Package: temporalEF (via r-universe)

August 25, 2024

Type Package

Title Temporal Eigenfunctions

Version 0.1-3

Date 2012-10-17

Depends R (>= 3.1.0)

Imports vegan, permute (>= 0.7-8)

Suggests analogue, testthat

Author Gavin L. Simpson

Maintainer Gavin L. Simpson <ucfagls@gmail.com>

Description Build sets of (asymmetric) one-dimensional, temporal eigenfunctions. The set of eigenfunctions are orthogonal patterns of temporal variation at a range of scales. Two types of temporal eigenfunctions are implemented; i) asymmetric temporal eigenfunctions (ATEs), based on the Asymmetric Eigenvector Map (AEM) method of Blanchet et al (2008), and ii) principal coordinates of temporal neighbour matrices (PCTNs), based on the concept of principal coordinates of neighbour matrices.

License GPL-2

RoxygenNote 6.0.1

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

RemoteUrl https://github.com/gavinsimpson/temporalEF

RemoteRef HEAD

RemoteSha fc0264581609e1b0ab905036df6437e63986cc9b

Contents

`ate` *Asymmetric temporal eigenfunctions*

Description

Generate a set of asymmetric temporal eigenfunctions

Usage

```
ate(x, ...)

## Default S3 method:
ate(x, N, weight = FALSE, FUN = NULL, link0 = TRUE, ...)

## S3 method for class 'ate'
print(x, digits = 3, ...)

## S3 method for class 'ate'
scores(x, choices, ...)

## S3 method for class 'ate'
eigenvals(x, ...)
```

Arguments

`x`	an R object. For `ate` currently only a sorted vector of time points. For the `print` method an object of class `"ate"`.
`...`	additional arguments passed to `FUN`.
`N`	numeric; the number of eigenvectors to return. If not supplied, N is taken from the appropriate dimension of `x`; for the default method, this is the length of `x`.
`weight`	logical; should a weighting matrix be generated and applied to the link matrix?
`FUN`	a function to be applied to the weighting matrix. Ignored if `weight` is `FALSE`.
`link0`	logical; should the link from t[0] be included?
`digits`	numeric; number of digits to display in output.
`choices`	numeric; vector indicating which eigenfunctions to return.

Details

The asymmetric eigenvector map (AEM) is a recently proposed method for describing orthongonal spatial functions for use in multivariate ordination. AEMs are asymmetric because they apply a directionality to the spatial dependencies modelled by the eigenfunctions. Asymmetric temporal eigenfunctions (ATEs) implement the AEM idea to model patterns of temporal dependence; i.e. a single spatial dimension.

Author(s)

Gavin L. Simpson

Examples

```
tp <- seq_len(10)
tefs <- ate(tp)
tefs.I <- moranI(tefs)
plot(tefs.I)
```

---

eigenfuns *Extract temporal eigenfunctions*

Description

Simple extractor function to access temporal eigenfunctions from various objects. Currently methods for class(es) ‘"ate"‘, ‘"pctn"‘, and ‘"tef"‘ are provided.

Usage

```
eigenfuns(x, ...)

## S3 method for class 'ate'
eigenfuns(x, take = NULL, ...)

## S3 method for class 'pctn'
eigenfuns(x, take = NULL, ...)

## S3 method for class 'tef'
eigenfuns(x, take = NULL, ...)
```

Arguments

x	the object from which to extract temporal eigenfunctions
...	additional arguments passed to methods
take	numeric; which eigenfunctions should be returned

Value

A data frame with one column per eigenfunction

Author(s)

Gavin L. Simpson

`makeLinks` *Build an asymmetric link matrix*

Description

Creates an asymmetric link matrix from provided time points.

Usage

```
makeLinks(tp, link0 = TRUE, dimnames = TRUE)
```

Arguments

`tp`	numeric vector of sorted time points
`link0`	logical; should the link from t[0] be included?
`dimnames`	logical; should dimnames be attached to the link matrix?

Value

binary matrix of links between time points. The object is a square asymmetric matrix with `length(tp)` rows and columns. Rows represent the `tp` time points, and columns are the network links between timepoints.

Author(s)

Gavin L. Simpson

See Also

`makeWeights` for a matching weight matrix.

Examples

```
tp <- seq_len(10)
makeLinks(tp)
```

makeWeightMat *Diagonal weight matrix for time points*

Description

A diagonal matrix of weights based on the inverse of time duration between time points.

Usage

```
makeWeightMat(tp, FUN = NULL, link0 = TRUE, dimnames = TRUE, ...)
```

Arguments

tp	numeric vector of sorted time points
FUN	a function to apply
link0	logical; should the link from t[0] be included?
dimnames	logical; should dimnames by attached to the matrix?
...	optional arguments passed to FUN

Value

a diagonal matrix of weights with dimensions `length(tp) - 1`

Author(s)

Gavin L. Simpson

Examples

```
tp <- seq_len(10)
makeWeightMat(tp)
```

makeWeights *Temporal weight matrix*

Description

Build a temporal weighting matrix

Usage

```
makeWeights(tp, FUN = NULL, link0 = TRUE, dimnames = TRUE, ...)
```

Arguments

tp	numeric vector of sorted time points
FUN	a function to apply
link0	logical; should the link from t[0] be included?
dimnames	logical; should dimnames by attached to the matrix?
...	optional arguments passed to FUN

Value

a square, symmetric matrix of weights.

Author(s)

Gavin L. Simpson

See Also

makeLinks for a matching link matrix

Examples

```
tp <- seq_len(10)
makeWeights(tp)
```

---

makeWeightVec *Weight vector for time points*

Description

A vector of weights based on the inverse of time duration between time points.

Usage

```
makeWeightVec(tp, FUN = NULL, link0 = TRUE, names = TRUE, ...)
```

Arguments

tp	numeric vector of sorted time points
FUN	a function to apply
link0	logical; should the link from t[0] be included?
names	logical; should names by attached to the vector?
...	optional arguments passed to FUN

Value

a matrix of weights of length length(tp) - 1

Author(s)

Gavin L. Simpson

Examples

```
tp <- seq_len(10)
makeWeightVec(tp)
```

`moranI` *Moran's I for temporal eigenfunctions*

Description

Calculate Moran's I for tempoeral eigenfunctions and assess significant of Moran's I values via a parametric test that assumes asymptotic normality or a permutation test.

Usage

```
moranI(x, ...)

## S3 method for class 'ate'
moranI(x, permute = FALSE, alternative = c("two.sided",
  "greater", "less"), nperm = 999, ...)
```

Arguments

`x`	an R object of class `"ate"`.
`...`	additional arguments passed to methods.
`permute`	logical; test I via a permutation test
`alternative`	character; the type of test to perform. The default is `"two.sided"`, which allows for both positive and negative spatial correlation (values of I). If you anticipate only one of positive or negative spatial association then you can specify `"greater"` or `"less"`, respectively, as the alternative hypothesis.
`nperm`	numeric; number of permutations to perform in permutation test if requested.

Value

Numeric vector of Moran's I statistics

Author(s)

Gavin L. Simpson

pctn *Principal coordinates of temporal neighbours*

Description

Computes the classic PCNM by the principal coordinate analysis of a truncated distance matrix, but for a one-dimensional process.

Usage

```
pctn(x, ...)

## Default S3 method:
pctn(x, threshold, distfun = dist, ...)

## S3 method for class 'pctn'
print(x, digits = 3, ...)

## S3 method for class 'pctn'
scores(x, choices, ...)

## S3 method for class 'pctn'
eigenvals(x, ...)
```

Arguments

x	an R object. For pctn currently only a sorted vector of time points.
...	additional arguments passed to other methods or on to distfun.
threshold	numeric; threshold beyond which the temporal separation of smaples is considered equal. The default if no value is supplied is to find the largest temporal separation between any two points. Separations greater than the threshold are given a notional separation of 4 times threshold.
distfun	function or character string naming a function that will be used to compute the temporal seperation between samples. Defaults to dist for the Euclidean distance. See Details for further information.
digits	numeric; number of digits to display in output.
choices	numeric; vector indicating which eigenfunctions to return.

Details

The default distance coefficient used to compute temporal separation is the Euclidean distance. If you want to use a different coefficient, you can supply a suitable function to argument distfun. This should be a function that returns an object of class "dist" or a square symmetric matrix that can be coerced to one. Arguments can be passed to distfun via ....

Author(s)

Gavin L. Simpson

Examples

```
tp <- seq_len(50)
mod <- pctn(tp)
mod
```

`plot.ate` *Plot asymmetric temporal eigenfunctions*

Description

A multi-panel layout showing the calculated tempoeral eigenfunctions.

Usage

```
## S3 method for class 'ate'
plot(x, pages = 1, ylim, ylab = nams, xlab = "",
  ask = FALSE, ...)
```

Arguments

`x`	an object of class `"ate"`
`pages`	numeric; the number of pages over which to spread the plots of the individual eigenfunction
`ylim`	numeric vector of limits for the y-axis
`xlab`, `ylab`	x and y-axis labels
`ask`	logical; should plotting be paused between pages?
`...`	additional arguments passed to `plot`

Value

A plot on the currently active device

Author(s)

Gavin L. Simpson

See Also

`ate` for creating ATE objects

Examples

```
tp <- seq_len(50)
mod <- ate(tp)
plot(mod, pages = 2)
```

`plot.moranI` *Plot Moran's I statistics for temporal eigenfunctions*

Description

A plot of Moran's I versus time.

Usage

```
## S3 method for class 'moranI'
plot(x, alpha = 0.05, type = "b",
  xlab = "Eigenfunctions", ylab = "Moran's I", pch = 21, bg = "red",
  col = "black", ...)
```

Arguments

`x`	an object of class `moranI`.
`alpha`	numeric; level of significance
`type`	character; the type of plotting to use. See `plot.default`.
`xlab`	the label for the x-axis of the plot.
`ylab`	the label for the y-axis of the plot.
`pch`	the plotting character to use
`bg`	fill colour of points
`col`	border colour of points
`...`	additional arguments passed to `plot`.

Value

A plot on the current device

Author(s)

Gavin L. Simpson

`plot.pctn` *Plot PCTN eigenfunctions*

Description

A multi-panel layout showing the calculated tempoeral eigenfunctions.

Usage

```
## S3 method for class 'pctn'
plot(x, pages = 1, ylim, ylab = nams, xlab = "",
  ask = FALSE, ...)
```

Arguments

`x`	an object of class `"pctn"`
`pages`	numeric; the number of pages over which to spread the plots of the individual eigenfunctions
`ylim`	numeric vector of limits for the y-axis
`xlab`, `ylab`	x and y-axis labels
`ask`	logical; should plotting be paused between pages?
`...`	additional arguments passed to `plot`

Value

A plot on the currently active device

Author(s)

Gavin L. Simpson

See Also

`pctn` for creating PCTN objects

Examples

```
tp <- seq_len(50)
mod <- pctn(tp)
plot(mod, pages = 2)
```

`plot.timelagAnalysis` *A time lag analysis plot.*

Description

A plot of $\sqrt{\text{lag}}$ against compositional dissimilarity.

Usage

```
## S3 method for class 'timelagAnalysis'
plot(x, ...)
```

Arguments

`x`	an object of class `"timelagAnalysis"`, the result of a call to `timelag`.
`...`	additional arguments passed to `plot`.

Details

TODO - smoothers etc.

Value

Invisibly returns its input `x`.

Author(s)

Gavin L. Simpson

Examples

```
## load analogue for Abernethy data set & distance()
if (require("analogue")) {

## Load Abernethy Forest data set
data("abernethy", package = "analogue")
## Load Abernethy Forest data set

## Remove the Depth and Age variables
abernethy2 <- abernethy[, -(37:38)]

## time lag analysis
dij <- as.dist(distance(abernethy2, method = "chord"))
tla <- timelag(dij)
plot(tla, pch = 19)
}
```

tef *Temporal eigenfunction ordination*

Description

Fits an ordination using temporal eigenfunctions as constraints.

Usage

```
tef(x, ...)

## Default S3 method:
tef(x, index, method = c("ate", "aem", "pctn", "pcnm"),
  ordination = c("rda"), ...)
```

Arguments

x	community data matrix or data frame
...	additional arguments passed to other methods. Arguments are also passed to the function generating the temporal eigenfunctions and the constrained ordination function.
index	numeric; the time ordering of the samples from which temporal eigenfunctions will be computed
method	character; which method to use to create the temporal eigenfunctions
ordination	character; the name of the constrained ordination function, from the **vegan** package to use. Only rda is support currently.

Details

TODO

Value

An object of class "tef", a list with the following components:

ordination the fitted constrained ordination.

tefs the computed set of temporal eigenfunctions.

Author(s)

Gavin L. Simpson

`timelag` *Time lag analysis*

Description

Time lag analysis for a multivariate data set

Usage

```
timelag(x, ...)

## S3 method for class 'dist'
timelag(x, ...)
```

Arguments

`x`	an R object. Only objects of class `"dist"` are currently supported.
`...`	additional arguments passed to other methods.

Details

Time lag analysis involves computing the compositional dissimilarity between samples at lag 1, at lag 2, etc, then the least squares slope between $\sqrt{\text{lag}}$ and dissimilarity.

Value

Returns an object of class `"timelagAnalysis"`, a list with components

data data frame; unpacked set of lags (`Lag`) and compositional dissimilarities (`Distance`).

Author(s)

Gavin L. Simpson

Examples

```
## load analogue for Abernethy data set & distance()
if (require("analogue")) {

## Load Abernethy Forest data set
data("abernethy", package = "analogue")
## Load Abernethy Forest data set

## Remove the Depth and Age variables
abernethy2 <- abernethy[, -(37:38)]

## time lag analysis
dij <- as.dist(distance(abernethy2, method = "chord"))
tla <- timelag(dij)
```

```
head(tla[[1]])
}
```

Index