surface contributions (percent.contribution), and parameter metadata (k). Optimized surfaces and tabular results are also written beneath GA.inputs$Results.dir.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[GA.prep()], [SS_optim()], [Combine_Surfaces()], [summary.resga_ms_optim()]

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")
gdist.inputs <- gdist.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = lower(Dc_list[[1]]),
  samples = pts
)

GA.inputs <- GA.prep(
  raster = raster_orig,
  Results.dir = file.path(tempdir(), "ResistanceGA2-ms"),
  pop.size = 20,
  maxiter = 10,
  run = 5,
  quiet = TRUE,
  monitor = FALSE
)

ms.out <- MS_optim(
  gdist.inputs = gdist.inputs,
  GA.inputs = GA.inputs,
  diagnostic_plots = FALSE
)

## End(Not run)
```

`Plot.trans` *Plot continuous surface transformation*

Description

Plots a transformed continuous resistance surface against the original resistance values

Usage

```
Plot.trans(
  PARM,
  Resistance,
  transformation,
  print.dir = NULL,
  marginal.plot = TRUE,
  marg.type = "histogram",
  Name = "layer"
)
```

Arguments

PARM	Parameters to transform a continuous surface or resistance values of a categorical surface. A vector of two parameters is required. The first term is the value of shape parameter (c), and the second term is the value of maximum scale parameter (b)
Resistance	A SpatRaster resistance surface, or a numeric vector of length 2 giving c(min, max) values.
transformation	Transformation equation to apply. Can be provided as the name of the transformation or its numeric equivalent (see details)
print.dir	Specify the directory where a .tiff of the transformation will be written (Default = NULL)
marginal.plot	Logical. Should distribution plots be added to the margins of the response plot (Default = TRUE). Requires that the resistance surface be specified for 'Resistance'.
marg.type	Type of marginal plot to add. One of: [density, histogram, boxplot]. See ggMarginal
Name	Name of resistance surface being transformed (optional). This will be added to the output file name.

Details

This function will create a ggplot object and plot, so it requires **ggplot2** to be installed.
Equation names can be:

1 = "Inverse-Reverse Monomolecular"
2 = "Inverse-Reverse Ricker"
3 = "Monomolecular"
4 = "Ricker"
5 = "Reverse Monomolecular"
6 = 'Reverse Ricker"
7 = "Inverse Monomolecular"
8 = "Inverse Ricker"
9 = "Distance"

The "Distance" equation sets all cell values equal to 1. Because of the flexibility of the Ricker function to take a monomolecular shape (try `Plot.trans(PARM=c(10,100), Resistance=c(1,10),transformation="Ricker` to see this), whenever a shape parameter >6 is selected in combination with a Ricker family transformation, the transformation reverts to a Distance transformation. In most applications, monomolecular families should be preferred and Ricker families should be explored only when there is a strong biological or ecological rationale for a unimodal or quadratic-type relationship. Otherwise, the extra flexibility of Ricker curves can increase the risk of overfitting. This constraint has not been implemented in the `Plot.trans` function.

Value

A `ggplot` object, or a `ggExtraPlot` object when a marginal distribution is requested. If `print.dir` is supplied, a TIFF copy is written there and the plot object is returned invisibly.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[Resistance.tran()], [GA.prep()]

Examples

```
Plot.trans(
  PARM = c(2.5, 100),
  Resistance = raster_orig[["cont_orig"]],
  transformation = "Monomolecular",
  marginal.plot = FALSE
)
```

---

raster_orig *Data file. Simulated original resistance surfaces*

Description

A `SpatRaster` with two layers generated from the `RandomFields` package. These are the raw (untransformed) surfaces used as optimization inputs.

Usage

```
data(raster_orig)
```

Format

`SpatRaster` with 2 layers

Details

- cat_orig. 5-class categorical resistance surface.
- cont_orig. Continuous resistance surface.

raster_true *Data file. Simulated true resistance surfaces*

Description

A SpatRaster with three layers generated from the RandomFields package. These surfaces were used with individual-based genetic simulations conducted using PopGenReport.

Usage

```
data(raster_true)
```

Format

SpatRaster with 3 layers

Details

- cat_true. 4-class categorical resistance surface. Reclass: 1=1, 2=75, 3=200, 4=25, 5=1.
- cont_true. Continuous resistance surface transformed via inverse monomolecular (shape = 2.75, max = 100).
- multi_true. Composite surface: sum of transformed categorical and continuous surfaces, rescaled to minimum of 1.

recommend_commute_settings
Recommend Commute-Distance Settings

Description

Recommend gdistance commute-distance settings for optimization workflows based on raster size, number of populations, and whether a selected-pair workflow is being used.

Usage

```
recommend_commute_settings(
  raster,
  n.Pops,
  keep = NULL,
  pairs_to_include = NULL
)
```

Arguments

raster	A SpatRaster used for commute-distance calculations.
n.Pops	Integer number of populations or sample locations.
keep	Optional 1/0 or logical keep vector from a gdist.prep selected-pair workflow.
pairs_to_include	Optional 1/0 or logical keep vector from a jl.prep selected-pair workflow. Supply either keep or pairs_to_include, not both.

Details

This helper is intentionally lightweight: it does not change package defaults, and it does not run any benchmarks itself. Instead, it maps a new workflow onto the benchmark regimes already represented under inst/benchmarks/:

- benchmark_gdistance_vs_julia.R compares exact gdistance::commuteDistance() runs with Julia/Circuitscape on 50, 100, and 250 cell-edge rasters using 20 populations.
- benchmark_gdistance_approx.R compares aggregate approximations on a 1000 cell-edge raster using 100 populations and aggregation factors 2 and 4.

The recommendation path is:

1. Workflows at or below the 250 x 250 / 20-population benchmark are treated as "small" and keep exact gdistance settings (commute.approx = "none").
2. Workflows between the exact and large aggregate benchmark regimes are treated as "intermediate" and use commute.approx = "aggregate" with approx.factor = 2.
3. Workflows at or above the 1000 x 1000 / 100-population benchmark are treated as "large" and use approx.factor = 4.
4. Selected-pair workflows shift the recommendation one band toward more exact settings because the package already reduces commute work when only a subset of pairs is retained.

When an aggregate approximation is recommended, rerun the final best surface once without approximation for final reporting.

Value

A named list containing the recommended commute.approx/approx.factor settings, the inferred benchmark workload band, and a short rationale.

See Also

gdist.prep, jl.prep

Examples

```
r <- terra::rast(
  nrows = 250, ncols = 250,
  xmin = 0, xmax = 250,
  ymin = 0, ymax = 250
)

recommend_commute_settings(r, n.Pops = 20)

keep <- c(rep(1L, 25), rep(0L, 20))
recommend_commute_settings(r, n.Pops = 10, keep = keep)
```

`Resist.boot` *Assess optimized resistance surfaces by population subsampling*

Description

Repeatedly partitions populations into fitting and holdout sets, refits an MLPE model for each candidate resistance-distance matrix, and summarizes how consistently each candidate ranks first.

Usage

```
Resist.boot(
  mod.names,
  dist.mat,
  n.parameters,
  sample.prop = 0.75,
  iters,
  obs,
  rank.method = "AIC",
  genetic.mat,
  keep = NULL,
  n.cores = NULL
)
```

Arguments

`mod.names`	Character vector naming the candidate surfaces. Names must correspond by position to `dist.mat` and `n.parameters`.
`dist.mat`	List of symmetric, `obs` by `obs` resistance-distance matrices from previously optimized surfaces.
`n.parameters`	Numeric vector giving the number of fitted parameters in each candidate model, including the intercept and variance parameters used for information-criterion calculations.

sample.prop	Proportion of populations sampled without replacement for model fitting in each iteration. Must leave at least two populations in at least four populations in the fitting set and two in the holdout set. Default is 0.75.
iters	Positive integer giving the number of subsampling iterations.
obs	Positive integer giving the number of populations represented by each matrix. At least six populations are required.
rank.method	Criterion used to rank candidates within each iteration. One of "AIC", "AICc", "R2", "LL", or "RMSE". Smaller values are preferred for AIC, AICc, and RMSE; larger values are preferred for marginal R-squared and log likelihood.
genetic.mat	Symmetric, obs by obs genetic-distance matrix in the same population order as every matrix in dist.mat.
keep	Optional binary vector in lower-triangle order. A value of 1 retains a pairwise observation and 0 omits it.
n.cores	Number of parallel workers. NULL or 1 runs serially. Values greater than one create a temporary PSOCK cluster, which is supported on Windows, macOS, and Linux.

Details

This procedure is a population-subsampling stability assessment, not a nonparametric bootstrap of independent observations. Pairwise distances are dependent, so whole populations are assigned to fitting or holdout sets before pairwise vectors are constructed.

AIC, AICc, marginal R-squared, and log likelihood describe fits to the sampled populations. RMSE is calculated from marginal predictions for the held-out populations. A candidate with a low average rank and a high Percent.top is comparatively stable across the sampled population sets. These summaries do not establish that a resistance surface is the true ecological process, and they should be interpreted alongside effect estimates, uncertainty, diagnostics, and a geographic-distance baseline.

Set a random seed before calling this function when reproducible population partitions are required. The same seed produces the same partitions for serial and parallel execution.

Value

A data frame with class resga_bootstrap. Each row represents a candidate surface and includes avg.AIC, avg.AICc, the mean AICc weight (avg.weight), mean within-iteration rank (avg.rank), mean marginal R-squared (avg.R2m), mean log likelihood (avg.LL), mean holdout RMSE (avg.RMSE), the count and percentage of first-place finishes (n and Percent.top), and the supplied parameter count (k).

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[MLPE.lmm()], [SS_optim()], [MS_optim()]

Examples

```
## Not run:
set.seed(42)
xy <- matrix(stats::runif(12), ncol = 2)
genetic <- as.matrix(stats::dist(xy))
distance <- as.matrix(stats::dist(xy * cbind(1.2, 0.8)))
alternative <- as.matrix(stats::dist(cbind(xy[, 1], xy[, 2]^2)))

boot.out <- Resist.boot(
  mod.names = c("distance", "alternative"),
  dist.mat = list(distance, alternative),
  n.parameters = c(4, 4),
  sample.prop = 0.67,
  iters = 20,
  obs = nrow(genetic),
  rank.method = "RMSE",
  genetic.mat = genetic,
  n.cores = 2
)

boot.out

## End(Not run)
```

---

resist_list *Data file. A list of three matrices*

Description

Sample file to be used with examples in the vignette

Usage

```
data(resist_list)
```

Format

A list of length 3

Details

Each matrix depicts the pairwise effective resistance distance (measured using ‘gdistance‘) between sample locations across ‘resist_true‘ surfaces

- resist_cat. Pairwise effective distance matrix calculated across the categorical resistance surface
- resist_cont. Pairwise effective distance matrix calculated across the continuous resistance surface
- resist_cont. Pairwise effective distance matrix calculated across the multivariate resistance surface

`Resistance.tran` *Apply a transformation to a continuous resistance surface*

Description

Applies one of eight resistance transformations (or a distance/flat transformation) to a continuous resistance surface.

Usage

```
Resistance.tran(transformation, shape, max, r, out = NULL)
```

Arguments

`transformation`	Transformation to apply. Either a numeric code (1–9) or the name of the transformation (see Details).
`shape`	Shape parameter value.
`max`	Maximum value parameter.
`r`	Resistance surface as a single-layer `SpatRaster` object.
`out`	Directory path for exporting the transformed surface as an `.asc` file. Default = `NULL` (no file written).

Details

Valid values for `transformation`:

1 = "Inverse-Reverse Monomolecular"
2 = "Inverse-Reverse Ricker"
3 = "Monomolecular"
4 = "Ricker"
5 = "Reverse Monomolecular"
6 = "Reverse Ricker"
7 = "Inverse Monomolecular"
8 = "Inverse Ricker"
9 = "Distance"

The Distance transformation sets all cell values equal to 1. Because the Ricker function can approximate a monomolecular shape at high shape parameter values, whenever a shape parameter > 6 is selected with a Ricker family transformation, the transformation automatically reverts to Distance. In most optimization workflows, monomolecular families should be treated as the recommended default, and Ricker families should only be explored when there is strong biological or ecological support for a unimodal or quadratic-type relationship. Otherwise, the additional flexibility may increase the risk of overfitting.

Value

A SpatRaster with the transformation applied.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

Examples

```
r_trans <- Resistance.tran(
  transformation = "Monomolecular",
  shape = 2.5,
  max = 100,
  r = raster_orig[["cont_orig"]]
)

r_trans
```

---

resistance_surfaces *Data file. Simulated resistance surfaces*

Description

A SpatRaster with three layers for use in vignette examples.

Usage

```
data(resistance_surfaces)
```

Format

SpatRaster with 3 layers

Details

- categorical. 3-class categorical resistance surface.
- continuous. Continuous resistance surface.
- feature. 2-class categorical (feature) resistance surface.

`ResistanceGA2-result-methods` *S3 methods for ResistanceGA2 result objects*

Description

S3 methods that provide concise printing, structured summaries, fitted response plots, and model-support plots for outputs returned by `Resist.boot`, `SS_optim`, `MS_optim`, and `all_comb`.

Usage

```
## S3 method for class 'resga_bootstrap'
print(x, n = 6L, ...)

## S3 method for class 'resga_bootstrap'
summary(object, n = 6L, ...)

## S3 method for class 'summary.resga_bootstrap'
print(x, ...)

## S3 method for class 'resga_bootstrap'
plot(
  x,
  metric = c("avg.weight", "Percent.top", "avg.rank"),
  top_n = 10L,
  main = NULL,
  ...
)

## S3 method for class 'resga_ss_optim'
print(x, n = 6L, ...)

## S3 method for class 'resga_ss_optim'
summary(object, n = 6L, ...)

## S3 method for class 'summary.resga_ss_optim'
print(x, ...)

## S3 method for class 'resga_ss_optim'
plot(
  x,
  type = c("response", "ranking"),
  metric = c("AICc", "delta.AICc", "R2m", "LL"),
  top_n = 10L,
  model = NULL,
  main = NULL,
  ...
```

```
)

## S3 method for class 'resga_ms_optim'
print(x, n = 6L, ...)

## S3 method for class 'resga_ms_optim'
summary(object, n = 6L, ...)

## S3 method for class 'summary.resga_ms_optim'
print(x, ...)

## S3 method for class 'resga_ms_optim'
plot(
  x,
  type = c("response", "contribution", "fit"),
  metric = c("mean", "AICc", "R2m", "LL"),
  top_n = 10L,
  main = NULL,
  ...
)

## S3 method for class 'resga_all_comb'
print(x, n = 6L, ...)

## S3 method for class 'resga_all_comb'
summary(object, n = 6L, ...)

## S3 method for class 'summary.resga_all_comb'
print(x, ...)

## S3 method for class 'resga_all_comb'
plot(
  x,
  metric = c("avg.weight", "Percent.top", "AICc"),
  top_n = 10L,
  main = NULL,
  ...
)
```

Arguments

x	A ResistanceGA2 result object.
n	A positive integer giving the maximum number of table rows to print.
...	Additional graphical arguments passed to `plot()` or `barplot()`.
object	A ResistanceGA2 result object.
metric	Character string naming the model-support metric to plot. Lower values are better for AIC, AICc, delta AICc, rank, and RMSE. Higher values are better for model weight, percent top-ranked, R-squared, and log-likelihood.

top_n	A positive integer giving the maximum number of models or surfaces to display.
main	Optional character string used as the plot title.
type	Plot to draw. For single-surface results, use "response" for observed response against marginal fitted values or "ranking" for a model-support bar plot. For multisurface results, use "response", "contribution", or "fit".
model	Optional model name for a single-surface response plot. The top-ranked model is used by default.

Details

The response plot compares observed values with predictions from the fixed effects of the fitted MLPE model. The dashed one-to-one line marks exact agreement. Systematic curvature or changing vertical spread indicates that the fitted mean or variance structure may be inadequate. This plot is a diagnostic, not evidence that an optimized resistance surface will predict new populations or landscapes.

Model-support plots retain the scale of the selected metric. AIC and AICc values are meaningful only within the candidate set fitted to the same response and observations. Percent contributions describe the fitted composite surface; they are not independent effect sizes and should not be applied to the component rasters a second time.

Value

The print() methods return x invisibly. The summary() methods return a named list with the class shown below:

- summary.resga_bootstrap: n_models, best_surface, metric, ranked table, and print limit n.
- summary.resga_ss_optim: model counts, top surface and AICc, run time, ranked table, the top fitted model, its fixed_effects table, and print limit n.
- summary.resga_ms_optim: top surface, fit_table, contribution_table, fitted model, fixed_effects, and print limit n.
- summary.resga_all_comb: whether the analysis contains replicated runs, the number of runs, top models by run, replicate consensus, and the single-run model and bootstrap tables when applicable.

The plot() methods return the data shown in the plot invisibly. Response plots return columns observed and fitted, where fitted excludes MLPE random effects and is on the response scale.

See Also

SS_optim, MS_optim, all_comb, Resist.boot, Diagnostic.Plots

Examples

```
ss <- readRDS(system.file(
  "extdata", "vignette-results", "ss_optim_results.rds",
  package = "ResistanceGA2"
))
```

```
summary(ss)
plot(ss, type = "response")
plot(ss, type = "ranking", metric = "AICc")

ms <- readRDS(system.file(
  "extdata", "vignette-results", "ms_optim_results.rds",
  package = "ResistanceGA2"
))
summary(ms)
plot(ms, type = "contribution")
```

`rga2_demo` *Data file. Idaho demo data for joint IBR/IBE examples*

Description

A named list bundling a small Columbia spotted frog example derived from the **GeNetIt** and **LandGenCourse** teaching materials.

Usage

```
data(rga2_demo)
```

Format

A named list of length 5

Details

- sites. A 12-row data frame of selected sites from `GeNetIt::ralu.site`, including `demo_id`, the original site attributes, and projected UTM coordinates (`x`, `y`).
- sample_coords. A 12 x 2 numeric matrix of sample coordinates in the same order as the genetic matrix.
- site_environment. A data frame of raster values extracted at the 12 sites from `rga2_demo_covariates`.
- genetic_matrix. A 12 x 12 symmetric chord-distance matrix imported from the LandGenCourse radish example.
- genetic_vector. The lower-triangle vector of `genetic_matrix` in package order.

See `data-raw/build_rga2_demo.R` for the pinned source revisions used to construct this demo.

rga2_demo_covariates *Data file. Idaho demo raster covariates*

Description

A `SpatRaster` containing the six Idaho raster layers used to construct `rga2_demo`. The layers are derived from **GeNetIt** and include `cti`, `err27`, `ffp`, `gsp`, `hli`, and `nlcd`.

Usage

```
data(rga2_demo_covariates)
```

Format

`SpatRaster` with 6 layers

rga2_demo_genetic *Data file. Idaho demo genetic distance matrix*

Description

A 12 x 12 symmetric chord-distance matrix used in `rga2_demo`. Row and column names match the `demo_id` values in `rga2_demo$sites`.

Usage

```
data(rga2_demo_genetic)
```

Format

A 12 x 12 numeric matrix

Run_CS.ini *Run Circuitscape via Julia with direct INI-style options*

Description

A low-level wrapper around **Circuitscape.jl** that writes a complete Circuitscape `.ini` file from R, runs `compute()`, and returns the raw result plus any generated output files.

Usage

```
Run_CS.ini(
  r = NULL,
  CS_Point.File = NULL,
  JULIA_HOME = NULL,
  EXPORT.dir = NULL,
  mask_file = NULL,
  source_file = NULL,
  ground_file = NULL,
  included_pairs_file = NULL,
  reclass_file = NULL,
  cs_options = list(),
  return = c("all", "compute", "matrix", "current_map"),
  rm.files = FALSE,
  quiet = TRUE,
  scratch = NULL,
  is_resistance = TRUE,
  sanitize_raster = FALSE,
  ...
)
```

Arguments

`r`	A single-layer `SpatRaster` or a path to an existing raster file. If `NULL`, supply `habitat_file` via `cs_options` or `....`
`CS_Point.File`	A `SpatVector` of point locations or a path to an existing Circuitscape point file. Required for raster `pairwise`, `one-to-all`, and `all-to-one` scenarios unless `point_file` is overridden directly.
`JULIA_HOME`	Path to the Julia `bin/` directory. If `NULL`, `JuliaConnectoR` uses the current Julia configuration / environment.
`EXPORT.dir`	Directory where the temporary `.ini`, `.out`, and any Circuitscape output rasters are written.
`mask_file`	Optional Circuitscape `mask_file`. May be a single-layer `SpatRaster` or an existing file path. When supplied, `use_mask` defaults to `TRUE` unless explicitly overridden.

source_file — Optional Circuitscape source_file. May be a single-layer SpatRaster, a point SpatVector with a value attribute, a numeric table in Circuitscape text-list format, or an existing file path.

ground_file — Optional Circuitscape ground_file. Same accepted forms as source_file. When using raster advanced mode and a raster is supplied, it is written as an ASCII grid.

included_pairs_file — Optional Circuitscape included_pairs_file. May be an existing file path, a 2-column table of node pairs (defaults to include mode), or a list with elements mode = "include"/"exclude" and pairs = <2-column table>. When supplied, use_included_pairs defaults to TRUE unless explicitly overridden.

reclass_file — Optional Circuitscape reclass_file. May be an existing file path or a numeric table, which is written to a temporary tab-delimited file as-is. When supplied, use_reclass_table defaults to TRUE unless explicitly overridden.

cs_options — Named list of Circuitscape .ini option overrides. Names should match Circuitscape keys exactly (for example, scenario, write_cur_maps, use_mask, mask_file).

return — What to return: "all" (default) returns a list with the raw Julia result, parsed pairwise matrix when available, current map when available, the final config, and file paths; "compute" returns the raw Julia result; "matrix" returns a parsed pairwise matrix when available; "current_map" returns the cumulative current map raster.

rm.files — Logical. Remove generated .ini, temporary inputs, and Circuitscape outputs after the run? Default = FALSE.

quiet — Logical. Suppress Circuitscape progress output emitted through Julia? Default = TRUE.

scratch — Optional scratch directory for temporary raster / point files.

is_resistance — Logical. Should the input habitat map be treated as resistances (TRUE) or conductances (FALSE) by default?

sanitize_raster — Logical. If TRUE, apply the same raster sanitizing used by Run_CS.jl before writing the temporary ASCII file. Default = FALSE.

... — Additional Circuitscape .ini option overrides supplied as named arguments. These are merged with cs_options.

Details

This function complements Run_CS.jl. Whereas Run_CS.jl is tailored to the package optimization workflow, Run_CS.ini is intended for direct Circuitscape runs where the user wants to specify Circuitscape settings in a way that closely mirrors the .ini file.

Option names in cs_options and ... should use the same keys that appear in a Circuitscape .ini file. The function currently exposes the full raster-mode option template written by the package for Circuitscape 4.0.5 and validates that overridden keys are known.

The file-oriented arguments (mask_file, source_file, ground_file, included_pairs_file, and reclass_file) are convenience layers over the corresponding Circuitscape options. They

are especially useful when working directly with terra objects or small R tables. For any more specialized file layout, users can still supply a file path directly through these arguments or through `cs_options`.

Value

Depending on `return`, either the raw Julia result, a parsed pairwise matrix, a cumulative current-map raster, or a list containing all of those plus the written `.ini` file path, generated output files, and the final Circuitscape configuration used for the run.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")

cs.out <- Run_CS.ini(
  r = raster_orig[["cont_orig"]],
  CS_Point.File = pts,
  JULIA_HOME = Sys.getenv("JULIA_BINDIR"),
  scenario = "pairwise",
  connect_four_neighbors_only = TRUE,
  write_cur_maps = TRUE,
  write_cum_cur_map_only = TRUE,
  rm.files = FALSE
)

## End(Not run)
```

Run_CS.jl *Run Circuitscape via Julia*

Description

Executes Circuitscape through Julia (via **JuliaConnectoR**) and returns pairwise resistance distances.

Usage

```
Run_CS.jl(
  jl.inputs = NULL,
  r,
  CurrentMap = FALSE,
  full.mat = FALSE,
  EXPORT.dir = NULL,
  output = "matrix",
```

```
  CS_Point.File = NULL,
  pairs_to_include = NULL,
  pop2ind = NULL,
  parallel = FALSE,
  cores = NULL,
  cholmod = TRUE,
  precision = FALSE,
  JULIA_HOME = NULL,
  rm.files = TRUE,
  scratch = NULL,
  is_resistance = TRUE
)
```

Arguments

`jl.inputs`	Object from `jl.prep`. When provided, all other Julia/CS settings are taken from this object.
`r`	A single-layer `SpatRaster` (from **terra**).
`CurrentMap`	Logical. Generate the cumulative current map? Default = `FALSE`.
`full.mat`	Logical. Return the full square distance matrix instead of only the lower triangle? Default = `FALSE`. Cannot be requested when `pairs_to_include` is used.
`EXPORT.dir`	Directory where Circuitscape results are written. Required when `CurrentMap` = `TRUE`. If the directory does not exist, it will be created automatically.
`output`	`"matrix"` (default) returns the lower-half pairwise resistance vector; `"raster"` returns a `SpatRaster` of the cumulative current map (requires `CurrentMap` = `TRUE`).
`CS_Point.File`	Sample locations. Only needed when `jl.inputs` is not provided. See `jl.prep`.
`pairs_to_include`	1/0 vector selecting pairs. Only needed when `jl.inputs` is not provided.
`pop2ind`	Per-population sample-size vector. Only needed when `jl.inputs` is not provided.
`parallel`	Logical. Run Circuitscape in parallel? Default = `FALSE`.
`cores`	Number of cores for parallel runs.
`cholmod`	Logical. Use CHOLMOD solver? Default = `TRUE`.
`precision`	Logical. Use single precision? Default = `FALSE`.
`JULIA_HOME`	Path to the Julia `bin/` directory. Only needed when `jl.inputs` is not provided.
`rm.files`	Logical. Remove temporary files after the run? Default = `TRUE`.
`scratch`	Scratch directory for temporary files.
`is_resistance`	Logical. Is `r` a resistance (vs conductance) surface? Default = `TRUE`.

Details

Users should supply terra objects directly. The raster and point files needed by Circuitscape are written to temporary files internally for each run.

Value

Lower-half pairwise resistance vector, a full square matrix (`full.mat = TRUE`), or a `SpatRaster` current map (`output = "raster"`).

Author(s)

Bill Peterman <Peterman.73@osu.edu>

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")
jl.inputs <- jl.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = lower(Dc_list[[1]]),
  CS_Point.File = pts,
  JULIA_HOME = Sys.getenv("JULIA_BINDIR"),
  run_test = FALSE
)

cs.result <- Run_CS.jl(jl.inputs = jl.inputs, r = raster_orig[["cont_orig"]])

## End(Not run)
```

Run_gdistance *Run gdistance cost/commute distance calculation*

Description

Calculates pairwise cost or commute distances across a resistance surface using the `gdistance` package.

Usage

```
Run_gdistance(
  gdist.inputs,
  r,
  scl = TRUE,
  return.error.value = FALSE,
  commute.approx = NULL,
  approx.factor = NULL,
  approx.scale = NULL
)
```

Arguments

gdist.inputs	Object created by gdist.prep.
r	A single-layer SpatRaster object (from the **terra** package).
scl	Logical. Scale the correction values (default TRUE). Set to FALSE to obtain absolute distance values. See geoCorrection for details.
return.error.value	Logical. If TRUE, return -99999 on warnings/errors instead of throwing an error. This is intended for internal GA fitness evaluation, where failed candidate surfaces should be penalized but optimization should continue. Default is FALSE for user-facing calls.
commute.approx	Optional override for the commute-distance approximation stored in gdist.inputs. NULL uses the setting from gdist.prep. "none" runs the exact gdistance commute solve; "aggregate" coarsens r first for faster iterative optimization.
approx.factor	Optional integer aggregation factor override used when commute.approx = "aggregate".
approx.scale	Optional logical override. If TRUE, aggregate approximation distances are multiplied by approx.factor^2.

Details

gdistance still operates on **raster** objects. This function keeps the public API terra-based by coercing the supplied SpatRaster internally before building the transition object. The default transition function uses vectorized sparse builders for projected 4- and 8-neighbor transition matrices and projected geographic corrections. Selected-pair runs reuse a single distance calculation over the required endpoints. Commute-distance runs solve the reduced Laplacian for all requested endpoints in one batch.

Value

A numeric distance vector (or matrix) of pairwise cost/commute distances. If return.error.value = TRUE, returns -99999 on error or warning.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

Examples

```
pts <- terra::vect(samples[, 2:3], type = "points")
gdist.inputs <- gdist.prep(
  n.Pops = nrow(samples),
  samples = pts,
  method = "costDistance"
)

r_surface <- terra::subset(terra::unwrap(resistance_surfaces), "continuous")
cd <- Run_gdistance(gdist.inputs, r_surface)
length(cd)
```

sample_pops *Data file. A list of three coordinate matrices*

Description

50 sample locations per list element corresponding to populations used in genetic simulations. Each element is a two-column numeric matrix (x, y).

Usage

```
data(sample_pops)
```

Format

A list of length 3; each element is a 50 x 2 coordinate matrix.

Details

- sample_cat. Sample locations for categorical surface simulation.
- sample_cont. Sample locations for continuous surface simulation.
- sample_multi. Sample locations for multi-surface simulation.

samples *Data file. Example sample location file*

Description

Sample file to be used with examples in the vignette

Usage

```
data(samples)
```

Format

A 25 x 3 data frame

Details

A data frame that can be saved as a .txt file for running examples in the vignette

SHNe *Calculate pairwise spatial heterogeneity in effective population size*

Description

Calculates the pairwise `di` and `dhm` covariates described by Prunier et al. (2017) from site-level effective population sizes.

Usage

```
SHNe(n.samples, pop.size, output = c("df", "matrix"))
```

Arguments

`n.samples`	Positive integer giving the number of sampled populations.
`pop.size`	Positive numeric vector of effective population sizes, ordered to match the populations used in all other pairwise inputs.
`output`	Output representation: `"df"` returns lower-triangle vectors and `"matrix"` returns square matrices.

Details

For populations i and j, the function calculates $d_{hm} = -2N_iN_j/(N_i + N_j)$ and $d_i = (N_i + N_j)/(N_iN_j)$. These are demographic covariates. Their coefficients in an MLPE model describe conditional associations and should not be interpreted as causal effects without an appropriate study design.

Value

With `output = "df"`, a data frame with `choose(n.samples, 2)` rows and columns `dhm` and `di`, in [lower()] order. With `output = "matrix"`, a named list of two `n.samples` square matrices whose lower triangles contain those values.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

References

Prunier, J. G., V. Dubut, L. Chikhi, and S. Blanchet. 2017. Contribution of spatial heterogeneity in effective population sizes to the variance in pairwise measures of genetic differentiation. Methods in Ecology and Evolution 8:1866-1877. doi:10.1111/2041210X.12820

See Also

[site_pairwise_diff()], [mlpe_data()], [mlpe()]

Examples

```
Ne <- rep(c(5, 25, 100), each = 3)
SHNe(n.samples = length(Ne), pop.size = Ne)

# Return as list of matrices
SHNe(n.samples = length(Ne), pop.size = Ne, output = "matrix")
```

`site_pairwise_diff` *Build Pairwise Environmental-Difference Covariates*

Description

Convert site-level environmental variables into pairwise covariates that are aligned with the lower-triangle ordering used throughout **ResistanceGA2**.

Usage

```
site_pairwise_diff(
  data,
  vars = NULL,
  scale = FALSE,
  transform = c("absolute", "squared"),
  keep = NULL,
  pop_n = NULL,
  suffix = NULL
)
```

Arguments

`data`	Numeric vector, matrix, or data frame of site-level environmental variables.
`vars`	Optional character or integer index selecting columns from `data`.
`scale`	Logical. If `TRUE`, standardize each site-level variable before computing pairwise differences. Constant variables are converted to zeros.
`transform`	Transformation applied to pairwise differences: `"absolute"` (default) or `"squared"`.
`keep`	Optional logical/integer vector used to retain a subset of the final pairwise observations.
`pop_n`	Optional integer vector giving the number of sampled individuals per population. When supplied, population-level environmental differences are expanded to the individual-level pair structure used by `jl.prep` with `pop2ind`.
`suffix`	Optional suffix appended to output column names. Defaults to `"_diff"` for absolute differences and `"_sqdiff"` for squared differences.

Details

These helpers are intended for isolation-by-environment workflows, where environmental mismatch between sample locations is used as one or more fixed effects in an MLPE model. The output can be supplied directly to the `covariates` argument of `gdist.prep` or `jl.prep`.

Value

A data frame of pairwise environmental-difference covariates.

Examples

```
env <- data.frame(
  temp = c(10, 12, 15, 11),
  precip = c(100, 120, 80, 110)
)

site_pairwise_diff(env)
site_pairwise_diff(env, vars = "temp", scale = TRUE)
```

---

site_pairwise_dist *Build Pairwise Environmental-Distance Covariates*

Description

Convert one or more site-level environmental variables into a single pairwise environmental-distance covariate aligned with the lower-triangle ordering used throughout **ResistanceGA2**.

Usage

```
site_pairwise_dist(
  data,
  vars = NULL,
  scale = TRUE,
  method = "euclidean",
  keep = NULL,
  pop_n = NULL,
  name = "env_dist",
  ...
)
```

Arguments

data	Numeric vector, matrix, or data frame of site-level environmental variables.
vars	Optional character or integer index selecting columns from data.
scale	Logical. If TRUE, standardize each site-level variable before calculating environmental distance. Constant variables are converted to zeros.
method	Distance metric passed to dist.
keep	Optional logical/integer vector used to retain a subset of the final pairwise observations.
pop_n	Optional integer vector giving the number of sampled individuals per population. When supplied, population-level environmental distances are expanded to the individual-level pair structure used by jl.prep with pop2ind.

name	Name of the output covariate column. Default = "env_dist".
...	Additional arguments passed to dist, such as p when method = "minkowski".

Value

A one-column data frame containing pairwise environmental distance.

Examples

```
env <- data.frame(
  temp = c(10, 12, 15, 11),
  precip = c(100, 120, 80, 110)
)

site_pairwise_dist(env)
site_pairwise_dist(env, method = "manhattan", name = "climate_dist")
```

SS_optim *Single surface optimization*

Description

Optimize each surface in a SpatRaster stack using a genetic algorithm executed with the ga function in **GA**.

Usage

```
SS_optim(
  gdist.inputs = NULL,
  jl.inputs = NULL,
  GA.inputs,
  dist_mod = TRUE,
  null_mod = TRUE,
  diagnostic_plots = TRUE
)
```

Arguments

gdist.inputs	Input list returned by [gdist.prep()]. Supply exactly one distance backend.
jl.inputs	Input list returned by [jl.prep()]. Supply exactly one distance backend.
GA.inputs	Settings and resistance surfaces returned by [GA.prep()].
dist_mod	Logical. Add a geographic-distance baseline to the common model comparison? Default is TRUE.
null_mod	Logical. Add an intercept-only baseline? Default is TRUE.
diagnostic_plots	Logical. Save [Diagnostic.Plots()] for optimized candidates? Default is TRUE.

Details

Each raster layer is optimized independently. AICc comparisons are meaningful only within the returned table, where candidates use the same response and retained observations. The smallest AICc identifies the best-supported candidate among those evaluated, not a uniquely true resistance process. Use `summary()` to inspect the leading model and its fixed effects, and `plot(..., type = "response")` to assess its response relationship.

Value

A list with class `resga_ss_optim`. It contains continuous and categorical optimization tables, the common model-comparison table (`AICc`), coefficient table (`MLPE`), elapsed `Run.Time`, fitted REML models by candidate (`MLPE.list`), optimized pairwise distance matrices (`cd`), parameter metadata (`k`), and fitted GA objects (`ga`). CSV tables, fitted objects, optimized surfaces, and optional diagnostics are written beneath `GA.inputs$Results.dir`.

Author(s)

Bill Peterman <Peterman.73@osu.edu>

See Also

[GA.prep()], [MS_optim()], [Resist.boot()], [summary.resga_ss_optim()]

Examples

```
## Not run:
pts <- terra::vect(sample_pops[[1]], type = "points")
gdist.inputs <- gdist.prep(
  n.Pops = nrow(sample_pops[[1]]),
  response = lower(Dc_list[[1]]),
  samples = pts
)

GA.inputs <- GA.prep(
  raster = raster_orig,
  Results.dir = file.path(tempdir(), "ResistanceGA2-ss"),
  pop.size = 20,
  maxiter = 10,
  run = 5,
  quiet = TRUE,
  monitor = FALSE
)

ss.out <- SS_optim(
  gdist.inputs = gdist.inputs,
  GA.inputs = GA.inputs,
  dist_mod = FALSE,
  null_mod = FALSE,
  diagnostic_plots = FALSE
)
```

```
## End(Not run)
```

`vignette_current_map` *Data file. Precomputed Circuitscape current map used in vignette examples*

Description

A `SpatRaster` cumulative current map produced by `Run_CS.jl` for the `"continuous"` layer of `resistance_surfaces` (after `terra::unwrap()` when needed) and `samples`.

Usage

```
data(vignette_current_map)
```

Format

`SpatRaster` with 1 layer

`vignette_pairwise_outputs`
Data file. Precomputed pairwise outputs used in vignette examples

---

Description

A named list containing lightweight, precomputed outputs from the current terra-based workflow.

Usage

```
data(vignette_pairwise_outputs)
```

Format

A named list of length 2

Details

- gdistance_commute. Full square commute-distance matrix returned by `Run_gdistance` for the `"continuous"` layer of `resistance_surfaces` (after `terra::unwrap()` when needed) and `samples`.
- circuitscape_matrix. Lower-half pairwise resistance vector returned by `Run_CS.jl` for the same surface and sample locations.

Index

∗ **datasets**
  Dc_list, 7
  jl_out, 20
  raster_orig, 31
  raster_true, 32
  resist_list, 36
  resistance_surfaces, 38
  rga2_demo, 42
  rga2_demo_covariates, 43
  rga2_demo_genetic, 43
  sample_pops, 50
  samples, 50
  vignette_current_map, 56
  vignette_pairwise_outputs, 56

all_comb, 3, *10*, *39*, *41*

Combine_Surfaces, 5
commuteDistance, *15*
costDistance, *15*

Dc_list, 7
Diagnostic.Plots, 7, *41*
dist, *53*, *54*

family, *26*

ga, *54*
GA.prep, *3*, *5*, 9, *28*
gdist.prep, *3*, *5*, 14, *16*, *28*, *33*, *49*, *52*
geoCorrection, *15*, *49*
ggMarginal, *30*
glFormula, *22*, *26*
graph_to_keep, *15*, 16, *19*

jl.prep, *3*, *5*, *16*, 18, *28*, *33*, *47*, *52*, *53*
jl_out, 20

lower, *17*, 21

mlpe, 22, *23*, *25*, *26*
MLPE.lmm, 23
mlpe_data, *23*, 25, *26*
mlpe_rga, 26
MS_optim, *3*, *5*, 28, *39*, *41*

optim, *12*

plot.resga_all_comb
  *(*ResistanceGA2-result-methods*)*, 39
plot.resga_bootstrap
  *(*ResistanceGA2-result-methods*)*, 39
plot.resga_ms_optim
  *(*ResistanceGA2-result-methods*)*, 39
plot.resga_ss_optim
  *(*ResistanceGA2-result-methods*)*, 39
Plot.trans, 29
print.resga_all_comb
  *(*ResistanceGA2-result-methods*)*, 39
print.resga_bootstrap
  *(*ResistanceGA2-result-methods*)*, 39
print.resga_ms_optim
  *(*ResistanceGA2-result-methods*)*, 39
print.resga_ss_optim
  *(*ResistanceGA2-result-methods*)*, 39
print.summary.resga_all_comb
  *(*ResistanceGA2-result-methods*)*, 39
print.summary.resga_bootstrap
  *(*ResistanceGA2-result-methods*)*, 39
print.summary.resga_ms_optim
  *(*ResistanceGA2-result-methods*)*,

39
print.summary.resga_ss_optim
(ResistanceGA2-result-methods), 39

raster_orig, 31
raster_true, 32
recommend_commute_settings, *15*, 32
Resist.boot, *3*, 34, *39*, *41*
resist_list, 36
Resistance.tran, *13*, 37
resistance_surfaces, 38, *56*
ResistanceGA2-result-methods, 39
rga2_demo, 42
rga2_demo_covariates, 43
rga2_demo_genetic, 43
Run_CS.ini, 44
Run_CS.jl, *45*, 46, *56*
Run_gdistance, *15*, 48, *56*

sample_pops, 50
samples, 50
SHNe, 51
site_pairwise_diff, 52
site_pairwise_dist, 53
SpatRaster, *33*
SpatVector, *14*, *18*, *44*
SS_optim, *3*, *39*, *41*, 54
summary.resga_all_comb
(ResistanceGA2-result-methods), 39
summary.resga_bootstrap
(ResistanceGA2-result-methods), 39
summary.resga_ms_optim
(ResistanceGA2-result-methods), 39
summary.resga_ss_optim
(ResistanceGA2-result-methods), 39

To.From.ID, *17*, *26*
transition, *15*

vignette_current_map, 56
vignette_pairwise_outputs, 56