

Package ‘parfossil’

February 15, 2012

Type Package

Title Parallelized functions for palaeoecological and palaeogeographical analysis

Version 0.2.0

Date 2010-12-10

Author Matthew Vavrek <matthew@matthewvavrek.com>

Maintainer Matthew Vavrek <matthew@matthewvavrek.com>

Depends fossil, foreach

Description The package provides a number of easily parallelized functions from the fossil package. This package is designed to be used with some type of parallel computing backend, such as multicore, snow or MPI.

License GPL (>= 2)

URL <http://matthewvavrek.com>

LazyLoad yes

Repository CRAN

Date/Publication 2010-12-12 09:11:01

R topics documented:

parfossil-package	2
par.nmfs	2
par.spp.est	4

Index	7
--------------	----------

parfossil-package	<i>Parallelized functions for palaeoecological and palaeogeographical analysis</i>
-------------------	--

Description

This package provides a number of functions from the `fossil` package that have been designed to be run on a parallel backend. The functions show a large speed up, even when just using a dual core versus single core set up, which can be very useful in situations with a large number of resampling replicates.

Details

The package requires some parallel backend to be loaded, using packages such as `multicore`, `Rmpi` or `snow`

Author(s)

Maintainer: Matthew Vavrek <matthew@matthewvavrek.com>

<code>par.nmnds</code>	<i>A parallelized function for estimating species diversity</i>
------------------------	---

Description

Estimate the diversity of a sample(s) using a number of species diversity estimators.

Usage

```
par.nmnds(dmat, mindim = 1, maxdim = 2, nits = 10, iconf = 0, epsilon = 1e-12, maxit = 500, trace = FALSE)
```

Arguments

<code>dmat</code>	Lower triangle distance matrix
<code>mindim</code>	optional, the minimum number of dimensions to use for an analysis; default is 1
<code>maxdim</code>	optional, the maximum number of dimensions to use for an analysis; default is 2
<code>nits</code>	optional, the number of iterations; how many times the data should be initially placed at random; default is 10
<code>iconf</code>	optional, initial configuration. If not specified, then a random configuration is used.
<code>epsilon</code>	optional, acceptable difference in stress.
<code>maxit</code>	optional, maximum number of iterations.
<code>trace</code>	if TRUE, will write progress indicator to the screen.

Details

Non-Metric Multidimensional Scaling (NMDS) is designed to find an optimal arrangement for a set of points in a reduced dimensional space.

Value

conf : list of configurations.

stress : list of final stress values.

r2 : total variance explained by each configuration. The first results are for the lowest number of dimensions (total number is (mindim - maxdim + 1) * nits).

Note

This is slight modification of the nmnds function found in the ecodist package, that has been changed to allow for parallelization of runs. This version of the function uses the foreach() function to parallelize the resampling loop, so any backend that can be used with that package can be used to enable the parallel processing with this package.

Author(s)

Sarah Goslee with modifications from Matthew Vavrek

References

The original nmnds function used as the basis for this parallelized version comes from the ecodist package.

Goslee, S.C., Urban, D.L. 2007. The ecodist Package for Dissimilarity-based Analysis of Ecological Data. Journal of Statistical Software. 22(7):1-19.

See Also

[ecol.dist](#)

Examples

```
## Not run:
#comparison of run times between the serial and parallel versions on the estimator
#please note that this example is designed for a multicore OS X or Linux computer
library(doMC)
registerDoMC()
data(fdata.mat)
par.nmnds(ecol.dist(fdata.mat))

## End(Not run)
```

 par.spp.est

A parallelized function for estimating species diversity

Description

Estimate the diversity of a sample(s) using a number of species diversity estimators.

Usage

```
par.spp.est(x, rand = 10, abund = TRUE, counter = FALSE)
```

Arguments

x	A vector, matrix or data frame with species as rows and locations/samples as columns
rand	The number of times to run the internal randomizations; default is set to 10
abund	If the data is abundance or presence/absence; default is set to TRUE for abundance
counter	Whether or not to provide a running total of progress of randomizations

Details

This function will accept a vector, matrix or data frame of species by samples and return a large matrix with various species estimation values.

Value

Returns a table with the following column names if abund=TRUE:

N.obs	Total sample size
S.obs	Number of observed species
S.obs(+95%)	95% upper confidence interval
S.obs(-95%)	95% lower confidence interval
Chao1	Chao Species Estimation
Chao1(upper)	95% upper confidence interval
Chao1(lower)	95% lower confidence interval
ACE	Abundance-based Coverage Estimator
ACE(upper)	95% upper confidence interval
ACE(lower)	95% lower confidence interval
Jack1	First Order Jackknife Estimator
Jack1(upper)	95% upper confidence interval
Jack1(lower)	95% lower confidence interval

Returns a table with the following column names if abund=FALSE:

N.obs	Total sample size
S.obs	Number of observed species
S.obs(+95%)	95% upper confidence interval
S.obs(-95%)	95% lower confidence interval
Chao2	Chao Species Estimation
Chao2(upper)	95% upper confidence interval
Chao2(lower)	95% lower confidence interval
ICE	Incidence-based Coverage Estimator
ICE(upper)	95% upper confidence interval
ICE(lower)	95% lower confidence interval
Jack1	First Order Jackknife Estimator
Jack1(upper)	95% upper confidence interval
Jack1(lower)	95% lower confidence interval

Note

This function can be very long to run due to its iterative nature, even when it is running in parallel. The randomizations are initially set to 10 so the process will run relatively quickly, but a low value for randomizations will not give nicely smoothed curves. Also, in some cases due to the nature of some of the functions, they provide no answer, such as is common with the Chao standard deviation. In this case, the Chao upper and lower bounds are simply 95% confidence intervals based on the actual Chao estimator.

This version of the function uses the `foreach()` function to parallelize the resampling loop, so any backend that can be used with that package can be used to enable the parallel processing with this package.

Author(s)

Matthew Vavrek

References

The original idea for a program similar to this came from the extremely useful EstimateS program by Robert K. Colwell

Colwell, R.K. 2010. EstimateS: Statistical estimation of species richness and shared species from samples. Version 8.2. User's Guide and application published at: <http://purl.oclc.org/estimates>.

See Also

[chao1](#), [jack1](#), [bootstrap](#)

Examples

```
## Not run:
#comparison of run times between the serial and parallel versions on the estimator
#please note that this example is designed for a multicore OS X or Linux computer
library(doMC)
registerDoMC()
data(fdata.mat)
system.time({spp.est(fdata.mat, rand = 100, abund = TRUE, counter = FALSE)})
system.time({par.spp.est(fdata.mat, rand = 100, abund = TRUE, counter = FALSE)})

#this example is for a multicore Windows computer, but HAS NOT BEEN TESTED
library(doSNOW)
library(snow)
cl <- makeCluster(c("localhost","localhost"), type = "SOCK")
registerDoSNOW(cl)
data(fdata.mat)
system.time({spp.est(fdata.mat, rand = 100, abund = TRUE, counter = FALSE)})
system.time({par.spp.est(fdata.mat, rand = 100, abund = TRUE, counter = FALSE)})
stopCluster(cl)

## End(Not run)
```

Index

*Topic **package**

parfossil-package, 2

bootstrap, 5

chao1, 5

ecol.dist, 3

jack1, 5

par.nmids, 2

par.spp.est, 4

parfossil (parfossil-package), 2

parfossil-package, 2