Package: covatest (via r-universe)

August 26, 2024

Type Package

Title Tests on Properties of Space-Time Covariance Functions

Version 1.2.3

Description Tests on properties of space-time covariance functions. Tests on symmetry, separability and for assessing different forms of non-separability are available. Moreover tests on some classes of covariance functions, such that the classes of product-sum models, Gneiting models and integrated product models have been provided. It is the companion R package to the papers of Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models <doi:10.1007/s00477-017-1472-2> and Cappello, C., De Iaco, S., Posa, D., 2020, covatest: an R package for selecting a class of space-time covariance functions <doi:10.18637/jss.v094.i01>.

Depends R(>= 3.4.0)

Imports utils, stats, graphics, mathjaxr, methods, lubridate, V8, zoo, gstat, sp (>= 0.9-72), spacetime (>= 1.0-0)

Suggests sf

RdMacros mathjaxr

License GPL (>= 2.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Collate 'sepindex.R' 'couples.R' 'blocks.R' 'covablocks.R' 'covastat.R' 'covaprop.R' 'covastatM.R' 'read.STdata.R' 'vv_13.R'

NeedsCompilation no

Author Sandra De Iaco [aut, cre], Claudia Cappello [aut], Donato Posa [aut], Sabrina Maggio [ctb]

Maintainer Sandra De Iaco <sandra.deiaco@unisalento.it>

Date/Publication 2023-07-03 09:50:02 UTC

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

RemoteUrl https://github.com/cran/covatest

RemoteRef HEAD

RemoteSha e9da288113dc40503718ee85fbd506ed4234f428

Contents

`blocks-class` *Class "blocks"*

Description

A class for overlapped blocks of the time series associated with the given spatial points specified through the function `couples`. Thus, it is necessary to execute couples first and then blocks

Usage

```
blocks(lb, ls, matdata, pardata1, pardata2, stpairs)

## S4 method for signature 'blocks'
boxplot(x, i, j, ...)

## S4 method for signature 'blocks'
show(object)

## S4 method for signature 'blocks'
x[i, j, drop = FALSE]

## S4 method for signature 'blocks'
summary(object, i, j)
```

Arguments

lb	integer, length of each block. The number of terms in each block must be greater than 5 and smaller than the quarter part of the length of each time series
ls	integer, number of overlapped data between two consecutive blocks. The number of overlapped terms between two consecutive blocks must in the interval $[0, lb/2]$
matdata	STFDF/STSDF or data.frame, which contains the coordinates of the spatial points, the identification code of the spatial points, the indentification code of the temporal points and the values of the variable, typically output from read.STdata
pardata1	integer, it represents the column in which the spatial ID is stored (if the spatio-temporal data set is given as data.frame) or the number of variables in the STFDF/STSDF (if the data are given as a STFDF/STSDF)
pardata2	integer, it represents the column in which the values of the variable are stored (if the spatio-temporal data set is given as data.frame) or the slot in which the values of the variable of interest are stored (if the data are given as a STFDF/STSDF). Note that for STFDF/STSDF the argument is set, by default, equal to 1 if the number of variables is equal to 1
stpairs	object of class couples, containing the spatial points and the corresponding temporal lags to be analyzed
x	object of class blocks for methods boxplot and extract
i	index specifying the block to be selected. If i=0 all blocks are selected automatically (option available only for boxplot and summary methods)
j	index specifying the spatial point to be selected. If j=0 all spatial points are selected automatically (option available only for boxplot and summary methods)
...	any arguments that will be passed to the panel plotting functions
object	object of class blocks for methods show and summary
drop	logical, the argument is set, by default, equal to FALSE to preserve the structure of the object. It is advisable not to change this option

Details

A message informs the user of the number of blocks extracted

Slots

mat.block matrix of dimension (*lb* x *overall number of blocks*); the columns of this matrix are associated with the different blocks, of length equal to lb, that can be extracted from the time series related to the selected spatial points defined in the slot stpairs of couples, identified as stpairs@sel.staz

array.block array of dimension (*lb* x *number of blocks for each selected spatial points* x *number of spatial points*). In each table of this array, the overlapped blocks for each spatial location are available

sel.staz numeric or character; contains the ID codes of the selected spatial points

Note

- "Error in matdata$[, clvr]$: subscript out of bounds" appears if `pardata2` does not exist in the argument `matdata`
- If "Error in matdata$[, clvr]$" appears, no data for some of the spatial points, specified in `stpairs`, are available. The user has to go back to `couples` and revise the vector of the selected spatial points (`sel.staz` and `sp.couples.in` arguments)
- A stop occurs if more than 75% of consecutive data are missing in the time series, since a large number of missing values do not guarantee the reliability of the tests
- A stop occurs if the length of the time series for each spatial points is less than 29
- A message appears if the length of the time series for each spatial point is greater than 29 and less than 89, since the length of the time series is low and may not guarantee the reliability of the tests
- A stop occurs if more than 80% of consecutive data are missing in one of the blocks, since the estimation of the covariance matrix is not reliable, when a large number of missing values occur
- If, in the last block of each selected spatial point, more than 15% of data are missing a warning message appears, since the estimation of the covariance matrix, when a large number of missing values occurs, is not reliable
- A warning message appears if the number of blocks, computed by fixing `lb` and `ls`, is less than 5. It is convenient that the number of blocks is close to the number of spatio-temporal comparisons defined in `couples`. This avoids singolarity in computing test statistics

References

Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models. Stochastic Environmental Research and Risk Assessment, **32** 17–35

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Palma, M., Posa, D., 2016. A general procedure for selecting a class of fully symmetric space-time covariance functions. Environmentrics, **27(4)** 212–224.

Li, B., Genton, M.G., Sherman, M., 2007, A nonparametric assessment of properties of spacetime covariance functions. Journal of the American Statistical Association, **102** 736–744.

See Also

`couples`

`read.STdata`

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
```

```
data(air)
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #


sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

block.sym <- blocks(lb = 40, ls = 10, matdata = rr_13, pardata1 = 1, pardata2 = 1,
stpairs = couples.sym)

### methods for blocks
#1. show
block.sym

#2. [  extract
block.sym[1,] #select the 1st block of each spatial location
block.sym[,1] #select all blocks of the 1st spatial location
block.sym[1:2, 1:3] #select the first two blocks of the first 3 spatial locations

#3. summary
summary(block.sym, 1:2, 1:3) #to obtain the summary associated to the first
#two blocks of the first 3 spatial locations

summary(block.sym, 0, 1) #to obtain the summary associated to all blocks of
#the 1st spatial location

#4. boxplot
boxplot(block.sym, 1:5, 1:2) #boxplots of the first 5 blocks of associated to
#the first 2 spatial locations

boxplot(block.sym, 0 ,1) #boxplots of all blocks of associated to the 1st
#spatial location
```

couples-class *Class "couples"*

Description

A class for spatial points and the corresponding temporal lags to be analyzed in order to test some covariance properties and some well known classes of space-time covariance functions models

Usage

```
couples(
  sel.staz,
  sp.couples.in,
  t.couples.in,
  typetest = "sym",
  typecode = numeric()
)

## S4 method for signature 'couples'
show(object)

## S4 method for signature 'couples'
x[i, by.row = FALSE, drop = FALSE]

## S4 method for signature 'couples'
summary(object)
```

Arguments

`sel.staz`	vector, the sequence of ID codes which denote the spatial points to be analyzed
`sp.couples.in`	two-column matrix: rows corresponding to the couples of different spatial points, chosen among the ones fixed in `sel.staz` argument, to be compared
`t.couples.in`	vector of only positive (negative) temporal lags to be analyzed. The corresponding negative (positive) temporal lags are included authomatically for `typetest = "sym", "sep", "tnSep"`. If some temporal lags, corresponding to some couples of spatial points, are not required for the specific test, they can be set equal to zero, through the specific `setzero` method
`typetest`	character, set `typetest = "sym"` for symmetry test (default choice), `typetest = "sep"` for separability test, `typetest = "tnSep"` for type of non separability test, `typetest = "productSum"` for the test on the product-sum class of models, `typetest = "intProduct"` for the test on the integrated product class of models, `typetest = "gneiting"` for the test on the Gneiting class of models
`typecode`	type of object, i.e. numeric() or character(), specifies the type of codification of the spatial points in the `data frame` or in the STFDF/STSDF
`object`	object of class `couples` for methods `show` and `summary`

x	object of class couples for method extract
i	index specifying rows or columns of the slot @couples.st. Rows or columns depending on the logical parameter by.row to be set
by.row	logical, if TRUE rows of the slot @couples.st are selected (the temporal lags associated to the i-th spatial couple are given). If FALSE (the default) columns of the slot @couples.st are selected. In particular, the spatial couples associated to the i-th temporal lag (i >= 3, temporal lags are stored from the third column) are given
drop	logical, the argument is set, by default, equal to FALSE to preserve the structure of the object. It is advisable not to change this option

Details

It is important to point out that:

- both positive and negative temporal lags are automatically considered in the slot @couples.st and @tl.couples for symmetry test (typetest = "sym"), separability test (typetest = "sep") and type of non separability tests (typetest = "tnSep"). If the symmetry hyphotesis has not been rejected, only positive temporal lags might be considered for the test on separability and type of non separability (typetest = "sep" and typetest = "tnSep"), hence the specific setzero method must be used to set the negative temporal lags equal to zero
- for typetest = "tnSep" the temporal lags should be chosen according to the results of the sample non separability ratios, plotted through a boxplot classified for temporal lags (see sepindex for more details)
- for model tests (typetest equal to "productSum", "intProduct" and "gneiting"), the number of analyzed spatial points must be used to create at least 3 spatial couples or multiple of 3, such that each triplet satisfies the condition

$$||\mathbf{h}_1||^{2\gamma} - ||\mathbf{h}_2||^{2\gamma} = ||\mathbf{h}_2||^{2\gamma} - ||\mathbf{h}_3||^{2\gamma}$$

where $\gamma \in]0, 1]$ only for typetest = "intProduct" and "gneiting". The number of positive temporal lags must be at least 3, or multiple of 3, too. The condition

$$u_1^{2\alpha} - u_2^{2\alpha} = u_2^{2\alpha} - u_3^{2\alpha}$$

where $\alpha \in]0, 1]$ must be satisfied for each triplet (only for typetest = "intProduct" and "gneiting"), as clarified in Cappello et al., 2018. The values of γ and α are usually fixed equal to 0.5 or 1 according that the behavior near the origin of the spatial and temporal marginal covariograms is linear or quadratic, respectively. Note that for each spatial triplet and each temporal triplet, 6 contrasts can be defined. However, for typetest = "intProduct" (test on the integrated model) the user has to set arbitrarily one temporal lag equal to zero for each spatial triplet in order to delete redundant contrasts, through the specific setzero method

Slots

couples.st matrix, in which the first two columns contain the couples of spatial points (denoted with order numbers) to be analyzed and the other columns the temporal lags associated with each spatial couples

sel.staz numeric or character, contains the ID codes of the selected spatial points

`sp.couples` data.frame, contains the couples of order numbers associated with the spatial points to be analyzed and the couples of the ID codes

`tl.couples` numeric, contains the temporal lags associated to the couples of the selected spatial points

`typetest` character; contains the code of the test to be performed

Note

Errors occur if

- some spatial points, given in the sequence at the beginning of the function, have not been used to generate the couples of spatial points
- there is at least one spatial couple with no specification of temporal lags
- no temporal lags have been specified
- the number of spatial points fixed in `sel.staz` is less than 2
- the construction of the `sp.couples.in` is not consistent with the test to be performed

References

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

See Also

setzero

Examples

```
sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

### methods for couples
#1. show
couples.sym

#2. [ extract
couples.sym[3, by.row = FALSE]
couples.sym[3, by.row = TRUE]

#3. summary
summary(couples.sym)
```

covablocks-class *Class "covablocks"*

Description

A class for the sample spatio-temporal covariances to be computed for each block of data and for the selected spatial and temporal lags fixed in `stpairs` (output from `couples`). Depending on the type of test the empirical variance, the sample spatial and temporal marginal covariances for each block of data are also computed. Moreover, the sample covariances between the spatio-temporal covariances at the specified spatial and temporal lags are determined.

Usage

```
covablocks(stblocks, stpairs, typetest = "sym")

## S4 method for signature 'covablocks'
show(object)
```

Arguments

`stblocks`	object of class `blocks`
`stpairs`	object of class `couples`, containing the spatial points and the corresponding temporal lags to be analyzed
`typetest`	character, set `typetest ="sym"` for symmetry test (default choice), `typetest ="sep"` for separability test, `typetest ="tnSep"` for type of non separability test,`typetest ="productSum"` for the test on the product-sum class of models, `typetest ="intProduct"` for the test on the integrated product class of models, `typetest ="gneiting"` for the test on the Gneiting class of models
`object`	object of class `covablocks` for method `show`

Details

- If `typetest` is equal to `"sym"` (symmetry test) or `"intProduct"` (test on the integrated product class of models) `mat.cova.h` and `mat.cova.u` are not available
- If `typetest` is equal to `"gneiting"` (test on the Gneiting class of models) `mat.cova.h` is not available
- If temporal lags in `stpairs` are not consistent with block length (`lb`) in `stblocks` an error message will be returned
- If the proportion between the maximum temporal lag in `stpairs` and the block length (`lb`) in `stblocks` is greater than 0.25 a warning message will be returned since the covariance estimation might not be reliable

Slots

`mat.cova` matrix of sample spatio-temporal covariances for each block, computed for the spatial and temporal lags given in `stpairs` (object of class `couples`)

`mat.cova.h` matrix of sample spatial marginal covariances for the specified lags

`mat.cova.u` matrix of sample temporal marginal covariances for the specified lags

`mat.cova.cova` matrix of sample covariances between space-time covariances for each block, computed for the spatial and temporal lags given in `stpairs` (object of class `couples`)

`typetest` character, contains the code of the test to be performed

References

Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models. Stochastic Environmental Research and Risk Assessment, **32** 17–35

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Palma, M., Posa, D., 2016. A general procedure for selecting a class of fully symmetric space-time covariance functions. Environmentrics, **27(4)** 212–224.

Li, B., Genton, M.G., Sherman, M., 2007, A nonparametric assessment of properties of spacetime covariance functions. Journal of the American Statistical Association, **102** 736–744.

See Also

blocks

couples

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
data(air)
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #

sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
```

```
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

block.sym <- blocks(lb = 40, ls = 10, matdata = rr_13, pardata1 = 1, pardata2 = 1,
stpairs = couples.sym)

covabl.sym <- covablocks(stblocks = block.sym, stpairs = couples.sym, typetest = "sym")

### method for covablock
#1. show
covabl.sym
```

---

covaprop-class *Class "covaprop"*

Description

A function for testing some properties (symmetry, separability, type of non-separability) of spatio-temporal covariance functions and some classes of space-time covariance models

Usage

```
covaprop(cblock, cstat, nonseptype = NULL, sign.level = 0.05)

## S4 method for signature 'covaprop'
show(object)
```

Arguments

cblock	object of class covablocks
cstat	object of class covastat or covastatM
nonseptype	integer, this argumet is required only for objects (cblock and cstat) defined to perform the type of non separability test, otherwise it has to be set equal to NULL (default choice). Set nonseptype=0 for testing the null hypothesis that the non separability is non positive; set nonseptype=1 for testing the null hypothesis that the non separability is non negative
sign.level	numeric, level of significance
object	object of class covaprop for method show

Details

- A message helps to decide for either reject the null hypothesis in favor of the alternative or not reject it, at a specific level of significance
- The test on full symmetry (when the slot `@typetest` is equal to `"sym"`) represents the first step for the selection of a suitable class of spatio-temporal covariance functions. According to the definition of full symmetry, the null hypothesis to be tested is

$$H_0 : C(\mathbf{h}, u) - C(\mathbf{h}, -u) = 0$$

- The test of separability (when the slot `@typetest` is equal to `"sep"`) represents the second step of the testing procedure. According to the definition of separability, the null hypothesis to be tested is

$$H_0 : C(\mathbf{h}, u)/C(\mathbf{h}, 0) - C(\mathbf{0}, u)/C(\mathbf{0}, 0) = 0$$

- The test on the type of non separability (when the slot `@typetest` is equal to `"tnSep"`) represents the third step for the selection of a suitable class of space-time covariance functions. According to the definition of type of non separability, the null hypothesis to be tested is that the non separability is non negative

$$H_0 : C(\mathbf{h}, u)/C(\mathbf{h}, 0) - C(\mathbf{0}, u)/C(\mathbf{0}, 0) > 0$$

or

$$H_0 : C(\mathbf{h}, u)/C(\mathbf{h}, 0) - C(\mathbf{0}, u)/C(\mathbf{0}, 0) < 0$$

if the null hypothesis to test is that the non separability is non positive

- If the slot `@typetest` is equal to `"productSum"` `"intProduct"` or `"gneiting"`, the goodness of a specific class of space-time covariance function will be tested. For this testing procedure the generic null hypothesis is:

$$H_0 : \mathbf{Af}(\mathbf{G}) = 0$$

For the analytic expression of each test statistic and its probability distribution see Cappello et al. (2018). In the same contribution the different `f(G)` are given for each test to be computed.

Slots

`test.statistics` numeric, the value of the test statistic

`p.value` numeric, the lower tail p value of the test statistic

`df` numeric, the degrees of freedom, if available

`typetest` character, contains the code of the test to be performed

Note

- A stop occurs if the type of test set in `cblock` is not consistent with the type of test set in `cstat`.
- If the message `Error in solve.default(): system is computationally singular:...` appears, the inverse of the matrix involved in the test statistic in (9) of Cappello et al. (2020) is computationally singular. In order to overcome this numerical problem often related to the tests on the models, the object `stpairs` (of class `couples`) has to be modified. In particular,

by considering that for each spatial triplet and each temporal triplet 6 contrasts can be defined, it is advisable to adopt one of the following options: 1) set equal to zero one of the highest temporal lags, for each spatial and each temporal triplet, through the specific `setzero` method, 2) substitute triplets associated with long spatial or temporal distances with others characterized by lower distances. Then, the user has to run again `blocks`, `covablocks`, `covastatM` and `covaprop`. The above error message also occurs if there are at least two spatial triplets, where:

- two couples are replicated;
- one couple is replicated.

In such cases it is enough to set equal to zero one temporal lag for each temporal triplet associated to the couples involved in the replications.

References

Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models. Stochastic Environmental Research and Risk Assessment, **32** 17–35

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Palma, M., Posa, D., 2016. A general procedure for selecting a class of fully symmetric space-time covariance functions. Environmentrics, **27(4)** 212–224.

Li, B., Genton, M.G., Sherman, M., 2007, A nonparametric assessment of properties of spacetime covariance functions. Journal of the American Statistical Association, **102** 736–744.

See Also

couples

blocks

covablocks

covastat

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
data(air)
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #
```

```
#--Example 1: test on symmetry--#

sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

block.sym <- blocks(lb = 40, ls = 10, matdata = rr_13, pardata1 = 1, pardata2 = 1,
stpairs = couples.sym)

covabl.sym <- covablocks(stblocks = block.sym, stpairs = couples.sym, typetest = "sym")

covast.sym <- covastat(matdata = rr_13, pardata1 = 1, pardata2 = 1,
stpairs = couples.sym, typetest = "sym")

test.sym <- covaprop(cblock = covabl.sym, cstat = covast.sym, nonseptype = NULL,
sign.level = 0.05)

#--Example 2: test on the Gneiting model--#
sel.staz.mod <- c("DETH061", "DEBY047", "DEHE051", "DEUB029", "DENI019",
"DENI051", "DETH026", "DESN049")

sp.couples.in.mod <- matrix(data = c("DETH061", "DEBY047",
                                     "DEHE051", "DEUB029",
                                     "DENI019", "DENI051",
                                     "DEHE051", "DETH026",
                                     "DEBY047", "DESN049",
                                     "DETH026", "DETH061"),
                            ncol = 2, byrow = TRUE)
                            t.couples.in.mod <- c(1, 2, 3)

couples.mod <- couples(sel.staz = sel.staz.mod,
sp.couples.in = sp.couples.in.mod, t.couples.in = t.couples.in.mod,
typetest = "gneiting", typecode = character())

zero.index <- matrix(data = c(3, 7, 6, 7), ncol = 2, byrow = TRUE)

couples.mod <- setzero(x = couples.mod, zero = FALSE, index = zero.index, value = 0)

block.mod <- blocks(lb = 60, ls = 10, matdata = rr_13, pardata1 = 1,
pardata2 = 1, stpairs = couples.mod)

covabl.gn <- covablocks(stblocks = block.mod, stpairs = couples.mod,
typetest = "gneiting")

covast.gn <- covastatM(matdata = rr_13, pardata1 = 1, pardata2 = 1,
```

```
stpairs = couples.mod, typetest = "gneiting", beta.data = seq(0.5, 1, by=0.1))

test.gn <- covaprop(cblock = covabl.gn, cstat = covast.gn, nonseptype = NULL,
sign.level = 0.05)

### method for covaprop
#1. show
test.sym
```

---

covastat-class *Class "covastat"*

Description

A class for the sample spatio-temporal covariances for the specified spatial and temporal lags, given in `stpairs` (object of class `couples`), for symmetry, separability and type of non separability tests. Depending on the type of test, the empirical variance, the sample spatial and temporal marginal covariances are also computed

Usage

```
covastat(matdata, pardata1, pardata2, stpairs, typetest = "sym")

## S4 method for signature 'covastat'
show(object)
```

Arguments

`matdata`	STFDF/STSDF or `data.frame`, which contains the coordinates of the spatial points, the identification code of the spatial points, the indentification code of the temporal points and the values of the variable, typically output from `read.STdata`
`pardata1`	integer, it represents the column in which the spatial ID is stored (if the spatio-temporal data set is given as data.frame) or the number of variables in the STFDF/STSDF (if the data are given as a STFDF/STSDF)
`pardata2`	integer, it represents the column in which the values of the variable are stored (if the spatio-temporal data set is given as data.frame) or the slot in which the values of the variable of interest are stored (if the data are given as a STFDF/STSDF). Note that for STFDF/STSDF the argument is set, by default, equal to 1 if the number of variables is equal to 1
`stpairs`	object of class `couples`, containing the spatial points and the corresponding temporal lags to be analyzed
`typetest`	character, set `typetest = "sym"` for symmetry test (default choice), `typetest = "sep"` for separability test, `typetest = "tnSep"` for type of non separability test
`object`	object of class `covastat` for method `show`

Details

- A message appears on the user's console if the `G` vector contains spatio-temporal negative covariances. The message returns the negative value/values and it will help to identify the spatial and the temporal lags involved
- If `typetest = "sym"` (symmetry test) `cova.h`, `cova.u`, `f.G` and `B` are not available

Slots

`G` matrix, containing the spatio-temporal covariances for the specified lags. For all tests, except for the symmetry test (`typetest = "sym"`), the sample variance and the sample spatial and temporal marginal covariances are also computed and stored in `G`

`cova.h` matrix, containing the sample spatial marginal covariances for the specified lags

`cova.u` matrix, containing the sample temporal marginal covariances for the specified lags

`f.G` array, containing the computation of specific functions of the elements of `G`, see references

`B` matrix, containing the computation of the derivatives of each element of `f.G` with respect to each element of `G`

`A` contrast matrix

`typetest` character, contains the code of the test to be performed

Note

- A stop occurs if the number of spatial points fixed in `stpairs` (object of class `couples`) is less than 2
- A stop occurs if more than 75\ series, since a large number of missing values do not guarantee the reliability of the tests

References

Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models. Stochastic Environmental Research and Risk Assessment, **32** 17–35

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Palma, M., Posa, D., 2016. A general procedure for selecting a class of fully symmetric space-time covariance functions. Environmentrics, **27(4)** 212–224.

Li, B., Genton, M.G., Sherman, M., 2007, A nonparametric assessment of properties of spacetime covariance functions. Journal of the American Statistical Association, **102** 736–744.

See Also

couples

`read.STdata`

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
data(air)
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #

sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

covast.sym <- covastat(matdata = rr_13, pardata1 = 1, pardata2 = 1,
stpairs = couples.sym, typetest = "sym")

### method for covastat
#1. show
covast.sym
```

`covastatM-class` *Class "covastatM"*

Description

A class for the sample spatio-temporal covariances for the specified spatial and temporal lags, given in `stpairs` (object of class `couple`), for the test on the type of class of models. Depending on the type of test, the empirical variance, the sample spatial and temporal marginal covariances are also computed

Usage

```
covastatM(
  matdata,
  pardata1,
  pardata2,
  stpairs,
  typetest = "productSum",
  beta.data = NULL
)

## S4 method for signature 'covastatM'
show(object)
```

Arguments

matdata	STFDF/STSDF or data.frame, which contains the coordinates of the spatial points, the identification code of the spatial points, the indentification code of the temporal points and the values of the variable, typically output from read.STdata
pardata1	integer, it represents the column in which the spatial ID is stored (if the spatio-temporal data set is given as data.frame) or the number of variables in the STFDF/STSDF (if the data are given as a STFDF/STSDF)
pardata2	integer, it represents the column in which the values of the variable are stored (if the spatio-temporal data set is given as data.frame) or the slot in which the values of the variable of interest are stored (if the data are given as a STFDF/STSDF). Note that for STFDF/STSDF the argument is set, by default, equal to 1 if the number of variables is equal to 1
stpairs	object of class couples, containing the spatial points and the corresponding temporal lags to be analyzed
typetest	character, set typetest = "productSum" for the test on the product-sum class of models (default choice), typetest = "intProduct" for the test on the integrated product class of models, typetest = "gneiting" for the test on the Gneiting class of models
beta.data	vector, this argument is required only for typetest = "gneiting", otherwise it has to be set equal to NULL (default choice). It contains the different values of the parameter beta, which can assume values in the range 0-1
object	object of class covastatM for method show

Details

- If typetest = "intProduct" (test on the integrated product class of models) cova.h and cova.u are not available
- If typetest = "gneiting" (test on the Gneiting class of models), cova.h is not available
- A message appears on the user's console if the G vector contains spatio-temporal negative covariances. The message returns the negative value/values and it will help to identify the spatial and the temporal lags involved

Slots

`G` matrix, containing the spatio-temporal covariances for the specified lags. For all tests, the sample variance and the sample spatial and temporal marginal covariances are also computed and stored in `G`

`cova.h` matrix, containing the sample spatial marginal covariances for the specified lags

`cova.u` matrix, containing the sample temporal marginal covariances for the specified lags

`f.G` array, containing the computation of specific functions of the elements of `G`, see references

`B` matrix, containing the computation of the derivatives of each element of `f.G` with respect to each element of `G`

`A` contrast matrix

`beta.data` vector, containing the different values of the parameter beta, available only for the test on the Gneiting class of model (`typetest = "gneiting"`)

`typetest` character, contains the code of the test to be performed

Note

- A stop occurs if the number of spatial points fixed in `stpairs` (object of class `couples`) is less than 2
- A stop occurs if more than 75\ series, since a large number of missing values do not guarantee the reliability of the tests

References

Cappello, C., De Iaco, S., Posa, D., 2018, Testing the type of non-separability and some classes of space-time covariance function models. Stochastic Environmental Research and Risk Assessment, **32** 17–35

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Palma, M., Posa, D., 2016. A general procedure for selecting a class of fully symmetric space-time covariance functions. Environmentrics, **27(4)** 212–224.

Li, B., Genton, M.G., Sherman, M., 2007, A nonparametric assessment of properties of spacetime covariance functions. Journal of the American Statistical Association, **102** 736–744.

See Also

couples

`read.STdata`

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
data(air)
```

```
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #

sel.staz.mod <- c("DERP016", "DENW065", "DENW063", "DEHE046", "DEUB029",
"DETH061", "DENW068", "DETH026", "DENI051")

sp.couples.in.mod <- matrix(data = c("DERP016", "DENW065", "DENW063", "DEHE046",
                                     "DEUB029", "DETH061", "DEHE046", "DENW063",
                                     "DERP016", "DENW068", "DETH026", "DENI051",
                                     "DEUB029", "DETH061", "DENI051", "DETH061",
                                     "DERP016", "DEUB029"),
                            ncol = 2, byrow = TRUE)

t.couples.in.mod <- c(1, 2, 3)

couples.mod <- couples(sel.staz = sel.staz.mod, sp.couples.in =
                         sp.couples.in.mod, t.couples.in = t.couples.in.mod,
                       typetest = "productSum", typecode = character())

zero.index <- matrix(data=c(3, 7, 6, 7, 9, 7), ncol=2, byrow = TRUE)

couples.mod <- setzero(x = couples.mod, zero = FALSE, index = zero.index, value = 0)

covast.ps <- covastatM(matdata = rr_13, pardata1 = 1, pardata2 = 1,
                       stpairs = couples.mod, typetest = "productSum", beta.data = NULL)

### method for covastat
#1. show
covast.ps
```

`read.STdata` *Imports a text file in R*

Description

A function for importing a text file containing spatio-temporal data. In particular, it (a) generates the spatial and temporal IDs, (b) converts the time series of each spatial point (with non-existing values for some dates) into a regularly spaced object within the observed time period, by filling the missing dates with 'NA' (c) converts the data into a `STFDF`, according to the standard of the `spacetime` package, or into a data frame

Usage

```
read.STdata(
  file,
  header = FALSE,
  dec = ".",
  sep = "",
  iclx,
  icly,
  iclt,
  icldate = c(icl.date = 0, iclty = 0, icltm = 0, icltd = 0),
  icltime = c(icl.time = 0, icltH = 0, icltM = 0, icltS = 0),
  iclvr,
  iclsp = 0,
  missing.v = NA,
  save.as = "data.frame",
  date.format = c("code", format = NA),
  bytime = NA,
  tlag,
  time.zone = ""
)
```

Arguments

file	the name of the data file and its extension. The file is searched in the current working directory, otherwise the absolute path has to be included in the file name. Note that data for each spatial point and each temporal point are given by row; each row of the file contains at least the x and y coordinates of a spatial point, the temporal code (or date) and the measurement of the variable of interest.
header	logical, value indicating whether the file contains the names of the variables in the first line. If this argument is missing, header is set to FALSE (default choice)
dec	character, used to indicate decimal points
sep	field separator character. If sep = "" (default choice) columns of the file are separated by white space or tabs (see read.table for more details)
iclx	numeric, the column in which the x-coordinate of the spatial points are stored
icly	numeric, the column in which the y-coordinate of the spatial points are stored
iclt	numeric, the column in which numeric temporal codes are stored. This argument is provided only if the icldate argument is not available: iclt and icldate are mutually exclusive. This argument is set equal to 0, if not available
icldate	numeric vector to set the columns in which the dates are stored. The user has to set icl.date if the date is stored in a single column, otherwise the user has to specify the colunn in which the years (iclty), the months (icltm) or the days (icltd) are stored separately. This argument is set equal to 0 (default choice) if not available
icltime	numeric vector to set the columns in which the time component (hour, minute, second) of a date (if available) is stored. The user has to set icl.time if the

time is stored in a single column, otherwise the user has to specify the column in which the hours (icltH), the minutes (icltM) or the seconds (icltS) are stored separately. This argument is set equal to 0 (default choice) if not available

iclvr — numeric, the column in which the values of the variable are stored

iclsp — numeric, the column in which the identification codes (IDs) for the spatial locations are stored. This argument is set equal to 0 (default choice) if IDs for the spatial locations are not available

missing.v — code used to indicate the presence of missing values in the imported data. By default this argument is set equal to NA

save.as — character, indicating the class of the data to be returned. It is allowed to choose between two options for saving the file ("STFDF" or "data.frame")

date.format — vector, whose first element date.format[1] denotes the class of the temporal component to be imported and the second one date.format[2] represents the corresponding format. Note that the supported class of dates are "yearmon", "yearqtr", "Date", "POSIX" (see **Base**, **lubridate**, **zoo**); moreover the personalized options "year" and "code" are also admissible and are used if the temporal coordinate is given by year or as a numerical code, respectively. By default, the argument date.format is set equal to ("code", format = NA). If the temporal component, provided for example in year and month, is given in separeted columns in the text file, the required format in date.format[2] is of the type "%Y %m"; in general the format requires the use of white spaces between two consecutive time units

bytime — character, which denotes the time disaggregation of interest, set NA (default choice) for numeric temporal code, otherwise "%Y" or "%y" if values are taken by year, "%m" if values are taken by month, "%d" if values are taken by day, "%q" if values are taken by quarter, "%H" if values are taken by hour, "%M" if values are taken by minute and "%S" if values are taken by seconds

tlag — numeric, time increment/lag between two temporal observations

time.zone — character, time zone for dates with time component

Details

- Uncomplete time series, for each spatial point, are filled with NA
- Some checks on the admissibility of the supported classes of dates are implemented
- Time indexes for temporal points are coded for data.frame output by using consecutive numbers starting from 1 (column 'timeIndex')
- The spatial points are coded by using the string 'id' and the consecutive numbers starting from 1 (column 'spatialIndex')

Value

object of the STFDF-class or data.frame, which contains coordinates of the spatial points, the spatial IDs, the temporal IDs, the dates (if available in the input file) and the observed values of the variable of interest

References

Bivand, R. S., Pebesma, E., Gomez-Rubio, V., 2013, *Applied spatial data analysis with R*, Second edition. New York: Springer. https://asdar-book.org/

Grolemund, G, Wickham, H., 2011, Dates and Times Made Easy with lubridate. Journal of Statistical Software, **40(3)** 1–25.

Pebesma, E.J., 2012, spacetime: Spatio-Temporal Data in R. Journal of Statistical Software, **51(7)** 1–30.

Zeileis, A., Grothendieck. G., 2005, zoo: S3 Infrastructure for Regular and Irregular Time Series. Journal of Statistical Software, **14(6)** 1–27.

See Also

STFDF-class

read.table

yearmon

yearqtr

Dates for dates without times

DateTimeClasses

timezones for OlsonNames

Examples

```
#example 1: import a text file, with dates stored in a single column (the 4th)
# and fill missing time points in monthly time series, with time lag equal to one


## Not run
## To run example 1 paste and copy the following lines (without the symbol '#')
## in the console:
#file_date <- system.file("extdata", "file_date.txt", package = "covatest")
#db.date <- read.STdata(file = file_date, header = TRUE, iclx = 2, icly = 3, iclt = 0,
#icldate = c(icl.date = 4, iclty = 0, icltm = 0, icltd = 0),
#icltime = c(icl.time = 0, icltH =0, icltM = 0, icltS = 0),
#iclvr = 5, iclsp = 1, missing.v = -99999, save.as = "data.frame",
#date.format = c("Date", "%d-%m-%Y"), bytime = "%m", tlag = 1)


#example 2: import a text file, with dates and times stored in different columns
# (from the 4th to the 9th) and fill missing time points in hourly time series,
# with time lag equal to three

## Not run
## To run example 2 paste and copy the following lines (without the symbol '#')
## in the console:
#file_datetime <- system.file("extdata", "file_datetime.txt", package = "covatest")
#db.datetime <- read.STdata(file = file_datetime, header = TRUE, iclx = 2, icly = 3, iclt = 0,
#icldate = c(icl.date = 0, iclty = 6, icltm = 5, icltd = 4),
#icltime = c(icl.time = 0, icltH = 7, icltM = 8, icltS = 9),
```

```
#iclvr = 10, iclsp = 1, missing.v = -99999, save.as = "data.frame",
#date.format = c("POSIX", "%Y %m %d %H %M %S"), bytime = "%H", tlag = 3)


#example 3: import a text file, with dates and times stored in different columns
# (from the 4th to the 9th) and fill missing time points in quarterly time series,
# with time lag equal to one

## Not run
## To run example 3 paste and copy the following lines (without the symbol '#')
## in the console:
#file_yq <- system.file("extdata", "file_yq.txt", package = "covatest")
#db.yq <- read.STdata(file = file_yq, header = TRUE, iclx = 2, icly = 3, iclt = 0,
#icldate = c(icl.date = 4, iclty = 0, icltm = 0, icltd = 0),
#icltime = c(icl.time = 0, icltH =0, icltM = 0, icltS = 0),
#iclvr = 5, iclsp = 1, missing.v = -99999, save.as = "data.frame",
#date.format = c("yearqtr", "%Y-Q%q"), bytime = "%q", tlag = 1)
```

---

sepindex-class *Class "sepindex"*

Description

A class for the non separability index (r) for different spatial and temporal lags:

$$r(\mathbf{h}, u, \Theta) = \rho(\mathbf{h}, u; \Theta)/[\rho(\mathbf{h}, 0; \Theta)\rho(\mathbf{0}, u; \Theta)]$$

with $\rho(\mathbf{h}, u; \Theta) > 0$; $\rho(\mathbf{h}, 0; \Theta) > 0$ and $\rho(\mathbf{0}, u; \Theta) > 0$. On the basis of this index, the type of non separability of the covariance function can be analyzed.

Usage

```
sepindex(vario_st, nt, ns, globalSill)

## S4 method for signature 'sepindex'
boxplot(x, ...)

## S4 method for signature 'sepindex'
show(object)

## S4 method for signature 'sepindex'
x[i, j, drop = FALSE]

## S4 method for signature 'sepindex'
summary(object)
```

Arguments

vario_st	object of class StVariogram, containing the spatio-temporal sample variogram, output from the function variogramST of the package gstat
nt	integer, the number of temporal lags in vario_st
ns	integer, the number of spatial lags in vario_st
globalSill	numeric, the value of the sample variance
x	object of class sepindex for methods boxplot and extract
...	any arguments that will be passed to the panel plotting functions
object	object of class sepindex for methods show and summary
i	index specifying elements to extract. Each row includes data for specific spatio-temporal lags
j	index specifying elements to extract. Set 1 for spatial lags (hs), 2 for temporal lags (ht) and 3 for the nonseparability index (SepIndex)
drop	logical, the argument is set, by default, equal to FALSE to preserve the structure of the object. It is advisable not to change this option

Slots

sep.index.ratio the empirical non separability index ratio and the corresponding spatio-temporal lags

cov.st the spatio-temporal sample covariance function and the corresponding spatio-temporal lags

cov.tm the purely temporal sample covariance function and the corresponding temporal lags

cov.sp the purely spatial sample covariance function and the corresponding spatial lags

References

Cappello, C., De Iaco, S., Posa, D., 2020, covatest: An R Package for Selecting a Class of Space-Time Covariance Functions. Journal of Statistical Software, **94(1)** 1–42.

De Iaco, S., Posa, D., 2013, Positive and negative non-separability for space-time covariance models. Journal of Statistical Planning and Inference, **143** 378–391.

Gräler, B., Pebesma, E.J., Heuvelink G., 2016, Spatio-Temporal Interpolation Using gstat. The R Journal, **8(1)** 204–218.

Pebesma, E.J., 2004, Multivariable geostatistics in S: the gstat package. Computers & Geosciences, **30** 683–691.

Rodriguez, A., Diggle, P.J., 2010, A class of convolution-based models for spatio-temporal processes with non-separable covariance structure. Scandinavian Journal of Statistics, **37(4)** 553–567.

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)
#library(gstat)
data(air)
```

```
ls()
if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))
rr = rural[,"2005::2010"]
unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))
r5to10 = rr[-unsel,]
rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #

#compute the Global Sill
C00_13 <- var(rr_13[,,"PM10"]@data[[1]], na.rm = TRUE)

#estimate the spatio-temporal variogram by using the function variogramST
#of the package gstat.
#For this aim see vv_13.Rd
data(vv_13)
nonsep.index <- sepindex(vario_st = vv_13, nt = 16, ns = 4, globalSill = C00_13)

##methods for sepindex

#1. show
nonsep.index

#2. summary
summary(nonsep.index)

#3. boxplot
boxplot(nonsep.index, ylab="Non-separability ratio")

#4. [ extract
nonsep.index[1:8, ] #selection of the first 8 rows
nonsep.index[1:8, 1:2] #selection of the first 2 columns
```

`setzero` *setzero*

Description

Through the function couples, `m` spatial couples and `n` temporal lags are provided, hence a set of `m` x `n` spatio-temporal lags are defined. If some of these lags are not required for the specific test, they can be set equal to zero by using the `setzero` method for object of class `couples`

Usage

```
setzero(x, zero = TRUE, index = NULL, value)

## S4 method for signature 'couples'
setzero(x, zero = TRUE, index = NULL, value)
```

Arguments

x	object of class couples
zero	logical, if TRUE (the default) all negative temporal lags are replaced with zero. If x@typetest is equal to "sym" (symmetry test) the argument setzero is ignored because both positive and negative temporal lags are required for computing the test
index	two column matrix. Each row of the matrix index contains the specific row and column, of the slot @couples.st, for which the spatio-temporal covariance is not required
value	numeric, the value to be replaced. Note that this method is reasonable to be used only to replace a value equal to zero

See Also

couples

Examples

```
sel.staz.sym <- c("DERP016", "DENW065", "DEHE051", "DETH026", "DENW063", "DENI019",
"DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049")

sp.couples.in.sym <- matrix(data = c("DERP016", "DENW065", "DEHE051", "DETH026",
"DENW063", "DENI019", "DENW068", "DEHE046", "DEUB029", "DEBY047", "DETH061", "DESN049"),
ncol = 2, byrow = TRUE)

t.couples.in.sym <- c(1, 2)

couples.sym <- couples(sel.staz = sel.staz.sym, sp.couples.in = sp.couples.in.sym,
t.couples.in = t.couples.in.sym, typetest = "sym", typecode = character())

zero.index <- matrix(data=c(1,3,1,4,2,5,2,6), ncol=2, byrow = TRUE)

setzero(couples.sym, zero = FALSE, index = zero.index, value = 0)
```

---

vv_13 *Precomputed variogram for PM10 in data set air*

Description

Precomputed variogram for PM10 in a subset of the air quality data set air

Usage

```
data(vv_13)
```

Format

Object of class StVariogram

Examples

```
# --start define the STFDF rr_13-- #
library(sp)
library(spacetime)


data(air)

ls()

if (!exists("rural")) rural = STFDF(stations, dates, data.frame(PM10 =
as.vector(air)))

rr <- rural[,"2005::2010"]

unsel = which(apply(as(rr, "xts"), 2, function(x) all(is.na(x))))

r5to10 = rr[-unsel,]

rr_13 <- r5to10[c("DEHE046","DESN049","DETH026","DENW063","DETH061","DEBY047",
"DENW065","DEUB029","DENW068","DENI019","DEHE051","DERP016","DENI051"),
"2005::2006"]
# --end define the STFDF rr_13-- #

## Not run
## To estimate the spatio-temporal variogram, paste and copy the following lines
## (without the symbol '#') in the console:
#
## vv_13 is obtained by running the function variogramST of the package gstat,
## as follows
#
# vv_13 <- gstat::variogramST(PM10~1, rr_13, width=60, cutoff = 220, tlags=0:15)
## End (Not run)
```

Index