

# Package ‘AmbientViewer’

May 23, 2025

**Title** Filtering and Visualisation for Somnify Data

**Version** 0.0.3

**Description**

This package helps importing, filtering and visualising data from Somnify devices (VitalThings).

**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,  
cli,  
dplyr,  
DT,  
ggnewscale,  
ggplot2,  
logging,  
lubridate,  
readr,  
rlang,  
scales,  
shiny,  
shinyjs,  
shinyWidgets,  
stringr,  
svglite,  
tidyr

**Depends** R (>= 4.3)

**LazyData** true

## Contents

ambient_viewer . . . . .	2
--------------------------	---

convert_angle_to_time . . . . .	3
convert_times_to_mean_angle . . . . .	3
example_epochs . . . . .	4
example_epochs_v1 . . . . .	5
example_sessions . . . . .	5
example_sessions_v1 . . . . .	6
filter_by_age_range . . . . .	7
filter_by_night_range . . . . .	8
filter_by_sex . . . . .	9
filter_epochs_from_sessions . . . . .	9
get_epochs_summary . . . . .	10
get_non_complying_sessions . . . . .	11
get_removed_sessions . . . . .	12
get_sessions_summary . . . . .	12
group_epochs_by_night . . . . .	13
group_sessions_by_night . . . . .	14
load_epochs . . . . .	15
load_sessions . . . . .	15
max_time . . . . .	16
mean_time . . . . .	16
min_time . . . . .	17
plot_actigram . . . . .	18
plot_bedtimes_waketimes . . . . .	18
plot_hypnogram . . . . .	19
plot_sleep_bubbles . . . . .	20
plot_sleep_clock . . . . .	20
plot_sleep_spiral . . . . .	21
plot_sleep_stages . . . . .	21
plot_timeseries . . . . .	22
plot_timeseries_sessions . . . . .	23
remove_sessions_no_sleep . . . . .	23
select_devices . . . . .	24
select_subjects . . . . .	25
set_data_type . . . . .	25
set_min_time_in_bed . . . . .	26
set_session_sleep_onset_range . . . . .	27
set_session_start_time_range . . . . .	28
shift_times_by_12h . . . . .	29

<b>Index</b>	<b>30</b>
--------------	-----------

---

ambient_viewer	<i>Ambient Viewer app</i>
----------------	---------------------------

---

## Description

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

## Usage

```
ambient_viewer()
```

---

`convert_angle_to_time` *Convert an angle to time*

---

**Description**

This function converts an angle in radians to time in the provided unit (can be "second", "minute" or "hour").

**Usage**

```
convert_angle_to_time(angle, unit = "second")
```

**Arguments**

angle	A numeric value representing the angle in radians.
unit	A string indicating the unit of time. Can be "second", "minute", or "hour".

**Value**

A numeric value representing the time in the specified unit.

**See Also**

[convert\\_times\\_to\\_mean\\_angle\(\)](#) to calculate the average angle from a vector of time values.

Other time processing: [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [max\\_time\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

**Examples**

```
convert_angle_to_time(pi/2, unit = "hour")
```

---

`convert_times_to_mean_angle`  
*Convert a vector of times to a mean angle*

---

**Description**

This function converts a vector of times to a mean angle in radians. It is useful to calculate average times spanning midnight

**Usage**

```
convert_times_to_mean_angle(times, unit = "second")
```

**Arguments**

times	A vector of times in seconds.
unit	A string indicating the unit of time. Can be "second", "minute", or "hour".

**Value**

A numeric value representing the mean angle in radians.

**See Also**

[convert\\_angle\\_to\\_time\(\)](#) to convert the mean angle back to time format.

Other time processing: [convert\\_angle\\_to\\_time\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [max\\_time\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

**Examples**

```
convert_times_to_mean_angle(c(23, 10, 0), unit = "hour")
```

---

example_epochs	<i>Example Epoch data</i>
----------------	---------------------------

---

**Description**

A data frame containing epoch data recorded by a Somnify 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	<i>Example Epoch data (Somnofy API v1)</i>
-------------------	--------------------------------------------

---

**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	<i>Example Sessions data</i>
------------------	------------------------------

---

**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	<i>Example Sessions data (Somnify API v1)</i>
---------------------	-----------------------------------------------

---

**Description**

A data frame containing sessions recorded by a Somnify 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	<i>Filter sessions by age range</i>
---------------------	-------------------------------------

---

### Description

Filter sessions by age range

### Usage

```
filter_by_age_range(
  sessions,
  min_age,
  max_age,
  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: <ul style="list-style-type: none"> <li>• birth_year</li> </ul>
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: <ul style="list-style-type: none"><li>• night</li></ul>
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	<i>Filter by sex</i>
---------------	----------------------

---

**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: <ul style="list-style-type: none"> <li>• sex</li> </ul>
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	<i>Filter epochs based on session IDs</i>
-----------------------------	-------------------------------------------

---

**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	<i>Summarise epoch information</i>
--------------------	------------------------------------

---

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: <ul style="list-style-type: none"><li>• night</li></ul> |

**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: <ul style="list-style-type: none"> <li>• time_at_sleep_onset</li> <li>• time_at_wakeup</li> <li>• time_in_bed</li> </ul>

**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: <ul style="list-style-type: none"> <li>• timestamp</li> </ul>

**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: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_sessions\\_by\\_night\\_max\\_time\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

## 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: <ul style="list-style-type: none"><li>• session_start</li></ul>

## 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: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [max\\_time\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

## Examples

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

---

load_epochs	<i>Load epoch data</i>
-------------	------------------------

---

**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	<i>Load session data</i>
---------------	--------------------------

---

**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	<i>Calculate the maximum time from 12pm to 12pm</i>
----------	-----------------------------------------------------

---

**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: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

**Examples**

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

---

mean_time	<i>Calculate the mean time from a vector of time strings</i>
-----------	--------------------------------------------------------------

---

**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)
```

**Arguments**

time\_vector      A vector of time strings in format "YYYY-MM-DD HH:MM:SS", "HH:MM:SS" or "HH:MM".

**Value**

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



See Also

Other time processing: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [max\\_time\(\)](#), [min\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

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	<i>Calculate the minimum time from 12pm to 12pm</i>
----------	-----------------------------------------------------

---

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: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [max\\_time\(\)](#), [mean\\_time\(\)](#), [shift\\_times\\_by\\_12h\(\)](#)

Examples

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

---

plot_actigram	<i>Plot an Actigram</i>
---------------	-------------------------

---

### 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: <ul style="list-style-type: none"> <li>• sleep_period</li> <li>• signal_quality_mean</li> <li>• sleep_stage</li> </ul>

### Value

A ggplot object representing the actigram

---

plot_bedtimes_waketimes	<i>Plot bedtimes and waketimes</i>
-------------------------	------------------------------------

---

### 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	<i>Plot Hypnogram</i>
----------------	-----------------------

---

**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: <ul style="list-style-type: none"><li>• timestamp</li><li>• sleep_stage</li></ul> |

**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	<i>Plot Sleep Bubbles</i>
--------------------	---------------------------

---

**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: <ul style="list-style-type: none"> <li>• time_at_sleep_onset</li> <li>• time_at_wakeup</li> <li>• night</li> </ul>

**Value**

A ggplot object containing the sleep bubbles graph.

**See Also**

Other plot sessions: [plot\\_sleep\\_clock\(\)](#), [plot\\_timeseries\\_sessions\(\)](#)

---

plot_sleep_clock	<i>Plot Sleep Clock</i>
------------------	-------------------------

---

**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: <ul style="list-style-type: none"> <li>• time_at_sleep_onset</li> <li>• time_at_wakeup</li> <li>• night</li> </ul>

**Value**

A ggplot object showing the sleep clock

**See Also**

Other plot sessions: [plot\\_sleep\\_bubbles\(\)](#), [plot\\_timeseries\\_sessions\(\)](#)

---

plot_sleep_spiral	<i>Plot Sleep Spiral</i>
-------------------	--------------------------

---

**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: <ul style="list-style-type: none"><li>• timestamp</li><li>• sleep_stage</li></ul>

**Value**

A ggplot object showing the sleep spiral

**See Also**

Other plot epochs: [plot\\_hypnogram\(\)](#), [plot\\_sleep\\_stages\(\)](#), [plot\\_timeseries\(\)](#)

---

plot_sleep_stages	<i>Plot Sleep Stages</i>
-------------------	--------------------------

---

**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: <ul style="list-style-type: none"> <li>• night</li> <li>• sleep_stage</li> </ul>

**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	<i>Plot epoch time series data for a given variable</i>
-----------------	---------------------------------------------------------

---

**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: <ul style="list-style-type: none"> <li>• timestamp</li> <li>• night</li> </ul>

**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: <ul style="list-style-type: none"> <li>• night</li> </ul>

### 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\\_sessions\(\)](#), [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)
```

---

select_devices	<i>Select devices by ID</i>
----------------	-----------------------------

---

**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\\_sessions\(\)](#), [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	<i>Select subjects by ID</i>
-----------------	------------------------------

---

**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: <ul style="list-style-type: none"> <li>• subject_id</li> </ul>
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\\_sessions\(\)](#), [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	<i>Set the data type for a dataframe</i>
---------------	------------------------------------------

---

**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: "somniafy_v1", "somniafy_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	<i>Set minimum time in bed</i>
---------------------	--------------------------------

---

**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: <ul style="list-style-type: none"> <li>• <code>time_in_bed</code></li> </ul>
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\\_sessions\(\)](#), [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: <ul style="list-style-type: none"> <li>time_at_sleep_onset</li> </ul>
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\\_sessions\(\)](#), [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

<code>sessions</code>	The sessions dataframe
<code>from_time</code>	Include sessions that started after this time (in format HH:MM)
<code>to_time</code>	Include sessions that started before this time (in format HH:MM)
<code>col_names</code>	A list to override default column names. This function uses columns: <ul style="list-style-type: none"> <li>• <code>session_start</code></li> </ul>
<code>flag_only</code>	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\\_sessions\(\)](#), [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	<i>Shift times to break at 12 pm</i>
--------------------	--------------------------------------

---

**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: [convert\\_angle\\_to\\_time\(\)](#), [convert\\_times\\_to\\_mean\\_angle\(\)](#), [group\\_epochs\\_by\\_night\(\)](#), [group\\_sessions\\_by\\_night\(\)](#), [max\\_time\(\)](#), [mean\\_time\(\)](#), [min\\_time\(\)](#)

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

# Index

- \* **data loading**
  - load\_epochs, 15
  - load\_sessions, 15
- \* **data tables**
  - get\_epochs\_summary, 10
  - get\_non\_complying\_sessions, 11
  - get\_removed\_sessions, 12
  - get\_sessions\_summary, 12
- \* **datasets**
  - example\_epochs, 4
  - example\_epochs\_v1, 5
  - example\_sessions, 5
  - example\_sessions\_v1, 6
- \* **filtering**
  - filter\_by\_age\_range, 7
  - filter\_by\_night\_range, 8
  - filter\_by\_sex, 9
  - filter\_epochs\_from\_sessions, 9
  - remove\_sessions\_no\_sleep, 23
  - select\_devices, 24
  - select\_subjects, 25
  - set\_min\_time\_in\_bed, 26
  - set\_session\_sleep\_onset\_range, 27
  - set\_session\_start\_time\_range, 28
- \* **plot epochs**
  - plot\_hypnogram, 19
  - plot\_sleep\_spiral, 21
  - plot\_sleep\_stages, 21
  - plot\_timeseries, 22
- \* **plot sessions**
  - plot\_sleep\_bubbles, 20
  - plot\_sleep\_clock, 20
  - plot\_timeseries\_sessions, 23
- \* **time processing**
  - convert\_angle\_to\_time, 3
  - convert\_times\_to\_mean\_angle, 3
  - group\_epochs\_by\_night, 13
  - group\_sessions\_by\_night, 14
  - max\_time, 16
  - mean\_time, 16
  - min\_time, 17
  - shift\_times\_by\_12h, 29
- ambient\_viewer, 2
- convert\_angle\_to\_time, 3, 4, 13, 14, 16, 17, 29
- convert\_angle\_to\_time(), 4
- convert\_times\_to\_mean\_angle, 3, 3, 13, 14, 16, 17, 29
- convert\_times\_to\_mean\_angle(), 3
- example\_epochs, 4
- example\_epochs\_v1, 5
- example\_sessions, 5
- example\_sessions\_v1, 6
- filter\_by\_age\_range, 7, 8–10, 24–28
- filter\_by\_night\_range, 7, 8, 9, 10, 24–28
- filter\_by\_night\_range(), 10
- filter\_by\_sex, 7, 8, 9, 10, 24–28
- filter\_epochs\_from\_sessions, 7–9, 9, 24–28
- get\_epochs\_summary, 10, 11–13
- get\_epochs\_summary(), 13
- get\_non\_complying\_sessions, 11, 11, 12, 13
- get\_removed\_sessions, 11, 12, 13
- get\_sessions\_summary, 11, 12, 12
- get\_sessions\_summary(), 11
- group\_epochs\_by\_night, 3, 4, 13, 14, 16, 17, 29
- group\_epochs\_by\_night(), 14
- group\_sessions\_by\_night, 3, 4, 13, 14, 16, 17, 29
- group\_sessions\_by\_night(), 13
- load\_epochs, 15, 15
- load\_sessions, 15, 15
- max\_time, 3, 4, 13, 14, 16, 17, 29
- max\_time(), 17
- mean\_time, 3, 4, 13, 14, 16, 16, 17, 29
- min\_time, 3, 4, 13, 14, 16, 17, 17, 29
- min\_time(), 16
- plot\_actigram, 18
- plot\_bedtimes\_waketimes, 18
- plot\_hypnogram, 19, 21, 22

`plot_hypnogram()`, 22  
`plot_sleep_bubbles`, 20, 21, 23  
`plot_sleep_clock`, 20, 20, 23  
`plot_sleep_spiral`, 19, 21, 22  
`plot_sleep_stages`, 19, 21, 21, 22  
`plot_sleep_stages()`, 19  
`plot_timeseries`, 19, 21, 22, 22  
`plot_timeseries()`, 23  
`plot_timeseries_sessions`, 20, 21, 23  
`plot_timeseries_sessions()`, 22  
  
`remove_sessions_no_sleep`, 7–10, 23,  
24–28  
  
`select_devices`, 7–10, 24, 24, 25–28  
`select_devices()`, 25  
`select_subjects`, 7–10, 24, 25, 26–28  
`select_subjects()`, 24  
`set_data_type`, 25  
`set_min_time_in_bed`, 7–10, 24, 25, 26, 27,  
28  
`set_session_sleep_onset_range`, 7–10,  
24–26, 27, 28  
`set_session_sleep_onset_range()`, 28  
`set_session_start_time_range`, 7–10,  
24–27, 28  
`set_session_start_time_range()`, 27  
`shift_times_by_12h`, 3, 4, 13, 14, 16, 17, 29