

Package ‘metricT’

October 16, 2026

Type Package

Title Directional Fragility Diagnostics for EEG Connectivity

Version 0.2.0

Description Implements Metric-T, a permutation-based diagnostic that flags when the reported direction of a group difference in electroencephalography (EEG) functional connectivity depends on the choice of metric. Direction consistency (DC) converts the weighted phase lag index (wPLI) and magnitude-squared coherence (MSC), both derived from the same cross-spectrum, into the percentage of channel-pair features for which one group exceeds the other; Metric-T is $T = DC(wPLI) - DC(MSC)$. Significance is assessed by aligned label permutation with marginal and Westfall-Young max-T multiplicity control. For small samples the reference set is enumerated completely: exact two-group tests, the complete two-group labelling space (in which cut-points, exclusions and batches are located on one distribution), and a stratified exact test for a continuous covariate. Core functions use base R only; a 'shiny' graphical interface is included. Method: Kimura (2026) <[doi:10.1016/j.neuri.2026.100286](https://doi.org/10.1016/j.neuri.2026.100286)>.

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

Depends R (>= 4.0.0)

Imports stats, graphics, grDevices, utils

Suggests shiny, testthat (>= 3.0.0)

Config/testthat/edition 3

URL <https://github.com/akimura753/metricT>

BugReports <https://github.com/akimura753/metricT/issues>

RoxygenNote 7.3.1

NeedsCompilation no

Author Akira Kimura [aut, cre] (ORCID:
<<https://orcid.org/0000-0003-0220-5678>>)

Maintainer Akira Kimura <a-kimura@paz.ac.jp>
Repository CRAN
Date/Publication 2026-10-16 09:50:13 UTC

Contents

metric_t	2
metric_t_trend	4
mt_bands	5
mt_clean_epochs	6
mt_connectivity	6
mt_dc	7
mt_epoch	8
mt_features_from_eeg_csv	8
mt_features_from_epochs	9
mt_labelings	10
mt_labelset	11
mt_label_space	12
mt_locate	13
mt_motion_bands	14
mt_null_plot	15
mt_read_npy	15
mt_read_npz	16
mt_simulate	16
mt_simulate_motion	18
mt_write_features_csv	19
mt_wy_maxt	20
plot.metricT	21
run_metricT_app	21
Index	23

metric_t	<i>Metric-T: diagnostic for directional fragility</i>
----------	---

Description

For every comparison x band condition, computes DC(wPLI), DC(MSC) and T = DC(wPLI) - DC(MSC) in percentage points, and tests T by permuting the group labels of all subjects. One relabelling is shared by all conditions (aligned permutations), which preserves their dependence and allows Westfall-Young max-T adjustment.

Usage

```
metric_t(
  features,
  group,
  comparisons = NULL,
  control = NULL,
  n_perm = 10000L,
  seed = 42L,
  wy_family = c("control", "band", "all"),
  perm_index = NULL,
  exact = FALSE,
  keep_null = TRUE,
  progress = NULL
)
```

Arguments

features	Named list by band; each element is a list with matrices <code>wpli</code> and <code>msc</code> (subjects x channel pairs).
group	Group label of each subject (row order of the matrices).
comparisons	List of length-2 character vectors <code>c(g1, g2)</code> . Default: each group against control first, then remaining pairs.
control	Optional reference group. If given, the Westfall-Young family is, within each band, the set of comparisons against it.
n_perm	Number of permutations (default 10000).
seed	Random seed.
wy_family	"control", "band" or "all".
perm_index	Optional <code>n_perm</code> x <code>n</code> matrix of permutation indices (1-based); used to replay permutations generated elsewhere.
exact	If TRUE (two groups only), every distinct assignment of the group labels is enumerated instead of sampled, and p-values are exact proportions over that complete reference set. Feasible for small samples (e.g. 4 versus 9 subjects gives 715 assignments).
keep_null	Keep the permutation matrix in the result.
progress	Optional function called with the fraction completed.

Value

Object of class `metricT`: table (one row per condition), `T_perm`, and settings. Column `p_min` is the smallest p-value attainable in that condition (the permutation probability of $|T| = 100$ pp); because DC keeps only the sign of each mean difference, strongly dependent features can make even a complete reversal non-significant, and `p_min` shows when that is the case.

References

Kimura A (2026). Metric-T: A permutation-based diagnostic for directional fragility in EEG functional connectivity analysis. *Neuroscience Informatics* 6:100286. doi:10.1016/j.neuri.2026.100286

Examples

```
sim <- mt_simulate(n = c(A = 12, B = 12), n_ch = 5, n_epochs = 10, seed = 1)
res <- metric_t(sim$features, sim$group, n_perm = 200)
res
```

metric_t_trend	<i>Metric-T for a continuous covariate, with stratified exact permutation</i>
----------------	---

Description

Extends Metric-T from two groups to a numeric covariate (for example years of visual experience). For each metric, direction consistency is the percentage of features positively associated with the covariate, and $T = DC(wPLI) - DC(MSC)$. The covariate is permuted among subjects within strata only (for example recording sessions), so that differences between strata are not treated as chance.

Usage

```
metric_t_trend(
  features,
  x,
  strata = NULL,
  method = c("rank", "linear"),
  exact = NULL,
  n_perm = 10000L,
  max_exact = 2e+05,
  sets = NULL,
  seed = 42L
)
```

Arguments

features	Named list by band, as for metric_t .
x	Numeric vector (one value per subject), or a numeric matrix or data frame with one column per covariate. Several covariates are permuted jointly, which allows a family-wise adjustment across them.
strata	Optional factor defining the strata; NULL for none.
method	"rank" (default) or "linear".
exact	TRUE to enumerate every within-stratum arrangement, FALSE to sample; NULL enumerates when the number of arrangements does not exceed max_exact.
n_perm	Number of random arrangements when not exact.
max_exact	Largest reference set enumerated when exact = NULL.
sets	Optional named list of logical or integer vectors selecting features (columns), tested in addition to the bands as pooled sets across bands.
seed	Random seed for sampled arrangements.

Details

With `method = "rank"` the covariate and every feature are replaced by their standardised ranks within strata, which makes T invariant to any monotone transformation. With `method = "linear"` raw values are centred within strata; for a 0/1 covariate and a single stratum this reproduces the two-group statistic of `metric_t`.

Each feature statistic is a sum of stratum contributions, so the complete reference set ($\prod_s n_s!$ arrangements) is obtained from $\sum_s n_s!$ evaluations followed by additions.

Value

An object of class `"metricT_trend"`: a list with `table` (covariate, band, DC values, T, two-sided p, smallest attainable `p_min`, and `p_family`, the max-|T| adjusted p over all bands and covariates), `n_perm`, `exact` and the stratum sizes.

See Also

`metric_t`, `mt_label_space`

Examples

```
sim <- mt_simulate(n = c(A = 4, B = 4), n_ch = 4, n_epochs = 8, seed = 2)
set.seed(1)
years <- round(runif(8, 0, 40))
session <- rep(c("day1", "day2"), each = 4)
metric_t_trend(sim$features, years, strata = session)
```

mt_bands

Default frequency bands (Hz), as in Kimura (2026)

Description

A band `c(lo, hi)` keeps Fourier bins with `lo <= f < hi`.

Usage

```
mt_bands()
```

Value

Named list of length-2 numeric vectors.

Examples

```
mt_bands()
```

mt_clean_epochs	<i>Detrend epochs and drop those with large excursions</i>
-----------------	--

Description

Minimal preprocessing for raw recordings: each channel of each epoch is linearly detrended (removing offset and slow drift), and an epoch is dropped when the peak-to-peak range of any channel exceeds max_ptp.

Usage

```
mt_clean_epochs(epochs, detrend = TRUE, max_ptp = NULL)
```

Arguments

epochs	Array, epochs x channels x samples (see mt_epoch).
detrend	Remove a least-squares line from each channel of each epoch.
max_ptp	Peak-to-peak rejection threshold in the unit of the data (e.g. microvolts); NULL keeps all epochs.

Value

The cleaned array, with attributes n_total and n_kept.

Examples

```
x <- matrix(rnorm(128 * 10 * 3), ncol = 3)
x[300, 1] <- 50 # one artefact
ep <- mt_clean_epochs(mt_epoch(x, sfreq = 128), detrend = TRUE, max_ptp = 20)
c(kept = attr(ep, "n_kept"), total = attr(ep, "n_total"))
```

mt_connectivity	<i>wPLI and MSC from the same cross-spectrum (one subject)</i>
-----------------	--

Description

Each epoch is Hann-tapered and Fourier transformed. The cross-spectrum $S_{ij}(f) = X_i(f) \text{Conj}(X_j(f))$ is averaged over the bins of a band within each epoch; epochs are the observations: $wPLI_{ij} = |\text{lmean}(\text{Im } S_{ij})| / \text{mean}(|\text{Im } S_{ij}|)$ and $MSC_{ij} = |\text{lmean}(S_{ij})|^2 / (\text{mean}(S_{ii}) \text{mean}(S_{jj}))$. Both metrics therefore come from identical cross-spectral estimates.

Usage

```
mt_connectivity(epochs, sfreq, bands = mt_bands())
```

Arguments

epochs	Array, epochs x channels x samples.
sfreq	Sampling frequency (Hz).
bands	Named list of bands, see mt_bands .

Value

Named list (one element per band) with numeric vectors `wpli` and `msc` over channel pairs in combination order.

Examples

```
ep <- mt_epoch(matrix(rnorm(128 * 20 * 3), ncol = 3), sfreq = 128)
con <- mt_connectivity(ep, sfreq = 128)
str(con$alpha)
```

mt_dc	<i>Direction consistency (DC)</i>
-------	-----------------------------------

Description

Percentage of features for which the mean of group 1 exceeds the mean of group 2. DC removes the native unit of a connectivity metric, so that different metrics can be compared on one scale ("unit erasure").

Usage

```
mt_dc(g1, g2)
```

Arguments

g1, g2	Numeric matrices (subjects x features) for the two groups.
--------	--

Value

A single number between 0 and 100.

Examples

```
g1 <- matrix(rnorm(5 * 10, mean = 0.5), nrow = 5) # 5 subjects, 10 features
g2 <- matrix(rnorm(6 * 10), nrow = 6)             # 6 subjects
mt_dc(g1, g2)
```

mt_epoch	<i>Cut a continuous recording into fixed-length epochs</i>
----------	--

Description

Cut a continuous recording into fixed-length epochs

Usage

```
mt_epoch(x, sfreq, epoch_len = 2, overlap = 0.5)
```

Arguments

x	Numeric matrix, samples in rows and channels in columns.
sfreq	Sampling frequency (Hz).
epoch_len	Epoch length in seconds (default 2).
overlap	Fractional overlap between successive epochs (default 0.5).

Value

Array of dimension epochs x channels x samples.

Examples

```
x <- matrix(rnorm(128 * 10 * 3), ncol = 3) # 10 s of 3 channels at 128 Hz
ep <- mt_epoch(x, sfreq = 128, epoch_len = 2, overlap = 0.5)
dim(ep) # epochs x channels x samples
```

mt_features_from_eeg_csv	<i>Features from one CSV file of preprocessed EEG per subject</i>
--------------------------	---

Description

Each file holds one recording: samples in rows, channels in columns, with a header row of channel names. Preprocessing (filtering, artifact removal, re-referencing) is expected to have been done upstream.

Usage

```
mt_features_from_eeg_csv(
  files,
  ids = NULL,
  sfreq,
  epoch_len = 2,
  overlap = 0.5,
  bands = mt_bands(),
  drop_cols = c("time", "Time", "t"),
  progress = NULL
)
```

Arguments

files	Character vector of CSV paths.
ids	Subject identifiers (default: file names without extension).
sfreq	Sampling frequency (Hz).
epoch_len, overlap	Passed to mt_epoch .
bands	Named list of bands.
drop_cols	Column names to ignore (e.g. a time column).
progress	Optional function called with the subject index.

Value

Feature list as in [mt_features_from_epochs](#).

Examples

```
files <- file.path(tempdir(), c("s1.csv", "s2.csv"))
n <- 128 * 20 # 20 s at 128 Hz
for (f in files) {
  d <- data.frame(time = seq_len(n) / 128, C3 = rnorm(n), C4 = rnorm(n), Pz = rnorm(n))
  write.csv(d, f, row.names = FALSE)
}
feats <- mt_features_from_eeg_csv(files, sfreq = 128)
dim(feats$alpha$wpli)
unlink(files)
```

mt_features_from_epochs

Subject-by-feature matrices from per-subject epoch arrays

Description

Subject-by-feature matrices from per-subject epoch arrays

Usage

```
mt_features_from_epochs(
  epochs_list,
  sfreq,
  bands = mt_bands(),
  progress = NULL
)
```

Arguments

epochs_list	List of arrays (epochs x channels x samples), one per subject.
sfreq	Sampling frequency (Hz); a single value or one per subject.
bands	Named list of bands.
progress	Optional function called with the subject index.

Value

Named list by band of list(wpli, msc) matrices (subjects x channel pairs).

Examples

```
ep <- lapply(1:4, function(i) mt_epoch(matrix(rnorm(128 * 20 * 3), ncol = 3), sfreq = 128))
names(ep) <- paste0("s", 1:4)
feats <- mt_features_from_epochs(ep, sfreq = 128)
dim(feats$alpha$wpli)                # subjects x channel pairs
```

mt_labelings

Metric-T under alternative labelling rules

Description

Applies Metric-T to the same features under several ways of assigning subjects to two classes, so that the dependence of the conclusion on the labelling rule can be read from one table.

Usage

```
mt_labelings(
  features,
  labelings,
  good = "good",
  poor = "poor",
  n_perm = 2000L,
  seed = 42L
)
```

Arguments

features	As for <code>metric_t</code> .
labelings	Data frame or named list; each element gives one label per subject (NA excludes the subject under that rule).
good, poor	The two labels; DC is the percentage of features with good > poor.
n_perm	Permutations per rule (0 skips the test; p-values are then NA).
seed	Random seed.

Value

Data frame with one row per rule and band.

Examples

```
sim <- mt_simulate_motion(rep(c(1, -1), each = 6), rep(c(1, -1), times = 6),
  duration = 30, seed = 1)
fast <- sim$covariates$cadence_hz > median(sim$covariates$cadence_hz)
steady <- sim$covariates$period_cv < median(sim$covariates$period_cv)
rules <- list(by_speed = ifelse(fast, "good", "poor"),
  by_accuracy = ifelse(steady, "good", "poor"))
mt_labelings(sim$features, rules, n_perm = 200)
```

mt_labelset

Metric-T over the complete set of admissible labellings

Description

Subjects whose class is clear keep their label; each of the m ambiguous subjects is assigned to either class in every possible way, giving 2^m labellings. Metric-T is evaluated on each one. The range of T and the proportion of labellings with a reversal describe how far the conclusion depends on the treatment of ambiguous cases.

Usage

```
mt_labelset(
  features,
  labels,
  ambiguous = NULL,
  good = "good",
  poor = "poor",
  prob = 0.5,
  max_exact = 65536L,
  seed = 42L
)
```

Arguments

features	As for <code>metric_t</code> .
labels	One label per subject; ambiguous subjects may be NA or any value other than good and poor.
ambiguous	Logical or index vector of ambiguous subjects (default: every subject whose label is neither good nor poor).
good, poor	The two class labels.
prob	Probability that each ambiguous subject belongs to good (length 1 or m), e.g. from a sigmoid membership function. Used as weights for the expected T; 0.5 weights all labellings equally.
max_exact	Largest number of labellings enumerated exactly; above this, max_exact labellings are drawn at random using prob.
seed	Random seed for the sampled case.

Value

Object of class `mt_labelset`: summary (one row per band), T (labellings by bands), reversal, n_good, influence (mean change in T when a subject is moved from poor to good) and exact.

Examples

```
sim <- mt_simulate_motion(rep(c(1, -1), each = 6), rep(c(1, -1), times = 6),
  duration = 30, seed = 1)
fast <- sim$covariates$cadence_hz > median(sim$covariates$cadence_hz)
steady <- sim$covariates$period_cv < median(sim$covariates$period_cv)
lab <- ifelse(fast & steady, "good", ifelse(!fast & !steady, "poor", NA))
table(lab, useNA = "ifany")
mt_labelset(sim$features, lab)
```

mt_label_space

Enumerate the complete two-group labelling space

Description

Computes Metric-T for every way of splitting the n subjects into two non-empty groups ($2^n - 2$ labellings). The result is the finite set on which exact permutation tests (labellings of one size), attainable significance levels, and alternative group definitions (cut-points, exclusions, batches) can all be located.

Usage

```
mt_label_space(features, max_n = 16L)
```

Arguments

features	Named list by band; each element is a list with matrices wpli and msc (subjects x features), as for metric_t .
max_n	Largest number of subjects for which the space is enumerated.

Details

The spread of direction consistency (DC) over the labelling space measures how many features carry independent information: with m independent features the standard deviation of DC is $50/\sqrt{m}$ percentage points, so $m_{eff} = 2500/Var(DC)$. Features whose group means tie (for example features that are constant across subjects) never count towards DC and inflate this number.

Value

An object of class "mt_label_space": a list with

labellings	Logical matrix, one row per labelling; TRUE marks group 1.
n1	Size of group 1 in each labelling.
T, DC_wPLI, DC_MSC	Matrices (labellings x bands), in percentage points.
summary	Data frame per band: SD of T, effective number of features for each metric, correlation between the two DC values, and the share of labellings in which the two DC values fall on opposite sides of 50 percent.

See Also

[mt_locate](#), [metric_t](#)

Examples

```
sim <- mt_simulate(n = c(A = 5, B = 5), n_ch = 4, n_epochs = 8, seed = 1)
sp <- mt_label_space(sim$features)
sp
mt_locate(sp, sim$group == "A")
```

mt_locate

Locate a labelling in the labelling space

Description

Reports Metric-T for one labelling and the share of labellings whose $|T|$ is at least as large, both in the complete space and among the labellings in which group 1 has the same size. The latter is the exact two-sided permutation p-value of `metric_t(exact = TRUE)`; its smallest attainable value is also returned.

Usage

```
mt_locate(space, group1)
```

Arguments

space	Result of <code>mt_label_space</code> .
group1	Logical vector (one element per subject) marking group 1, or the integer positions of the subjects in group 1.

Value

Data frame with one row per band: T, the two DC values, share_all (complete space), p_exact (same group sizes), p_min (smallest attainable p_exact: the share of those labellings with $|T| = 100$ pp, as in `metric_t`) and the number of labellings with the same group sizes.

Examples

```
sim <- mt_simulate(n = c(A = 5, B = 5), n_ch = 4, n_epochs = 8, seed = 1)
sp <- mt_label_space(sim$features)
mt_locate(sp, sim$group == "A") # the observed labelling
mt_locate(sp, 1:3)             # any other split, e.g. a batch or a cut-point
```

mt_motion_bands	<i>Frequency bands for paced repetitive movement (Hz)</i>
-----------------	---

Description

movement covers the repetition frequency of a paced squat (roughly 0.4 to 0.9 Hz); harmonic covers its second and third harmonics.

Usage

```
mt_motion_bands()
```

Value

Named list of length-2 numeric vectors.

Examples

```
mt_motion_bands()
```

mt_null_plot	<i>Permutation null distribution of T for one condition</i>
--------------	---

Description

Permutation null distribution of T for one condition

Usage

```
mt_null_plot(x, condition = 1L)
```

Arguments

x	A metricT object computed with keep_null = TRUE.
condition	Row number of x\$table.

Value

The permutation values of T for the chosen condition, invisibly. Called for its side effect of drawing a histogram.

Examples

```
sim <- mt_simulate(n = c(A = 6, B = 6), n_ch = 4, n_epochs = 8, seed = 1)
res <- metric_t(sim$features, sim$group, exact = TRUE)
mt_null_plot(res, condition = 1)
```

mt_read_npy	<i>Read a NumPy .npy file (base R, no Python needed)</i>
-------------	--

Description

Supports little-endian float64/float32/int32/int64/bool and fixed-width unicode arrays, which covers the files written by the Python reference pipeline. Pickled object arrays are skipped with a warning.

Usage

```
mt_read_npy(path)
```

Arguments

path	Path to a .npy file.
------	----------------------

Value

An R array (or vector) with the NumPy shape, or NULL if unsupported.

Examples

```
f <- system.file("extdata", "tiny_epochs.npy", package = "metricT")
dim(mt_read_npy(f))
```

mt_read_npz

Read the feature archive written by the Python reference pipeline

Description

The published pipeline saves `features.npz` with arrays named `<band>__wpli` and `<band>__msc` (subjects x channel pairs) and `subj_group`. This reads the archive directly in R.

Usage

```
mt_read_npz(path)
```

Arguments

`path` Path to a .npz file.

Value

List with features and group, ready for `metric_t`.

Examples

```
f <- system.file("extdata", "B_features.npz", package = "metricT")
dat <- mt_read_npz(f)
names(dat)
```

mt_simulate

Simulate EEG with tunable coupling-mode composition

Description

Each channel mixes a phase-lagged oscillation (seen by wPLI) and a common zero-lag oscillation (seen by MSC but not by wPLI) plus white noise. With the defaults, group A has stronger but mostly zero-lag coupling and group B weaker but mostly phase-lagged coupling, so MSC ranks A above B while wPLI ranks B above A in the band containing f_0 : the situation Metric-T is designed to flag. Intended for demonstration and software testing.

Usage

```
mt_simulate(
  n = c(A = 20, B = 20),
  zero_lag = c(0.58, 0.42),
  strength = c(1, 0.9),
  subject_sd = 0.05,
  gain_sd = 0.7,
  n_ch = 8L,
  n_epochs = 30L,
  sfreq = 256,
  epoch_len = 2,
  f0 = 10,
  noise = 0.5,
  bands = mt_bands(),
  seed = 1L,
  keep_epochs = FALSE
)
```

Arguments

<code>n</code>	Named integer vector of group sizes.
<code>zero_lag</code>	Zero-lag fraction per group (same order as <code>n</code>).
<code>strength</code>	Amplitude of the oscillatory (coupled) part per group.
<code>subject_sd</code>	Between-subject SD of the zero-lag fraction and (relative) strength.
<code>gain_sd</code>	SD (log scale) of subject-by-channel gains of the coupled part; values above 0 make channel pairs vary less in unison across subjects.
<code>n_ch, n_epochs, sfreq, epoch_len</code>	Recording layout.
<code>f0</code>	Oscillation frequency (Hz).
<code>noise</code>	Noise standard deviation.
<code>bands</code>	Named list of bands.
<code>seed</code>	Random seed.
<code>keep_epochs</code>	Also return the simulated epochs.

Value

List with features, group and optionally epochs.

Examples

```
sim <- mt_simulate(n = c(A = 6, B = 6), n_ch = 4, n_epochs = 8, seed = 1)
names(sim$features)
table(sim$group)
```

mt_simulate_motion	<i>Simulate pose-estimated repetitive movement</i>
--------------------	--

Description

Generates joint-angle time series for a paced repetitive task such as the squat, one recording per subject, from two latent qualities: speed and accuracy (both standardised; 0 is average). The generative assumptions are (i) faster performers move their joints more simultaneously (smaller inter-joint phase lags) and with less joint-specific noise, and (ii) more accurate performers repeat the same period and the same inter-joint timing. Under these assumptions MSC rises mainly with speed, while wPLI rises with accuracy and falls with speed. The assumptions are for studying the behaviour of Metric-T, not established facts about squatting.

Usage

```
mt_simulate_motion(
    z_speed,
    z_accuracy,
    fs = 30,
    duration = 60,
    f0 = 0.63,
    n_joints = 6L,
    lag_unit = 0.35,
    lag_jitter = 0.5,
    period_cv = 0.08,
    noise = 0.8,
    epoch_len = 8,
    overlap = 0.5,
    bands = mt_motion_bands(),
    seed = 1L,
    keep_signals = FALSE
)
```

Arguments

z_speed, z_accuracy	Latent speed and accuracy of each subject.
fs	Frame rate (Hz).
duration	Task duration (s).
f0	Repetition frequency of an average performer (Hz).
n_joints	Number of joint-angle channels (2 to 6).
lag_unit	Inter-joint phase lag step (rad) of an average performer.
lag_jitter	SD (rad) of the slowly varying lag deviation of an average performer.
period_cv	Repetition-period variability of an average performer.
noise	Joint-specific noise SD of an average performer.

epoch_len, overlap	Windowing for the cross-spectrum.
bands	Named list of bands.
seed	Random seed (NULL leaves the generator state untouched).
keep_signals	Also return the simulated recordings.

Value

List with features and covariates: observed cadence (Hz) and repetition-period CV per subject, as video analysis would give.

Examples

```
z_speed <- rep(c(1, -1), each = 4)
z_accuracy <- rep(c(1, -1), times = 4)
sim <- mt_simulate_motion(z_speed, z_accuracy, duration = 30, seed = 1)
sim$covariates
dim(sim$features$movement$wpli)
```

mt_write_features_csv *Write and read features as a plain CSV table*

Description

Wide layout: one row per subject; columns subject, group, then one column per feature named <band>__<metric>__<pair>. Any software that can compute wPLI and MSC can produce this file.

Usage

```
mt_write_features_csv(features, group, path, subject = NULL)

mt_read_features_csv(path)
```

Arguments

features, group	As for metric_t .
path	CSV path.
subject	Optional subject identifiers.

Value

mt_read_features_csv: list with features, group, subject.

Examples

```
sim <- mt_simulate(n = c(A = 4, B = 4), n_ch = 4, n_epochs = 8, seed = 1)
f <- tempfile(fileext = ".csv")
mt_write_features_csv(sim$features, sim$group, f)
back <- mt_read_features_csv(f)
table(back$group)
unlink(f)
```

mt_wy_maxt	<i>Westfall-Young max-T step-down adjustment</i>
------------	--

Description

Westfall-Young max-T step-down adjustment

Usage

```
mt_wy_maxt(T_obs, T_perm, exact = FALSE)
```

Arguments

T_obs	Observed statistics for the m conditions of one family.
T_perm	m x B matrix of statistics under B aligned permutations (column b is one relabelling applied to every condition).
exact	TRUE when the columns of T_perm are the complete set of label assignments (the observed one included).

Value

Adjusted p-values, in the order of T_obs.

Examples

```
T_obs <- c(40, -10, 20)
T_perm <- matrix(sample(seq(-100, 100, by = 10), 3 * 500, replace = TRUE), nrow = 3)
mt_wy_maxt(T_obs, T_perm)
```

plot.metricT	<i>Plot direction consistency for both metrics</i>
--------------	--

Description

One panel per comparison; bars show DC(wPLI) and DC(MSC) by band, the dashed line marks 50

Usage

```
## S3 method for class 'metricT'
plot(x, col = c("#3b6fb0", "#cf5b4e"), ...)
```

Arguments

x	A metricT object.
col	Two colours for wPLI and MSC.
...	Unused.

Value

The object x, invisibly. Called for its side effect of drawing a plot.

Examples

```
sim <- mt_simulate(n = c(A = 6, B = 6), n_ch = 4, n_epochs = 8, seed = 1)
res <- metric_t(sim$features, sim$group, exact = TRUE)
plot(res)
```

run_metricT_app	<i>Launch the Metric-T graphical interface</i>
-----------------	--

Description

Launch the Metric-T graphical interface

Usage

```
run_metricT_app(...)
```

Arguments

...	Passed to shiny::runApp.
-----	--------------------------

Value

No return value, called for side effects (starts the 'shiny' application).

Examples

```
if (interactive()) {  
  run_metricT_app()  
}
```

Index

`metric_t`, [2](#), [4](#), [5](#), [11–14](#), [16](#), [19](#)
`metric_t_trend`, [4](#)
`mt_bands`, [5](#), [7](#)
`mt_clean_epochs`, [6](#)
`mt_connectivity`, [6](#)
`mt_dc`, [7](#)
`mt_epoch`, [6](#), [8](#), [9](#)
`mt_features_from_eeg_csv`, [8](#)
`mt_features_from_epochs`, [9](#), [9](#)
`mt_label_space`, [5](#), [12](#), [14](#)
`mt_labelings`, [10](#)
`mt_labelset`, [11](#)
`mt_locate`, [13](#), [13](#)
`mt_motion_bands`, [14](#)
`mt_null_plot`, [15](#)
`mt_read_features_csv`
 (`mt_write_features_csv`), [19](#)
`mt_read_npy`, [15](#)
`mt_read_npz`, [16](#)
`mt_simulate`, [16](#)
`mt_simulate_motion`, [18](#)
`mt_write_features_csv`, [19](#)
`mt_wy_maxt`, [20](#)

`plot.metricT`, [21](#)

`run_metricT_app`, [21](#)