Package 'mvSUSY'

May 9, 2026

Version 0.1.0

Title Multivariate Surrogate Synchrony

Description
Multivariate Surrogate Synchrony ('mvSUSY') estimates the synchrony within datasets that contain more than two time series. 'mvSUSY' was developed from Surrogate Synchrony ('SUSY') with respect to implementing surrogate controls, and extends synchrony estimation to multivariate data. 'mvSUSY' works as described in Meier & Tschacher (2021).

Imports data.table, RcppAlgos, ggplot2, ggsci

Suggests plotly

License GPL-2

URL https://wtschacher.github.io/mvSUSY/

BugReports https://github.com/wtschacher/mvSUSY/issues

NeedsCompilation no

Author Wolfgang Tschacher [aut, cre],
Deborah Meier [aut],
Jan Gorecki [ctb]

Maintainer Wolfgang Tschacher <wolfgang.tschacher@unibe.ch>

Repository CRAN

Date/Publication 2023-11-07 19:30:02 UTC

Contents

`as.data.frame.mvsusy` *mvsusy to data.frame conversion method*

Description

Turns `mvsusy` class object into a `data.frame`.

Usage

```
## S3 method for class 'mvsusy'
as.data.frame(x, row.names=NULL, optional=FALSE, ...)
```

Arguments

`x`	A `mvsusy` object.
`row.names`	Ignored, only for consistency to generic `as.data.frame` method.
`optional`	Ignored, only for consistency to generic `as.data.frame` method.
`...`	Ignored.

Value

Returns `data.frame`.

See Also

`mvsusy`

Examples

```
set.seed(1)
data = as.data.frame(replicate(5, sample(10, 5000, TRUE)))
res = mvsusy(data, segment=10L, Hz=10L)
df = as.data.frame(res)
df
```

`mvsusy` *Multivariate Surrogate Synchrony*

Description

Multivariate Surrogate Synchrony (mvSUSY) estimates the synchrony within datasets that contain more than two time series. mvSUSY was developed from Surrogate Synchrony (SUSY) with respect to implementing surrogate controls, and extends synchrony estimation to multivariate data.

Usage

```
mvsusy(x, segment, Hz, method=c("lambda_max","omega"), max_pseudo=1000, seed=1)
```

Arguments

`x`	A data.frame of numeric columns.
`segment`	Integer, size in seconds. Must not be be larger than half the time series (`nrow(x)/2`).
`Hz`	Integer, measures per second (sampling rate).
`method`	Character, either `"lambda_max"` or `"omega"`.
`max_pseudo`	Numeric, maximum number of surrogate ("psuedo") data, default 1000.
`seed`	Numeric, same seed generates the same random surrogate data.

Details

Data are entered as a file where multiple time series are in columns. First row with column names. For example, if the data represent the interaction of a group of five people with one time series from each group member, the file has five columns, and its first row contains the member names. The time series are divided in segments of appropriate length (parameter `segment`). Synchrony computation is done separately in each segment, then all segment synchronies are aggregated to yield the "real synchrony" of the multiple time series. Segments are non-overlapping, and the number of segments that fit into the time series may have a remainder (usually a few seconds at the end of the time series), which is not considered. Rows having missing values are removed. "Real synchrony" is controlled for spurious synchrony and non-stationarity using surrogate analysis. Surrogate ("pseudo") data are generated by random segment-shuffling of each column separately. From *m* columns with *s* segments, `s! / (s - m)!` surrogates can be generated. This often very high number of surrogates should be curbed by setting `max_pseudo` appropriately. For each of the `max_pseudo` surrogate datasets, (pseudo) synchrony is computed in the same way as real synchrony. The mean and standard deviation of surrogate synchronies are provided in the output as *mean(synchrony-pseudo)* and *sd(synchrony-pseudo)*. The *effect size (ES)* of synchrony is computed by the difference between *mean(synchrony-real)* and *mean(synchrony-pseudo)* standardized by *sd(synchrony-pseudo)*. Tests against the null-hypothesis `mean(synchrony-real) = mean(synchrony-pseudo)` are performed by a *t-statistic* and a Wilcoxon test (*statistic-nonpar*). Two methods are available to assess mvSUSY: *"lambda_max"* and *"omega"*. *lambda_max* is computed by the eigendecomposition of the correlation matrix. The correlation matrix of the *m* columns (time series) can be described by *m* eigenvalues *lambda*, the largest of which provides an assessment of multivariate synchrony, i.e. the coupling between the time series (columns) of the data. *lambda* is computed in each segment, then

aggregated across all segments. *omega* is a measure of multivariate synchrony that makes use of the actually measured degree of entropy, a measure of disorder of a dataset, with its equivalent to *Shannon information*. Landsberg suggested to normalize entropy *S* by the potential entropy *Spot* possible in a system, providing the measure of *omega* ("Landsberg order") as `omega = 1 - S / Spot`. The entropies can be computed based on the variance-covariance matrix of the multiple time series (Shiner, Davison & Landsberg, 1999). Again, *omega* is computed in each segment then aggregated. Tschacher, Scheier & Grawe (1998) applied these methods in psychotherapy research.

Value

Object of class `mvsusy` is returned.

References

Meier D & Tschacher W (2021). "Beyond Dyadic Coupling: The Method of Multivariate Surrogate Synchrony (mv-SUSY)". Entropy, 23, 1385.
Shiner JS, Davison M, & Landsberg PT (1999). On measures for order and its relation to complexity. In Tschacher W & Dauwalder J-P (Eds). Dynamics, Synergetics, Autonomous Agents. Singapore: World Scientific:, pp. 49-63.
Tschacher W, Scheier C, & Grawe K (1998). Order and Pattern Formation in Psychotherapy. Nonlinear Dyn. Psychol. Life Sci., 2, 195-215.

See Also

`plot.mvsusy`, `as.data.frame.mvsusy`, `print.mvsusy`

Examples

```
set.seed(1)
data = as.data.frame(replicate(5, sample(10, 5000, TRUE)))

## compute mvSUSY (lambda_max method)
res = mvsusy(data, segment=10, Hz=10)
res
plot(res, type="eigenvalue")

## omega method
res = mvsusy(data, segment=10, Hz=10, method="omega")
res
plot(res, type="density")

## export to flat file via data.frame and write.csv
df = as.data.frame(res)
df
```

`plot.mvsusy` *mvsusy plot method*

Description

Generate plot for a `mvsusy` object.

Usage

```
  ## S3 method for class 'mvsusy'
plot(x,
    type=c("eigenvalue","density","free scale","segment-wise","time series"),
    ...,
    bins,
    plotly)
```

Arguments

`x`	A `mvsusy` object.
`type`	Character to specify type of the plot, one of *eigenvalue, density, free scale, segment-wise, time series*.
`...`	Ignored.
`bins`	Numeric. Number of *bins* used in *free scale* plot type, by default number of segments divided by 2.
`plotly`	Logical. Passing `FALSE` will force *time series* plot not to use `plotly` even if installed.

Details

Eigenvalue plot type works only for mvSUSY computed using *lambda_max* method. Time series plot will by default use `plotly` package, if installed, to provide interactive plot.

Value

Returns `NULL` invisibly. Generate plot as a side effect.

See Also

`mvsusy`

Examples

```
set.seed(1)
data = as.data.frame(replicate(5, sample(10, 5000, TRUE)))
res = mvsusy(data, segment=10, Hz=10)
plot(res, type="eigenvalue")
plot(res, type="density")
```

```
plot(res, type="free scale")
plot(res, type="segment-wise")
plot(res, type="time series")
plot(res, type="time series", plotly=FALSE)
```

`print.mvsusy` *mvsusy print method*

Description

Prints information about an `mvsusy` object.

Usage

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

Arguments

`x`	A `susy` object.
`...`	Extra arguments passed to `print.data.frame` method.

Value

Returns `x` invisibly. Display output to console as a side effect.

See Also

`mvsusy`

Examples

```
set.seed(1)
data = as.data.frame(replicate(5, sample(10, 5000, TRUE)))
res = mvsusy(data, segment=10L, Hz=10L)
res
```

Index