Package 'SpaceTimeBSS'

October 12, 2022

Type Package

Title Blind Source Separation for Multivariate Spatio-Temporal Data

Version 0.1-0

Date 2022-08-10

Maintainer Christoph Muehlmann <christoph.muehlmann@tuwien.ac.at>

Description Simultaneous/joint diagonalization of local autocovariance matrices to estimate spatio-temporally uncorrelated random fields.

License GPL (>= 2)

Imports Rcpp (>= 1.0.2), JADE, Matrix, methods

Suggests sftime, sf, spacetime, xts, zoo

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Christoph Muehlmann [aut, cre]
(<https://orcid.org/0000-0001-7330-8434>),
Nikolaus Piccolotto [aut] (<https://orcid.org/0000-0001-6876-6502>),
Sandra De Iaco [aut] (<https://orcid.org/0000-0003-1820-2068>),
Klaus Nordhausen [aut] (<https://orcid.org/0000-0002-3758-8501>)

Repository CRAN

Date/Publication 2022-08-13 12:10:05 UTC

R topics documented:

SpaceTimeBSS-package *Blind Source Separation for Multivariate Spatio-Temporal Data*

Description

Simultaneous/joint diagonalization of local autocovariance matrices to estimate spatio-temporally uncorrelated random fields.

Details

Package:	SpaceTimeBSS
Type:	Package
Version:	0.1-0
Date:	2021-08-10
License:	GPL (>= 2)

Solving the second order blind source separation problem for multivariate space-time random fields. The random fields can be irregular in space but must be regular in time. The main function of this package is:

- `stbss` This function computes local autocovariance matrices. The considered temporal lags are integer numbers and the spatial lags are defined by spatial kernel functions. Then, these local autocovariance matrices and the sample covariance are simultaneously/jointly diagonalized.

Joint diagonalization is computed with the `frjd` (fast real joint diagonalization) algorithm from the package `JADE`.

The available finite realizations of the space time random fields can be defined by matrices or an object of classes `STFDF`, `STSDF` or `st_sftime`.

Author(s)

Christoph Muehlmann, Nikolaus Piccolotto, Sandra De Iaco, Klaus Nordhausen

Maintainer: Christoph Muehlmann <christoph.muehlmann@tuwien.ac.at>

coef.stbss *Coef Method for an Object of Class 'stbss'*

Description

Extracts the estimated unmixing matrix of an object of class `'stbss'`.

Usage

```
## S3 method for class 'stbss'
coef(object, ...)
```

Arguments

object	object of class 'stbss'. Usually result of stbss.
...	further arguments to be passed to or from methods.

Value

Returns the estimated unmixing matrix of an object of class 'stbss' as a numeric matrix. For a description of the class 'stbss' see stbss.

See Also

stbss

---

lacov *Local Autocovariance Matrices*

Description

Computation of local autocovariance matrices for a multivariate space-time dataset based on a given set of spatio-temporal kernel functions.

Usage

```
lacov(x, coords, time, kernel_type, kernel_parameters,
      lags, kernel_list = NULL, center = TRUE)
```

Arguments

x	either a numeric matrix of dimension c(n, p) where the p columns correspond to the entries of the space-time random field and the n rows are the observations.
coords	a numeric matrix of dimension c(n,2) where each row represents the spatial coordinates of the corresponding observation over a 2D spatial domain.
time	a numeric vector of length n where each entry represents the temporal coordinate of the corresponding observation.
kernel_type	either a string or a string vector of length K (or 1) indicating which spatio-temporal kernel function to use. Implemented choices are 'ring', 'ball' or 'gauss'.
kernel_parameters	a numeric vector of length K (or 1) for the 'ball' and 'gauss' kernel function or a list of length K (or 1) for the 'ring' kernel, see details.

lags	an integer vector of length K (or 1) that provides the temporal lags for the spatio-temporal kernel functions, see details.
kernel_list	a list of spatio-temporal kernel matrices with dimension c(n,n), see details. Usually computed by the function stkmat.
center	logical. If TRUE the data x is centered prior computing the local covariance matrices. Default is TRUE.

Details

Local autocovariance matrices are defined by

$$LACov(f) = 1/(nF_{f,n}) \sum_{i,j} f(s_i - s_j, t_i - t_j)(x(s_i, t_i) - \bar{x})(x(s_j, t_j) - \bar{x})',$$

with

$$F_{f,n}^2 = 1/n \sum_{i,j} f^2(s_i - s_j, t_i - t_j).$$

Here, $x(s_i, t_i)$ are the p random field values at location s_i, t_i, $\bar{x}$ is the sample mean vector, and the space-time kernel function f determines the locality. The following kernel functions are implemented and chosen with the argument kernel_type:

- 'ring': the spatial parameters are inner radius r_i and outer radius r_o, with $r_i < r_o$, and $r_i, r_o \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(r_i < d_s \leq r_o) I(d_t = u)$$

- 'ball': the spatial parameter is the radius r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(d_s \leq r) I(d_t = u)$$

- 'gauss': Gaussian function where 95% of the mass is inside the spatial parameter r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = exp(-0.5(\Phi^{-1}(0.95) d_s / r)^2) I(d_t = u)$$

Above, $I()$ represents the indicator function. The argument kernel_type determines the used kernel function as presented above, the argument lags provides the used temporal lags for the kernel functions (u in the above formulas) and the argument kernel_parameters gives the spatial parameters for the kernel function. Each of the arguments kernel_type, lags and kernel_parameters can be of length K or 1. Specifically, kernel_type can be either one kernel, then each local autocovariance matrix use the same kernel type, or of length K which leads to different kernel functions for the provided kernel parameters. lags can be either one integer, then for each kernel the same temporal lag is used, or an integer vector of length K which leads to different temporal lags. In the same fashion kernel_parameters is a vector of length K or 1. If kernel_type equals 'ball' or 'gauss' then the corresponding entry of kernel_parameters gives the single spatial radius parameter. In contrast, if (at least one entry of) kernel_type equals 'ring', then kernel_parameters must be a list of length K (or 1) where each entry is a numeric vector of length 2 defining the inner and outer spatial radius. See examples below.

Alternatively, a list of kernel matrices can be given directly to the function lacov through the kernel_list argument. A list with kernel matrices can be computed with the function stkmat.

Value

lacov returns a list of length K where each entry is a numeric matrix of dimension c(p, p) corresponding to a local autocovariance matrix.

See Also

stkmat, stbss

Examples

```
# space and time coordinates
n_t <- 50
n_sp <- 10
st_coords <- as.matrix(expand.grid(1:n_sp, 1:n_sp, 1:n_t))

# simulate three latent white noise fields
field_1 <- rnorm(nrow(st_coords))
field_2 <- rnorm(nrow(st_coords))
field_3 <- rnorm(nrow(st_coords))

# compute the observed field
latent_field <- cbind(field_1, field_2, field_3)
mixing_matrix <- matrix(rnorm(9), 3, 3)
observed_field <- latent_field

# lacov with different ring kernels and same lags
lacov_r <- lacov(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                 kernel_type = 'ring',
                 kernel_parameters = list(c(0, 1), c(1, 2)), lags = 1)

# lacov with same ball kernels and different lags
lacov_b <- lacov(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                 kernel_type = 'ball', kernel_parameters = 1, lags = c(1, 2, 3))

# lacov with different gauss kernels and different lags
lacov_g <- lacov(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                 kernel_type = 'gauss', kernel_parameters = 1, lags = 1:3)

# lacov mixed kernels
lacov_m <- lacov(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                 kernel_type = c('ball', 'ring', 'gauss'),
                 kernel_parameters = list(1, c(1:2), 3), lags = 1:3)

# lacov with a kernel list
kernel_list <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
                      kernel_type = 'ring',
                      kernel_parameters = list(c(0, 1)), lags = 1)
lacov_k <- lacov(observed_field, kernel_list = kernel_list)
```

print.stbss *Print Method for an Object of Class 'stbss'*

Description

Prints the estimated unmixing matrix, the (pseudo-)eigenvalues and the diagonalized local autocovariance matrices for an object of class `'stbss'`.

Usage

```
## S3 method for class 'stbss'
print(x, ...)
```

Arguments

`x`	object of class `'stbss'`. Usually result of `stbss`.
`...`	additional arguments for the method `print.listof`.

Value

No return value.

See Also

`stbss`

stbss *Space Time Blind Source Separation*

Description

For a given multivariate space-time dataset, `stbss` estimates the realization of spatio-temporally uncorrelated random fields through a linear transformation which is defined by a so-called mixing matrix and a location vector. This is done assuming a spatio-temporal blind source separation model and simultaneously/jointly diagonalizing the sample covariance matrix and one/many local autocovariance matrices.

Usage

```
stbss(x, ...)

## Default S3 method:
stbss(x, coords, time, kernel_type,
      kernel_parameters, lags, ordered = TRUE, kernel_list = NULL, ...)
## S3 method for class 'STFDF'
```

```
stbss(x, ...)
## S3 method for class 'STSDF'
stbss(x, ...)
## S3 method for class 'sftime'
stbss(x, ...)
```

Arguments

x	either a numeric matrix of dimension c(n, p) where the p columns correspond to the entries of the space-time random field and the n rows are the observations, an object of class STFDF, an object of class STSDF or an object of class st_sftime.
coords	a numeric matrix of dimension c(n,2) where each row represents the coordinates of a point in the spatial domain over a 2D spatial domain. Only needed if x is a matrix and the argument kernel_list is NULL.
time	a numeric vector of length n where each entry represents the time of a point in the temporal domain. Only needed if x is a matrix and the argument kernel_list is NULL.
kernel_type	either a string or a string vector of length K (or 1) indicating which spatio-temporal kernel function to use. Implemented choices are 'ring', 'ball' or 'gauss'.
kernel_parameters	a numeric vector of length K (or 1) for the 'ball' and 'gauss' kernel function or a list of length K (or 1) for the 'ring' kernel, see details.
lags	an integer vector of length K (or 1) that provides the temporal lags for the spatio-temporal kernel functions, see details.
ordered	logical. If TRUE the entries of the latent field are ordered by the sum of squared (pseudo-)eigenvalues of the diagonalized local covariance matrix/matrices. Default is TRUE.
kernel_list	a list of spatio-temporal kernel matrices with dimension c(n,n), see details. Usually computed by the function stkmat.
...	further arguments for the fast real joint diagonalization algorithm that jointly diagonalizes the local covariance matrices. See details and frjd.

Details

It is assumed that the p-variate space-time random field $x(s,t)$ is formed by

$$x(s,t) = Az(s,t) + b,$$

where $z(s,t)$ is the latent p-variate space-time random field, A and b are the mixing matrix and a location vector and s and t are the space and time coordinates. Furthermore, it is assumed that $z(s,t)$ is white and consists of space-time uncorrelated components. The goal is to reverse the linear form by estimating an unmixing matrix and the location vector. This is done by simultaneously/jointly diagonalizing local autocovariance matrices which are defined by

$$LACov(f) = 1/(nF_{f,n}) \sum_{i,j} f(s_i - s_j, t_i - t_j)(x(s_i,t_i) - \bar{x})(x(s_j,t_j) - \bar{x})',$$

with

$$F^2_{f,n} = 1/n \sum_{i,j} f^2(s_i - s_j, t_i - t_j).$$

Here, $x(s_i, t_i)$ are the p random field values at location s_i, t_i, $\bar{x}$ is the sample mean vector, and the space-time kernel function f determines the locality. The following kernel functions are implemented and chosen with the argument `kernel_type`:

- `'ring'`: the spatial parameters are inner radius r_i and outer radius r_o, with $r_i < r_o$, and $r_i, r_o \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(r_i < d_s \leq r_o) I(d_t = u)$$

- `'ball'`: the spatial parameter is the radius r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(d_s \leq r) I(d_t = u)$$

- `'gauss'`: Gaussian function where 95% of the mass is inside the spatial parameter r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = exp(-0.5(\Phi^{-1}(0.95) d_s / r)^2) I(d_t = u)$$

Above, $I()$ represents the indicator function. The argument `kernel_type` determines the used kernel function as presented above, the argument `lags` provides the used temporal lags for the kernel functions (u in the above formulas) and the argument `kernel_parameters` gives the spatial parameters for the kernel function. Each of the arguments `kernel_type`, `lags` and `kernel_parameters` can be of length `K` or `1`. Specifically, `kernel_type` can be either one kernel, then each local autocovariance matrix use the same kernel type, or of length `K` which leads to different kernel functions for the provided kernel parameters. `lags` can be either one integer, then for each kernel the same temporal lag is used, or an integer vector of length `K` which leads to different temporal lags. In the same fashion `kernel_parameters` is a vector of length `K` or `1`. If `kernel_type` equals `'ball'` or `'gauss'` then the corresponding entry of `kernel_parameters` gives the single spatial radius parameter. In contrast, if (at least one entry of) `kernel_type` equals `'ring'`, then `kernel_parameters` must be a list of length `K` (or `1`) where each entry is a numeric vector of length `2` defining the inner and outer spatial radius.

Internally, `stbss` calls `stkmat` to compute a list of `c(n,n)` kernel matrices based on the parameters given, where each entry of those matrices corresponds to $f(s_i - s_j, t_i - t_j)$. Alternatively, such a list of kernel matrices can be given directly to the function `stbss` through the `kernel_list` argument. This is useful when `stbss` is called numerous times with the same coordinates/kernel functions as the computation of the kernel matrices is then only done once prior the actual `stbss` calls. For details see also `lacov`.

If more than one local autocovariance matrix is used `stbss` jointly diagonalizes these matrices with the function `frjd`. `...` provides arguments for `frjd`, useful arguments might be:

- `eps`: tolerance for convergence.
- `maxiter`: maximum number of iterations.

Value

stbss returns a list of class 'stbss' with the following entries:

s	object of class(x) containing the estimated source space-time values.
coords	coordinates of the observations. Is NULL if class(x) is not a matrix or if kernel_list is provided at the stbss call.
time	time of the observations. Is NULL if kernel_list is provided or if class(x) is not a matrix at the stbss call.
w	estimated unmixing matrix.
w_inv	inverse of the estimated unmixing matrix.
pevals	(pseudo-)eigenvalues for each latent field entry.
d	matrix of stacked (jointly) diagonalized local autocovariance matrices with dimension c(length(kernel_parameters)*p,p).
x_mu	columnmeans of x.
cov_inv_sqrt	square root of the inverse sample covariance matrix of x.

See Also

stkmat, frjd

Examples

```
# space and time coordinates
n_t <- 50
n_sp <- 10
st_coords <- as.matrix(expand.grid(1:n_sp, 1:n_sp, 1:n_t))

# simulate three latent white noise fields
field_1 <- rnorm(nrow(st_coords))
field_2 <- rnorm(nrow(st_coords))
field_3 <- rnorm(nrow(st_coords))

# compute the observed field
latent_field <- cbind(field_1, field_2, field_3)
mixing_matrix <- matrix(rnorm(9), 3, 3)
observed_field <- latent_field

# apply stbss with lag 1 and a ring kernel
stbss_res <- stbss(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                   kernel_type = 'ring', kernel_parameters = list(c(0, 1)), lags = 1)

# print object
print(stbss_res)

# unmixing matrix
w_unmix <- coef(stbss_res)

# apply the same stbss with a kernel list
kernel_list <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
```

```
                      kernel_type = 'ring', kernel_parameters = list(c(0, 1)), lags = 1)
stbss_res_k <- stbss(observed_field, kernel_list = kernel_list)

# apply stbss with three ball kernels
stbss_res_b <- stbss(observed_field, coords = st_coords[, 1:2], time = st_coords[, 3],
                     kernel_type = 'ball', kernel_parameters = 1:3, lags = 1:3)
```

---

stkmat *Spatio-Temporal Kernel Matrices*

Description

Computation of spatio-temporal kernel matrices for given kernel functions.

Usage

```
stkmat(coords, time, kernel_type, kernel_parameters, lags)
```

Arguments

`coords`	a numeric matrix of dimension `c(n,2)` where each row represents the spatial coordinates of the corresponding observation over a 2D spatial domain.
`time`	an integer vector of length `n` where each entry represents the temporal coordinate of the corresponding observation.
`kernel_type`	either a string or a string vector of length `K` (or `1`) indicating which spatio-temporal kernel function to use. Implemented choices are `'ring'`, `'ball'` or `'gauss'`.
`kernel_parameters`	a numeric vector of length `K` (or `1`) for the `'ball'` and `'gauss'` kernel function or a list of length `K` (or `1`) for the `'ring'` kernel, see details.
`lags`	an integer vector of length `K` (or `1`) that provides the temporal lags for the spatio-temporal kernel functions, see details.

Details

The following kernel functions are implemented and chosen with the argument `kernel_type`:

- `'ring'`: the spatial parameters are inner radius r_i and outer radius r_o, with $r_i < r_o$, and $r_i, r_o \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(r_i < d_s \leq r_o) I(d_t = u)$$

- `'ball'`: the spatial parameter is the radius r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = I(d_s \leq r) I(d_t = u)$$

- 'gauss': Gaussian function where 95% of the mass is inside the spatial parameter r, with $r \geq 0$, the temporal parameter is the temporal lag u:

$$f(d_s, d_t) = exp(-0.5(\Phi^{-1}(0.95)d_s/r)^2)I(d_t = u)$$

Above, $I()$ represents the indicator function. The argument kernel_type determines the used kernel function as presented above, the argument lags provides the used temporal lags for the kernel functions (u in the above formulas) and the argument kernel_parameters gives the spatial parameters for the kernel function. Each of the arguments kernel_type, lags and kernel_parameters can be of length K or 1. Specifically, kernel_type can be either one kernel, then each local autocovariance matrix use the same kernel type, or of length K which leads to different kernel functions for the provided kernel parameters. lags can be either one integer, then for each kernel the same temporal lag is used, or an integer vector of length K which leads to different temporal lags. In the same fashion kernel_parameters is a vector of length K or 1. If kernel_type equals 'ball' or 'gauss' then the corresponding entry of kernel_parameters gives the single spatial radius parameter. In contrast, if (at least one entry of) kernel_type equals 'ring', then kernel_parameters must be a list of length K (or 1) where each entry is a numeric vector of length 2 defining the inner and outer spatial radius. See examples below.

The output of this function can be used with the function stbss to avoid unnecessary computation of kernel matrices when stbss is called multiple times with the same coordinate/kernel function setting. Additionally, the output can be used with the function lacov.

Value

stkmat returns a list of length K containing numeric matrices of dimension c(n,n) corresponding to the spatio-temporal kernel matrices.

See Also

stbss, lacov

Examples

```
# space and time coordinates
n_t <- 50
n_sp <- 10
coords <- runif(n_sp ^ 2 * 2) * n_sp
dim(coords) <- c(n_sp ^ 2, 2)
time <- 1:n_t

st_coords <- as.matrix(expand.grid(1:nrow(coords), 1:length(time)))
st_coords <- cbind(coords[st_coords[, 1], ], time[st_coords[, 2]])

# different ring kernels and same lags
stkmat_r <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
                   kernel_type = 'ring',
                   kernel_parameters = list(c(0, 1), c(1, 2)), lags = c(1, 1))

# same ball kernels and different lags
stkmat_b <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
```

```
                  kernel_type = 'ball', kernel_parameters = 1:3, lags = c(1, 2, 3))

# different gauss kernels and different lags
stkmat_g <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
                  kernel_type = 'gauss', kernel_parameters = 1:3, lags = 1:3)

# mixed kernels
stkmat_m <- stkmat(coords = st_coords[, 1:2], time = st_coords[, 3],
                  kernel_type = c('ball', 'ring', 'gauss'),
                  kernel_parameters = list(1, c(1:2), 3), lags = 1:3)
```

Index