Package: zeitR (via r-universe)

July 1, 2026

Title Actigraphy Data Parsing and Analysis for R

Version 0.1.0

Description Provides tools for importing, parsing, and analysing raw actigraphy data from wrist-worn devices. Implements a full sleep analysis pipeline — off-wrist detection, main sleep period detection, nap detection, and WASO computation — validated epoch-for-epoch against the Condor circadiaBase Python reference pipeline using ActTrust hardware. Also computes standard non-parametric circadian rhythm variables (interdaily stability, intradaily variability, relative amplitude, L5, M10). Device-specific parameter presets can be swapped to adapt the pipeline to other actigraph models. Designed to complement slumbR in the Circadia Lab ecosystem.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

Depends R (>= 4.1.0)

Imports cli (>= 3.6.0), lubridate (>= 1.9.0), tibble (>= 3.0.0), tidyr (>= 1.3.0), tools, zoo (>= 1.8.0)

Suggests dplyr (>= 1.1.0), forcats (>= 1.0.0), ggplot2 (>= 3.4.0), mclust, RcppRoll, rlang (>= 1.1.0), knitr, pkgdown, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://zeitr.circadia-lab.uk, https://github.com/circadia-bio/zeitR

BugReports https://github.com/circadia-bio/zeitR/issues

Config/pak/sysreqs libicu-dev

Repository https://circadia-bio.r-universe.dev

Date/Publication 2026-06-29 05:18:54 UTC

RemoteUrl https://github.com/circadia-bio/zeitR

RemoteRef main

RemoteSha 3a0d920bcd81d5962ffebc93abd2d568e823fb97

Contents

`acttrust_params` *ActTrust device parameter preset*

Description

Returns a named list of all device- and study-specific parameter values used by the zeitR pipeline when processing ActTrust recordings. Passing this list (or a modified copy) to `run_pipeline()` via the `params` argument fully controls which values each detector stage receives, without having to touch the individual function calls.

Usage

```
acttrust_params()
```

Details

The list is organised into four named sections that map directly to the four pipeline stages:

`offwrist` Parameters for `detect_offwrist_bimodal()`. These values were validated against the Condor circadiaBase pipeline using ActTrust hardware. The refinement stage (`.bimodal_refine_acttrust`) is device-specific and currently only validated for ActTrust.

`sleep` Parameters for `detect_sleep_crespo()`. `sleep_quantile` (0.365) is the ActTrust CSPD wrapper value; the original Crespo (2012) algorithm uses 1/3.

`nap` Parameters for `detect_naps_crespo()`, passed as the `params` argument. Equivalent to `.cspd_nap_params()`.

`waso` Parameters for `compute_waso()`.

To adapt the pipeline for a different device, copy the list, modify the relevant values, and pass it to `run_pipeline()`:

```
p <- acttrust_params()
p$sleep$sleep_quantile <- 1/3   # use the original Crespo threshold
result <- run_pipeline("recording.txt", params = p)
```

Value

A named list with elements `offwrist`, `sleep`, `nap`, and `waso`.

See Also

`run_pipeline()` for the pipeline entry point.

`check_consistency` *Check actigraphy timestamps for consistency issues*

Description

Scans a recording tibble for three classes of timestamp problem:

Usage

```
check_consistency(x, gap_s = 120, datetime_col = "datetime")
```

Arguments

`x`	A tibble as returned by `read_acttrust()` or `prepare_actigraphy()`, containing a `datetime` column.
`gap_s`	`numeric(1)`. Gap threshold in seconds. Intervals longer than this are flagged. Default is `120` (2 minutes).
`datetime_col`	`character(1)`. Name of the datetime column. Default is `"datetime"`.

Details

- **Gaps** — intervals between consecutive epochs longer than `gap_s` seconds.
- **Backward jumps** — timestamps that go backwards in time.
- **Year artefacts** — timestamps in the years 1970 or 2000, which typically indicate firmware epoch-counter rollover bugs.

Value

A tibble with one row per detected issue and columns:

`row integer` — row index in `x` where the issue occurs.

`datetime POSIXct` — timestamp at that row.

`issue character` — one of `"gap"`, `"backward_jump"`, or `"year_artefact"`.

`detail character` — human-readable description.

Returns a zero-row tibble if no issues are found.

Examples

```
## Not run:
rec    <- read_acttrust("recordings/P001.txt")
issues <- check_consistency(rec)
issues

## End(Not run)
```

`compute_npcra` *Non-parametric circadian rhythm analysis (NPCRA)*

Description

Computes the standard non-parametric circadian rhythm analysis variables from an actigraphy recording, following Gonçalves et al. (2014) and Van Someren et al. (1999). All variables are derived from the 24-hour average activity profile built from **hourly means** (p = 24).

Usage

```
compute_npcra(
  x,
  epoch_s = NULL,
  L5_hours = 5,
  M10_hours = 10,
  window_days = NULL
)
```

Arguments

`x`	A `zeitr_recording` as returned by `read_actigraphy()`, or a data frame / tibble with at least `datetime` and `activity` columns. If a `state` column is present, off-wrist epochs (`state == 4`) are excluded before computing all NPCRA variables.
`epoch_s`	`numeric(1)`. Epoch duration in seconds. If `NULL` (default), estimated automatically from the median inter-epoch interval.

L5_hours	numeric(1). Width of the least-active window in hours. Default is 5.
M10_hours	numeric(1). Width of the most-active window in hours. Default is 10.
window_days	numeric(1) or NULL. If supplied, the recording is split into non-overlapping windows of this length (in days) and NPCRA variables are computed for each window. A window_start column is added to the output. Partial final windows (shorter than window_days) are included but flagged via a lower n_days value. Default NULL computes a single estimate over the full recording.

Details

The following variables are computed:

- IS **Interdaily stability** — consistency of the 24 h rest-activity pattern across days (range 0–1; higher = more stable).
- IV **Intradaily variability** — fragmentation of the rest-activity rhythm (>= 0; higher = more fragmented).
- RA **Relative amplitude** — contrast between the most active 10 h window (M10) and least active 5 h window (L5) (range 0–1).
- L5 Mean activity during the least active 5 consecutive hours.
- L5_onset Clock time of the L5 window onset (hh:mm).
- M10 Mean activity during the most active 10 consecutive hours.
- M10_onset Clock time of the M10 window onset (hh:mm).

Value

A tibble with columns participant_id, window_start (if window_days is set), IS, IV, RA, L5, L5_onset, M10, M10_onset, n_days, n_epochs.

References

Gonçalves, B. S. B., Adamowicz, T., Louzada, F. M., Moreno, C. R., & Araujo, J. F. (2014). A fresh look at the use of nonparametric analysis in actimetry. *Sleep Medicine Reviews*, 20, 84–91. doi:10.1016/j.smrv.2014.06.002

Van Someren, E. J. W., Swaab, D. F., Colenda, C. C., Cohen, W., McCall, W. V., & Rosenquist, P. B. (1999). Bright light therapy: Improved sensitivity to its effects on rest-activity rhythms in Alzheimer patients by application of nonparametric methods. *Chronobiology International*, 16(4), 505–518. doi:10.3109/07420529908998724

Examples

```
## Not run:
rec   <- read_actigraphy("recordings/P001.txt")

# Single estimate over the full recording
compute_npcra(rec)

# Per-fortnight estimates
```

```
compute_npcra(rec, window_days = 14)

## End(Not run)
```

`compute_waso` *Compute WASO and nightly sleep statistics*

Description

Scores each epoch within detected sleep periods as wake or sleep using `score_epochs_cole_kripke()`, then computes per-night statistics. This is a faithful port of the Python `condor_pipeline` `detect_waso`: night boundaries are built by `.nights_df()` (the `search_gap = FALSE` path of the reference `nights_df`), each night's ZCM is scored with Cole-Kripke, and the epoch-level `state` is rebuilt from a zero (wake) base so that within-night WASO-wake epochs and all epochs outside detected nights are scored as wake.

Usage

```
compute_waso(x, wake_thresh = 60L, search_gap = FALSE)
```

Arguments

`x`	A tibble as returned by `detect_naps_crespo()` (or `detect_sleep_crespo()` if nap detection is skipped), containing columns `datetime`, `ZCMn`, and `state`.
`wake_thresh`	`integer(1)`. Minimum duration in epochs of a wake bout required to delimit a new sleep period boundary. Default is `60`.
`search_gap`	`logical(1)`. Reserved for API compatibility with the reference pipeline. The reference `detect_waso` always builds boundaries with `search_gap = FALSE`, so this argument currently has no effect.

Details

The following metrics are computed for each detected night / nap:

Metric	Definition
`tbt`	Total Bed Time — epochs from bed time to get-up time
`tst`	Total Sleep Time — `tbt - waso - sol - soi`
`waso`	Wake After Sleep Onset — wake epochs between sleep onset and final wake
`sol`	Sleep Onset Latency — epochs from bed time to first sleep epoch
`soi`	Sleep Offset Inertia — trailing wake epochs at end of sleep period
`nw`	Number of awakenings — count of wake-onset transitions
`eff`	Sleep efficiency — `tst / tbt`

Value

A list with two elements:

`nights` A tibble with one row per detected night/nap and columns `night`, `is_nap`, `bed_time`, `get_up_time`, `tbt`, `tst`, `waso`, `sol`, `soi`, `nw`, `eff`.

`data` The input tibble `x` with `state` and `sleep` updated with epoch-level Cole-Kripke wake/sleep scores.

References

Cole, R. J., Kripke, D. F., Gruen, W., Mullaney, D. J., & Gillin, J. C. (1992). Automatic sleep/wake identification from wrist activity. *Sleep*, 15(5), 461–469. doi:10.1093/sleep/15.5.461

Examples

```
## Not run:
rec    <- read_acttrust("recordings/P001.txt")
prep   <- prepare_actigraphy(rec)
prep   <- detect_offwrist_bimodal(prep)
prep   <- detect_sleep_crespo(prep)
prep   <- detect_naps_crespo(prep)
result <- compute_waso(prep)
result$nights

## End(Not run)
```

`detect_naps_crespo` *Detect secondary sleep periods (naps) using the Crespo nap algorithm*

Description

Faithful port of the Python `nap_wrapper`: runs the full CSPD model in nap mode (`detect_naps = TRUE`) on the currently-awake epochs (`state == 0`) and merges the detected naps into the sleep state. Must be run *after* `detect_sleep_crespo()`.

Usage

```
detect_naps_crespo(x, epoch_h = NULL, params = .cspd_nap_params())
```

Arguments

`x`	A tibble as returned by `detect_sleep_crespo()`, containing columns `datetime`, `activity`, and `state`. Nap detection runs on the wake (`state == 0`) subset only, mirroring the Python `nap_bool` mask.
`epoch_h`	`numeric(1)`. Number of epochs per hour. If `NULL` (default), derived from the wake-subsequence epoch duration as `3600 / duration`.
`params`	CSPD nap configuration list (default `.cspd_nap_params()`), the port of `nap_wrapper`'s parameter set.

Details

Nap detection uses a nap-mode MSP (a high zero-proportion combined with a low adaptive-median activity, `.crespo_nap_msp()`) followed by the same bed-time / get-up-time refiners as the main sleep detection, with the nap parameter set (`.cspd_nap_params()`) and nap-specific minimum-length post-processing. Detected naps are written as `state == 1` (merged into "sleep"), matching `nap_wrapper`, which assigns `state[wake] = 1 - refined_output` (i.e. naps are *not* a distinct state).

Value

The input tibble `x` with `state` and `sleep` columns updated. Nap epochs become `state == 1` and `sleep == 1`; off-wrist (`state == 4`) and existing main-sleep epochs are preserved.

References

Crespo, C., Aboy, M., Fernández, J. R., & Mojón, A. (2012). Automatic identification of activity-rest periods based on actigraphy. *Journal of Medical and Biological Engineering*, 32(4), 249–256. doi:10.5405/jmbe.1033

See Also

`detect_sleep_crespo()` for main sleep period detection.

Examples

```
## Not run:
rec  <- read_acttrust("recordings/P001.txt")
prep <- prepare_actigraphy(rec)
prep <- detect_offwrist_bimodal(prep)
prep <- detect_sleep_crespo(prep)
prep <- detect_naps_crespo(prep)

## End(Not run)
```

---

detect_offwrist_bimodal

Off-wrist detection using the Condor bimodal activity/temperature model

---

Description

Detects periods where the actigraph was not worn using the bimodal algorithm developed by Condor Instruments. The algorithm proceeds in three stages:

Usage

```
detect_offwrist_bimodal(
  x,
  hws = 10L,
  activity_quantile = 0.15,
  min_norm_activity = 0.015,
  nbins = 100L,
  min_offwrist_length = 10L,
  min_temp_threshold = 0.35
)
```

Arguments

`x`	A tibble as returned by `prepare_actigraphy()`, containing columns `datetime`, `activity` (PIM), `int_temp`, `ext_temp`, `state`, and `offwrist`.
`hws`	`integer(1)`. Half-window size (in epochs) for rolling feature extraction. Default is `10` (matching the Python pipeline).
`activity_quantile`	`numeric(1)`. Quantile used to define "low activity". Default is `0.15`.
`min_norm_activity`	`numeric(1)`. Minimum normalised activity threshold below which the low-activity cutoff is clamped. Default is `0.015`.
`nbins`	`integer(1)`. Number of histogram bins used when fitting the GMM threshold. Default is `100`.
`min_offwrist_length`	`integer(1)`. Minimum number of consecutive off-wrist epochs required to retain a detected period. Shorter runs are discarded. Default is `10`.
`min_temp_threshold`	`numeric(1)`. Minimum normalised temperature threshold; the fitted GMM threshold is clamped to this value if it falls below it. Default is `0.35`.

Details

1. **Feature extraction** — rolling median of PIM activity and rolling signal/variance of internal temperature are computed over a symmetric window of half-width `hws` epochs.
2. **Bimodal temperature threshold** — among epochs with low activity median (below the `activity_quantile` quantile of normalised activity), a 2-component Gaussian Mixture Model is fitted to the normalised temperature distribution. The threshold between the two components is taken as the minimum of the GMM density between the two means (or the minimum of the smoothed histogram if that is lower). Ashman's D is computed as a bimodality quality metric.
3. **Initial classification & refinement** — epochs simultaneously exhibiting low activity median AND low temperature are marked as off-wrist. Short spurious off-wrist runs shorter than `min_offwrist_length` epochs are removed.

Off-wrist epochs are encoded as `state == 4` (matching the Python pipeline convention). The `offwrist` column is set to `0.25` for off-wrist epochs for actogram overlay plotting.

Value

The input tibble `x` with `state` and `offwrist` columns updated. Off-wrist epochs have `state == 4` and `offwrist == 0.25`.

References

The bimodal off-wrist algorithm was developed by Julius A. P. P. de Paula at Condor Instruments (2023). It is not published in peer-reviewed literature but the source code is available in the circadiaBase pipeline repository. The Ashman D statistic is described in:

Ashman, K. M., Bird, C. M., & Zepf, S. E. (1994). Detecting bimodality in astronomical datasets. *The Astronomical Journal*, 108, 2348. doi:10.1086/117248

Examples

```
## Not run:
rec  <- read_acttrust("recordings/P001.txt")
prep <- prepare_actigraphy(rec)
prep <- detect_offwrist_bimodal(prep)
sum(prep$state == 4)  # number of off-wrist epochs

## End(Not run)
```

`detect_sleep_crespo` *Detect main sleep periods using the Crespo algorithm*

Description

Identifies the main sleep period(s) in an actigraphy recording using the algorithm described in Crespo et al. (2012). The method applies an adaptive median filter to the activity signal, mitigates spuriously long zero runs, and thresholds the result at a quantile of the filtered signal. Morphological closing and opening operations are then used to smooth the binary sleep/wake estimate.

Usage

```
detect_sleep_crespo(
  x,
  epoch_h = NULL,
  median_filter_h = 8,
  pad_h = 1,
  sleep_quantile = 0.365,
  morph_size = 61L,
  consec_zeros_thr = 15L,
  awake_zeros_thr = 2L,
  sleep_zeros_thr = 30L,
  zero_mitigation_q = 0.33,
  min_short_window_thr = 1,
  refine = TRUE,
  condition = 0L
)
```

Arguments

x	A tibble as returned by detect_offwrist_bimodal() (or prepare_actigraphy() if off-wrist detection is skipped), containing columns datetime, activity, and state. The detector runs on the on-wrist subset (state != 4) only, mirroring the Python cspd_wrapper.
epoch_h	numeric(1). Number of epochs per hour. If NULL (default), derived from the epoch duration (the mode of the on-wrist inter-epoch interval), as 3600 / duration.
median_filter_h	numeric(1). Length of the preprocessing median filter window in hours. Default is 8.
pad_h	numeric(1). Padding length in hours added before the adaptive median filter. Default is 1.
sleep_quantile	numeric(1). Quantile of the filtered activity used as the MSP sleep/wake threshold. Default is 0.365 (the ActTrust CSPD value used by cspd_wrapper; the standalone Crespo algorithm uses 1/3).
morph_size	integer(1). Size of the structuring element used in morphological closing/opening. Default is 61 epochs.
consec_zeros_thr	integer(1). Runs of zeros longer than this threshold are treated as invalid (zero mitigation). Default is 15.
awake_zeros_thr	integer(1). Threshold for consecutive zeros within wake periods. Default is 2.
sleep_zeros_thr	integer(1). Threshold for consecutive zeros within sleep periods. Default is 30.
zero_mitigation_q	numeric(1). Quantile of activity used to determine the mitigation level for invalid zero runs. Default is 0.33.
min_short_window_thr	numeric(1). Minimum value of the adaptive median threshold; if the fitted quantile falls below this, the threshold is clamped here. Default is 1.0.
refine	logical(1). If TRUE (default), the MSP detection is refined into final sleep periods by the CSPD bed-time / get-up-time refiners (.cspd_refine_periods), reproducing the Python refined_output. If FALSE, the raw MSP detection is used directly.
condition	integer(1). Initial condition flag (default 0). The MSP stage bumps it to 2 when its activity-median threshold clamps to min_short_window_thr; the refiner uses that effective condition. Affects only refine = TRUE.

Value

The input tibble x with state and sleep columns updated. Sleep epochs have state == 1 and sleep == 1; off-wrist epochs (state == 4) are preserved and excluded from the sleep column. With refine = TRUE the sleep epochs delimit the refined sleep PERIODS (the Python refined_output); per-epoch wake/sleep within them is scored later by compute_waso().

References

Crespo, C., Aboy, M., Fernández, J. R., & Mojón, A. (2012). Automatic identification of activity-rest periods based on actigraphy. *Journal of Medical and Biological Engineering*, 32(4), 249–256. doi:10.5405/jmbe.1033

See Also

`detect_naps_crespo()` for secondary sleep period (nap) detection.

Examples

```
## Not run:
rec  <- read_acttrust("recordings/P001.txt")
prep <- prepare_actigraphy(rec)
prep <- detect_offwrist_bimodal(prep)
prep <- detect_sleep_crespo(prep)

## End(Not run)
```

---

label_states *Convert integer epoch states to a labelled factor*

Description

Converts the integer `state` column produced by the zeitR pipeline into a human-readable factor. Useful for display, plotting, and export — the internal `state` column always stays integer to preserve Python reference parity.

Usage

```
label_states(x)
```

Arguments

`x`	integer (or numeric) vector of epoch states, as found in `result$data$state`.

Details

Integer	Label
`0`	`"wake"`
`1`	`"sleep"`
`4`	`"off-wrist"`
`7`	`"nap"`

Any value not in the table above is silently converted to `NA`.

Value

An ordered factor with levels `c("wake", "sleep", "nap", "off-wrist")`, the same length as `x`.

Examples

```
label_states(c(0L, 1L, 0L, 4L, 1L, 7L))
# [1] wake  sleep wake  off-wrist sleep nap
# Levels: wake < sleep < nap < off-wrist

## Not run:
result <- run_pipeline("recordings/P001.txt")
result$data$state_label <- label_states(result$data$state)

## End(Not run)
```

`prepare_actigraphy` *Prepare a raw actigraphy tibble for analysis*

Description

Transforms the output of `read_acttrust()` (or any reader that returns a compatible tibble) into the working data frame expected by all detection functions. Specifically:

Usage

```
prepare_actigraphy(x)
```

Arguments

`x`	A tibble as returned by `read_acttrust()`, containing at minimum `int_temp` and `ext_temp` columns.

Details

1. Clamps `int_temp` and `ext_temp` to the physiological range [0, 42] °C.
2. Adds min-max scaled temperature columns `int_temp_` and `ext_temp_` (range [0, 1]) for plotting.
3. Ensures `state`, `offwrist`, and `sleep` columns are present and initialised to `0`.

The original tibble is never modified; a copy is returned.

Value

A tibble of the same dimensions as `x` with additional or updated columns: `state`, `offwrist`, `sleep`, `int_temp_`, `ext_temp_`.

Examples

```
## Not run:
rec  <- read_acttrust("recordings/P001.txt")
prep <- prepare_actigraphy(rec)

## End(Not run)
```

`read_actigraphy` *Read an actigraphy file into a zeitr_recording object*

Description

A device-agnostic wrapper that reads a raw actigraphy file and returns a `zeitr_recording` object with `$epochs` (a tidy tibble) and `$metadata` (a named list of device and recording information).

Usage

```
read_actigraphy(path, device = "acttrust", tz = "UTC", ...)
```

Arguments

`path`	`character(1)`. Path to the raw actigraphy file.
`device`	`character(1)`. Device type. One of `"acttrust"` (default). Additional devices will be added in future versions.
`tz`	`character(1)`. Recording time zone passed to the underlying reader. Default is `"UTC"`.
`...`	Additional arguments forwarded to the device-specific reader (e.g. `encoding` for `read_acttrust()`).

Details

Currently supported devices:

- `"acttrust"` — Condor Instruments ActTrust / ActTrust2 (`.txt`)

Value

A `zeitr_recording` S3 object — a named list with:

`$epochs` A tibble with one row per epoch and columns `datetime`, `activity`, `int_temp`, `ext_temp`, `ZCMn`, `state`, `offwrist`, `sleep`.

`$metadata` A named list with `subject`, `device_id`, `device_model`, `firmware_version`, `interval_s`, `source_file`, and `participant_id` (derived from the filename stem).

See Also

`read_actigraphy_dir()` to read a whole directory at once.

Examples

```
## Not run:
rec <- read_actigraphy("recordings/P001.txt")
rec$epochs
rec$metadata

## End(Not run)
```

`read_actigraphy_dir` *Read all actigraphy files in a directory*

Description

Applies `read_actigraphy()` to every file matching `pattern` in `folder`. Returns a `zeitr_study` object — a named list of `zeitr_recording` objects, one per file. Files that fail to parse are skipped with a warning.

Usage

```
read_actigraphy_dir(
  folder,
  device = "acttrust",
  pattern = "*.txt",
  tz = "UTC",
  ...
)
```

Arguments

`folder`	`character(1)`. Path to a directory containing actigraphy files.
`device`	`character(1)`. Device type passed to `read_actigraphy()`. Default is `"acttrust"`.
`pattern`	`character(1)`. Glob pattern for file discovery. Default is `"*.txt"`.
`tz`	`character(1)`. Recording time zone. Default is `"UTC"`.
`...`	Additional arguments forwarded to `read_actigraphy()`.

Value

A `zeitr_study` S3 object — a named list of `zeitr_recording` objects. Names are participant IDs (filename stems).

See Also

`study_summary()` to summarise a `zeitr_study`.

Examples

```
## Not run:
study <- read_actigraphy_dir("recordings/", tz = "America/Sao_Paulo")
study_summary(study)

## End(Not run)
```

`read_acttrust` *Read a Condor Instruments ActTrust actigraphy file*

Description

Parses a Condor ActTrust `.txt` export into a tidy tibble. The file format consists of a variable-length key-value header block followed by semicolon-delimited epoch rows. The header ends at the line beginning with `DATE/TIME`.

Usage

```
read_acttrust(path, tz = "UTC", encoding = "latin1")
```

Arguments

`path`	`character(1)` or `fs::path`. Path to the ActTrust `.txt` file.
`tz`	`character(1)`. Time zone string passed to `lubridate::parse_date_time()`. Defaults to `"UTC"`. Set to the local recording time zone for correct circadian alignment.
`encoding`	`character(1)`. File encoding. Defaults to `"latin1"`, which matches Condor's default export encoding.

Value

A tibble with one row per epoch and the following columns:

`datetime POSIXct` — epoch timestamp.

`activity double` — PIM activity count.

`int_temp double` — internal (on-body) temperature, °C.

`ext_temp double` — external (ambient) temperature, °C. `NA` if unavailable.

`ZCMn double` — normalised zero-crossing mode count. `NA` if unavailable.

`state double` — state column, initialised to `0`.

`offwrist double` — off-wrist indicator, initialised to `0`.

`sleep double` — sleep indicator, initialised to `0`.

The tibble carries a `"zeitr_recording"` class and a `metadata` attribute (a named list with `subject`, `device_id`, `device_model`, `firmware_version`, `interval_s`, `source_file`).

Examples

```
## Not run:
rec <- read_acttrust("recordings/P001.txt")
rec
attr(rec, "metadata")

## End(Not run)
```

`run_pipeline` *Run the full actigraphy sleep analysis pipeline*

Description

Orchestrates the complete pipeline for a single ActTrust recording:

Usage

```
run_pipeline(
  path,
  tz = "UTC",
  gap_s = 120,
  params = acttrust_params(),
  offwrist_args = list(),
  sleep_args = list(),
  nap_args = list(),
  quiet = FALSE
)
```

Arguments

`path`	`character(1)`. Path to the ActTrust `.txt` file.
`tz`	`character(1)`. Recording time zone. Passed to `read_acttrust()`. Default is `"UTC"`.
`gap_s`	`numeric(1)`. Gap threshold (seconds) for `check_consistency()`. Default is `120`.
`params`	Device parameter preset, as returned by `acttrust_params()`. When supplied, values from `params` are used as defaults for each detector stage, with any explicit `offwrist_args`, `sleep_args`, or `nap_args` taking precedence. Defaults to `acttrust_params()`.
`offwrist_args`	`list`. Additional arguments passed to `detect_offwrist_bimodal()`. Default is an empty list.
`sleep_args`	`list`. Additional arguments passed to `detect_sleep_crespo()`. Default is an empty list.
`nap_args`	`list`. Additional arguments passed to `detect_naps_crespo()`. Default is an empty list.

quiet — logical(1). If TRUE, suppresses the timestamp-issue warning emitted when check_consistency() finds problems. Useful in batch or testing contexts where the warning is expected. Default is FALSE.

Details

1. **Read** — read_acttrust()
2. **Consistency check** — check_consistency()
3. **Prepare** — prepare_actigraphy()
4. **Off-wrist detection** — detect_offwrist_bimodal()
5. **Main sleep period detection** — detect_sleep_crespo()
6. **Nap detection** — detect_naps_crespo()
7. **WASO + nightly statistics** — compute_waso()

Value

A zeitr_result S3 object — a named list with:

subject_id character — derived from the input filename stem.

source_file character — absolute path to the input file.

data tibble — final epoch-level data frame with all state columns populated.

nights tibble — per-night sleep statistics.

issues tibble — timestamp consistency issues (0 rows if none).

metadata list — device and subject metadata from the file header.

See Also

run_pipeline_batch() for processing a directory of files.

Examples

```
## Not run:
result <- run_pipeline("recordings/P001.txt", tz = "America/Sao_Paulo")
result$nights
result$data

## End(Not run)
```

run_pipeline_batch *Run the pipeline on all files in a directory*

Description

Applies `run_pipeline()` to every file matching `pattern` in `folder`, returning a list of `zeitr_result` objects. Files that fail are skipped with a warning.

Usage

```
run_pipeline_batch(folder, pattern = "*.txt", ...)
```

Arguments

`folder`	`character(1)`. Path to a directory containing ActTrust files.
`pattern`	`character(1)`. Glob pattern for file discovery. Default is `"*.txt"`.
`...`	Additional arguments forwarded to `run_pipeline()`.

Value

A named list of `zeitr_result` objects, one per successfully processed file. Names are the file stem (subject IDs).

Examples

```
## Not run:
results <- run_pipeline_batch("recordings/", tz = "America/Sao_Paulo")
lapply(results, function(r) r$nights)

## End(Not run)
```

score_epochs_cole_kripke *Score actigraphy epochs as wake or sleep using the Cole-Kripke algorithm*

Description

Applies the Cole-Kripke algorithm to a vector of zero-crossing mode (ZCM) activity counts, scoring each epoch as wake (`1`) or sleep (`0`) using a weighted sum of activity in a surrounding window.

Usage

```
score_epochs_cole_kripke(
  zcm,
  P = 0.000464,
  weights_before = c(34.5, 133, 529, 375, 408, 400.5, 1074, 2048.5, 2424.5),
  weights_after = c(1920, 149.5, 257.5, 125, 111.5, 120, 69, 40.5)
)
```

Arguments

`zcm`	`numeric` vector of ZCM activity counts, one value per epoch.
`P`	`numeric(1)`. Scaling factor. Default is `0.000464` (Cole et al., 1992).
`weights_before`	`numeric(9)`. Weights applied to the 9 epochs *before* the current epoch. Defaults to the values from Cole et al. (1992), Table 2.
`weights_after`	`numeric(8)`. Weights applied to the 8 epochs *after* the current epoch. Defaults to the values from Cole et al. (1992), Table 2.

Details

Each epoch's score is computed as:

$$D_i = P \sum_{j=1}^{9} W_j^- \cdot A_{i-j} + P \sum_{j=1}^{8} W_j^+ \cdot A_{i+j}$$

where A_i is the ZCM count at epoch i, W^- and W^+ are the before and after weight vectors from Cole et al. (1992), and $P = 0.000464$. Epochs with $D_i \geq 1$ are scored as wake.

Value

An integer vector the same length as `zcm`, with `1` indicating wake and `0` indicating sleep.

References

Cole, R. J., Kripke, D. F., Gruen, W., Mullaney, D. J., & Gillin, J. C. (1992). Automatic sleep/wake identification from wrist activity. *Sleep*, 15(5), 461–469. doi:10.1093/sleep/15.5.461

Examples

```
## Not run:
rec    <- read_acttrust("recordings/P001.txt")
scores <- score_epochs_cole_kripke(rec$ZCMn)
table(scores)  # 0 = sleep, 1 = wake

## End(Not run)
```

study_summary *Summarise a zeitr_study across participants*

Description

Computes per-participant summary statistics from a zeitr_study object (as returned by read_actigraphy_dir()). For each recording, the function computes NPCRA variables (IS, IV, RA, L5, M10) and basic recording quality metrics.

Usage

```
study_summary(study, epoch_s = NULL, L5_hours = 5, M10_hours = 10)
```

Arguments

study	A zeitr_study object as returned by read_actigraphy_dir(), or a named list of zeitr_recording objects.
epoch_s	numeric(1). Epoch duration in seconds. If NULL (default), estimated separately for each recording.
L5_hours	numeric(1). Width of the L5 window in hours. Default is 5.
M10_hours	numeric(1). Width of the M10 window in hours. Default is 10.

Value

A tibble with one row per participant and columns:

participant_id Participant identifier (filename stem).

n_epochs Total number of epochs in the recording.

n_days Recording duration in days.

start POSIXct — first epoch timestamp.

end POSIXct — last epoch timestamp.

IS Interdaily stability.

IV Intradaily variability.

RA Relative amplitude.

L5 Mean activity in the least active 5 h window.

L5_onset Clock time of L5 midpoint.

M10 Mean activity in the most active 10 h window.

M10_onset Clock time of M10 midpoint.

See Also

compute_npcra() for single-recording NPCRA, read_actigraphy_dir() to create a zeitr_study.

Examples

```
## Not run:
study <- read_actigraphy_dir("recordings/", tz = "America/Sao_Paulo")
study_summary(study)

## End(Not run)
```

Index