Package 'wcc'

July 18, 2026

Encoding UTF-8

Date 2026-07-17

Title Windowed Cross Correlation

Maintainer Steven Boker <smb3u@virginia.edu>

Author Steven Boker [aut, cre],
Minquan Xu [aut],
Sareena Chadha [aut],
Christopher Welker [aut],
Jingyun Wu [aut],
Pascal Deboeck [aut],
Aaron Peikert [aut],
Claude [aut]

Description Calculates Windowed Cross Correlation for pairs of time series.
Provides support for surrogate analysis for nonparametric test of significance.
Calculates aggregate statistics over a range of parameter values.
Plots the results as Windowed Cross Correlation plots and heat maps.
The software is described in Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological Methods, 7(3), 338.

SystemRequirements GNU make

ByteCompile no

Language en-US

License Apache License (== 2.0)

Depends R (>= 4.1.0)

Imports pheatmap, grid, gtable

LazyLoad yes

Collate 'GLLAfunctions.R' 'wccCalc.R' 'wccCalcBatch.R'
'wccAggregate.R' 'wccFindDyadParam.R' 'wccHeatMap.R'
'wccPeakPick.R' 'wccPlot.R' 'wccSurrogateDyads.R'

Biarch true

Version 0.4.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-07-17 23:00:02 UTC

Contents

`wcc-package` *wcc*

Description

Functions for Windowed Cross Correlation.

Details

Calculates Windowed Cross Correlation for estimating association between pairs of nonstationary time series. Provides support for surrogate analysis for nonparametric test of significance. Calculates aggregate statistics over a range of parameter values. Plots the results as wcc plots and heat maps.

Author(s)

Steven Boker, Minquan Xu, Sareena Chadha, Christopher Welker, Jingyun Wu, Pascal Deboeck

Maintainer: Steven Boker <smb3u@virginia.edu>

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

Moulder, R., Boker, S., Ramseyer, F., & Tschacher, W. (2018). Determining synchrony between behavioral time series: An application of surrogate data generation for establishing falsifiable null-hypotheses. Psychological Methods. 23:4 pp 757–773

See Also

See wccCalc for the core function that calculates a windowed cross correlation matrix.
See wccPeakPick for the core function that estimates time lag of maximum association from a windowed cross correlation matrix.
See wccAggregate to call `wccCalc` and `wccPeakPick` and then calculate aggregate statistics from a windowed cross correlation matrix.
See wccPlot to plot a section of a windowed cross correlation matrix and peak picking line from a file created by wccAggregate.
See wccSurrogateDyads to generate a distribution of the aggregate statistics conforming to the null hypothesis that the pairing of the dyad does not matter.
See wccFindDyadParam to explore a range of parameter values for `wccCalc` that are compared between surrogate and real dyads.
See wccHeatmap to visualize results of selected aggregated statistics for combinations of parameters calculated by wccFindDyadParam.

Examples

```
#Create a windowed cross correlation plot
tSeries1 <- sin(c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
tSeries2 <- sin(1+c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
wccPlot(inSeries1=tSeries1, inSeries2=tSeries2, startwindow=1, endwindow=500,
    wMax=100, tMax=100, wInc=1, tInc=1, Lsize=8, pspan=.25, type="Max",
    samplespersecond=10)

# Create two arrays of timeseries with dyadic dependence
array1 <- matrix(NA, nrow=10, ncol=500)
array2 <- matrix(NA, nrow=10, ncol=500)
for(i in 1:10) {
    array1[i,] <- sin(c(1:500)/runif(1, min=5, max=20)) + rnorm(500, mean=0, sd=.5)
    array2[i,] <- array1[i,] + rnorm(500, mean=0, sd=.5)
}
# Select parameters to explore
wMaxVec <- c(50,100)
tMaxVec <- c(25,50)
wccFDPout <- wccFindDyadParam(inArray1=array1, inArray2=array2, wMaxvector=wMaxVec,
    tMaxvector=tMaxVec, nSurrogates=30, samplespersecond=1, method="r")
# Plot a heatmap
wccHeatmap(xparam=wccFDPout$wMax, yparam=wccFDPout$tMax, aggstat=wccFDPout$maxMean,
    xlabel="Window Max", ylabel="Max Lag")
```

`wccAggregate` *wccAggregate*

Description

This function runs `wccCalc` and peakpicking on a pair of time series and returns aggregate statistics.

Usage

```
wccAggregate(inSeries1=NA, inSeries2=NA, wMax=50, tMax=50, wInc=1, tInc=1, Lsize=8,
     pspan=.25, type="Max", samplespersecond=1,
     method=c("c", "cumr", "cumc", "r"), embedD=9, ...)
```

Arguments

inSeries1	A vector of numeric values. This is the first time series on which windowed cross correlation is calculated. When inSeries1 is leading inSeries2, the peak correlation closest to zero in negative in wccPlot. Note that inSeries1 must be of the same length as inSeries2. Note that the only difference between inSeries1 and inSeries2 is which is treated as positive lag and which is negative lag in wccPlot.
inSeries2	A vector of numeric values. This is the second time series on which windowed cross correlation is calculated. Must be of the same length as inSeries1. When inSeries2 is leading inSeries1, the peak correlation closest to zero in positive in wccPlot.
wMax	A numeric indicating the number of samples in a window (default=50). This must be greater or equal to 5 samples. It is recommended that wMax be greater than 15.
tMax	The maximum number of samples to lag the windows (default=50). The cross correlation is calculated for windows that are time lagged against each other, allowing for processes than require some time to evolve. For instance, if the two time series are from motion capture of two individuals conversing with one another, the speaker's movements are likely to occur prior to the responses of the listener. tMax should be set to be greater than the number of samples that represents the maximum time lag that is likely to occur.
wInc	The number of samples incremented between successive windows (default=1). This is most often set to 1. If the sample rate is very high and the time series is long, wInc may be increased. However, wccCalc's maximum sensitivity to nonstationary change in the time series occurs when wInc is 1.
tInc	The number of samples incremented between successive lags (default=1). This is most often set to 1. If the sample rate is high and the maximum lag is high then tInc may be increased. However, wccCalc's maximum sensitivity to changes in lags occurs when the tInc is set to 1.
Lsize	An integer specifying the width required for a peak to be identified by wccPeakPick (default=8). An Lsize that is too small may find spurious peaks that are just noise, whereas an Lsize that is too large may miss actual peaks.
pspan	A numeric specifying the amount of smoothing applied by the loess function within wccPeakPick (default=.25).
type	A character string indicating whether a maximum, 'Max', or minimum, 'Min', should be detected by wccPeakPick (default="Max").
samplespersecond	A numeric indicating the sampling rate of the time series. This will determine the units of the first derivative aggregate statistics. (default=1)

method	Backend passed through to wccCalc: "c" (default, original compiled windcross), "r" (original pure-R loop), "cumc" (cumulative-sum C backend), or "cumr" (cumulative-sum pure-R reference). See wccCalc for details and the missing-data caveat for the cum* methods.
embedD	A numeric indicating the embedding dimension passed to GLLAEmbed to calculate derivatives of the peak picking lag timeseries. The default (9) will provide a medium smooth at the cost of some loss of resolution in the timing of jumps in the second derivative. The smallest legal value is 5, which provides less smoothing and the best time resolution. If your sampling rate is low, then a smaller value of embedD is useful, while if your sampling rate is high, a larger value of embedD will provide better noise rejection by smoothing.
...	Reserved for the deprecated windcross=TRUE/FALSE argument, mapped to method="c" / method="r" with a warning.

Details

wccAggregate is a wrapper for the wccCalc and wccPeakpick functions. The returns from these two function calls are then aggregated into 10 different statistics representing characteristics of wccCalc and wccPeakpick for the pair of input time series. wccAggregate also must be called with a filename in the savefile argument prior to calling wccPlot so that the raw results can be plotted. The wccAggregate function returns a single row dataframe so that it can be called repeatedly by the wccSurrogateDyad function to create a distribution of surrogate Windowed Cross Correlation results. The arguments are also saved in this dataframe so that the wccFindDyadParam function can explore the space of arguments to find appropriate values from a hold-out set of timeseries to then be used to calculate surrogate distributions for the full data set.

Value

wccAggregate returns a data frame with one row and the following columns:

'wMax'	The value of wMax in the argument passed to wccCalc.
'tMax'	The value of tMax in the argument passed to wccCalc.
'wInc'	The value of wInc in the argument passed to wccCalc.
'tInc'	The value of tInc in the argument passed to wccCalc.
'Lsize'	The value of Lsize in the argument passed to wccPeakPick.
'pspan'	The value of pspan in the argument passed to wccPeakPick.
'type'	The value of type as a numeric translation of the type argument passed to wccPeakPick. This is +1 w
'samples'	Length of the time series inSeries1 and inSeries2
'windows'	The number of windows returned by wccCalc.
'pctMissing'	Proportion of missing values in the matrix returned by wccCalc.
'pctMissingWindows'	Proportion of windows (i.e., rows) in the matrix returned by wccCalc that include at least one missing
'maxMean'	Mean of the Fischer's Z transform of the maximum correlation vector returned returned by wccPeakP
'maxVar'	Variance of the Fischer's Z transform of the maximum correlation vector returned returned by wccPea
'totalMean'	Mean of the Fischer's Z transform of all values in the matrix returned by wccCalc.
'totalVar'	Variance of the Fischer's Z transform of all values in the matrix returned by wccCalc .
'zeroLagMean'	Mean of the Fischer's Z transform of all values in the zero lag column of the matrix returned by wccC
'zeroLagVar'	Variance of the Fischer's Z transform of all values in the zero lag column of the matrix returned by wc
'lagMean'	Mean of the lag vector returned by wccPeakPick.

'lagVar'	Variance of the lag vector returned by `wccPeakPick`.
'dlagMean'	Mean of the first derivative of the lag vector returned by `wccPeakPick`.
'dlagVar'	Variance of the first derivative of the lag vector returned by `wccPeakPick`.
'd2lagMean'	Mean of the second derivative of the lag vector returned by `wccPeakPick`.
'd2lagVar'	Variance of the second derivative of the lag vector returned by `wccPeakPick`.
'elapsedSeconds'	Total elapsed seconds required to perform all calculations and function calls.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

See Also

`wccCalc`, `wccPeakPick`, `loess`.

Examples

```
#Calculate aggregated statistics for windowed cross correlation between two time series
tSeries1 <- sin(c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
tSeries2 <- sin(c(1:1000)/15) + rnorm(1000, mean=0, sd=.5)
wccAggregate(inSeries1=tSeries1, inSeries2=tSeries2, wMax=50, tMax=50, wInc=1, tInc=1,
     Lsize=8, pspan=.25, type="Max", samplespersecond=10, method="c")
```

---

wccCalc *Windowed Cross Correlation function*

Description

This function calculates a windowed cross correlation matrix from two time series

Usage

```
wccCalc(inSeries1, inSeries2, wMax=50, tMax=50, wInc=1, tInc=1,
        method=c("c", "cumr", "cumc", "r"), ...)
```

Arguments

`inSeries1`	A vector of numeric values. This is the first time series on which windowed cross correlation is calculated. When inSeries1 is leading `inSeries2`, the peak correlation closest to zero in negative in wccPlot. Note that `inSeries1` must be of the same length as `inSeries2`. Note that the only difference between `inSeries1` and `inSeries2` is which is treated as positive lag and which is negative lag in wccPlot.

`inSeries2` A vector of numeric values. This is the second time series on which windowed cross correlation is calculated. Must be of the same length as `inSeries1`. When `inSeries2` is leading `inSeries1`, the peak correlation closest to zero in positive in wccPlot.

`wMax` A numeric indicating the number of samples in a window (default=50). This must be greater or equal to 5 samples. It is recommended that wMax be greater than 15.

`tMax` The maximum number of samples to lag the windows (default=50). The cross correlation is calculated for windows that are time lagged against each other, allowing for processes than require some time to evolve. For instance, if the two time series are from motion capture of two individuals conversing with one another, the speaker's movements are likely to occur prior to the responses of the listener. tMax should be set to be greater than the number of samples that represents the maximum time lag that is likely to occur.

`wInc` The number of samples incremented between successive windows (default=1). This is most often set to 1. If the sample rate is very high and the time series is long, wInc may be increased. However, `wccCalc`'s maximum sensitivity to nonstationary change in the time series occurs when wInc is 1.

`tInc` The number of samples incremented between successive lags (default=1). This is most often set to 1. If the sample rate is high and the maximum lag is high then tInc may be increased. However, `wccCalc`'s maximum sensitivity to changes in lags occurs when the tInc is set to 1.

`method` Backend selector for the windowed cross correlation. `"c"` (default) calls the original compiled `windcross` binary; `"r"` is the original pure-R loop. `"cumc"` uses a cumulative-sum C backend that computes each cell in O(1) from prefix sums, giving O(n * nLags) cost independent of window size; `"cumr"` is the pure-R reference for that algorithm. The `cum*` methods do not support missing data: any NA in the input is an error. When `tInc > 1` the `cum*` methods use lags `0, tInc, 2*tInc, ..., tMax`, matching `"c"`; the legacy `"r"` method shifts by the column index instead.

`...` Reserved for the deprecated `windcross=TRUE/FALSE` argument, mapped to `method="c"` / `method="r"` with a warning.

Details

`wccCalc` calculates cross correlations between windows, i.e., sampled sections of two time series. These windows are sampled at time lagged intervals from one another so that the association between the time series can be estimated even though the time lag of maximum association between the two time series may be itself time varying. For instance, when two individuals converse, their body motions may be associated such that the leader and follower take turns as speaker and listener. This same kind of nonstationary association has been observed between many physiological time series. At any particular elapsed time, the maximum correlation closest to a lag of zero (whether a positive or negative lag) is taken to be the lagged offset between the two time series. For series that have oscillatory structure (such as head nodding in conversation) the maximum correlation closest to a lag of zero can be interpreted as the phase lag between the series.

Note that missing values in `inSeries1` and `inSeries2` are treated such that each window calculates correlation on pairwise complete observations. If there are fewer than 5 non-missing pairs in any

window then the correlation for that window is returned as NA.

Value

Returns an N x P matrix of numeric values of cross correlations, where N is the number of windows calculated and P is 2*floor(tMax/tInc) + 1, the total lagged values of each window. Each row of the returned matrix represents one elapsed time at sample T. The number of columns is always odd and the center column, C = floor(tMax/tInc) + 1 is the cross correlation for zero lag for the windows from `inSeries1` and `inSeries2` whose last element occurs at elapsed sample T. Columns L less than the center column contain the cross correlation values for the windows where the window from `inSeries1` occurs `L*tInc` samples earlier than the window from `inSeries2` whose last value is at sample T. Similarly columns L greater than the center column contain cross corrlations for windows where the window from `inSeries2` occurs `L*tInc` samples earlier than the window from `inSeries1` whose last value is at sample T.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

Examples

```
#Calculate windowed cross correlation between two time series
tSeries1 <- sin(c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
tSeries2 <- sin(c(1:1000)/15) + rnorm(1000, mean=0, sd=.5)
wccCalcOut <- wccCalc(inSeries1=tSeries1, inSeries2=tSeries2, wMax=50, tMax=50, wInc=1,
     tInc=1, method="r")
```

`wccCalcBatch` *Batched Windowed Cross Correlation*

Description

Computes the Windowed Cross Correlation grid for a list of series pairs in one call, reusing per-series prefix sums across pairs. Real dyads and surrogate pairings drawn from the same series pool can therefore be evaluated together cheaply.

Usage

```
wccCalcBatch(seriesArray1, seriesArray2, pairs=NULL,
             wMax=50, tMax=50, wInc=1, tInc=1, method=c("cumc"))
```

Arguments

seriesArray1	A D1 x n numeric matrix; each row is one series. When pairs is not supplied, seriesArray1 and seriesArray2 must have the same number of rows.
seriesArray2	A D2 x n numeric matrix; each row is one series. Must have the same number of columns as seriesArray1.
pairs	A P x 2 integer matrix; row p selects the pair (seriesArray1[pairs[p,1],], seriesArray2[pairs[p,2],]). When NULL (default), it is set to cbind(1:D, 1:D), i.e. the real dyads.
wMax	Window size in samples (default=50). Must be at least 5.
tMax	Maximum lag in samples (default=50).
wInc	Window increment in samples (default=1).
tInc	Lag increment in samples (default=1).
method	Backend for the batched computation. Currently only "cumc" (batched cumulative-sum C backend) is available on the CPU-only build.

Details

Missing data is not supported: any NA in seriesArray1 or seriesArray2 raises an error.

The C backend computes per-series prefix sums once and reuses them across all pairs, so the batched call scales with the number of unique series rather than the number of pairs. This makes the function particularly efficient when many surrogate pairings are drawn from a small pool of series, as in wccSurrogateDyads and wccFindDyadParam.

Value

An nRow x nCol x P numeric array. Slab [,,p] equals wccCalc(seriesArray1[pairs[p,1],], seriesArray2[pairs[p,2],], wMax=wMax, tMax=tMax, wInc=wInc, tInc=tInc, method="cumc").

See Also

wccCalc for the single-dyad interface and the definition of the returned grid.

Examples

```
D <- 3
n <- 500
arr1 <- t(sapply(1:D, function(i) sin(c(1:n)/10) + rnorm(n, sd=.5)))
arr2 <- t(sapply(1:D, function(i) sin(c(1:n)/12) + rnorm(n, sd=.5)))
grids <- wccCalcBatch(arr1, arr2, wMax=50, tMax=50)
dim(grids)
```

wccFindDyadParam *wccFindDyadParam*

Description

This function calculates the difference between surrogate and actual dyads for a range of possible arguments to wccCalc and wccPeakPick.

Usage

```
wccFindDyadParam(inArray1=NA, inArray2=NA, wMaxvector=c(50), tMaxvector=c(50),
   wIncvector=c(1), tIncvector=c(1), Lsizevector=c(8), pspanvector=c(.25), type="Max",
    nSurrogates=NA, samplespersecond=1,
    method=c("c", "cumr", "cumc", "r"), embedD=9, ...)
```

Arguments

inArray1	An N by T matrix of numeric values representing N timeseries of length T. These are the first set of time series which are randomly paired with timeseries from 'inArray2' and on which windowed cross correlation are calculated. Must be of the same order N by T as 'inArray2'. Note that the only difference between 'inArray1' and 'inArray2' is which is treated as positive lag and which is negative lag in wccPlot. When 'inArray1' is leading 'inArray2', the peak correlation closest to zero in negative in wccPlot.
inArray2	An N by T matrix of numeric values representing N timeseries of length T. These are the second set of time series which are randomly paired with timeseries from 'inArray1' and on which windowed cross correlation are calculated. Must be of the same order N by T as 'inArray1'. When 'inArray2' is leading 'inArray1', the peak correlation closest to zero in positive in wccPlot.
wMaxvector	A numeric vector indicating the number of samples in a window for each test (default=c(50)). This must be greater or equal to 5 samples. It is recommended that wMax be greater than 15.
tMaxvector	A numeric vector indicating the maximum number of samples to lag the windows for each test (default=c(50)). The cross correlation is calculated for windows that are time lagged against each other, allowing for processes than require some time to evolve. For instance, if the two time series are from motion capture of two individuals conversing with one another, the speaker's movements are likely to occur prior to the responses of the listener. tMax should be set to be greater than the number of samples that represents the maximum time lag that is likely to occur.
wIncvector	A numeric vector indicating the number of samples incremented between successive windows for each test (default=c(1)). If the sample rate is very high and the time series is long, wInc may be increased. However, wccCalc's maximum sensitivity to nonstationary change in the time series occurs when wInc is 1.

`tIncvector` A numeric vector indicating the number of samples incremented between successive lags for each test (default=c(1)). If the sample rate is high and the maximum lag is high then tInc may be increased. However, `wccCalc`'s maximum sensitivity to changes in lags occurs when the tInc is set to 1.

`Lsizevector` A numeric vector indicating the width required for a peak to be identified by `wccPeakPick` for each test (default=c(8)). An Lsize that is too small may find spurious peaks that are just noise, whereas an Lsize that is too large may miss actual peaks.

`pspanvector` A numeric vector indicating the amount of smoothing applied by the loess function within `wccPeakPick` for each test (default=c(.25)).

`type` A character string vector indicating whether a maximum, 'Max', or minimum, 'Min', should be detected by `wccPeakPick` for each test (default="Max").

`nSurrogates` A numeric indicating the number of surrogates to generate (default=NA). If there are N timeseries in 'inArray1', then the maximum number of unique surrogates that can be generated is N*(N-1)/2

`samplespersecond`
A numeric indicating the sampling rate of the time series (default=1). This will determine the units of the first and second derivative aggregate statistics.

`method` Backend passed through to `wccCalc`: `"c"` (default, original compiled windcross), `"r"` (original pure-R loop), `"cumc"` (cumulative-sum C backend, which activates the batched code path), or `"cumr"` (cumulative-sum pure-R reference). See `wccCalc` for details and the missing-data caveat for the `cum*` methods.

`embedD` A numeric indicating the embedding dimension passed to `wccGLLAEmbed` to calculate derivatives of the peak picking lag timeseries. The default (9) will provide a medium smooth at the cost of some loss of resolution in the timing of jumps in the second derivative. The smallest legal value is 5, which provides less smooth and the best time resolution. If your sampling rate is low, then a smaller value of `embedD` is useful, while if your sampling rate is high, a larger value of `embedD` will provide better noise rejection by smoothing.

`...` Reserved for the deprecated `windcross=TRUE/FALSE` argument, mapped to `method="c"` / `method="r"` with a warning.

Details

This function iterates over vectors of possible argument values given to `wccCalc` and `wccPeakPick` and for each combination of arguments returns the difference between the surrogate dyadic null hypothesis distribution and the true distribution of each of the statistics returned by wccAggregate. The intended use of this function is to explore the space of possible argument values using a small hold-out set of dyadic timeseries in order to inform the selection of parameters to use for the full dataset. It is recommended that between 10 and 20 dyadic timeseries be used as the hold-out set to input to this function. The two arguments 'wMax' and 'tMax' are the most commonly explored parameters since they tend to have the most impact on results.

Results from combinations of parameters can be visualized using the wccHeatmap function. For any particular hypothesis, one or more aggregate statistics may be most useful in assessing the association between a dyad's time series. Thus, one may wish to focus in on that statistic when choosing parameters. wccHeatmapwMaxtMax allows the visualization of the effect of selecting the 'wMax' and 'tMax' parameters on the statistic of interest.

Note that surrogate and real dyads will be calculated for all combinations of parameters in the argument vectors. Thus, if one wishes to explore the effects of 8 choices of 'wMax' and 8 choices of 'tMax', a total of 64 surrogate analyses will be run. If one uses a holdout set of 20 and asks for 100 surrogates, this will result in 7,680 calls to wccAggregate. Such an exploration of the parameter space can take hours to run. Add to that 2 choices of 'wInc' and 2 choices of 'tInc' and you may be talking about days of processing. Be careful when choosing how many combinations to explore at one time.

Value

Returns a dataframe with some of the columns from the wccAggregate function, some columns with redefined meanings, and some new columns.

For each aggregated statistic a column giving the proportion of real dyads' statistics are greater than .95 or less than .05 of the values in the surrogate statistics distribution. For each aggregated statistic a column giving the two sided Kolmogorov-Smirnov Test probabilities comparing the surrogate and real dyad distributions. These probabilities are those returned by a call to ks.test() with alternative="two.sided". For each aggregated statistic a column giving the quantile difference scores for the difference between the surrogate distribution and the real distribution of dyadic timeseries. These quantile differences are the mean of the difference between calls to quantile() with probs=seq(.1,.9, by=.1) for the real dyads and the surrogate dyads. Each row in the dataframe presents the results from one combination of chosen parameters.

The returned data frame is defined as follows.

'wMax'	The value of wMax in the argument passed to `wccCalc`.
'tMax'	The value of tMax in the argument passed to `wccCalc`.
'wInc'	The value of wInc in the argument passed to `wccCalc`.
'tInc'	The value of tInc in the argument passed to `wccCalc`.
'Lsize'	The value of Lsize in the argument passed to `wccPeakPick`.
'pspan'	The value of pspan in the argument passed to `wccPeakPick`.
'type'	The value of type as a numeric translation of the type argument passed to `wccPeakPick`. This is +1 w
'samples'	Length of the time series inSeries1 and inSeries2
'windows'	The number of windows returned by `wccCalc`.
'pctMissing'	Proportion of missing values in the matrix returned by `wccCalc`.
'pctMissingWindows'	Proportion of windows (i.e., rows) in the matrix returned by `wccCalc` that include at least one missing
'maxMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'maxVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'totalMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'totalVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'zeroLagMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'zeroLagVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'lagMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'lagVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'dlagMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'dlagVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'd2lagMean'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'd2lagVar'	The proportion of the real data that are either greater than 95% less than 5% of the surrogate distribut
'maxMean'	Mean of the Fischer's Z transform of the maximum correlation vector returned returned by `wccPeakP`
'maxVar'	Variance of the Fischer's Z transform of the maximum correlation vector returned returned by `wccPea`

'totalMeanKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'totalVarKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'zeroLagMean'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'zeroLagVarKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'lagMeanKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'lagVarKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'dlagMeanKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'dlagVarKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'd2lagMeanKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'd2lagVarKS'	The probability of the two tailed Kolmogorov-Smirnov test for the difference between the surrogate d
'maxMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'maxVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the Variance
'totalMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'totalVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the variance
'zeroLagMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'zeroLagVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the variance
'lagMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'lagVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the variance
'dlagMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'dlagVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the variance
'd2lagMeanQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the mean of
'd2lagVarQdiff'	The quantile difference (using deciles) between the surrogate and the real distribution of the variance

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

William J. Conover (1971). Practical Nonparametric Statistics. New York: John Wiley & Sons. Pages 295-301 (one-sample Kolmogorov test), 309-314 (two-sample Smirnov test).

See Also

wccHeatmap visualizes results selected aggregated statistics for combinations of parameters. See wccAggregate for definitions of the statistics that are compared between surrogate and real dyads. See ks.test for definition of the two tailed Kolmogorov-Smirnov Test.

Examples

```
# Create two arrays of timeseries with dyadic dependence
array1 <- matrix(NA, nrow=10, ncol=500)
array2 <- matrix(NA, nrow=10, ncol=500)
for(i in 1:10) {
    array1[i,] <- sin(c(1:500)/runif(1, min=5, max=20)) + rnorm(500, mean=0, sd=.5)
    array2[i,] <- array1[i,] + rnorm(500, mean=0, sd=.5)
}
# Select parameters to explore
wMaxVec <- c(50,100)
```

```
tMaxVec <- c(25,50)
wccFindDyadParam(inArray1=array1, inArray2=array2, wMaxvector=wMaxVec, tMaxvector=tMaxVec,
    nSurrogates=30, samplespersecond=1, method="c")
```

---

wccGLLAEmbed *wccGLLAEmbed*

Description

This function creates a time delay embedded matrix from a time series.

Usage

```
wccGLLAEmbed(x, embed=4, tau=1, groupby=NA, label="x", idColumn=TRUE)
```

Arguments

x	A numeric vector.
embed	A numeric wholenumber value indicating the embedding dimension, the number of columns in the target time delay embedded matrix. embed must be greater than or equal to order + 1
tau	An numeric wholenumber value indicating the number of samples by which each column in the time delay embedded matrix is lagged. tau must be greater than or equal to 1.
groupby	If NA, then do not group the rows of the output matrix. If not missing, it is a numeric wholenumber vector of the same length as x. Each element in the vector x is treated as belonging to the group specified by the wholenumber in the matching element of groupby.
label	A character string specifying the label prefix for each column of the output matrix. The number of the column is pasted onto this prefix so that columns have unique names.
idColumn	A logical specifying whether to rbind the group as specified by groupby to the first column of the time delay embedded matrix. This column is always named "ID"

Details

wccGLLAEmbed creates a time delay embedded matrix from a time series. A time delay embedded matrix has rows corresponding to the samples of elapsed time in the time series and columns that are lagged by tau samples from each other. For example, for a time series x of length N composed of a single group and with embed = 3 and tau=1, the time delay embedding matrix would have N-2 rows and 3 columns. Column 1 would contain x[1:(N-2)], column 2 would contain x[2:(N-1)], and column 3 would contain x[3:N]. Thus each row contains a time lagged vector representing not only the state of the system at a moment in time, but also a snapshot of the dynamics of the system at that moment. The number of columns in a N x D time delay embedding matrix is often referred

to as the embedding dimension of the matrix, since each row can be thought of as a point in a D dimensional space.

When multiple groups' or individuals' time series are time delay embedded, one must prevent the data from group 1 to overlap on the same row with data from group 2 and so forth. Thus, if a group has N elements, the resulting submatrix for that group will have embed columns and N - ((embed - 1) * tau) rows. If any group has less than (tau + 1 + ((embed - 2) * tau)) elements then that group will not appear in the time delay embedding matrix since there is not sufficient data in the time series to compose 1 row.

Missing values are retained in the output time delay embedded matrix. This is useful for methods such as latent differential equations where full information maximum likelihood can account for the missingness, but will result in missing valued rows in the output of Generalized Local Linear Approximation as calculated using the wccGLLAWMatrix function.

Time delay embedding is a technique that represents a time series in a state space that contains both the instantaneous state of a system and its local dynamics. Takens (1981) showed that time delay embedding captures all of the dynamics in a system if the embedding dimension is sufficient. Choosing the appropriate embedding dimension for representing the dynamics of a system is something of an art (Sauer, Yorke, and Casdagli, 1991). For calculating time series deriviatives using Generalized Local Linear Approximation, the minimum embedding dimension is recommended to be two plus the order of the highest derivative to be calculated. Higher embedding dimensions than required will perform smoothing on the time series, which may be useful, but will also attenuate the values of the derivatives calculated.

Value

Returns a time delay embedding matrix. Optionally this includes a column of ID numbers representing the group to which each row belongs.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

Boker, S. M., Deboeck, P. R., Edler, C., & Keel, P. K. (2010) Generalized Local Linear Approximation of Derivatives from Time Series. In Statistical Methods for Modeling Human Dynamics: An Interdisciplinary Dialogue, S.-M. Chow & E. Ferrar (Eds). Boca Raton, FL: Taylor & Francis, pp 179–212.

Sauer, T., Yorke, J., & Casdagli, M. (1991). Embedology. Journal of Statistical Physics, 65(3,4), 95-116.

Takens, F. (1981). Detecting strange attractors in turbulence. In D. Rand & L.-S. Young(Eds.), Dynamical systems and turbulence, warwick 1980: Proceedings of a symposium held at the University of Warwick (pp. 366-381). Berlin: Springer-Verlag.

See Also

wccGLLAWMatrix for the definition of the matrix required to estimate derivatives using Generalized Local Linear Approximation.

Examples

```
# Create a time delay embedding matrix from a single time series.
theSeries <- sin((1:20)/10)
length(theSeries)
embeddedMatrix <- wccGLLAEmbed(theSeries, embed=4, tau=1, label="x", idColumn=FALSE)
dim(embeddedMatrix)
# Calculate derivatives for the series
embeddedMatrix %*% wccGLLAWMatrix(embed=4, tau=1, deltaT=1, order=2)


theFrame <- data.frame(theSeries=c(sin((1:20)/10)), sin((1:20)/15),theGroups=c(rep(1,20),rep(2,20)))
dim(theFrame)
embeddedMatrix <- wccGLLAEmbed(theFrame$theSeries, embed=4, tau=1, label="x",
     groupby=theFrame$theGroups, idColumn=TRUE)
dim(embeddedMatrix)
# Calculate derivatives for the series
embeddedMatrix[,2:5] %*% wccGLLAWMatrix(embed=4, tau=1, deltaT=1, order=2)
```

`wccGLLAWMatrix` *W Matrix*

Description

This function returns a matrix, W, which when premultiplied by a time delay embedded matrix will calculate the time derivatives of the time series in the time delay embedded matrix.

Usage

```
wccGLLAWMatrix(embed=NA, tau=NA, deltaT=1, order=2)
```

Arguments

`embed`	A numeric wholenumber value indicating the embedding dimension, the number of columns in the target time delay embedded matrix. embed must be greater than or equal to order + 1. 'embed' must be equal to that given to 'wccGLLAEmbed' to create the target time delay embedded matrix.
`tau`	An numeric wholenumber value indicating the number of samples by which each column in the time delay embedded matrix is lagged. 'tau' must be greater than or equal to 1. 'tau' must be equal to that given to 'wccGLLAEmbed' to create the target time delay embedded matrix.
`deltaT`	A numeric value indicating the elapsed time between samples in the time series that was time delay embedded. deltaT must be greater than 0.
`order`	A numeric wholenumber value indicating the highest derivative to be estimated when the W matrix is premultiplied by the time delay embedding matrix.

Details

The W matrix is defined by the Generalized Local Linear Approximation method for calculating derivatives of time series. First time delay embed the time series using 'wccGLLAEmbed' . Then call 'wccGLLAWMatrix' to return the appropriate W matrix for that embedding. Finally postmultiply the time delay embedding matrix by the W matrix, which creates an output matrix whose rows match the rows of the time delay embedded matrix and whose columns are the calculated derivatives of the time series.

Value

Returns the W matrix that can postmultiply a time delay embedded matrix to calculate derivatives.

References

Boker, S. M., Deboeck, P. R., Edler, C., & Keel, P. K. (2010) Generalized Local Linear Approximation of Derivatives from Time Series. In Statistical Methods for Modeling Human Dynamics: An Interdisciplinary Dialogue, S.-M. Chow & E. Ferrar (Eds). Boca Raton, FL: Taylor & Francis, pp 179–212.

See Also

wccGLLAEmbed for creating a time delay embedding matrix.

Examples

```
# Create a time delay embedding matrix from a single time series.
theSeries <- sin((1:20)/10)
length(theSeries)
embeddedMatrix <- wccGLLAEmbed(theSeries, embed=4, tau=1, label="x", idColumn=FALSE)
dim(embeddedMatrix)
# Calculate derivatives for the series
embeddedMatrix %*% wccGLLAWMatrix(embed=4, tau=1, deltaT=1, order=2)
```

`wccHeatmap` *wccHeatmap*

Description

This function plots a heatmap of a chosen aggregate statistic from the results of `wccFindDyadParam`

Usage

```
wccHeatmap(xparam=NA, yparam=NA, aggstat=NA, xlabel=NA, ylabel=NA, pdffile=NA)
```

Arguments

xparam	One of the parameter columns from the dataframe returned by wccFindDyadParam. This parameter will be used as the x axis of the heatmap plot.
yparam	One of the parameter columns from the dataframe returned by wccFindDyadParam. This parameter will be used as the y axis of the heatmap plot.
aggstat	One of the aggregate statistics columns from the dataframe returned by wccFindDyadParam. aggstat will be the value that determines the color in each cell of the heatmap.
xlabel	A character string that will be used as the label for the x axis.
ylabel	A character string that will be used as the label for the y axis.
pdffile	A character string with the filename of a pdf file to create. If pdffile=NA then the plot is sent to the default device.

Details

This function produces a heat map plot of selected aggregated statistics returned by wccFindDyadParam for combinations of selected parameter values. First run wccFindDyadParam using two or more parameter vectors for a grid search. Next select two colums of parameters and one aggregated statistic to plot and send it to wccHeatmap. If the optional parameter pdffile exists, then a publication quality pdf file will be saved. Otherwise the heatmap will be displayed on the default device.

Value

Returns nothing.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

See Also

wccFindDyadParam for the definition of the dataframe columns used by wccHeatmap.

Examples

```
# Create two arrays of timeseries with dyadic dependence
array1 <- matrix(NA, nrow=10, ncol=500)
array2 <- matrix(NA, nrow=10, ncol=500)
for(i in 1:10) {
    array1[i,] <- sin(c(1:500)/runif(1, min=5, max=20)) + rnorm(500, mean=0, sd=.5)
    array2[i,] <- array1[i,] + rnorm(500, mean=0, sd=.5)
}
# Select parameters to explore
wMaxVec <- c(50,100)
tMaxVec <- c(25,50)
wccFDPout <- wccFindDyadParam(inArray1=array1, inArray2=array2, wMaxvector=wMaxVec,
```

```
    tMaxvector=tMaxVec, nSurrogates=30, samplespersecond=1, method="r")
# Plot the heatmap
wccHeatmap(xparam=wccFDPout$wMax, yparam=wccFDPout$tMax, aggstat=wccFDPout$maxMean,
    xlabel="Window Max", ylabel="Max Lag")
```

---

wccPeakPick *wccPeakPick*

Description

This function finds the maximum or minimum correlation peak nearest to zero lag for each row in a windowed crosscorrelation matrix.

Usage

```
wccPeakPick(tAllCor=NA, Lsize=8, pspan=.25, type="Max")
```

Arguments

tAllCor	A numeric matrix returned by the wccCalc function.
Lsize	A numeric value for the required width of the peak (default=8). For instance, Lsize = 8 means that a peak must have 4 values descending on either side in order to be called a peak. Thus for elapsed time T and lag offset L if a peak is found at tAllCor[T,L] then tAllCor[T,(L-1):(L-Lsize)] < tAllCor[T,L] and tAllCor[T(L+1):(L+Lsize)] < tAllCor[T,L]. An Lsize that is too small may find spurious peaks that are just noise, whereas an Lsize that is too large may miss actual peaks.
pspan	A numeric value used by the loess function to indicate what proportion of data to use in order to do the loess smooth (default=.25).
type	A character string (default='Max'). 'Max' indicates that the maximum correlation closest to zero lag should be picked whereas 'Min' indicates that the minimum correlation closest to zero lag should be picked.

Details

This function operates on the result of wccCalc, a windowed crosscorrelation matrix in order to find the peak correlation closest to a lag of zero. The lag and the value of the peak correlation are returned for each row of the windowed crosscorrelation matrix, i.e., for the elapsed time of each window calculated by wccCalc. The wccCalc windows are sampled at time lagged intervals from one another so that the association between the time series can be estimated even though the time lag of maximum association between the two time series may be itself time varying. For instance, when two individuals converse, their body motions may be associated such that the leader and follower take turns as the two individuals swap roles as speaker and listener.

This same kind of nonstationary association has been observed between many physiological time series. At any particular elapsed time, the maximum correlation closest to a lag of zero (whether a positive or negative lag) is taken to be the lagged offset between the two time series. For series that

have oscillatory structure (such as head nodding in conversation) the maximum correlation closest to a lag of zero can be interpreted as the phase lag between the series. The variability in the lag is one measure of the degree of nonstationarity present in the association between two time series.

Value

Returns a list with two vectors: `maxIndex` is the lag of the peak for each window and `maxValue` is the calculated correlation at that peak. If `wccPeakPick` is called with type = 'Min' then the returned vectors are named `minIndex` and `minValue`. If no peak is found within the row tAllCor[T,] then maxIndex[T] and maxValue[T] are returned as NAs

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

See Also

`wccCalc` for the definition of the windowed crosscorrelation matrix. `loess` for the definition of the smoothing function.

Examples

```
#Calculate windowed cross correlation between two time series
tSeries1 <- sin(c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
tSeries2 <- sin(c(1:1000)/15) + rnorm(1000, mean=0, sd=.5)
wccCalcOut <- wccCalc(inSeries1=tSeries1, inSeries2=tSeries2, wMax=50, tMax=50, wInc=1,
     tInc=1, method="r")

#Calculate the peak picking vectors
wccPeakPick(wccCalcOut, Lsize=8, pspan=.25, type="Max")
```

---

wccPlot *wccPlot*

Description

This function plots a Windowed Cross Correlation heatmap.

Usage

```
wccPlot(inSeries1=NA, inSeries2=NA, startwindow=1, endwindow=200, wMax=50, tMax=50,
    wInc=1, tInc=1, Lsize=8, pspan=.25, type="Max", samplespersecond=1,
    method=c("c", "cumr", "cumc", "r"), ...)
```

Arguments

inSeries1	A vector of numeric values. This is the first time series on which windowed cross correlation is calculated. When inSeries1 is leading inSeries2, the peak correlation closest to zero in negative in wccPlot. Note that inSeries1 must be of the same length as inSeries2. Note that the only difference between inSeries1 and inSeries2 is which is treated as positive lag and which is negative lag in wccPlot.
inSeries2	A vector of numeric values. This is the second time series on which windowed cross correlation is calculated. Must be of the same length as inSeries1. When inSeries2 is leading inSeries1, the peak correlation closest to zero in positive in wccPlot.
startwindow	A numeric value specifying the first window to plot (default=1). Windows are plotted as vertical stripes in the heatmap and the elapsed time of this window is wMax + tMax + (startwindow * wInc / samplespersecond)
endwindow	A numeric value specifying the last window to plot (default=200).
wMax	A numeric indicating the number of samples in a window (default=50). This must be greater or equal to 5 samples. It is recommended that wMax be greater than 15.
tMax	The maximum number of samples to lag the windows (default=50). The cross correlation is calculated for windows that are time lagged against each other, allowing for processes than require some time to evolve. For instance, if the two time series are from motion capture of two individuals conversing with one another, the speaker's movements are likely to occur prior to the responses of the listener. tMax should be set to be greater than the number of samples that represents the maximum time lag that is likely to occur.
wInc	The number of samples incremented between successive windows (default=1). This is most often set to 1. If the sample rate is very high and the time series is long, wInc may be increased. However, wccCalc's maximum sensitivity to nonstationary change in the time series occurs when wInc is 1.
tInc	The number of samples incremented between successive lags (default=1). This is most often set to 1. If the sample rate is high and the maximum lag is high then tInc may be increased. However, wccCalc's maximum sensitivity to changes in lags occurs when the tInc is set to 1.
Lsize	An integer specifying the width required for a peak to be identified by wccPeakPick (default=8). An Lsize that is too small may find spurious peaks that are just noise, whereas an Lsize that is too large may miss actual peaks.
pspan	A numeric specifying the amount of smoothing applied by the loess function within wccPeakPick (default=.25).
type	A character string indicating whether a maximum, 'Max', or minimum, 'Min', should be detected by wccPeakPick (default='Max'). If type='Zero' then a topographic map is generated rather than a line from wccPeakPickwccPeakPick.
samplespersecond	A numeric indicating the sampling rate of the time series. This will determine the units of the first derivative aggregate statistics. (default=1)

method	Backend passed through to wccCalc: "c" (default, original compiled windcross), "r" (original pure-R loop), "cumc" (cumulative-sum C backend), or "cumr" (cumulative-sum pure-R reference). See wccCalc for details and the missing-data caveat for the cum* methods.
...	Reserved for the deprecated windcross=TRUE/FALSE argument, mapped to method="c" / method="r" with a warning.

Details

This function runs wccCalc and wccPeakPick using the arguments passed in and plots a heatmap of a selected section of windows. The vertical axis is labeled with the lags in the units of seconds. Each calculated window is plotted as a vertical slice of the heat map and correlations near 1 are plotted in yellow and correlations near -1 are plotted in red. Lags of inSeries1 are plotted as positive and lags of inSeries2 are plotted as negative lags. The horizontal black line in the middle of the plot represents a lag of zero. Thus, if a feature in inSeries1 occurs prior to a similar feature in inSeries2, there will be a positive correlation shown at a negative lag, i.e., below the horizontal zero lag line. The result of wccPeakPick is overlayed as a black line that traces the peak correlation closest to a lag of zero. The horizontal axis is labeled with the elapsed time to the right edge of two windows when they are at zero lag.

It is recommended that the plot includes no more than 1000 windows since if a large number of windows are plotted, the resulting heatmap will overwrite one window with the next window. Other considerations include whether the interval of time of interest is near the beginning or end of the timeseries. The elapsed time associated with the first window will be tMax*tInc*samplespersecond. Thus an elapsed time of 0 will never occur. It is recommended that the startwindow and endwindow be set to bracket a sample of the timeseries that is of interest to the researcher. If endwindow - startwindow is about 500, a balance between sensitivity to nonstationarity and plotting resolution is achieved.

Value

Returns nothing.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

See Also

wccCalc and wccPeakPick for the calculation of the wccCalc matrix and wccPeakPick timeseries that are used by this function.

Examples

```
#Create a windowed cross correlation plot for two simulated time series with a phase offset
tSeries1 <- sin(c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
tSeries2 <- sin(1+c(1:1000)/10) + rnorm(1000, mean=0, sd=.5)
wccPlot(inSeries1=tSeries1, inSeries2=tSeries2, startwindow=1, endwindow=500,
```

```
wMax=100, tMax=100, wInc=1, tInc=1, Lsize=8, pspan=.25, type="Max",
samplespersecond=10)
```

---

wccSurrogateDyads *wccSurrogateDyads*

Description

This function randomly pairs timeseries and creates a distribution of the null hypothesis that the pairing of dyads does not matter.

Usage

```
wccSurrogateDyads(inArray1=NA, inArray2=NA, wMax=50, tMax=50, wInc=1, tInc=1, Lsize=8,
     pspan=.25, type="Max", nSurrogates=NA,
     method=c("c", "cumr", "cumc", "r"), embedD=9, ...)
```

Arguments

inArray1	An N by T matrix of numeric values representing N timeseries of length T. These are the first set of time series which are randomly paired with timeseries from inArray2 and on which windowed cross correlation are calculated. Must be of the same order N by T as inArray2. Note that the only difference between inArray1 and inArray2 is which is treated as positive lag and which is negative lag in wccPlot. When inArray1 is leading inArray2, the peak correlation closest to zero in negative in wccPlot.
inArray2	An N by T matrix of numeric values representing N timeseries of length T. These are the second set of time series which are randomly paired with timeseries from inArray1 and on which windowed cross correlation are calculated. Must be of the same order N by T as inArray1. When inArray2 is leading inArray1, the peak correlation closest to zero in positive in wccPlot.
wMax	A numeric indicating the number of samples in a window (default=50). This must be greater or equal to 5 samples. It is recommended that wMax be greater than 15.
tMax	The maximum number of samples to lag the windows (default=50). The cross correlation is calculated for windows that are time lagged against each other, allowing for processes than require some time to evolve. For instance, if the two time series are from motion capture of two individuals conversing with one another, the speaker's movements are likely to occur prior to the responses of the listener. tMax should be set to be greater than the number of samples that represents the maximum time lag that is likely to occur.
wInc	The number of samples incremented between successive windows (default=1). This is most often set to 1. If the sample rate is very high and the time series is long, wInc may be increased. However, wccCalc's maximum sensitivity to nonstationary change in the time series occurs when wInc is 1.

tInc	The number of samples incremented between successive lags (default=1). This is most often set to 1. If the sample rate is high and the maximum lag is high then tInc may be increased. However, wccCalc's maximum sensitivity to changes in lags occurs when the tInc is set to 1.
Lsize	An integer specifying the width required for a peak to be identified by wccPeakPick (default=8). An Lsize that is too small may find spurious peaks that are just noise, whereas an Lsize that is too large may miss actual peaks.
pspan	A numeric specifying the amount of smoothing applied by the loess function within wccPeakPick (default=.25).
type	A character string indicating whether a maximum, 'Max', or minimum, 'Min', should be detected by wccPeakPick (default='Max').
nSurrogates	A numeric indicating the number of surrogates to generate (default=NA). If there are N timeseries in inArray1, then the maximum number of unique surrogates that can be generated is N*(N-1)/2
method	Backend passed through to wccCalc: "c" (default, original compiled windcross), "r" (original pure-R loop), "cumc" (cumulative-sum C backend), or "cumr" (cumulative-sum pure-R reference). See wccCalc for details and the missing-data caveat for the cum* methods.
embedD	A numeric indicating the embedding dimension passed to wccGLLAEmbed to calculate derivatives of the peak picking lag timeseries. The default (9) will provide a medium smooth at the cost of some loss of resolution in the timing of jumps in the second derivative. The smallest legal value is 5, which provides less smooth and the best time resolution. If your sampling rate is low, then a smaller value of embedD is useful, while if your sampling rate is high, a larger value of embedD will provide better noise rejection by smoothing.
...	Reserved for the deprecated windcross=TRUE/FALSE argument, mapped to method="c" / method="r" with a warning.

Details

This function generates and returns a distribution of aggregated statistics from running Windowed Cross Correlation and peakpicking (Boker, Rotondo, Xu, & King 2002) on pairs of timeseries that conform to the null hypothesis that the true pairing of the timeseries does not matter. This is called a surrogate analysis (Moulder, Boker, Ramsayer, & Tschacher 2018) and is a nonparametric way of estimating whether the observed association between timeseries is due to dyadic interactions occuring while the timeseries were measured or simply due to the distributions of scores within all the timeseries. This is important to consider since the timeseries may be measurements of individual processes that are constrained in some way. For instance, consider dyadic timeseries of head motions. All persons' heads are constrained in the motions they can perform due to similarities in skeletal and muscular dynamics that are shared by most humans. One does not want these shared constraints to show up as significant associations within dyads when they are actually due to similarities between individuals.

The function returns a dataframe whose columns contain different ways of aggregating the associations and each row contains results from one random pairing of dyads. The randomization allows reuse of dyads, so it is best if the number of possible random pairings is large relative to the number of surrogates requested. If there are N rows in inArray1 and inArray2, the maximum number of

unique surrogates that can be generated is N*(N-1)/2. With that constraint, we recommend at least N = 10 dyads as a source pool since that allows 45 unique surrogates to be generated. The only exclusion to the random pairing of dyads is that if the random pairing results in an actual dyad, it is not included.

Value

Returns a dataframe whose columns are defined in wccAggregate and whose rows are the results of each random pairing of dyads.

References

Boker, S. M., Rotondo, J. L., Xu, M., & King, K. (2002). Windowed cross-correlation and peak picking for the analysis of variability in the association between behavioral time series. Psychological methods, 7(3), 338.

Moulder, R., Boker, S., Ramseyer, F., & Tschacher, W. (2018). Determining synchrony between behavioral time series: An application of surrogate data generation for establishing falsifiable null-hypotheses. Psychological Methods. 23:4 pp 757–773

See Also

wccAggregate for the definition of the columns of the returned dataframe.

Examples

```
#Create two arrays of timeseries with dyadic dependence
array1 <- matrix(NA, nrow=10, ncol=500)
array2 <- matrix(NA, nrow=10, ncol=500)
for(i in 1:10) {
    array1[i,] <- sin(c(1:500)/runif(1, min=5, max=20)) + rnorm(500, mean=0, sd=.5)
    array2[i,] <- array1[i,] + rnorm(500, mean=0, sd=.5)
}
wccSurrogateDyads(inArray1=array1, inArray2=array2, wMax=50, tMax=50, wInc=1, tInc=1,
    Lsize=8, pspan=.25, type="Max", nSurrogates=20, method="c")
```

Index