Package ‘AmbientViewer’

October 13, 2025

Title Filtering and Visualisation for Somnofy Data

Version 0.0.10

Description This package helps importing, filtering and visualising sleep data.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Suggests devtools,
mockery,
testthat,
tibble

Config/testthat/edition 3

Imports bslib,
circular,
cli,
dplyr,
DT,
edfReader,
ggnewscale,
ggplot2,
logging,
lubridate,
markdown,
plotly,
readr,
rlang,
rmarkdown,
scales,
shiny,
shinyjs,
shinyWidgets,
stringr,
svglite,
tidyr

Depends R (>= 4.3)

LazyData true

Contents

`ambient_viewer` *Ambient Viewer app*

Description

This function launches the Ambient Viewer app, a Shiny application for visualizing and analyzing sleep data.

Usage

```
ambient_viewer()
```

`chronotype` *Calculate the Chronotype*

Description

This function calculates the Chronotype metric based on the mid-sleep time If sleep duration on free days is greater than on workdays, it applies a correction as described in Roenneberg et al. (2019).

Usage

```
chronotype(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions data frame
`col_names`	A list to override default column names. This function uses columns: • `time_at_midsleep` • `sleep_period` • `is_workday`

Value

The Chronotype value in hours

See Also

Other sleep metrics: `composite_phase_deviation()`, `interdaily_stability()`, `sleep_regularity_index()`, `social_jet_lag()`

Examples

```
chronotype(example_sessions)
```

---

composite_phase_deviation *Calculate Composite Phase Deviation (CPD)*

Description

This function calculates the Composite Phase Deviation (CPD) metric, used to measure the regularity of the sleep patterns.

Usage

```
composite_phase_deviation(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions data frame
`col_names`	A list to override default column names. This function uses columns: • `time_at_midsleep` • `is_workday` • `night`

Value

The Composite Phase Deviation (CPD) value

See Also

Other sleep metrics: `chronotype()`, `interdaily_stability()`, `sleep_regularity_index()`, `social_jet_lag()`

Examples

```
composite_phase_deviation(example_sessions)
```

`edfs_to_csv` *Convert EDF files in a folder to a single CSV*

Description

This function scans a specified folder (and its subdirectories) for EDF files whose filenames contain "BRP" (case-insensitive), reads their headers to extract start times and durations, and compiles this information into a single CSV file.

Usage

```
edfs_to_csv(
  folder_in = ".",
  filter_pattern = NULL,
  file_out = "edf_summary.csv"
)
```

Arguments

`folder_in`	The input folder containing EDF files. Default is the current directory.
`file_out`	The name of the output CSV file. Default is "edf_summary.csv".

Value

A CSV file summarizing the start times and durations of the EDF files.

`example_epochs` *Example Epoch data*

Description

A data frame containing epoch data recorded by a Somnofy device.

Usage

```
example_epochs
```

Format

`example_epochs`**:**

A data frame with 18,755 rows and 15 columns. Each row represents a time-point (or epoch) in a session. Epochs are 30 seconds long. The columns are as follows:

- timestamp: The time at which the epoch was recorded in UTC.
- subject_id: The ID of the subject.
- signal_quality_mean: The mean signal quality of the epoch.

- movement_fast_mean: The mean movement detected during the epoch.
- movement_fast_nonzero_pct
- distance_mean: the distance of the subject from the device in meters.
- motion_data_count: The number of data points in the epoch (30).
- light_ambient_mean: The ambient light level during the epoch.
- sound_amplitude_mean: The sound amplitude during the epoch.
- temperature_ambient_mean: The ambient temperature during the epoch.
- humidity_mean: The ambient humidity during the epoch.
- pressure_mean: The ambient pressure during the epoch.
- indoor_air_quality_mean: The indoor air quality during the epoch.
- epoch_duration: The precise duration of the epoch (seconds).
- sleep_stage: The sleep stage as established with the VT algorithm. They are encoded as numbers 0-5

Source

data-raw/example_epochs.csv

`example_epochs_v1` *Example Epoch data (Somnofy API v1)*

Description

A data frame containing epoch data recorded by a Somnofy device.

Usage

```
example_epochs_v1
```

Format

`example_epochs_v1`:

A data frame with 1,373 rows and 16 columns. The corresponding session ID is contained in the file name. Each row represents a time-point (or epoch) in a session. Epochs are 30 seconds long. The columns are as follows:

- timestamp: The time at which the epoch was recorded in UTC.
- signal_quality_mean: The mean signal quality of the epoch.
- movement_fast_mean: The mean movement detected during the epoch.
- movement_fast_nonzero_pct
- distance_mean: the distance of the subject from the device in meters.
- motion_data_count: The number of data points in the epoch (30).
- light_ambient_mean: The ambient light level during the epoch.
- sound_amplitude_mean: The sound amplitude during the epoch.
- temperature_ambient_mean: The ambient temperature during the epoch.

- humidity_mean: The ambient humidity during the epoch.
- pressure_mean: The ambient pressure during the epoch.
- indoor_air_quality_mean: The indoor air quality during the epoch.
- epoch_duration: The precise duration of the epoch (seconds).
- sleep_stage: The sleep stage as established with the VT algorithm. They are encoded as numbers 0-5

Source

data-raw/SEtXSxcMEhYXKQAA.example_epochs_v1.csv

`example_sessions` *Example Sessions data*

Description

A data frame containing sessions recorded by a Somnofy device.

Usage

```
example_sessions
```

Format

`example_sessions`:

A data frame with 124 rows and 60 columns. Each row represents a session. Columns contain metadata about the session, including:

- session_start: The start time of the session in UTC.
- session_end: The end time of the session in UTC.
- subject_id: The ID of the subject.
- device_serial_number: The serial number of the device used.
- time_at_sleep_onset: The time at which the subject fell asleep.
- time_at_wakeup: The time at which the subject woke up. Columns also include various metrics averaged over the session, such as:
- mean heart rate
- mean respiration rate Finally, some columns contain environmental parameters, such as:
- Temperature
- Humidity
- Light intensity
- Noise level
- Atmospheric pressure

Source

data-raw/example_sessions.csv

`example_sessions_v1` *Example Sessions data (Somnofy API v1)*

Description

A data frame containing sessions recorded by a Somnofy device.

Usage

```
example_sessions_v1
```

Format

`example_sessions_v1`**:**

A data frame with 87 rows and 70 columns. Each row represents a session. Columns contain metadata about the session, including:

- user_id: The ID of the recorded subject.
- sex: The sex of the recorded subject.
- birth_year: The year of birth of the recorded subject.
- session_start: The start time of the session in UTC.
- session_end: The end time of the session in UTC.
- time_at_sleep_onset: The time at which the subject fell asleep.
- time_at_wakeup: The time at which the subject woke up. Columns also include various metrics averaged over the session, such as:
- mean heart rate
- mean respiration rate Finally, some columns contain environmental parameters, such as:
- Temperature
- Humidity
- Light intensity
- Noise level
- Atmospheric pressure

Source

data-raw/example_sessions_v1.csv

`filter_by_age_range` *Filter sessions by age range*

Description

Filter sessions by age range

Usage

```
filter_by_age_range(
  sessions,
  min_age = NULL,
  max_age = NULL,
  col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`sessions`	The sessions dataframe
`min_age`	The minimum age of the subjects (inclusive)
`max_age`	The maximum age of the subjects (inclusive)
`col_names`	A list to override default column names. This function uses columns: • `birth_year`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that belong to subjects within the specified age range

See Also

Other filtering: `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessions()`, `remove_sessions_no_sleep()`, `select_devices()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- filter_by_age_range(example_sessions_v1, min_age = 11, max_age = 18)
```

`filter_by_night_range` *Filter sessions for nights within a night range*

Description

Filter sessions for nights within a night range

Usage

```
filter_by_night_range(
  sessions,
  from_night,
  to_night,
  col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`sessions`	The sessions dataframe
`from_night`	The start night of the range (inclusive) in YYYY-MM-DD format
`to_night`	The end night of the range (inclusive) in YYYY-MM-DD format
`col_names`	A list to override default column names. This function uses columns: • `night`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that fall within the specified night range

See Also

Other filtering: `filter_by_age_range()`, `filter_by_sex()`, `filter_epochs_from_sessions()`, `remove_sessions_no_sleep()`, `select_devices()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- filter_by_night_range(example_sessions, "2025-04-07", "2025-04-10")
```

`filter_by_sex` *Filter by sex*

Description

Filter by sex

Usage

```
filter_by_sex(sessions, sex, col_names = NULL, flag_only = FALSE)
```

Arguments

`sessions`	The sessions dataframe
`sex`	The sex to filter for (M, F, or NULL for both)
`col_names`	A list to override default column names. This function uses columns: • `sex`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that belong to the specified sex

See Also

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_epochs_from_sessions()`, `remove_sessions_no_sleep()`, `select_devices()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- filter_by_sex(example_sessions_v1, "M")
```

`filter_epochs_from_sessions` *Filter epochs based on session IDs*

Description

Filter epochs based on session IDs

Usage

```
filter_epochs_from_sessions(
  epochs,
  sessions,
  session_col_names = NULL,
  epoch_col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`epochs`	The epochs dataframe
`sessions`	The sessions dataframe
`session_col_names`	A list to override default session column names. This function uses columns: • `id`
`epoch_col_names`	A list to override default epoch column names. This function uses columns: • `session_id`
`flag_only`	If TRUE, only flags the filtered epochs without removing them from the table

Value

The epochs dataframe with only the epochs that belong to the specified sessions

See Also

`filter_by_night_range()` to filter sessions by night range.

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `remove_sessions_no_sleep(`
`select_devices()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`,
`set_session_start_time_range()`

Examples

```
# Apply filtering to sessions to keep specific nights, and filter epochs accordingly
filtered_sessions <- filter_by_night_range(example_sessions, "2025-04-07", "2025-04-10")
filtered_epochs <- filter_epochs_from_sessions(example_epochs, filtered_sessions)
```

`get_epochs_summary` *Summarise epoch information*

Description

This function displays the number of sessions in the epoch data, as well as the start and end dates of the epoch data

Usage

```
get_epochs_summary(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs dataframe
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `session_id`

Value

A single-row dataframe summarising epoch information

See Also

`get_sessions_summary()` to summarise session information.

Other data tables: `get_non_complying_sessions()`, `get_removed_sessions()`, `get_sessions_summary()`

Examples

```
get_epochs_summary(example_epochs)
```

`get_non_complying_sessions` *Get non-complying sessions (i.e. where there is more than one session on the same day)*

Description

Get non-complying sessions (i.e. where there is more than one session on the same day)

Usage

```
get_non_complying_sessions(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `night`

Value

The sessions dataframe with only the sessions that are non-complying

See Also

Other data tables: `get_epochs_summary()`, `get_removed_sessions()`, `get_sessions_summary()`

Examples

```
duplicate_sessions <- get_non_complying_sessions(example_sessions)
```

`get_removed_sessions` *Get a table of sessions that were removed during filtering*

Description

Get a table of sessions that were removed during filtering

Usage

```
get_removed_sessions(sessions, filtered_sessions, col_names = NULL)
```

Arguments

`sessions`	The original sessions dataframe
`filtered_sessions`	The filtered sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `id` • `sleep_period`

Value

The sessions dataframe with only the sessions that were removed during filtering

See Also

Other data tables: `get_epochs_summary()`, `get_non_complying_sessions()`, `get_sessions_summary()`

Examples

```
filtered_sessions <- set_session_start_time_range(example_sessions, "22:00", "06:00")
removed_sessions <- get_removed_sessions(example_sessions, filtered_sessions)
```

`get_sessions_summary` *Make a summary of session information*

Description

This function summarises session information, including the number of sessions, mean session length, mean time at sleep onset and wakeup, subject and device ID.

Usage

```
get_sessions_summary(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe.
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset` • `time_at_wakeup` • `time_in_bed` • `sleep_period`

Value

A single-row dataframe summarizing session information.

See Also

`get_epochs_summary()` to summarise epoch information.

Other data tables: `get_epochs_summary()`, `get_non_complying_sessions()`, `get_removed_sessions()`

Examples

```
get_sessions_summary(example_sessions)
```

`group_epochs_by_night` *Create a grouping by night for epoch data*

Description

Create a grouping by night for epoch data

Usage

```
group_epochs_by_night(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs dataframe
`col_names`	A list to override default column names. This function uses columns: • `timestamp`

Details

The function creates a new column `night` that groups the epochs by night. Timepoints before 12 PM are considered part of the previous night.

Value

The epochs dataframe with the `night` column added

See Also

`group_sessions_by_night()` to group session data by night.

Other time processing: `group_sessions_by_night()`, `max_time()`, `mean_time()`, `min_time()`, `sd_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
epochs <- group_epochs_by_night(example_epochs)
```

---

group_sessions_by_night

Create a grouping by night for session data

Description

Create a grouping by night for session data

Usage

```
group_sessions_by_night(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `session_start`

Details

The function creates a new column `night` that groups the sessions by night depending on their start time. Sessions that start before 12 PM are considered part of the previous night.

Value

The sessions dataframe with the `night` column added

See Also

`group_epochs_by_night()` to group epoch data by night.

Other time processing: `group_epochs_by_night()`, `max_time()`, `mean_time()`, `min_time()`, `sd_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
sessions <- group_sessions_by_night(example_sessions)
```

`interdaily_stability` *Calculate Interdaily Stability (IS)*

Description

This function calculates the Interdaily Stability (IS) metric from a binary awake/asleep variable

Usage

```
interdaily_stability(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs data frame
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `is_asleep`

Value

The Interdaily Stability (IS) value

See Also

Other sleep metrics: `chronotype()`, `composite_phase_deviation()`, `sleep_regularity_index()`, `social_jet_lag()`

Examples

```
interdaily_stability(example_epochs)
```

`load_epochs` *Load epoch data*

Description

Load epoch data

Usage

```
load_epochs(epochs_file)
```

Arguments

`epochs_file`	The path to the epochs file

Details

The function loads the epoch data from a CSV file and groups the epochs by night.

Value

A dataframe containing the epoch data

See Also

Other data loading: `load_sessions()`

`load_sessions` *Load session data*

Description

Load session data

Usage

```
load_sessions(sessions_file)
```

Arguments

`sessions_file`	The path to the sessions file

Details

The function loads the session data from a CSV file and groups the sessions by night.

Value

A dataframe containing the session data

See Also

Other data loading: `load_epochs()`

`max_time` *Calculate the maximum time from 12pm to 12pm*

Description

This function calculates the maximum time from a vector of time strings in the format "YYYY-MM-DD HH:MM:SS". It considers a time window from 12pm to 12pm the next day, so 11:00 is considered later than 13:00.

Usage

```
max_time(time_vector)
```

Arguments

`time_vector`	A vector of time strings in the format "YYYY-MM-DD HH:MM:SS".

Value

A string representing the maximum time in the format "HH:MM".

See Also

`min_time()` to calculate the minimum time in the same format.

Other time processing: `group_epochs_by_night()`, `group_sessions_by_night()`, `mean_time()`, `min_time()`, `sd_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
max_time(c("2025-04-08 23:00:00", "2025-04-09 01:00:00", "2025-04-09 02:30:00"))
```

mean_time — *Calculate the mean time from a vector of time strings*

Description

This function calculates the mean time from a vector of time strings in the format "YYYY-MM-DD HH:MM:SS".

Usage

```
mean_time(time_vector, unit = "HH:MM")
```

Arguments

`time_vector`	A vector of time strings in format "YYYY-MM-DD HH:MM:SS", "HH:MM:SS" or "HH:MM".
`unit`	The unit of time for the result. Can be "HH:MM" (default), "hour", "minute" or "second".

Value

A string representing the mean time in the format "HH:MM".

See Also

Other time processing: `group_epochs_by_night()`, `group_sessions_by_night()`, `max_time()`, `min_time()`, `sd_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
# Use on a vector of time strings representing full dates
time_vector <- c("2025-04-08 23:00:00", "2025-04-09 01:00:00")
mean_time(time_vector)

# Use on time-only strings
time_vector <- c("22:56", "01:32")
mean_time(time_vector)

# Use on a dataframe column
mean_time(example_sessions$time_at_sleep_onset)
```

`min_time` *Calculate the minimum time from 12pm to 12pm*

Description

This function calculates the minimum time from a vector of time strings in the format "YYYY-MM-DD HH:MM:SS". It considers a time window from 12pm to 12pm the next day, so 11:00 is considered later than 13:00.

Usage

```
min_time(time_vector)
```

Arguments

`time_vector`	A vector of time strings in the format "YYYY-MM-DD HH:MM:SS".

Value

A string representing the minimum time in the format "HH:MM".

See Also

`max_time()` to calculate the maximum time in the same format.

Other time processing: `group_epochs_by_night()`, `group_sessions_by_night()`, `max_time()`, `mean_time()`, `sd_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
min_time(c("2025-04-08 23:00:00", "2025-04-09 01:00:00", "2025-04-09 02:30:00"))
```

`plot_actigram` *Plot an Actigram*

Description

Generate an actigram from the Somnofy epoch data.

Usage

```
plot_actigram(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs data frame
`col_names`	A list to override default column names. This function uses columns: • `sleep_period` • `signal_quality_mean` • `sleep_stage`

Value

A ggplot object representing the actigram

`plot_bedtimes_waketimes` *Plot bedtimes and waketimes*

Description

Plot bedtimes and waketimes

Usage

```
plot_bedtimes_waketimes(
  sessions,
  groupby = "night",
  color_by = "default",
  col_names = NULL
)
```

Arguments

`sessions`	The sessions dataframe
`groupby`	The grouping variable for the plot. Can be "night", "workday", or "weekday".
`color_by`	The variable to color the bars by. Can be "default" or any other column name in the sessions dataframe. Note that if color_by is anything else than "default", groupby will be set to "night".
`col_names`	A list to override default column names. This function uses columns: • `night` • `time_at_sleep_onset` • `time_at_wakeup` • `is_workday`

Value

A ggplot graph showing the bedtimes and waketimes

`plot_hypnogram` *Plot Hypnogram*

Description

Plot Hypnogram

Usage

```
plot_hypnogram(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs dataframe
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `sleep_stage`

Value

A ggplot object showing the hypnogram as bars

See Also

`plot_sleep_stages()` to show the proportion of each sleep stage per day

Other plot epochs: `plot_sleep_spiral()`, `plot_sleep_stages()`, `plot_timeseries()`

`plot_sleep_bubbles` *Plot Sleep Bubbles*

Description

This function creates a bubble plot of sleep sessions, where the size and colour of the bubbles represents the sleep duration.

Usage

```
plot_sleep_bubbles(sessions, color_by = "default", col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe.
`color_by`	The variable to color the bubbles by. Can be "default" or any other column name in the sessions dataframe.
`col_names`	A list to override default column names. This function uses columns: • `sleep_period` • `night`

Value

A ggplot object containing the sleep bubbles graph.

See Also

Other plot sessions: `plot_sleep_clock()`, `plot_timeseries_sessions()`

`plot_sleep_clock` *Plot Sleep Clock*

Description

Plot Sleep Clock

Usage

```
plot_sleep_clock(sessions, color_by = "default", col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe
`color_by`	The variable to color the segments by. Can be "default" or any other column name in the sessions dataframe.
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset` • `time_at_wakeup` • `night`

Value

A ggplot object showing the sleep clock

See Also

Other plot sessions: `plot_sleep_bubbles()`, `plot_timeseries_sessions()`

`plot_sleep_spiral` *Plot Sleep Spiral*

Description

Plot Sleep Spiral

Usage

```
plot_sleep_spiral(epochs, color_by = "default", col_names = NULL)
```

Arguments

`epochs`	The epochs dataframe
`color_by`	The variable to color the spiral by. Can be "default" or any other column name in the epochs dataframe.
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `is_asleep`

Value

A ggplot object showing the sleep spiral

See Also

Other plot epochs: `plot_hypnogram()`, `plot_sleep_stages()`, `plot_timeseries()`

`plot_sleep_stages` *Plot Sleep Stages*

Description

Plot Sleep Stages

Usage

```
plot_sleep_stages(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs dataframe
`col_names`	A list to override default column names. This function uses columns: • `night` • `sleep_stage`

Value

A ggplot object showing the proportion of sleep stages for each night

See Also

`plot_hypnogram()` to show the detailed sleep stages over time

Other plot epochs: `plot_hypnogram()`, `plot_sleep_spiral()`, `plot_timeseries()`

`plot_timeseries` *Plot epoch time series data for a given variable*

Description

Plot epoch time series data for a given variable

Usage

```
plot_timeseries(
  epochs,
  variable,
  color_by = "default",
  exclude_zero = FALSE,
  col_names = NULL
)
```

Arguments

`epochs`	The epochs dataframe
`variable`	The variable to plot (e.g., "temperature_ambient_mean")
`color_by`	The variable to color the points by. Can be "default" or any other column name in the epochs dataframe.
`exclude_zero`	Logical, whether to exclude zero values from the plot (default: FALSE)
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `night`

Value

A ggplot object

See Also

`plot_timeseries_sessions()` to plot session data.

Other plot epochs: `plot_hypnogram()`, `plot_sleep_spiral()`, `plot_sleep_stages()`

`plot_timeseries_sessions`
Plot session time series data for a given variable

Description

Plot session time series data for a given variable

Usage

```
plot_timeseries_sessions(
  sessions,
  variable,
  color_by = "default",
  exclude_zero = FALSE,
  col_names = NULL
)
```

Arguments

`sessions`	The sessions dataframe
`variable`	The variable to plot (e.g., "time_at_sleep_onset")
`color_by`	The variable to color the points by. Can be "default" or any other column name in the sessions dataframe.
`exclude_zero`	Logical, whether to exclude zero values from the plot (default: FALSE)
`col_names`	A list to override default column names. This function uses columns: • `night`

Value

A ggplot object

See Also

`plot_timeseries()` to plot epoch data.

Other plot sessions: `plot_sleep_bubbles`(), `plot_sleep_clock`()

remove_sessions_no_sleep *Remove sessions with no sleep*

Description

Remove sessions with no sleep

Usage

```
remove_sessions_no_sleep(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `sleep_period`

Value

The sessions dataframe with only the sessions that have a sleep period greater than 0

See Also

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessio` `select_devices()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- remove_sessions_no_sleep(example_sessions)
```

sd_time *Calculate the circular standard deviation of a vector of times*

Description

This function calculates the standard deviation of a vector of time strings, accounting for the circular nature of time (e.g., 23:59 is close to 00:00).

Usage

```
sd_time(time_vector, unit = "hour")
```

Arguments

`time_vector`	A vector of time strings in format "YYYY-MM-DD HH:MM:SS", "HH:MM:SS" or "HH:MM".
`unit`	The unit of time for the result. Can be "second", "minute", or "hour". Default is "hour".

Value

A numeric value representing the standard deviation in the specified unit.

See Also

Other time processing: `group_epochs_by_night()`, `group_sessions_by_night()`, `max_time()`, `mean_time()`, `min_time()`, `shift_times_by_12h()`, `time_diff()`

Examples

```
sd_time(c("23:59", "00:01"))
```

`select_devices` *Select devices by ID*

Description

Select devices by ID

Usage

```
select_devices(sessions, device_ids, col_names = NULL, flag_only = FALSE)
```

Arguments

`sessions`	The sessions dataframe
`device_ids`	The device IDs to select
`col_names`	A list to override default column names. This function uses columns: • `device_id`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions recorded by the specified devices

See Also

`select_subjects()` to select sessions by subject ID.

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessio` `remove_sessions_no_sleep()`, `select_subjects()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- select_devices(example_sessions, c("VTGVSRTHCA"))
```

---

select_subjects *Select subjects by ID*

Description

Select subjects by ID

Usage

```
select_subjects(sessions, subject_ids, col_names = NULL, flag_only = FALSE)
```

Arguments

`sessions`	The sessions dataframe
`subject_ids`	The subject IDs to select
`col_names`	A list to override default column names. This function uses columns: • `subject_id`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that belong to the specified subjects

See Also

`select_devices()` to select sessions by device ID.

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessio`
`remove_sessions_no_sleep()`, `select_devices()`, `set_min_time_in_bed()`, `set_session_sleep_onset_range()`,
`set_session_start_time_range()`

Examples

```
filtered_sessions <- select_subjects(example_sessions, c("sub_01JNDH3Z5NP0PSV82NFBGPV31X"))
```

`set_data_type` *Set the data type for a dataframe*

Description

Set the data type for a dataframe

Usage

```
set_data_type(df, data_type)
```

Arguments

`df`	The dataframe to set the data type for
`data_type`	The data type to set. Currently available data types: "somnofy_v1", "somnofy_v2"

Details

The dataframe type is used by Ambient Viewer functions to determine the correct column names. Note: you do not need to set the data type if you are using the `load_sessions` or `load_epochs` functions.

Value

The dataframe with the data type set

Examples

```
example_sessions <- set_data_type(example_sessions, "somnofy_v2")
```

`set_min_time_in_bed` *Set minimum time in bed*

Description

Set minimum time in bed

Usage

```
set_min_time_in_bed(
  sessions,
  min_time_in_bed,
  col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`sessions`	The sessions dataframe
`min_time_in_bed`	The minimum time in bed in hours
`col_names`	A list to override default column names. This function uses columns: • `time_in_bed`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that meet the minimum time in bed requirement

See Also

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessio`
`remove_sessions_no_sleep()`, `select_devices()`, `select_subjects()`, `set_session_sleep_onset_range()`, `set_session_start_time_range()`

Examples

```
filtered_sessions <- set_min_time_in_bed(example_sessions, 2)
```

---

set_session_sleep_onset_range

Set sleep onset time range

Description

Set sleep onset time range

Usage

```
set_session_sleep_onset_range(
  sessions,
  from_time,
  to_time,
  col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`sessions`	The sessions dataframe
`from_time`	Include sessions where sleep started after this time (in format HH:MM)
`to_time`	Include sessions where sleep started before this time (in format HH:MM)
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions where sleep started within the specified time range

See Also

`set_session_start_time_range()` to filter sessions based on start time.

Other filtering: `filter_by_age_range()`, `filter_by_night_range()`, `filter_by_sex()`, `filter_epochs_from_sessio`
`remove_sessions_no_sleep()`, `select_devices()`, `select_subjects()`, `set_min_time_in_bed()`,
`set_session_start_time_range()`

Examples

```
filtered_sessions <- set_session_sleep_onset_range(example_sessions, "22:00", "06:00")
```

`set_session_start_time_range`

Set session start time range

Description

Set session start time range

Usage

```
set_session_start_time_range(
  sessions,
  from_time,
  to_time,
  col_names = NULL,
  flag_only = FALSE
)
```

Arguments

`sessions`	The sessions dataframe
`from_time`	Include sessions that started after this time (in format HH:MM)
`to_time`	Include sessions that started before this time (in format HH:MM)
`col_names`	A list to override default column names. This function uses columns: • `session_start`
`flag_only`	If TRUE, only flags the filtered sessions without removing them from the table

Value

The sessions dataframe with only the sessions that started within the specified time range

See Also

set_session_sleep_onset_range() to filter sessions based on sleep onset time.

Other filtering: filter_by_age_range(), filter_by_night_range(), filter_by_sex(), filter_epochs_from_sessio
remove_sessions_no_sleep(), select_devices(), select_subjects(), set_min_time_in_bed(),
set_session_sleep_onset_range()

Examples

```
filtered_sessions <- set_session_start_time_range(example_sessions, "22:00", "06:00")
```

---

shift_times_by_12h *Shift times to break at 12 pm*

Description

This function shifts times so that the day starts at 12 PM. This is useful for plotting night data

Usage

```
shift_times_by_12h(times)
```

Arguments

times	A vector of times in POSIXct format, character convertible to POSIXct, or numerical (in hours).

Value

A vector of times in POSIXct format (or numerical if numerical provided as input) shifted to start at 12 PM

See Also

Other time processing: group_epochs_by_night(), group_sessions_by_night(), max_time(), mean_time(), min_time(), sd_time(), time_diff()

Examples

```
# Shift a vector of times in HH:MM format
shift_times_by_12h(c("02:30", "16:00"))
#> "14:30" "04:00"

# Shift times in YYYY-MM-DD HH:MM:SS format
shift_times_by_12h(c("2025-04-08 23:00:00", "2025-04-09 01:00:00"))
#> "2025-04-08 11:00" "2025-04-09 13:00"

# Shift sessions start times to start at 12 PM
shifted_times <- shift_times_by_12h(example_sessions$session_start)
```

```
# Use dplyr::mutate to dicrectly add the shifted times to a dataframe
epochs <- example_epochs |>
  dplyr::mutate(shifted_time = shift_times_by_12h(timestamp))
```

`sleeptimes_boxplot` *Plot boxplots for sleep onset, midsleep, and wakeup times*

Description

Plot boxplots for sleep onset, midsleep, and wakeup times

Usage

```
sleeptimes_boxplot(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset` • `time_at_wakeup` • `time_at_midsleep`

Value

A ggplot object with three horizontal boxplots (onset, midsleep, wakeup)

`sleeptimes_density` *Plot density curves for sleep onset, midsleep, and wakeup times with a dashed line showing the median*

Description

Plot density curves for sleep onset, midsleep, and wakeup times with a dashed line showing the median

Usage

```
sleeptimes_density(sessions, col_names = NULL, adjust = 1)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset` • `time_at_wakeup` • `time_at_midsleep`
`adjust`	The bandwidth adjustment for the density estimate (default 1)

Value

A ggplot object with three overlaid density curves (sleep onset, midsleep, wakeup)

`sleeptimes_histogram` *Plot histograms for sleep onset, midsleep, and wakeup times*

Description

Plot histograms for sleep onset, midsleep, and wakeup times

Usage

```
sleeptimes_histogram(sessions, col_names = NULL, binwidth = 0.25)
```

Arguments

`sessions`	The sessions dataframe
`col_names`	A list to override default column names. This function uses columns: • `time_at_sleep_onset` • `time_at_wakeup` • `time_at_midsleep`
`binwidth`	The width of the bins for the histogram (default 0.25)

Value

A ggplot object with three overlaid histograms (sleep onset, midsleep, wakeup)

`sleep_regularity_index` *Calculate the Sleep Regularity Index (SRI)*

Description

The Sleep Regularity Index (SRI) is a measure of the regularity of sleep patterns. It is calculated as the percentage of epochs where the sleep state remains the same after 24 hours.

Usage

```
sleep_regularity_index(epochs, col_names = NULL)
```

Arguments

`epochs`	The epochs data frame
`col_names`	A list to override default column names. This function uses columns: • `timestamp` • `is_asleep`

Value

The Sleep Regularity Index (SRI) value

See Also

Other sleep metrics: `chronotype()`, `composite_phase_deviation()`, `interdaily_stability()`, `social_jet_lag()`

Examples

```
sleep_regularity_index(example_epochs)
```

`sleep_report` *Generate a patient sleep report in PDF format*

Description

This function generates a sleep report in PDF format using an R Markdown template. It is designed to work with Somnofy data, so some values may not be available when using other data sources such as GGIR.

Usage

```
sleep_report(
  sessions,
  title = "",
  col_names = NULL,
  output_file = "Sleep_report.pdf"
)
```

Arguments

`sessions`	The sessions dataframe
`title`	The title of the report. Default is an empty string.
`col_names`	A list to override default column names. This function uses columns: • `night` • `time_at_sleep_onset` • `time_at_wakeup` • `time_at_midsleep` • `sleep_onset_latency`
`output_file`	Path for the output PDF. Default is "Sleep_report.pdf"

`social_jet_lag` *Calculate Social Jet Lag*

Description

This function calculates the Social Jet Lag (SJL) metric as the difference in mid-sleep times between workdays and free days.

Usage

```
social_jet_lag(sessions, col_names = NULL)
```

Arguments

`sessions`	The sessions data frame
`col_names`	A list to override default column names. This function uses columns: • `time_at_midsleep` • `is_workday`

Value

The Social Jet Lag (SJL) value in hours

See Also

Other sleep metrics: `chronotype()`, `composite_phase_deviation()`, `interdaily_stability()`, `sleep_regularity_index()`

Examples

```
social_jet_lag(example_sessions)
```

`time_diff` *Compute the forward time difference from t1 to t2 (wrapping at 24)*

Description

This function returns the time from t1 to t2, always moving forward on the clock. For example, from 07:00 to 22:00 is 15 hours, from 22:00 to 07:00 is 9 hours.

Usage

```
time_diff(t1, t2, unit = "hour")
```

Arguments

`t1`	First time (character, POSIXct, or numeric hour)
`t2`	Second time (character, POSIXct, or numeric hour)
`unit`	The unit of time. Can be "second", "minute", or "hour". Default is "hour".

Value

The forward difference in the specified unit (numeric, always positive, $0 <= x < 24$)

See Also

Other time processing: `group_epochs_by_night()`, `group_sessions_by_night()`, `max_time()`, `mean_time()`, `min_time()`, `sd_time()`, `shift_times_by_12h()`

Examples

```
time_diff("07:00", "22:00") # 15
time_diff("22:00", "07:00") # 9
time_diff("07:00", "22:00", unit = "minute") # 540
```

Index