Package: spEDM (via r-universe)

February 10, 2025

Title Spatial Empirical Dynamic Modeling

Version 1.4

Description Inferring causal associations in cross-sectional earth system data through empirical dynamic modeling (EDM), with extensions to convergent cross mapping from Sugihara et al. (2012) <doi:10.1126/science.1227079>, partial cross mapping as outlined in Leng et al. (2020) <doi:10.1038/s41467-020-16238-0>, and cross mapping cardinality as described in Tao et al. (2023)<doi:10.1016/j.fmre.2023.01.007>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

URL https://stscl.github.io/spEDM/, https://github.com/stscl/spEDM

BugReports https://github.com/stscl/spEDM/issues

Depends R (>= 4.1.0)

LinkingTo Rcpp, RcppThread, RcppArmadillo

Imports dplyr, ggplot2, methods, sdsfun (>= 0.7.0), sf, terra

Suggests knitr, Rcpp, RcppThread, RcppArmadillo, rmarkdown, spData

VignetteBuilder knitr

NeedsCompilation yes

Author Wenbo Lv [aut, cre, cph] (<https://orcid.org/0009-0002-6003-3800>)

Maintainer Wenbo Lv <lyu.geosocial@gmail.com>

Repository CRAN

Date/Publication 2025-02-10 14:00:02 UTC

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

Contents

`detectThreads` *detect the number of available threads*

Description

detect the number of available threads

Usage

```
detectThreads()
```

Value

An integer

Examples

```
detectThreads()
```

`embedded` *generate embeddings*

Description

generate embeddings

Usage

```
## S4 method for signature 'sf'
embedded(data, target, E = 3, tau = 1, nb = NULL)

## S4 method for signature 'SpatRaster'
embedded(data, target, E = 3, tau = 1)
```

Arguments

data	The observation data.
target	Name of target variable.
E	(optional) Dimensions of the embedding.
tau	(optional) Step of spatial lags.
nb	(optional) The neighbours list.

Value

A matrix

Examples

```
columbus = sf::read_sf(system.file("shapes/columbus.gpkg", package="spData"))
embedded(columbus,target = "CRIME", E = 3)
```

---

gccm *geographical convergent cross mapping*

Description

geographical convergent cross mapping

Usage

```
## S4 method for signature 'sf'
gccm(
  data,
  cause,
  effect,
  libsizes,
  E = c(3, 3),
  tau = 1,
  k = 4,
  theta = 1,
  algorithm = "simplex",
  pred = NULL,
  nb = NULL,
  threads = detectThreads(),
  bidirectional = TRUE,
  trend.rm = TRUE,
  progressbar = TRUE
)

## S4 method for signature 'SpatRaster'
```

```
gccm(
  data,
  cause,
  effect,
  libsizes,
  E = c(3, 3),
  tau = 1,
  k = 4,
  theta = 1,
  algorithm = "simplex",
  pred = NULL,
  threads = detectThreads(),
  bidirectional = TRUE,
  trend.rm = TRUE,
  progressbar = TRUE
)
```

Arguments

data	The observation data.
cause	Name of causal variable.
effect	Name of effect variable.
libsizes	A vector of library sizes to use.
E	(optional) Dimensions of the embedding.
tau	(optional) Step of spatial lags.
k	(optional) Number of nearest neighbors to use for prediction.
theta	(optional) Weighting parameter for distances, useful when algorithm is smap.
algorithm	(optional) Algorithm used for prediction.
pred	pred (optional) Row numbers(vector) of lattice data or row-column numbers(matrix) of grid data used for predictions.
nb	(optional) The neighbours list.
threads	(optional) Number of threads.
bidirectional	(optional) whether to identify bidirectional causal associations.
trend.rm	(optional) Whether to remove the linear trend.
progressbar	(optional) whether to print the progress bar.

Value

A list.

xmap cross mapping prediction results

varname names of causal and effect variable

bidirectional whether to identify bidirectional causal associations

Examples

```
columbus = sf::read_sf(system.file("shapes/columbus.gpkg", package="spData"))

g = gccm(columbus,"HOVAL","CRIME",libsizes = seq(5,40,5),E = c(6,5))
g
plot(g, ylimits = c(0,0.8))
```

`simplex` *simplex forecasting*

Description

simplex forecasting

Usage

```
## S4 method for signature 'sf'
simplex(
  data,
  target,
  lib,
  pred = lib,
  E = 1:10,
  tau = 1,
  k = 4,
  nb = NULL,
  threads = detectThreads()
)

## S4 method for signature 'SpatRaster'
simplex(
  data,
  target,
  lib,
  pred = lib,
  E = 1:10,
  tau = 1,
  k = 4,
  threads = detectThreads()
)
```

Arguments

`data`	The observation data.
`target`	Name of target variable.

lib	Row numbers(vector) of lattice data or row-column numbers(matrix) of grid data for creating the library from observations.
pred	(optional) Row numbers(vector) of lattice data or row-column numbers(matrix) of grid data used for predictions.
E	(optional) Dimensions of the embedding.
tau	(optional) Step of spatial lags.
k	(optional) Number of nearest neighbors to use for prediction.
nb	(optional) The neighbours list.
threads	(optional) Number of threads.

Value

A matrix

Examples

```
columbus = sf::read_sf(system.file("shapes/columbus.gpkg", package="spData"))

simplex(columbus,target = "CRIME",lib = 1:49)
```

---

smap *smap forecasting*

Description

smap forecasting

Usage

```
## S4 method for signature 'sf'
smap(
  data,
  target,
  lib,
  pred = lib,
  E = 3,
  tau = 1,
  k = 4,
  theta = c(0, 1e-04, 3e-04, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 0.5, 0.75, 1, 1.5, 2, 3,
    4, 6, 8),
  nb = NULL,
  threads = detectThreads()
)

## S4 method for signature 'SpatRaster'
```

```
smap(
  data,
  target,
  lib,
  pred = lib,
  E = 3,
  tau = 1,
  k = 4,
 theta = c(0, 1e-04, 3e-04, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 0.5, 0.75, 1, 1.5, 2, 3,
    4, 6, 8),
  threads = detectThreads()
)
```

Arguments

data	The observation data.
target	Name of target variable.
lib	Row numbers(vector) of lattice data or row-column numbers(matrix) of grid data for creating the library from observations.
pred	(optional) Row numbers(vector) of lattice data or row-column numbers(matrix) of grid data used for predictions.
E	(optional) Dimensions of the embedding.
tau	(optional) Step of spatial lags.
k	(optional) Number of nearest neighbors to use for prediction.
theta	(optional) Weighting parameter for distances
nb	(optional) The neighbours list.
threads	(optional) Number of threads.

Value

A matrix

Examples

```
columbus = sf::read_sf(system.file("shapes/columbus.gpkg", package="spData"))

smap(columbus,target = "INC",lib = 1:49)
```

Index