Package 'graph4lg'

July 23, 2019

Type Package

Title Build Graphs for Landscape Genetics Analysis

Version 0.1.1

Date 2019-07-18

Author Paul Savary

Maintainer Paul Savary <savarypaul660@gmail.com>

Description Build graphs for landscape genetics analysis. This set of functions can be used to import and convert spatial and genetic data initially in different formats, import landscape graphs created with 'GRAPHAB' software (Foltete et al., 2012) <doi:10.1016/j.envsoft.2012.07.002>, make diagnosis plots of isolation by distance relationships in order to choose how to build genetic graphs, create graphs with a large range of pruning methods, weight their links with several genetic distances, plot and analyse graphs,compare them with other graphs. It uses functions from other packages such as 'adegenet' (Jombart, 2008) <doi:10.1093/bioinformatics/btn129> and 'igraph' (Csardi et Nepusz, 2006) <https://bit.ly/2028mcO>. It also implements methods commonly used in landscape genetics to create graphs, described by Dyer et Nason (2004) <doi:10.1111/j.1365-294X.2004.02177.x> and Greenbaum et Fefferman (2017) <doi:10.1111/mec.14059>, and to analyse distance data (van Strien et al., 2015) <doi:10.1038/hdy.2014.62>.

Depends R(>= 3.1.0)

License GPL-2

Encoding UTF-8

LazyData true

Imports stringr, adegenet, ade4, stats, Matrix, vegan, utils, methods, pegas, Rdpack, MASS, igraph, doBy, ggplot2, mclust, tidyr, sp, rgdal, sf, Imap, diveRsity, raster, ecodist

RdMacros Rdpack

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-07-23 09:30:02 UTC

R topics documented:

`add_nodes_attr` *Add attributes to the nodes of a graph*

Description

The function adds attributes to the nodes of a graph from either an object of class `data.frame` or from a shapefile layer. The nodes' IDs in the input objects must be the same as in the graph object.

Usage

```
add_nodes_attr(graph, input = "df", data, dir_path = NULL,
  layer = NULL, index = "Id", include = "all")
```

Arguments

`graph`	A graph object of class `igraph`.
`input`	A character string indicating the nature of the input data from which come the attributes to add to the nodes. • If 'input = "shp"', then attributes come from the attribute table of a shapefile layer of type point. • If 'input = "df"', then attributes come from an object of class `data.frame` In both cases, input attribute table or dataframe must have a column with the exact same values as the nodes' IDs.
`data`	(only if 'input = "df"') The name of the object of class `data.frame` with the attributes to add to the nodes.
`dir_path`	(only if 'input = "shp"') The path (character string) to the directory containing the shapefile layer of type point whose attribute table contains the attributes to add to the nodes.
`layer`	(only if 'input = "shp"') The name (character string) of the shapefile layer of type point (without extension, ex.: "nodes" refers to "nodes.shp" layer) whose attribute table contains the attributes to add to the nodes.
`index`	The name (character string) of the column with the nodes names in the input data (column of the attribute table or of the dataframe).
`include`	A character string (vector) indicating which columns of the input data will be added as nodes' attributes. By default, 'include = "all"', i.e. every column of the input data is added. Alternatively, 'include' can be a vector with the names of the columns to add (ex.: "c('x', 'y', 'pop_name')").

Details

The graph can be created with the function `graphab_to_igraph` from shapefile layers created with GRAPHAB. Values of the metrics computed at the node level with GRAPHAB can then be added to such a graph with this function.

Value

A graph object of class `igraph`

Author(s)

P. Savary

Examples

```
path <- system.file('extdata', package = 'graph4lg')
links <- "liens_rast2_1_11_01_19-links"
graph <- graphab_to_igraph(dir_path = path,
                           links = links,
                           fig = FALSE)
df_nodes <- data.frame(Id = igraph::V(graph)$name,
                       Area = runif(50, min = 10, max = 60))
graph <- add_nodes_attr(graph,
                        data = df_nodes,
                        input = "df",
                        index = "Id",
                        include = "Area")
```

---

compar_r_fisher *Compare two correlation coefficients obtained from different sample sizes*

Description

The function compares two correlation coefficients obtained from different sample sizes using Z-Fisher transformation.

Usage

```
compar_r_fisher(data)
```

Arguments

`data`	An object of class `data.frame` with at least 4 columns of data used to perform the test. 4 columns must be called "n1", "n2", "r1" and "r2". • n1 and n2 are the sizes of the samples from which r1 and r2 were computed respectively. • r1 and r2 are Pearson's correlation coefficients

Details

The Z-Fisher method consists in computing z scores from the correlation coefficients and to compare these z scores. z scores are computed as follows : Let n1 and r1 be the sample size and the correlation coefficient, z1 = (1/2)*log((1+r1) / (1-r1)) Then, a test's statistic is computed from z1 and z2 : Z = (z1-z2) / sqrt((1/(n1-3)) + (1/(n2-3))) If Z is above the limit given by the alpha value, then the difference between r1 and r2 is significant

Value

An object of class `data.frame` with the same columns as 'data' and 4 columns more : z1, z2 (respective z-scores), Z (test's statistic) and p (p-value) of the test.

Author(s)

P. Savary

Examples

```
df <- data.frame(n1 = rpois(n = 40, lambda = 85),
                 n2 = rpois(n = 40, lambda = 60),
                 r1 = runif(n = 40, min = 0.6, max = 0.85),
                 r2 = runif(n = 40, min = 0.55, max = 0.75))
data <- compar_r_fisher(df)
```

`convert_cd` *Fit a model to convert cost-distances into Euclidean distances*

Description

The function fits a model to convert cost-distances into Euclidean distances as implemented in GRAPHAB software.

Usage

```
convert_cd(mat_euc, mat_ld, to_convert, method = "log-log", fig = TRUE,
  line_col = "black", pts_col = "#999999")
```

Arguments

`mat_euc`	A symmetric `matrix` with pairwise geographical Euclidean distances between populations or sample sites. It will be the explanatory variable, and only values from the off diagonal lower triangle will be used.
`mat_ld`	A symmetric `matrix` with pairwise landscape distances between populations or sample sites. These distances can be cost-distances or resistance distances, among others. It will be the explained variable, and only values from the off diagonal lower triangle will be used.
`to_convert`	A numeric value or numeric vector with Euclidean distances to convert into cost-distances.
`method`	A character string indicating the method used to fit the model. • If 'method = "log-log"' (default), then the model takes the following form : log(ld) ~ A + B * log(euc) • If 'method = "lm"', then the model takes the following form : ld ~ A + B * euc
`fig`	Logical (default = TRUE) indicating whether a figure is plotted

`line_col` (if 'fig = TRUE') Character string indicating the color used to plot the line (default: "blue"). It must be a hexadecimal color code or a color used by default in R.

`pts_col` (if 'fig = TRUE') Character string indicating the color used to plot the points (default: "#999999"). It must be a hexadecimal color code or a color used by default in R.

Details

IDs in 'mat_euc' and 'mat_ld' must be the same and refer to the same sampling site or populations, and both matrices must be ordered in the same way. Matrix of Euclidean distance 'mat_euc' can be computed using the function `mat_geo_dist`. Matrix of landscape distance 'mat_ld' can be computed using GRAPHAB software. Before the log calculation, 0 distance values are converted into 1, so that they are 0 after this calculation.

Value

A list of output (converted values, estimated parameters, R2) and optionally a ggplot2 object to plot

Author(s)

P. Savary

References

FoltÃªte J, Clauzel C, Vuidel G (2012). "A software tool dedicated to the modelling of landscape networks." *Environmental Modelling \& Software*, **38**, 316–327.

Examples

```
path <- system.file('extdata', package = 'graph4lg')
links <- "liens_rast2_1_11_01_19-links"
graph <- graphab_to_igraph(dir_path = path,
                           links = links,
                           fig = FALSE)
mat_ld <- igraph::get.adjacency(graph, attr = "weight",
                                type = "both", sparse = FALSE )
pts <- rgdal::readOGR(dsn = path, layer = "patches")
pts <- data.frame(sp::coordinates(pts))
pts$ID <- as.character(1:50)
names(pts) <- c("x", "y", "ID")
pts <- pts[, c("ID", "x", "y")]

mat_euc <- suppressWarnings(mat_geo_dist(data= pts,
      ID = "ID",
      x = "x",
      y = "y"))
 to_convert <- c(15000, 20000, 25000)
 res <- convert_cd(mat_euc = mat_euc,
 mat_ld = mat_ld, to_convert = to_convert)
```

data_pc_genind *data_pc_genind genetic dataset*

Description

Setophaga plumbea genetic dataset Data collected in French Guadeloupe by A. Khimoun and S. Garnier 432 individuals, 20 populations 12 microsatellite loci (3 digits coding) 169 different alleles

Usage

```
data_pc_genind
```

Format

An object of type 'genind'

Source

https://datadryad.org/resource/doi:10.5061/dryad.v8324

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("data_pc_genind")
length(unique(data_pc_genind@pop))
```

data_pc_gpop *data_pc_gpop genetic dataset*

Description

Setophaga plumbea genetic dataset Data collected in French Guadeloupe by A. Khimoun and S. Garnier 432 individuals, 20 populations 12 microsatellite loci (3 digits coding) 169 different alleles

Usage

```
data_pc_gpop
```

Format

A 'data.frame' with the format needed to save as a GENEPOP software file :

1st column Individual identifier beginning with population name and ending with a comma

Other columns 12 loci's columns with microsatellites' data with 3 digits coding, alleles not separated, and missing data noted "000000"

Each group of individuals from the same population are separated by a line with only the character value "Pop" in firts column. First lines include loci names and file name

Source

https://datadryad.org/resource/doi:10.5061/dryad.v8324

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("data_pc_gpop")
str(data_pc_gpop)
```

---

data_pc_gstud *data_pc_gstud genetic dataset*

Description

Setophaga plumbea genetic dataset Data collected in French Guadeloupe by A. Khimoun and S. Garnier 432 individuals, 20 populations 12 microsatellite loci (3 digits coding) 169 different alleles

Usage

```
data_pc_gstud
```

Format

A 'data.frame' with columns:

Species Species identifier: in this case 'PC'

Cluster Population name

ID Individual ID

id Population ID used during the sampling stage

Latitude Sampling site's (population) latitude (WGS 84 - UTM 20 N)

Longitude Sampling site's (population) longitude (WGS 84 - UTM 20 N)

DkiD102 to DkiD12 12 loci's columns with microsatellites' data with 3 digits coding, alleles separated by ":", and blank missing data (class 'locus' from **gstudio**)

Source

https://datadryad.org/resource/doi:10.5061/dryad.v8324

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("data_pc_gstud")
str(data_pc_gstud)
length(unique(data_pc_gstud$Cluster))
```

---

data_pc_loci *data_pc_loci genetic dataset*

Description

Setophaga plumbea genetic dataset Data collected in French Guadeloupe by A. Khimoun and S. Garnier 432 individuals, 20 populations 12 microsatellite loci (3 digits coding) 169 different alleles

Usage

```
data_pc_loci
```

Format

An object of class 'loci' and 'data.frame' with the columns :

population Population name

Other columns 12 loci's columns with microsatellites' data with 3 digits coding, alleles separated by "/", and missing data noted "NA/NA"

Row names correspond to individuals' ID

Source

https://datadryad.org/resource/doi:10.5061/dryad.v8324

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("data_pc_loci")
length(unique(data_pc_loci$population))
```

`data_pc_str` *data_pc_str genetic dataset*

Description

Setophaga plumbea genetic dataset Data collected in French Guadeloupe by A. Khimoun and S. Garnier 432 individuals, 20 populations 12 microsatellite loci (3 digits coding) 169 different alleles

Usage

```
data_pc_str
```

Format

An object of class 'data.frame' ready to be saved as a STRUCTRE software file. The resulting text file would include the following columns :

1st column Individual names

2nd column Population names

Other columns 12 loci's columns with microsatellites' data with 3 digits coding, and missing data noted "-9"

Each individual is displayed on two rows, with one allele on each row. There are as many rows as twice the number of individuals.

Source

`https://datadryad.org/resource/doi:10.5061/dryad.v8324`

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("data_pc_str")
```

data_simul_genind *data_simul_genind genetic dataset*

Description

Genetic dataset from genetic simulation on CDPOP 1500 individuals, 50 populations 20 microsatellite loci (3 digits coding) 50 generations simulated

Usage

```
data_simul_genind
```

Format

An object of type 'genind'

Details

The simulation was made with CDPOP during 50 generations. Dispersal was possible between the 50 populations. Its probability depended on the cost distance between populations, calculated on a simulated resistance surface (raster). Mutations were not possible. There were initially 600 alleles in total (many disappeared because of drift). Population stayed constant with a sex-ratio of 1. Generations did not overlap. This simulation includes a part of stochasticity and these data result from only 1 simulation run.

References

Landguth EL, Cushman S (2010). "CDPOP: a spatially explicit cost distance population genetics program." *Molecular Ecology Resources*, **10**(1), 156–161.

Examples

```
data("data_simul_genind")
length(unique(data_simul_genind@pop))
```

data_tuto *data_tuto : data used to generate the vignette*

Description

Data used to generate the vignette

Usage

```
data_tuto
```

Format

Several outputs or inputs to show how the package works in a list

dmc Output of the function 'dist_max_corr'

graph_ci Genetic independence graph example

mat_dps Genetic distance matrix example

mat_pg Second genetic distance matrix example

Examples

```
data("data_tuto")
mat_dps <- data_tuto[[1]]
str(mat_dps)
```

---

dist_max_corr *Compute the distance at which the correlation between genetic distance and landscape distance is maximal*

Description

The function enables to compute the distance at which the correlation between genetic distance and landscape distance is maximal, using a method similar to that employed by van Strien et al. (2015). Iteratively, distance threshold values are tested. For each value, all the population pairs separated by a landscape distance larger than the threshold are removed before the Mantel correlation coefficient between genetic distance and landscape distance is computed. The distance threshold at which the correlation is the strongest is then identified. A figure showing the evolution of the correlation coefficients when landscape distance threshold increases is plotted.

Usage

```
dist_max_corr(mat_gd, mat_ld, interv, from = NULL, to = NULL,
  fig = TRUE, thr_gd = NULL, line_col = "black",
  pts_col = "#999999")
```

Arguments

mat_gd	A symmetric matrix with pairwise genetic distances between populations or sample sites.
mat_ld	A symmetric matrix with pairwise landscape distances between populations or sample sites. These distances can be Euclidean distances, cost-distances or resistance distances, among others.
interv	A numeric value indicating the interval between the different distance thresholds for which the correlation coefficients are computed.
from	(optional) The minimum distance threshold value at which the correlation coefficient is computed.

to	(optional) The maximum distance threshold value at which the correlation coefficient is computed.
fig	Logical (default = TRUE) indicating whether a figure is plotted.
thr_gd	(optional) A numeric value used to remove genetic distance values from the data before the calculation. All genetic distances values above 'thr_gd' are removed from the data. This parameter can be used especially when there are outliers.
line_col	(optional, if fig = TRUE) A character string indicating the color used to plot the line (default: "blue"). It must be a hexadecimal color code or a color used by default in R.
pts_col	(optional, if fig = TRUE) A character string indicating the color used to plot the points (default: "#999999"). It must be a hexadecimal color code or a color used by default in R.

Details

IDs in 'mat_gd' and 'mat_ld' must be the same and refer to the same sampling sites or populations, and both matrices must be ordered in the same way. The correlation coefficient between genetic distance and landscape distance computed is a Mantel correlation coefficient. If there are less than 50 pairwise values, the correlation is not computed, as in van Strien et al. (2015). Such a method can be subject to criticism from a strict statistical point of view given correlation coefficients computed from samples of different size are compared. The matrix of genetic distance 'mat_gd' can be computed using `mat_gen_dist`. The matrix of landscape distance 'mat_ld' can be computed using `mat_geo_dist` when the landscape distance needed is a Euclidean geographical distance. Mantel correlation coefficients are computed using the function `mantel`.

Value

A list of objects:

- The distance at which the correlation is the highest.
- The vector of correlation coefficients at the different distance thresholds
- The vector of the different distance thresholds
- A ggplot2 object to plot

Author(s)

P. Savary

References

Van Strien MJ, Holderegger R, Van Heck HJ (2015). "Isolation-by-distance in landscapes: considerations for landscape genetics." *Heredity*, **114**(1), 27.

Examples

```
data(data_simul_genind)
mat_gen <- mat_gen_dist(data_simul_genind, dist = "basic")
mat_dist <- suppressWarnings(mat_geo_dist(data = pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
res_dmc <- dist_max_corr(mat_gd = mat_gen,
                         mat_ld = mat_dist,
                         interv = 10000)
```

---

genepop_to_genind *Convert a GENEPOP file into a genind object*

Description

The function converts a text file in the format used by GENEPOP software into a genind object

Usage

```
genepop_to_genind(path, n.loci, pop_names = NULL)
```

Arguments

path	A character string with the path leading to the GENEPOP file in format .txt, or alternatively the name of this file in the working directory.
n.loci	The number of loci in the GENEPOP file (integer).
pop_names	(optional) Populations' names in the same order as in the GENEPOP file. Vector object (class character) of the same length as the number of populations. Without this parameter, populations are numbered from 1 to the number of populations.

Details

This function uses functions from **pegas** package. GENEPOP file should only include microsatellites loci with allele names of length 3. The loci line(s) must not start with a spacing.

Value

An object of type genind.

Author(s)

P. Savary

References

Raymond M (1995). "GENEPOP: Population genetics software for exact tests and ecumenism. Vers. 1.2." *Journal of Heredity*, **86**, 248–249.

See Also

For more details about GENEPOP file formatting : `http://genepop.curtin.edu.au/help_input.html#Input` For the opposite conversion, see `genind_to_genepop`. The output file can be used to compute pairwise FST matrix with `mat_pw_fst`

Examples

```
path_in <- system.file('extdata', 'gpop_51_sim22_01_25.txt',
                       package = 'graph4lg')
file_n <- file.path(tempdir(), "gpop_51_sim22_01_25.txt")
file.copy(path_in, file_n, overwrite = TRUE)
genepop_to_genind(path = file_n, n.loci = 20,
                  pop_names = as.character(order(as.character(1:50))))
file.remove(file_n)
```

---

genind_to_genepop *Convert a genind object into a GENEPOP file*

Description

The function converts an object of class `genind` into a GENEPOP file. It then allows to use the functionalities of the GENEPOP software and its derived package **GENEPOP** on R, as well as some functions from other packages (differentiation test, F-stats calculations, HWE test,...). It is designed to be used with diploid microsatellite data and alleles coded with 3 digits

Usage

```
genind_to_genepop(x, output = "data.frame")
```

Arguments

`x`	An object of class `genind` from package **adegenet**.
`output`	A character string indicating the option used to select what the function will return: • If `output = "data.frame"`(default), then the function will return an object 'x' of class `data.frame` ready to be saved as a text file with the following command: `write.table(x,file = "file_name.txt",quote=FALSE,row.nar` • If `output = "path_to_file/file_name.txt"`, then the function will write a text file named 'file_name.txt' in the directory corresponding to 'path_to_file'. Without 'path_to_file', the text file is written in the current working directory. The text file has the format required by GENEPOP software.

Value

An object of type `data.frame` if `ouput = "data.frame"`. If `output` is the path and/or the file name of a text file, then nothing is returned in R environment but a text file is created with the specified file name, either in the current working directory or in the specified folder.

Warning

Confusion: Do not confound this function with `genind2genpop` from **adegenet**. The latter converts an object of class `genind` into an object of class `genpop`, whereas `genind_to_genepop` converts an object of class `genind` into a text file compatible with GENEPOP software (Rousset, 2008).

Allele coding: This function can handle genetic data with different alleles coding lengths (for example alleles '99' and '101' at one locus). BUT, it is highly advisable not to use it with this type of data because applications using GENEPOP file won't be able to use the output file.

Individuals order: When individuals in input data are not ordered by populations, individuals from the same population can be separated by individuals from other populations. It can be problematic when calculating then pairwise distance matrices. Therefore, in such a case, individuals are ordered by populations and populations ordered in alphabetic order.

Author(s)

P. Savary

References

Raymond M (1995). "GENEPOP: Population genetics software for exact tests and ecumenism. Vers. 1.2." *Journal of Heredity*, **86**, 248–249.

See Also

For more details about GENEPOP file formatting : `http://genepop.curtin.edu.au/help_input.html#Input`. For the opposite conversion, see `genepop_to_genind`. The output file can be used to compute pairwise FST matrix with `mat_pw_fst`

Examples

```
data(data_pc_genind)
x <- data_pc_genind
df_genepop <- suppressWarnings(genind_to_genepop(x,
                                                 output = "data.frame"))
```

`gen_graph_indep` *Create an independence graph of genetic differentiation from genetic data of class genind*

Description

The function allows to create genetic graphs from genetic data by applying the conditional independence principle. Populations whose allelic frequencies covary significantly once the covariance with the other populations has been taken into account are linked on the graphs.

Usage

```
gen_graph_indep(x, dist = "basic", cov = "sq", pcor = "magwene",
  alpha = 0.05, test = "EED", adj = "none", output = "igraph")
```

Arguments

`x`	An object of class `genind` that contains the multilocus genotype (format 'locus') of the individuals as well as their population and their geographical coordinates.
`dist`	A character string indicating the method used to compute the multilocus genetic distance between populations • If 'dist = 'basic" (default), then the multilocus genetic distance is computed using a formula of Euclidean genetic distance (Excoffier et al., 1992) • If 'dist = 'weight", then the multilocus genetic distance is computed as in Fortuna et al. (2009). It is a Euclidean genetic distance giving more weight to rare alleles • If 'dist = 'PG", then the multilocus genetic distance is computed as in popgraph::popgraph function, following several steps of PCA and SVD (Dyer et Nason, 2004). • If 'dist = 'PCA", then the genetic distance is computed following a PCA of the matrix of allelic frequencies by population. It is a Euclidean genetic distance between populations in the multidimensional space defined by all the independent principal components.
`cov`	A character string indicating the formula used to compute the covariance matrix from the distance matrix • If 'cov = 'sq" (default), then the covariance matrix is calculated from the matrix of squared distances as in Everitt et Hothorn (2011) • If 'cov = 'dist", then the covariance matrix is calculated from the matrix of distances as in Dyer et Nason (2004) and popgraph::popgraph function
`pcor`	A character string indicating the way the partial correlation matrix is computed from the covariance matrix. • If 'pcor = 'magwene", the steps followed are the same as in Magwene (2001) and in popgraph::popgraph function. It is the recommended option as it meets mathematical requirements.

- If 'pcor = 'other", the steps followed are the same as used by Fortuna et al. (2009). They are not consistent with the approach of Magwene (2001).

`alpha` A numeric value corresponding to the statistical tolerance threshold used to test the difference from 0 of the partial correlation coefficients. By default, 'alpha=0.05'.

`test` A character string indicating the method used to test the significance of the partial correlation coefficients.

- If 'test = 'EED" (default), then the Edge Exclusion Deviance criterion is used (Whittaker, 2009). Although other methods exist, this is the most common and thus the only one implemented here.

`adj` A character string indicating the way of adjusting p-values to assess the significance of the p-values

- If 'adj = 'none" (default), there is no p-value adjustment correction
- If 'adj = 'holm", p-values are adjusted using the sequential Bonferroni correction (Holm, 1979)
- If 'adj = 'bonferroni", p-values are adjusted using the classic Bonferroni correction
- If 'adj = 'BH", p-values are adjusted using Benjamini et Hochberg (1995) correction controlling false discovery rate

`output` A character string indicating the matrices included in the output list.

- If 'output = 'all" (default), then D (distance matrix), C (covariance matrix), Rho (partial correlation matrix), M (graph incidence matrix) and S (strength matrix) are included
- If 'output = 'dist_graph", then the distance matrix D is returned only with the values corresponding to the graph's edges
- If 'output = 'str_graph", then the strength values matrix S is returned only with the values corresponding to the graph's edges
- If 'output = 'inc", then the binary adjacency matrix M is returned
- If 'output = 'igraph", then a graph of class `igraph` is returned

Details

The function allows to vary many parameters such as the genetic distance used, the formula used to compute the covariance, the statistical tolerance threshold, the p-values adjustment, among others.

Value

A `list` of objects of class `matrix`, an object of class `matrix` or a graph object of class `igraph`

Author(s)

P. Savary

References

Dyer RJ, Nason JD (2004). "Population graphs: the graph theoretic shape of genetic structure." *Molecular ecology*, **13**(7), 1713–1727. Benjamini Y, Hochberg Y (1995). "Controlling the false discovery rate: a practical and powerful approach to multiple testing." *Journal of the royal statistical society. Series B (Methodological)*, 289–300. Bowcock AM, Ruiz-Linares A, Tomfohrde J, Minch E, Kidd JR, Cavalli-Sforza LL (1994). "High resolution of human evolutionary trees with polymorphic microsatellites." *nature*, **368**(6470), 455–457. Everitt B, Hothorn T (2011). *An introduction to applied multivariate analysis with R*. Springer Science \& Business Media. Excoffier L, Smouse PE, Quattro JM (1992). "Analysis of molecular variance inferred from metric distances among DNA haplotypes: application to human mitochondrial DNA restriction data." *Genetics*, **131**(2), 479–491. Fortuna MA, Albaladejo RG, FernÃ¡ndez L, Aparicio A, Bascompte J (2009). "Networks of spatial genetic variation across species." *Proceedings of the National Academy of Sciences*, **106**(45), 19044–19049. Holm S (1979). "A simple sequentially rejective multiple test procedure." *Scandinavian journal of statistics*, 65–70. Magwene PM (2001). "New tools for studying integration and modularity." *Evolution*, **55**(9), 1734–1745. Wermuth N, Scheidt E (1977). "Algorithm AS 105: fitting a covariance selection model to a matrix." *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **26**(1), 88–92. Whittaker J (2009). *Graphical models in applied multivariate statistics*. Wiley Publishing.

Examples

```
data(data_pc_genind)
dist_graph_test <- gen_graph_indep(x = data_pc_genind, dist = "basic",
                                   cov = "sq", pcor = "magwene",
                                   alpha = 0.05, test = "EED",
                                   adj = "none", output = "igraph")
```

`gen_graph_thr` *Create a graph of genetic differentiation using a link's weight threshold*

Description

The function allows to construct a genetic graph whose links' weights are larger or lower than a specific threshold

Usage

```
gen_graph_thr(mat_w, mat_thr = NULL, thr, mode = "larger")
```

Arguments

`mat_w`	A symmetric (pairwise) `matrix` whose elements will be the links' weights
`mat_thr`	(optional) A symmetric (pairwise) distance `matrix` whose values will be used for the pruning based on the threshold value.
`thr`	The threshold value (logically between min(mat_thr) and max(mat_thr))

mode
- If 'mode = 'larger" (default), all the links whose weight is larger than 'thr' are removed.
- If 'mode = 'lower", all the links whose weight is lower than 'thr' are removed.

Details

If 'mat_thr' is not defined, 'mat_w' is used for the pruning. Matrices 'mat_w' and 'mat_thr' must have the same dimensions and the same rows' and columns' names. Values in 'mat_thr' matrix must be positive. Negative values from 'mat_w' are transformed into zeros. The function works only for undirected graphs.

Value

A graph object of class `igraph`

Author(s)

P. Savary

Examples

```
mat_w <- mat_gen_dist(x = data_simul_genind, dist = 'DPS')
suppressWarnings(mat_thr <- mat_geo_dist(pts_pop_simul,
                  ID = "ID",
                  x = "x",
                 y = "y"))
mat_thr <- mat_thr[row.names(mat_w), colnames(mat_w)]
graph <- gen_graph_thr(mat_w, mat_thr, thr = 6000, mode = "larger")
```

---

gen_graph_topo *Create a graph of genetic differentiation with a specific topology*

Description

The function allows to construct a genetic graph with a specific topology from genetic and/or geographical distance matrices

Usage

```
gen_graph_topo(mat_w, mat_topo = NULL, topo = "gabriel")
```

Arguments

mat_w	A symmetric (pairwise) matrix whose elements will be the links' weights
mat_topo	(optional) A symmetric (pairwise) distance matrix whose values will be used for the pruning method.
topo	Which topology does the created graph have?

- If 'topo = 'gabriel" (default), the resulting graph will be a Gabriel graph (Gabriel et al., 1969). It means that there is a link between nodes x and y if and only if $d_{xy}^2 \leq \min(\sqrt{d_{xz}^2 + d_{yz}^2})$, with z any other node of the graph.
- If 'topo = 'mst", the resulting graph will have the topology of a minimum spanning tree. It means that the graph will not include any cycle (tree) and it will be the subgraph with a tree topology with the minimum total links' weight (based on 'mat_topo' values).
- If 'topo = 'percol", if the link of the resulting graph with the minimum weight is removed, then the graph breaks into two components.
- If 'topo = 'comp", a complete graph whose links are weighted with values from 'mat_w' is created.

Details

If 'mat_topo' is not defined, 'mat_w' is used for the pruning. Matrices 'mat_w' and 'mat_topo' must have the same dimensions and the same rows' and columns' names. Values in 'mat_topo' matrix must be positive. Negative values from 'mat_w' are transformed into zeros. The function works only for undirected graphs.

Value

A graph object of class igraph

Author(s)

P. Savary

References

Gabriel KR, Sokal RR (1969). "A new statistical approach to geographic variation analysis." *Systematic zoology*, **18**(3), 259–278.

Examples

```
mat_w <- mat_gen_dist(x = data_simul_genind, dist = 'DPS')
suppressWarnings(mat_topo <- mat_geo_dist(pts_pop_simul,
                  ID = "ID",
                  x = "x",
                 y = "y"))
mat_topo <- mat_topo[row.names(mat_w), colnames(mat_w)]
graph <- gen_graph_topo(mat_w, mat_topo, topo = "mst")
```

`graphab_to_igraph` *Import landscape graphs from GRAPHAB software*

Description

The function imports a landscape graph created with GRAPHAB software and converts it into a graph object of class `igraph`. The graph has weighted links and is undirected. Nodes have spatial coordinates. Other nodes attributes can be included. It takes shapefiles layers created with GRAPHAB as input.

Usage

```
graphab_to_igraph(dir_path, nodes = "patches", links, weight = "cost",
  fig = FALSE, crds = FALSE)
```

Arguments

`dir_path`	A character string indicating the path of the GRAPHAB project directory. This directory normally contains several spatial layer files in format .shp: • the spatial layer of the habitat patches corresponding to the nodes of the graph (usually named 'patches.shp'). • (alternatively) an exported spatial layer of the nodes (faster option). • the links' spatial layer file used to import the graph.
`nodes`	A character string indicating the names of the nodes' spatial layer in format .shp (without extension, ex.: "nodes" refers to "nodes.shp" layer). This layer has been created with GRAPHAB and has therefore coordinates in a projected coordinates reference system. Default: nodes = "patches", referring to the spatial polygon layer of the habitat patches.
`links`	A character string indicating the name of the links' spatial layer in format .shp (without extension, ex.: "link_graph" refers to "link_graph.shp" layer). This layer has been created with GRAPHAB and has therefore coordinates in a projected coordinates reference system. It includes in the attribute tables between patches' Euclidean as well as cost-distance. These distances are used to weight the link.
`weight`	A character string ("euc" or "cost") indicating whether to weight the links with Euclidean distance or cost-distance (default) values.
`fig`	Logical (default = FALSE) indicating whether to plot a figure of the resulting spatial graph. The figure is plotted using function `plot_graph_lg`. The plotting can be long if the graph has many nodes and links.
`crds`	Logical (default = FALSE) indicating whether to create an object of class `data.frame` with the nodes spatial coordinates. Such a `data.frame` has 3 columns: 'ID', 'x', 'y'.

Details

Nodes attributes can be added to the graph using the function `add_nodes_attr`.

Value

A graph object of class `igraph` (if crds = FALSE) or a list of objects: a graph object of class `igraph` and a `data.frame` with the nodes spatial coordinates (if crds = TRUE).

Author(s)

P. Savary

References

FoltÃªte J, Clauzel C, Vuidel G (2012). "A software tool dedicated to the modelling of landscape networks." *Environmental Modelling \& Software*, **38**, 316–327.

Examples

```
path <- system.file('extdata',package='graph4lg')
links <- "liens_rast2_1_11_01_19-links"
graph <- graphab_to_igraph(dir_path = path,
                           links = links,
                           fig = FALSE)
```

`graph_modul_compar` *Compare the partition into modules of two graphs*

Description

The function computes the Adjusted Rand Index (ARI) to compare two graphs' partitions into modules or clusters more generally. Both graphs must have the same number of nodes, but not necessarily the same number of links. They must also have the same nodes' names and in the same order.

Usage

```
graph_modul_compar(x, y, mode = "graph", nb_modul = NULL,
  algo = "fast_greedy", weight = "inv", data = NULL)
```

Arguments

`x` The first graph object

- If `mode = 'graph'` (default), `x` is a graph object of class `igraph`. Then, its nodes must have the same names as in graph `y`.
- If `mode = 'data.frame'`, `x` refers to a column of the `data.frame` 'data'. Then `x` must be a character string indicating the name of the column of 'data' with the modules' labels of the nodes in the first graph. In that case, the column can be of class `numeric`, `character` or `factor` but will be converted into a `numeric` vector in any case.

- If `mode = 'vector'`, `x` is a vector of class `character`, `factor` or `numeric`. In that case, it must have the same length as vector `y` and will be converted into a `numeric` vector.

`y` The second graph object Same classes possible as for `x`. Must be of the same format as `x`

`mode` A character string indicating whether x and y are igraph objects, vectors or columns from a data.frame. `mode` can be 'graph', 'data.frame' or 'vector'.

`nb_modul` (if x and y are igraph objects) A numeric value or numeric vector with 2 elements indicating the number of modules to create in both graphs.

- If `nb_modul` is a numeric value, then the same number of modules are created in both graphs.
- If `nb_modul` is a numeric vector of length 2, then the numbers of modules created in graphs `x` and `y` are the first and second elements of `nb_modul`, respectively.

`algo` (if x and y are igraph objects) A character string indicating the algorithm used to create the modules with **igraph**.

- If `algo = 'fast_greedy'` (default), function `cluster_fast_greedy` from **igraph** is used (Clauset et al., 2004).
- If `algo = 'walktrap'` (default), function `cluster_walktrap` from **igraph** is used (Pons et Latapy, 2006) with 4 steps (default options).
- If `algo = 'louvain'`, function `cluster_louvain` from **igraph** is used (Blondel et al., 2008). In that case, the number of modules created in each graph is imposed.
- If `algo = 'optimal'`, function `cluster_optimal` from **igraph** is used (Brandes et al., 2008) (can be very long). In that case, the number of modules created in each graph is imposed.

`weight` (optional, if x and y are igraph objects) A character string or character vector indicating how to weight graphs' links during the calculation of the modularity.

- If `weight = 'inv'` (default), then links are weighted with the inverse values of their initial weights.
- If `weight = 'w'`, then links are weighted with their initial weights values.
- If `weight = 'none'`, then links are not weighted during the calculation.

Two different weightings can be used to create the modules of the two graphs.

- If `weight` is a character string, then the same algorithm is used for both graphs.
- If `weight` is a character vector of length 2, then the link weighting used by the algorithm to create the modules of graphs `x` and `y` is determined by the first and second elements of `weight`, respectively.

If the graphs' links are not weighted, then this argument is ignored. Links with large weights are considered as stronger connections in the modularity calculation.

`data` (if x and y are columns from a data.frame) An object of class data.frame with at least two columns and as many rows as there are nodes in the graphs compared. The columns indicate the modules of each node in 2 different classifications.

Details

This index takes values between -1 and 1. It measures how often pairs of nodes pertaining to the same module in one graph also pertain to the same module in the other graph. Therefore, large values indicate that both partitions are similar. The Rand Index can be defined as the frequency of agreement between two classifications into discrete classes. It is the number of times a pair of elements are classified into the same class or in two different classes in both compared classifications, divided by the total number of possible pairs of elements. The Rand Index is between 0 and 1 but its maximum value depends on the number of elements. Thus, another 'adjusted' index was created, the Adjusted Rand Index. According to the Hubert et Arabie's formula, the ARI is computed as follows: $ARI = \frac{Index - Expectedindex}{Maximumindex - Expectedindex}$ where the values of Index, Expected index and Maximum index are computed from a contingency table. This function uses `adjustedRandIndex` from package **mclust** which applies the Hubert and Arabie's formula for the ARI. This function works for undirected graphs only.

Value

The value of the ARI

Author(s)

P. Savary

References

Dyer RJ, Nason JD (2004). "Population graphs: the graph theoretic shape of genetic structure." *Molecular ecology*, **13**(7), 1713–1727. Hubert L, Arabie P (1985). "Comparing partitions." *Journal of classification*, **2**(1), 193–218. Clauset A, Newman ME, Moore C (2004). "Finding community structure in very large networks." *Physical review E*, **70**(6). Blondel VD, Guillaume J, Lambiotte R, Lefebvre E (2008). "Fast unfolding of communities in large networks." *Journal of Statistical Mechanics - Theory and Experiment*, **10**. Brandes U, Delling D, Gaertler M, Gorke R, Hoefer M, Nikoloski Z, Wagner D (2008). "On modularity clustering." *IEEE transactions on knowledge and data engineering*, **20**(2), 172–188. Pons P, Latapy M (2006). "Computing communities in large networks using random walks." *J. Graph Algorithms Appl.*, **10**(2), 191–218.

Examples

```
data(data_simul_genind)
data(pts_pop_simul)
mat_dist <- suppressWarnings(graph4lg::mat_geo_dist(data=pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
graph_obs <- gen_graph_thr(mat_w = mat_dist, mat_thr = mat_dist,
                           thr = 9500, mode = "larger")
mat_gen <- mat_gen_dist(x = data_simul_genind, dist = "DPS")
graph_pred <- gen_graph_topo(mat_w = mat_gen, mat_topo = mat_dist,
                           topo = "gabriel")
ARI <- graph_modul_compar(x = graph_obs, y = graph_pred)
```

graph_node_compar *Compare the local properties of the nodes from two graphs*

Description

The function computes a correlation coefficient between the graph-theoretic metric's values computed at the node-level in two graphs sharing the same nodes. It allows to assess whether the connectivity properties of the nodes in one graph are similar to that of the same nodes in the other graph. Alternatively, the correlation is computed between a graph-theoretic metric's values and the values of an attribute associated to the nodes of a graph.

Usage

```
graph_node_compar(x, y, metrics = c("siw", "siw"), method = "spearman",
  weight = TRUE, test = TRUE)
```

Arguments

x	An object of class igraph. Its nodes must have the same names as in graph y.
y	An object of class igraph. Its nodes must have the same names as in graph x.
metrics	Two-elements character vector specifying the graph-theoretic metrics computed at the node-level in the graphs or the nodes' attribute values to be correlated to these metrics. Graph-theoretic metrics can be: • Degree (metrics = c("deg",...)) • Closeness centrality index (metrics = c("close",...)) • Betweenness centrality index (metrics = c("btw",...)) • Strength (sum of the weights of the links connected to a node) (metrics = c("str",...)) • Sum of the inverse weights of the links connected to a node (metrics = c("siw",...), default) • Mean of the inverse weights of the links connected to a node (metrics = c("miw",...)) Nodes' attributes must have the same names as in the igraph object, and must refer to an attribute with numerical values. The vector metrics is composed of two character values. When a nodes' attribute has the same name as a metric computable from the graph, nodes' attributes are given priority.
method	A character string indicating which correlation coefficient is to be computed ("pearson", "kendall" or "spearman" (default)).
weight	Logical which indicates whether the links are weighted during the calculation of the centrality indices betweenness and closeness. (default: weight = TRUE). Links' weights are interpreted as distances when computing the shortest paths. They should then be inversely proportional to the strength of the relationship between nodes (e.g. to fluxes).
test	Logical. Should significance testing be performed? (default = TRUE)

Details

The correlation coefficients between the metrics can be computed in different ways, as initial assumptions (e.g. linear relationship) are rarely verified. Pearson's r, Spearman's rho and Kendall's tau can be computed (from function `cor`). When `x` is similar to `y`, then the correlation is computed between two metrics characterizing the nodes of the same graph.

Value

A `list` summarizing the correlation analysis.

Author(s)

P. Savary

Examples

```
data(data_simul_genind)
data(pts_pop_simul)
mat_dist <- suppressWarnings(graph4lg::mat_geo_dist(data = pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
graph_obs <- gen_graph_thr(mat_w = mat_dist, mat_thr = mat_dist,
                           thr = 9500, mode = "larger")
mat_gen <- mat_gen_dist(x = data_simul_genind, dist = "DPS")
graph_pred <- gen_graph_topo(mat_w = mat_gen, mat_topo = mat_dist,
                             topo = "gabriel")
res_cor <- graph_node_compar(x = graph_obs, y = graph_pred,
                             metrics = c("siw", "siw"), method = "spearman",
                             test = TRUE, weight = TRUE)
```

---

graph_plot_compar *Visualize the topological differences between two spatial graphs on a map*

Description

The function enables to compare two spatial graphs by plotting them highlighting the topological similarities and differences between them. Both graphs should share the same nodes and cannot be directed graphs.

Usage

```
graph_plot_compar(x, y, crds)
```

Arguments

x	A graph object of class igraph. Its nodes must have the same names as in graph y.
y	A graph object of class igraph. Its nodes must have the same names as in graph x.
crds	A data.frame with the spatial coordinates of the graph's nodes (both x and y). It must have three columns: • ID: Name of the graph's nodes (character string). The names must be the same as the nodes' names of the graphs of class igraph (igraph::V(graph)$name) • x: Longitude of the graphs' nodes. • y: Latitude of the graphs' nodes.

Details

The graphs x and y of class igraph must have node names (not necessarily in the same order as IDs in crds, given a merging is done).

Value

A ggplot2 object to plot

Author(s)

P. Savary

Examples

```
data(pts_pop_simul)
data(data_simul_genind)
mat_w <- mat_gen_dist(data_simul_genind, dist = "DPS")
mat_dist <- mat_geo_dist(data = pts_pop_simul,
                         ID = "ID",
                         x = "x",
                         y = "y")
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
g1 <- gen_graph_topo(mat_w = mat_w, topo = "mst")
g2 <- gen_graph_topo(mat_w = mat_w, mat_topo = mat_dist, topo = "gabriel")
g <- graph_plot_compar(x = g1, y = g2,
                       crds = pts_pop_simul)
```

`graph_topo_compar` *Compute an index comparing graphs' topologies*

Description

The function computes several indices in order to compare two graphs' topologies. One of the graph has the "true" topology the other is supposed to reproduce. The indices are then a way to assess the reliability of the latter graph. Both graphs must have the same number of nodes, but not necessarily the same number of links. They must also have the same nodes' names and in the same order.

Usage

```
graph_topo_compar(obs_graph, pred_graph, mode = "mcc",
  directed = FALSE)
```

Arguments

`obs_graph`	A graph object of class `igraph` with n nodes. It is the observed graph that `pred_graph` is supposed to approach.
`pred_graph`	A graph object of class `igraph` with n nodes. It is the predicted graph that is supposed to be akin to `obs_graph`.
`mode`	A character string specifying which index to compute in order to compare the topologies of the graphs. • If 'mode = 'mcc" (default), the Matthews Correlation Coefficient (MCC) is computed. • If 'mode = 'kappa", the Kappa index is computed. • If 'mode = 'fdr", the False Discovery Rate (FDR) is computed. • If 'mode = 'acc", the Accuracy is computed. • If 'mode = 'sens", the Sensitivity is computed. • If 'mode = 'spec", the Specificity is computed. • If 'mode = 'prec", the Precision is computed.
`directed`	Logical (TRUE or FALSE) specifying whether both graphs are directed or not.

Details

The indices are calculated from a confusion matrix counting the number of links that are in the "observed" graph ("true") and also in the "predicted" graph (true positives : TP), that are in the "observed" graph but not in the "predicted" graph (false negatives : FN), that are not in the "observed" graph but in the "predicted" graph (false positives : FP) and that are not in the "observed" graph and not in the "predicted" graph neither (true negatives: TN). K is the total number of links in the graphs. K is equal to $n \times (n-1)$ if the graphs are directed and to $\frac{n \times (n-1)}{2}$ if they are not directed, with n the number of nodes. OP = TP + FN, ON = TN + FP, PP = TP + FP and PN = FN + TN.

The Matthews Correlation Coefficient (MCC) is computed as follows: $MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

The Kappa index is computed as follows: $Kappa = \frac{K \times (TP+TN) - (ON \times PN) - (OP \times PP)}{K^2 - (ON \times PN) - (OP \times PP)}$

The False Discovery Rate (FDR) is calculated as follows: $FDR = \frac{FP}{TP+FP}$

The Accuracy is calculated as follows: $Acc = \frac{TP+TN}{K}$

The Sensitivity is calculated as follows: $Sens = \frac{TP}{TP+FN}$

The Specificity is calculated as follows: $Spec = \frac{TN}{TN+FP}$

The Precision is calculated as follows: $Prec = \frac{TP}{TP+FP}$

Self loops are not taken into account.

Value

The value of the index computed

Author(s)

P. Savary

References

Dyer RJ, Nason JD (2004). "Population graphs: the graph theoretic shape of genetic structure." *Molecular ecology*, **13**(7), 1713–1727. Baldi P, Brunak S, Chauvin Y, Andersen CA, Nielsen H (2000). "Assessing the accuracy of prediction algorithms for classification: an overview." *Bioinformatics*, **16**(5), 412–424. Matthews BW (1975). "Comparison of the predicted and observed secondary structure of T4 phage lysozyme." *Biochimica et Biophysica Acta (BBA)-Protein Structure*, **405**(2), 442–451.

Examples

```
data(data_simul_genind)
data(pts_pop_simul)
mat_dist <- suppressWarnings(graph4lg::mat_geo_dist(data=pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
graph_obs <- gen_graph_thr(mat_w = mat_dist, mat_thr = mat_dist,
                           thr = 15000, mode = "larger")
mat_gen <- mat_gen_dist(x = data_simul_genind, dist = "DPS")
graph_pred <- gen_graph_topo(mat_w = mat_gen, mat_topo = mat_dist,
                           topo = "gabriel")
graph_topo_compar(obs_graph = graph_obs,
                  pred_graph = graph_pred,
                  mode = "mcc",
                  directed = FALSE)
```

graph_to_shp *Export a spatial graph to shapefile layers*

Description

The function enables to export a spatial graph to shapefile layers.

Usage

```
graph_to_shp(graph, crds, mode = "both", crds_crs, layer_name, dir_path,
  metrics = FALSE)
```

Arguments

graph	A graph object of class igraph
crds	(if 'mode = 'spatial") A data.frame with the spatial coordinates of the graph's nodes. It must have three columns: • ID: Name of the graph's nodes (will be converted into character string). The names must the same as the nodes' names of the graph object of class igraph (igraph::V(graph)$name) • x: Longitude (numeric) of the graphs' nodes in the coordinates reference system indicated with the argument crds_crs. • y: Latitude (numeric) of the graphs' nodes in the coordinates reference system indicated with the argument crds_crs.
mode	Indicates which shapefile layers will be created • If 'mode = 'both" (default), then two shapefile layers are created, one for the nodes and another for the links. • If 'mode = 'node", a shapefile layer is created for the nodes only. • If 'mode = 'link", a shapefile layer is created for the links only.
crds_crs	A character string indicating the Coordinates Reference System of the spatial coordinates of the nodes and of the shapefile layers created. The projection and datum are given in the PROJ.4 format.
layer_name	A character string indicating the suffix of the name of the layers to be created.
dir_path	A character string corresponding to the path to the directory in which the shapefile layers will be exported. If dir_path = "wd", then the layers are created in the current working directory.
metrics	(not possible if 'mode = 'link") Logical. Should metrics be calculated and integrated in the attribute table of the nodes' shapefile layer? (default: FALSE) Metrics calculated are degrees, betweenness centrality and sum of inverse weight (if links are weighted)

Value

Create shapefile layers in the directory specified with the parameter 'dir_path'.

Author(s)

P. Savary

Examples

```
data(data_tuto)
mat_w <- data_tuto[[1]]
gp <- gen_graph_topo(mat_w = mat_w, topo = "gabriel")
crds_crs1 <- "+proj=lcc +lat_1=49 +lat_2=44 +lat_0=46.5 +lon_0=3 "
crds_crs2 <- "+x_0=700000 +y_0=6600000 +ellps=GRS80 +units=m +no_defs"
crds_crs <- paste(crds_crs1, crds_crs2, sep = "")
crds <- pts_pop_simul
layer_name <- "graph_dps_gab"
graph_to_shp(graph = gp, crds = pts_pop_simul, mode = "both",
             crds_crs = crds_crs,
             layer_name = "test_fonct",
             dir_path = tempdir(),
             metrics = TRUE)
```

---

gstud_to_genind *Convert a file from* **gstudio** *or* **popgraph** *into a genind object*

Description

The function converts a file formatted to use **gstudio** or **popgraph** package into a genind object (**adegenet** package)

Usage

```
gstud_to_genind(x, pop_col, ind_col = NULL)
```

Arguments

x	An object of class `data.frame` with loci's columns in format `locus` (defined in package **gstudio**) with as many rows as individuals and as many columns in format `locus` as there are loci and additional columns
pop_col	A character string indicating the name of the column with populations' names in `x`
ind_col	(optional) A character string indicating the name of the column with individuals' ID in `x`

Details

This function uses functions from **pegas** package. It can handle genetic data where alleles codings do not have same length, (99:101, for example). If the names of the loci include '.' characters, they will be replaced by '_'.

Value

An object of class genind.

Author(s)

P. Savary

Examples

```
data("data_pc_gstud")
x <- data_pc_gstud
pop_col <- "Cluster"
ind_col <- "ID"
data_genind <- gstud_to_genind(x, pop_col, ind_col)
```

---

g_percol *Prune a graph using the 'percolation threshold' method*

Description

The function allows to prune a graph by removing the links with the largest weights until the graph breaks into two components. The returned graph is the last graph with only one component.

Usage

```
g_percol(x, val_step = 20)
```

Arguments

x	A symmetric matrix of pairwise distances between nodes
val_step	The number of classes to create to search for the threshold value without testing all the possibilities. By default, 'val_step = 20'.

Value

A graph object of type igraph

Author(s)

P. Savary

Examples

```
data(data_simul_genind)
suppressWarnings(mat_w <- graph4lg::mat_geo_dist(data=pts_pop_simul,
                            ID = "ID",
                            x = "x",
                            y = "y"))
g_percol(x = mat_w)
```

loci_to_genind *Convert a loci object into a genind object*

Description

This function is exactly the same as `loci2genind` from **pegas** package

Usage

```
loci_to_genind(x, ploidy = 2, na.alleles = c("NA"))
```

Arguments

`x`	An object of class `loci` to convert
`ploidy`	An integer indicating the ploidy level (by default, 'ploidy = 2')
`na.alleles`	A character vector indicating the coding of the alleles to be treated as missing data (by default, 'na.alleles = c("NA")')

Value

An object of class `genind`

Author(s)

P. Savary

Examples

```
data("data_pc_loci")
genind <- loci_to_genind(data_pc_loci, ploidy = 2, na.alleles = "NA")
```

mat_gen_dist *Compute a pairwise matrix of genetic distances between populations*

Description

The function computes a pairwise matrix of genetic distances between populations and allows to implement several formula.

Usage

```
mat_gen_dist(x, dist = "basic", null_val = FALSE)
```

Arguments

x	An object of class genind that contains the multilocus genotypes (format 'locus') of the individuals as well as their populations.
dist	A character string indicating the method used to compute the multilocus genetic distance between populations • If 'dist = 'basic" (default), then the multilocus genetic distance is computed using a formula of Euclidean genetic distance (Excoffier et al., 1992) • If 'dist = 'weight", then the multilocus genetic distance is computed as in Fortuna et al. (2009). It is a Euclidean genetic distance giving more weight to rare alleles • If 'dist = 'PG", then the multilocus genetic distance is computed as in popgraph::popgraph function, following several steps of PCA and SVD (Dyer et Nason, 2004). • If 'dist = 'DPS", then the genetic distance used is equal to 1 - the proportion of shared alleles (Bowcock, 1994) • If 'dist = 'FST", then the genetic distance used is the pairwise FST (Weir et Cockerham, 1984) • If 'dist = 'FST_lin", then the genetic distance used is the linearised pairwise FST (Weir et Cockerham, 1984)(FST_lin = FST/(1-FST)) • If 'dist = 'PCA", then the genetic distance is computed following a PCA of the matrix of allelic frequencies by population. It is a Euclidean genetic distance between populations in the multidimensional space defined by all the independent principal components. • If 'dist = 'GST", then the genetic distance used is the G'ST (Hedrick, 2005) • If 'dist = 'D", then the genetic distance used is Jost's D (Jost, 2008)
null_val	(optional) Logical. Should negative and null FST, FST_lin, GST or D values be replaced by half the minimum positive value? This option allows to compute Gabriel graphs from these "distances". Default is null_val = FALSE. This option only works if 'dist = 'FST" or 'FST_lin' or 'GST' or 'D'

Details

Negative values are converted into 0. Euclidean genetic distance d_{ij} between population i and j is computed as follows:

$$d_{ij}^2 = \sum_{k=1}^{n} (x_{ki} - x_{kj})^2$$

where x_{ki} is the allelic frequency of allele k in population i and n is the total number of alleles. Note that when 'dist = 'weight", the formula becomes

$$d_{ij}^2 = \sum_{k=1}^{n} (1/(K * p_k))(x_{ki} - x_{kj})^2$$

where K is the number of alleles at the locus of the allele k and p_k is the frequency of the allele k in all populations. Note that when 'dist = 'PCA", n is the number of conserved independent principal components and x_{ki} is the value taken by the principal component k in population i.

Value

An object of class `matrix`

Author(s)

P. Savary

References

Bowcock AM, Ruiz-Linares A, Tomfohrde J, Minch E, Kidd JR, Cavalli-Sforza LL (1994). "High resolution of human evolutionary trees with polymorphic microsatellites." *nature*, **368**(6470), 455–457. Excoffier L, Smouse PE, Quattro JM (1992). "Analysis of molecular variance inferred from metric distances among DNA haplotypes: application to human mitochondrial DNA restriction data." *Genetics*, **131**(2), 479–491. Dyer RJ, Nason JD (2004). "Population graphs: the graph theoretic shape of genetic structure." *Molecular ecology*, **13**(7), 1713–1727. Fortuna MA, Albaladejo RG, FernÃ¡ndez L, Aparicio A, Bascompte J (2009). "Networks of spatial genetic variation across species." *Proceedings of the National Academy of Sciences*, **106**(45), 19044–19049. Weir BS, Cockerham CC (1984). "Estimating F-statistics for the analysis of population structure." *evolution*, **38**(6), 1358–1370. Hedrick PW (2005). "A standardized genetic differentiation measure." *Evolution*, **59**(8), 1633–1638. Jost L (2008). "GST and its relatives do not measure differentiation." *Molecular ecology*, **17**(18), 4015–4026.

Examples

```
data(data_simul_genind)
x <- data_simul_genind
D <- mat_gen_dist(x = x, dist = "basic")
```

---

mat_geo_dist *Compute Euclidean geographic distances between points*

Description

The function computes Euclidean geographic distance between points given their coordinates in a metric projected Coordinates Reference System.

Usage

```
mat_geo_dist(data, ID = NULL, x = NULL, y = NULL)
```

Arguments

`data` An object of class :

- `data.frame` with 3 columns: 2 columns with the points' spatial coordinates and another column with points' IDs
- `SpatialPointsDataFrame`

ID	(if data is of class data.frame) A character string indicating the name of the column of data with the points' IDs
x	(if data is of class data.frame) A character string indicating the name of the column of data with the points' longitude
y	(if data is of class data.frame) A character string indicating the name of the column of data with the points' latitude

Details

Calculate classical Euclidean geographic distance (in m) between two points using Pythagora's theorem

Value

A pairwise matrix of geographic distances between points

Author(s)

P. Savary

Examples

```
data(pts_pop_simul)
mat_dist <- mat_geo_dist(data=pts_pop_simul,
             ID = "ID",
             x = "x",
             y = "y")
```

---

mat_pw_dps *Compute a pairwise genetic distance matrix between populations using Bowcock et al. (1994) formula*

Description

The function computes the pairwise DPS, a genetic distance based on the proportion of shared alleles.

Usage

```
mat_pw_dps(x)
```

Arguments

x	An object of class genind

Details

The formula used is inspired from MSA software :

$$D_{PS} = 1 - \frac{\sum_d^D \sum_k^K \min(f_{a_{kd}i}, f_{a_{kd}j})}{D}$$

such as a_{kd} is the allele k at locus d D is the total number of loci K is the allele number at each locus $\gamma_{a_{kd}ij} = 0$ if individuals i and j do not share allele a_{kd} $\gamma_{a_{kd}ij} = 1$ if one of individuals i and j has a copy of a_{kd} $\gamma_{a_{kd}ij} = 2$ if both individuals have 2 copies of a_{kd} (homozygotes) $f_{a_{kd}i}$ is allele a_{kd} frequency in individual i (0, 0.5 or 1). More information in : Bowcock et al., 1994[1] and MSA manual[2]. This function uses functions from **adegenet** package Note that in the paper of Bowcock et al. (1994), the denominator is 2D. But, in MSA software manual, the denominator is D.

Value

A pairwise matrix of genetic distances between populations

Author(s)

P. Savary

References

Bowcock AM, Ruiz-Linares A, Tomfohrde J, Minch E, Kidd JR, Cavalli-Sforza LL (1994). "High resolution of human evolutionary trees with polymorphic microsatellites." *nature*, **368**(6470), 455–457.

Examples

```
data("data_simul_genind")
dist_bowcock <- mat_pw_dps(data_simul_genind)
```

`mat_pw_d_j` *Compute a pairwise Jost's D matrix between populations*

Description

The function computes the pairwise Jost's D matrix between populations from an object of class `genind` or directly from a GENEPOP file.

Usage

```
mat_pw_d_j(x, pop_names = NULL)
```

[1] https://www.ncbi.nlm.nih.gov/pubmed/7510853
[2] http://i122server.vu-wien.ac.at/MSA/info.html/Distances.html#DPS

Arguments

x	An object of class genind, or the character string indicating the path of the GENEPOP file.
pop_names	(optional) A vector of class character of the same length as the number of populations (rows' and columns' number in the returned matrix). It contains the name of the populations.

Details

The formula used is that of Jost (2008) This functions uses directly the function diffCalc from **diveRsity**. See http://genepop.curtin.edu.au/help_input.html for details on the GENEPOP file format and see Raymond (1995) for detail about GENEPOP software.

Value

A pairwise matrix of G'ST with as many rows and columns as there are populations in the input data.

Warnings

The order of populations matters :

- If x is an object of class genind, individuals are re-ordered by populations and populations are ordered in alphabetic order.
- If x is the path to a GENEPOP file, populations' order in pop_names must be the same as in the GENEPOP file.

Negative values are converted into 0

Author(s)

P. Savary

References

Jost L (2008). "GST and its relatives do not measure differentiation." *Molecular ecology*, **17**(18), 4015–4026. Raymond M (1995). "GENEPOP: Population genetics software for exact tests and ecumenism. Vers. 1.2." *Journal of Heredity*, **86**, 248–249.

Examples

```
data("data_pc_genind")
mat_d_j <- mat_pw_d_j(data_pc_genind)

path_in <- system.file('extdata', 'gpop_51_sim22_01_25.txt',
                       package = 'graph4lg')
file_n <- file.path(tempdir(), "gpop_51_sim22_01_25.txt")
file.copy(path_in, file_n, overwrite = TRUE)
mat_pw_d_j(x = file_n, pop_names = as.character(order(as.character(1:50))))
file.remove(file_n)
```

mat_pw_fst *Compute a pairwise FST matrix between populations*

Description

The function computes the pairwise FST matrix between populations from an object of class genind or directly from a GENEPOP file.

Usage

```
mat_pw_fst(x, pop_names = NULL)
```

Arguments

x	An object of class genind, or the character string indicating the path of the GENEPOP file.
pop_names	(optional) A vector of class character of the same length as the number of populations (rows' and columns' number in the returned matrix). It contains the name of the populations.

Details

The formula used is that of Weir et Cockerham (1984). This functions uses directly the function diffCalc from **diveRsity**. See http://genepop.curtin.edu.au/help_input.html for details on the GENEPOP file format and see Raymond (1995) for detail about GENEPOP software.

Value

A pairwise matrix of FST with as many rows and columns as there are populations in the input data.

Warnings

The order of populations matters :

- If x is an object of class genind, individuals are re-ordered by populations and populations are ordered in alphabetic order.
- If x is the path to a GENEPOP file, populations' order in pop_names must be the same as in the GENEPOP file.

Negative values are converted into 0

Author(s)

P. Savary

References

Weir BS, Cockerham CC (1984). "Estimating F-statistics for the analysis of population structure." *evolution*, **38**(6), 1358–1370. Raymond M (1995). "GENEPOP: Population genetics software for exact tests and ecumenism. Vers. 1.2." *Journal of Heredity*, **86**, 248–249.

Examples

```
data("data_pc_genind")
mat_d_j <- mat_pw_d_j(data_pc_genind)
path_in <- system.file('extdata', 'gpop_51_sim22_01_25.txt',
                       package = 'graph4lg')
file_n <- file.path(tempdir(), "gpop_51_sim22_01_25.txt")
file.copy(path_in, file_n, overwrite = TRUE)
mat_pw_fst(x = file_n, pop_names = as.character(order(as.character(1:50))))
file.remove(file_n)
```

---

mat_pw_gst *Compute a pairwise G'ST matrix between populations*

Description

The function computes the pairwise G'ST matrix between populations from an object of class `genind` or directly from a GENEPOP file.

Usage

```
mat_pw_gst(x, pop_names = NULL)
```

Arguments

`x`	An object of class `genind`, or the character string indicating the path of the GENEPOP file.
`pop_names`	(optional) A vector of class `character` of the same length as the number of populations (rows' and columns' number in the returned matrix). It contains the name of the populations.

Details

The formula used is that of Hedrick (2005) This functions uses directly the function `diffCalc` from **diveRsity**. See `http://genepop.curtin.edu.au/help_input.html` for details on the GENEPOP file format and see Raymond (1995) for detail about GENEPOP software.

Value

A pairwise `matrix` of G'ST with as many rows and columns as there are populations in the input data.

Warnings

The order of populations matters :

- If x is an object of class genind, individuals are re-ordered by populations and populations are ordered in alphabetic order.
- If x is the path to a GENEPOP file, populations' order in pop_names must be the same as in the GENEPOP file.

Negative values are converted into 0

Author(s)

P. Savary

References

Hedrick PW (2005). "A standardized genetic differentiation measure." *Evolution*, **59**(8), 1633–1638. Raymond M (1995). "GENEPOP: Population genetics software for exact tests and ecumenism. Vers. 1.2." *Journal of Heredity*, **86**, 248–249.

Examples

```
data("data_pc_genind")
mat_gst <- mat_pw_gst(data_pc_genind)
path_in <- system.file('extdata', 'gpop_51_sim22_01_25.txt',
                       package = 'graph4lg')
file_n <- file.path(tempdir(), "gpop_51_sim22_01_25.txt")
file.copy(path_in, file_n, overwrite = TRUE)
mat_pw_gst(x = file_n, pop_names = as.character(order(as.character(1:50))))
file.remove(file_n)
```

---

plot_graph_lg *Plot graphs*

Description

The function enables to plot graphs, whether spatial or not.

Usage

```
plot_graph_lg(graph, crds, mode = "spatial", weight = TRUE,
  width = "w", pts_col = "#F2B950")
```

Arguments

graph	A graph object of class igraph
crds	(if 'mode = 'spatial") A data.frame with the spatial coordinates of the graph's nodes. It must have three columns : • ID: A character string indicating the name of the graph's nodes. The names must be the same as the nodes' names of the graph of class igraph (igraph::V(graph)$name) • x: A character string indicating the longitude of the graphs' nodes. • y: A character string indicating the latitude of the graphs' nodes.
mode	A character string indicating whether the graph is spatial ('mode = 'spatial" (default)) or not ('mode = 'aspatial")
weight	A character string indicating whether the links of the graph have weights (T)(default) or not (F)
width	A character string indicating whether the width of the link should be proportional to links' weights ("w", default) or to the inverse of links' weights ("inv", convenient with distances)
pts_col	(optional) A character string indicating the color used to plot the nodes (default: "#F2B950"). It must be a hexadecimal color code or a color used by default in R.

Details

When the graph is not spatial ('mode = 'aspatial"), the nodes coordinates are calculated with Fruchterman et Reingold algorithm. The graph object x of class igraph must have nodes' names (not necessarily in the same order as IDs in crds, given a merging is done).

Value

A ggplot2 object to plot

Author(s)

P. Savary

References

Fruchterman TM, Reingold EM (1991). "Graph drawing by force-directed placement." *Software: Practice and experience*, **21**(11), 1129–1164.

Examples

```
data(pts_pop_simul)
data(data_simul_genind)
mat_w <- mat_gen_dist(data_simul_genind, dist = "DPS")
gp <- gen_graph_topo(mat_w = mat_w, topo = "mst")
g <- plot_graph_lg(graph = gp,
                              crds = pts_pop_simul,
                              mode = "spatial",
                              weight = TRUE)
```

`plot_graph_modul` *Plot the graphs making visible their partition into modules*

Description

The function computes a partition of the graph into modules and plots the graph colouring the nodes with colors corresponding to their respective modules.

Usage

```
plot_graph_modul(graph, algo = "fast_greedy", nb_modul = NULL,
  weight = "inv", crds, mode = "spatial", weight_plot = TRUE,
  width = "inv")
```

Arguments

`graph`	A graph object of class `igraph` to partition into modules and to plot. The graph must be undirected. If 'crds' is not NULL, then the graph's nodes must have names corresponding to the ID's column of 'crds'.
`algo`	(if x and y and graph objects) A character string indicating the algorithm used to create the modules with **igraph**: • If `algo = 'fast_greedy'` (default), function `cluster_fast_greedy` from **igraph** is used (Clauset et al., 2004). • If `algo = 'walktrap'`, function `cluster_walktrap` from **igraph** is used (Pons et Latapy, 2006) with 4 steps (default options). • If `algo = 'louvain'`, function `cluster_louvain` from **igraph** is used (Blondel et al., 2008). In that case, the number of modules created in each graph is imposed. • If `algo = 'optimal'`, function `cluster_optimal` from **igraph** is used (Brandes et al., 2008) (can be very long). In that case, the number of modules created in each graph is imposed.
`nb_modul`	Numeric value indicating the number of modules to create.
`weight`	(optional) A character string or character vector indicating how to weight graphs' links during the calculation of the modularity. • If `weight = 'inv'` (default), then links are weighted with the inverse values of their initial weights. • If `weight = 'w'`, then links are weighted with their initial weights values. • If `weight = 'none'`, then links are not weighted during the calculation. If the graphs' links are not weighted, then this argument is ignored. Links with large weights are considered as stronger connections in the modularity calculation.
`crds`	(if 'mode = 'spatial") A `data.frame` with the spatial coordinates of the graph's nodes. It must have three columns :

- ID: A character string indicating the name of the graph's nodes. The names must be the same as the nodes' names of the graph of class `igraph(igraph::V(graph)$name)`
- x: A character string indicating the longitude of the graphs' nodes.
- y: A character string indicating the latitude of the graphs' nodes.

`mode` A character string indicating whether the graph is spatial ('mode = 'spatial" (default)) or not ('mode = 'aspatial")

`weight_plot` Logical indicating whether the links of the graph have to be displayed on the plot

`width` Logical indicating whether the width of the links on the plot should be proportional to links' weights ("w", default) or to to the inverse of links' weights ("inv", convenient with distances)

Details

When the graph is not spatial ('mode = 'aspatial"), the nodes coordinates are calculated with Fruchterman et Reingold algorithm.

Value

A ggplot2 object to plot

Author(s)

P. Savary

References

Hubert L, Arabie P (1985). "Comparing partitions." *Journal of classification*, **2**(1), 193–218. Fruchterman TM, Reingold EM (1991). "Graph drawing by force-directed placement." *Software: Practice and experience*, **21**(11), 1129–1164. Clauset A, Newman ME, Moore C (2004). "Finding community structure in very large networks." *Physical review E*, **70**(6). Blondel VD, Guillaume J, Lambiotte R, Lefebvre E (2008). "Fast unfolding of communities in large networks." *Journal of Statistical Mechanics - Theory and Experiment*, **10**. Brandes U, Delling D, Gaertler M, Gorke R, Hoefer M, Nikoloski Z, Wagner D (2008). "On modularity clustering." *IEEE transactions on knowledge and data engineering*, **20**(2), 172–188. Pons P, Latapy M (2006). "Computing communities in large networks using random walks." *J. Graph Algorithms Appl.*, **10**(2), 191–218.

Examples

```
data(data_simul_genind)
data(pts_pop_simul)
mat_dist <- suppressWarnings(graph4lg::mat_geo_dist(data=pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
graph <- gen_graph_thr(mat_w = mat_dist, mat_thr = mat_dist,
                           thr = 9500, mode = "larger")
plot_graph_modul(graph = graph, crds = pts_pop_simul)
```

plot_w_hist *Plot histograms of links weight*

Description

The function enables to plot histogram to visualize the distribution of the links' weights

Usage

```
plot_w_hist(graph, fill = "#396D35")
```

Arguments

graph	A graph object of class igraph whose links are weighted
fill	A character string indicating the color used to fill the bars (default: "#396D35"). It must be a hexadecimal color code or a color used by default in R.

Value

A ggplot2 object to plot

Author(s)

P. Savary

Examples

```
data(data_simul_genind)
mat_w <- mat_gen_dist(data_simul_genind, dist = "DPS")
gp <- gen_graph_topo(mat_w = mat_w, topo = "gabriel")
hist <- plot_w_hist(gp)
```

pop_gen_index *Compute population-level genetic indices*

Description

The function computes population-level genetic indices from an object of class genind.

Usage

```
pop_gen_index(x, pop_names = NULL, indices = c("Nb_ind", "A", "He",
  "Ho"))
```

Arguments

x	An object of class genind from package **adegenet**.
pop_names	(optional) A character vector indicating populations' names. It is of the same length as the number of populations. Without this argument, populations are given the names they have initially in the 'genind' object (which is sometimes only a number). The order of the populations' names must match with their order in the 'genind' object. The function does not reorder them. Users must be careful.
indices	(optional) A character vector indicating the population-level indices to compute. These indices can be: • Mean allelic richness by locus by population (indices = c("A",...)) • Mean expected heterozygosity by locus by population (indices = c("He",...)) • Mean observed heterozygosity by locus by population (indices = c("Ho",...)) • Number of individuals by population (indices = c("Nb_ind",...)) • Total allelic richness by population (indices = c("A_tot",...)) By default, indices = c("Nb_ind","A","He","Ho").

Value

An object of class data.frame whose rows correspond to populations and columns to populations' attributes (ID, size, genetic indices). By default, the first column corresponds to the populations' names (ID). The order of the columns depends on the vector 'indices'.

Author(s)

P. Savary

Examples

```
data(data_simul_genind)
x <- data_simul_genind
pop_names <- levels(x@pop)
df_pop_indices <- pop_gen_index(x = x,
                  pop_names = pop_names,
                  indices = c("Nb_ind", "A", "He", "Ho"))
```

---

pts_pop_pc *pts_pop_pc : details on Setophaga plumbea populations*

Description

Setophaga plumbea dataset Data from French Guadeloupe 20 populations sampled by A. Khimoun and S. Garnier

Usage

```
pts_pop_pc
```

Format

An object of class 'data.frame' with the following columns :

id Population ID of the 20 populations, as used during the sampling stage

Longitude Sampling site's (population) longitude (WGS 84)

Latitude Sampling site's (population) latitude (WGS 84)

loc Character string indicating whether the population is located on "Basse-Terre" (BT) or on "Grande-Terre" (GT) (2 main Guadeloupe's islands)

id_publi Population ID of the 20 populations, as used in the paper of Khimoun et al. (2017).

Source

`https://datadryad.org/resource/doi:10.5061/dryad.v8324` There are as many rows as there are sampled populations. In the genetic dataset, some populations were removed, and other were grouped, based on location and numbers of sampled individuals.

References

Khimoun A, Peterman W, Eraud C, Faivre B, Navarro N, Garnier S (2017). "Landscape genetic analyses reveal fine-scale effects of forest fragmentation in an insular tropical bird." *Molecular Ecology*.

Examples

```
data("pts_pop_pc")
str(pts_pop_pc)
```

`pts_pop_simul` *pts_pop_simul : details on simulated populations*

Description

Simulation dataset 50 populations located on a simulated landscape

Usage

```
pts_pop_simul
```

Format

An object of class 'data.frame' with the following columns :

ID Population ID of the 50 populations

x Site's longitude (RGF93)

y Site's latitude (RGF93)

References

Landguth EL, Cushman S (2010). "CDPOP: a spatially explicit cost distance population genetics program." *Molecular Ecology Resources*, **10**(1), 156–161. There are as many rows as there are sampled populations.

Examples

```
data("pts_pop_simul")
str(pts_pop_simul)
```

---

reorder_mat *Reorder the rows and columns of a symmetric matrix*

Description

The function reorders the rows and columns of a symmetric matrix according to a specified order.

Usage

```
reorder_mat(mat, order)
```

Arguments

mat	An object of class matrix
order	A character vector with the rows and columns names of the matrix in the order in which they will be ordered by the function. All its elements must be rows and columns names of the matrix mat.

Details

The matrix mat must be symmetric and have rows and columns names. Its values are not modified.

Value

A reordered symmetric matrix

Author(s)

P. Savary

Examples

```
mat <- matrix(rnorm(36), 6)
mat[lower.tri(mat)] <- t(mat)[lower.tri(mat)]
row.names(mat) <- colnames(mat) <- c("A", "C", "E", "B", "D", "F")
order <- c("A", "B", "C", "D", "E", "F")
mat <- reorder_mat(mat = mat, order = order)
```

scatter_dist *Plot scatterplots of genetic distance vs landscape distance*

Description

The function enables to plot scatterplots to visualize the relationship between genetic distance (or differentiation) and landscape distance (Euclidean distance, cost-distance, etc.)between populations or sample sites.

Usage

```
scatter_dist(mat_gd, mat_ld, method = "loess", thr_gd = NULL,
  thr_ld = NULL, se = TRUE, smooth_col = "black",
  pts_col = "#999999")
```

Arguments

mat_gd	A symmetric matrix with pairwise genetic distances between populations or sample sites.
mat_ld	A symmetric matrix with pairwise landscape distances between populations or sample sites. These distances can be Euclidean distances, cost-distances or resistance distances, among others.
method	A character string indicating the smoothing method used to fit a line on the scatterplot. Possible values are the same as with function 'geom_smooth()' from **ggplot2** : 'lm', 'glm', 'gam', 'loess' (default).
thr_gd	(optional) A numeric value used to remove values from the data before to plot. All genetic distances values above thr_gd are removed from the data.
thr_ld	(optional) A numeric value used to remove values from the data before to plot. All landscape distances values above thr_ld are removed from the data.
se	Logical (optional, default = TRUE) indicating whether the confidence interval around the smooth line is displayed.
smooth_col	(optional) A character string indicating the color used to plot the smoothing line (default: "blue"). It must be a hexadecimal color code or a color used by default in R.
pts_col	(optional) Character string indicating the color used to plot the points (default: "#999999"). It must be a hexadecimal color code or a color used by default in R.

Details

IDs in mat_gd and mat_ld must be the same and refer to the same sampling sites or populations, and both matrices must be ordered in the same way. Matrix of genetic distance mat_gd can be computed using mat_gen_dist. Matrix of landscape distance mat_ld can be computed using mat_geo_dist when the landscape distance needed is a Euclidean geographical distance.

Value

A ggplot2 object to plot

Author(s)

P. Savary

Examples

```
data(data_tuto)
mat_dps <- data_tuto[[1]]
mat_dist <- suppressWarnings(mat_geo_dist(data = pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
scatterplot_ex <- scatter_dist(mat_gd = mat_dps,
                               mat_ld = mat_dist)
```

---

scatter_dist_g *Plot scatterplots of distances to visualize the graph pruning intensity*

Description

The function enables to plot scatterplots of the relationship between two distances (often a genetic distance and a landscape distance between populations or sample sites), while highlighting the population pairs between which a link was conserved during the creation of a graph whose nodes are populations (or sample sites). It thereby allows to visualize the graph pruning intensity.

Usage

```
scatter_dist_g(mat_y, mat_x, graph, thr_y = NULL, thr_x = NULL,
  pts_col_1 = "#999999", pts_col_2 = "black")
```

Arguments

mat_y	A symmetric (complete) matrix with pairwise (genetic or landscape) distances between populations or sample sites. These values will be the points' coordinates on the y axis. mat_y is the matrix used to weight the links of the graph x, whose nodes correspond to rows' and columns' names of mat_y.
mat_x	A symmetric (complete) matrix with pairwise (genetic or landscape) distances between populations or sample sites. These values will be the points' coordinates on the x axis. mat_x and mat_y must have the same rows' and columns' names, ordered in the same way.

graph	A graph object of class igraph. Its nodes must have the same names as the rows' and columns' of mat_y and mat_x matrices. x must have weighted links. Links' weights have to be values from mat_y matrix. graph must be an undirected graph.
thr_y	(optional) A numeric value used to remove values from the data before to plot. All values from mat_y above thr_y are removed from the data.
thr_x	(optional) A numeric value used to remove values from the data before to plot. All values from mat_x above thr_x are removed from the data.
pts_col_1	(optional) A character string indicating the color used to plot the points associated to all populations or sample sites pairs (default: "#999999"). It must be a hexadecimal color code or a color used by default in R.
pts_col_2	(optional) A character string indicating the color used to plot the points associated to populations or sample sites pairs connected on the graph (default: "black"). It must be a hexadecimal color code or a color used by default in R.

Details

IDs in mat_y and mat_x must be the same and refer to the same sampling sites or populations, and both matrices must be ordered in the same way. Matrices of genetic distance can be computed using mat_gen_dist. Matrices of landscape distance can be computed using mat_geo_dist when the landscape distance needed is a Euclidean geographical distance. This function is based upon scatter_dist function.

Value

A ggplot2 object to plot

Author(s)

P. Savary

Examples

```
data(data_tuto)
mat_gen <- data_tuto[[1]]
mat_dist <- suppressWarnings(mat_geo_dist(data=pts_pop_simul,
      ID = "ID",
      x = "x",
      y = "y"))
mat_dist <- mat_dist[order(as.character(row.names(mat_dist))),
                     order(as.character(colnames(mat_dist)))]
x <- gen_graph_topo(mat_w = mat_gen, mat_topo = mat_dist, topo = "gabriel")
scat <- scatter_dist_g(mat_y = mat_gen, mat_x = mat_dist,
                       graph = x)
```

`structure_to_genind` *Convert a file in STRUCTURE format into a genind object*

Description

The function converts a text file in STRUCTURE format into a genind object to use in R

Usage

```
structure_to_genind(path, pop_names = NULL, loci_names = NULL,
  ind_names = NULL)
```

Arguments

`path`	A character string indicating the path to the STRUCTURE file in format .txt, or alternatively the name of the file in the working directory. The STRUCTURE file must only have : • A first column with the IDs of the individuals (can be a simple number) • A second column with the IDs of the populations (can be a simple number) • Some loci columns : as many columns as loci in the data The row for loci's names is optional but recommended. Each individual is displayed on 2 rows.
`pop_names`	(optional) A character vector indicating the populations' names in the same order as in the STRUCTURE file. It is of the same length as the number of populations. Without this argument, populations are numbered from 1 to the total number of individuals.
`loci_names`	A character vector with the names of the loci if not specified in the file's first row. This argument is mandatory if the STRUCTURE file does not include the names of the loci in the first row. In other cases, the names of the loci is extracted from the file's first row
`ind_names`	(optional) A character vector indicating the individuals' names in the same order as in the STRUCTURE file. It is of the same length as the number of individuals. Without this argument, individuals are numbered from 1 to the total number of individuals.

Details

The columns' order of the resulting object can be different from that of objects returned by `gstud_to_genind` and `genepop_to_genind`, depending on alleles' and loci's coding This function uses functions from **pegas** package. For details about STRUCTURE file format : STRUCTURE user manual[3]

Value

An object of type `genind`.

[3] `http://www.ccg.unam.mx/~vinuesa/tlem09/docs/structure_doc.pdf`

Author(s)

P. Savary

Examples

```
data("data_pc_genind")
loci_names <- levels(data_pc_genind@loc.fac)
pop_names <- levels(data_pc_genind@pop)
ind_names <- row.names(data_pc_genind@tab)
path_in <- system.file('extdata', 'tab_gstud_structure.txt',
                       package = 'graph4lg')
file_n <- file.path(tempdir(), "tab_gstud_structure.txt")
file.copy(path_in, file_n, overwrite = TRUE)
str <- structure_to_genind(path = file_n, loci_names = loci_names,
                           pop_names = pop_names, ind_names = ind_names)
file.remove(file_n)
```