

Package: OmicsChrono (via r-universe)

October 7, 2026

Title Calendar-Aware Temporal Metadata for Longitudinal Omics

Version 0.99.0

Description Longitudinal and circadian omics experiments encode sample time as elapsed experiment time, civil timestamps, or light-cycle coordinates. OmicsChrono validates and standardizes temporal sample metadata in SummarizedExperiment and derived Bioconductor containers while preserving assays and feature metadata. It supports elapsed-time annotation, sampling-window alignment, time-zone and daylight-saving diagnostics, Zeitgeber-time annotation, and temporal coverage summaries as reproducible preprocessing for downstream time-course analyses.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.5.0)

Imports clock, S4Vectors, SummarizedExperiment, methods, stats

Suggests BiocStyle, fission, knitr, rmarkdown, SingleCellExperiment, testthat (>= 3.0.0)

VignetteBuilder knitr

biocViews Software, TimeCourse, Preprocessing

URL <https://github.com/mateopedersen/OmicsChrono>

BugReports <https://github.com/mateopedersen/OmicsChrono/issues>

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Config/pak/sysreqs zlib1g-dev

Repository <https://biocstaging.r-universe.dev>

Date/Publication 2026-10-06 08:57:24 UTC

RemoteUrl <https://github.com/BiocStaging/OmicsChrono>

RemoteRef HEAD

RemoteSha 959908a08a6d2854ec2768c2ad8ccad0097e52e2

Contents

addZeitgeberTime	2
alignSampleTimes	3
annotateSampleTime	4
temporalCoverage	5
temporalDesignData	6
validateSampleTime	7
Index	9

addZeitgeberTime	<i>Add Zeitgeber and photoperiod annotations</i>
------------------	--

Description

Compute time since an explicitly specified local lights-on time and classify each sample as light or dark under the supplied controlled photoperiod.

Usage

```
addZeitgeberTime(  
  x,  
  timestamp,  
  lights_on,  
  timezone,  
  photoperiod = 12,  
  ambiguous = "error",  
  overwrite = FALSE  
)
```

Arguments

x	A ‘SummarizedExperiment’ or derived object.
timestamp	Name of a civil-time column in ‘colData(x)’.
lights_on	Local clock time in ‘HH:MM’ or ‘HH:MM:SS’ format.
timezone	One IANA time zone or one zone per sample.
photoperiod	Number of hours per cycle in the light phase; must be between zero and 24.
ambiguous	DST fall-back policy as in [annotateSampleTime()].
overwrite	Replace existing ‘chrono_zeitgeber_hours’, ‘chrono_light_phase’, and ‘chrono_hours_until_phase_change’ columns.

Details

ZT0 is defined by the provided local lights-on clock time. The calculation uses local clock coordinates modulo 24 hours and the supplied light duration; it does not infer biological phase or query external data. Studies with non-24-hour cycles should use a separately specified experimental schedule.

Value

An object of the same class with three Zeitgeber fields in 'colData'.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = matrix(1:2, nrow = 1)),
  colData = S4Vectors::DataFrame(collection =
    c("2024-06-01 07:00:00", "2024-06-01 19:00:00"))
)
out <- addZeitgeberTime(se, "collection", lights_on = "07:00",
  timezone = "Europe/Oslo", photoperiod = 12)
SummarizedExperiment::colData(out)$chrono_zeitgeber_hours
```

alignSampleTimes	<i>Align observed sample times to expected sampling windows</i>
------------------	---

Description

Match samples to expected time points using a deterministic one-to-one nearest-offset assignment. Samples and expected times use the same numeric unit. Ties are resolved by input sample order and then expected-time order.

Usage

```
alignSampleTimes(observed, expected, tolerance, subject = NULL, sample = NULL)
```

Arguments

observed	Numeric vector of observed times.
expected	Numeric vector of expected times.
tolerance	Maximum absolute difference for a match, in the same unit as 'observed' and 'expected'.
subject	Optional subject identifier per observation.
sample	Optional sample identifier per observation.

Value

An 'S4Vectors::DataFrame' with one row per expected slot and one row per unmatched observed sample. 'status' is 'matched', 'missing', or 'unexpected'.

Examples

```
alignSampleTimes(c(0.1, 5.8, 12.3), c(0, 6, 12), tolerance = 0.5,
  subject = c("a", "a", "a"))
```

annotateSampleTime	<i>Annotate a SummarizedExperiment with normalized sample time</i>
--------------------	--

Description

Add normalized time fields to `'colData(x)'`, leaving assay values, feature metadata, sample order, and the input class intact. Character civil timestamps require `'timezone'`; DST-ambiguous times error by default and may be explicitly resolved with `'ambiguous = "earliest"'` or `'latest"'`.

Usage

```
annotateSampleTime(
  x,
  time,
  subject = NULL,
  time_unit = NULL,
  timezone = NULL,
  ambiguous = "error",
  baseline = "first",
  overwrite = FALSE
)
```

Arguments

<code>x</code>	A <code>'SummarizedExperiment'</code> or derived object.
<code>time</code>	Name of a <code>'colData(x)'</code> column containing sample times.
<code>subject</code>	Optional subject column name or one subject identifier per sample. If omitted, all samples are treated as one subject.
<code>time_unit</code>	Unit for bare numeric values or <code>'T'</code> tokens: <code>"minutes"</code> , <code>"hours"</code> , or <code>"days"</code> .
<code>timezone</code>	One IANA time zone or one zone per sample for civil times. Supply <code>'NULL'</code> for elapsed-time data or absolute <code>'POSIXct'</code> instants.
<code>ambiguous</code>	How an ambiguous local civil timestamp is treated for diagnostics. The default <code>"error"</code> requires explicit disambiguation; <code>"earliest"</code> and <code>"latest"</code> select one occurrence and report a warning.
<code>baseline</code>	Baseline rule. The default <code>"first"</code> sets elapsed time zero at the first valid sample for each subject, in the supplied sample order.
<code>overwrite</code>	Replace derived columns with the same names. Defaults to <code>'FALSE'</code> so existing metadata is never silently overwritten.

Details

Elapsed-time inputs are converted to hours. Civil timestamps are parsed as local wall times in the supplied time zone(s), then differenced as absolute instants in UTC. Derived columns are `'chrono_timestamp_utc'`, `'chrono_timestamp_local'`, `'chrono_elapsed_hours'`, `'chrono_study_day'`, `'chrono_day_index'`, `'chrono_clock_hour'`, `'chrono_day_of_week'`, `'chrono_is_dst'`, and `'chrono_sampling_interval_hours'`. Non-monotonic input order is preserved and can be inspected with `[validateSampleTime()]`.

Value

An object of the same class as 'x', with temporal fields appended to 'colData(x)'.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = matrix(1:6, nrow = 2)),
  colData = S4Vectors::DataFrame(subject = c("a", "a", "b"),
    minute = c(0, 15, 0))
)
out <- annotateSampleTime(se, "minute", subject = "subject", time_unit = "minutes")
SummarizedExperiment::colData(out)$chrono_elapsed_hours
```

temporalCoverage	<i>Summarize temporal coverage</i>
------------------	------------------------------------

Description

Report sample counts, temporal ranges, intervals, and interval variability by subject and optional condition from the sample metadata in a 'SummarizedExperiment'.

Usage

```
temporalCoverage(x, time, subject = NULL, condition = NULL)
```

Arguments

x	A 'SummarizedExperiment' or derived object.
time	Name of a numeric 'colData(x)' time column.
subject	Optional subject column name or one subject identifier per sample.
condition	Optional condition column.

Value

An 'S4Vectors::DataFrame' with one row per subject/condition group.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = matrix(1:8, nrow = 2)),
  colData = S4Vectors::DataFrame(subject = rep(c("a", "b"), each = 2),
    hour = c(0, 6, 0, 12))
)
temporalCoverage(se, "hour", subject = "subject")
```

temporalDesignData	<i>Construct explicit temporal design covariates</i>
--------------------	--

Description

Copy selected sample-level time, subject, and condition fields into a ‘DataFrame’, optionally adding sine and cosine terms for a user-specified cycle length. No model is fitted and phase is not inferred.

Usage

```
temporalDesignData(
  x,
  time,
  subject = NULL,
  condition = NULL,
  cycle_length = NULL
)
```

Arguments

x	A ‘SummarizedExperiment’ or derived object.
time	Name of a numeric time column, normally “chrono_elapsed_hours”.
subject	Optional subject column name or one subject identifier per sample.
condition	Optional condition column name.
cycle_length	Optional positive numeric cycle length in the same unit as ‘time’. When supplied, include cyclical sine/cosine covariates.

Value

An ‘S4Vectors::DataFrame’ in original sample order.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = matrix(1:4, nrow = 1)),
  colData = S4Vectors::DataFrame(hour = c(0, 6, 12, 18),
                                   subject = rep("mouse1", 4))
)
temporalDesignData(se, "hour", subject = "subject", cycle_length = 24)
```

validateSampleTime	<i>Validate temporal sample metadata</i>
--------------------	--

Description

Inspect a sample-time column for missing or malformed values, ambiguous civil times, duplicate subject-time combinations, negative elapsed values, and non-monotonic sample order. Civil timestamps must be ISO local times and need an explicit IANA time zone. Numeric elapsed times need ‘time_unit’; tokens such as “15 min”, “day3”, “T4”, and “baseline” are also supported.

Usage

```
validateSampleTime(
  x,
  time,
  subject = NULL,
  time_unit = NULL,
  timezone = NULL,
  ambiguous = "error"
)
```

Arguments

x	A ‘SummarizedExperiment’ or derived object.
time	Name of a ‘colData(x)’ column containing sample times.
subject	Optional subject column name or one subject identifier per sample. If omitted, all samples are treated as one subject.
time_unit	Unit for bare numeric values or ‘T’ tokens: “minutes”, “hours”, or “days”.
timezone	One IANA time zone or one zone per sample for civil times. Supply ‘NULL’ for elapsed-time data or absolute ‘POSIXct’ instants.
ambiguous	How an ambiguous local civil timestamp is treated for diagnostics. The default “error” requires explicit disambiguation; “earliest” and “latest” select one occurrence and report a warning.

Details

This function reports metadata problems without changing ‘x’. Civil times use ‘Numeric and unit-bearing elapsed-time tokens are normalized to hours only for validation. No assay data are read or modified.

Value

An ‘S4Vectors::DataFrame’ with one row per diagnostic and columns ‘sample’, ‘subject’, ‘code’, ‘severity’, ‘message’, and ‘value’. A report with zero rows means no issues were detected.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(  
  assays = list(counts = matrix(1:6, nrow = 2)),  
  colData = S4Vectors::DataFrame(subject = c("a", "a", "b"),  
                                   sample_time = c("0 min", "15 min", "baseline"))  
)  
validateSampleTime(se, "sample_time", subject = "subject")
```

Index

`addZeitgeberTime`, [2](#)
`alignSampleTimes`, [3](#)
`annotateSampleTime`, [4](#)

`temporalCoverage`, [5](#)
`temporalDesignData`, [6](#)

`validateSampleTime`, [7](#)