

# Package ‘EcoGenetics’

March 3, 2017

**Type** Package

**Title** Spatial Analysis of Phenotypic, Genotypic and Environmental Data

**Version** 1.2.1

**Date** 2017-03-2

**Author** Leandro Roser, Juan Vilardi, Beatriz Saidman and Laura Ferreyra

**Maintainer** Leandro Roser <learoser@gmail.com>

**Description** Management and exploratory analysis of spatial data in population biology. Easy integration of information from multiple sources with 'ecogen' objects.

**License** GPL (>= 2)

**URL** <https://github.com/cran/EcoGenetics>,  
<https://github.com/leandroroser/EcoGenetics-devel>,  
<https://leandroroser.github.io/EcoGenetics-Tutorial>

**LazyLoad** yes

**Depends** R (>= 3.0), methods

**Imports** d3heatmap, edgebundleR, ggplot2, grid, htmlwidgets, igraph,  
jsonlite, magrittr, networkD3, party, pheatmap, plotly, raster,  
reshape2, rgdal, rkt, SoDA, sp

**Suggests** adegenet

**Collate** 'ZZZ.R' 'auxiliar.R' 'int.genind.R' 'ecogen.1OF6.definition.R'  
'ecogen.2OF6.constructor.R' 'ecogen.3OF6.basic.methods.R'  
'ecogen.4OF6.brackets.R' 'ecogen.5OF6.get&set.R'  
'ecogen.6OF6.converters.R' 'accesors.R' 'classes.R' 'control.R'  
'deprecated.R' 'eco.formula.R' 'eco.NDVI.R' 'eco.NDVI.post.R'  
'eco.alfreq.R' 'eco.association.R' 'eco.bearing.R'  
'eco.cbind.R' 'eco.clear.R' 'eco.convert.R' 'eco.cormantel.R'  
'eco.correlog.R' 'eco.detrend.R' 'eco.forestplot.R'  
'eco.format.R' 'eco.gsa.R' 'eco.kin.loiselle.R'  
'eco.lagweight.R' 'eco.lmtree.R' 'eco.lsa.R' 'eco.malecot.R'  
'eco.mantel.R' 'eco.merge.R' 'eco.pairtest.R'  
'eco.plotCorrelog.R' 'eco.plotGlobal.R' 'eco.plotLocal.R'  
'eco.plotWeight.R' 'eco.post.geneland.R' 'eco.rankplot.R'

'eco.rasterplot.R' 'eco.rbind.R' 'eco.remove.R'  
 'eco.slide.con.R' 'eco.slide.matrix.R' 'eco.split.R'  
 'eco.subset.R' 'eco.theilsen.R' 'eco.variogram.R'  
 'eco.weight.R' 'int.break.R' 'int.convert.R' 'int.crosscor.R'  
 'int.geary.R' 'int.jackknife.R' 'int.joincount.R'  
 'int.kin.loiselle.R' 'int.mantel.R' 'int.moran.R'  
 'int.multitable.R' 'int.order.R' 'int.random.test.R'  
 'miscellaneous.R' 'plot.generic.R' 'plot.methods.R'  
 'roxygen.auxiliar.R' 'show\_summary.methods.R'

**RoxygenNote** 5.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2017-03-03 18:01:46

## R topics documented:

EcoGenetics-package . . . . .	4
aue.dataAngle . . . . .	18
aue.df2image . . . . .	19
aue.image2df . . . . .	20
aue.phenosimil . . . . .	20
aue.rescale . . . . .	21
aue.sort . . . . .	22
coordinates . . . . .	23
eco . . . . .	24
eco.alfreq . . . . .	24
eco.association . . . . .	25
eco.bearing . . . . .	26
eco.cbind . . . . .	28
eco.clear . . . . .	29
eco.convert . . . . .	29
eco.cormantel . . . . .	31
eco.correlog . . . . .	34
eco.detrend . . . . .	40
eco.forestplot . . . . .	42
eco.format . . . . .	43
eco.formula . . . . .	45
eco.gsa . . . . .	47
eco.kin.loiselle . . . . .	50
eco.lagweight . . . . .	51
eco.listw2ew . . . . .	54
eco.lmtree . . . . .	55
eco.lsa . . . . .	57
eco.malecot . . . . .	63
eco.mantel . . . . .	68
eco.merge . . . . .	71

eco.NDVI . . . . .	72
eco.NDVI.post . . . . .	73
eco.order . . . . .	75
eco.pairtest . . . . .	76
eco.plotCorrelog . . . . .	77
eco.plotCorrelogB . . . . .	78
eco.plotGlobal . . . . .	79
eco.plotLocal . . . . .	80
eco.plotWeight . . . . .	81
eco.post.geneland . . . . .	82
eco.rankplot . . . . .	83
eco.rasterplot . . . . .	85
eco.rasterplot,eco.multilsa-method . . . . .	87
eco.rbind . . . . .	88
eco.remove . . . . .	89
eco.slide.con . . . . .	90
eco.slide.matrix . . . . .	91
eco.split . . . . .	92
eco.subset . . . . .	93
eco.theilsen . . . . .	94
eco.variogram . . . . .	95
eco.weight . . . . .	97
eco2 . . . . .	102
eco3 . . . . .	102
ecogen . . . . .	103
ecogen2geneland . . . . .	106
ecogen2genepop . . . . .	107
ecogen2genind . . . . .	108
ecogen2gstudio . . . . .	109
ecogen2hierfstat . . . . .	110
ecogen2spagedi . . . . .	111
environment . . . . .	112
genepop2ecogen . . . . .	113
genotype . . . . .	113
grf.multiplot . . . . .	114
grf.seqmultiplot . . . . .	114
phenotype . . . . .	115
plot,eco.multilsa,ANY-method . . . . .	115
spagedi2ecogen . . . . .	116
structure . . . . .	117
summary,eco.mlm-method . . . . .	117
tab . . . . .	118
table.sokal . . . . .	119

## Description

EcoGenetics provides geostatistical tools for the spatial analysis of phenotypic, genotypic and environmental data. The package has been designed to make easy the storing, handling and integration of the available information from multiple sources, under an S4 philosophy.

## Details

Package: EcoGenetics  
 Type: Package  
 Version: 1.2.0-3  
 Date: 2016-07-15  
 License: GPL (>=2)

---

### I. STRUCTURE OF THE PACKAGE

EcoGenetics has four basic modules. The **base module** is composed by general functions ([multiple lm](#), [detrrending spatial data utility](#), etc.). The **general spatial module** computes global ([Moran's I](#), [Mantel test](#), etc.) and local ([Getis-Ord's G\\*](#), [local Moran's I](#), etc.) spatial tests. These analyses use a [spatial weights matrix](#), provided by the **spatial weights module**. The **lag analysis module** performs two basic analyses: the obtention of [variograms](#) and [correlograms](#) (see also [this link](#)). This module uses the other tool provided by the spatial weights module: a sequence of [spatial weights matrices](#).

The package have also special plot methods, as [rankplot](#), [forestplot](#), [rasterplot](#) (implemented the last two for [local spatial analysis](#)). Several conversor of data from/to other programs are available, (as to [genepop](#) - an [importer](#) tool is also defined for [genepop](#)-, [SPAGeDi](#), , etc.). Basic manipulation of genetic matrices is allowed by [eco.convert](#) and [eco.format](#). Tools for computation of NDVI in Landsat imaginery, post-process of rasters and temporal analysis can be found in [eco.NDVI](#), [eco.NDVI.post](#) and [eco.theilsen](#). Other useful functions are [aue.sort](#) (for ordering alleles), [eco.alfreq](#) (to comput histograms of allelic frequencies for detection of bottlenecks), [eco.post.geneland](#) and [eco.pairtest](#).

The results obtained with the main functions defined in EcoGenetics are object of class [S4](#). As a default characteristic of the package design, these objects have a "show" method for a general overview of the results, and methods to extract the results stored in slots (generic [accessors](#) and double square brackets ("[[") definitions).

For storing and pre-processing the data to analyze, the package defines a special class: the class [ecogen](#).

### II. STRUCTURE OF ECOGEN OBJECTS: HANDLING AND INTEGRATING INFORMATION

Ecological genetics research requires the integration of data originated in different sources. The class `ecogen` has been designed for handling multidimensional data. Its basic structure is the following:

- An **XY** slot, storing a data frame with geographic coordinates.
- A **P** slot, storing a phenotypic data frame.
- A **G** slot, storing a genotypic data frame.
- An **A** slot containing as allelic frequencies the information of G (only available for codominant markers.)
- An **E** slot, storing an environmental data frame.
- A **S** slot, storing a data frame with classes assigned to the individuals.
- A **C** slot, for a custom data frame.
- An **OUT** slot, containing a list for the storage of the results.

For dominant (presence-absence) markers, the slot A is empty.

### **III. A BRIEF OVERVIEW OF THE DATA HANDLING METHODS DEFINED FOR ECOGEN OBJECTS**

The construction of a new "ecogen" object from a data frame is made with the homonymous function.

```
library("EcoGenetics")
```

```
data(eco.test)
```

```
eco <- ecogen(XY = coordinates, P = phenotype, G = genotype, E = environment, S = structure,
order.G = TRUE)
```

# The following methods can be used with ecogen objects:

```
# - - - - -
```

#### **# 1. subsetting/ordering by row-method, using single square brackets ("[]")**

```
eco.sub <- eco[1:50]
```

```
eco.ord <- eco[c(50:1)]
```

```
eco.ord[["XY"]] # object ordered and subset rows. This is a powerful method for handling the stack
of ecogen data frames
```

```
eco <- eco[order(eco[["S"]], 1)] #ordering S[, 1] and then the object
```

#### **# 2. merging-method, for two objects**

```
eco1 <- eco
```

```
merged <- eco.merge(eco, eco1)
```

#### **# 3. subsetting-method, in reference to a group in the S slot (in this case, "1")**

```
eco.subS <- eco.subset(eco, "pop", 1)
```

#### **# 4. binding by row-method (duplicated row names not allowed)**

```
eco2 <- eco; rownames(eco2[["P"]]) <- 226:450
```

```
eco.r <- eco.rbind(eco, eco2)
```

#### **# 5. binding by column-method**

```

eco.c <- eco.cbind(eco, eco1)
# 6. get-method using generic accessors and "[" (equivalent methods):
ecoslot.XY(eco) ; eco[["XY"]]
ecoslot.P(eco); eco[["P"]]
ecoslot.G(eco); eco[["G"]]
ecoslot.A(eco); eco[["A"]] (only available for codominant data)
ecoslot.E(eco); eco[["E"]]
ecoslot.S(eco); eco[["S"]]
ecoslot.C(eco); eco[["C"]]
ecoslot.OUT(eco); eco[["OUT"]]
# 7. set-method using accessors and "[" (equivalent methods):
eco.temp <- ecogen(XY = coordinates, P = phenotype)
eco.temp
ecoslot.G(eco.temp, order.G = TRUE) <- genotype
## this is equivalent, in square brackets notation, to:
eco[["G", order.G=TRUE]] <- genotype
ecoslot.E(eco.temp) <- environment
## identical to eco[["E"]] <- environment
ecoslot.S(eco.temp) <- structure
## identical to eco[["S"]] <- structure
eco.temp
# 8. appending-method (storing information generated via accessors)
## fitting a multiple linear regression model:
linear.analysis <- eco.lmtree(eco[["P"]], eco[["E"]], "mlm")
## storing the results:
ecoslot.OUT(eco) <- linear.analysis
eco
## Storing multiple result at once:
a <- c(1:10)
b <- c(2:30)
ecoslot.OUT(eco) <- list(eco, a, b)
eco
## the use of the accessor OUT has its equivalence in double square brackets notation:
eco[["OUT"]] <- list(eco, a, b)
## summary table
ecoslot.OUT(eco)

```

## the data frame shows the results stored in alphabetical order and their classes. The specification of a second name, return the corresponding stored object:

```
ecoslot.OUT(eco, "a")
```

## note that the append method is a particular case of the ## set method with accessors / "[" for the slot OUT.

### # 9. removing-method

## removing objects a and b from eco

```
eco <- eco.remove(eco, a, b)
```

-----

The workspace can be [cleared](#), only storing the desired object:

```
ls()
```

```
eco.clear(eco)
```

```
ls()
```

The cleaner and compact option, is to save the object as [RDS](#), which is a way to save R objects as a file:

```
saveRDS(eco, "myObject.rds")
```

The object can be loaded in the future with the function readRDS:

```
eco <- readRDS("myObject.rds")
```

————— o —————

### Author(s)

Leandro Roser, Juan Vilardi, Beatriz Saidman and Laura Ferreyra

Maintainer: Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

- Anselin L. 1995. Local indicators of spatial association-LISA. *Geographical analysis*. 27: 93-115.
- Borcard D., F. Gillet, and P. Legendre. 2011. *Numerical ecology with R*. Springer Science & Business Media.
- Chander G., B. Markham, and D. Helder. 2009. Summary of current radiometric calibration coefficients for Landsat MSS, TM, ETM+, and EO-1 ALI sensors. *Remote sensing of environment*, 113: 893-903.
- Chavez P. 1989. Radiometric calibration of Landsat Thematic Mapper multispectral images. *Photogrammetric Engineering and Remote Sensing*, 55: 1285-1294.
- Chavez P. 1996. Image-based atmospheric corrections-revisited and improved. *Photogrammetric engineering and remote sensing*, 62: 1025-1035.
- Double M., R. Peakall, N. Beck, and Y. Cockburn. 2005. Dispersal, philopatry, and infidelity: dissecting local genetic structure in superb fairy-wrens (*Malurus cyaneus*). *Evolution* 59: 625-635.
- Dray S., and A., Dufour. 2007. The ade4 package: implementing the duality diagram for ecologists. *Journal of statistical software*, 22: 1-20.

- Freedman D., and P. Diaconis. 1981. On the histogram as a density estimator: L<sup>2</sup> theory. *Probability theory and related fields*, 57: 453-476.
- Geary R. 1954. The contiguity ratio and statistical mapping. *The incorporated statistician*, 115-146.
- Getis A., and J. Ord. 1992. The analysis of spatial association by use of distance statistics. *Geographical analysis*, 24: 189-206.
- Goslee S. 2011. Analyzing remote sensing data in R: the landsat package. *Journal of Statistical Software*, 43: 1-25.
- Goudet J. 2005. Hierfstat, a package for R to compute and test hierarchical F-statistics. *Molecular Ecology Notes*, 5: 184-186.
- Guillot G., F. Mortier and A. Estoup. 2005. GENELAND: a computer package for landscape genetics. *Molecular Ecology Notes*, 5: 712-715.
- Jombart T. 2008. adegenet: a R package for the multivariate analysis of genetic markers. *Bioinformatics*, 24: 1403-1405.
- Kalisz S., J. Nason, F.M. Handazawa, and S. Tonsor. 2001. Spatial population genetic structure in *Trillium grandiflorum*: the roles of dispersal, mating, history, and selection. *Evolution* 55: 1560-1568.
- Legendre P., and L. Legendre. 2012. *Numerical ecology*. Third English edition. Elsevier Science, Amsterdam, Netherlands.
- Lichstein J., T. Simons, S. Shiner, and K. Franzreb. 2002. Spatial autocorrelation and autoregressive models in ecology. *Ecological monographs*, 72: 445-463.
- Loiselle B., V. Sork, J. Nason, and C. Graham. 1995. Spatial genetic structure of a tropical understory shrub, *Psychotria officinalis* (Rubiaceae). *American Journal of Botany* 1420-1425.
- Moran P. 1950. Notes on continuous stochastic phenomena. *Biometrika*, 17-23.
- Oden, N., 1984. Assessing the significance of a spatial correlogram. *Geographical Analysis*, 16: 1-16.
- Oden N., and R. Sokal. 1986. Directional autocorrelation: an extension of spatial correlograms to two dimensions. *Systematic Zoology*, 35: 608-617.
- Ord J., and A. Getis. 1995. Local spatial autocorrelation statistics: distributional issues and an application. *Geographical analysis*, 27: 286-306.
- Reich R., R. Czaplewski and W. Bechtold. 1994. Spatial cross-correlation of undisturbed, natural shortleaf pine stands in northern Georgia. *Environmental and Ecological Statistics*, 1: 201-217.
- Sokal R., and N. Oden 1978. Spatial autocorrelation in biology: 1. Methodology. *Biological journal of the Linnean Society*, 10: 199-228.
- Sokal R., and N. Oden. 1978. Spatial autocorrelation in biology. 2. Some biological implications and four applications of evolutionary and ecological interest. *Biological Journal of the Linnean Society*, 10: 229-49.
- Sokal R. 1979. Ecological parameters inferred from spatial correlograms. In: G. Patil and M. Rosenzweig, editors. *Contemporary Quantitative Ecology and related Ecometrics*. International Co-operative Publishing House: Fairland, MD, pp. 167-96.
- Sokal R. 1986. Spatial data analysis and historical processes. In: E. Diday, Y. Escoufier, L. Lebart, J. Pages, Y. Schektman, and R. Tomassone, editors. *Data analysis and informatics, IV*. North-Holland, Amsterdam, The Netherlands, pp. 29-43.



Sokal R., N. Oden and B. Thomson. 1998. Local spatial autocorrelation in a biological model. *Geographical Analysis*, 30: 331-354.

Sokal R., and B. Thomson. 2006. Population structure inferred by local spatial autocorrelation: an example from an Amerindian tribal population. *American journal of physical anthropology*, 129: 121-131.

Song C., C. Woodcock, K. Seto, M. Lenney and S. Macomber. 2001. Classification and change detection using Landsat TM data: when and how to correct atmospheric effects?. *Remote sensing of Environment*, 75: 230-244.

Sturges H. 1926. The choice of a class interval. *Journal of the American Statistical Association*, 21: 65-66.

Tucker C. 1979. Red and photographic infrared linear combinations for monitoring vegetation. *Remote sensing of Environment*, 8: 127-150.

Vekemans, X., and O. Hardy. 2004. New insights from fine-scale spatial genetic structure analyses in plant populations. *Molecular Ecology*, 13: 921-935.

Wu. 1986. Jackknife, bootstrap and other resampling methods in regression analysis. *the Annals of Statistics*, 1261-1295.

## Examples

```
## Not run:

#---Detrending spatial data with polynomial interpolation---#

data(eco2)

## original data
data1 <- matrix(eco2[["P"]][,1], 30, 30)
image(data1)

## original data + trend
data2 <- matrix(eco2[["P"]][,2], 30, 30)
image(data2)

## data detrending
data2.det <- eco.detrend(Z = eco2[["P"]][,2], XY = eco2[["XY"]], degree = 1)
data2.det <- ecoslot.RES(data2.det)
data2.det <- matrix(data2.det$df1, 30, 30)
image(data2.det)

#---Multiple Linear Regression fit---#

data(eco.test)
mymod <- "E1+E2*E3"
mod <- eco.lmtree(df1 = eco[["XY"]], df2 = eco[["E"]],
```

```

analysis = "mlm", mod.class = mymod)
summary(mod)

#---Multiple Conditional Inference Trees---#

data(eco.test)
mymod <- "E1+E2+E3"
mod <- eco.lmtree(df1 = eco[["P"]], df2 = eco[["E"]],
analysis = "mctree", mod.class = mymod, fact = eco[["S"]]$structure)
summary(mod)

#---Global spatial analysis---#

## Moran's I

### one test
data(eco.test)
con <- eco.weight(eco[["XY"]], method = "circle", d1 = 0, d2 = 2)
global <- eco.gsa(Z = eco[["P"]][, 1], con = con, , method = "I", nsim = 200)
global

require(adegenet)
con2<-chooseCN(eco[["XY"]], type = 1, result.type = "listw", plot.nb = FALSE)
global <- eco.gsa(Z = eco[["P"]][, 1], con = con2, , method = "I", nsim = 200)
global

### multiple tests
con <- eco.weight(eco[["XY"]], method = "circle", d1 = 0, d2 = 2)
global <- eco.gsa(Z = eco[["P"]], con = con, , method = "I", nsim = 200)
global

## Geary's C

data(eco.test)
global.C <- eco.gsa(Z = eco[["P"]][, 1], con = con, method = "C", nsim = 200)
global.C

## Bivariate's Moran's Ixy

data(eco.test)
global.Ixy <- eco.gsa(Z = eco[["P"]][, 1], Y = eco[["E"]][, 1],
con = con, method = "CC", nsim = 200)
global.Ixy

## Join-Count

```

```

data(eco.test)
global.JC <- eco.gsa(Z = 2* eco[["A"]][, 1], ncod = 1,
con = con, method = "JC", nsim = 5)
global.JC

# Mantel test

data(eco.test)
eco.mantel(d1 = dist(eco[["P"]]), d2 = dist(eco[["E"]]), nsim = 99)

## Partial Mantel test

data(eco.test)
eco.mantel(d1 = dist(eco[["P"]]), d2 = dist(eco[["E"]]),
dc = dist(eco[["XY"]]), nsim = 99)

#---Local spatial analysis---#

## Getis-Ord's G*

data(eco.test)
require(ggplot2)
con<- eco.weight(eco[["XY"]], method = "knearest", k = 4, self = TRUE)
### self = TRUE for G*
getis.ak <- eco.lsa(eco[["P"]][, 1], con, method = "G*", nsim = 99,
adjust = "none")
getis.ak

### to plot the results, the function "eco.lsa" calls "eco.rankplot"
### (see ?eco.rankplot) when test = "permutation" and "eco.forestplot"
### (see ?eco.forestplot) when test = "bootstrap"

p <- plot(getis.ak) ### rankplot graph
p ### points with colors of the color-scale:
### points with P < 0.05. Yellow points :
### points with P > 0.05

p <- plot(getis.ak, significant = FALSE)
p ### all points have a color of the color-scale

### bootstrap example
getis.akb <- eco.lsa(eco[["P"]][, 1], con, method = "G*", nsim = 99,
test = "bootstrap")
p <- plot(getis.akb) ### forestplot graph
p + ggplot2::theme_bw() ### the plot can be modified with ggplot2
### In this case, the background color is modified

## Getis-Ord's G

```

```

data(eco.test)
require(ggplot2)
con<- eco.weight(eco[["XY"]], method = "knearest", k = 4)
### self = FALSE for G
getis <- eco.lsa(eco[["P"]][, 1], con, method = "G", nsim = 99, adjust = "none")
plot(getis)

## Local Moran's I

#-----
# TESTING PHENOTYPIC DATA-
#-----

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE

# test for the first trait of the data frame P
localmoran <- eco.lsa(eco[["P"]][, 1], con, method = "I", nsim = 99)

plot(localmoran)

# test for several variables

all.traits <- apply(eco[["P"]], 2, eco.lsa, con, method = "I", nsim = 99)

# Observed statistic and P-values tables (individuals x traits)
stat.P <- sapply(all.traits, function(x) return(ecoslot.OUT(x)[,1]))
pval.P <- sapply(all.traits, function(x) return(ecoslot.OUT(x)[,4]))

# Plot of the phenotypic spatial patterns

par(mfrow = c(2,4))
for(i in 1:8) {
  image(matrix(stat.P[,i], 15,15))
}

par(mfrow = c(2,4))
for(i in 1:8) {
  image(matrix(pval.P[,i], 15,15))
}

#-----
# TESTING GENOTYPIC DATA-
#-----

# eco[["A"]] is a matrix with the genetic data of "eco"
# as frequencies for each allele in each individual
# (use [["G"]] for presence-absence data).

head(eco[["A"]])      # head of the matrix - 40 alleles

```

```

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE

# test for a single allele
localmoran.geno <- eco.lsa(eco[["A"]][, 32], con, method = "I", nsim = 99)

# test for several alleles - 40 alleles (it runs in less than 1 min
# for 99 simulations per allele; 999 simulations takes ~ 11 s per allele,
# less than 8 min in total.)
all.alleles <- apply(eco[["A"]], 2, eco.lsa, con, method = "I", nsim = 99)

# plot all alleles to get an overview of the spatial patterns
lapply(all.alleles, plot)

# Observed statistic and P-values tables (individuals x loci)
stat.G <- sapply(all.alleles, function(x) return(ecoslot.OUT(x)[,1]))
pval.G <- sapply(all.alleles, function(x) return(ecoslot.OUT(x)[,4]))

# counting individuals with P < 0.05 for each allele
# (5 * 225 / 100 ~ 12 significant tests by random)
signif <- lapply(all.alleles, function(x) sum(ecoslot.OUT(x)[,4] < 0.05))
signif <- unlist(signif)

# filtering alleles, loci with > 12 significant individual tests

A.local <- eco[["A"]][, signif > 12]      #filtered matrix
stat.G.f <- stat.G[, signif > 12]
pval.G.f <- pval.G[, signif > 12]

# Plot of the genotypic spatial patterns

# one plot possibility, using the EcoGenetics method <rankplot>
all.local <- all.alleles[signif > 12]
lapply(all.local, plot)

# other plot possibility, using <image>
par(mfrow = c(3,4))
for(i in 1:12) {
  image(matrix(stat.G[,i], 15,15))
}

par(mfrow = c(3,4))
for(i in 1:12) {
  image(matrix(pval.G[,i], 15,15))
}

## Local Geary's C

data(eco.test)
require(ggplot2)
con<- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
### row standardized weights = TRUE

```

```

localgeary <- eco.lsa(eco[["P"]][, 1], con, method = "C", nsim = 99,
adjust = "none")
plot(localgeary)

#---Moran's I, Geary's C and bivariate Moran's Ixy correlograms---#

## Moran's I correlogram

### single test
data(eco.test)
require(ggplot2)
moran <- eco.correlog(Z=eco[["P"]][,1], XY = eco[["XY"]], method = "I",
smax=10, size=1000)
plot(moran)

### multiple tests
moran2 <- eco.correlog(Z=eco[["P"]], XY = eco[["XY"]], method = "I",
smax=10, size=1000)
plot(moran2, var = "P2") ## single plots
plot(moran2, var = "P3") ## single plots

graf <- plot(moran2, meanplot = TRUE) ## multiple plot with mean correlogram
## and jackknifed confidence intervals.

plot(graf[[1]])
plot(graf[[2]])

### correlogram plots support the use of ggplot2 syntax
moranplot <- plot(moran2, var = "P3") + theme_bw() + theme(legend.position="none")
moranplot

## Geary's C correlogram

data(eco.test)
require(ggplot2)
geary <- eco.correlog(Z = eco[["P"]][,1], XY = eco[["XY"]], method = "C",
smax=10, size=1000)
plot(geary)

## Moran's Ixy cross-correlogram

data(eco.test)
require(ggplot2)
cross <- eco.correlog(Z = eco[["P"]][,1], XY = eco[["XY"]], Y = eco[["E"]][, 1],
method = "CC", int = 2, smax = 15)
plot(cross)

#---Mantel and partial Mantel correlograms---#

```

```

data(eco.test)
require(ggplot2)

## Mantel correlogram

corm <- eco.cormantel(M = dist(eco[["P"]]), size=1000,smax=7, XY = eco[["XY"]],
  nsim = 99)
plot(corm)

corm <- eco.cormantel(M = dist(eco[["P"]]), size=1000,smax=7, XY = eco[["XY"]],
  nsim = 99, test = "bootstrap")
plot(corm)

## Partial Mantel correlogram

corm <- eco.cormantel(M = dist(eco[["P"]]), MC = dist(eco[["E"]]),
  size=1000, smax=7, XY = eco[["XY"]], nsim = 99)
plot(corm)

### correlogram plots support the use of ggplot2 syntax
mantelplot <- plot(corm) + theme_bw() + theme(legend.position="none")
mantelplot

#---Empirical variogram---#

data(eco.test)
require(ggplot2)
variog <- eco.variogram(Z = eco[["P"]][, 2],XY = eco[["XY"]])
plot(variog)

### variogram plots support the use of ggplot2 syntax
variogplot <- plot(variog) + theme_bw() + theme(legend.position="none")
variogplot

#---Computing NDVI with atmospheric correction
#---over a time series, and estimation of the temporal mean---#

require(raster)
data(tab)
data(eco3)
set.seed(6)

temp <- list()

## we create 4 simulated rasters for the data included in the object tab:

```

```

for(i in 1:4) {
temp[[i]] <- runif(19800, 0, 254)
temp[[i]] <- matrix(temp[[i]], 180, 110)
temp[[i]] <- raster(temp[[i]], crs="+proj=utm")
extent(temp[[i]])<-c(3770000, 3950000, 6810000, 6920000)
}

writeRaster(temp[[1]], "20040719b4.tif", overwrite = T)
writeRaster(temp[[2]], "20040719b3.tif", overwrite = T)
writeRaster(temp[[3]], "20091106b4.tif", overwrite = T)
writeRaster(temp[[4]], "20091106b3.tif", overwrite = T)

## Computing NDVI images:

eco.NDVI(tab, "COST", "NDVI", "LT5")

## Mean NDVI image computed over the NDVI images that we calculated:

eco.NDVI.post(tab, "COST", "NDVI", what = c("mean", "var"))
mean.ndvi <- raster("NDVI.COST.mean.tif")
plot(mean.ndvi)

## Extraction of the mean NDVI for each point in the object eco and plot of the data:

ndvi <- extract(mean.ndvi, eco3[["XY"]])
ndvi<- aue.rescale(ndvi)
plot(eco3[["XY"]][, 1], eco3[["XY"]][, 2], col=rgb(ndvi, 0, 0),
pch=15, main = "Mean NDVI", xlab = "X", ylab = "Y")

#---Theil-sen estimation for a raster---#

require("raster")
set.seed(6)

temp <- list()

for(i in 1:100) {
temp[[i]] <- runif(36,-1, 1)
temp[[i]] <- matrix(temp[[i]], 6, 6)
temp[[i]] <- raster::raster(temp[[i]])
}

temp <- brick(temp)

writeRaster(temp,"temporal.tif", overwrite=T)
rm(temp)
ndvisim <- brick("temporal.tif")
date <- seq(from = 1990.1, length.out = 100, by = 0.2)
eco.theilsen(ndvisim, date)
pvalue <- raster("pvalue.tif")
slope <- raster("slope.tif")
par(mfrow = c(1, 2))

```



```

plot(pvalue, main = "p-value")
plot(slope, main = "slope")

#---Conversion of ecogen data to several formats and format tool---#

## Interconverting an ecogen genetic data frame among several formats

data(eco3)

### One allele per column
loc2al <- eco.convert(eco3[["G"]], "matrix", "alleles.matrix", ploidy = 2)
loc2al

### Inverse operation (collapse alleles into locus)
al2loc <- eco.convert(loc2al, "alleles.matrix", "matrix", ploidy = 2)
al2loc

### Separating alleles with a character string
loc2loc <- eco.convert(eco3[["G"]], "matrix", "matrix", ploidy = 2, sep.out = "/")
loc2loc

### Inverse operation (removing separator)
loc2loc.nosep <- eco.convert(loc2loc, "matrix", "matrix", ploidy = 2, sep.in = "/", sep.out = "")
loc2loc.nosep

## Format tool

data(eco.test)

### Adding zeros

example <- as.matrix(genotype[1:10,])
mode(example) <- "character"
### example data
example
recoded <- eco.format(example, ncod = 1, ploidy = 2, nout = 3)
### recoded data
recoded
### leading zeros
recoded2 <- eco.format(example, ncod = 1, ploidy = 2, nout = 3,
fill.mode = "first")
### recoded data
recoded2

### Tetraploid data, separating alleles with a "/"
tetrap <- as.matrix(example)
### simulated tetraploid example data
tetrap <- matrix(paste(example,example, sep = ""), ncol = ncol(example))
recoded <- eco.format(tetrap, ncod = 1, ploidy = 4, sep.out = "/")

```

```
### recoded data
recoded

## Creating input data for Geneland with an ecogen object

data(eco.test)
eco.2geneland(eco, 1)

## Exporting an ecogen genetic data frame into Genepop format

data(eco.test)
eco.2genepop(eco, grp = "pop", name = "infile.genepop.txt")
### an output file "infile.genepop.txt" is generated in the working directory

## Converting a diploid ecogen genetic data frame into a gstudio object

data(eco.test)
gsteco <- eco.2gstudio(eco, "separated")
gsteco

## Converting an ecogen genetic data frame into a hierfstat data frame

data(eco.test)
hiereco <- eco.2hierfstat(eco, "pop")

## Exporting an ecogen genetic data frame into SPAGeDI format

data(eco.test)
eco.2spagedi(eco, "pop", ndig = 1,int=2, smax=6, name="infile.spagedi.txt")

## End(Not run)
```

---

aue.dataAngle

*Angles for an XY coordinates matrix*

---

### **Description**

Angles for an XY coordinates matrix

### **Usage**

```
aue.dataAngle(XY, maxpi = FALSE, deg = FALSE, latlon = FALSE)
```

**Arguments**

XY	XY matrix with projected coordinates
maxpi	angles bounded between 0 - pi?
deg	angles in decimal degrees?
latlon	Are the coordinates in decimal degrees format? Defalut FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

aue.df2image

*Transforming a data frame into a raster*

---

**Description**

This is the inverse function of `aue.image2df`

**Usage**

```
aue.df2image(x, origin = c("upperleft", "lowerleft"))
```

**Arguments**

x	output of <code>aue.image2df</code>
origin	Origin of the reference for the coordinates. Default: upperleft.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

aue.image2df	<i>Transforming a raster into a data frame with cartesian coordinates</i>
--------------	---

---

**Description**

This function returns a data frame with the column number (x), row number (y) and cell value (z) of each pixel in a raster.

**Usage**

```
aue.image2df(mat, origin = c("upperleft", "lowerleft"),
             out = c("data.frame", "matrix"))
```

**Arguments**

mat	Input raster matrix.
origin	Origin of the reference for the coordinates. Default: upperleft.
out	output format: "data.frame" (default) or "matrix".

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:

ras <- matrix(eco[["P"]][,1],15,15)
image(ras)
ras.row <- aue.image2df(ras)
ras.row
image(matrix(ras.row[,3], 15, 15))

## End(Not run)
```

---

aue.phenosimil	<i>Phenotypic similarity for vector, matrix or data frame according to Ritland (1996)</i>
----------------	---

---

**Description**

Phenotypic similarity for vector, matrix or data frame according to Ritland (1996)

**Usage**

```
aue.phenosimil(X)
```

**Arguments**

X                      Data frame, matrix or vector. If X is not a vector, the program returns a list of matrices.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

aue.rescale                      *Scaling a data frame or matrix to [0, 1] or [-1, 1] ranges*

---

**Description**

This program scales each column of a data frame or a matrix to [0,1] range, computing  $((X)_{ij} - (X_{\min})_i) / \text{range}((X)_i)$  for each individual  $j$  of the variable  $i$  or to [-1,1] range computing  $2 * ((X)_{ij} - (X_{\min})_i) / \text{range}((X)_i) - 1$

**Usage**

```
aue.rescale(dfm, method = c("zero.one", "one.one"))
```

**Arguments**

dfm                      Data frame, matrix or vector to scale.  
method                      Scaling method: "zero.one" for scaling to [0, 1], "one-one" for scaling to [-1,1].

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:

data(eco.test)
require(adeigenet)
pc <- dudi.pca(eco[["P"]], scannf = FALSE, nf = 3)
pc <- pc$li
pc <- aue.rescale(pc)
plot(eco[["XY"]][, 1], eco[["XY"]][, 2], col = rgb(pc), pch = 16,
cex = 1.5, xlab = "X", ylab = "Y")

## End(Not run)
```

---

aue.sort	<i>Ordering the content of cells in a matrix. Ordering alleles in a genetic matrix.</i>
----------	---

---

### Description

This program takes a matrix and orders the content of each cell. It was specially designed for genetic data, but can be used with any data that can be rearranged by the function `order`. The arguments `ploidy` and `ncode` determine the mode of ordering the data. The cells corresponding to each individual  $i$  and loci  $j$  are ordered in ascending order in default option (it can be passed `decreasing = TRUE` as argument, if descending order is desired). For example, a locus with ploidy = 2 and `ncod = 1`, coded as 51, for an individual, will be recoded as 15. A locus with ploidy = 3 and coded as 143645453, will be recoded as 143453645 (alleles 143, 454 and 645).

### Usage

```
aue.sort(X, ncod = NULL, ploidy = 2, sep.loc = "", chk.plocod = TRUE,
...)
```

### Arguments

<code>X</code>	Any matrix with content to order.
<code>ncod</code>	Number of digits coding each allele (e.g., 1: x, 2: xx, 3: xxx, etc.). If <code>NULL</code> , <code>ncode</code> will be obtained from the ploidy and the maximum number of characters in the data cells.
<code>ploidy</code>	Ploidy of the data.
<code>sep.loc</code>	Character string separating alleles.
<code>chk.plocod</code>	Default <code>TRUE</code> . The function checks coherence in ploidy and number of digits coding alleles.
<code>...</code>	Additional arguments passed to <code>order</code>

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:

# Example 1-----

geno <- c(12, 52, 62, 45, 54, 21)
geno <- matrix(geno, 3, 2)

# ordering the data
aue.sort(geno, ploidy = 2)
```

```
# decreasing sort order
aue.sort(geno, ploidy = 2, decreasing = TRUE)

# Example 2-----

geno2 <- c(123456, 524556, 629359, 459459, 543950, 219405)
geno2 <- matrix(geno2, 3, 2)

# ordering the data as diploid
aue.sort(geno2, ploidy = 2) # the data is ordered using blocks of 3 characters

# ordering the data as triploid
aue.sort(geno2, ploidy = 3) # the data is ordered using blocks of 2 characters

# error: the ploidy and the number of characters are not congruent
aue.sort(geno2, ploidy = 5)

# error: the ploidy and the number of characters are not congruent
aue.sort(geno2, ploidy = 5)

# Example 3-----

# any character data
generic <- c("aldk", "kdbf", "ndnd", "ndkd")
generic <- matrix(generic, 2, 2)
aue.sort(generic, ploidy = 2)
aue.sort(generic, ploidy = 4)

## End(Not run)
```

---

coordinates

*Coordinates*

---

### **Description**

Data frame with cartesian coordinates of 225 simulated individuals.

### **Usage**

```
data(eco.test)
coordinates
```

### **Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

eco

*Eco*

---

**Description**

ecogen object with simulated data of 225 individuals.

**Usage**

```
data(eco.test)
eco
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

eco.alfreq

*Allelic frequency histograms for an ecogen genetic data frame*

---

**Description**

This program computes the frequency of each allele and plots a histogram for the number of alleles with a given frequency. The distribution is expected to be L-shaped under mutation-drift equilibrium. When a factor is given, the program plots a histogram for each group.

**Usage**

```
eco.alfreq(eco, grp = NULL)
```

**Arguments**

eco	Object of class "ecogen".
grp	Optional factor (column of the S slot) for plots by group.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

Luikart G., F. Allendorf, F. Cornuet, and W. Sherwin. 1998. Distortion of allele frequency distributions provides a test for recent population bottlenecks. *Journal of Heredity*, 89: 238-247.



**Examples**

```
## Not run:

data(eco.test)
eco.alfreq(eco)
eco.alfreq(eco, "pop")

## End(Not run)
```

---

eco.association	<i>Chi-square and Fisher's exact test for association of loci and alleles with a factor</i>
-----------------	---

---

**Description**

Chi-square and Fisher's exact test for association of loci and alleles with a factor

**Usage**

```
eco.association(eco, assoc = c("within", "between"), x,
  method = c("fisher.test", "chisq.test"), nrep = 99, adjust = "none",
  ndig = NA)
```

**Arguments**

eco	Object of class "ecogen".
assoc	"between" if the association test should be performed between a factor and a loci, or "within" if the association test should be performed between a factor and alleles within loci. For haploid data, use option "within".
x	The name of the S slot column with the groups for the association test.
method	Test method ("chisq.test" or "fisher.test"). Default is "fisher.test".
nrep	Number of repetitions for the permutation test.
adjust	Correction method of P-values for multiple tests, passed to <a href="#">p.adjust</a> . Default is "none" (no correction).
ndig	Number of digits coding each alleles (e.g. 2: xx, or 3: xxx) when assoc is "within".

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[chisq.test](#) [fisher.test](#) [fisher.test](#)

## Examples

```
## Not run:

data(eco.test)
eco.association(eco, "within", "pop")
eco.association(eco, "within", "pop", adjust="fdr")
eco.association(eco, "within", "pop", method = "chisq.test")
eco.association(eco, "between", "pop", ndig = 1)
eco.association(eco, "between", "pop", method = "chisq.test", ndig = 1)

## End(Not run)
```

---

eco.bearing

*Angular Spatial Weights*

---

## Description

Construction of angular spatial weights

## Usage

```
eco.bearing(con, theta, XY = NULL, latlon = FALSE)
```

## Arguments

con	an eco.weight or eco.lagweight object
theta	reference angle in degrees, between 0 and 180, counterclockwise, with 0 representing the positive x axis, 90 representing the positive y axis, 180 representing the negative x axis. Note that 0 and 180 yield identical results.
XY	Matrix/data frame with projected coordinates. Default NULL.
latlon	Are the coordinates in decimal degrees format? Defalut FALSE. If TRUE, the coordinates must be in the XY matrix/data frame with longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

## Details

This program computes an angular weights object (AW) (or a list of AW). If a weights object is passed as argument ("con") the program computes an AW with this element. If XY is passed, the program first compute a matrix of N x N, where N is the number of rows in XY, and then uses the matrix as input to compute the AW. Each element in the weights matrices is then weighted by the squared cosine of the angle formed with the x positive axis by aline connecting the pair of points. Note that this method assumes that the distances in the eco.weight or eco.lagweight object are projected as great-circle distances (for example, using latlon = TRUE during weights construction

or UTM coordinates for elements passed with "con", or latlon set TRUE in this function for a coordinates element passed with XY).

Note also that when angular weights are constructed for XY coordinates, the output consists of a weights object with values bounded between 0 and 1, being 1 if the direction pointed by the vector V connecting the elements i, j in the matrix points in the same direction of the reference vector R (with an angle theta with the positive x axis), and 0 if V is perpendicular to R.

## Value

An object of class eco.weight with a bearing weights matrix

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples for the function eco.weight and eco.lagweight.

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## References

Rosenberg, M. 2000. The bearing correlogram: a new method of analyzing directional spatial autocorrelation. Geographical Analysis, 32: 267-278.

## Examples

```
## Not run:

data(eco3)

"circle" method

con <- eco.weight(eco3[["XY"]], method = "circle", d1 = 0, d2 = 500)
bearing_con <- eco.bearing(con, 90)

W_list <- eco.lagweight(eco[["XY"]])
bearing_W_list <- eco.bearing(W_list, 90)

## End(Not run)
```

---

`eco.cbind`*Combining the columns of ecogen object*

---

**Description**

Combining the columns of ecogen object

**Usage**

```
eco.cbind(eco1, eco2, ..., missing = c("0", "MEAN", "NA"))
```

**Arguments**

<code>eco1</code>	Object of class "ecogen".
<code>eco2</code>	Object of class "ecogen".
<code>...</code>	Other "ecogen" objects to combine and the specification of the data frames to combine. Can be any of the following(s): "P", "G", "E", "S", "C", or "ALL" (default). If a "G" data frame is provided, the program also generates the A slot coding the missing data as "0" in default option (see the argument "missing"). The XY slot is generated automatically if present.
<code>missing</code>	Missing data manipulation. It can take three values ("0", "NA" or "MEAN"- i.e, the mean frequency of the corresponding allele). Missing elements are coded as 0 in the default option.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:

data(eco.test)
eco.example <- eco.cbind(eco,eco,"ALL")
eco.example
eco.example2 <- eco.cbind(eco, eco,"P", "G", missing="NA")
eco.example2

## End(Not run)
```

---

eco.clear	<i>Clearing the working environment, maintaining only the specified objects</i>
-----------	---

---

**Description**

This function removes all the elements of the working environment, with the exception of those included in the argument of the function. Hidden elements can also be removed by setting all = TRUE.

**Usage**

```
eco.clear(..., all = FALSE)
```

**Arguments**

...	Objects to retain.
all	Remove also hidden elements? Default FALSE.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:  
  
data(eco.test)  
ls()  
eco.clear(eco)  
ls()  
  
## End(Not run)
```

---

eco.convert	<i>Conversion utility for genetic data</i>
-------------	--

---

**Description**

This function interconverts genetic data among matrix format (one locus or one allele per column) and list format (one locus or one allele per column).

**Usage**

```
eco.convert(X, input = c("matrix", "alleles.matrix", "list", "alleles.list"),
  output = c("matrix", "alleles.matrix", "list", "alleles.list"),
  ncod = NULL, ploidy = 2, sep.in, sep.out, chk.names = TRUE,
  chk.plocod = TRUE)
```

**Arguments**

X	Input data.
input	Input data format.
output	Output data format.
ncod	Number of digits coding each allele.
ploidy	Ploidy of the data.
sep.in	Character separating alleles in the input data if present.
sep.out	Character separating alleles in the output data. Default option do not separate alleles.
chk.names	Default TRUE. The function makes checks of individuals and loci names during conversion.
chk.plocod	Default TRUE. The function checks coherence in/between ploidy and number of digits coding alleles for loci data during conversion.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:

data(eco3)

# One allele per column
loc2al <- eco.convert(eco3[["G"]], "matrix", "alleles.matrix", ploidy = 2)
loc2al

# Inverse operation (collapse alleles into locus)
al2loc <- eco.convert(loc2al, "alleles.matrix", "matrix", ploidy = 2)
al2loc

# Separating alleles with a character string
loc2loc <- eco.convert(eco3[["G"]], "matrix", "matrix", ploidy = 2, sep.out = "/")
loc2loc

# Inverse operation (removing separator)
loc2loc.nosep <- eco.convert(loc2loc, "matrix", "matrix", ploidy = 2, sep.in = "/", sep.out = "")
loc2loc.nosep
```

```

# Locus to list
loc2list <- eco.convert(eco3[["G"]], "matrix", "list", ploidy = 2)
loc2list

# Locus to allele list
al2list <- eco.convert(eco3[["G"]], "matrix", "alleles.list", ploidy = 2)
al2list

# The inverse operations are also defined. All the formats are interconvertible.
# Locus operations have defined a within operation (matrix to matrix, list to list),
# with the purpose of put/remove separators between alleles. The program accepts any ploidy level.

## End(Not run)

```

---

eco.cormantel	<i>Mantel and partial Mantel correlograms, omnidirectional and directional</i>
---------------	--

---

## Description

This program computes a Mantel correlogram for the data M, or a partial Mantel correlogram for the data M conditioned on MC, with P-values or bootstrap confidence intervals.

## Usage

```

eco.cormantel(M, XY, MC = NULL, int = NULL, smin = 0, smax = NULL,
  nclass = NULL, seqvec = NULL, size = NULL, bin = c("sturges", "FD"),
  nsim = 99, classM = c("dist", "simil"), method = c("pearson",
  "spearman", "kendall"), test = c("permutation", "bootstrap"),
  alternative = c("auto", "two.sided", "greater", "less"), adjust = "holm",
  sequential = TRUE, latlon = FALSE, angle = NULL, as.deg = TRUE, ...)

```

## Arguments

M	Distance or similarity matrix.
XY	Data frame or matrix with individual's positions (projected coordinates).
MC	Distance or similarity matrix (optional).
int	Distance interval in the units of XY.
smin	Minimum class distance in the units of XY.
smax	Maximum class distance in the units of XY.
nclass	Number of classes.
seqvec	Vector with breaks in the units of XY.
size	Number of individuals per class.

bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
nsim	Number of Monte-Carlo simulations.
classM	Are M and MC distance or similarity matrices? Default option is classM = "dist" (distance). For similarity, classM = "simil". An incorrect option selected will generate an inverted plot.
method	Correlation method used for the construction of the statistic ("pearson", "spearman" or "kendall"). Kendall's tau computation is slow.
test	If test = "bootstrap", the program generates a bootstrap resampling and the associated confidence intervals. If test = "permutation" (default) a permutation test is made and the P-values are computed.
alternative	The alternative hypothesis. If "auto" is selected (default) the program determines the alternative hypothesis. Other options are: "two.sided", "greater" and "less".
adjust	Correction method of P-values for multiple tests, passed to <a href="#">p.adjust</a> . Default is "holm".
sequential	Should be performed a Holm-Bonferroni (Legendre and Legendre, 2012) adjustment of P-values? Default TRUE.
latlon	Are the coordinates in decimal degrees format? Default FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .
angle	for computation of bearing correlogram (angle between 0 and 180). Default NULL (omnidirectional).
as.deg	in case of bearing correlograms for multiple angles, generate an output for each lag in function of the angle? Default TRUE.
...	Additional arguments passed to <a href="#">cor</a> .

### Value

The program returns an object of class "eco.correlog" with the following slots:

- > OUT analysis output
- > IN input data of the analysis
- > BEAKS breaks
- > CARDINAL number of elements in each class
- > NAMES variables names
- > METHOD analysis method
- > DISTMETHOD method used in the construction of breaks
- > TEST test method used (bootstrap, permutation)
- > NSIM number of simulations
- > PADJUST P-values adjust method for permutation tests

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See `help("EcoGenetics accessors")` and the Examples section below.



**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

- Freedman D., and P. Diaconis. 1981. On the histogram as a density estimator: L<sup>2</sup> theory. *Probability theory and related fields*, 57: 453-476.
- Legendre P., and L. Legendre. 2012. *Numerical ecology*. Third English edition. Elsevier Science, Amsterdam, Netherlands.
- Oden N., and R. Sokal. 1986. Directional autocorrelation: an extension of spatial correlograms to two dimensions. *Systematic Zoology*, 35:608-617
- Rosenberg, M. 2000. The bearing correlogram: a new method of analyzing directional spatial autocorrelation. *Geographical Analysis*, 32: 267-278.
- Sokal R. 1986. Spatial data analysis and historical processes. In: E. Diday, Y. Escoufier, L. Lebart, J. Pages, Y. Schektman, and R. Tomassone, editors. *Data analysis and informatics, IV*. North-Holland, Amsterdam, The Netherlands, pp. 29-43.
- Sturges H. 1926. The choice of a class interval. *Journal of the American Statistical Association*, 21: 65-66.

**Examples**

```
## Not run:

data(eco.test)
require(ggplot2)

#####
## Omnidirectional correlogram
#####

corm <- eco.cormantel(M = dist(eco[["P"]]), size=1000,smax=7, XY = eco[["XY"]],
  nsim = 99)
eco.plotCorrelog(corm)

corm <- eco.cormantel(M = dist(eco[["P"]]), size=1000,smax=7, XY = eco[["XY"]],
  nsim = 99, test = "bootstrap")
eco.plotCorrelog(corm)

#####
## A directional approach based in bearing correlograms
#####

corm_b <- eco.cormantel(M = dist(eco[["P"]]), size=1000,smax=7, XY = eco[["XY"]],
  nsim = 99, angle = seq(0, 170, 10))

# use eco.plotCorrelogB for this object
eco.plotCorrelogB(corm_b)
```

```

# plot for the first distance class,
# use a number between 1 and the number of classes to select the corresponding class
eco.plotCorrelogB(moran_b, interactivePlot = FALSE, var = 1)

# partial Mantel correlogram
corm <- eco.cormantel(M = dist(eco[["P"]]), MC = dist(eco[["E"]]),
size=1000, smax=7, XY = eco[["XY"]], nsim = 99)
eco.plotCorrelog(corm)

# standard correlogram plots support the use of ggplot2 syntax
require(ggplot2)
mantelplot <- eco.plotCorrelog(corm, interactivePlot = FALSE)
mantelplot <- mantelplot + theme_bw() + theme(legend.position="none")
mantelplot

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.OUT(corm)      # slot OUT
ecoslot.BREAKS(corm)   # slot BREAKS

## End(Not run)

```

---

eco.correlog	<i>Moran's I, Geary's C and bivariate Moran's I correlograms, omnidirectional and directional</i>
--------------	---

---

## Description

This program computes Moran's, Geary's and bivariate Moran's correlograms, for single or multiple variables, with P-values or bootstrap confidence intervals. Correlograms can be omnidirectional or directional, the latter based in the bearing method (Rosenberg, 2000). The program allows high flexibility for the construction of intervals. For detailed information about the range partition methods see [eco.lagweight](#)

## Usage

```

eco.correlog(Z, XY, Y = NULL, int = NULL, smin = 0, smax = NULL,
nclass = NULL, size = NULL, seqvec = NULL, method = c("I", "C", "CC"),
nsim = 99, test = c("permutation", "bootstrap"), alpha = 0.05,
alternative = c("auto", "two.sided", "greater", "less"), adjust = "holm",
sequential = TRUE, include.zero = TRUE, cumulative = FALSE,

```

```
bin = c("sturges", "FD"), row.sd = FALSE, latlon = FALSE,
angle = NULL, as.deg = TRUE)
```

### Arguments

Z	Vector, matrix or data frame with variable/s (in matrix or data frame formats, variables in columns).
XY	Data frame or matrix with individual's positions (projected coordinates).
Y	Vector with the second variable for Mantel's Ixy cross-correlograms. If Z has multiple variables, the program will compute the cross-correlograms for each with Y.
int	Distance interval in the units of XY.
smin	Minimum class distance in the units of XY.
smax	Maximum class distance in the units of XY.
nclass	Number of classes.
size	Number of individuals per class.
seqvec	Vector with breaks in the units of XY.
method	Correlogram method. Could be I for Moran's I, C for Geary's C and CC for Bivariate Moran's Ixy. If method = "CC", the program computes for the first interval (d = 0) the corresponding P-value and CI with <a href="#">cor.test</a> .
nsim	Number of Monte-Carlo simulations.
test	If test = "bootstrap", the program generates a bootstrap resampling and the associated confidence intervals of the null hypothesis. If test = "permutation" (default) a permutation test is made and the P-values are computed.
alpha	Value for alpha. Default alpha = 0.05.
alternative	The alternative hypothesis. If "auto" is selected (default) the program determines the alternative hypothesis. Other options are: "two.sided", "greater" and "less".
adjust	P-values correction method for multiple tests passed to <a href="#">p.adjust</a> . Default is "holm".
sequential	Should be performed a Holm-Bonferroni (Legendre and Legendre, 2012) adjustment of P-values? Default TRUE.
include.zero	Should be included the distance = 0 in cross correlograms (i.e., the intra-individual correlation)? Default TRUE.
cummulative	Should be constructed a cumulative correlogram?.
bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
row.sd	Logical. Should be row standardized the matrix? Default FALSE (binary weights).
latlon	Are the coordinates in decimal degrees format? Default FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

angle	for computation of bearing correlogram (angle between 0 and 180). Default NULL (omnidirectional).
as.deg	in case of bearing correlograms for multiple angles, generate an output for each lag in function of the angle? Default TRUE.

### Value

The program returns an object of class "eco.correlog" with the following slots:

- > OUT analysis output
- > IN analysis input data
- > BEAKS breaks
- > CARDINAL number of elements in each class
- > NAMES variables names
- > METHOD analysis method
- > DISTMETHOD method used in the construction of breaks
- > TEST test method used (bootstrap, permutation)
- > NSIM number of simulations
- > PADJUST P-values adjust method for permutation tests

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

- Freedman D., and P. Diaconis. 1981. On the histogram as a density estimator: L<sup>2</sup> theory. *Probability theory and related fields*, 57: 453-476.
- Geary R. 1954. The contiguity ratio and statistical mapping. *The incorporated statistician*, 115-146.
- Legendre P., and L. Legendre. 2012. *Numerical ecology*. Third English edition. Elsevier Science, Amsterdam, Netherlands
- Moran P. 1950. Notes on continuous stochastic phenomena. *Biometrika*, 17-23.
- Reich R., R. Czaplewski and W. Bechtold. 1994. Spatial cross-correlation of undisturbed, natural shortleaf pine stands in northern Georgia. *Environmental and Ecological Statistics*, 1: 201-217.
- Rosenberg, M. 2000. The bearing correlogram: a new method of analyzing directional spatial autocorrelation. *Geographical Analysis*, 32: 267-278.
- Sokal R. and N. Oden 1978. Spatial autocorrelation in biology: 1. Methodology. *Biological journal of the Linnean Society*, 10: 199-228.
- Sokal R. and N. Oden. 1978. Spatial autocorrelation in biology. 2. Some biological implications and four applications of evolutionary and ecological interest. *Biological Journal of the Linnean Society*, 10: 229-49.

Sokal R. 1979. Ecological parameters inferred from spatial correlograms. In: G. Patil and M. Rosenzweig, editors. Contemporary Quantitative Ecology and related Ecometrics. International Co-operative Publishing House: Fairland, MD, pp. 167-96.

Sturges H. 1926. The choice of a class interval. Journal of the American Statistical Association, 21: 65-66.

## Examples

```
## Not run:

data(eco.test)
require(ggplot2)

#####
# Moran's I correlogram
#####

## single test with phenotypic traits
moran <- eco.correlog(Z=eco[["P"]][,1], XY = eco[["XY"]],
method = "I", smax=10, size=1000)

# interactive plot via plotly
eco.plotCorrelog(moran)

# standard plot via ggplot2
eco.plotCorrelog(moran, interactivePlot = FALSE)

#-----
## A directional approach based in bearing correlograms
#-----

moran_b <- eco.correlog(Z=eco[["P"]][,1], XY = eco[["XY"]],
method = "I", smax = 10, size = 1000, angle = seq(0, 175, 5))

# use eco.plotCorrelogB for this object
eco.plotCorrelogB(moran_b)

# plot for the first distance class,
# use a number between 1 and the number of classes to select the corresponding class
eco.plotCorrelogB(moran_b, var = 1)

#-----
## Multivariable correlograms
#-----

## multiple tests with phenotypic traits
moran2 <- eco.correlog(Z=eco[["P"]], XY = eco[["XY"]],
method = "I", smax=10, size=1000)
```

```

eco.plotCorrelog(moran2, var ="P2") ## single plots
eco.plotCorrelog(moran2, var ="P3") ## single plots

## Multivariable interactive plot with mean correlogram
## and jackknifed confidence intervals.

graf <- eco.plotCorrelog(moran2, meanplot = TRUE)

# Only mean
graf$mean.correlog

# Mean and variables
graf$multi.correlog

# Information
- correlogram data for individual variables
- manhattan distance matrix
- mean correlogram data
- method used for analysis
- names and numbers (column in data frame) of significant variables

graf$data

# plot only alleles
graf <- eco.plotCorrelog(moran2, meanplot = FALSE)
graf

# Both plots can also be constructed using ggplot2

gg_graf <- eco.plotCorrelog(moran2, meanplot = TRUE, interactivePlot = FALSE)
gg_graf[[1]]
gg_graf[[2]]

gg_graf <- eco.plotCorrelog(moran2, meanplot = FALSE, interactivePlot = FALSE)
gg_graf

# standard ggplot2 correlograms support the use of ggplot2 syntax
require(ggplot2)
moranplot <- eco.plotCorrelog(moran2, var ="P3", interactivePlot = FALSE)
moranplot <- moranplot + theme_bw() + theme(legend.position="none")
moranplot

moranplot2 <- gg_graf[[2]] + theme_bw() + theme(legend.position="none")
moranplot2

#-----
Analyzing genetic data

```

```

#-----

# single test with genotypic traits

# eco[["A"]] is a matrix with the genetic data of "eco"
# as frequencies for each allele in each individual. Each allele
# can be analyzed as single traits.

head(eco[["A"]])      # head of the matrix

# analyzing allele 1
moran <- eco.correlog(Z=[["A"]][,1], XY = eco[["XY"]], method = "I",
smax=10, size=1000)
eco.plotCorrelog(moran)

# multiple tests with genotypic traits.
# nsim is set to 10 only for speed in the example
moran2 <- eco.correlog(Z = eco[["A"]], XY = eco[["XY"]],
method = "I",smax=10, size=1000, nsim=99)

## multiple plot with mean
## correlogram and jackknifed
## confidence intervals.

graf <- eco.plotCorrelog(moran2, meanplot = TRUE)

## the same example, but with nsim = 99.
moran3 <- eco.correlog(Z = eco[["A"]], XY = eco[["XY"]], method = "I",
smax=10, size=1000, nsim=99)

## plot for alleles with at least one significant value after
## Bonferroni-Holm sequential P correction
## (set adjust "none" for no family-wise
## P correction in "eco.correlog")

eco.plotCorrelog(moran3, meanplot = TRUE, significant.M = TRUE)

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.OUT(moran)      # slot OUT
ecoslot.BREAKS(moran)  # slot BREAKS

#-----#

#####
# Geary's C correlogram

```

```
#####

geary <- eco.correlog(Z = eco[["P"]][,1], XY = eco[["XY"]], method = "C",
smax=10, size=1000)
# Interactive plot
eco.plotCorrelog(geary)
# ggplot2 plot
eco.plotCorrelog(geary, interactivePlot = FALSE)

#-----#

#####
# Bivariate Moran's Ixy
#####

cross <- eco.correlog(Z=eco[["P"]][,1], XY = eco[["XY"]], Y = eco[["P"]][, 1],
method = "CC", int= 2, smax=15)
# Interactive plot
eco.plotCorrelog(cross)
# ggplot2 plot
eco.plotCorrelog(cross, interactivePlot = FALSE)

## End(Not run)
```

---

eco.detrend

*Detrending spatial data with polynomial interpolation*


---

## Description

This program performs a Trend Surface Analysis (Borcard et al. 2011, Legendre and Legendre 2012, Lichstein et al 2002) for the data Z and the given coordinates, projected or in decimal degrees format, in which case will be projected with [geoXY](#).

## Usage

```
eco.detrend(Z, XY, degree, center = TRUE, scale = FALSE, raw = FALSE,
latlon = FALSE)
```

## Arguments

Z	Data frame, matrix or vector with dependent variables.
XY	Data frame, matrix or vector with projected coordinates (X, XY or XYZ). For longitude-latitude data in decimal degrees format, use the option latlon = TRUE.
degree	Polynomial degree.
center	Should the data be centered? Default TRUE
scale	Should the data be scaled? Default FALSE



raw	Use raw and not orthogonal polynomials? Default FALSE
latlon	Are the coordinates in decimal degrees format? Default FALSE. If TRUE, the coordinates must be in a data.frame/matrix with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

### Value

An object of class "eco.detrend" with the following slots:

> POLY.DEG polynomial degree used in the analysis

> RES detrended data

> XY projected coordinates

> MODEL models selected with the Akaike criterion

> ANALYSIS object of class "eco.mlm" with the regression results for each variable

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below.

### References

Borcard D., F. Gillet, and P. Legendre. 2011. Numerical ecology with R. Springer Science & Business Media.

Legendre P., and L. Legendre. 2012. Numerical ecology. Third English edition. Elsevier Science, Amsterdam, Netherlands.

Lichstein J., T. Simons, S. Shriver, and K. Franzreb. 2002. Spatial autocorrelation and autoregressive models in ecology. Ecological monographs, 72: 445-463.

### Examples

```
## Not run:
```

```
data(eco2)
```

```
# original data
```

```
data1 <- matrix(eco2[["P"]][,1], 30, 30)
```

```
image(data1)
```

```
# original data + trend
```

```
data2 <- matrix(eco2[["P"]][,2], 30, 30)
```

```
image(data2)
```

```
# data detrending
```

```
data2.det <- eco.detrend(Z = eco2[["P"]][,2], XY = eco2[["XY"]], degree = 1)
```

```
#-----
```

```
# ACCESSORS USE EXAMPLE
```

```
#-----
```

```

# the slots are accesed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

data2.det <- ecoslot.RES(data2.det)      # detrended data in slot RES

data2.det <- matrix(data2.det[,1], 30, 30)
image(data2.det)

## End(Not run)

```

---

eco.forestplot      *Forestplot graphs*

---

### Description

This program generates a forest plot for the confidence interval of each individual of the input data (as row number) and the corresponding observed value of the statistic.

### Usage

```

eco.forestplot(input, xlabel = NULL, ylabel = NULL, titlelabel = NULL,
  legendlabel = NULL, interactivePlot = TRUE, ...)

## S4 method for signature 'eco.lsa'
eco.forestplot(input, rescaled = FALSE, xlabel, ylabel,
  titlelabel, legendlabel, interactivePlot = TRUE)

## S4 method for signature 'dataframeORMatrix'
eco.forestplot(input, xlabel, ylabel, titlelabel,
  legendlabel, interactivePlot = TRUE)

```

### Arguments

input	Matrix/data frame, with three columns in the following order: observed value, lower and upper values of the confidence interval.
xlabel	Optional label for x axis.
ylabel	Optional label for y axis.
titlelabel	Optional title label.
legendlabel	Optional legend label.
interactivePlot	Show an interactive plot via plotly? (default: TRUE)

```
...           Additional elements to the generic.  
rescaled      rescale values to [-1, 1] range?
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:  
  
require(ggplot2)  
# simulated confidence intervals for the null hypothesis of a variable "a"  
set.seed(8)  
a<-runif(10, -2, 2)  
infer <- runif(10, -1, 1)  
super <- runif(10, -1, 1)  
infer2 <- pmin(infer, super)  
super2 <- pmax(infer, super)  
data <- data.frame(a, infer2, super2)  
forest <- eco.forestplot(data)  
forest  
  
# the forestplot method support the use of ggplot2 syntax with ggplot2 graphs  
forest <- eco.forestplot(data, interactivePlot = FALSE)  
forest <- forest + theme_bw() + theme(legend.position="none")  
forest  
  
## End(Not run)
```

---

eco.format

*Format tool for genetic data*

---

**Description**

Format tool for genetic data

**Usage**

```
eco.format(data, ncod = NULL, nout = 3, ploidy = 2, sep.in, sep.out,  
  fill.mode = c("last", "first", "none"), recode = c("none", "all",  
  "column"), show.codes = FALSE)
```

**Arguments**

data	Genetic data frame.
ncod	Number of digits coding each allele in the input file.
nout	Number of digits in the output.
ploidy	Ploidy of the data.
sep.in	Character separating alleles in the input data if present.
sep.out	Character separating alleles in the output data. Default
fill.mode	Add zeros at the beginning ("first") or the end ("last") of each allele. Default = "last".
recode	Recode mode: "none" for no recoding (default), "all" for recoding the data considering all the individuals values at once (e.g., protein data), or "column" for recoding the values by column (e.g., microsatellite data).
show.codes	May we returned tables with the equivalence between the old and new codes when recode = "all" or recode = "column"?

**Details**

The function can format data with different ploidy levels. It allows to: - add/remove zeros at the beginning/end of each allele - separate alleles with a character - divide alleles into columns - bind alleles from separate columns - transform character data into numeric data

"NA" is considered special character (not available data).

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:

data(eco.test)

# Adding zeros

example <- as.matrix(genotype[1:10,])
mode(example) <- "character"
# example data
example
recoded <- eco.format(example, ncod = 1, ploidy = 2, nout = 3)
# recoded data
recoded

# Tetraploid data, separating alleles with a "/"
tetrap <- as.matrix(example)
# simulated tetraploid example data
tetrap <- matrix(paste(example,example, sep = "/"), ncol = ncol(example))
```

```

recoded <- eco.format(tetrap, ncod = 1, ploidy = 4, sep.out = "/")
# recoded data
recoded

# Example with a single character
ex <- c("A","T","G","C")
ex <- sample(ex, 100, rep= T)
ex <- matrix(ex, 10, 10)
colnames(ex) <- letters[1:10]
rownames(ex) <- LETTERS[1:10]
# example data
ex
recoded <- eco.format(ex, ploidy = 1, nout = 1, recode = "all", show.codes = TRUE)
# recoded data
recoded

# Example with two strings per cell and missing values:
ex <- c("Ala", "Asx", "Cys", "Asp", "Glu", "Phe", "Gly", "His", "Ile",
"Lys", "Leu", "Met", "Asn", "Pro", "Gln", "Arg", "Ser", "Thr",
"Val", "Trp")
ex1 <- sample(ex, 100, rep= T)
ex2 <- sample(ex, 100, rep= T)
ex3 <- paste(ex1, ex2, sep="")
missing.ex3 <- sample(1:100, 20)
ex3[missing.ex3] <-NA
ex4 <- matrix(ex3, 10, 10)
colnames(ex4) <- letters[1:10]
rownames(ex4) <- LETTERS[1:10]
# example data
ex4
recoded <- eco.format(ex4, ncod = 3, ploidy = 2,
nout = 2, recode = "column")
# recoded data
recoded

# Example with a vector, following the latter example:
ex1 <- as.data.frame(ex1)
# example data
ex1
recoded <- eco.format(ex1, ploidy = 1, recode = "all")
# recoded data
recoded

## End(Not run)

```

**Description**

The function creates an explicit formula that can be used as input to parse ecogen objects into other methods. For this purpose, each element in the formula is substituted with an explicit name of columns in ecogen objects if:

- The name in formula corresponds to the name of a column in the data frames of the object, or -
- The name in formula is surrounded by U() and corresponds to a slot name. Complete data frames (as P, E, etc) or subsets (as U[, 1:5]) can be passed with this method.

In other situations, text is not replaced.

**Usage**

```
eco.formula(eco, formula, out.mode = c("formula", "expression"),
  expand.tables = "+")
```

**Arguments**

eco	Object of class "ecogen".
formula	formula with names of columns from the slots XY, P, G, A, E, or C
out.mode	Output results explicit formula (default) or expression.
expand.tables	method for tables coercion. Default is "+"

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:
require(vegan)
data(eco.test)

# Note that in this example "Condition" is not replaced;
# the function Condition has a special meaning in rda,
# indicating conditioning variables; in eco.formula it is only text.

form <- eco.formula(eco, P1 + P2 + P3 + U(A) ~ E1 + E2 + Condition(X+Y))
rda(form)

form2 <- eco.formula(eco, P1 + P2 + P3 + U(A) ~ E1 + E2 + X + Y)
lm(form2)

# parsing with magrittr
eco.formula(eco, P1 + P2 + P3 + E (A) ~ U(E) + Condition(X+Y)) %>% rda

## End(Not run)
```

**Description**

Univariate and multivariable global spatial analysis. This program computes Moran's I, Geary's C, Bivariate Moran's I or Join-count statistics with P-values.

The program allows the analysis of a single variable or multiple variables. In the last case, the variables must be in columns and the individuals in rows.

For join-count analysis, a ploidy argument must be supplied. The data is then ordered with the function `ae.sort`. This step is required in the analysis of genotypes. An individual with the alleles A and B, coded as AB, is identical to other coded as BA. The ordination step ensures that both are considered in a single category. For the analysis of frequencies of single alleles, the input is count data (ploidy-times the frequency, as provided by the slot A of an ecogen object: the count data A' can be obtained as `A' <- ploidy * A`), using the function with the arguments `ploidy = 1`.

**Usage**

```
eco.gsa(Z, con, Y = NULL, method = c("I", "C", "CC", "JC"), nsim = 99,
        alternative = c("auto", "two.sided", "greater", "less"), ploidy = 1,
        adjust = "fdr", plotit = TRUE)
```

**Arguments**

Z	Vector with a variable, or matrix/data frame with variables in columns.
con	An object of class <code>eco.weight</code> obtained with the function <code>eco.weight</code> , a listw object or a matrix, giving the spatial weights for the analysis. If "con" is a matrix, an attribute "xy" including the projected coordinates is required.
Y	Vector with the second variable for Moran's Ixy. If Z has multiple variables, the program will compute the coefficient for each with Y.
method	Method of analysis: "I" for Moran's I, "C" for Geary's C, "CC" for the Bivariate Moran's or "JC" for Join-count.
nsim	Number of Monte-Carlo simulations.
alternative	The alternative hypothesis. If "auto" is selected (default) the program determines the alternative hypothesis. Other options are: "two.sided", "greater" and "less".
ploidy	For join count analysis: number of elements for the values of the vector passed, given value: for example, if <code>ploidy=1</code> , "13" and "31" are considered a same level ("31" is sorted by the program as "13"); if <code>ploidy = 1</code> , "13" and "31" represent two different levels.
adjust	Correction method of P-values for multiple tests, passed to <code>p.adjust</code> . Default is "fdr".
plotit	should be generated a plot for univariate results?

**Value**

The program returns an object of class "eco.gsa" with the following slots:

- > METHOD method used in the analysis
- > OBS observed value when a single variable is tested
- > EXP expected value when a single variable is tested
- > PVAL P-value when a single variable is tested
- > ALTER alternative hypothesis when a single variable is tested
- > NSIM number of simulations
- > MULTI table with observed and expected values, P-values and alternative hypotheses when multiple variables are tested

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

- Geary R. 1954. The contiguity ratio and statistical mapping. *The incorporated statistician*, 115-146.
- Moran P. 1950. Notes on continuous stochastic phenomena. *Biometrika*, 17-23.
- Reich R., R. Czaplewski and W. Bechtold. 1994. Spatial cross-correlation of undisturbed, natural shortleaf pine stands in northern Georgia. *Environmental and Ecological Statistics*, 1: 201-217.
- Sokal R. and N. Oden 1978. Spatial autocorrelation in biology: 1. Methodology. *Biological journal of the Linnean Society*, 10: 199-228.
- Sokal R. and N. Oden. 1978. Spatial autocorrelation in biology. 2. Some biological implications and four applications of evolutionary and ecological interest. *Biological Journal of the Linnean Society*, 10: 229-49.

**Examples**

```
## Not run:

data(eco.test)

# Moran's I

### one test
con <- eco.weight(eco[["XY"]], method = "circle", d1 = 0, d2 = 2)
global <- eco.gsa(Z = eco[["P"]][, 1], con = con, method = "I", nsim = 200)
global

require(adegenet)
con2<-chooseCN(eco[["XY"]], type = 1, result.type = "listw", plot.nb = FALSE)
```



```

global <- eco.gsa(Z = eco[["P"]][, 1], con = con2, method = "I", nsim = 200)
global

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

# observed value
ecoslot.OBS(global)

# p-value
ecoslot.PVAL(global)

#-----
# multiple tests
#-----
data(eco3)
con <- eco.weight(eco3[["XY"]], method = "circle", d1 = 0, d2 = 500)
global <- eco.gsa(Z = eco3[["P"]], con = con, method = "I", nsim = 200)
global
# Plot method for multivariable eco.gsa objects:
eco.plotGlobal(global)

#-----
# accessor use in multiple tests
#-----

ecoslot.MULTI(global)

#-----

# Gearys's C

con <- eco.weight(eco[["XY"]], method = "circle", d1 = 0, d2 = 2)
global.C <- eco.gsa(Z = eco[["P"]][, 1], con = con, method = "C", nsim = 200)
global.C

#-----

# Bivariate's Moran's Ixy

con <- eco.weight(eco[["XY"]], method = "circle", d1 = 0, d2 = 2)
global.Ixy <- eco.gsa(Z = eco[["P"]][, 1], Y = eco[["E"]][, 1],
con = con, method = "CC", nsim = 200)
global.Ixy

#-----

```

```

# Join-count

## using the allelic frequency matrix of an ecogen object.
## The data is diploid. Frequencies are transformed into counts
## as ploidy * frequency_matrix:

Z = 2* eco[["A"]]

jc <- eco.gsa(Z[, 1], con = con, method = "JC")
eco.plotGlobal(jc)

# multiple tests
# using the first ten alleles of the matrix
global.JC <- eco.gsa(Z[, 1:10], ncod = 1, con = con, method = "JC", nsim = 99)
global.JC
# plot method for multivariable join-count
eco.plotGlobal(global.JC)

# counting joins between genotypes in the first locus the G matrix:
global.JC <- eco.gsa(Z = eco[["G"]][, 1], ploidy = 2, con = con, method = "JC", nsim = 99)
global.JC
eco.plotGlobal(global.JC)

## End(Not run)

```

---

eco.kin.loiselle

*Obtention of the multilocus Loiselle's Fij matrix*


---

### Description

Obtention of the multilocus Loiselle's Fij matrix

### Usage

```
eco.kin.loiselle(eco)
```

### Arguments

eco                    Object of class ecogen.

### References

Kalisz, S., J. Nason, F.M. Handazawa, and S. Tonsor. 2001. Spatial population genetic structure in *Trillium grandiflorum*: the roles of dispersal, mating, history, and selection. *Evolution* 55: 1560-1568.

Loiselle, B., V. Sork, J. Nason, and C. Graham. 1995. Spatial genetic structure of a tropical understory shrub, *Psychotria officinalis* (Rubiaceae). *American Journal of Botany* 1420-1425.

**Examples**

```
## Not run:

data(eco.test)
loiselle <- eco.kin.loiselle(eco)
loiselle[1:5, 1:5]

## End(Not run)
```

---

eco.lagweight	<i>Obtention of a list of spatial weights for classes defined by inter-individual distances or nearest-neighbors</i>
---------------	--

---

**Description**

This program returns a list of weights matrices (binary or row-standardized), one for each spatial class. For a given maximum and minimum inter-individual distance (IID), the data can be partitioned in different ways. The program set as default the highest IID as the maximum, and the lowest as the minimum. These values can be changed with "smax" and "smin", respectively. Intervals may be generated with the parameters "int" (which divides the range each int distance units), "nclass" (which divides the range in n-classes) and "size" (a fixed size of pairs included in each class). When a partition argument is not given (int, nclass or size) the program determines the number of classes using the Sturge's rule (default) or the Freedman- Diaconis method. Two additional methods can be used: a list with nearest-neighbors matrices, from 1 to k nearest-neighbors, may be generated with the argument "kmax". A custom vector with breaks may be provided by the user with the argument "seqvec". See the examples.

**Usage**

```
eco.lagweight(XY, int = NULL, smin = 0, smax = NULL, kmax = NULL,
  nclass = NULL, seqvec = NULL, size = NULL, bin = c("sturges", "FD"),
  cumulative = FALSE, row.sd = FALSE, self = FALSE, latlon = FALSE)
```

**Arguments**

XY	Matrix/data frame with projected coordinates.
int	Distance interval in the units of XY.
smin	Minimum class distance in the units of XY.
smax	Maximum class distance in the units of XY.
kmax	Number of nearest-neighbors.
nclass	Number of classes.
seqvec	Vector with breaks in the units of XY.

size	Number of individuals per class.
bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
cummulative	Logical. Should be created matrices considering cummulative distances instead of discrete classes? Default FALSE.
row.sd	Logical. Should be row standardized each matrix? Default FALSE (binary weights).
self	Logical. Should be included a first matrix with ones in the diagonal and zeros zeros in the other cells? Default FALSE.
latlon	Are the coordinates in decimal degrees format? Defalut FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

### Value

The program returns an object of class "eco.lagweight" with the following slots:

- > PAR parameters used for the construction of breaks
- > PAR.VAL values of the parameters used for the construction of breaks
- > ROW.SD row standardization (logical)
- > SELF data self-included (logical)
- > W weights list
- > XY coordinates
- > MEAN mean class distances
- > LOGMEAN mean of the class distances logarithm
- > CARDINAL number of elements in each class
- > BREAKS breaks
- > METHOD breaks construction method

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

- Freedman D., and P. Diaconis. 1981. On the histogram as a density estimator: L 2 theory. Probability theory and related fields, 57: 453-476.
- Sturges H. 1926. The choice of a class interval. Journal of the American Statistical Association, 21: 65-66.

**Examples**

```

## Not run:
data(eco.test)

# method sturges-smax: in this case, the program generates
# classes using the Sturge's rule.
# As smax and smin are undefined, the program uses the default
# options (smin = 0, and smax = maximum inter-individual distance)
classlist <- eco.lagweight(eco[["XY"]])
classlist

# method sturges-smax: idem, but smax = 16
classlist <- eco.lagweight(eco[["XY"]], smax=16)

## using smax <16 in this case generates empty classes,
## which is not allowed

# method sturges-smax: idem, but smin = 3
classlist <- eco.lagweight(eco[["XY"]], smin = 3, smax = 15)

# method sturges-smax: complete range,
# and cumulative = TRUE (instead of using
# lower and upper limits for each class, only the upper is used in turn)
classlist <- eco.lagweight(eco[["XY"]], cumulative = TRUE)

# method n.classes-smax: complete range partitioned in 4 classes
classlist <- eco.lagweight(eco[["XY"]], nclass = 4)

# method n.classes-smax: idem, but smax = 15
classlist <- eco.lagweight(eco[["XY"]], nclass = 4, smax = 15)

# method int-smax: the complete range partitioned each <int> units
# of inter-individual distance
classlist <- eco.lagweight(eco[["XY"]], int = 2)

# method int-smax: idem, but smax = 15 and smin = 3
classlist <- eco.lagweight(eco[["XY"]], int = 2, smin = 3, smax = 15)

# method equal.size: n individuals in each class,
# partitioning the complete range.
classlist <- eco.lagweight(eco[["XY"]], size = 1000)

## In the latter example, as an inter-individual distance
## appear more than one time (different individuals pairs,
## identical distances), with a size <700 the limits
## of some classes cannot be defined, and this is not allowed

# method equal.size: n individuals in each class,
# but smax = 15
classlist <- eco.lagweight(eco[["XY"]], size = 1000, smax = 15)

# method kmax: sequence from k = 1 to k = n, in this case, n = 3

```

```

classlist <- eco.lagweight(eco[["XY"]], kmax = 3)

# method kmax: idem, but elements self-included
# (i.e., the pairs i-i, for all individuals i, are included)
classlist <- eco.lagweight(eco[["XY"]], kmax = 3, self = TRUE)

# method seqvec: a vector with the breaks is used
vec <- seq(0, 10, 2)
classlist <- eco.lagweight(eco[["XY"]], seqvec = vec)

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.BREAKS(classlist) # information about breaks. It includes the upper and lower limits

## End(Not run)

```

---

eco.listw2ew

*Conversion from listw to ecoweight*


---

### Description

Conversion from listw to ecoweight

### Usage

```
eco.listw2ew(X)
```

### Arguments

X                    A listw object

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

eco.lmtree

*Fitting Multiple Linear Regression models by stepwise AIC selection and Multiple Classification and Regression Trees via party*

## Description

This program fits for each dependent variable, a Multiple Linear Regression model calling the function `step` for choosing the best model by AIC criterion, or a Multiple Classification and Regression Trees model, using the package `party`. The summary of the model returns information about the significance of the models, F-statistics and degrees of freedom, when is fitted a "mlm"; otherwise, when the model fitted is a "mctree", the summary returns the plots of those trees with significant splits.

## Usage

```
eco.lmtree(df1, df2, analysis = c("mlm", "mctree"), mod.class = "+",
           fact = NULL, ...)
```

## Arguments

<code>df1</code>	Data frame with dependent variables as columns.
<code>df2</code>	Data frame with independent variables as columns.
<code>analysis</code>	Class of analysis to perform. "mlm" for multiple linear regression analysis, or "mctree" for a multiple classification tree analysis.
<code>mod.class</code>	"+" for additive model, "*" for model with interaction, in both cases, these models will include all terms in the dependent data frame. If other model than these two is desired, it could be specified as a string with the names of those columns of the independent variable that should be used as terms. This string corresponds to the right side "x" of a formula $y \sim x$ (see examples).
<code>fact</code>	Optional factor for estimating the frequencies of individuals from different levels in each node, when the analysis performed is "mctree".
<code>...</code>	Further arguments passed to <code>lm</code> or <code>ctree</code>

## Value

When the analysis selected is "mlm", the output object has three main slots:

- > `MLM`: the results of the model
- > `SUMMARY.MLM` the summary for each variable returned by the `lm` function
- > `ANOVA.MLM` with the ANOVAs results.

When the analysis selected is "mctree", the output object has also three main slots:

- > `TREES`: Trees returned by the multiple `ctree` analysis.
- > `PREDICTIONS`: Predictions of the analysis.
- > `FREQUENCIES`: Number of individuals predicted in each node.

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

Hothorn T., K. Hornik, and A. Zeileis. 2006. Unbiased Recursive Partitioning: A Conditional Inference Framework. *Journal of Computational and Graphical Statistics*, 15: 651-674.

### Examples

```
## Not run:

data(eco.test)

# mlm additive model
mod <- eco.lmtree(df1 = eco[["P"]], df2 = eco[["E"]],
analysis = "mlm")
mod
summary(mod)

# mctree additive model
mod <- eco.lmtree(df1 = eco[["P"]], df2 = eco[["E"]],
analysis = "mctree", fact = eco[["S"]]$pop)

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

summary(mod)

ecoslot.FREQUENCIES(mod)      # slot FREQUENCIES

# frequency table with counts of individuals in populations x terminal nodes
tabfreq <- do.call(cbind, ecoslot.FREQUENCIES(mod))
namestab <- lapply(ecoslot.FREQUENCIES(mod), ncol)
namestab <- lapply(namestab, rep)
namestab <- rep(names(namestab), namestab)
colnames(tabfreq) <- namestab
tabfreq

# mlm custom model
```



```

mymod <- "E1+E2*E3"
mod <- eco.lmtree(df1 = eco[["P"]], df2 = eco[["E"]],
analysis = "mlm", mod.class = mymod)
summary(mod)

# mctree custom model
mod <- eco.lmtree(df1 = eco[["P"]], df2 = eco[["E"]],
analysis = "mctree", mod.class = mymod, fact = eco[["S"]]$pop)

summary(mod)

## End(Not run)

```

---

eco.lsa

*Local spatial analysis*


---

### Description

Univariate and multivariable local spatial analysis. This program computes Getis-Ord G and G\*, and LISA's (local Moran and local Geary) statistics for the data Z, with P-values or bootstrap confidence intervals.

### Usage

```

eco.lsa(var, con, method = c("G*", "G", "I", "C"), zerocon = NA,
nsim = 99, conditional = c("auto", "TRUE", "FALSE"),
test = c("permutation", "bootstrap"), alternative = c("auto", "two.sided",
"greater", "less"), adjust = "none", multi = c("matrix", "list"),
pop = NULL)

```

### Arguments

var	Vector, matrix or data frame for the analysis. Multiple variables in columns.
con	An object of class <code>eco.weight</code> obtained with the function <code>eco.weight</code> , a "listw" object, or a matrix, containing the weights for the analysis. If a matrix, an attribute "xy" with the projected coordinates is required.
method	Method of analysis: "G" for Getis-Ord G, "G*" for Getis-Ord G*, "I" for local Moran's I or "C" for local Geary's C.
zerocon	If <code>zerocon = 0</code> the program assigns the value 0 to those individuals with no connections; if <code>zerocon = NA</code> the program assigns NA. Default is NA.
nsim	Number of Monte-Carlo simulations.
conditional	Logical. Should be used a conditional randomization? (Anselin 1998, Sokal and Thomson 2006). The option "auto" sets <code>conditional = TRUE</code> for LISA methods and G, as suggested by Sokal (2008).

test	If test = "bootstrap", for each individual test, the program generates a bootstrap resampling and the associated confidence intervals of the null hypothesis. If test = "permutation" (default) a permutation test is made and the P-value is computed.
alternative	The alternative hypothesis for "permutation" test. If "auto" is selected (default) the program determines the alternative hypothesis in each individual test. Other options are: "two.sided", "greater" and "less".
adjust	Correction method of P-values for multiple tests, passed to <code>p.adjust</code> . Default is "none" (no correction).
multi	multiple output format results. "list" for object with a list of individual test for each variable, or "matrix" for results as matrices of multiples variables.
pop	numeric factor with the population of each individual. Optional for multiple tests with multi = "matrix".

### Value

For single test, the program returns an object of class "eco.lsa" with the following slots:

> OUT results - table with output results.

-> If test = "permutation": observed value of the statistic , null confidence interval and #rescaled observed value to [-1, 1] range, as in Sokal (2006)

-> If test = "bootstrap": observed and expected value of the statistic, alternative hypothesis, null confidence interval and rescaled observed value to [-1, 1] range, as in Sokal (2006)

> METHOD method (coefficient) used in the analysis

> TEST test method used (bootstrap, permutation)

> NSIM number of simulations

> PADJUST P-values adjust method for permutation tests

> COND conditional randomization (logical)

> XY input coordinates

For multiple test, if the parameter multi = "list", the program returns a list of eco.lsa objects (one element for each variable).

For multiple test, if the parameter multi = "matrix", the program returns an object of class "eco.multilsa" with the following slots:

> METHOD method used in the analysis

> TEST test method used (bootstrap, permutation)

> NSIM number of simulations

> PADJUST P-values adjust method for permutation tests

> COND conditional randomization (logical)

> XY input coordinates

> OBS observed value

> EXP expected value

> ALTER test alternative

> PVAL pvalue for permutation test

> LWR lower confidence interval bound of the null hypothesis  
 > UPPR upper confidence interval bound of the null hypothesis  
 > OBS.RES rescaled observed value to [-1, 1] range, as in Sokal (2006)

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

Anselin L. 1995. Local indicators of spatial association-LISA. *Geographical analysis*. 27: 93-115.  
 Getis A., and J. Ord. 1992. The analysis of spatial association by use of distance statistics. *Geographical analysis*, 24: 189-206.  
 Ord J., and A. Getis. 1995. Local spatial autocorrelation statistics: distributional issues and an application. *Geographical analysis*, 27: 286-306.  
 Sokal R., N. Oden and B. Thomson. 1998. Local spatial autocorrelation in a biological model. *Geographical Analysis*, 30: 331-354.  
 Sokal R. and B. Thomson. 2006. Population structure inferred by local spatial autocorrelation: an example from an Amerindian tribal population. *American journal of physical anthropology*, 129: 121-131.

### Examples

```
## Not run:

data(eco.test)

#-----#

#####
# LOCAL MORAN'S I
#####

DETAILED EXAMPLE

#-----#
# TESTING PHENOTYPIC DATA-
#-----#

set.seed(10)

# test for a single variable-----#
#computing weights
```

```

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE

# test for the first trait of the data frame P
localmoran <- eco.lsa(eco[["P"]][, 1], con, method = "I", nsim = 99)

# "rankplot" graph
eco.plotLocal(localmoran)

# test for several variables-----

# ordering the factor "pop" in increasing order and the object "eco"
# in relation to this ordered factor prior to the multivariate analysis.
# This step is important for "localplot" graphs.

eco <- eco[order(eco[["S"]][,1])]

#computing weights with the ordered object

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE

all.traits <- eco.lsa(eco[["P"]], con, method = "I", nsim = 99)

# Plot of the phenotypic spatial patterns

# "rasterplot" graph
eco.plotLocal(all.traits)

# in grey: non significant results (P > 0.05)
# set significant = FALSE for showing significant and no significant results
eco.plotLocal(all.traits, significant = FALSE)

# single plots using "rankplot" graphs
all.single.traits <- eco.lsa(eco[["P"]],con, method = "I", nsim = 99, multi="list")
eco.plotLocal(all.single.traits)

# removing legends for a better visualization
eco.plotLocal(all.single.traits, legend = FALSE)
# - individual plots support ggplot2 syntax (plot equivalent to the previous):
eco.plotLocal(all.single.traits) + ggplot2::theme(legend.position="none")

#-----
# TESTING GENOTYPIC DATA-
#-----

# eco[["A"]] is a matrix with the genetic data of "eco"
# as frequencies for each allele in each individual.

head(eco[["A"]])      # head of the matrix - 40 alleles

# ordering the factor "pop" in increasing order and the object "eco"

```

```

# in relation to this ordered factor prior to the multivariate analysis.
# This step is important for "localplot" graphs

data(eco.test) # for security this resets the data (unordered)

eco <- eco[order(eco[["S"]][,1])] # ordering

# computing weights with the ordered object

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE

# test for a single allele
localmoran.geno <- eco.lsa(eco[["A"]][, 32], con, method = "I", nsim = 99)

# test for several alleles - 40 alleles (it runs in less than 1 min
# for 99 simulations per allele; 999 simulations takes ~ 11 s per allele,
# less than 8 min in total.)
all.alleles <- eco.lsa(eco[["A"]], con, method = "I", nsim = 99)

# plot all alleles to get an overview of the spatial patterns
eco.plotLocal(all.alleles)

# in grey: non significant results (P > 0.05)
# set significant = FALSE for showing significant and no significant results
eco.plotLocal(all.alleles, significant = FALSE)

# counting individuals with P < 0.05 for each allele (5 * 225 /100 ~ 12 significant tests
# by random)
signif <- apply(ecoslot.PVAL(all.alleles), 2, function(x) sum (x < 0.05))

# filtering alleles, loci with > 12 significant individual tests

A.local <- eco[["A"]][, signif > 12]      #filtered matrix

all.alleles.f <- eco.lsa(eco[["A"]][, signif > 12] , con, method = "I", nsim = 99)

# Plot of the genotypic spatial patterns using "localplot" graphs

eco.plotLocal(all.alleles.f)

## using "rankplot" graphs

all.sf <- eco.lsa(A.local, 2, eco.lsa, con, method = "I", nsim = 99, multi = "list")
eco.plotLocal(all.sf, legend = FALSE)

#####
# GETIS-ORD'S G*
#####

```

```

con<- eco.weight(eco[["XY"]], method = "knearest", k = 4, self = TRUE) # self = TRUE for G*
getis.ak <- eco.lsa(eco[["P"]][, 1], con, method = "G*", nsim = 99, adjust = "none")
getis.ak

### to plot the results, the function "eco.lsa" calls "eco.rankplot"
### (see ?eco.rankplot) when test = "permutation" and "eco.forestplot" (see ?eco.forestplot)
### when test = "bootstrap"

p <- eco.plotLocal(getis.ak) # rankplot graph
p # points with colors of the color-scale:
# points with P < 0.05. Yellow points : points with P > 0.05
p <- eco.plotLocal(getis.ak, significant = FALSE)
p # all points have a color of the color-scale

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.OUT(getis.ak)

## bootstrap example
getis.akb <- eco.lsa(eco[["P"]][, 1], con, method = "G*", nsim = 99, test = "bootstrap")
p <- eco.plotLocal(getis.akb) # forestplot graph

p2 <- eco.plotLocal(getis.akb, interactivePlot = FALSE)
p2 + ggplot2::theme_bw() # the plot can be modified with ggplot2
# In this case, the background is modified (white color)

#-----#

#####
# GETIS-ORD'S G
#####

con <- eco.weight(eco[["XY"]], method = "knearest", k = 4)
# self = FALSE for G
getis <- eco.lsa(eco[["P"]][, 1], con, method = "G", nsim = 99)
eco.plotLocal(getis)

#-----#

#####
# LOCAL GEARY'S C
#####

con<- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
# row standardized weights = TRUE
localgeary <- eco.lsa(eco[["P"]][, 1], con, method = "C", nsim = 99, adjust = "none")
eco.plotLocal(localgeary)

```

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

---

 eco.malecot

*Global and local kinship analyses*


---

## Description

The program computes a global multilocus correlogram, or a local analysis, using a kinship matrix. When a kinship matrix is not given as input, the program computes the Loiselle's Fij (Kalisz et al., 2001; Loiselle et al., 1995). The program can compute a bearing correlogram (Rosenberg 2000, Born et al. 2012) for the obtention of a directional approach to the global test.

## Usage

```
eco.malecot(eco, method = c("global", "local"), kinmatrix = NULL,
  int = NULL, smin = 0, smax = NULL, nclass = NULL, kmax = NULL,
  seqvec = NULL, size = NULL, type = c("knearest", "radialdist"),
  cubic = TRUE, testclass.b = TRUE, testmantel.b = TRUE,
  jackknife = TRUE, cummulative = FALSE, normLocal = TRUE, nsim = 99,
  test = c("permutation", "bootstrap"), alternative = c("auto", "two.sided",
  "greater", "less"), sequential = TRUE, conditional = c("AUTO", "TRUE",
  "FALSE"), bin = c("sturges", "FD"), row.sd = FALSE, adjust = "holm",
  latlon = FALSE, angle = NULL)
```

## Arguments

eco	Object of class ecogen.
method	Analysis method: "global" or "local".
kinmatrix	Alternative kinship matrix. The program computes the Loiselle's kinship matrix (codominant data) with the genetic information of the ecogen object if kinmatrix = NULL (Default option).
int	Distance interval in the units of XY.
smin	Minimum class distance in the units of XY.
smax	Maximum class distance in the units of XY.
nclass	Number of classes.
kmax	Number of nearest-neighbors for local analysis.
seqvec	Vector with breaks in the units of XY.
size	Number of individuals per class.
type	Weight mode for local analysis: "knearest" for nearest neighbors, "radialdist" for radial distances. Default is knearest.

cubic	Should be performed a cubic interpolation ( $res \sim \ln(d_{ij})$ ) with the regression residuals ( $res$ ) of $(kinship)_{ij} \sim \ln(d_{ij})$ ? Default TRUE.
testclass.b	Should be performed a permutation test in each individual class? Default TRUE.
testmantel.b	Should be performed a Mantel test for testing the slope ( $b$ )? Default TRUE.
jackknife	Should be performed jackknife in each individual class for computing the standard deviation (SD) of the coancestry (class) values? Default TRUE.
cummulative	Should be constructed a cumulative correlogram?.
normLocal	Normalize the local kinship values ( $[local\_kinship - mean]/sd$ )? Default TRUE
nsim	Number of Monte-Carlo simulations.
test	If test = "bootstrap", the program generates a bootstrap resampling and the associated confidence intervals of the null hypothesis. If test = "permutation" (default) a permutation test is made and the P-values are computed.
alternative	The alternative hypothesis. If "auto" is selected (default) the program determines the alternative hypothesis. Other options are: "two.sided", "greater" and "less".
sequential	Should be performed a Holm-Bonferroni (Legendre and Legendre, 2012) adjustment of P-values for global analysis? Default TRUE.
conditional	Logical. Should be used a conditional randomization? (Anselin 1998, Sokal and Thomson 2006). The option "auto" sets conditional = TRUE for LISA methods and G, as suggested by Sokal (2008).
bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
row.sd	Logical. Should be row standardized the matrix? Default FALSE (binary weights).
adjust	P-values correction method for multiple tests passed to <code>p.adjust</code> . Default is "holm".
latlon	Are the coordinates in decimal degrees format? Default FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <code>geoXY</code> .
angle	direction for computation of bearing correlogram (angle between 0 and 180). Default NULL (omnidirectional).

## Details

The GLOBAL ANALYSIS mode, computes a multilocus correlogram, with a detailed summary (see the content of the slot OUT in the "return" section). It also computes (see details about the slot SP in the "return" section): - the slope of the kinship individual values vs the logarithm of the distance,  $(kinship)_{ij} \sim \ln(d_{ij})$ , with a jackknife confidence interval - a Mantel test for testing the association between  $(kinship)_{ij}$  and  $\ln(d_{ij})$  - The  $S_p$  statistic (Vekemans and Hardy, 2004) with confidence intervals - A cubic interpolation of  $(kinship)_{ij} \sim \ln(d_{ij})$  residuals vs  $\ln(d_{ij})$

A directional approach is based in bearing analysis method. This compute the directional correlogram using the method of Rosenberg (2000). The slope is computed for the logarithm of  $D'$  (Born et al 2012), where  $D'$  is the distance matrix between individuals weighted by  $\cos(\alpha - B)^2$ ,



being  $\alpha$  the angle between individuals and  $B$  the desired direction angle, with  $B = 0$  the positive x axis,  $B = 90$  the positive y axis, and  $B = 180$  the negative x axis.

The LOCAL ANALYSIS mode, computes a local kinship estimate, based in a weighted mean (for each individual). The signification of each local statistic is computed using a permutation test, as in eco.lsa (see ?"eco.lsa"). Default option do not adjust the individual P values for multiple corrections.

## Value

For the global analysis, the program returns an object of class "eco.IBD" with the following slots:

> OUT analysis output.

In the permutation test case contains: - d.mean: mean class distance; - d.log: mean logarithm of the class distance; - obs, exp, alter, p.val: observed, and expected value of the statistic under randomization, alternative, P value; - mean.jack, sd.jack, Jack.CI.inf, Jack.CI.sup: jackknifed mean and SD, and confidence intervals for the statistic; - null.lwr, nul.uppr: lower and upper bound of the jackknife confidence interval for the statistic; - cardinal: number of individuals in each class;

In the bootstrap test case contains: - d.mean: mean class distance; - d.log: mean logarithm of the class distance; - obs: observed value of the statistic; - mean.jack, sd.jack, Jack.CI.inf, Jack.CI.sup: jackknifed mean and SD, and confidence intervals for the statistic; - null.lwr, nul.uppr: lower and upper bound of the jackknife confidence interval for the statistic; - cardinal: number of individuals in each class;

> GLOBALTEST Oden's (1984) global test of significance for the correlogram. The test consists in checking if the most significant kinship coefficient is significant at a Bonferroni- corrected significance level of  $\alpha' = \alpha/k$ , where  $k$  is the number of distance classes of the correlogram;  $\alpha$  is set to 0.05. The program return the values: "SIGNIFICANT" or "NOT-SIGNIFICANT"

> IN analysis input data

> SP Sp statistic results

It contains:

- the regression model; - information about the distance interval used for the regression (restricted); - slope (bhat) information (bhat = estimate, SD= bhat jackknife SD, theta = bhat jackknife mean, CI 5% and 95% = 95% confidence interval for bhat); - X-intercept = dij intercept (in the original units) for the line with slope "bhat", F1 = first class statistic value, and F1 5% and 95% = confidence interval for the first class statistic; - mantel.obs.b = observed value of the Mantel test between kinship(Fij) and  $\ln(d_{ij})$ ; mantel.pval.b = Mantel test P value; - sp = Sp statistics (sp = Sp observed value, CI 5% and 95% = 95% confidence interval for Sp); - cubic\_model = cubic model for (kinship) $_{ij} \sim \ln(d_{ij})$  r esiduals vs  $\ln(d_{ij})$ ;

> BEAKS breaks

> CARDINAL number of elements in each class

> NAMES variables names

> METHOD analysis method

> DISTMETHOD method used in the construction of breaks

> TEST test method used (bootstrap, permutation)

> NSIM number of simulations

> PADJUST P-values adjust method for permutation tests

---

For the local analysis, the program returns an object of class "eco.lsa" with the following slots:

> OUT results

> In the permutation test case contains:

- d.mean: mean class distance - obs, exp, alter, p.val: observed, and expected value of the statistic under randomization, alternative, P value; - null.lwr, nul.uppr: lower and upper bound of the jackknife confidence interval for the statistic; - cardinal: number of individuals in each class;

> In the bootstrap test case contains: - d.mean: mean class distance; - obs: observed value of the statistic; - null.lwr, nul.uppr: lower and upper bound of the jackknife; confidence interval for the statistic; - cardinal: number of individuals in each class;

> METHOD method (coefficient) used in the analysis

> TEST test method used (bootstrap, permutation)

> NSIM number of simulations

> PADJUST P-values adjust method for permutation tests

> COND conditional randomization (logical)

> XY input coordinates

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### References

- Born C., P. le Roux, C. Spohr, M. McGeoch, B. Van Vuuren. 2012. Plant dispersal in the sub-Antarctic inferred from anisotropic genetic structure. *Molecular Ecology* 21: 184-194.
- Double M., R. Peakall, N. Beck, and Y. Cockburn. 2005. Dispersal, philopatry, and infidelity: dissecting local genetic structure in superb fairy-wrens (*Malurus cyaneus*). *Evolution* 59: 625-635.
- Kalisz S., J. Nason, F.M. Handazawa, and S. Tonsor. 2001. Spatial population genetic structure in *Trillium grandiflorum*: the roles of dispersal, mating, history, and selection. *Evolution* 55: 1560-1568.
- Loiselle B., V. Sork, J. Nason, and C. Graham. 1995. Spatial genetic structure of a tropical understory shrub, *Psychotria officinalis* (Rubiaceae). *American Journal of Botany* 1420-1425.
- Oden, N., 1984. Assessing the significance of a spatial correlogram. *Geographical Analysis*, 16: 1-16.
- Rosenberg, M. 2000. The bearing correlogram: a new method of analyzing directional spatial autocorrelation. *Geographical Analysis*, 32: 267-278.
- Vekemans, X., and O. Hardy. 2004. New insights from fine-scale spatial genetic structure analyses in plant populations. *Molecular Ecology*, 13: 921-935.

**Examples**

```

## Not run:

data(eco.test)

# ---global analysis---

globaltest <- eco.malecot(eco=eco, method = "global", smax=10,
                          size=1000)
eco.plotCorrelog(globaltest) # Significant mean class coancestry classes at
# individual level (alpha = 0.05,
# out of the red area),
# and family-wise P corrected values (red-blue
# points, indicated in the legend)

# ecoslot.SP(globaltest) contains:
# - the slope (bhat) and values with confidence intervals
# of the regression reg = kinship ~ ln(distance_between_individuals)
#- A Mantel test result for assesing the relation between
# between kinship and ln(distance_between_individuals)
#- A cubic interpolation between the residuals of reg and
# ln(distance_between_individuals)
#- the sp statistic and its confidence interval

# ecoslot.OUT(globaltest) contains:
# - In permutation case, the values of mean and log-mean distance
# classes; observed class value; expected + alternative + P value,
# the bootstrap null confidence intervals and
# jackknife statistics (jackknifed mean, jackknifed SD, and
# CI for the class statistic)

# - In bootstrap case, the values of mean and log-mean distance
# classes; the bootstrap null confidence intervals and
# jackknife statistics (jackknifed mean, jackknifed SD, and
# CI for the class statistic)

A directional approach based in bearing correlograms, 30 degrees
globaltest_30 <- eco.malecot(eco=eco, method = "global", smax=10,
                             size=1000, angle = 30)
eco.plotCorrelog(globaltest, angle = 30)

#-----#
# ---local analysis---

(using the spatial weights).

# ---local analysis with k nearest neighbors---

```

```

localktest <- eco.malecot(eco=eco, method = "local",
                        type = "knearest", kmax = 5,
                        adjust = "none")
eco.plotLocal(localktest)

# ---local analysis with radial distance---

localdtest <- eco.malecot(eco=eco, method = "local",
                        type = "radialdist", smax = 3,
                        adjust = "none")

eco.plotLocal(localdtest)                # rankplot graphic (see ?"eco.rankplot")
                                        # Significant values
                                        # in blue-red scale,
                                        # non significant
                                        # values in yellow

eco.plotLocal(localktest, significant = FALSE) # significant and non
                                              # significant values
                                              # in blue-red scale

# The slot OUT of localktest (ecoslot.OUT(localktest)) and localdtest
# (ecoslot.OUT(localdtest)) contains:
# - the mean distance per individual, observed value of the
#   statistic, expected + alternative + P value + null hypothesis
#   confidence intervals, or bootstrap confidence intervals in
#   permutation or bootstrap cases, respectively.

## End(Not run)

```

---

eco.mantel

*Mantel and partial Mantel tests, with truncation option*


---

## Description

This program computes the Mantel test between the distance matrices  $d_1$  and  $d_2$ , or a partial Mantel test between the distance matrices. The test can be performed for truncated distances (Legendre et al. 2015) or with a direction Falsetti and Sokal (1993) using a weights object generated with [eco.bearing](#).  $d_1$  and  $d_2$ , conditioned on  $d_c$ .

## Usage

```

eco.mantel(d1, d2, dc = NULL, con = NULL, thres = NULL,
          truncMat = c("d2", "d1", "dc"), method = c("pearson", "spearman",
          "kendall"), nsim = 99, alternative = c("auto", "two.sided", "less",
          "greater"), plotit = TRUE, ...)

```

**Arguments**

d1	Distance matrix.
d2	Distance matrix.
dc	Distance matrix (optional).
con	binary eco.weight object used for truncation or a weights object obtained with eco.bearing.
thres	threshold distance used for truncation. distances above the threshold are set as 4 times the threshold. If null, and con is not null, it is set to the maximum distance observed in d2
truncMat	Matrix to be used for truncation (default = d2)
method	Correlation method used for the construction of the statistic ("pearson", "spearman" or "kendall"). Kendall's tau computation is slow.
nsim	Number of Monte-Carlo simulations.
alternative	The alternative hypothesis. If "auto" is selected (default) the program determines the alternative hypothesis. Other options are: "two.sided", "greater" and "less".
plotit	Should be generated a plot of the simulations?
...	Additional arguments passed to <a href="#">cor</a> .

**Value**

An object of class "eco.gsa" with the following slots:

- > METHOD method used in the analysis
- > OBS observed value
- > EXP expect value
- > PVAL P-value
- > ALTER alternative hypothesis
- > NSIM number of simulations

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

- Falsetti A., and Sokal R. 1993. Genetic structure of human populations in the British Isles. *Annals of Human Biology* 20: 215-229.
- Legendre P. 2000. Comparison of permutation methods for the partial correlation and partial Mantel tests. *Journal of Statistical Computation and Simulation*, 67: 37-73.

Legendre P., and M. Fortin. 2010. Comparison of the Mantel test and alternative approaches for detecting complex multivariate relationships in the spatial analysis of genetic data. *Molecular Ecology Resources*, 10: 831-844.

Mantel N. 1967. The detection of disease clustering and a generalized regression approach. *Cancer research*, 27: 209-220.

Smouse P. Long and R. Sokal. 1986. Multiple regression and correlation extensions of the Mantel test of matrix correspondence. *Systematic zoology*, 627-632.

## Examples

```
## Not run:

data(eco.test)

eco.mantel(d1 = dist(eco[["P"]]), d2 = dist(eco[["E"]]), nsim = 99) # ordinary Mantel test

pm <- eco.mantel(d1 = dist(eco[["P"]]), d2 = dist(eco[["E"]]),
dc = dist(eco[["XY"]]), nsim = 99) # partial Mantel test

# checking threshold in a correlogram:
corm <- eco.cormantel(M = dist(eco[["P"]]), XY = eco[["XY"]], nsim = 99)
eco.plotCorrelog(corm)
Correlation is around 0 when distance between points is > 5

# create a weights object for truncation
con <- eco.weight(eco@XY, method="circle", d2=5)
#compute a truncated mantel test
eco.mantel(dist(eco[["P"]]), dist(eco[["XY"]]), con=con)

# analyze test mantel in a direction of 35 degrees
con <- eco.bearing(XY = eco[["X"]], thata = 37)
eco.mantel(dist(eco[["P"]]), dist(eco[["XY"]]), con = con2)

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.OBS(pm) # slot OBS (observed value)
ecoslot.PVAL(pm) # slot PVAL (P-value)

## End(Not run)
```

---

eco.merge	<i>Merging two ecogen objects. Ordering the rows of an ecogen object according to the rows of another</i>
-----------	---

---

### Description

Merging two ecogen objects. Ordering the rows of an ecogen object according to the rows of another

### Usage

```
eco.merge(e1, e2, ...)
```

### Arguments

e1	Object of class "ecogen".
e2	Object of class "ecogen".
...	Data frames to merge. Could be any combination of the following: "XY", "P", "G", "E" and "C", or "ALL". If a "G" data frame is provided, the program generates also the INT slot coding the missing data as "0".

### Details

This program generates an ecogen object binding the columns of the individuals with matching row names in e1 and e2. If the objects have different number of rows, the result is a merged data frame with the rows in the order of the first object. If the objects have the same number of rows, but in a different order, the product is an object with the rows ordered as the first object. The algorithm matches sequentially the data frame pairs of each slot that the user wishes to merge.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:  
data(eco.test)  
eco  
eco1 <- eco[2:20]  
merged <- eco.merge(eco, eco1)  
merged  
  
## End(Not run)
```

---

eco.NDVI                      *Generating atmospherically corrected NDVI and MSAVI2 images for temporal series of Landsat 5 and 7*

---

## Description

This program generates atmospherically corrected images of NDVI and MSAVI2. The images of multiple dates can be processed in a single run. The bands 4 and 3 of each date (previously subsetted to the region of analysis) must be in the working directory. A table with information for each image as described in the parameter tab (see also the example) is needed for processing the information.

## Usage

```
eco.NDVI(tab, correct = c("COST", "DOS"), method = c("NDVI", "MSAVI2"),
         landsat = c("LT5", "LT7.L", "LT7.H"), datatype = c("FLT4S", "FLT8S",
         "INT4U", "INT4S", "INT2U", "INT2S", "INT1U", "INT1S", "LOG1S"))
```

## Arguments

tab	data.frame with 7 columns: The date of the images (format: YYYY/MM/DD), the sun elevation (both values could be extracted from Landsat headers), the name of the band 4, the name of the band 3, the starting haze value of the band 4, the starting haze value of the band 3, and the name of the output file. Each row corresponds to an image of different date.
correct	Correction method ("COST", "DOS").
method	Vegetation index ("NDVI", "MSAVI2").
landsat	Satellite data source ("LT5" for Landsat 5, "LT7.L" for Landsat 7 low gain and "LT7.H" for Landsat 7 high gain).
datatype	type of data, see <a href="#">dataType</a> . Default "FLT4S".

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## References

- Chander G., B. Markham, and D. Helder. 2009. Summary of current radiometric calibration coefficients for Landsat MSS, TM, ETM+, and EO-1 ALI sensors. *Remote sensing of environment*, 113: 893-903.
- Chavez P. 1989. Radiometric calibration of Landsat Thematic Mapper multispectral images. *Photogrammetric Engineering and Remote Sensing*, 55: 1285-1294.
- Chavez P. 1996. Image-based atmospheric corrections-revisited and improved. *Photogrammetric engineering and remote sensing*, 62: 1025-1035.
- Goslee S. 2011. Analyzing remote sensing data in R: the landsat package. *Journal of Statistical Software*, 43: 1-25.



Song C., C. Woodcock, K. Seto, M. Lenney and S. Macomber. 2001. Classification and change detection using Landsat TM data: when and how to correct atmospheric effects?. Remote sensing of Environment, 75: 230-244.

Tucker C. 1979. Red and photographic infrared linear combinations for monitoring vegetation. Remote sensing of Environment, 8: 127-150.

### Examples

```
## Not run:
require(raster)

data(tab)

temp <-list()

# we create 4 simulated rasters for the data included in the object tab:

for(i in 1:4) {
temp[[i]] <- runif(19800, 0, 254)
temp[[i]] <- matrix(temp[[i]], 180, 110)
temp[[i]] <- raster(temp[[i]], crs="+proj=utm")
extent(temp[[i]])<-c(3770000, 3950000, 6810000, 6920000)
}

writeRaster(temp[[1]], "20040719b4.tif", overwrite=T)
writeRaster(temp[[2]], "20040719b3.tif", overwrite=T)
writeRaster(temp[[3]], "20091106b4.tif", overwrite=T)
writeRaster(temp[[4]], "20091106b3.tif", overwrite=T)

# Computing NDVI images:

eco.NDVI(tab, "COST", "NDVI", "LT5")

example <- raster("NDVICOST20040719.tif")
image(example)

## End(Not run)
```

---

eco.NDVI.post

*Postprocessing for NDVI and MSAVI 2 temporal series of Landsat 5 and 7*

---

### Description

This program must be used sequentially after `eco.NDVI`. The inputs required (tab, correct, method) are the same described and used in that function. The algorithm stacks the images and save the stack into the working directory with the name "time.tif". If the user wishes, the program can also compute images of max, min, mean and var by pixel over the temporal sequence. Default is "mean".

**Usage**

```
eco.NDVI.post(tab, correct = c("COST", "DOS"), method = c("NDVI", "MSAVI2"),
  datatype = c("FLT4S", "FLT8S", "INT4U", "INT4S", "INT2U", "INT2S", "INT1U",
    "INT1S", "LOG1S"), what = c("mean", "max", "min", "var", "none"))
```

**Arguments**

tab	Table used with <code>eco.NDVI</code> .
correct	Correction method used in <code>eco.NDVI</code> .
method	The vegetation index used in <code>eco.NDVI</code> .
datatype	Type of data, see <code>dataType</code> . Default "FLT4S".
what	Functions to apply over the created stack. The values permitted are: "none", "max", "min", "mean" and "var". The functions are implemented with <code>calc</code> . If more than one function would be applied, must be used the following syntax: <code>c("fun_1", "fun:2", "fun_i")</code> .

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

- Chander G., B. Markham, and D. Helder. 2009. Summary of current radiometric calibration coefficients for Landsat MSS, TM, ETM+, and EO-1 ALI sensors. *Remote sensing of environment*, 113: 893-903.
- Chavez P. 1989. Radiometric calibration of Landsat Thematic Mapper multispectral images. *Photogrammetric Engineering and Remote Sensing*, 55: 1285-1294.
- Chavez P. 1996. Image-based atmospheric corrections-revisited and improved. *Photogrammetric engineering and remote sensing*, 62: 1025-1035.
- Goslee S. 2011. Analyzing remote sensing data in R: the landsat package. *Journal of Statistical Software*, 43: 1-25.
- Song C., C. Woodcock, K. Seto, M. Lenney and S. Macomber. 2001. Classification and change detection using Landsat TM data: when and how to correct atmospheric effects?. *Remote sensing of Environment*, 75: 230-244.
- Tucker C. 1979. Red and photographic infrared linear combinations for monitoring vegetation. *Remote sensing of Environment*, 8: 127-150.

**See Also**

eco.NDVI  
extract

**Examples**

```
## Not run:
require(raster)

data(tab)
data(eco3)
temp <- list()

# we create 4 simulated rasters for the data included in the object tab:

for(i in 1:4) {
  temp[[i]] <- runif(19800, 0, 254)
  temp[[i]] <- matrix(temp[[i]], 180, 110)
  temp[[i]] <- raster(temp[[i]], crs="+proj=utm")
  extent(temp[[i]])<-c(3770000, 3950000, 6810000, 6920000)
}

writeRaster(temp[[1]], "20040719b4.tif", overwrite = T)
writeRaster(temp[[2]], "20040719b3.tif", overwrite = T)
writeRaster(temp[[3]], "20091106b4.tif", overwrite = T)
writeRaster(temp[[4]], "20091106b3.tif", overwrite = T)

# Computing NDVI images:

eco.NDVI(tab, "COST", "NDVI", "LT5")

# Mean NDVI image computed over the NDVI images that we calculated:

eco.NDVI.post(tab, "COST", "NDVI", what = c("mean", "var"))
mean.ndvi <- raster("NDVI.COST.mean.tif")
plot(mean.ndvi)

# Extraction of the mean NDVI for each point in the object eco and plot
# of the data:

ndvi <- extract(mean.ndvi, eco3[["XY"]])
ndvi<- aue.rescale(ndvi)
plot(eco3[["XY"]][, 1], eco3[["XY"]][, 2], col=rgb(ndvi, 0, 0),
     pch=15, main = "Mean NDVI", xlab = "X", ylab = "Y")

## End(Not run)
```

**Description**

Functions deprecated in EcoGenetics version 1.2.0-2

**Usage**

```
eco.order(...)
```

**Arguments**

```
...           parameters
```

---

eco.pairtest	<i>Kruskall - Wallis + Wilcoxon (Mann-Whitney U) and aov + Tukey-HSD tests for an ecogen object</i>
--------------	---

---

**Description**

Kruskall - Wallis + Wilcoxon (Mann-Whitney U) and aov + Tukey-HSD tests for an ecogen object

**Usage**

```
eco.pairtest(eco, df = c("P", "E", "A", "C"), x, test = c("wilcoxon",
  "tukey"), adjust = "fdr", only.p = TRUE, ...)
```

**Arguments**

eco	Object of class "ecogen".
df	The data frame for the analysis. Could be "P", "E" "A" or "C". For dominant data, "A" is here considered here a synonym of the G data frame.
x	The name of the S slot column with the groups for the analysis.
test	Test to perform ("wilcoxon", "tukey").
adjust	P-values correction method for multiple tests passed to <a href="#">p.adjust</a> . Defalut is "fdr".
only.p	Should be only returned a matrix with P-values? Default TRUE.
...	Additional arguments passed to <a href="#">wilcox.test</a> or <a href="#">TukeyHSD</a> .

**Details**

This program returns the Wilcoxon (Mann-Whitney U) or Tukey-HSD statistics and p values for the multiple comparisons of the variables contained in the selected data frame, among the levels of a factor of the slot "S".

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[wilcox.test](#) [TukeyHSD](#)

**Examples**

```
## Not run:
data(eco3)
wil <- eco.pairtest(eco = eco3, df = "P", x = "structure")
wil
wil <- eco.pairtest(eco = eco3,df = "E", x = "structure")
wil
wil <- eco.pairtest(eco = eco3, df = "P", x = "structure", only.p = FALSE)
wil
wil <- eco.pairtest(eco = eco3,df = "P", x = "structure", test = "tukey")
wil

## End(Not run)
```

---

eco.plotCorrelog      *eco.plotCorrelog*

---

**Description**

Plot method for correlograms and variograms. For examples see [eco.correlog](#) [eco.cormantel](#) [eco.variogram](#)

**Usage**

```
eco.plotCorrelog(x, var = NULL, xlabel = NULL, ylabel = NULL,
  title = NULL, legend = TRUE, background = c("grey", "white"),
  errorbar = FALSE, intervals = TRUE, significant.S = TRUE,
  significant.M = FALSE, xlim = NULL, ylim = NULL, nsim = 999,
  interactivePlot = TRUE, meanplot = TRUE, randtest = c("permutation",
  "bootstrap", "none"), alpha = 0.05, quiet = FALSE)
```

**Arguments**

x	Result of correlogram or variogram analysis
var	Individual variable to plot for multiple analyses with <a href="#">eco.correlog</a> To plot multiple variables in a same plot, use only the argument x (see examples)
xlabel	Label for X axis (default: NULL)
ylabel	Label for Y axis (default: NULL)
title	Title of the plot (default: NULL)
legend	Show legends in ggplot graphs? (default: TRUE)
background	background color ("grey" or "white")
errorbar	Show error-bars? (default: FALSE)
intervals	Show bootstrap CI in kinship analysis? (default: TRUE)

significant.S	With single variables and permutation test: show different colours for significant points? (default: TRUE)
significant.M	With multiple variables: show only significant correlograms? (default: FALSE)
xlim	X axis limits (as vector: c(min, max); default: NULL)
ylim	Y axis limits (as vector: c(min, max); default: NULL)
nsim	Number of simulations for permutation or bootstrap tests.
interactivePlot	Show an interactive plot via plotly? (default: TRUE)
meanplot	Show a line with the mean when plot is for multiple variables? (default: TRUE)
randtest	Randomization test (one of: "permutation", "bootstrap", "none")
alpha	significance (alpha) for P (or P-adjusted) values (Default = 0.05)
quiet	print quietly? Default FALSE

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[eco.correlog](#) [eco.cormantel](#) [eco.variogram](#)

---

eco.plotCorrelogB      *eco.plotCorrelogB*

---

**Description**

Plot method for bearing correlograms For examples see [eco.correlog](#). It constructs an angular correlogram for each distance class taken as fixed.

**Usage**

```
eco.plotCorrelogB(x, var = NULL, xlabel = NULL, ylabel = NULL,
  title = NULL, legend = TRUE, background = c("grey", "white"),
  significant.S = TRUE, xlim = NULL, ylim = NULL,
  interactivePlot = TRUE)
```

**Arguments**

x	Result of correlogram analysis, with output using angles as independent variables for fixed distances (instead of distances as independent variables)
var	Individual variable to plot; var is a number between 1 and the number of distance classes indicating the corresponding class (for example, with 5 distance classes, the number 3 indicates the third) To plot multiple variables in a same plot, use only the argument x (see examples)

xlabel	Label for X axis (default: NULL)
ylabel	Label for Y axis (default: NULL)
title	Title of the plot (default: NULL)
legend	Show legends in ggplot graphs? (default: TRUE)
background	background color ("grey" or "white")
significant.S	With single variables and permutation test: show different colours for significant points? (default: TRUE)
xlim	X axis limits (as vector: c(min, max); default: NULL)
ylim	Y axis limits (as vector: c(min, max); default: NULL)
interactivePlot	Show an interactive plot via plotly? (default: TRUE)

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[eco.correlog](#)

---

eco.plotGlobal

*GSA plot methods*

---

**Description**

This function allows to plot eco.gsa multiple objects. For examples see [eco.gsa](#)

**Usage**

```
eco.plotGlobal(input, interactivePlot = TRUE, background = c("grey",
  "white"), xlabel = NULL, ylabel = NULL, title = NULL, legend = TRUE,
  rescaled = FALSE, alpha = 0.05)
```

**Arguments**

input	eco.gsa object
interactivePlot	Show an interactive plot via plotly? (default: TRUE)
background	background color ("grey" or "white")
xlabel	Label for X axis (default: NULL)
ylabel	Label for Y axis (default: NULL)
title	Title of the plot (default: NULL)
legend	Show legends in ggplot graphs? (default: TRUE)
rescaled	rescale join-count heatmap?
alpha	significance level for the join-count heatmap

**Author(s)**

Leandro Roser

---

`eco.plotLocal`*eco.plotLocal*

---

**Description**For examples see [eco.lsa](#)**SINGLE VARIABLES:**

Using permutation test: The function calls `eco.rasterplot`, who generates a plot for a numeric or factor variable. The X and Y axes in the plot correspond to the rank of the X and Y coordinates, respectively. Additional parameters can be passed to `eco.rankplot`.

Using bootstrap test: The function calls `eco.forestplot`, who computes a forest plot for the confidence interval of each individual of the input data (as row number) and the corresponding observed value of the statistic. Additional parameters can be passed to `eco.forestplot`.

**MULTIPLE VARIABLES:** multiple output format results. "list" for object with a list of individual test for each variable, or "matrix" for results as matrices of multiples variables.

For results as matrices (option `multi = "matrix"` in `eco.lsa`): The function class `eco.rasterplot`, who generates a multivariate plot for a data matrix (raster). Additional parameters can be passed to `eco.rasterplot`. The `rasterplot` graph is a flexible tools for multiple data sources (environmental, genetic, phenotypic, etc.).

For results as list (option `multi = "list"` in `eco.lsa`): The function generates plots for individual variables calling `eco.rankplot`. Additional parameters can be passed to `eco.rankplot`.

**Usage**

```
eco.plotLocal(x, interactivePlot = TRUE, multi = c("d3heatmap", "ggplot"),
  significant = TRUE, alpha = 0.05, rescaled = FALSE, limits = NULL,
  title = NULL, z.name = NULL, grp = NULL, vertical = TRUE,
  legend = TRUE, n = 4, nrow = 2, byrow = TRUE, ...)
```

**Arguments**

<code>x</code>	Result of <code>eco.lsa</code> analysis
<code>interactivePlot</code>	Show an interactive plot via <code>plotly</code> ? (default: TRUE)
<code>multi</code>	for multivariable plot, use <code>d3heatmap</code> or <code>ggplot2</code> ? (Default: <code>d3heatmap</code> ). In <code>d3heatmap</code> , NA values are set to 0.
<code>significant</code>	Show all non significant points with a same colour?
<code>alpha</code>	significance (alpha) for P (or P-adjusted) values (Default: 0.05)
<code>rescaled</code>	rescale statistics between -1 and 1? (Default: FALSE)



limits	When multiple variables are used, values used as limits for computing the gradient for the plot
title	title of the plot
z.name	name of the variables axis in multivariable plot (using ggplot2 like plots)
grp	groups for multivariable plot (using ggplot2 like plots)
vertical	vertical multivariable plot? (Default: true)
legend	Show legend?
n	number of plot per screen in multivariable plots as list
nrow	number of rows in multivariable plots as lists
byrow	plot by row in multivariable plots by list
...	additional arguments passed to eco.rankplot, eco.forestplot, eco.resterplot o grf.seqmultiplot, depending of the selected plot

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

eco.plotWeight      *Plot for a connection network*

---

**Description**

Plot method for an eco.weight object. For examples see [eco.weight](#) This function can make a static plot with the original coordinates and an additional graph with the coordinates transformed as ranks. It also can construct dynamic plots (force network and circle network)

**Usage**

```
eco.plotWeight(x, type = c("simple", "igraph", "edgebundle", "network"),
  group = NULL, fontSize = 10, ebColor = NULL, vertex.size = 10,
  vertex.label = NA, bounded = FALSE, ...)
```

**Arguments**

x	connection network
type	plot type: "edgebundle", for a circular network, "network" for a tension network"
group	vector with classes assigned to the individuals, in the same original order
fontSize	argument passed to <a href="#">forceNetwork</a> contained in the weight object (which is the order of the table used to construct the weights)
ebColor	vector with edge colors for the groups of the edgebundler plot (Experimental feature)
vertex.size	parameter passed to <a href="#">plot.igraph</a>

vertex.label    parameter passed to [plot.igraph](#)  
 bounded        logical value to enable (TRUE) or disable (FALSE) the bounding box limiting  
                  theF graph's extent see [forceNetwork](#).  
 ...             additional arguments passed to [plot.igraph](#)

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
# see the examples in the function eco.weight:
# ?eco.weight
```

---

eco.post.geneland        *Log posterior probability plot for Geneland repetitions with fixed K*

---

**Description**

Log posterior probability plot for Geneland repetitions with fixed K

**Usage**

```
eco.post.geneland(niter, burnin)
```

**Arguments**

niter            Number of mcmc iterations per repetition.  
 burnin          Number of mcmc to burn-in.

**Details**

This program returns, for a series of Geneland repetitions with fixed K, and a specified burn-in value, a plot of the log posterior probability vs the repetition number. This allow to choose the best run. The working directory will be setted to the folder containing the results created by Geneland. The program expects each subfolder (run) having a number as name, indicating the corresponding number of run. (1, 2, etc., see the example).

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:
require("Geneland")
data(eco.test)

# We create a folder in the working directory for the results and
# save the data frames of the object "eco" in the format required
# by Geneland:

path.1 <- getwd()
path <- paste(path.1, "/test/", sep="")
dir.create(path)
setwd(path)
ecogen2geneland(eco, ploidy = 2)

# Auxiliar function for running some repetitions with fixed K = 4.
# Each repetition is saved in the folder "test":
simul <- function(i) {
  path <- getwd()
  path <- paste(path, "/", i, sep = "")
  dir.create(path)
  MCMC(coordinates = read.table("XY.txt"),
        geno.dip.codom = read.table("G.txt"),
        varnpop = TRUE, npopmin = 4, npopmax = 4, spatial = TRUE,
        freq.model = "Correlated", nit = 500, thinning = 10,
        path.mcmc = path)
}

# 5 repetitions with K = 4
lapply(1:5, simul)

# Check that in the folder "test" are the simulated result.
# Your results must have that appearance.

# Plot of the repetition order number vs the corresponding
# posterior probability, with a burn-in of 10 mcmc:
eco.post.geneland(5, 10)

## End(Not run)
```

---

eco.rankplot

*Rankplot graphs*


---

**Description**

This function generates a plot for a numeric or factor variable. A data frame/matrix with XY coordinates is required. The X and Y axes in the plot correspond to the rank of the X and Y coordinates, respectively.

**Usage**

```
eco.rankplot(input, XY, xlabel = NULL, ylabel = NULL, title = NULL,
  legendlabel = NULL, background = c("grey", "white"), ...)

## S4 method for signature 'eco.lsa,missing,missing'
eco.rankplot(input, XY, xlabel, ylabel,
  title, legendlabel, background = c("grey", "white"), significant = TRUE,
  rescaled = FALSE, ns = NULL, interactivePlot = TRUE)

## S4 method for signature 'numeric,dataframeORmatrix,missing'
eco.rankplot(input, XY, xlabel,
  ylabel, title, legendlabel, background = c("grey", "white"),
  interactivePlot = TRUE)

## S4 method for signature 'factor,dataframeORmatrix,missing'
eco.rankplot(input, XY, xlabel,
  ylabel, title, legendlabel, background = c("grey", "white"),
  interactivePlot = TRUE)
```

**Arguments**

input	Numeric/factor variable.
XY	Data frame or matrix with X-Y coordinates.
xlabel	Optional label for x axis.
ylabel	Optional label for y axis.
title	Optional title label.
legendlabel	Optional legend label.
background	color of the background ("grey" or "white")-
...	Additional elements to the generic.
significant	should be colored only the individuals with significant result?. This argument can be used with <a href="#">eco.lsa</a> results. Default TRUE
rescaled	rescale values to [-1, 1] range?
ns	color for non significant individuals, when significant = TRUE. This argument can be used with <a href="#">eco.lsa</a> results.
interactivePlot	Show an interactive plot via plotly? (default: TRUE)

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:
data(eco3)
```

```

# The data set eco3 has 50 points in two sites,
# but they are not visible in a usual X-Y plot
# due to the small distance among them

var <- eco3[["P"]][,1]
plot(eco3[["XY"]], col = var)
x <- sample(1:100, 30)
y <- sample(1:100, 30)

# in a rankplot graph, the inter-individual distances are
# reduced to a single scale
rankeco3 <- eco.rankplot(var, eco3[["XY"]])
rankeco3

# the rankplot method support the use of ggplot2 syntax with ggplot2 graphs
require(ggplot2)
rankeco3 <- eco.rankplot(var, eco3[["XY"]], interactivePlot = FALSE)
rankeco3 <- rankeco3 + theme_bw() + theme(legend.position="none")
rankeco3

## End(Not run)

```

---

eco.rasterplot

*rasterplot graphs*


---

## Description

This function generates a multivariate plot for a data matrix (raster), with an option for filtering the data and to graph using groups. The rasterplot graph is a flexible tools for multiple data sources (environmental, genetic, phenotypic, etc.).

## Usage

```

eco.rasterplot(x, filter = NULL, condition = NULL, grp = NULL,
  limits = NULL, title = NULL, z.name = NULL, vertical = TRUE,
  interactivePlot = TRUE, ...)

```

## Arguments

x	data matrix (raster)
filter	optional data matrix used as filter
condition	condition used to filter data
grp	factor with groups to use in the plot. Default NULL
limits	values limits used for computing the data gradient for the plot
title	plot title
z.name	name for the legend

```

vertical          should be partitioned the populations on the x axis? Default TRUE.
interactivePlot   Show an interactive plot via plotly? (default: TRUE)
...              additional arguments

```

## Examples

```

## Not run:
data(eco.test)

# using the ecogen object "eco" to perform a multiple-lsa
con <- eco.weight(eco[["XY"]], method = "knearest", k = 4, row.sd = TRUE)
test.lsa <- eco.lsa(eco[["P"]], con = con, method = "I", nsim = 99, multi = "matrix")

# the default method for this object, is a rasterplot
eco.plotLocal(test.lsa)

# adding a factor
test.lsa <- eco.lsa(eco[["P"]], con = con, method = "I",
  nsim = 99, multi = "matrix", pop = eco[["S"]][,1])
eco.plotLocal(test.lsa)

# The generic rasterplot method requires a data matrix, and, as option, a condition
# and a filter matrix. The condition is an expression, containing the word "filter" and
# logical elements, e.g., "filter < 50", "filter < 50 || filter > 2", etc. ).
# Filter is used as a logical matrix (TRUE-FALSE, in relation to the passed condition),
# for filtering the data. If a condition is passed but not a filter matrix, the condition
# is applied over the data matrix, also using the word "filter".
# Internally, the multi.lsa plot uses three fundamental elements.
# - a data matrix: in the example, ecoslot.OBS(test.lsa)
# - a filter matrix: in the example, ecoslot.PVAL(test.lsa); i.e.,
#   the data matrix will be filtered
# - by P-value using the third element, an expression.
# - an expression: in the example: "filter < 0.05"

# Combining the three elements, the multivariate plot can be manually constructed:
my.plot <- eco.rasterplot(x= ecoslot.OBS(test.lsa),
  filter = ecoslot.PVAL(test.lsa), condition = "filter < 0.05")

# add population
my.plot <- eco.rasterplot(x= ecoslot.OBS(test.lsa),
  filter = ecoslot.PVAL(test.lsa),
  condition = "filter < 0.05", grp = ecoslot.POP(test.lsa))

# Extra manipulation with ggplot2 graphs (ggplot2 commands allowed by rasterplot)
my.plot <- eco.rasterplot(x= ecoslot.OBS(test.lsa),
  filter = ecoslot.PVAL(test.lsa), condition = "filter < 0.05",
  interactivePlot = FALSE)

```

```

## rotate plot

my.plot + coord_flip()

## change design
my.plot + theme_grey()

# using the data as filter
eco.rasterplot(x= ecoslot.OBS(test.lsa), filter = ecoslot.OBS(test.lsa),
condition = "filter > 0 & filter < 3")

# example of bad syntax (incorrect use of && over matrices)
eco.rasterplot(x= ecoslot.OBS(test.lsa), filter = ecoslot.OBS(test.lsa),
condition = "filter > 0 && filter < 3")

## End(Not run)

```

---

eco.rasterplot,eco.multilsa-method

*rasterplot graph for eco.lsa results*

---

## Description

Plot method for local spatial analysis

## Usage

```

## S4 method for signature 'eco.multilsa'
eco.rasterplot(x, grp = NULL, limits = NULL,
  title = NULL, z.name = NULL, vertical = TRUE, significant = TRUE,
  rescaled = FALSE, alpha = 0.05, interactivePlot = TRUE, ...)

```

## Arguments

x	eco.multilsa object returned by <a href="#">eco.lsa</a> or
grp	factor with groups to use in the plot. Default NULL
limits	values limits used for computing the data gradient for the plot
title	plot title
z.name	name for the legend
vertical	should be partitioned the populations on the x axis? Default TRUE.
significant	plot only significant results? Default TRUE
rescaled	plot the rescaled observed values ([-1,1] range)?

alpha            threshold P value for results with permutation tests. default = 0.05.  
interactivePlot            Show an interactive plot via plotly? (default: TRUE)  
...            additional arguments

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[eco.lsa](#)

---

eco.rbind            *Combining the rows of two ecogen objects*

---

**Description**

Combining the rows of two ecogen objects

**Usage**

```
eco.rbind(..., check_colnames = TRUE, check_rownames = TRUE)
```

**Arguments**

...            "ecogen" objects to combine.  
check\_colnames    Check for duplicated column names? Default TRUE.  
check\_rownames    Check for duplicated row names? Default TRUE.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:  
  
data(eco.test)  
  
# split eco into a list of ecogen objects by population  
x <- eco.split(eco, "pop", asList = TRUE)  
  
# re-bind the objects  
eco.r <- eco.rbind(eco)  
  
# create a new objects with the first and second population
```



```

eco.r <- eco.rbind(x[[1]], x[[3]])

# duplicated row names are not allowed by eco.rbind with default options
eco2 <- eco
eco.rbind(eco, eco2)

eco.rbind(eco, eco2, check_rownames = FALSE)

## End(Not run)

```

---

eco.remove

*Creating an updated ecogen object by removing results of the slot OUT*


---

### Description

Creating an updated ecogen object by removing results of the slot OUT

### Usage

```
eco.remove(eco, ...)
```

### Arguments

eco	Object of class "ecogen".
...	Objects to remove from eco, typed without quotations.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```

## Not run:

data(eco.test)
variog <- eco.variogram(eco[["P"]][, 1], eco[["XY"]])

# Assignment of values can be made with the corresponding accessors,
# using the generic notation of EcoGenetics
# (<ecoslot.> + <name of the slot> + <name of the object>).
# See help("EcoGenetics accessors")

ecoslot.OUT(eco) <- variog
we.are.numbers <- c(1:10)
we.are.characters <- c("John Coltrane", "Charlie Parker")
ecoslot.OUT(eco) <- list(we.are.numbers, we.are.characters)
ecoslot.OUT(eco)

```

```
eco <- eco.remove(eco, we.are.numbers)
ecoslot.OUT(eco)
```

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

---

eco.slide.con

*Sliding window along a network*

---

## Description

This program applies a function defined by the user, over the individuals included in a connection network. For a given variable, the program computes recursively a function for the individuals of the network, using all the individuals connected to each. The function uses a connection network generated with the function `eco.weight`.

## Usage

```
eco.slide.con(x, con, fun)
```

## Arguments

<code>x</code>	Input variable or matrix.
<code>con</code>	Connection network.
<code>fun</code>	Function to apply in each focal point.

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## Examples

```
## Not run:

data(eco2)
myMatrix <- eco2[["P"]]
con <- eco.weight(XY = eco2[["XY"]], method = "knearest", k = 5)
result <- eco.slide.con(myMatrix, con, function(x) mean(x, na.rm = TRUE))

image(matrix(myMatrix[, 1], 30, 30)) # original image
image(matrix(result[, 1], 30, 30)) # smoothed image

data(eco3)
myMatrix2 <- eco3[["P"]]
con <- eco.weight(XY = eco3[["XY"]], method="knearest", k = 5)
eco.plotWeight(con)
# smoothing values in myMatrix2 using the connection network:
```

```

result <- eco.slide.con(myMatrix2, con, function(x) mean(x, na.rm = TRUE))

## End(Not run)

```

---

eco.slide.matrix      *Sliding window for matrix data*

---

### Description

This program applies a function defined by the user, using a moving window (circle area or square) and assigning the value to the focal pixel.

### Usage

```

eco.slide.matrix(mat, r, slide, fun, window = c("square", "circle"),
  within = TRUE)

```

### Arguments

mat	Input raster matrix.
r	Half a side for square, radius for circle, diagonal length for rhombus.
slide	Number of elements between two focal pixels, for column and row dimension
fun	Function to apply in each focal pixel.
window	Window type. Default "square".
within	Should be computed the function in borders focal pixels only if the area is within the matrix? Default TRUE.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```

## Not run:

data(eco.test)
ras <- matrix(eco[["P"]][, 1], 15, 15)
image(ras)
ras.square <- eco.slide.matrix(ras, 1, 1, mean, "square")
image(ras.square)

# or allowing more control over the function:
ras.square <- eco.slide.matrix(ras, r = 3, slide = 1, function(x) mean(x, na.rm = TRUE), "square")
image(ras.square)

```

```
# sliding a circle:
ras.circle <- eco.slide.matrix(ras, r = 3, slide = 1, mean, "circle", within = FALSE)
image(ras.circle)

## End(Not run)
```

---

eco.split

*Splitting an ecogen object by groups*


---

### Description

The function splits an ecogen objects into the groups of a hierarchy contained in the slot S. If asList is TRUE, a list with the objects is created, that can be assigned to a name with regular rules, using the operator "<-". Else, the function creates one ecogen object for each level in the workspace with the following nomenclature: <name of ecogen object>.<name of hierarchical level>.

### Usage

```
eco.split(eco, hier, name = NULL, overwrite = FALSE, missing = c("0",
  "NA", "MEAN"), asList = TRUE)
```

### Arguments

eco	Object of class "ecogen".
hier	The name of the S slot column with labels assigning individuals to groups.
name	Name used for the output objects. Default is the name of the input, followed by a suffix (see Description).
overwrite	Overwrite files in workspace with same name if exist? Missing elements are treated as zeros in the default option.
missing	Missing data argument This can take three values ("0", "NA" or "MEAN"), as described in <a href="#">ecogen</a> .
asList	Return a list with the objects instead of creating objects in workspace? Default = TRUE

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```
## Not run:
data(eco3)
eco3
#list of objects
x <- eco.split(eco3, "structure", asList = TRUE)
#r-binding
eco.bind <- eco.rbind(x)
# note that different subsets can also be created
S1.3 <- eco.rbind(x[[1]], x[[3]])

# split and create objects with prefix "eco3"
eco.split(eco3,"structure", asList = FALSE)
# split and create objects with prefix "newObjects"
eco.split(eco3,"structure", "newObjects", asList = FALSE)

## End(Not run)
```

---

eco.subset

*Subsetting an ecogen object by group*


---

**Description**

Subsetting an ecogen object by group

**Usage**

```
eco.subset(eco, hier, grp, missing = c("0", "NA", "MEAN"))
```

**Arguments**

eco	Object of class "ecogen".
hier	The name of the S slot column with labels assigning individuals to groups.
grp	Label for the subset of individuals, contained in hier.
missing	Missing data argument This can take three values ("0", "NA" or "MEAN"), as described in <a href="#">ecogen</a> . Missing elements are treated as zeros in the default option.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## Examples

```
## Not run:
data(eco3)
eco3
eco.sub <-eco.subset(eco3,"structure", 1)
eco.sub

## End(Not run)
```

---

eco.theilsen

*Theil-sen regression for a raster time series*

---

## Description

This function computes the theil-sen estimator and the P-value associated for each pixel over time in a stack of images, writing the values in a raster (one for the estimators and one for the P-values). It is recommended to use a "RasterBrick", that is more efficient in managing memory.

## Usage

```
eco.theilsen(stacked, date, adjust = "none")
```

## Arguments

stacked	Stacked images ("RasterLayer" or "RasterBrick").
date	data vector with dates for each image.
adjust	P-values correction method for multiple tests passed to <a href="#">p.adjust</a> . Defalut is "none".

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## References

Sen, P. 1968. Estimates of the regression coefficient based on Kendall's tau. Journal of the American Statistical Association, Taylor and Francis Group, 63: 1379-1389.

Theil H. 1950. A rank-invariant method of linear and polynomial regression analysis, Part 3 Proceedings of Koninalijke Nederlandse Akademie van Weinenschatpen A, 53: 397-1412.

## See Also

[rkt](#).

**Examples**

```
## Not run:
require("raster")
set.seed(6)

temp <- list()
for(i in 1:100) {
  temp[[i]] <- runif(36,-1, 1)
  temp[[i]] <- matrix(temp[[i]], 6, 6)
  temp[[i]] <- raster(temp[[i]])
}

temp <- brick(temp)

writeRaster(temp,"temporal.tif", overwrite=T)
rm(temp)
ndvisim <- brick("temporal.tif")

date <- seq(from = 1990.1, length.out = 100, by = 0.2)

eco.theilsen(ndvisim, date)

pvalue <- raster("pvalue.tif")
slope <- raster("slope.tif")
par(mfrow = c(1, 2))
plot(pvalue, main = "p-value")
plot(slope, main = "slope")

## End(Not run)
```

---

eco.variogram

*Empirical variogram*


---

**Description**

This program computes the empirical variogram of a selected variable. If the coordinates are given in decimal degrees, set `latlon = TRUE`. The program return a table with the mean class distances (`d.mean`) and the semivariances (`obs`) for each class.

**Usage**

```
eco.variogram(Z, XY, int = NULL, smin = 0, smax = NULL, nclass = NULL,
  seqvec = NULL, size = NULL, bin = c("sturges", "FD"), row.sd = FALSE,
  latlon = FALSE, angle = NULL)
```

**Arguments**

Z	Vector for the analysis.
XY	Data frame or matrix with individual's position (projected coordinates).
int	Distance interval in the units of XY.
smin	Minimum class distance in the units of XY.
smax	Maximum class distance in the units of XY.
nclass	Number of classes.
seqvec	Vector with breaks in the units of XY.
size	Number of individuals per class.
bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
row.sd	Logical. Should be row standardized the matrix? Default FALSE (binary weights).
latlon	Are the coordinates in decimal degrees format? Default FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <code>geoXY</code> .
angle	direction for computation of bearing correlogram (angle between 0 and 180). Default NULL (omnidirectional).

**Value**

The program returns an object of class "eco.correlog" with the following slots:

- > OUT analysis output
- > IN analysis input data
- > BEAKS breaks
- > CARDINAL number of elements in each class
- > DISTMETHOD method used in the construction of breaks

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**References**

- Borcard D., F. Gillet, and P. Legendre. 2011. Numerical ecology with R. Springer Science & Business Media.
- Legendre P., and L. Legendre. 2012. Numerical ecology. Third English edition. Elsevier Science, Amsterdam, Netherlands.
- Rosenberg, M. 2000. The bearing correlogram: a new method of analyzing directional spatial autocorrelation. *Geographical Analysis*, 32: 267-278.



**Examples**

```
## Not run:

data(eco.test)
variog <- eco.variogram(Z = eco[["P"]][, 2],XY = eco[["XY"]])
plot(variog)

# variogram plots support the use of ggplot2 syntax
require(ggplot2)
variogplot <- eco.plotCorrelog(variog) + theme_bw() + theme(legend.position="none")
variogplot

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.OUT(variog)      # slot OUT
ecoslot.BREAKS(variog)  # slot BREAKS

## End(Not run)
```

---

eco.weight

*Spatial weights*


---

**Description**

Spatial weights for individuals (nodes) with coordinates XY

**Usage**

```
eco.weight(XY, method = c("circle", "knearest", "inverse", "circle.inverse",
  "exponential", "circle.exponential"), W = NULL, d1 = 0, d2 = NULL,
  k = NULL, p = 1, alpha = 1, dist.method = "euclidean",
  row.sd = FALSE, max.sd = FALSE, self = FALSE, latlon = FALSE,
  ties = c("unique", "min", "random", "ring", "first"))
```

**Arguments**

XY	Matrix/data frame with projected coordinates.
method	Method of spatial weight matrix: "circle", "knearest", "inverse", "circle.inverse", "exponential", "circle.exponential".

W	Custom weight matrix, with rownames and colnames identical to the XY data frame with coordinates
d1	Minimum distance for circle matrices.
d2	Maximum distance for circle matrices.
k	Number of neighbors for nearest neighbor distance. When equidistant neighbors are present, the program select them randomly.
p	Power for inverse distance. Default = 1.
alpha	Alpha value for exponential distance. Default = 1.
dist.method	Method of computing distance when XY is in metric units. If latlon is TRUE, the method is euclidean.
row.sd	Logical. Should be row standardized the matrix? Default FALSE (binary weights).
max.sd	Logical. Should be divided each weight by the maximum of the matrix? Default FALSE (binary weights).
self	Should be the individuals self-included in circle or knearest weights? Defalut FALSE.
latlon	Are the coordinates in decimal degrees format? Defalut FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <code>geoXY</code> .
ties	ties handling method for "knearest" method: "unique" (default) for counting the ties as an unique neighbor (i.e), "min" for counting all the ties in a given category but each is counted as a neighbor, "random" for choosing at random a neighbor, "ring" for ring of neighbors, "first" for sequential k values for each neighbor.

## Details

This program computes a weights matrix (square matrix with individuals in rows and columns, and weights  $w_{ij}$  in cells (i and j, individuals)) under the following available methodologies:

- circle: all the connection between individuals i and j, included in a distance radius, higher than d1 and lower than d2, with center in the individual i, have a value of 1 for binary weights. This distance requires the parameters d1 and d2 (default d1 = 0).
- knearest: the connections between an individual and its nearest neighbors of each individual i have a value of 1 for binary weights. This distance requires the parameter k.
- inverse: inverse distance with exponent p (distance =  $1/d_{ij}^p$ , with  $d_{ij}$  the distance between individuals i and j). This distance requires the parameter p (default p = 1).
- circle inverse: combination of "circle" and "inverse". It is the matrix obtained by multiplying each element in a "circle" binary matrix, and an "inverse" matrix. This distance requires the parameters p, d1 and d2 (default p = 1, d1 = 0).
- exponential: inverse exponential distance with parameter alpha (distance =  $1/e^{(\alpha * d_{ij})}$ , with  $d_{ij}$  the distance between individuals i and j). This distance requires the parameter alpha (default alpha = 1).
- circle exponential: combination of "circle" and "exponential". It is the matrix obtained by multiplying each element in a "circle" binary matrix, and an "exponential" matrix. This distance requires the parameters alpha, d1 and d2 (default alpha = 1, d1 = 0).

In addition to these methods, a spatial weight object can be created assigning a custom W matrix ("W" argument). In this case, the "method" is argument automatically set by the program to "custom" (see te example).

---

In row standardization, each weight  $w_{ij}$  for the individual  $i$ , is divided by the sum of the row weights (i.e.,  $w_{ij} / \text{sum}(w_{ij})$ , where  $\text{sum}(w_{ij})$  is computed over an individual  $i$  and all individuals  $j$ ).

When self is TRUE, the connection  $j = i$  is also included.

---

#### PLOTS FOR ECO.WEIGHT OBJECTS:

A plot method is available (function "eco.plotWeight") showing static or interactive plots, In the case of using the function eco.plotWeight for the argument type="simple", the connections are shown in two plots: an X-Y graph, with the individuals as points, representing the original coordinates, and in a plot with coordinates transformed as ranks (i.e., each coordinate takes an ordered value from 1 to the number of individuals). The other static method (type="igraph") uses the igraph package to generate a visual attractive graph (force network). Two interactive methods are available: type = "network", to plot an interactive force network, and type = "edgebundle" to plot a circular network. For the cases type = "inverse" or type = "exponential", the program generates a plot of weights values vs distance See the examples below.

#### Value

An object of class eco.weight with the following slots:

- > W weights matrix
- > XY input coordinates
- > METHOD weights construction method
- > PAR parameters used for the construction of weights
- > PAR.VAL values of the parameters used for the construction of weights
- > ROW.SD row standardization (logical)
- > SELF data self-included (logical)
- > NONZERO percentage of non-zero connections
- > NONZEROIND percentage of individuals with non-zero connections
- > AVERAGE average number of connection per individual

**ACCESS TO THE SLOTS** The content of the slots can be accessed with the corresponding accessors, using the generic notation of EcoGenetics (<ecoslot.> + <name of the slot> + <name of the object>). See help("EcoGenetics accessors") and the Examples section below

#### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**Examples**

```

## Not run:

data(eco3)

# 1) "circle" method

con <- eco.weight(eco3[["XY"]], method = "circle", d1 = 0, d2 = 500)

#---- Different plot styles for the graph ----#

# simple
eco.plotWeight(con, type = "simple")

# igraph
eco.plotWeight(con, type = "igraph", group = eco3[["S"]]$structure)

# network (interactive)
## click in a node to see the label
eco.plotWeight(con, type = "network", bounded = TRUE, group = eco3[["S"]]$structure)

# edgebundle (interactive)
## in the following plot, the assignment a group factor,
## generates clustered nodes.
## hover over the nodes to see the individual connections
eco.plotWeight(con, type = "edgebundle", fontSize=8, group = eco3[["S"]]$structure)

# 2) "knearest" method

con <- eco.weight(eco3[["XY"]], method = "knearest", k = 10)
eco.plotWeight(con)
eco.plotWeight(con, type = "network", bounded = TRUE, group = eco3[["S"]]$structure)

# 3) "inverse" method
## scale dependent. In the example, the original coordinates (in km) are converted into m
con <- eco.weight(eco3[["XY"]]/1000, method = "inverse", max.sd = TRUE, p = 0.1)
con
eco.plotWeight(con)

# 4) "circle.inverse" method
con <- eco.weight(eco3[["XY"]], method = "circle.inverse", d2 = 1000)
con
eco.plotWeight(con)

# 5) "exponential" method
## scale dependent. In the example, the original coordinates (in km) are converted into m
con <- eco.weight(eco3[["XY"]]/1000, method = "exponential", max.sd = TRUE, alpha = 0.1)
eco.plotWeight(con)

```

```

# 6) "circle.exponential" method
con <- eco.weight(eco3[["XY"]], method = "circle.exponential", d2 = 2000)
con
eco.plotWeight(con)

# 7) CUSTOM WEIGHT MATRIX

## A eco.weight object can be created with a custom W matrix. In this case,
## the rows and the columns of W (weight matrix) must have names,
## that must coincide (also in order) with the name of the XY (position) matrix.

require(igraph)
## this example generates a network with the package igraph
tr <- make_tree(40, children = 3, mode = "undirected")
plot(tr, vertex.size = 10, vertex.label = NA)

## conversion from igraph to weight matrix
weights <- as.matrix(as_adj(tr))

## weight matrix requires named rows and columns
myNames <- 1:nrow(weights)
rownames(weights) <- colnames(weights) <- myNames

## extract coordinates from the igraph object
coord <- layout.auto(tr)
rownames(coord) <- myNames
plot(layout.auto(tr))

## custom weight object
customw <- eco.weight(XY = coord, W = weights)

## simple plot of the object
eco.plotWeight(customw, type = "simple")

## create a vector with groups to have coloured plots
myColors <- c(rep(1,13), rep(2, 9), rep(3, 9), rep(4, 9))

eco.plotWeight(customw, type = "igraph",group = myColors)

## in the following plot, the argument bounded is set to FALSE,
## but if you have many groups, it probably should be set to TRUE.
# click in a node to see the label
eco.plotWeight(customw,type = "network", bounded = FALSE, group = myColors)

## in the following plot, the assignment a group factor,
# generates clustered nodes.
# hover over the name of the nodes to see the individual connections
eco.plotWeight(customw, type = "edgebundle", group = myColors)

#### CONVERSION FROM LISTW OBJECTS ####
require(adeget)
```

```

# Delaunay triangulation
temp <-chooseCN(eco3[["XY"]], type = 1, result.type = "listw", plot.nb = FALSE)
con <- eco.listw2ew(temp)
eco.plotWeight(con, "network", bounded = TRUE, group = eco3[["S"]]$structure)

#-----
# ACCESSORS USE EXAMPLE
#-----

# the slots are accessed with the generic format
# (ecoslot. + name of the slot + name of the object).
# See help("EcoGenetics accessors")

ecoslot.METHOD(con)      # slot METHOD
ecoslot.PAR(con)           # slot PAR
ecoslot.PAR.VAL(con)       # slot PAR.VAL

## End(Not run)

```

---

eco2

*Eco2*

---

### Description

ecogen object with simulated data of 900 individuals.

### Usage

```
data(eco2)
eco2
```

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

eco3

*Eco3*

---

### Description

ecogen object with simulated data of 173 individuals.

### Usage

```
data(eco3)
eco3
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

ecogen

*Creating a new ecogen object*

---

**Description**

Creating a new ecogen object

**Usage**

```
ecogen(XY = data.frame(), P = data.frame(), G = data.frame(),
       E = data.frame(), S = data.frame(), C = data.frame(),
       G.processed = TRUE, order.G = FALSE, type = c("codominant", "dominant"),
       ploidy = 2, sep = "", ncod = NULL, missing = c("0", "NA", "MEAN"),
       NA.char = "NA", poly.level = 5, rm.empty.ind = FALSE, order.df = TRUE,
       set.names = NULL, valid.names = FALSE)
```

**Arguments**

XY	Data frame with m columns (coordinates) and n rows (individuals).
P	Data frame with n rows (individuals), and phenotypic data in columns.
G	Data of class: "data.frame", with individuals in rows and genotypic data in columns (loci). The ploidy and the type (codominant, dominant) of the data, must be passed with the arguments "ploidy" and "type". Missing data is coded as NA. Dominant data must be coded with binary values (0 for absence - 1 for presence).
E	Data frame with n rows (individuals), and environmental data in columns.
S	Data frame with n rows (individuals), and groups (factors) in columns. The program converts non factor data into factor.
C	data frame with n rows (individuals), and custom variables in columns.
G.processed	If TRUE, the slot G will include a processed data frame ( removed non informative loci (the data non available for all the individuals), removed non polymorphic loci (for dominant data) and ordered alleles in ascending order.
order.G	Genotypes must be ordered in G slot? (codominant data) Default FALSE.
type	Marker type: "codominant" or "dominant".
ploidy	Ploidy of the G data frame. Default ploidy = 2.
sep	Character separating alleles (codominant data). Default option is no character separating alleles.
ncod	Number of characters coding each allele (codominant data).

missing	Missing data treatment ("0", "NA", or "MEAN") for the A slot. Missing elements are set to 0 in the default option. missing elements are recoded as "NA" or the mean allelic frequency across individuals in "NA" and "MEAN" options, respectively.
NA.char	Character simbolizing missing data in the input. Default is "NA".
poly.level	Polymorphism threshold in percentage (0 - 100), for remotion of non polymorphic loci (for dominant data). Default is 5 (5%).
rm.empty.ind	Remotion of noinformtive individuals (row of "NAs"). Default if FALSE.
order.df	Order data frames rows (all data frames with a same row names order). This option works when the names of the data frames are used (i.e., set.names and valid.names are NULL), otherwise order.df = TRUE/FALSE has no effect in the function. Defalut TRUE. If FALSE, row names of all data frames must be ordered. The use of data frames with row names in different order will return an error. In both cases, the program sets an internal names attribute of the object as the row names of the first non-empty data frame found in the following order: XY, P, G, E, S, C. This attribute is used as reference to order rows when order.df = TRUE.
set.names	Character vector with names for the rows of the non-empty data frames. This argument is incompatible with valid.names
valid.names	Logical. Create valid row names? This argument is incompatible with set.names. The program will name individuals with valid tags I.1, I.2, etc.

## Details

This is a generic function for creating an ecogen object. In default option, the missing data input value is "NA", but any missing data character can be passed with the option NA.char. The output in the slot G will have coded the missing data as NA. For dominant markers, the slot A is unnecessary and is treated in ecogen methods as a symbolic link to G.

### ACCESS TO THE SLOTS. MODIFICATION OF ECOGEN OBJECTS

The content of the slots can be extracted with the corresponding accesors ecoslot.XY, ecoslot.P, ecoslot.G, ecoslot.A, ecoslot.E, ecoslot.C and ecoslot.OUT. Data can be assigned individually to the slots, also with the corresponding accessors. The correct use of ecogen objects requires the implementation of accessors, as they ensure the data check and pre-processing. The use of accessors enables to modify or fill the slots of ecogen objects, without the need of creating a new object each time. See *help("EcoGenetics accessors")* for a detailed description and examples about ecogen accessors.

### OTHER SLOT ACCESS METHODS FOR ECOGEN OBJECTS

The use of brackets is defined for ecogen objects:

- Single bracket: the single bracket ("[") is used to subset all the ecogen data frames (P, G, E, S, A and C) by row, at once. The notation for an object is `eco[from:to]`, where `eco` is any ecogen object, and `from: to` is the row subset. For example: `eco[1:10]` , subsets the object `eco` from row 1 to row 10, for all the data frames at once.
- Double square brackets: the double square brackets are symbolic abbreviations of the accessors (i.e., it is a call to the corresponding accessor). The usage is: `eco[["X"]]`, where X is any slot of interest: `eco[["P"]]`, `eco[["G"]]`, `eco[["A"]]`, `eco[["E"]]`, `eco[["S"]]`, `eco[["C"]]` and `eco[["OUT"]]`.



Double square brackets can be used in get/set mode. See Examples below and in help("EcoGenetics accessors").

### ABOUT THE CONSTRUCTION OF NEW ECOGEN OBJECTS

The construction of a new ecogen object can be made in two different ways. First, a new object can be created, incorporating all the information at once. Second, the data can be added in each slot, using the corresponding accessor / "[[". The accessor/double square brackets methods allow temporal modification of any created ecogen object and ensures its modularity. These methods are not only functions to get/assign values to the slots, they provide a basic pre-processing of the data during assignment, generating a set of coherent information.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:

#Example with G data of class "data.frame", corresponding to
#microsatellites of a diploid organism:
data(eco.test)
eco <- ecogen(XY = coordinates, P = phenotype, G = genotype,
E = environment, S = structure)

#Example with G data of class "data.frame", corresponding to a
#presence - absence molecular marker:
dat <- sample(c(0,1),100,rep = TRUE)
dat <- data.frame(matrix(dat,10,10))
eco <- ecogen(G = dat, type = "dominant")

# DINAMIC ASSIGNMENT WITH ACCESSORS AND "[["

eco <- ecogen(XY = coordinates, P = phenotype)
eco

ecoslot.G(eco, order.G = TRUE) <- genotype

# this is identical to
eco[["G", order.G=TRUE]] <- genotype

ecoslot.E(eco) <- environment

# this is identical to
eco[["E"]] <- environment

#-----
# See additional examples in help("EcoGenetics accessors")
#-----

# Storing data in the slot OUT
```

```

singers <- c("carlos_gardel", "billie_holiday")

ecoslot.OUT(eco) <- singers

# Storing several data

golden.number <- (sqrt(5) + 1) / 2
ecoslot.OUT(eco) <- list(singers, golden.number) # several objects must be passed as a list

# this is identical to:

eco[["OUT"]] <- list(singers, golden.number)

## End(Not run)

```

---

ecogen2geneland

*Creating input data for Geneland with an ecogen object*


---

### Description

This function creates four data frames (XY.txt, NAMES.txt, P.txt, G.txt) in the indicated directory (default: working directory), which can be loaded in Geneland.

### Usage

```
ecogen2geneland(eco, dir = "", ncod = NULL, ploidy = 2)
```

### Arguments

eco	Object of class "ecogen"
dir	output path. Default = "" (current directory).
ncod	Number of digits coding each allele (e.g., 1: x, 2: xx, 3: xxx, etc.).
ploidy	Ploidy of the data.

### Value

XY.txt Matrix with coordinates.  
 NAMES.txt Matrix with row names.  
 P.txt Matrix with phenotypic data.  
 G.txt Matrix with genotypic data.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## Examples

```
## Not run:  
  
data(eco.test)  
ecogen2geneland(eco, dir = "", ncod=1)  
  
## End(Not run)
```

---

ecogen2genepop

*Exporting an ecogen genetic data frame into Genepop format*

---

## Description

This function converts the genetic data of an ecogen object into a Genepop input file.

## Usage

```
ecogen2genepop(eco, dir = "", outName = "infile.genepop.txt", grp = NULL,  
               nout = 3, sep = "")
```

## Arguments

eco	Object of class "ecogen".
dir	output path. Default = "" (current directory).
outName	The name of the output file.
grp	The name of the S slot column with groups in which the sample must be divided (e.g., populations). If groups are not given (grp = NULL), all individuals will be assigned to a single one.
nout	Number of digits in the output file
sep	Character separating alleles.

## Value

A Genepop file in the working directory.

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## Examples

```
## Not run:

data(eco.test)
ecogen2genepop(eco, dir = "", outName = "infile.genepop.txt", grp = "pop")
# an output file "infile.genepop.txt" is generated in the working directory

## End(Not run)
```

---

ecogen2genind

*Conversion form ecogen to genind and genind to ecogen*

---

## Description

These functions export from ecogen to genind and viceversa  
genind2ecogen

## Usage

```
ecogen2genind(from)
```

```
genind2ecogen(from)
```

## Arguments

from            Object of class "ecogen" / "genind"

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## Examples

```
## Not run:

data(eco.test)

# ecogen to genind
outGenind <- ecogen2genind(eco)
outGenind

# genind to ecogen
outEco <- genind2ecogen(outGenind)
```

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

---

ecogen2gstudio      *Conversion from ecogen to gstudio and gstudio to ecogen*

---

### Description

These functions converts the genetic data of an ecogen object in a gstudio data frame or viceversa.  
Conversion from gstudio to ecogen

### Usage

```
ecogen2gstudio(from, type = c("codominant", "dominant"))

gstudio2ecogen(from, ID = "ID", lat = "Latitude", lon = "Longitude",
               struct = NULL)
```

### Arguments

from	Input object of class "ecogen" or "gstudio" (depending the direction of conversion)
type	The type of data: "codominant" (for codominant data); "dominant" for presence - absence data.
ID	name of the column with ID (default "ID")
lat	name of the column with latitude (default "Latitude")
lon	name of the column with longitude (default "Longitude")
struct	vector with name of the columns with structures (default NULL)

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:

data(eco.test)
togstudio <- ecogen2gstudio(eco, type = "codominant")
togstudio
toeco <- gstudio2ecogen(togstudio, ID = "ID", lat = "Latitude",
                       lon = "Longitude", ID = "ID", struct = "pop")
toeco
# as ID, Latitude and Longitude are column names in the <togstudio> data frame
# (that match default parameter values for gstudio2ecogen),
# the latter is identical to this:
```

```
toeco <- gstudio2ecogen(togstudio, struct = "pop")
toeco

## End(Not run)
```

---

ecogen2hierfstat	<i>Converting an ecogen genetic data frame into a hierfstat data frame</i>
------------------	--

---

### Description

This function converts the genetic data of an ecogen object in a hierfstat data frame.

### Usage

```
ecogen2hierfstat(eco, pop = NULL)
```

### Arguments

eco	Object of class "ecogen".
pop	The name of the S slot column with the groups for the hierfstat data frame.

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:

data(eco.test)
hiereco <- ecogen2hierfstat(eco, "pop")
require("hierfstat")
basic.stats(hiereco)

## End(Not run)
```

ecogen2spagedi

*Exporting an ecogen genetic data frame into SPAGeDi format***Description**

This function converts the genetic data of an ecogen object in a SPAGeDi input file. When distance classes are required, they can be constructed by combining the parameters "int", "smin", "smax", "nclass", "seqvec" and "size", as described in the function [eco.lagweight](#). A distance matrix can also be included using the "distmat" parameter. Missing data must be coded as a single "NA" in the G data frame.

**Usage**

```
ecogen2spagedi(eco, pop = NULL, ndig, dir = "",
  outName = "infile.spagedi.txt", smin = 0, smax = NULL, int = NULL,
  nclass = NULL, seqvec = NULL, size = NULL, bin = c("sturges", "FD"),
  distmat = NULL, latlon = FALSE)
```

**Arguments**

eco	Object of class "ecogen".
pop	The name of the S slot column with the groups for the output data. The default option includes all the individuals into a single group.
ndig	Number of digits coding each allele (e.g., 1: x, 2: xx, or 3: xxx).
dir	output path. Default = "" (current directory).
outName	The name of the output file.
smin	Minimum class distance in the units of the XY slot data.
smax	Maximum class distance in the units of the XY slot data.
int	Distance interval in the units of the XY slot data.
nclass	Number of classes.
seqvec	Vector with breaks in the units of the XY slot data.
size	Number of individuals per class.
bin	Rule for constructing intervals when a partition parameter (int, nclass or size) is not given. Default is Sturge's rule (Sturges, 1926). Other option is Freedman-Diaconis method (Freedman and Diaconis, 1981).
distmat	Distance matrix to include (optional).
latlon	Are the coordinates in decimal degrees format? Defalut FALSE. If TRUE, the coordinates must be in a matrix/data frame with the longitude in the first column and latitude in the second. The position is projected onto a plane in meters with the function <a href="#">geoXY</a> .

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

## References

Freedman D., and P. Diaconis. 1981. On the histogram as a density estimator: L 2 theory. Probability theory and related fields, 57: 453-476.

Hardy O. and X Vekemans. 2002. SPAGeDi: a versatile computer program to analyse spatial genetic structure at the individual or population levels. Molecular ecology notes, 2: 18-620.

Sturges H. 1926. The choice of a class interval. Journal of the American Statistical Association, 21: 65-66.

## Examples

```
## Not run:  
  
data(eco.test)  
ecogen2spagedi(eco, dir = "", pop = "pop", ndig = 1,int=2, smax=6, outName="infile.spagedi.txt")  
  
## End(Not run)
```

---

environment

*environment*

---

## Description

Data frame with simulated environmental variables of 225 individuals.

## Usage

```
data(eco.test)  
environment
```

## Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>



---

genepop2ecogen	<i>Importing a Genepop file</i>
----------------	---------------------------------

---

**Description**

This function converts a Genepop file into an object with a genetic matrix (G) and a structures matrix (S).

**Usage**

```
genepop2ecogen(genefile = NULL)
```

**Arguments**

genefile      Genepop file.

**Value**

A list with the objects G (genetic matrix) and S (structures matrix).

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>, adapting code written by Emiel van Loon and Scott Davis

**Examples**

```
## Not run:  
# ingpop, file with Genepop format in the folder "/extdata" of the package  
  
ecopath <- paste(path.package("EcoGenetics"), "/extdata/ingpop", sep = "  
ingpop <- genepop2ecogen(ecopath)  
ingpop  
  
## End(Not run)
```

---

genotype	<i>genotype</i>
----------	-----------------

---

**Description**

Data frame with simulated microsatellite data of 225 individuals.

**Usage**

```
data(eco.test)  
genotype
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

grf.multiplot      *Multiple plot function for ggplot*

---

**Description**

Multiple plot function for ggplot

**Usage**

```
grf.multiplot(..., plotlist = NULL, cols = 1, layout = NULL)
```

**Arguments**

...	ggplot objects
plotlist	List of ggplot object
cols	Number of columns in layout
layout	A matrix specifying the layout. If present, 'cols' is ignored.

**Author(s)**

Hadley Wickham

---

grf.seqmultiplot      *Plot a ggplot sequence in layers of n plots arranged in k rows*

---

**Description**

Plot a ggplot sequence in layers of n plots arranged in k rows

**Usage**

```
grf.seqmultiplot(x, n, nrow, byrow = TRUE)
```

**Arguments**

x	list of ggplot objects
n	number of plot in layout
nrow	Number of rows in layout
byrow	plot by row?

**Author(s)**

Leandro Roser

---

phenotype	<i>phenotype</i>
-----------	------------------

---

**Description**

Data frame with simulated morphometric data of 225 individuals.

**Usage**

```
data(eco.test)
phenotype
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

plot,eco.multilsa,ANY-method	
	<i>plot eco.multilsa</i>

---

**Description**

Plot method for local spatial analysis

**Usage**

```
## S4 method for signature 'eco.multilsa,ANY'
plot(x)
```

**Arguments**

x                   eco.multilsa object

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**

[eco.lsa](#)

---

spagedi2ecogen      *Importing a SPAGeDi file, via conversion to ecogen*

---

### Description

This function converts a SPAGeDi file into a ecogen object

### Usage

```
spagedi2ecogen(infile, sep = "", missCode = NULL, type = c("codominant",  
"dominant"), ...)
```

### Arguments

infile	Path to the SPAGeDi file.
sep	Character separating alleles (codominant data). Default option is no character separating alleles.
missCode	characters to represent missing genotypes in codominant markers. If NULL, is computed as "0" times the numer of characters coding alleles.
type	Marker type: "codominant" or "dominant".
...	additional arguments passed to ecogen

### Author(s)

Leandro Roser <leandroroser@ege.fcen.uba.ar>

### Examples

```
## Not run:  
  
data(eco.test)  
ecogen2spagedi(eco, dir = "", pop = "pop", ndig = 1, int=2, smax=6, outName="infile.spagedi.txt")  
spagedi2ecogen("infile.spagedi.txt", sep = "")  
  
## End(Not run)
```

---

structure	<i>structure</i>
-----------	------------------

---

**Description**

Factor with simulated groups of 225 individuals.

**Usage**

```
data(eco.test)
structure
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

summary, eco.mlm-method	<i>Summary for eco.lmtree output</i>
-------------------------	--------------------------------------

---

**Description**

Summary for eco.lmtree output

**Usage**

```
## S4 method for signature 'eco.mlm'
summary(object)

## S4 method for signature 'eco.mctree'
summary(object)
```

**Arguments**

object            Output object of [eco.lmtree](#).

**Value**

A Table with a summary of the analysis for "mlm" analysis, the plot of the trees with significant splits for "mctree" analysis.

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

**See Also**[eco.lmtree](#)**Examples**

```
## Not run:

data(eco.test)
#' mod <- eco.lmtree(DF1 = eco$P, DF2 = eco$E,
analysis = "mlm")
summary(mod)                                #summary for "mlm" analysis

mod <- eco.lmtree(DF1 = eco$P, DF2 = eco$E,
analysis = "mctree", fact = eco$$structure)
summary(mod)                                #summary for "mctree" analysis

## End(Not run)
```

---

tab

*tab*

---

**Description**

Data frame with information of bands 3 and 4 for two dates, corresponding to real Landsat 5 images and used in this package as pedagogic material complementing simulated data. Date and sun elevation data were extracted from the header provided with the image in <http://glovis.usgs.gov/>. The starting haze values (SHV) were estimated checking the profiles of the bands.

**Usage**

```
data(tab)
tab
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

---

<code>table.sokal</code>	<i>table.sokal</i>
--------------------------	--------------------

---

**Description**

Allelic frequency table from 50 villages, analyzed in Sokal et al. (1986).

**Usage**

```
data(sokal1986)
table.sokal
```

**Author(s)**

Leandro Roser <leandroroser@ege.fcen.uba.ar>

# Index

## \*Topic **data**

- coordinates, 23
- eco, 24
- eco2, 102
- eco3, 102
- environment, 112
- genotype, 113
- phenotype, 115
- structure, 117
- tab, 118
- table.sokal, 119

## \*Topic **export**

- aue.dataAngle, 18

- accessors, 4, 6
- accessors), 6
- aue.dataAngle, 18
- aue.df2image, 19
- aue.image2df, 20
- aue.phenosimil, 20
- aue.rescale, 21
- aue.sort, 4, 22, 47

- binding by column-method, 5

- binding by row-method, 5

- calc, 74
- chisq.test, 25
- cleared, 7
- coordinates, 23
- cor, 32, 69
- cor.test, 35
- correlograms, 4
- ctree, 55

- dataType, 72, 74

- detrending spatial data utility, 4

- eco, 24
- eco.alfreq, 4, 24
- eco.association, 25

- eco.bearing, 26, 68
- eco.cbind, 28
- eco.clear, 29
- eco.convert, 4, 29
- eco.cormantel, 31, 77, 78
- eco.correlog, 34, 77–79
- eco.detrend, 40
- eco.forestplot, 42
- eco.forestplot,dataframeORmatrix-method  
(eco.forestplot), 42
- eco.forestplot,eco.lsa-method  
(eco.forestplot), 42
- eco.forestplot,generic-method  
(eco.forestplot), 42
- eco.format, 4, 43
- eco.formula, 45
- eco.gsa, 47, 79
- eco.kin.loiselle, 50
- eco.lagweight, 34, 51, 111
- eco.listw2ew, 54
- eco.lmtree, 55, 117, 118
- eco.lsa, 57, 80, 84, 87, 88, 115
- eco.malecot, 63
- eco.mantel, 68
- eco.merge, 71
- eco.NDVI, 4, 72, 73, 74
- eco.NDVI.post, 4, 73
- eco.order, 75
- eco.pairtest, 4, 76
- eco.plotCorrelog, 77
- eco.plotCorrelogB, 78
- eco.plotGlobal, 79
- eco.plotLocal, 80
- eco.plotWeight, 81
- eco.post.geneland, 4, 82
- eco.rankplot, 83
- eco.rankplot,eco.lsa,missing,missing-method  
(eco.rankplot), 83
- eco.rankplot,eco.lsa-method



- (eco.rankplot), 83
- eco.rankplot, factor, dataframeORmatrix, missing method, 83
- eco.rankplot, factor-method (eco.rankplot), 83
- eco.rankplot, numeric, dataframeORmatrix, missing method, 83
- eco.rankplot, numeric-method (eco.rankplot), 83
- eco.rasterplot, 85
- eco.rasterplot, eco.multilsa-method, 87
- eco.rbind, 88
- eco.remove, 89
- eco.slide.con, 90
- eco.slide.matrix, 91
- eco.split, 92
- eco.subset, 93
- eco.theilsen, 4, 94
- eco.variogram, 77, 78, 95
- eco.weight, 47, 57, 81, 97
- eco2, 102
- eco3, 102
- ecogen, 4, 5, 92, 93, 103
- ecogen2geneland, 106
- ecogen2genepop, 107
- ecogen2genind, 108
- ecogen2gstudio, 109
- ecogen2hierfstat, 110
- ecogen2spagedi, 111
- EcoGenetics (EcoGenetics-package), 4
- EcoGenetics-package, 4
- environment, 112
- fisher.test, 25
- forceNetwork, 81, 82
- forestplot, 4
- genepop, 4
- genepop2ecogen, 113
- genind2ecogen (ecogen2genind), 108
- genotype, 113
- geoXY, 19, 26, 32, 35, 40, 41, 52, 64, 96, 98, 111
- Getis-Ord's  $G^*$ , 4
- grf.multiplot, 114
- grf.seqmultiplot, 114
- gstudio2ecogen (ecogen2gstudio), 109
- importer, 4
- lm, 55
- local Moran's I, 4
- local spatial analysis, 4
- Mantel test, 4
- missing method, 5
- Moran's I, 4
- multiple lm, 4
- order, 22
- p.adjust, 25, 32, 35, 47, 58, 64, 76, 94
- phenotype, 115
- plot, eco.multilsa, ANY-method, 115
- plot, eco.multilsa-method (plot, eco.multilsa, ANY-method), 115
- plot.igraph, 81, 82
- rankplot, 4
- rasterplot, 4
- RDS, 7
- removing-method, 7
- rkt, 94
- S4, 4
- SPAGeDi, 4
- spagedi2ecogen, 116
- spatial weights matrices, 4
- spatial weights matrix, 4
- step, 55
- structure, 117
- subsetting-method, 5
- summary, eco.lmtree-method (summary, eco.mlm-method), 117
- summary, eco.mctree-method (summary, eco.mlm-method), 117
- summary, eco.mlm-method, 117
- tab, 118
- table.sokal, 119
- this link, 4
- TukeyHSD, 76
- variograms, 4
- wilcox.test, 76