

Package: PhysioCrossModal (via r-universe)

October 3, 2026

Title Cross-Modal Coupling Analysis for PhysioExperiment Objects

Version 0.8.4

Description Provides cross-modal coupling, connectivity, and synchrony analysis between physiological signals of different modalities (EEG, EMG, ECG, EDA, MoCap, fNIRS, etc.). Introduces a MultiPhysioExperiment container class for holding multiple PhysioExperiment objects at different sampling rates with temporal alignment. Includes spectral coherence, phase synchrony (PLV, PLI, wPLI), directed coupling (Granger causality), and time-domain coupling (cross-correlation) methods.

Depends R (>= 4.2), PhysioExperiment (>= 2.1.3)

Imports methods, parallel, SummarizedExperiment, S4Vectors, stats, digest

Suggests PhysioCore, signal, PhysioPreprocess, PhysioEEG, fdasrvf, testthat (>= 3.2.0), ggplot2, knitr, rmarkdown

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Collate 'PhysioCrossModal-package.R' 'reexports.R'

'MultiPhysioExperiment-class.R'

'MultiPhysioExperiment-methods.R' 'utils-spectral.R'

'align-elastic.R' 'align-signals.R' 'coupling-spectral.R'

'coupling-phase.R' 'granger-nonparametric.R'

'coupling-directed.R' 'coupling-pac.R' 'coupling-frontier.R'

'coupling-leida-metrics.R' 'coupling-timedom.R'

'coupling-wavelet.R' 'coupling-xwt.R' 'coupling-wrapper.R'

'coupling-matrix.R' 'stats-significance.R' 'benchmark-lodo.R'

'crossmodal-data.R' 'vis-coupling.R' 'fusion-multiblock.R'

'fusion-twoblock.R' 'fusion-coupled.R' 'fusion-association.R'

'fusion-index.R' 'zzz.R'

RoxygenNote 7.3.3

URL <https://github.com/x-biosignal/PhysioCrossModal>,
<https://x-biosignal.r-universe.dev/PhysioCrossModal>,
<https://x-biosignal.github.io/PhysioCrossModal>

BugReports <https://github.com/x-biosignal/PhysioCrossModal/issues>

Config/pak/sysreqs zlib1g-dev

Repository <https://x-biosignal.r-universe.dev>

Date/Publication 2026-10-03 13:11:45 UTC

RemoteUrl <https://github.com/x-biosignal/PhysioCrossModal>

RemoteRef HEAD

RemoteSha 98d2b0f224738de11b607e24c889e01bfb7b82da

Contents

alignObservationBlocks	3
alignSignals	4
alignToRate	6
bootstrapCI	7
cca	8
ciPLV	9
coherence	10
coherenceMatrix	12
comodulogram	13
coupledNMF	15
couplingAnalysis	16
couplingAnalysisCached	18
couplingMatrix	19
couplingResults	20
crossBlockFactor	21
crossCorrelation	22
crossSpectrum	24
crossWaveletTransform	25
directedTransferFunction	26
distanceCorrelation	28
elasticAlign	29
fuseBlocks	30
grangerCausality	31
jive	34
laggedCoherence	35
leidaStates	36
leidaTransitions	38
lodoGeneralization	38
make_coupled_signals	40
make_directed_signals	41
make_eeg_emg	42

mergePhysio	44
modulationIndex	45
multimodalGaitIndex	47
MultiPhysioExperiment-crossmodal	48
MultiPhysioExperiment-crossmodal-methods	49
multipleFactorAnalysis	49
multitaperCoherence	50
mutualInformation	52
orthogonalizedAEC	53
pairwisePhaseConsistency	54
partialDirectedCoherence	55
phaseAmplitudeCoupling	56
phaseLagIndex	58
phaseLockingValue	59
phaseSlopeIndex	61
plotCoherenceSpectrum	62
plotComodulogram	63
plotCouplingMatrix	64
plotCouplingTimecourse	65
plotWaveletCoherence	66
plsBlocks	67
representationalSimilarity	68
rvCoefficient	69
slidingCrossCorrelation	70
srvfMean	71
surrogateMatrixTest	72
surrogateTest	74
transferEntropy	76
warpApply	77
waveletCoherence	78
waveletPLV	79
weightedPLI	81
Index	83

alignObservationBlocks

Align feature blocks using explicit observation keys

Description

Establishes row correspondence before positional multiblock analyses such as `jive()` or `multipleFactorAnalysis()`. Keys may describe, for example, participant, trial and cycle together. Their meaning and granularity must be chosen by the caller; equality does not verify units, timing or independence.

Usage

```
alignObservationBlocks(blocks, keys, by, missing = c("error", "intersection"))
```

Arguments

blocks	Named list of at least two finite numeric matrices, with observations in rows and features in columns.
keys	Named list of data.frames in the same order as blocks. Each table has one row per observation in its corresponding block.
by	Nonempty character vector naming the composite key columns. Columns must be plain character, integer, numeric or logical vectors, with identical types across blocks, without missing or non-finite values.
missing	Policy for unequal observation sets: "error" (default) or explicit "intersection". No imputation or aggregation is performed.

Value

A list with blocks aligned in first-block observation order, observations containing their composite keys, row_index mapping output rows to each input block, excluded_rows listing omitted input row indices, and by and missing recording the chosen contract. Duplicate keys are always errors. The caller should retain the input keys with this mapping.

Examples

```
a <- matrix(1:6, nrow = 3)
b <- matrix(7:12, nrow = 3)
ids <- data.frame(participant = c("P1", "P2", "P3"))
aligned <- alignObservationBlocks(
  list(emg = a, motion = b[3:1, , drop = FALSE]),
  list(emg = ids, motion = ids[3:1, , drop = FALSE]), "participant")
```

alignSignals	<i>Align multiple PhysioExperiment objects to a common sampling rate</i>
--------------	--

Description

Takes two or more named PhysioExperiment objects and resamples them so they all share a single sampling rate, then wraps the result in a [MultiPhysioExperiment](#).

Usage

```
alignSignals(
  ...,
  method = c("lowest_rate", "common_rate", "resample", "elastic"),
  target_rate = NULL
)
```

Arguments

<code>...</code>	Named <code>PhysioExperiment</code> objects.
<code>method</code>	Character: one of "lowest_rate" (default), "common_rate", "resample", or "elastic".
<code>target_rate</code>	Numeric scalar: required when <code>method = "resample"</code> .

Details

Three alignment strategies are available:

"lowest_rate" (default) Resample all signals to the lowest sampling rate present.

"common_rate" Resample all signals to the highest sampling rate present.

"resample" Resample all signals to the rate given by `target_rate` (which must be supplied).

"elastic" Resample to the lowest rate, then additionally apply an elastic (SRVF) time warping of every object onto the first, so that shared events line up in phase. The warping is computed from each object's first channel and carried to all its channels; it is stored in `metadata()$elastic_warp`. Objects whose duration differs from the reference after rate alignment are left un-warped. See [elasticAlign\(\)](#).

If every input already shares the same sampling rate, no resampling is performed regardless of the chosen method (elastic warping, if requested, is still applied).

Value

A [MultiPhysioExperiment](#) containing the (possibly resampled) input objects.

See Also

[alignToRate\(\)](#), [mergePhysio\(\)](#), [MultiPhysioExperiment\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
eeg <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(500 * 2), nrow = 500)),
  samplingRate = 500
)
emg <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000 * 2), nrow = 1000)),
  samplingRate = 1000
)
mpe <- alignSignals(EEG = eeg, EMG = emg, method = "lowest_rate")
```

alignToRate

*Resample a PhysioExperiment to a target sampling rate***Description**

Resamples all assays in a `PhysioExperiment` object so that the resulting data correspond to the specified target sampling rate. If the **PhysioPreprocess** package is available and `method = "linear"`, its `resample()` function is used; otherwise an internal interpolation is applied.

Usage

```
alignToRate(x, target_rate, method = c("linear", "spline", "fft"))
```

Arguments

<code>x</code>	A <code>PhysioExperiment</code> object.
<code>target_rate</code>	Numeric scalar: desired sampling rate in Hz.
<code>method</code>	Character: resampling method. One of "linear" (default), "spline", or "fft".

Value

A new `PhysioExperiment` with resampled data and updated `samplingRate`. The assay name is preserved from the original.

See Also

[alignSignals\(\)](#), [mergePhysio\(\)](#), [MultiPhysioExperiment\(\)](#)

Examples

```
eeg_data <- matrix(rnorm(1000 * 4), nrow = 1000, ncol = 4)
pe <- PhysioExperiment(
  assays = list(raw = eeg_data),
  colData = S4Vectors::DataFrame(
    label = paste0("Ch", 1:4),
    type = rep("EEG", 4)
  ),
  samplingRate = 1000
)
pe500 <- alignToRate(pe, target_rate = 500)
samplingRate(pe500) # 500
```

bootstrapCI

*Bootstrap confidence interval for coupling***Description**

Computes a bootstrap confidence interval for the coupling statistic between two signals using the moving-block bootstrap (to preserve temporal autocorrelation).

Usage

```
bootstrapCI(
  x,
  y = NULL,
  sr = NULL,
  method,
  n_boot = 199L,
  ci = 0.95,
  block_len = NULL,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  cores = 1L,
  ...
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when x is an MPE.
sr	Numeric sampling rate in Hz (required when x/y are numeric).
method	Character coupling method (same options as surrogateTest).
n_boot	Integer number of bootstrap replicates (default 199).
ci	Numeric confidence level (default 0.95).
block_len	Integer block length for block bootstrap, or <code>NULL</code> for automatic (<code>ceiling(sqrt(n))</code>).
modality_x, modality_y	Character modality names for MPE input.
channels_x, channels_y	Integer channel indices (default 1).
cores	Integer number of parallel cores to use (default 1L). When <code>cores > 1</code> , bootstrap replicates are computed in parallel.
...	Additional arguments passed to the coupling function.

Value

A list with components:

observed The full coupling result from the original signals.

statistic Numeric scalar: the extracted coupling statistic.

ci_lower Numeric scalar: lower CI bound.

ci_upper Numeric scalar: upper CI bound.

ci_level Numeric scalar: confidence level used.

boot_distribution Numeric vector of bootstrap statistics.

References

Efron, B., & Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall/CRC.

See Also

[surrogateTest\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
result <- bootstrapCI(x, y, sr = sr, method = "coherence",
                      n_boot = 19, nperseg = 128L)
c(result$ci_lower, result$ci_upper)
```

cca

Canonical correlation analysis between two modalities

Description

Finds the linear combinations of two modality blocks whose scores are maximally correlated – the directions in which the modalities co-vary. Ridge regularisation keeps it well-posed when a block has more features than observations.

Usage

```
cca(X, Y, ncomp = NULL, lambda_x = 0, lambda_y = 0)
```

Arguments

X, Y	Matrices with the same n rows (observations); columns are the features of each modality.
ncomp	Number of canonical pairs (default: $\min(\text{ncol}(X), \text{ncol}(Y))$).
lambda_x, lambda_y	Ridge penalties; if 0 (default) a small ridge is added automatically when a block is rank-deficient (features $\geq$ n).

Value

a cca object: cor (canonical correlations, decreasing), xcoef, ycoef (canonical vectors), xscores, yscores (canonical variates).

References

Hotelling H (1936); Vinod HD (1976) ridge CCA.

See Also

[plsBlocks\(\)](#), [rvCoefficient\(\)](#)

Examples

```
set.seed(1); z <- matrix(rnorm(50 * 2), 50, 2)
X <- z %%% matrix(rnorm(2 * 5), 2, 5) + matrix(rnorm(50 * 5, 0, .3), 50, 5)
Y <- z %%% matrix(rnorm(2 * 4), 2, 4) + matrix(rnorm(50 * 4, 0, .3), 50, 4)
cca(X, Y)$cor
```

ciPLV

Corrected imaginary phase-locking value (ciPLV)

Description

ciPLV (Bruna et al. 2018) corrects the imaginary part of the PLV for the size of its real part, giving a phase-synchrony estimate that is robust to zero-lag (volume-conduction) coupling.

Usage

```
ciPLV(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  order = 4L,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x, y	Signals (or a single object from which a pair is extracted).
sr	Sampling rate in Hz.
freq_band	Numeric c(low, high) band in Hz.
order	Butterworth filter order for the band-pass (default 4).
modality_x, modality_y, channels_x, channels_y	Passed to the internal signal-pair extractor.

Value

A single ciPLV magnitude in $[0, 1]$.

References

Bruna R, et al. (2018). Phase locking value revisited. *J Neural Eng* 15:056011.

coherence	<i>Magnitude-squared coherence between two signals</i>
-----------	--

Description

Computes the magnitude-squared coherence between two signals, which quantifies the linear frequency-domain relationship between them. The coherence is defined as:

$$C_{xy}(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f) \cdot S_{yy}(f)}$$

where S_{xy} is the cross-spectral density and S_{xx}, S_{yy} are the auto-spectral densities.

Usage

```
coherence(
  x,
  y = NULL,
  sr = NULL,
  freq_range = NULL,
  nperseg = 256L,
  noverlap = NULL,
  method = c("welch", "multitaper"),
  type = c("magnitude", "imaginary", "lagged"),
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	Numeric vector, PhysioExperiment, or MultiPhysioExperiment.
y	Numeric vector or PhysioExperiment, or NULL when x is an MPE.
sr	Numeric sampling rate in Hz (required when x/y are numeric).
freq_range	Numeric vector of length 2 giving the frequency range c(low, high) in Hz to restrict the output. NULL returns all frequencies.
nperseg	Integer segment length for Welch's method (default 256).
noverlap	Integer overlap between segments, or NULL for floor(nperseg / 2).

method	Character spectral estimation method. Currently only "welch" is implemented; "multitaper" is reserved for future use.
type	Character coherence type: "magnitude" (magnitude-squared coherence, the default), "imaginary" (the imaginary part of coherency, insensitive to zero-lag/volume-conduction coupling; Nolte et al., 2004), or "lagged" (Pascual-Marqui, 2007: $\text{Im}(C) / \sqrt{1 - \text{Re}(C)^2}$ with C the complex coherency).
modality_x	Character modality name in MPE for the x signal.
modality_y	Character modality name in MPE for the y signal.
channels_x	Integer which channel to extract from x (default 1).
channels_y	Integer which channel to extract from y (default 1).

Details

Unlike the `coherence()` function in `PhysioAnalysis` (which computes coherence across all channel pairs within a single `PhysioExperiment`), this function is designed for cross-modal analysis: it takes two separate signals (or `PhysioExperiment` / `MultiPhysioExperiment` objects) and computes the pairwise coherence between them.

Value

A named list with components:

coherence Numeric vector of coherence values at each frequency: magnitude-squared coherence in $[0, 1]$ for `type = "magnitude"`, or the (signed) imaginary / lagged coherence for the other types.

type The coherence type computed.

frequencies Numeric vector of corresponding frequencies in Hz.

confidence_limit Numeric scalar giving the 95% threshold: $1 - \alpha^{(1 / (L - 1))}$ where $\alpha = 0.05$ and L is the number of segments (Halliday et al. 1995). Coherence values above this limit are statistically significant.

n_segments Integer number of segments used in Welch estimation.

References

- Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.
- Halliday, D. M., Rosenberg, J. R., Amjad, A. M., Breeze, P., Conway, B. A., & Farmer, S. F. (1995). A framework for the analysis of mixed time series/point process data – theory and application to the study of physiological tremor, single motor unit discharges and electromyograms. *Progress in Biophysics and Molecular Biology*, 64(2–3), 237–278.
- Nolte, G., et al. (2004). Identifying true brain interaction from EEG data using the imaginary part of coherency. *Clinical Neurophysiology*, 115(10), 2292–2307. doi:10.1016/j.clinph.2004.04.029
- Pascual-Marqui, R. D. (2007). Instantaneous and lagged measurements of linear and nonlinear dependence between groups of multivariate time series. *arXiv:0711.1455*.

See Also

[crossSpectrum\(\)](#), [multitaperCoherence\(\)](#), [laggedCoherence\(\)](#), [coherenceMatrix\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
# Two coupled 20 Hz sinusoids
sr <- 500
t <- seq(0, 10, length.out = sr * 10)
x <- sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
result <- coherence(x, y, sr = sr)
plot(result$frequencies, result$coherence, type = "l",
      xlab = "Frequency (Hz)", ylab = "Coherence")
abline(h = result$confidence_limit, lty = 2, col = "red")
```

coherenceMatrix	<i>Coherence matrix across channel pairs</i>
-----------------	--

Description

Computes magnitude-squared coherence for all pairs of channels between two sets of signals (or between all channels within a single set). Returns a matrix of peak coherence values plus the full spectra.

Usage

```
coherenceMatrix(
  x,
  y = NULL,
  sr = NULL,
  channels_x = NULL,
  channels_y = NULL,
  modality_x = NULL,
  modality_y = NULL,
  ...
)
```

Arguments

x	A <code>PhysioExperiment</code> or <code>MultiPhysioExperiment</code> object.
y	A <code>PhysioExperiment</code> , or <code>NULL</code> when x is an MPE.
sr	Numeric sampling rate in Hz (used only for numeric inputs).
channels_x	Integer vector of channel indices for x, or <code>NULL</code> for all channels.
channels_y	Integer vector of channel indices for y, or <code>NULL</code> for all channels.
modality_x, modality_y	Character modality names for MPE input.
...	Additional arguments passed to coherence (e.g. <code>nperseg</code> , <code>freq_range</code>).

Value

A list with components:

matrix Numeric matrix of peak coherence values, with dimensions $[n_channels_x \times n_channels_y]$.

spectra List of lists containing the full coherence result for each pair.

frequencies Numeric vector of frequencies from the first pair.

channel_names_x Character vector of x channel names.

channel_names_y Character vector of y channel names.

References

Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.

See Also

[coherence\(\)](#), [couplingMatrix\(\)](#), [plotCouplingMatrix\(\)](#), [surrogateMatrixTest\(\)](#)

Examples

```
sr <- 200; n <- sr * 2
pe1 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(n * 2), nrow = n)),
  samplingRate = sr
)
pe2 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(n * 2), nrow = n)),
  samplingRate = sr
)
result <- coherenceMatrix(pe1, pe2, nperseg = 64L)
result$matrix
```

comodulogram

Comodulogram (phase-frequency by amplitude-frequency PAC)

Description

Computes phase-amplitude coupling over a grid of phase frequencies and amplitude frequencies, yielding a comodulogram matrix whose peak locates the dominant coupling.

Usage

```
comodulogram(
  x,
  y = NULL,
  sr = NULL,
  phase_freqs,
```

```

    amp_freqs,
    method = c("tort", "canolty", "ozkurt", "plv"),
    phase_bw = 2,
    amp_bw = 10,
    n_bins = 18L,
    order = 4L,
    modality_x = NULL,
    modality_y = NULL,
    channels_x = 1L,
    channels_y = 1L
  )

```

Arguments

`x`, `y`, `sr`, `method`, `n_bins`, `order`, `modality_x`, `modality_y`, `channels_x`, `channels_y`

As in [phaseAmplitudeCoupling\(\)](#).

`phase_freqs` Numeric vector of phase centre frequencies in Hz.

`amp_freqs` Numeric vector of amplitude centre frequencies in Hz.

`phase_bw` Half-bandwidth of each phase band in Hz (default: 2).

`amp_bw` Half-bandwidth of each amplitude band in Hz (default: 10).

Value

A list with the matrix (rows = phase frequencies, columns = amplitude frequencies), `phase_freqs`, `amp_freqs`, `method`, and `peak` (the (phase, amplitude) frequency of the maximum and its value).

See Also

[phaseAmplitudeCoupling\(\)](#), [plotComodulogram\(\)](#)

Examples

```

## Not run:
set.seed(1); t <- (0:4999) / 500; ph <- 2 * pi * 6 * t
sig <- sin(ph) + (0.1 + (1 + cos(ph - pi)) / 2) * sin(2 * pi * 50 * t)
cm <- comodulogram(sig, sr = 500, phase_freqs = c(4, 6, 8),
                   amp_freqs = c(30, 50, 70))
cm$peak

## End(Not run)

```

coupledNMF

*Coupled non-negative matrix factorization across two modalities***Description**

Factorises two non-negative blocks that share the observation dimension into a SHARED factor matrix W and modality-specific loadings H_x, H_y , minimising $||X - W H_x||^2 + ||Y - W H_y||^2$. The shared W is the common latent drive recovered jointly from both modalities.

Usage

```
coupledNMF(
  X,
  Y,
  rank,
  max_iter = 500L,
  tol = 1e-07,
  n_restart = 5L,
  seed = NULL
)
```

Arguments

X, Y	Non-negative matrices with the same n rows (shared observations, e.g. time or gait-cycle points); columns are each modality's channels.
rank	Number of shared factors.
max_iter, tol	Iteration budget and convergence tolerance.
n_restart	Random restarts (best kept; default 5).
seed	Optional RNG seed.

Value

a coupled_nmf object: shared ($n \times \text{rank}$ shared factor W), H_x, H_y ($\text{rank} \times p$ and $\text{rank} \times q$ loadings), vaf_x, vaf_y, vaf (combined), iterations.

References

Lee & Seung (2001) NMF; Cichocki et al. (2009) coupled/joint NMF.

See Also

[plsBlocks\(\)](#), [jive\(\)](#)

Examples

```
set.seed(1); W <- matrix(runif(50 * 2), 50, 2)      # shared drive
X <- W %%% matrix(runif(2 * 6), 2, 6); Y <- W %%% matrix(runif(2 * 4), 2, 4)
fit <- coupledNMF(X, Y, rank = 2, n_restart = 2)
fit$vaf
```

couplingAnalysis

Unified interface for cross-modal coupling analysis

Description

Dispatches to the appropriate coupling function based on the method parameter. Accepts the same flexible inputs as the underlying functions: two numeric vectors (with sr), two `PhysioExperiment` objects, or a `MultiPhysioExperiment` with named modalities.

Usage

```
couplingAnalysis(
  x,
  y = NULL,
  mpe = NULL,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  method = c("coherence", "plv", "pli", "wpli", "granger", "crosscorrelation",
    "wavelet_coherence", "wavelet_plv", "multitaper_coherence", "dtf", "pdc",
    "transferentropy", "pac"),
  sr = NULL,
  ...,
  granger_method = c("parametric", "nonparametric")
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> . When mpe is provided this argument is ignored.
y	Numeric vector or <code>PhysioExperiment</code> (NULL when x is an MPE or when mpe is provided).
mpe	A MultiPhysioExperiment object. When supplied, x and y are ignored and signals are extracted from mpe using modality_x / modality_y.
modality_x, modality_y	Character names of the modalities to extract from mpe.
channels_x, channels_y	Integer channel indices to extract (default 1).

method	Character string specifying the coupling method. One of: "coherence", "plv", "pli", "wpli", "granger", "crosscorrelation", "wavelet_coherence", or "wavelet_plv".
sr	Numeric sampling rate in Hz. Required when x and y are numeric vectors.
...	Additional arguments passed to the specific coupling function (e.g. freq_band, order, max_lag, nperseg, etc.).
granger_method	Named-only estimator selector used when method = "granger": "parametric" (default) or "nonparametric".

Value

The result from the dispatched coupling function. See individual function documentation for details:

- "coherence": see [coherence](#)
- "plv": see [phaseLockingValue](#)
- "pli": see [phaseLagIndex](#)
- "wpli": see [weightedPLI](#)
- "granger": see [grangerCausality](#)
- "crosscorrelation": see [crossCorrelation](#)
- "wavelet_coherence": see [waveletCoherence](#)
- "wavelet_plv": see [waveletPLV](#)

References

- Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.
- Lachaux, J.-P., Rodriguez, E., Martinerie, J., & Varela, F. J. (1999). Measuring phase synchrony in brain signals. *Human Brain Mapping*, 8(4), 194–208.
- Granger, C. W. J. (1969). Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3), 424–438.

See Also

[coherence\(\)](#), [phaseLockingValue\(\)](#), [grangerCausality\(\)](#), [crossCorrelation\(\)](#), [surrogateTest\(\)](#)

Examples

```
# Numeric vectors
sr <- 500
t <- seq(0, 10, length.out = sr * 10)
x <- sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))

# Coherence
res <- couplingAnalysis(x, y, method = "coherence", sr = sr)

# Cross-correlation
res <- couplingAnalysis(x, y, method = "crosscorrelation", sr = sr)
```

couplingAnalysisCached

Cached coupling analysis over a MultiPhysioExperiment

Description

Wraps [couplingAnalysis\(\)](#) with a per-object memoisation cache held in the couplingResults slot of a [MultiPhysioExperiment](#). Results are keyed by a [digest](#) of the analysis arguments (method, modalities, channels, and any extra arguments), so a repeated identical request returns the stored result without recomputing.

Usage

```
couplingAnalysisCached(
  mpe,
  method,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  ...,
  cache = TRUE
)
```

Arguments

mpe	A MultiPhysioExperiment object.
method	Character coupling method (see couplingAnalysis()).
modality_x, modality_y	Character modality names to extract from mpe.
channels_x, channels_y	Integer channel indices (default 1).
...	Additional arguments forwarded to couplingAnalysis() and folded into the cache key.
cache	Logical; if TRUE (default) reuse and populate the cache. FALSE forces recomputation but still stores the fresh result.

Value

A list with result (the coupling result), mpe (the object with its couplingResults cache updated) and cached (TRUE when the result was served from the cache).

See Also

[couplingAnalysis\(\)](#), [couplingResults\(\)](#)

Examples

```
eeg <- PhysioExperiment(assays = list(raw = matrix(rnorm(1000), 500, 2)),
                          samplingRate = 250)
emg <- PhysioExperiment(assays = list(raw = matrix(rnorm(1000), 500, 2)),
                          samplingRate = 250)
mpe <- MultiPhysioExperiment(EEG = eeg, EMG = emg)
out <- couplingAnalysisCached(mpe, "coherence", "EEG", "EMG")
out2 <- couplingAnalysisCached(out$mpe, "coherence", "EEG", "EMG")
out2$cached # TRUE
```

couplingMatrix	<i>Generic coupling matrix across channel pairs</i>
----------------	---

Description

Computes any coupling method for all pairs of channels between two sets of signals, extracting a scalar statistic for each pair. This is the multi-channel generalisation of [couplingAnalysis](#).

Usage

```
couplingMatrix(
  x,
  y = NULL,
  sr = NULL,
  method,
  channels_x = NULL,
  channels_y = NULL,
  modality_x = NULL,
  modality_y = NULL,
  ...
)
```

Arguments

x	A PhysioExperiment or MultiPhysioExperiment object.
y	A PhysioExperiment, or NULL when x is an MPE.
sr	Numeric sampling rate in Hz.
method	Character coupling method, one of "coherence", "plv", "pli", "wpli", "granger", "crosscorrelation".
channels_x, channels_y	Integer vectors of channel indices, or NULL for all channels.
modality_x, modality_y	Character modality names for MPE input.
...	Additional arguments passed to the coupling function.

Value

A list with components:

matrix Numeric matrix of coupling values.

method Character method used.

channel_names_x Character vector of x channel names.

channel_names_y Character vector of y channel names.

References

Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.

See Also

[couplingAnalysis\(\)](#), [coherenceMatrix\(\)](#), [plotCouplingMatrix\(\)](#), [surrogateMatrixTest\(\)](#)

Examples

```
sr <- 200; n <- sr * 2
pe1 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(n * 2), nrow = n)),
  samplingRate = sr
)
pe2 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(n * 2), nrow = n)),
  samplingRate = sr
)
result <- couplingMatrix(pe1, pe2, method = "coherence", nperseg = 64L)
result$matrix
```

couplingResults

Cached coupling results for a container

Description

Returns the coupling results computed for x in this session by [couplingAnalysisCached](#).

Usage

```
couplingResults(x)
```

```
## S4 method for signature 'MultiPhysioExperiment'
couplingResults(x)
```

```
couplingResults(x) <- value
```

```
## S4 replacement method for signature 'MultiPhysioExperiment'
couplingResults(x) <- value
```

Arguments

`x` A MultiPhysioExperiment.
`value` A named list of results to cache for `x`, or `NULL` / `list()` to clear them.

Details

The cache is keyed on the signal values, the clock, the assay, the channels, the method and its parameters, so a result is only returned for inputs it was actually computed from. It is held for the session rather than stored inside the container: a cache carried inside a saved object outlives any evidence that it still matches its inputs.

Value

A named list of cached results (empty when nothing is cached).

See Also

[couplingAnalysisCached\(\)](#)

crossBlockFactor	<i>Cross-block common-factor model</i>
------------------	--

Description

Fits a maximum-likelihood common-factor model to the pooled, standardised modality blocks and flags which latent factors are SHARED across modalities (load on more than one block) versus modality-specific – a latent-variable view of multi-modal integration.

Usage

```
crossBlockFactor(blocks, nfactors, load_threshold = 0.4)
```

Arguments

`blocks` A list of ≥ 2 matrices, each $n \times p_k$ (shared rows). Requires more observations than total features.
`nfactors` Number of common factors.
`load_threshold` A factor is considered to load on a block if any of the block's variables has $|loading| \geq$ this (default 0.4).

Value

a `crossblock_factor` object: loadings (variables $\times$ factors, with a block attribute), shared (logical per factor: loads on ≥ 2 blocks), `n_shared`, and the fitted `factanal`.

References

Spearman/Thurstone common-factor model; McDonald (1985).

See Also

`jive()`, `multipleFactorAnalysis()`

Examples

```
set.seed(1); f <- matrix(rnorm(80 * 1), 80, 1)
a <- f %%% matrix(rnorm(4), 1, 4) + matrix(rnorm(80 * 4, 0, .4), 80, 4)
b <- f %%% matrix(rnorm(3), 1, 3) + matrix(rnorm(80 * 3, 0, .4), 80, 3)
crossBlockFactor(list(a = a, b = b), nfactors = 1)$shared
```

crossCorrelation	<i>Cross-correlation between two signals</i>
------------------	--

Description

Computes the cross-correlation between two signals at various lags, measuring their time-domain coupling. Cross-correlation quantifies how similar one signal is to a time-shifted version of another. A positive peak lag indicates that y leads x (i.e., y must be shifted forward to align with x), while a negative peak lag indicates x leads y.

Usage

```
crossCorrelation(
  x,
  y = NULL,
  sr = NULL,
  max_lag = NULL,
  normalize = TRUE,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when x is a <code>MultiPhysioExperiment</code> .
sr	Numeric sampling rate in Hz (required when x/y are numeric).
max_lag	Integer maximum lag in samples. If <code>NULL</code> , defaults to <code>floor(length(x) / 4)</code> .

<code>normalize</code>	Logical; if TRUE (default), compute normalized cross-correlation (Pearson-like, values in [-1, 1]).
<code>modality_x</code>	Character modality name in MPE for the x signal.
<code>modality_y</code>	Character modality name in MPE for the y signal.
<code>channels_x</code>	Integer which channel to extract from x (default 1).
<code>channels_y</code>	Integer which channel to extract from y (default 1).

Details

The function accepts numeric vectors, `PhysioExperiment` objects, or a `MultiPhysioExperiment` with named modalities, using `.extract_signal_pair()` internally for flexible input handling.

Value

A named list with components:

correlation Numeric vector of cross-correlation values at each lag.

lags Integer vector of lag values in samples.

lag_seconds Numeric vector of lag values in seconds.

peak_lag Integer lag (in samples) at which the absolute correlation is maximised.

peak_lag_seconds Numeric peak lag converted to seconds.

peak_correlation Numeric cross-correlation value at the peak lag.

References

Chatfield, C. (2004). *The Analysis of Time Series: An Introduction* (6th ed.). Chapman & Hall/CRC.

See Also

[slidingCrossCorrelation\(\)](#), [coherence\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
# Simple example: y is a delayed copy of x
sr <- 500
set.seed(1)
x <- rnorm(1000)
y <- c(rep(0, 10), x[1:990])
result <- crossCorrelation(x, y, sr = sr)
result$peak_lag      # should be 10
result$peak_lag_seconds
```

crossSpectrum	<i>Cross-spectral density between two signals</i>
---------------	---

Description

Computes the cross-spectral density (CSD) between two signals using Welch's method. The CSD captures both the magnitude and phase relationship between two signals as a function of frequency.

Usage

```
crossSpectrum(
  x,
  y = NULL,
  sr = NULL,
  nperseg = 256L,
  noverlap = NULL,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when <code>x</code> is an MPE.
sr	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).
nperseg	Integer segment length for Welch's method (default 256).
noverlap	Integer overlap between segments, or <code>NULL</code> for <code>floor(nperseg / 2)</code> .
modality_x	Character modality name in MPE for the <code>x</code> signal.
modality_y	Character modality name in MPE for the <code>y</code> signal.
channels_x	Integer which channel to extract from <code>x</code> (default 1).
channels_y	Integer which channel to extract from <code>y</code> (default 1).

Value

A named list with components:

csd Complex vector of cross-spectral density values.

frequencies Numeric vector of corresponding frequencies in Hz.

magnitude Numeric vector of CSD magnitude (`Mod(csd)`).

phase Numeric vector of CSD phase in radians (`Arg(csd)`).

References

- Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.
- Halliday, D. M., Rosenberg, J. R., Amjad, A. M., Breeze, P., Conway, B. A., & Farmer, S. F. (1995). A framework for the analysis of mixed time series/point process data – theory and application to the study of physiological tremor, single motor unit discharges and electromyograms. *Progress in Biophysics and Molecular Biology*, 64(2–3), 237–278.

See Also

[coherence\(\)](#), [multitaperCoherence\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
t <- seq(0, 5, length.out = sr * 5)
x <- sin(2 * pi * 10 * t) + 0.1 * rnorm(length(t))
y <- cos(2 * pi * 10 * t) + 0.1 * rnorm(length(t))
result <- crossSpectrum(x, y, sr = sr)
plot(result$frequencies, result$magnitude, type = "l",
      xlab = "Frequency (Hz)", ylab = "|CSD|")
```

crossWaveletTransform *Cross-wavelet transform (XWT) with red-noise significance*

Description

Computes the cross-wavelet transform $W_{xy} = W_x W_y^*$ of two signals via the Morlet wavelet: its modulus is the common time-frequency power and its argument is the phase lead/lag between the signals. Common power is tested against an AR(1) red-noise background (Torrence & Compo 1998; Grinsted et al. 2004) by Monte-Carlo: AR(1) surrogates matched to each signal's lag-1 autocorrelation and variance are transformed identically, and the per-frequency siglvl quantile of the surrogate cross-power sets the threshold.

Usage

```
crossWaveletTransform(
  x,
  y = NULL,
  sr = NULL,
  frequencies = seq(1, 40, by = 1),
  n_cycles = 7,
  significance = TRUE,
  n_surrogates = 100L,
  siglvl = 0.95,
  modality_x = NULL,
  modality_y = NULL,
```

```

    channels_x = 1L,
    channels_y = 1L
  )

```

Arguments

<code>x, y</code>	Signals (or a single object from which a pair is extracted).
<code>sr</code>	Sampling rate in Hz.
<code>frequencies</code>	Frequencies (Hz) to analyse (default <code>seq(1, 40, 1)</code>).
<code>n_cycles</code>	Morlet wavelet cycles (default 7).
<code>significance</code>	If TRUE (default), compute red-noise significance.
<code>n_surrogates</code>	Number of AR(1) surrogate pairs (default 100).
<code>siglvl</code>	Significance quantile (default 0.95).
<code>modality_x, modality_y, channels_x, channels_y</code>	Passed to the pair extractor.

Value

An object of class "cross_wavelet": a list with power (time x frequency cross-wavelet modulus), phase (relative phase, radians; positive = x leads y), cross (complex XWT), frequencies, times, coi (cone of influence), and — when significance — `sig_level` (per frequency), `sig_ratio` (power / level), and `significant` (logical mask).

References

Torrence C, Compo GP (1998). A practical guide to wavelet analysis. *Bull Amer Meteor Soc* 79:61-78. Grinsted A, Moore JC, Jevrejeva S (2004). Application of the cross wavelet transform and wavelet coherence to geophysical time series. *Nonlin Processes Geophys* 11:561-566.

Examples

```

set.seed(1); sr <- 100; t <- seq(0, 20, by = 1 / sr)
x <- sin(2 * pi * 10 * t) + rnorm(length(t), sd = 0.5)
y <- sin(2 * pi * 10 * t - pi / 2) + rnorm(length(t), sd = 0.5) # y lags x
xw <- crossWaveletTransform(x, y, sr = sr, frequencies = seq(4, 16, by = 1),
                           n_surrogates = 50)
mean(xw$significant)

```

directedTransferFunction

Directed Transfer Function for multichannel / multimodal signals

Description

Frequency-resolved directed connectivity via the Directed Transfer Function (Kaminski & Bli-nowska 1991), computed from the transfer function of a multivariate autoregressive model fitted with `PhysioExperiment::mvarFit()`. This cross-modal implementation is numerically identical to PhysioEEG's `eegDTF` on the same input. Because the transfer function inverts the whole VAR system, DTF reflects both direct and indirect pathways.

Usage

```
directedTransferFunction(
  x,
  y = NULL,
  sr = NULL,
  order = 5L,
  freqs = NULL,
  normalized = TRUE,
  ffDTF = FALSE,
  band = NULL,
  mvar_method = "ols",
  modality_x = NULL,
  modality_y = NULL,
  channels_x = NULL,
  channels_y = NULL
)
```

Arguments

<code>x</code>	A multichannel matrix (time x channels) or <code>PhysioExperiment</code> (with <code>y = NULL</code>), or one signal / <code>PhysioExperiment</code> of a pair.
<code>y</code>	The second signal / <code>PhysioExperiment</code> of a pair, or <code>NULL</code> for a single multichannel input.
<code>sr</code>	Sampling rate in Hz (required for numeric-matrix input).
<code>order</code>	MVAR model order (default: 5), or <code>NULL</code> to select automatically.
<code>freqs</code>	Frequencies in Hz (default: 128 points from 0 to the Nyquist frequency).
<code>normalized</code>	Row-normalize the DTF (default: <code>TRUE</code>).
<code>ffDTF</code>	Use the full-frequency DTF normalization (default: <code>FALSE</code>).
<code>band</code>	Optional length-2 band in Hz over which to average the returned matrix (default: all frequencies).
<code>mvar_method</code>	MVAR estimator (default: "ols").
<code>modality_x, modality_y, channels_x, channels_y</code>	Passed to the shared signal-pair extraction when <code>y</code> is supplied.

Value

A list with `dtf` (`n x n x n_freqs` array), the band-averaged directed matrix (rows = targets, columns = sources), frequencies, channels, and order. For a two-channel input, `dtf_xy` and `dtf_yx` give the directional summaries.

References

Kaminski, M. J., & Blinowska, K. J. (1991). A new method of the description of the information flow in the brain structures. *Biological Cybernetics*, 65(3), 203-210.

See Also

[partialDirectedCoherence\(\)](#), [transferEntropy\(\)](#), [grangerCausality\(\)](#)

Examples

```
set.seed(1)
X <- matrix(rnorm(1500), ncol = 3)
res <- directedTransferFunction(X, sr = 100, order = 4)
dim(res$dtf)
```

distanceCorrelation	<i>Distance correlation between two modality blocks</i>
---------------------	---

Description

A dependence measure that captures NON-linear as well as linear cross-modal relationships and is zero if and only if the modalities are independent (Szekely et al. 2007). In [0, 1].

Usage

```
distanceCorrelation(X, Y, permutations = 0L, seed = NULL)
```

Arguments

X, Y	Matrices with the same n rows.
permutations	Row permutations for the null p-value (default 0).
seed	Optional RNG seed.

Value

a distance_correlation object: dcor, dcov, p_value, n.

References

Szekely GJ, Rizzo ML, Bakirov NK (2007) *Ann Stat* 35:2769-2794.

See Also

[rvCoefficient\(\)](#), [representationalSimilarity\(\)](#)

Examples

```
set.seed(1); x <- rnorm(60); y <- x^2 + rnorm(60, 0, 0.3) # nonlinear
distanceCorrelation(x, y)$dcor
```

elasticAlign

*Elastic (SRVF) alignment of two curves***Description**

Aligns curve y to curve x by an optimal time warping computed in the Square-Root Velocity Function representation, separating phase (the warping) from amplitude (the residual shape difference). A pure-R dynamic-programming solver is used by default; if the **fdasrvf** package is available it is used for higher accuracy.

Usage

```
elasticAlign(
  x,
  y,
  t = NULL,
  lambda = 0,
  method = c("srvf"),
  max_step = 6L,
  smooth = FALSE,
  use_fdasrvf = NA
)
```

Arguments

<code>x, y</code>	Numeric curves of the same length.
<code>t</code>	Optional grid (defaults to an evenly spaced grid on 0..1).
<code>lambda</code>	Non-negative warping-roughness penalty (passed to fdasrvf when used; the pure-R solver ignores it beyond boundary/monotonicity).
<code>method</code>	Currently "srvf".
<code>max_step</code>	Maximum DP step (bounds the local warping slope) for the pure-R solver (default: 6).
<code>smooth</code>	Smooth the pure-R warping with a monotone spline (default: FALSE).
<code>use_fdasrvf</code>	NA (default) uses fdasrvf if installed; TRUE requires it; FALSE forces the pure-R solver.

Value

A list with `gamma` (the warping), `aligned` (y warped to x), `amplitude_distance`, `phase_distance`, the SRVFs, and the solver used.

References

Srivastava, A., Wu, W., Kurtek, S., Klassen, E., & Marron, J. S. (2011). Registration of functional data using Fisher-Rao metric. arXiv:1103.3817.

See Also

[srvfMean\(\)](#), [warpApply\(\)](#)

Examples

```
t <- seq(0, 1, length.out = 80)
x <- exp(-((t - 0.5)^2) / (2 * 0.08^2))
y <- exp(-((t - 0.65)^2) / (2 * 0.08^2))
al <- elasticAlign(x, y)
al$amplitude_distance
```

fuseBlocks

Feature-level fusion of modality blocks

Description

Concatenates several modality blocks into a single design matrix for downstream (multi-view) machine learning, scaling each block so a large or high-variance modality does not dominate the fused features.

Usage

```
fuseBlocks(blocks, method = c("zscore", "mfa", "frobenius"))
```

Arguments

blocks	A list of matrices, each $n \times p_k$ (shared rows).
method	Block scaling before concatenation: "zscore" (per-column standardise, default), "mfa" (divide each block by its leading singular value), or "frobenius" (unit Frobenius norm per block).

Value

a fused_blocks object: matrix ($n \times \sum(p_k)$) and block_index (column ranges per block).

See Also

[multipleFactorAnalysis\(\)](#), [multimodalGaitIndex\(\)](#)

Examples

```
fuseBlocks(list(a = matrix(rnorm(20), 10, 2), b = matrix(rnorm(30) * 50, 10, 3)))$matrix
```

Description

Computes directed coupling between two physiological signals using either autoregressive prediction or nonparametric spectral factorization. Both scalar and frequency-resolved Granger causality are supported.

Usage

```
grangerCausality(
  x,
  y = NULL,
  sr = NULL,
  order = 5L,
  freq_range = NULL,
  method = c("parametric", "nonparametric"),
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  spectral_estimator = c("multitaper", "welch"),
  n_fft = NULL,
  segment_length = NULL,
  overlap = 0.5,
  time_bandwidth = 4,
  n_tapers = NULL,
  factor_tolerance = 1e-08,
  factor_max_iterations = 1000L,
  eigen_floor = 1e-10
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when x is a <code>MultiPhysioExperiment</code> .
sr	Numeric sampling rate in Hz (required when x/y are numeric vectors).
order	Integer AR model order (default 5). Higher order captures longer-range temporal dependencies but requires more data.
freq_range	Numeric vector of length 2 specifying the frequency band in Hz over which to compute spectral Granger causality. With the parametric method, <code>NULL</code> returns time-domain GC. With the nonparametric method, <code>NULL</code> aggregates the full DC-to-Nyquist spectrum.

method	Character estimator. "parametric" (default) uses the established MVAR ordinary-least-squares path. "nonparametric" estimates a complete cross-spectral matrix and applies Wilson factorization followed by Geweke's frequency-domain decomposition.
modality_x	Character modality name in MultiPhysioExperiment for the x signal.
modality_y	Character modality name in MultiPhysioExperiment for the y signal.
channels_x	Integer channel index to extract from x (default 1).
channels_y	Integer channel index to extract from y (default 1).
spectral_estimator	Nonparametric cross-spectral estimator: "multitaper" (default) or "welch".
n_fft	Even FFT length for the nonparametric estimator. NULL chooses a deterministic length from segment_length.
segment_length	Segment length for cross-spectral estimation. NULL chooses the largest usable power of two up to 512 samples.
overlap	Fractional segment overlap in $[0, 1)$.
time_bandwidth	DPSS time-bandwidth product for multitaper estimation.
n_tapers	Number of DPSS tapers; NULL uses $\text{floor}(2 * \text{time_bandwidth}) - 1$.
factor_tolerance	Relative Wilson-factor update tolerance.
factor_max_iterations	Maximum Wilson iterations.
eigen_floor	Relative positive eigenvalue floor used only for round-off-scale or ill-conditioned cross-spectral bins.

Details

For the time-domain (parametric) method, Granger causality from x to y is defined as:

$$GC_{x \rightarrow y} = \log(\sigma_{restricted}^2 / \sigma_{unrestricted}^2)$$

where the restricted model predicts y from its own past only, and the unrestricted model predicts y from both its own past and x's past.

For spectral Granger causality (when freq_range is specified), the AR coefficients are transformed to the frequency domain via the transfer function $H(f) = A(f)^{-1}$, and the spectral GC is computed as:

$$GC_{x \rightarrow y}(f) = \log(S_{yy}^{restricted}(f) / S_{yy}^{unrestricted}(f))$$

The returned values are then averaged over the specified frequency band.

The parametric frequency-band path uses separate restricted and unrestricted univariate AR models and is an approximation to the full Geweke spectral decomposition. Its established output is retained for compatibility. The nonparametric path instead estimates a complete Hermitian 2x2 cross-spectral matrix with common windows/tapers, applies Wilson's minimum-phase factorization, and evaluates Geweke directional spectra. A scalar is the trapezoidal frequency mean over the selected band, with DC and Nyquist receiving endpoint half weights.

Cross-spectral eigenvalue adjustment is limited to the reported relative `eigen_floor`; materially indefinite matrices fail. Wilson update convergence and spectral reconstruction are checked separately. The relative reconstruction limit is $\min(0.05, \max(0.02, 100 * \text{factor_tolerance}))$, so it remains between 2% and 5%; the observed error and limit are returned in the result. Nonparametric estimation requires at least 32 paired samples and at least two spectral realizations.

Granger causality is a stationary linear predictive-direction measure. It is not proof of physical, mechanistic, or interventional causality and does not resolve unmeasured confounding, common drivers, volume conduction, instantaneous mixing, or non-stationarity.

Value

A list with components:

gc_xy Numeric Granger causality from x to y (x drives y). A positive value indicates that x's past helps predict y beyond y's own past.

gc_yx Numeric Granger causality from y to x (y drives x).

net_gc Numeric net Granger causality (`gc_xy - gc_yx`). Positive values indicate that x drives y more than y drives x.

order Integer AR model order used, or NA for nonparametric estimation.

method Character estimator used.

Nonparametric results additionally contain the frequency grid, directional spectra, estimator/factorization metadata, innovation covariance, the factorization reconstruction limit, and regularization diagnostics.

References

Granger, C. W. J. (1969). Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3), 424-438.

Geweke, J. (1982). Measurement of linear dependence and feedback between multiple time series. *Journal of the American Statistical Association*, 77(378), 304-313.

Wilson, G. T. (1972). The factorization of matricial spectral densities. *SIAM Journal on Applied Mathematics*, 23(4), 420-426.

Dhamala, M., Rangarajan, G., & Ding, M. (2008). Analyzing information flow in brain networks with nonparametric Granger causality. *NeuroImage*, 41(2), 354-362.

Dhamala, M., Rangarajan, G., & Ding, M. (2008). Estimating Granger causality from Fourier and wavelet transforms of time series data. *Physical Review Letters*, 100, 018701.

See Also

[coherence\(\)](#), [phaseLockingValue\(\)](#)

Examples

```
# Create directed signals: x drives y with a 5-sample lag
set.seed(42)
n <- 5000
x <- rnorm(n)
```

```

y <- numeric(n)
for (i in 6:n) y[i] <- 0.6 * x[i - 5] + 0.4 * rnorm(1)

result <- grangerCausality(x, y, sr = 500, order = 10)
result$gc_xy   # Should be positive (x drives y)
result$net_gc  # Should be positive

```

jive

JIVE: Joint and Individual Variation Explained

Description

Decomposes several modality blocks that share an observation dimension into a low-rank JOINT structure common to all blocks, low-rank INDIVIDUAL structure specific to each, and residual noise (Lock et al. 2013). The joint scores are the integrated multi-modal representation; the per-block variance split tells you how much of each modality is shared versus idiosyncratic.

Usage

```

jive(
  blocks,
  rank_joint = 1L,
  rank_individual = 1L,
  center = TRUE,
  scale = TRUE,
  max_iter = 1000L,
  tol = 1e-08
)

```

Arguments

blocks	A list of ≥ 2 matrices, each $n \times p_k$: the same n observations (e.g. subjects, or gait-cycle points), different features per modality.
rank_joint	Rank of the joint structure.
rank_individual	Rank of each block's individual structure (a scalar, recycled, or a per-block vector).
center, scale	Row-center each feature and scale each block to unit Frobenius norm (so blocks contribute comparably); both default TRUE.
max_iter, tol	Iteration budget and convergence tolerance.

Value

a jive object: joint and individual (lists of $n \times p_k$ matrices), joint_scores ($n \times \text{rank_joint}$, the shared component scores), variance (data frame of joint / individual / residual fraction per block), ranks, iterations.

References

Lock EF, Hoadley KA, Marron JS, Nobel AB (2013) Ann Appl Stat 7:523-542.

See Also

`multipleFactorAnalysis()`, `cca()`

Examples

```
set.seed(1)
shared <- matrix(rnorm(40 * 2), 40, 2) # common subject scores
k <- shared %*% matrix(rnorm(2 * 6), 2, 6) + matrix(rnorm(40 * 6, 0, .3), 40, 6)
e <- shared %*% matrix(rnorm(2 * 5), 2, 5) + matrix(rnorm(40 * 5, 0, .3), 40, 5)
fit <- jive(list(kin = k, emg = e), rank_joint = 2, rank_individual = 1)
fit$variance
```

laggedCoherence	<i>Lagged coherence between two signals</i>
-----------------	---

Description

Convenience wrapper for `coherence` with `type = "lagged"`, the lagged (non-instantaneous) coherence of Pascual-Marqui (2007), $\text{Im}(C) / \sqrt{1 - \text{Re}(C)^2}$ where C is the complex coherence. Like the imaginary part of coherence it is insensitive to zero-lag / volume-conduction coupling.

Usage

```
laggedCoherence(
  x,
  y = NULL,
  sr = NULL,
  freq_range = NULL,
  nperseg = 256L,
  noverlap = NULL,
  method = c("welch", "multitaper"),
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

<code>x</code>	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
<code>y</code>	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when <code>x</code> is an MPE.
<code>sr</code>	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).

freq_range	Numeric vector of length 2 giving the frequency range c(low, high) in Hz to restrict the output. NULL returns all frequencies.
nperseg	Integer segment length for Welch's method (default 256).
noverlap	Integer overlap between segments, or NULL for floor(nperseg / 2).
method	Character spectral estimation method. Currently only "welch" is implemented; "multitaper" is reserved for future use.
modality_x	Character modality name in MPE for the x signal.
modality_y	Character modality name in MPE for the y signal.
channels_x	Integer which channel to extract from x (default 1).
channels_y	Integer which channel to extract from y (default 1).

Value

A named list as returned by [coherence](#), with type = "lagged".

References

Pascual-Marqui, R. D. (2007). Instantaneous and lagged measurements of linear and nonlinear dependence between groups of multivariate time series. *arXiv:0711.1455*.

See Also

[coherence](#)

Examples

```
sr <- 200
t <- seq(0, 10, length.out = sr * 10)
x <- sin(2 * pi * 10 * t)
y <- sin(2 * pi * 10 * t + pi / 2)      # 90-degree lag
lc <- laggedCoherence(x, y, sr = sr, freq_range = c(8, 12))
max(abs(lc$coherence))
```

leidaStates

LEiDA dynamic functional-connectivity states

Description

Leading Eigenvector Dynamics Analysis (Cabral et al. 2017): at each time point the instantaneous phase-coherence matrix is reduced to its leading eigenvector; these are clustered over time into a small set of recurrent connectivity states. Complements state-resolved network analysis by finding the states data-drivenly rather than from external labels.

Usage

```
leidaStates(
  x,
  sr,
  freq_band,
  n_states = 4L,
  order = 4L,
  nstart = 10L,
  seed = NULL
)
```

Arguments

<code>x</code>	A numeric matrix, time (rows) by channels (columns).
<code>sr</code>	Sampling rate in Hz.
<code>freq_band</code>	Numeric <code>c(low, high)</code> band in Hz.
<code>n_states</code>	Number of clusters/states (default 4).
<code>order</code>	Butterworth filter order (default 4).
<code>nstart</code>	k-means restarts (default 10).
<code>seed</code>	Optional integer seed for reproducible clustering.

Value

An object of class "leida": a list with states (per-time cluster labels), centroids (states x channels leading-eigenvector centroids), occupancy (fraction of time in each state), dwell (mean run length in samples per state), and `n_states`.

References

Cabral J, et al. (2017). Cognitive performance in healthy older adults relates to spontaneous switching between states of functional connectivity during rest. *Sci Rep* 7:5135.

Examples

```
set.seed(1); sr <- 100; n <- 2000; t <- seq_len(n) / sr
# two regimes: channels 1-2 coherent first half, 3-4 coherent second half
base <- sin(2 * pi * 10 * t)
X <- cbind(base + rnorm(n, sd = .3), base + rnorm(n, sd = .3),
           sin(2 * pi * 10 * t + 1) + rnorm(n, sd = .3),
           sin(2 * pi * 10 * t + 1) + rnorm(n, sd = .3))
res <- leidaStates(X, sr = sr, freq_band = c(8, 12), n_states = 2, seed = 1)
res$occupancy
```

leidaTransitions	<i>LEiDA state-transition metrics</i>
------------------	---------------------------------------

Description

From a `leidaStates()` result, computes the state-to-state transition probability matrix, the switching rate, fractional occupancy, mean dwell time, and the occupancy-weighted transition entropy (predictability of the next state).

Usage

```
leidaTransitions(x, sr = NULL)
```

Arguments

<code>x</code>	A "leida" object from <code>leidaStates()</code> .
<code>sr</code>	Optional sampling rate (Hz); if given, dwell times are returned in seconds and the switching rate additionally in Hz.

Value

An object of class "leida_transitions": a list with `transition` (row-stochastic state x state matrix), `counts` (raw transition counts), `switching_rate` (fraction of steps that change state), `switching_rate_hz` (switches per second, or NA), `occupancy`, `dwell`, and `entropy` (mean bits of uncertainty about the next state).

Examples

```
set.seed(1); sr <- 100; n <- 2000; t <- seq_len(n) / sr
base <- sin(2 * pi * 10 * t)
X <- cbind(base + rnorm(n, sd = .3), base + rnorm(n, sd = .3),
           sin(2 * pi * 10 * t + 1) + rnorm(n, sd = .3),
           sin(2 * pi * 10 * t + 1) + rnorm(n, sd = .3))
res <- leidaStates(X, sr = sr, freq_band = c(8, 12), n_states = 2, seed = 1)
leidaTransitions(res, sr = sr)
```

lodoGeneralization	<i>Leave-one-site-out generalization benchmark</i>
--------------------	--

Description

Evaluates model transportability by training on all-but-one sites and testing on the held-out site (LODO). Supports continuous outcomes (`family = "gaussian"`) and binary outcomes (`family = "binomial"`).

Usage

```
lodoGeneralization(
  data,
  outcome,
  site,
  features = NULL,
  family = c("gaussian", "binomial"),
  positive_class = NULL,
  threshold = 0.5,
  min_train_rows = 20L,
  scale_features = TRUE
)
```

Arguments

<code>data</code>	Data frame containing outcome, site label, and features.
<code>outcome</code>	Outcome column name.
<code>site</code>	Site/facility column name used for LODO splitting.
<code>features</code>	Feature column names. If NULL, numeric columns excluding outcome and site are used.
<code>family</code>	Modeling family: "gaussian" or "binomial".
<code>positive_class</code>	Positive class label for binomial metrics. If NULL, the second factor level is used.
<code>threshold</code>	Classification threshold for binomial predictions.
<code>min_train_rows</code>	Minimum training rows required per fold.
<code>scale_features</code>	Logical; z-score features using training statistics.

Value

A list with:

fold_metrics Per-site metrics.
predictions Row-level held-out predictions.
aggregate Mean metrics across folds.
settings Benchmark settings and feature list.

See Also

[couplingAnalysis\(\)](#), [surrogateTest\(\)](#)

Examples

```
set.seed(1)
n <- 120
df <- data.frame(
  site_id = rep(c("A", "B", "C"), each = 40),
  f1 = rnorm(n),
```

```

    f2 = rnorm(n)
  )
  df$y <- 0.8 * df$f1 - 0.4 * df$f2 + rnorm(n, sd = 0.2)

  res <- lodoGeneralization(
    data = df,
    outcome = "y",
    site = "site_id",
    family = "gaussian"
  )
  head(res$fold_metrics)

```

make_coupled_signals *Generate coupled sinusoidal signals*

Description

Creates a pair of sinusoidal signals with a shared oscillatory component at a specified coupling frequency and controllable noise level. Useful for demonstrating and testing spectral coherence, phase synchrony, and other cross-modal coupling measures.

Usage

```

make_coupled_signals(
  sr1 = 500,
  sr2 = 500,
  coupling_freq = 20,
  coupling_strength = 0.8,
  noise = 0.2,
  duration = 10
)

```

Arguments

sr1	Numeric sampling rate in Hz for the first signal (default 500).
sr2	Numeric sampling rate in Hz for the second signal (default 500).
coupling_freq	Numeric frequency in Hz of the shared oscillatory component (default 20).
coupling_strength	Numeric amplitude of the shared component (default 0.8).
noise	Numeric standard deviation of the additive Gaussian noise (default 0.2).
duration	Numeric duration in seconds (default 10).

Details

Both signals share the same sinusoidal component at `coupling_freq`, scaled by `coupling_strength`, with additive Gaussian noise scaled by `noise`.

Value

A named list with components:

x Numeric vector – first coupled signal.

y Numeric vector – second coupled signal.

sr_x Numeric sampling rate of signal x.

sr_y Numeric sampling rate of signal y.

coupling_freq Numeric coupling frequency used.

References

Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.

See Also

[make_eeg_emg\(\)](#), [make_directed_signals\(\)](#), [coherence\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
signals <- make_coupled_signals(sr1 = 500, sr2 = 500,
                              coupling_freq = 20,
                              coupling_strength = 0.8,
                              noise = 0.2, duration = 10)
result <- coherence(signals$x, signals$y, sr = signals$sr_x)
```

make_directed_signals *Generate directed coupling signals*

Description

Creates a pair of signals where x drives y with a specified lag and coupling strength. Signal x is white noise, and y is a mixture of a lagged copy of x and independent noise. This is useful for testing directed coupling measures such as Granger causality.

Usage

```
make_directed_signals(n = 5000, sr = 500, lag_samples = 10, coupling = 0.7)
```

Arguments

n	Integer number of samples (default 5000).
sr	Numeric sampling rate in Hz (default 500).
lag_samples	Integer number of samples by which x leads y (default 10).
coupling	Numeric coupling strength in [0, 1] (default 0.7).

Details

The generating model is:

$$y(t) = \text{coupling} \cdot x(t - \text{lag_samples}) + (1 - \text{coupling}) \cdot \epsilon(t)$$

where $\epsilon(t) \sim N(0, 1)$.

Value

A named list with components:

x Numeric vector – driving signal (white noise).

y Numeric vector – driven signal (lagged mixture).

sr Numeric sampling rate.

lag_samples Integer lag used.

References

Granger, C. W. J. (1969). Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3), 424–438.

See Also

[grangerCausality\(\)](#), [make_coupled_signals\(\)](#), [crossCorrelation\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
signals <- make_directed_signals(n = 5000, sr = 500,
                                lag_samples = 10, coupling = 0.7)
result <- grangerCausality(signals$x, signals$y, sr = signals$sr,
                           order = 15)
result$gc_xy # should be positive (x drives y)
result$net_gc # should be positive
```

make_eeg_emg

Generate simulated EEG-EMG PhysioExperiment pair

Description

Creates a pair of `PhysioExperiment` objects simulating simultaneous EEG and EMG recordings with corticomuscular coherence (CMC) at a specified frequency. The first channel of each modality contains a shared oscillatory component; remaining channels contain independent noise.

Usage

```
make_eeg_emg(
  n_sec = 10,
  n_eeg_ch = 3,
  n_emg_ch = 2,
  sr_eeg = 500,
  sr_emg = 1000,
  cmc_freq = 20
)
```

Arguments

n_sec	Numeric recording duration in seconds (default 10).
n_eeg_ch	Integer number of EEG channels (default 3).
n_emg_ch	Integer number of EMG channels (default 2).
sr_eeg	Numeric EEG sampling rate in Hz (default 500).
sr_emg	Numeric EMG sampling rate in Hz (default 1000).
cmc_freq	Numeric corticomuscular coherence frequency in Hz (default 20).

Details

This function is useful for demonstrating and testing multi-channel and multi-modal coupling analyses such as coherence matrices and MultiPhysioExperiment workflows.

Value

A named list with components:

eeg A PhysioExperiment object with EEG data (n_eeg_ch channels at sr_eeg Hz).

emg A PhysioExperiment object with EMG data (n_emg_ch channels at sr_emg Hz).

References

Halliday, D. M., Rosenberg, J. R., Amjad, A. M., Breeze, P., Conway, B. A., & Farmer, S. F. (1995). A framework for the analysis of mixed time series/point process data – theory and application to the study of physiological tremor, single motor unit discharges and electromyograms. *Progress in Biophysics and Molecular Biology*, 64(2–3), 237–278.

See Also

[make_coupled_signals\(\)](#), [make_directed_signals\(\)](#), [MultiPhysioExperiment\(\)](#), [coherenceMatrix\(\)](#)

Examples

```
data <- make_eeg_emg(n_sec = 5, n_eeg_ch = 3, n_emg_ch = 2)
mpe <- MultiPhysioExperiment(EEG = data$eeg, EMG = data$emg)
modalities(mpe)
```

mergePhysio

*Merge two PhysioExperiment objects by combining channels***Description**

Horizontally concatenates the channels (columns) of two PhysioExperiment objects that share the **same** sampling rate. Channel labels are prefixed to avoid name collisions.

Usage

```
mergePhysio(x, y, prefix = c("x_", "y_"))
```

Arguments

x	A PhysioExperiment object.
y	A PhysioExperiment object.
prefix	Character vector of length 2: prefixes added to channel labels from x and y respectively (default c("x_", "y_")).

Details

Both objects must have the same number of time points (rows) and the same sampling rate. Only the first assay of each object is merged.

Value

A single PhysioExperiment with the columns of both inputs.

See Also

[alignToRate\(\)](#), [alignSignals\(\)](#), [MultiPhysioExperiment\(\)](#)

Examples

```
pe1 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(100 * 4), nrow = 100)),
  colData = S4Vectors::DataFrame(label = paste0("A", 1:4)),
  samplingRate = 500
)
pe2 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(100 * 4), nrow = 100)),
  colData = S4Vectors::DataFrame(label = paste0("B", 1:4)),
  samplingRate = 500
)
merged <- mergePhysio(pe1, pe2)
```

modulationIndex	<i>Modulation-index comodulogram with surrogate inference</i>
-----------------	---

Description

Computes Tort, Canolty, or Ozkurt phase-amplitude coupling over a declared frequency grid. Each surrogate amplitude-source signal produces one complete matrix, so cell-wise p-values share a coherent null replicate. P-values use the conservative Phipson-Smyth correction and are adjusted over the complete grid.

Usage

```
modulationIndex(
  x,
  phase_freqs,
  amp_freqs,
  sr = NULL,
  y = NULL,
  method = c("tort", "canolty", "ozkurt"),
  phase_bw = 2,
  amp_bw = 10,
  n_bins = 18L,
  order = 4L,
  n_surrogates = 199L,
  surrogate_type = c("timeshift", "phase", "iaaft", "aaft"),
  alpha = 0.05,
  p_adjust_method = "BH",
  seed = NULL,
  cores = 1L,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	A numeric signal or <code>PhysioExperiment</code> providing the phase source, or a <code>MultiPhysioExperiment</code> .
phase_freqs	Strictly increasing phase-band centre frequencies in Hz.
amp_freqs	Strictly increasing amplitude-band centre frequencies in Hz.
sr	Sampling rate in Hz, required for numeric input.
y	Optional amplitude-source signal or <code>PhysioExperiment</code> . When <code>NULL</code> , a non- <code>MultiPhysioExperiment</code> x supplies both sources.
method	"tort", "canolty", or "ozkurt".

phase_bw, amp_bw	Half-bandwidths in Hz.
n_bins	Number of phase bins for Tort MI.
order	Bandpass filter order.
n_surrogates	Number of full-grid surrogates. Use zero to disable inference explicitly.
surrogate_type	"timeshift", "phase", "iaaft", or "aaft".
alpha	Significance level.
p_adjust_method	A method from <code>stats::p.adjust.methods</code> , applied once across the complete frequency grid.
seed	Optional non-negative integer seed.
cores	Positive integer number of worker processes.
modality_x, modality_y, channels_x, channels_y	Passed to the shared signal-pair extractor.

Details

PAC is an association and can be produced by waveform shape, filter leakage, edge transients, non-stationarity, or common inputs. A high or significant value does not establish cross-frequency communication or a biological mechanism. The returned threshold is the unadjusted cell-wise null quantile; the significant mask is instead defined from adjusted p-values. Benjamini-Hochberg controls the false discovery rate, not strong family-wise error. Requested and exact realized pass bands are both retained; a structured diagnostic records any adjustment at the filter's numerical boundary.

Value

A `pac_comodulogram` list containing the observed matrix, grid and realized bands, peak, cell-wise p-values, adjusted p-values, significance mask, null thresholds and summaries, surrogate settings, and structured diagnostic strings. Inferential matrices are NULL when `n_surrogates = 0`.

References

- Tort, A. B. L., et al. (2010). *Journal of Neurophysiology*, 104, 1195-1210.
- Canolty, R. T., et al. (2006). *Science*, 313, 1626-1628.
- Ozkurt, T. E., & Schnitzler, A. (2011). *Journal of Neuroscience Methods*, 201, 438-443.
- Phipson, B., & Smyth, G. K. (2010). *Statistical Applications in Genetics and Molecular Biology*, 9, Article 39.

See Also

[comodulogram\(\)](#), [phaseAmplitudeCoupling\(\)](#), [plotComodulogram\(\)](#)

Examples

```
## Not run:
set.seed(1)
t <- (0:4999) / 500
ph <- 2 * pi * 6 * t
sig <- sin(ph) + (0.1 + (1 + cos(ph - pi)) / 2) * sin(2 * pi * 50 * t)
mi <- modulationIndex(sig, c(4, 6, 8), c(40, 50, 60), sr = 500,
                      n_surrogates = 99, seed = 1)

plotComodulogram(mi)

## End(Not run)
```

multimodalGaitIndex	<i>Composite multi-modal gait deviation index</i>
---------------------	---

Description

A Gait-Deviation-Index-style score (Schwartz & Rozumalski 2008) generalised across modalities: how far each subject's multi-modal gait feature vector lies from a normative reference cloud, scaled so a typical control scores ~100 and every 10 points is one standard deviation of deviation. Optionally broken down per modality to show which modality drives the deviation.

Usage

```
multimodalGaitIndex(
  features,
  reference,
  blocks = NULL,
  ve = 0.95,
  mean_index = 100,
  sd_index = 10
)
```

Arguments

features	An $n \times p$ matrix of subjects' multi-modal gait features (kinematics, kinetics, EMG summaries, spatiotemporal, ... concatenated).
reference	An $m \times p$ matrix of the same features for a normative (control) sample.
blocks	Optional named list mapping each modality to its column indices in features, for a per-modality breakdown.
ve	Reference variance retained by the PCA whitening (default 0.95).
mean_index, sd_index	Scaling (default 100 and 10, as for GDI).

Value

a `multimodal_index` object: `index` (per subject; ~100 = typical), `distance`, `reference_mean_index`, and (if `blocks` given) `by_modality` (subject x modality index matrix).

References

Schwartz MH, Rozumalski A (2008) Gait Posture 28:351-357.

See Also

[jive\(\)](#), [fuseBlocks\(\)](#)

Examples

```
set.seed(1); ctrl <- matrix(rnorm(60 * 8), 60, 8)
subj <- rbind(matrix(rnorm(5 * 8), 5, 8),           # typical
              matrix(rnorm(5 * 8, 3), 5, 8))        # deviant
multimodalGaitIndex(subj, ctrl)$index
```

MultiPhysioExperiment-crossmodal

The multimodal container

Description

PhysioCrossModal used to define its own MultiPhysioExperiment class, holding experiments / alignment / sampleMap / couplingResults. That class no longer exists here: the ecosystem has a single multimodal container, defined in **PhysioExperiment**, and this package is an analysis package that takes it as input.

Details

The constructor and every accessor keep their names and return types, so existing code runs unchanged: MultiPhysioExperiment(), experiments(), modalities(), samplingRates(), nModalities(), alignment(), alignment<-, length(), names(), [, [,. alignment() is now a view computed from the container's shared clock rather than a separately stored table, so the alignment and the signals cannot drift apart.

Containers saved under the old class are converted by [migrateContainer](#) or read with [readPhysioRDS](#).

See Also

[MultiPhysioExperiment](#), [couplingAnalysis](#), [couplingResults](#)

`MultiPhysioExperiment-crossmodal-methods`*Methods on the multimodal container*

Description

`length()`, `names()`, `[]` and `[[` for the multimodal container are defined once, in **PhysioExperiment**, and are imported from there. They are not redefined here: two definitions of the same method on the same class is exactly the divergence this package's own container was unified away to prevent.

Details

One behaviour changed in the move, deliberately. The `[]` defined here selected a time window by computing `floor(t * rate) + 1` from each modality's sampling rate alone. That ignored each stream's start offset on the shared clock, so a window taken from modalities that did not start together returned mismatched spans; it also rebuilt the alignment table from defaults, resetting every offset to zero, and discarded the sample map. Selection is now made on each sample's actual time, the clock is preserved, and each stream's offset is updated to its first retained sample. For windows whose bounds land on sample times - including all simultaneously started data with grid-aligned bounds - the selection is unchanged.

The cache accessors `couplingResults()` and `couplingResults<-` are defined in `coupling-wrapper.R`, where the cache itself lives.

See Also

[timeWindow](#), [MultiPhysioExperiment](#)

`multipleFactorAnalysis`*Multiple Factor Analysis (multi-block PCA)*

Description

A global PCA over several blocks in which each block is first scaled by its leading singular value, so a large or high-variance modality cannot dominate the shared factors (Escofier & Pages 1994). Gives global factor scores on the shared observations, with each block's contribution to each factor.

Usage

```
multipleFactorAnalysis(  
  blocks,  
  ncomp = 2L,  
  center = TRUE,  
  scale_columns = FALSE  
)
```

Arguments

`blocks` A list of ≥ 2 matrices, each $n \times p_k$ (shared rows).
`ncomp` Number of global factors (default 2).
`center, scale_columns` Center (and, if TRUE, unit-variance scale) each column before the block weighting.

Value

an mfa object: scores ($n \times ncomp$ global factor scores), eigenvalues, explained (proportion of variance per factor), contributions (block $\times$ factor, each block's share of a factor's inertia), `block_weights`.

References

Escofier B, Pages J (1994) Comput Stat Data Anal 18:121-140.

See Also

[jive\(\)](#), [rvCoefficient\(\)](#)

Examples

```
set.seed(1)
s <- matrix(rnorm(30 * 2), 30, 2)
b1 <- s %*% matrix(rnorm(2 * 5), 2, 5) + matrix(rnorm(30 * 5, 0, .3), 30, 5)
b2 <- s %*% matrix(rnorm(2 * 4), 2, 4) + matrix(rnorm(30 * 4, 0, .3), 30, 4)
multipleFactorAnalysis(list(kin = b1, grf = b2))$explained
```

multitaperCoherence *Multitaper coherence*

Description

Computes magnitude-squared coherence using multitaper spectral estimation with Discrete Prolate Spheroidal Sequences (DPSS / Slepian tapers). Multitaper estimation provides lower variance than Welch's method for short signals or when frequency resolution is critical.

Usage

```
multitaperCoherence(
  x,
  y = NULL,
  sr = NULL,
  nw = 4,
  k = NULL,
  freq_range = NULL,
```

```

    modality_x = NULL,
    modality_y = NULL,
    channels_x = 1L,
    channels_y = 1L,
    ...
)

```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when x is an MPE.
sr	Numeric sampling rate in Hz (required when x/y are numeric).
nw	Numeric time-bandwidth product (default 4). Controls the trade-off between frequency resolution and variance.
k	Integer number of tapers (default $2 * nw - 1$).
freq_range	Optional numeric vector c(low, high) to restrict output frequencies.
modality_x, modality_y	Character modality names for MPE input.
channels_x, channels_y	Integer channel indices (default 1).
...	Currently unused.

Value

A list with components:

coherence Numeric vector of magnitude-squared coherence values in $[0, 1]$.

frequencies Numeric vector of corresponding frequencies in Hz.

confidence_limit Numeric scalar giving the 95\ threshold based on the number of tapers.

References

Thomson, D. J. (1982). Spectrum estimation and harmonic analysis. *Proceedings of the IEEE*, 70(9), 1055–1096.

Carter, G. C. (1987). Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2), 236–255.

See Also

[coherence\(\)](#), [crossSpectrum\(\)](#), [couplingAnalysis\(\)](#)

Examples

```

sr <- 500
t <- seq(0, 10, length.out = sr * 10)
x <- sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
result <- multitaperCoherence(x, y, sr = sr)

```

mutualInformation	<i>Mutual Information (histogram estimator)</i>
-------------------	---

Description

Computes the mutual information (Shannon, 1948) between two signals – the amount of information one signal carries about the other, a symmetric, *undirected*, nonlinear measure of statistical dependence. It is the information-theoretic complement of [transferEntropy](#) (directed): $I(X; Y) = \sum p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$, zero iff the two signals are independent.

Usage

```
mutualInformation(x, y, bins = 16L)
```

Arguments

x, y	Numeric vectors of equal length.
bins	Number of equal-width bins per signal (default 16).

Details

This is the **histogram** estimator: each signal is discretized into bins equal-width bins and the mutual information of the resulting joint distribution is returned, in **nats** (natural log), matching `sklearn.metrics.mutual_info_score` on the same binning. The histogram estimator has a positive finite-sample bias (a shuffled surrogate has non-zero MI), so compare against a shuffled control or use more data / fewer bins for a cleaner estimate.

Value

A single numeric value, the mutual information in nats.

References

Shannon, C. E. (1948). "A mathematical theory of communication." *Bell System Technical Journal*, 27(3), 379–423.

See Also

[transferEntropy](#) for the directed (model-free) coupling, [coherence](#) for frequency-domain coupling.

Examples

```
set.seed(1)
a <- rnorm(1000); b <- a + rnorm(1000)
mutualInformation(a, b) # > 0 (dependent)
mutualInformation(a, rnorm(1000)) # near the finite-sample floor
```

orthogonalizedAEC	<i>Orthogonalised amplitude-envelope correlation (leakage-corrected)</i>
-------------------	--

Description

Amplitude-envelope correlation after pairwise orthogonalisation (Hipp et al. 2012; Brookes et al. 2012): the instantaneous zero-lag (leakage) component is projected out before correlating band-limited amplitude envelopes, so shared instantaneous mixing does not create spurious coupling. Symmetrised over the two orthogonalisation directions.

Usage

```
orthogonalizedAEC(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  order = 4L,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x, y	Signals (or a single object from which a pair is extracted).
sr	Sampling rate in Hz.
freq_band	Numeric c(low, high) band in Hz.
order	Butterworth filter order for the band-pass (default 4).
modality_x, modality_y, channels_x, channels_y	Passed to the internal signal-pair extractor.

Value

A single leakage-corrected envelope correlation in $[-1, 1]$.

References

Hipp JF, et al. (2012). Large-scale cortical correlation structure of spontaneous oscillatory activity. *Nat Neurosci* 15:884-890.

Examples

```
set.seed(1); sr <- 200; n <- 4000
x <- as.numeric(stats::filter(rnorm(n), rep(1/5, 5), sides = 2)); x[is.na(x)] <- 0
y <- c(rep(0, 10), x[seq_len(n - 10)]) + rnorm(n, sd = 0.5)
orthogonalizedAEC(x, y, sr = sr, freq_band = c(8, 12))
```

pairwisePhaseConsistency
Pairwise phase consistency (PPC)

Description

The bias-free phase-synchrony estimator of Vinck et al. (2010): $PPC = (|\sum e^{i\Delta\phi}|^2 - N) / (N(N-1))$. Unlike the PLV it is not inflated by the sample size.

Usage

```
pairwisePhaseConsistency(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  order = 4L,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x, y	Signals (or a single object from which a pair is extracted).
sr	Sampling rate in Hz.
freq_band	Numeric c(low, high) band in Hz.
order	Butterworth filter order for the band-pass (default 4).
modality_x, modality_y, channels_x, channels_y	Passed to the internal signal-pair extractor.

Value

A single PPC value (approaches 1 for perfect locking, 0 for none).

References

Vinck M, et al. (2010). The pairwise phase consistency. *NeuroImage* 51:112-122.

partialDirectedCoherence

Partial Directed Coherence for multichannel / multimodal signals

Description

Frequency-resolved directed connectivity via Partial Directed Coherence (Baccala & Sameshima 2001), computed from the coefficient matrix of an MVAR model ([PhysioExperiment::mvarFit\(\)](#)); numerically identical to PhysioEEG's eegPDC. Unlike DTF, PDC reflects only direct influences, so a purely indirect pathway gives PDC near zero.

Usage

```
partialDirectedCoherence(
  x,
  y = NULL,
  sr = NULL,
  order = 5L,
  freqs = NULL,
  generalized = TRUE,
  band = NULL,
  mvar_method = "ols",
  modality_x = NULL,
  modality_y = NULL,
  channels_x = NULL,
  channels_y = NULL
)
```

Arguments

x	A multichannel matrix (time x channels) or PhysioExperiment (with y = NULL), or one signal / PhysioExperiment of a pair.
y	The second signal / PhysioExperiment of a pair, or NULL for a single multichannel input.
sr	Sampling rate in Hz (required for numeric-matrix input).
order	MVAR model order (default: 5), or NULL to select automatically.
freqs	Frequencies in Hz (default: 128 points from 0 to the Nyquist frequency).
generalized	Use generalized PDC (inverse-noise weighting; default TRUE).
band	Optional length-2 band in Hz over which to average the returned matrix (default: all frequencies).
mvar_method	MVAR estimator (default: "ols").
modality_x, modality_y, channels_x, channels_y	Passed to the shared signal-pair extraction when y is supplied.

Value

A list as in `directedTransferFunction()` with a pdc array and band-averaged matrix; for a two-channel input, pdc_xy and pdc_yx.

References

Baccala, L. A., & Sameshima, K. (2001). Partial directed coherence: a new concept in neural structure determination. *Biological Cybernetics*, 84(6), 463-474.

See Also

`directedTransferFunction()`, `transferEntropy()`

Examples

```
set.seed(1)
X <- matrix(rnorm(1500), ncol = 3)
res <- partialDirectedCoherence(X, sr = 100, order = 4)
dim(res$pdcs)
```

phaseAmplitudeCoupling

Phase-amplitude coupling

Description

Quantifies how the amplitude of a fast oscillation (in `amp_band`) is modulated by the phase of a slow oscillation (in `phase_band`). The phase is taken from `x` and the amplitude envelope from `y` (or from `x` itself when `y` is `NULL`), each via a bandpass filter and the Hilbert transform. Supported measures are the Tort modulation index (a normalized Kullback-Leibler distance of the phase-amplitude distribution from uniform, in 0..1), the Canolty mean vector length (raw), the Ozkurt normalized mean vector length (in 0..1), and a phase-locking value.

Usage

```
phaseAmplitudeCoupling(
  x,
  y = NULL,
  sr = NULL,
  phase_band = c(4, 8),
  amp_band = c(30, 80),
  method = c("tort", "canolty", "ozkurt", "plv"),
  n_bins = 18L,
  order = 4L,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	A numeric signal or <code>PhysioExperiment</code> (phase source), or a <code>MultiPhysioExperiment</code> .
y	A numeric signal or <code>PhysioExperiment</code> (amplitude source), or <code>NULL</code> to use x for both.
sr	Sampling rate in Hz (required for numeric input).
phase_band	Numeric length-2 phase frequency band in Hz (default: <code>c(4, 8)</code>).
amp_band	Numeric length-2 amplitude frequency band in Hz (default: <code>c(30, 80)</code>).
method	"tort", "canolty", "ozkurt", or "plv".
n_bins	Number of phase bins for the Tort modulation index (default: 18).
order	Bandpass filter order (default: 4).
modality_x, modality_y, channels_x, channels_y	Passed to the shared signal-pair extraction.

Value

A list with `pac` (the coupling value), `method`, `phase_band`, `amp_band`, and, for the Tort method, the per-bin normalized amplitude distribution and `bin_centers`.

References

Tort, A. B. L., et al. (2010). Measuring phase-amplitude coupling between neuronal oscillations of different frequencies. *Journal of Neurophysiology*, 104(2), 1195-1210.

Canolty, R. T., et al. (2006). High gamma power is phase-locked to theta oscillations in human neocortex. *Science*, 313(5793), 1626-1628.

See Also

[comodulogram\(\)](#), [plotComodulogram\(\)](#), [surrogateTest\(\)](#)

Examples

```
set.seed(1)
t <- (0:4999) / 500
ph <- 2 * pi * 6 * t
sig <- sin(ph) + (0.1 + (1 + cos(ph - pi)) / 2) * sin(2 * pi * 50 * t)
phaseAmplitudeCoupling(sig, sr = 500, phase_band = c(4, 8),
                        amp_band = c(35, 65))$pac
```

phaseLagIndex	<i>Phase Lag Index (PLI)</i>
---------------	------------------------------

Description

Computes the Phase Lag Index between two signals. PLI measures the asymmetry of the distribution of phase differences and is insensitive to zero-lag (volume conduction) effects.

Usage

```
phaseLagIndex(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x	Numeric vector, PhysioExperiment, or MultiPhysioExperiment.
y	Numeric vector or PhysioExperiment (NULL when x is an MPE).
sr	Numeric sampling rate in Hz (required when x/y are numeric).
freq_band	Numeric vector of length 2 specifying the frequency band c(low, high) in Hz. Required.
modality_x	Character modality name in MPE for x signal.
modality_y	Character modality name in MPE for y signal.
channels_x	Integer channel index to extract from x (default 1).
channels_y	Integer channel index to extract from y (default 1).

Details

$$PLI = \left| \frac{1}{N} \sum_{t=1}^N \text{sign}(\sin(\phi_x(t) - \phi_y(t))) \right|$$

A PLI of 0 indicates either no coupling or symmetric (zero-lag) coupling. A PLI of 1 indicates perfect non-zero-lag phase locking.

Value

A named list with components:

pli Numeric scalar in $[0, 1]$ – the Phase Lag Index.

phase_diff Numeric vector of instantaneous phase differences (radians).

References

Stam, C. J., Nolte, G., & Daffertshofer, A. (2007). Phase lag index: Assessment of functional connectivity from multi channel EEG and MEG with diminished bias from common sources. *Human Brain Mapping*, 28(11), 1178–1193.

See Also

[phaseLockingValue\(\)](#), [weightedPLI\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t)
y <- sin(2 * pi * 10 * t + pi / 3)
result <- phaseLagIndex(x, y, sr = sr, freq_band = c(8, 12))
result$pli
```

phaseLockingValue	<i>Phase Locking Value (PLV)</i>
-------------------	----------------------------------

Description

Computes the Phase Locking Value between two signals, measuring the consistency of phase difference across time. A PLV of 1 indicates perfect phase locking; a PLV of 0 indicates no consistent phase relationship.

Usage

```
phaseLockingValue(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  method = c("hilbert", "wavelet"),
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

<code>x</code>	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
<code>y</code>	Numeric vector or <code>PhysioExperiment</code> (NULL when <code>x</code> is an MPE).
<code>sr</code>	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).
<code>freq_band</code>	Numeric vector of length 2 specifying the frequency band c(low, high) in Hz. Required.
<code>method</code>	Character; phase extraction method. Currently only "hilbert" is supported. "wavelet" is reserved for future use.
<code>modality_x</code>	Character modality name in MPE for <code>x</code> signal.
<code>modality_y</code>	Character modality name in MPE for <code>y</code> signal.
<code>channels_x</code>	Integer channel index to extract from <code>x</code> (default 1).
<code>channels_y</code>	Integer channel index to extract from <code>y</code> (default 1).

Details

The signals are bandpass-filtered to `freq_band` and then Hilbert- transformed to extract instantaneous phase. PLV is computed as:

$$PLV = \left| \frac{1}{N} \sum_{t=1}^N e^{i(\phi_x(t) - \phi_y(t))} \right|$$

Value

A named list with components:

plv Numeric scalar in [0, 1] – the Phase Locking Value.

phase_diff Numeric vector of instantaneous phase differences (radians, [-pi, pi]).

References

Lachaux, J.-P., Rodriguez, E., Martinerie, J., & Varela, F. J. (1999). Measuring phase synchrony in brain signals. *Human Brain Mapping*, 8(4), 194–208.

See Also

[phaseLagIndex\(\)](#), [weightedPLI\(\)](#), [waveletPLV\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
# Two phase-locked sinusoids
sr <- 500
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t)
y <- sin(2 * pi * 10 * t + pi / 4)
result <- phaseLockingValue(x, y, sr = sr, freq_band = c(8, 12))
result$plv
```

phaseSlopeIndex	<i>Phase Slope Index (directed, volume-conduction robust)</i>
-----------------	---

Description

The Phase Slope Index (Nolte et al. 2008) estimates the direction of information flow from the slope of the coherency phase across a frequency band, and is insensitive to instantaneous (zero-lag) mixing. Positive values indicate that x leads (drives) y.

Usage

```
phaseSlopeIndex(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  nperseg = 256L,
  noverlap = NULL,
  normalize = TRUE,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

x, y	Signals (or a single object from which a pair is extracted).
sr	Sampling rate in Hz.
freq_band	Numeric c(low, high) band in Hz.
nperseg, noverlap	Welch segment length / overlap for the cross-spectrum.
normalize	If TRUE (default), scale into [-1, 1] by the summed coherency magnitude; if FALSE, return the raw imaginary sum.
modality_x, modality_y, channels_x, channels_y	Passed to the internal signal-pair extractor.

Value

A single Phase Slope Index value.

References

Nolte G, et al. (2008). Robustly estimating the flow direction of information in complex physical systems. *Phys Rev Lett* 100:234101. doi:[10.1103/PhysRevLett.100.234101](https://doi.org/10.1103/PhysRevLett.100.234101)

Examples

```
set.seed(1); sr <- 200; n <- 4000
x <- as.numeric(stats::filter(rnorm(n), rep(1/5, 5), sides = 2)); x[is.na(x)] <- 0
y <- c(rep(0, 8), x[seq_len(n - 8)]) # y lags x
phaseSlopeIndex(x, y, sr = sr, freq_band = c(1, 40))
```

plotCoherenceSpectrum *Plot coherence spectrum*

Description

Displays the coherence as a function of frequency from the result of [coherence](#). Optionally draws the 95% a horizontal dashed line.

Usage

```
plotCoherenceSpectrum(
  result,
  show_threshold = TRUE,
  title = "Coherence Spectrum",
  ...
)
```

Arguments

result	A list returned by coherence , containing at least \$coherence and \$frequencies. If \$confidence_limit is present it is drawn as a threshold line.
show_threshold	Logical; if TRUE (default) and a confidence limit is available, draw it on the plot.
title	Character string for the plot title (default "Coherence Spectrum").
...	Additional arguments (currently unused).

Value

A ggplot object.

References

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.
[doi:10.1007/9783319242774](https://doi.org/10.1007/9783319242774)

See Also

[coherence\(\)](#), [multitaperCoherence\(\)](#), [plotCouplingMatrix\(\)](#)

Examples

```

sr <- 500
t <- seq(0, 10, length.out = sr * 10)
x <- sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 20 * t) + 0.2 * rnorm(length(t))
res <- coherence(x, y, sr = sr)
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plotCoherenceSpectrum(res)
}

```

plotComodulogram	<i>Plot a comodulogram</i>
------------------	----------------------------

Description

Renders a comodulogram (from [comodulogram\(\)](#)) as a heatmap.

Usage

```

plotComodulogram(
  result,
  title = "Comodulogram",
  mask = NULL,
  nonsignificant_alpha = 0.2
)

```

Arguments

result	A comodulogram result list.
title	Plot title.
mask	Optional logical significance mask or numeric p-value matrix. When NULL, uses result\$significant if present.
nonsignificant_alpha	Opacity for non-significant cells.

Value

A ggplot object.

See Also

[comodulogram\(\)](#), [modulationIndex\(\)](#)

Examples

```
## Not run:
plotComodulogram(comodulogram(sig, sr = 500,
  phase_freqs = c(4, 6, 8), amp_freqs = c(30, 50, 70)))

## End(Not run)
```

plotCouplingMatrix	<i>Plot a coupling matrix as a heatmap</i>
--------------------	--

Description

Displays a matrix of coupling values (e.g., coherence between all channel pairs) as a colour-coded heatmap using `ggplot2::geom_tile()`.

Usage

```
plotCouplingMatrix(
  mat,
  title = "Coupling Matrix",
  low_colour = "white",
  high_colour = PhysioExperiment::physioPalette(2, "sequential")[1],
  ...
)
```

Arguments

<code>mat</code>	Numeric matrix of coupling values. Row and column names, if present, are used as axis labels.
<code>title</code>	Character string for the plot title (default "Coupling Matrix").
<code>low_colour</code>	Character colour for the low end of the scale (default "white").
<code>high_colour</code>	Character colour for the high end of the scale (default: the dark end of the colorblind-safe sequential palette, <code>PhysioExperiment::physioPalette(2, "sequential")[1]</code>).
<code>...</code>	Additional arguments passed to <code>ggplot2::theme()</code> .

Value

A `ggplot` object.

References

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.
[doi:10.1007/9783319242774](https://doi.org/10.1007/9783319242774)

See Also

[coherenceMatrix\(\)](#), [couplingMatrix\(\)](#), [surrogateMatrixTest\(\)](#)

Examples

```
mat <- matrix(runif(16), nrow = 4,
              dimnames = list(paste0("Ch", 1:4), paste0("Ch", 1:4)))
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plotCouplingMatrix(mat)
}
```

plotCouplingTimecourse

Plot time-varying coupling from sliding-window analysis

Description

Displays the peak cross-correlation over time from the result of [slidingCrossCorrelation](#).

Usage

```
plotCouplingTimecourse(result, title = "Coupling Time Course", ...)
```

Arguments

result	A list returned by slidingCrossCorrelation , containing at least \$times and \$peak_correlations.
title	Character string for the plot title (default "Coupling Time Course").
...	Additional arguments (currently unused).

Value

A ggplot object.

References

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.
[doi:10.1007/9783319242774](https://doi.org/10.1007/9783319242774)

See Also

[slidingCrossCorrelation\(\)](#), [crossCorrelation\(\)](#), [plotCoherenceSpectrum\(\)](#)

Examples

```
sr <- 500
set.seed(1)
x <- rnorm(5000)
y <- c(rep(0, 10), x[1:4990])
res <- slidingCrossCorrelation(x, y, sr = sr,
                             window_sec = 1, step_sec = 0.5)
if (requireNamespace("ggplot2", quietly = TRUE)) {
```

```
    plotCouplingTimecourse(res)
  }
```

plotWaveletCoherence *Plot wavelet coherence time-frequency map*

Description

Displays wavelet coherence (or wavelet PLV) as a filled time-frequency heatmap with optional Cone of Influence (COI) overlay.

Usage

```
plotWaveletCoherence(
  result,
  show_coi = TRUE,
  title = "Wavelet Coherence",
  fill_label = "Coherence",
  ...
)
```

Arguments

result	List returned by waveletCoherence or waveletPLV .
show_coi	Logical; overlay COI boundary (default TRUE if coi is present in the result).
title	Character plot title (default "Wavelet Coherence").
fill_label	Character legend label (default "Coherence").
...	Additional arguments passed to <code>ggplot2::theme()</code> .

Value

A `ggplot` object.

References

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.
[doi:10.1007/9783319242774](https://doi.org/10.1007/9783319242774)

See Also

[waveletCoherence\(\)](#), [waveletPLV\(\)](#), [plotCoherenceSpectrum\(\)](#)

Examples

```

sr <- 200
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
result <- waveletCoherence(x, y, sr = sr, frequencies = seq(5, 20))
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plotWaveletCoherence(result)
}

```

plsBlocks

*Two-block partial least squares (PLS-SVD)***Description**

Finds the linear combinations of two modality blocks with maximal cross-modal COVARIANCE (unlike CCA's correlation), by the singular value decomposition of the cross-covariance and successive deflation. Robust when features outnumber observations, so it suits high-dimensional modalities.

Usage

```
plsBlocks(X, Y, ncomp = 2L)
```

Arguments

X, Y	Matrices with the same n rows (observations).
ncomp	Number of PLS components (default 2).

Value

a pls_blocks object: xweights, yweights (the paired directions), xscores, yscores, covariance (per-component score covariance) and correlation (per-component score correlation).

References

Wold H (1975); Bookstein (1994) PLS-SVD; Krishnan et al. (2011).

See Also

[cca\(\)](#), [coupledNMF\(\)](#)

Examples

```

set.seed(1); z <- matrix(rnorm(40), 40, 1)
X <- z %*% matrix(rnorm(6), 1, 6) + matrix(rnorm(40 * 6, 0, .3), 40, 6)
Y <- z %*% matrix(rnorm(5), 1, 5) + matrix(rnorm(40 * 5, 0, .3), 40, 5)
plsBlocks(X, Y, ncomp = 1)$correlation

```

representationalSimilarity

Representational similarity across modalities (RSA)

Description

Compares the *geometry of observations* between modalities: each block's representational dissimilarity matrix (RDM – pairwise distances between the shared observations, using that modality's features) is compared across blocks. Two modalities are "representationally similar" if observations that are far apart in one are far apart in the other.

Usage

```
representationalSimilarity(
  blocks,
  method = c("euclidean", "correlation"),
  compare = c("spearman", "pearson"),
  permutations = 0L,
  seed = NULL
)
```

Arguments

blocks	A list of ≥ 2 matrices, each $n \times p_k$ (shared rows).
method	RDM distance: "euclidean" (default) or "correlation" (1 - cor).
compare	RDM comparison: "spearman" (default) or "pearson".
permutations	For a PAIR of blocks, Mantel permutations for a p-value (default 0).
seed	Optional RNG seed.

Value

an *rsa* object: similarity (block x block RSA matrix) and, for two blocks with permutations > 0 , *p_value* (Mantel).

References

Kriegeskorte N, et al. (2008) Front Syst Neurosci 2:4.

See Also

[rvCoefficient\(\)](#), [distanceCorrelation\(\)](#)

Examples

```
set.seed(1); z <- matrix(rnorm(30 * 2), 30, 2)
a <- z %%% matrix(rnorm(2 * 5), 2, 5); b <- z %%% matrix(rnorm(2 * 4), 2, 4)
representationalSimilarity(list(a = a, b = b))$similarity
```

rvCoefficient	<i>RV coefficient between two modality blocks</i>
---------------	---

Description

The multivariate generalisation of a squared correlation: how much the two blocks' configurations of observations agree, in $[0, 1]$ (0 = unrelated, 1 = identical up to rotation/scaling).

Usage

```
rvCoefficient(X, Y, center = TRUE, permutations = 0L, seed = NULL)
```

Arguments

X, Y	Matrices with the same n rows (observations).
center	Column-centre each block first (default TRUE).
permutations	Row permutations for the null p-value (default 0 = skip).
seed	Optional RNG seed for the permutation test.

Value

an `rv_coefficient` object: `rv`, `p_value`, `n`.

References

Robert P, Escoufier Y (1976) Appl Stat 25:257-265.

See Also

[distanceCorrelation\(\)](#), [representationalSimilarity\(\)](#), [cca\(\)](#)

Examples

```
set.seed(1); z <- matrix(rnorm(40 * 2), 40, 2)
X <- z %*% matrix(rnorm(2 * 5), 2, 5) + matrix(rnorm(40 * 5, 0, .3), 40, 5)
Y <- z %*% matrix(rnorm(2 * 4), 2, 4) + matrix(rnorm(40 * 4, 0, .3), 40, 4)
rvCoefficient(X, Y)$rv
```

slidingCrossCorrelation

Sliding-window cross-correlation

Description

Computes cross-correlation in sliding (overlapping) windows to track how time-domain coupling varies over time. For each window position, [crossCorrelation](#) is called and the results are assembled into a matrix.

Usage

```
slidingCrossCorrelation(
    x,
    y = NULL,
    sr = NULL,
    window_sec = 1,
    step_sec = 0.5,
    max_lag = NULL,
    modality_x = NULL,
    modality_y = NULL,
    channels_x = 1L,
    channels_y = 1L
)
```

Arguments

x	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
y	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when x is a <code>MultiPhysioExperiment</code> .
sr	Numeric sampling rate in Hz (required when x/y are numeric).
window_sec	Numeric window length in seconds (default 1).
step_sec	Numeric step size in seconds (default 0.5).
max_lag	Integer maximum lag in samples for each window. If <code>NULL</code> , defaults to <code>floor(window_samples / 4)</code> .
modality_x	Character modality name in MPE for the x signal.
modality_y	Character modality name in MPE for the y signal.
channels_x	Integer which channel to extract from x (default 1).
channels_y	Integer which channel to extract from y (default 1).

Value

A named list with components:

correlations Numeric matrix of dimensions (n_windows x n_lags) containing cross-correlation values.

times Numeric vector of window centre times in seconds.

lags Integer vector of lag values in samples.

peak_lags Numeric vector of peak lags (in samples) for each window.

peak_correlations Numeric vector of peak correlation values for each window.

References

Chatfield, C. (2004). *The Analysis of Time Series: An Introduction* (6th ed.). Chapman & Hall/CRC.

See Also

[crossCorrelation\(\)](#), [plotCouplingTimecourse\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
set.seed(1)
x <- rnorm(5000)
y <- c(rep(0, 10), x[1:4990])
result <- slidingCrossCorrelation(x, y, sr = sr,
                                window_sec = 1, step_sec = 0.5)

dim(result$correlations)
result$peak_lags
```

srvfMean

Elastic (Karcher) mean of a set of curves

Description

Computes the elastic template mean of repeated curves by iteratively aligning every curve to the current mean and averaging the aligned curves. Because the phase variability is removed before averaging, peaks are preserved rather than blurred out (unlike a naive cross-curve mean).

Usage

```
srvfMean(curves, t = NULL, iterations = 5L, max_step = 6L, use_fdasrvf = NA)
```

Arguments

curves	A list of numeric curves of the same length, or a matrix whose columns are curves.
t	Optional grid (defaults to 0..1).
iterations	Number of refinement iterations (default: 5).
max_step	Maximum DP step for the pure-R solver (default: 6).
use_fdasrvf	As in elasticAlign() .

Value

A list with the elastic mean curve, the warpings (a matrix, one column per curve), and the aligned curves.

See Also

[elasticAlign\(\)](#)

Examples

```
t <- seq(0, 1, length.out = 80)
curves <- lapply(c(-0.1, 0, 0.1), function(s)
  exp(-((t - (0.5 + s))^2) / (2 * 0.05^2)))
m <- srvfMean(curves)
max(m$mean)
```

surrogateMatrixTest	<i>Surrogate-based significance test for coupling matrices</i>
---------------------	--

Description

Tests each element of a coupling matrix for significance using surrogate testing, with correction for multiple comparisons (FDR or Bonferroni).

Usage

```
surrogateMatrixTest(
  x,
  y = NULL,
  method,
  n_surrogates = 199L,
  surrogate_type = c("phase", "timeshift", "iaaft", "aaft"),
  correction = c("fdr", "bonferroni", "none"),
  alpha = 0.05,
  channels_x = NULL,
  channels_y = NULL,
  modality_x = NULL,
```

```

    modality_y = NULL,
    cores = 1L,
    ...
)

```

Arguments

<code>x</code>	PhysioExperiment or MultiPhysioExperiment.
<code>y</code>	PhysioExperiment or NULL (when <code>x</code> is an MPE).
<code>method</code>	Character coupling method (same options as couplingAnalysis).
<code>n_surrogates</code>	Integer number of surrogates (default 199).
<code>surrogate_type</code>	Character surrogate generation method: "phase" (default; Fourier phase randomization), "timeshift" (circular time shift), "iaaft" (iterative amplitude-adjusted Fourier transform, preserving the power spectrum and value distribution; Schreiber & Schmitz, 1996), or "aaft" (non-iterative amplitude-adjusted FT).
<code>correction</code>	Character correction method: "fdr" (default), "bonferroni", or "none".
<code>alpha</code>	Numeric significance level (default 0.05).
<code>channels_x, channels_y</code>	Integer vectors of channel indices, or NULL for all channels.
<code>modality_x, modality_y</code>	Character modality names for MPE input.
<code>cores</code>	Integer number of parallel cores (default 1).
<code>...</code>	Additional arguments passed to the coupling function.

Value

A list with components:

- matrix** Coupling matrix (observed values).
- p_values** Matrix of raw p-values (same dimensions).
- p_adjusted** Matrix of corrected p-values.
- significant** Logical matrix indicating significance.
- correction** Character correction method used.
- alpha** Numeric significance level.

References

Theiler, J., Eubank, S., Longtin, A., Galdrikian, B., & Farmer, J. D. (1992). Testing for nonlinearity in time series: the method of surrogate data. *Physica D: Nonlinear Phenomena*, 58(1–4), 77–94.

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1), 289–300.

See Also

[surrogateTest\(\)](#), [couplingMatrix\(\)](#), [coherenceMatrix\(\)](#)

surrogateTest

Surrogate-based significance test for coupling

Description

Tests whether the observed coupling between two signals is statistically significant by comparing it against a null distribution generated from surrogate signals. Surrogates are created by randomising the phases (or circularly shifting) one signal, thereby destroying the cross-signal relationship while preserving the autocorrelation structure.

Usage

```
surrogateTest(
  x,
  y = NULL,
  sr = NULL,
  method,
  n_surrogates = 199L,
  surrogate_type = c("phase", "timeshift", "iaaft", "aaft"),
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  cores = 1L,
  ...
)
```

Arguments

x	Numeric vector, PhysioExperiment, or MultiPhysioExperiment.
y	Numeric vector or PhysioExperiment, or NULL when x is an MPE.
sr	Numeric sampling rate in Hz (required when x/y are numeric).
method	Character coupling method, one of "coherence", "plv", "pli", "wpli", "granger", "crosscorrelation", "wavelet_coherence", "wavelet_plv".
n_surrogates	Integer number of surrogates (default 199).
surrogate_type	Character surrogate generation method: "phase" (default; Fourier phase randomization), "timeshift" (circular time shift), "iaaft" (iterative amplitude-adjusted Fourier transform, preserving the power spectrum and value distribution; Schreiber & Schmitz, 1996), or "aaft" (non-iterative amplitude-adjusted FT).
modality_x, modality_y	Character modality names for MPE input.
channels_x, channels_y	Integer channel indices (default 1).

<code>cores</code>	Integer number of parallel cores to use (default 1L). When <code>cores > 1</code> , surrogate computations are distributed across cores using <code>parallel::mclapply()</code> (Unix) or <code>parallel::parLapply()</code> (Windows). Falls back to sequential computation on error.
<code>...</code>	Additional arguments passed to the coupling function (e.g. <code>freq_band</code> , <code>nperseg</code> , <code>order</code>).

Details

The p-value is computed as $(\text{sum}(\text{surr} \geq \text{obs}) + 1) / (n_{\text{surr}} + 1)$, following the conservative correction of Phipson & Smyth (2010).

Value

A list with components:

observed The full coupling result from the original signals.

statistic Numeric scalar: the extracted coupling statistic.

p_value Numeric scalar: surrogate p-value.

surrogate_distribution Numeric vector of surrogate statistics.

threshold_95 Numeric scalar: 95th percentile of surrogates.

References

Theiler, J., Eubank, S., Longtin, A., Galdrikian, B., & Farmer, J. D. (1992). Testing for nonlinearity in time series: the method of surrogate data. *Physica D: Nonlinear Phenomena*, 58(1–4), 77–94.

Phipson, B., & Smyth, G. K. (2010). Permutation P-values should never be zero: calculating exact P-values when permutations are randomly drawn. *Statistical Applications in Genetics and Molecular Biology*, 9(1), Article 39.

See Also

[bootstrapCI\(\)](#), [surrogateMatrixTest\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
result <- surrogateTest(x, y, sr = sr, method = "coherence",
                       n_surrogates = 19, nperseg = 128L)
result$p_value
```

transferEntropy	<i>Transfer entropy (model-free directed information)</i>
-----------------	---

Description

Estimates the transfer entropy from x to y and back (Schreiber 2000), a model-free measure of directed information flow, with a histogram (binning) estimator or the Kraskov-Stoegbauer-Grassberger (KSG) k -nearest-neighbour estimator. With `effective = TRUE` the estimate is bias-corrected by subtracting the mean transfer entropy of IAAFT surrogates of the source, so independent signals give an effective transfer entropy near zero.

Usage

```
transferEntropy(
  x,
  y = NULL,
  sr = NULL,
  k = 1L,
  l = 1L,
  delay = 1L,
  estimator = c("ksg", "histogram"),
  knn = 4L,
  bins = 8L,
  effective = FALSE,
  n_surrogate = 19L,
  seed = NULL,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

<code>x, y</code>	Two numeric signals or <code>PhysioExperiment</code> objects (or a <code>MultiPhysioExperiment</code> via <code>x</code>).
<code>sr</code>	Sampling rate in Hz (required for numeric input).
<code>k</code>	Target (destination) embedding length (default: 1).
<code>l</code>	Source embedding length (default: 1).
<code>delay</code>	Source-to-target lag in samples (default: 1).
<code>estimator</code>	"ksg" (k-NN, in nats) or "histogram" (binning, in bits).
<code>knn</code>	Number of neighbours for the KSG estimator (default: 4).
<code>bins</code>	Number of bins per dimension for the histogram estimator (default: 8).
<code>effective</code>	Subtract the surrogate (IAAFT) null mean (default: FALSE).

n_surrogate Number of surrogates for effective (default: 19).
seed Optional RNG seed for reproducible surrogates.
modality_x, modality_y, channels_x, channels_y
 Passed to the shared signal-pair extraction.

Value

A list with `te_xy`, `te_yx`, and `net` (`te_xy - te_yx`); when `effective = TRUE` also `eff_xy`, `eff_yx`, and the surrogate null standard deviations.

References

Schreiber, T. (2000). Measuring information transfer. *Physical Review Letters*, 85(2), 461-464.
 Kraskov, A., Stoegbauer, H., & Grassberger, P. (2004). Estimating mutual information. *Physical Review E*, 69(6), 066138.

See Also

[directedTransferFunction\(\)](#), [grangerCausality\(\)](#), [surrogateTest\(\)](#)

Examples

```

set.seed(1)
x <- rnorm(600); y <- c(rep(0, 5), 0.7 * x[1:595]) + rnorm(600, sd = 0.5)
transferEntropy(x, y, sr = 100, delay = 5, estimator = "histogram")$net

```

warpApply

Apply a computed warping to a companion signal

Description

Carries a warping computed by [elasticAlign\(\)](#) onto another channel or modality (e.g. warp an EMG envelope by the diffeomorphism that aligns the EEG).

Usage

```
warpApply(signal, gamma, t = NULL)
```

Arguments

signal Numeric signal to warp (same length as the warping grid).
gamma A warping from [elasticAlign\(\)](#).
t Optional grid (defaults to 0..1).

Value

The warped signal.

See Also[elasticAlign\(\)](#)**Examples**

```
t <- seq(0, 1, length.out = 50)
warpApply(sin(2 * pi * t), gamma = t^1.5, t = t)[1:3]
```

waveletCoherence	<i>Time-frequency wavelet coherence</i>
------------------	---

Description

Computes wavelet coherence between two signals using complex Morlet wavelets. The cross-wavelet spectrum and auto-spectra are smoothed with a Gaussian temporal window (width proportional to `smoothing_cycles / frequency`), and coherence is computed as:

Usage

```
waveletCoherence(
  x,
  y = NULL,
  sr = NULL,
  frequencies = seq(1, 40, by = 1),
  n_cycles = 7,
  smoothing_cycles = 3,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  ...
)
```

Arguments

<code>x</code>	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
<code>y</code>	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when <code>x</code> is an MPE.
<code>sr</code>	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).
<code>frequencies</code>	Numeric vector of centre frequencies in Hz (default <code>seq(1, 40, by = 1)</code>).
<code>n_cycles</code>	Numeric number of wavelet cycles (default 7).
<code>smoothing_cycles</code>	Numeric number of cycles for the temporal smoothing Gaussian (default 3).
<code>modality_x, modality_y</code>	Character modality names for MPE input.
<code>channels_x, channels_y</code>	Integer channel indices (default 1).
<code>...</code>	Currently unused.

Details

$$C(t, f) = \frac{|\langle W_{xy}(t, f) \rangle|^2}{\langle |W_x(t, f)|^2 \rangle \cdot \langle |W_y(t, f)|^2 \rangle}$$

Value

A list with components:

coherence Numeric matrix [time x frequency] of coherence values in [0, 1].

phase Numeric matrix [time x frequency] of phase differences (radians).

frequencies Numeric vector of centre frequencies.

times Numeric vector of time points (seconds from start).

coi Numeric vector of Cone of Influence frequencies. At each time point, frequencies below this value are affected by edge artifacts.

References

Torrence, C., & Compo, G. P. (1998). A practical guide to wavelet analysis. *Bulletin of the American Meteorological Society*, 79(1), 61–78.

Grinsted, A., Moore, J. C., & Jevrejeva, S. (2004). Application of the cross wavelet transform and wavelet coherence to geophysical time series. *Nonlinear Processes in Geophysics*, 11(5/6), 561–566.

See Also

[waveletPLV\(\)](#), [coherence\(\)](#), [plotWaveletCoherence\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 200
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
y <- 0.8 * sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
result <- waveletCoherence(x, y, sr = sr, frequencies = seq(5, 20))
```

waveletPLV

Time-frequency wavelet PLV

Description

Computes Phase Locking Value in the time-frequency domain using complex Morlet wavelets. The phase difference between the two signals is computed at each time-frequency point, and PLV is the magnitude of the smoothed unit-phase vector:

Usage

```

waveletPLV(
  x,
  y = NULL,
  sr = NULL,
  frequencies = seq(1, 40, by = 1),
  n_cycles = 7,
  smoothing_cycles = 3,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L,
  ...
)

```

Arguments

<code>x</code>	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
<code>y</code>	Numeric vector or <code>PhysioExperiment</code> , or <code>NULL</code> when <code>x</code> is an MPE.
<code>sr</code>	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).
<code>frequencies</code>	Numeric vector of centre frequencies in Hz (default <code>seq(1, 40, by = 1)</code>).
<code>n_cycles</code>	Numeric number of wavelet cycles (default 7).
<code>smoothing_cycles</code>	Numeric number of cycles for the temporal smoothing Gaussian (default 3).
<code>modality_x, modality_y</code>	Character modality names for MPE input.
<code>channels_x, channels_y</code>	Integer channel indices (default 1).
<code>...</code>	Currently unused.

Details

$$\text{PLV}(t, f) = \left| \langle e^{i\Delta\phi(t,f)} \rangle \right|$$

Value

A list with components:

plv Numeric matrix [time x frequency] of PLV values in $[0, 1]$.

frequencies Numeric vector of centre frequencies.

times Numeric vector of time points (seconds from start).

coi Numeric vector of Cone of Influence frequencies. At each time point, frequencies below this value are affected by edge artifacts.

References

- Torrence, C., & Compo, G. P. (1998). A practical guide to wavelet analysis. *Bulletin of the American Meteorological Society*, 79(1), 61–78.
- Lachaux, J.-P., Rodriguez, E., Martinerie, J., & Varela, F. J. (1999). Measuring phase synchrony in brain signals. *Human Brain Mapping*, 8(4), 194–208.

See Also

[waveletCoherence\(\)](#), [phaseLockingValue\(\)](#), [plotWaveletCoherence\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 200
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t) + 0.3 * rnorm(length(t))
y <- sin(2 * pi * 10 * t + pi/4) + 0.3 * rnorm(length(t))
result <- waveletPLV(x, y, sr = sr, frequencies = seq(5, 20))
```

weightedPLI

Weighted Phase Lag Index (wPLI)

Description

Computes the weighted Phase Lag Index between two signals. wPLI weights each phase-difference sample by the magnitude of the imaginary component of the cross-spectrum, reducing the influence of noise sources that produce phase differences near 0 or π .

Usage

```
weightedPLI(
  x,
  y = NULL,
  sr = NULL,
  freq_band,
  debiased = TRUE,
  modality_x = NULL,
  modality_y = NULL,
  channels_x = 1L,
  channels_y = 1L
)
```

Arguments

<code>x</code>	Numeric vector, <code>PhysioExperiment</code> , or <code>MultiPhysioExperiment</code> .
<code>y</code>	Numeric vector or <code>PhysioExperiment</code> (NULL when <code>x</code> is an MPE).
<code>sr</code>	Numeric sampling rate in Hz (required when <code>x/y</code> are numeric).

freq_band	Numeric vector of length 2 specifying the frequency band c(low, high) in Hz. Required.
debiased	Logical; if TRUE (default), also compute the debiased wPLI estimator.
modality_x	Character modality name in MPE for x signal.
modality_y	Character modality name in MPE for y signal.
channels_x	Integer channel index to extract from x (default 1).
channels_y	Integer channel index to extract from y (default 1).

Details

$$\text{wPLI} = \frac{|\sum \text{Im}(S_{xy})| \cdot \text{sign}(\text{Im}(S_{xy}))|}{\sum |\text{Im}(S_{xy})|}$$

When debiased = TRUE, the debiased estimator from Vinck et al. (2011) is also returned:

$$\text{wPLI}_{\text{debiased}} = \frac{(\sum \text{Im}(S_{xy}))^2 - \sum \text{Im}(S_{xy})^2}{(\sum |\text{Im}(S_{xy})|)^2 - \sum \text{Im}(S_{xy})^2}$$

Value

A named list with components:

wpli Numeric scalar in [0, 1] – the weighted PLI.

wpli_debiased Numeric scalar – the debiased wPLI, or NULL if debiased = FALSE.

n_samples Integer – number of time-domain samples used.

References

Vinck, M., Oostenveld, R., van Wingerden, M., Battaglia, F., & Pennartz, C. M. A. (2011). An improved index of phase-synchronization for electrophysiological data in the presence of volume-conduction, noise and sample-size bias. *NeuroImage*, 55(4), 1548–1565.

Lachaux, J.-P., Rodriguez, E., Martinerie, J., & Varela, F. J. (1999). Measuring phase synchrony in brain signals. *Human Brain Mapping*, 8(4), 194–208.

See Also

[phaseLockingValue\(\)](#), [phaseLagIndex\(\)](#), [couplingAnalysis\(\)](#)

Examples

```
sr <- 500
t <- seq(0, 2, length.out = sr * 2)
x <- sin(2 * pi * 10 * t)
y <- sin(2 * pi * 10 * t + pi / 4)
result <- weightedPLI(x, y, sr = sr, freq_band = c(8, 12))
result$wpli
```

Index

alignObservationBlocks, 3
alignSignals, 4
alignSignals(), 6, 44
alignToRate, 6
alignToRate(), 5, 44

bootstrapCI, 7
bootstrapCI(), 75

cca, 8
cca(), 35, 67, 69
ciPLV, 9
coherence, 10, 12, 17, 35, 36, 52, 62
coherence(), 13, 17, 23, 25, 33, 41, 51, 62, 79
coherenceMatrix, 12
coherenceMatrix(), 12, 20, 43, 64, 73
comodulogram, 13
comodulogram(), 46, 57, 63
coupledNMF, 15
coupledNMF(), 67
couplingAnalysis, 16, 18, 19, 48, 73
couplingAnalysis(), 5, 8, 12, 18, 20, 23, 25, 39, 41, 42, 51, 59, 60, 71, 75, 79, 81, 82
couplingAnalysisCached, 18, 20
couplingAnalysisCached(), 21
couplingMatrix, 19
couplingMatrix(), 13, 64, 73
couplingResults, 20, 48, 49
couplingResults(), 18
couplingResults, MultiPhysioExperiment-method (couplingResults), 20
couplingResults<-(couplingResults), 20
couplingResults<-, MultiPhysioExperiment-method (couplingResults), 20
crossBlockFactor, 21
crossCorrelation, 17, 22, 70
crossCorrelation(), 17, 42, 65, 71
crossSpectrum, 24
crossSpectrum(), 12, 51

crossWaveletTransform, 25

digest, 18
directedTransferFunction, 26
directedTransferFunction(), 56, 77
distanceCorrelation, 28
distanceCorrelation(), 68, 69

elasticAlign, 29
elasticAlign(), 5, 72, 77, 78

fuseBlocks, 30
fuseBlocks(), 48

grangerCausality, 17, 31
grangerCausality(), 17, 28, 42, 77

jive, 34
jive(), 3, 15, 22, 48, 50

laggedCoherence, 35
laggedCoherence(), 12
leidaStates, 36
leidaStates(), 38
leidaTransitions, 38
lodoGeneralization, 38

make_coupled_signals, 40
make_coupled_signals(), 42, 43
make_directed_signals, 41
make_directed_signals(), 41, 43
make_eeg_emg, 42
make_eeg_emg(), 41
mergePhysio, 44
mergePhysio(), 5, 6
migrateContainer, 48
modulationIndex, 45
modulationIndex(), 63
multimodalGaitIndex, 47
multimodalGaitIndex(), 30
MultiPhysioExperiment, 4, 5, 16, 18, 48, 49

MultiPhysioExperiment(), [5](#), [6](#), [43](#), [44](#)
 MultiPhysioExperiment-crossmodal, [48](#)
 MultiPhysioExperiment-crossmodal-methods, [49](#)
 multipleFactorAnalysis, [49](#)
 multipleFactorAnalysis(), [3](#), [22](#), [30](#), [35](#)
 multitaperCoherence, [50](#)
 multitaperCoherence(), [12](#), [25](#), [62](#)
 mutualInformation, [52](#)

 orthogonalizedAEC, [53](#)

 pairwisePhaseConsistency, [54](#)
 partialDirectedCoherence, [55](#)
 partialDirectedCoherence(), [28](#)
 phaseAmplitudeCoupling, [56](#)
 phaseAmplitudeCoupling(), [14](#), [46](#)
 phaseLagIndex, [17](#), [58](#)
 phaseLagIndex(), [60](#), [82](#)
 phaseLockingValue, [17](#), [59](#)
 phaseLockingValue(), [17](#), [33](#), [59](#), [81](#), [82](#)
 phaseSlopeIndex, [61](#)
 PhysioExperiment::mvarFit(), [27](#), [55](#)
 plotCoherenceSpectrum, [62](#)
 plotCoherenceSpectrum(), [65](#), [66](#)
 plotComodulogram, [63](#)
 plotComodulogram(), [14](#), [46](#), [57](#)
 plotCouplingMatrix, [64](#)
 plotCouplingMatrix(), [13](#), [20](#), [62](#)
 plotCouplingTimecourse, [65](#)
 plotCouplingTimecourse(), [71](#)
 plotWaveletCoherence, [66](#)
 plotWaveletCoherence(), [79](#), [81](#)
 plsBlocks, [67](#)
 plsBlocks(), [9](#), [15](#)

 readPhysioRDS, [48](#)
 representationalSimilarity, [68](#)
 representationalSimilarity(), [28](#), [69](#)
 rvCoefficient, [69](#)
 rvCoefficient(), [9](#), [28](#), [50](#), [68](#)

 slidingCrossCorrelation, [65](#), [70](#)
 slidingCrossCorrelation(), [23](#), [65](#)
 srvfMean, [71](#)
 srvfMean(), [30](#)
 surrogateMatrixTest, [72](#)
 surrogateMatrixTest(), [13](#), [20](#), [64](#), [75](#)
 surrogateTest, [7](#), [74](#)
 surrogateTest(), [8](#), [17](#), [39](#), [57](#), [73](#), [77](#)

 timeWindow, [49](#)
 transferEntropy, [52](#), [76](#)
 transferEntropy(), [28](#), [56](#)

 warpApply, [77](#)
 warpApply(), [30](#)
 waveletCoherence, [17](#), [66](#), [78](#)
 waveletCoherence(), [66](#), [81](#)
 waveletPLV, [17](#), [66](#), [79](#)
 waveletPLV(), [60](#), [66](#), [79](#)
 weightedPLI, [17](#), [81](#)
 weightedPLI(), [59](#), [60](#)