

Package ‘daoh’

September 22, 2026

Type Package

Title Days Alive and Out of Hospital (DAOH) Calculation

Version 0.2.6

Description Calculates Days Alive and Out of Hospital (DAOH) from administrative admission/discharge/mortality data using three algorithms (nights, days, exact) and three death-handling approaches (midday, midnight, zero). Includes tools for comparing methods (Bland-Altman, ICC, reclassification), and plotting.

License MIT + file LICENSE

Encoding UTF-8

Imports data.table (>= 1.14.0), ggplot2 (>= 3.4.0), scales

Suggests testthat (>= 3.0.0), knitr, rmarkdown, irr

LazyData true

VignetteBuilder knitr

URL <https://github.com/davecumin/daoh>

BugReports <https://github.com/davecumin/daoh/issues>

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author David Cumin [aut, cre]

Maintainer David Cumin <d.cumin@auckland.ac.nz>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2026-09-22 02:10:02 UTC

Contents

bland_altman_daoh	2
calc_daoh	3
daoh_icc	5
daoh_reclassify	6

daoh_reclassify_centile	7
daoh_summary	8
example_daystay	9
example_death	9
example_population	10
hospital_time	11
load_example	12
merge_intervals	13
plot_daoh_ba	14
plot_daoh_dist	15
plot_daoh_reclassify	15
Index	17

bland_altman_daoh	<i>Bland-Altman statistics for two DAOH variants</i>
-------------------	--

Description

Computes the mean difference, standard deviation of differences, and 95% limits of agreement between two DAOH vectors (matched by patientID x indexDate).

Usage

```
bland_altman_daoh(res_a, res_b, use_pc = FALSE)
```

Arguments

- res_a, res_b data.frames (output of [calc_daoh\(\)](#)) for two methods. Must have columns patientID, indexDate, daoh.
- use_pc Logical. If FALSE (default) compare daoh (days); if TRUE, compare daohPC (percentage of the period).

Value

A list with elements mean_diff, sd_diff, loa_lower, loa_upper, centile_01, centile_99, units ("days" or "%", which [plot_daoh_ba\(\)](#) uses to label the axes), and data (data.frame of paired values for plotting).

calc_daoh

*Calculate Days Alive and Out of Hospital (DAOH)***Description**

The main calculation function. For each patient-index-date pair, computes DAOH using the specified hospital-time algorithm and death-handling approach.

Usage

```
calc_daoh(
  events,
  index_dates,
  period = 90,
  method = c("nights", "days", "exact"),
  death_method = c("midday", "midnight", "zero"),
  gap_hours = 12,
  origin = as.Date("1970-01-01")
)
```

Arguments

events	data.frame with one row per hospital event and columns: patientID Character or factor patient identifier. admission Date or POSIXct admission date/time. discharge Date or POSIXct discharge date/time. dod Date of death, or NA if alive (optional; if absent, all patients are assumed to have survived).
index_dates	data.frame with columns patientID and indexDate (Date). Each row defines one DAOH observation window. A patient may have multiple index dates (e.g., multiple surgical episodes).
period	Numeric. Follow-up period in days. Default 90.
method	Character. Hospital-time algorithm: "nights" (default), "days", or "exact".
death_method	Character. How to handle death: "midday" (default), "midnight", or "zero" (sets DAOH = 0 for any death in period).
gap_hours	Numeric. Gap tolerance in hours for merging adjacent admissions. Default 12.
origin	Date. Numeric reference date. Default "1970-01-01".

Value

A data.frame with one row per patient-index-date pair and columns:

patientID Patient identifier.
indexDate Index date.

n_episodes Number of merged hospital episodes in period.
 dih Days in hospital (numeric, using chosen algorithm).
 dd Days dead within the period (0 if survived or death=0).
 daoh DAOH in days.
 daohPC DAOH as a percentage of the period (0-100).

DAOH formula

$$\text{DAOH} = \max(0, T - H - D)$$

where:

T Period length in days (e.g., 90).

H Total hospital time within the period (days), computed from merged, boundary-truncated intervals. See [hospital_time\(\)](#).

D Total dead time within the period (days), computed from date of death. See [dead_time\(\)](#).

Overlapping hospital events are merged using a 12-hour gap tolerance before summing: any two admissions separated by ≤ 12 hours are treated as a single continuous episode. This removes double-counting and models the clinical reality that rapid re-admissions represent continuous care.

Algorithm differences

For N merged hospital episodes in the period, the systematic difference between algorithms is:

$$H^{\text{days}} - H^{\text{nights}} \approx N_{\text{episodes}}$$

because each episode contributes one additional day under the days algorithm (admission = 00:00, discharge = 24:00 vs. both at noon). Therefore:

$$\text{DAOH}^{\text{nights}} - \text{DAOH}^{\text{days}} \approx N_{\text{episodes}}$$

Implementation

Internally uses **data.table** for a fully vectorised pipeline:

1. `foverlaps` matches each admission to every index period it overlaps (handles multiple index dates per patient correctly).
2. Admissions are clipped to their matched period boundary.
3. A single-pass cumulative-max sweep (via `ave`) groups overlapping/near-adjacent clips into merged episodes — equivalent to [merge_intervals\(\)](#), $O(n \log n)$ overall.
4. Dead time and DAOH are computed by vectorised arithmetic.

Column assignments use `data.table::set()` and base-R `ave()` throughout (no `:=`), so the function works correctly when loaded via `devtools::load_all()` as well as from an installed package.

Examples

```
# --- Example 1: Simple day-stay (highlights nights vs days difference) ---
events <- data.frame(
  patientID = "P1",
  admission = as.Date("2020-03-10"),
  discharge = as.Date("2020-03-10"), # same-day admission
  dod       = NA
)
idx <- data.frame(patientID = "P1", indexDate = as.Date("2020-03-10"))

calc_daoh(events, idx, period = 30, method = "nights")$daoh # 30 days
calc_daoh(events, idx, period = 30, method = "days")$daoh   # 29 days

# --- Example 2: Death handling ---
events2 <- data.frame(
  patientID = "P2",
  admission = as.Date("2020-03-10"),
  discharge = as.Date("2020-03-12"),
  dod       = as.Date("2020-03-20") # died 8 days after discharge
)
idx2 <- data.frame(patientID = "P2", indexDate = as.Date("2020-03-10"))

calc_daoh(events2, idx2, period = 30, method = "nights",
  death_method = "midday")$daoh # credits pre-death days
calc_daoh(events2, idx2, period = 30, method = "nights",
  death_method = "zero")$daoh   # returns 0
```

daoh_icc

Intraclass Correlation Coefficient across DAOH methods

Description

Computes two-way mixed ICC (consistency) treating each DAOH calculation method as a rater, following Shrout & Fleiss (1979). Requires the **irr** package.

Usage

```
daoh_icc(results_list, use_pc = FALSE)
```

Arguments

results_list	Named list of data.frames (output of <code>calc_daoh()</code>). All elements must share the same patientID x indexDate pairs.
use_pc	Logical. Use daoh in days (default, FALSE) or daohPC as a percentage (TRUE). ICC is invariant to this per-period rescaling, so the value is the same either way; the argument is provided only for a consistent interface.

Value

The output of `irr::icc()` for the combined method matrix.

daoh_reclassify	<i>Quartile reclassification across two DAOH methods</i>
-----------------	--

Description

Assesses the practical impact of switching from one DAOH method to another by computing the proportion of patients who move between quartiles. Analogous to the Net Reclassification Index (NRI).

Usage

```
daoh_reclassify(res_a, res_b, n_groups = 4, use_pc = FALSE)
```

Arguments

<code>res_a, res_b</code>	data.frames (output of <code>calc_daoh()</code>).
<code>n_groups</code>	Integer. Number of groups (default 4 = quartiles).
<code>use_pc</code>	Logical. Use daoh in days (default, FALSE) or daohPC as a percentage (TRUE). Group membership is invariant to this per-period rescaling, so the result is the same either way.

Value

A list with:

<code>confusion_matrix</code>	Table of group assignments under a vs b.
<code>pct_reclassified</code>	Percentage of patients changing group.
<code>mean_group_shift</code>	Mean absolute group shift.

Examples

```
# See vignette("getting_started", package = "daoh")
```

daoh_reclassify_centile

Centile-boundary reclassification across two DAOH methods

Description

For each specified centile boundary, classifies patients as inside or outside that boundary under each algorithm — using each algorithm's own empirical threshold — then reports the proportion classified differently. This quantifies the clinical impact of algorithm choice at cut-points such as "bottom 10% poor outcome", which are common in adaptive trial enrichment and secondary outcome definitions.

Usage

```
daoh_reclassify_centile(
  res_a,
  res_b,
  boundaries = c(0.05, 0.1, 0.9, 0.95),
  use_pc = FALSE
)
```

Arguments

res_a, res_b	data.frames (output of calc_daoh()). Must have columns patientID, indexDate, and daohPC (or daoh if use_pc = FALSE).
boundaries	Numeric vector of centile probabilities at which to evaluate reclassification. Values below 0.5 define lower ("poor outcome") boundaries; values at or above 0.5 define upper ("excellent outcome") boundaries. Default c(0.05, 0.10, 0.90, 0.95).
use_pc	Logical. Use daoh in days (default, FALSE) or daohPC as a percentage (TRUE). Thresholds and counts are reported in the chosen unit; the reclassification proportions are unaffected.

Details

Unlike [daoh_reclassify\(\)](#), which uses a pooled distribution to set group boundaries, this function applies each algorithm's threshold independently. That reflects the realistic scenario in which a researcher picks a single algorithm, defines a centile-based threshold from their own study population, and applies it — so the threshold itself changes with the algorithm.

Value

A data.frame with one row per boundary and columns:

boundary Human-readable label, e.g. "Bottom 5%" or "Top 10%".

centile The probability supplied in boundaries.

threshold_a Empirical centile value for algorithm A.
 threshold_b Empirical centile value for algorithm B.
 n_patients Number of matched patient-index pairs evaluated.
 n_reclassified Number of patients classified differently across the boundary.
 pct_reclassified Percentage classified differently.

See Also

[daoh_reclassify\(\)](#) for group-based (quartile) reclassification.

Examples

```
# See vignette("getting_started", package = "daoh")
```

daoh_summary	<i>Compute DAOH summary statistics across all methods and periods</i>
--------------	---

Description

Compute DAOH summary statistics across all methods and periods

Usage

```
daoh_summary(results_list, quantiles = c(0.1, 0.25, 0.5), use_pc = FALSE)
```

Arguments

results_list	Named list of data.frames, each the output of calc_daoh() for one method/period combination.
quantiles	Numeric vector of quantiles to report. Default <code>c(0.10, 0.25, 0.50)</code> .
use_pc	Logical. Summarise daoh in days (default, FALSE) or daohPC as a percentage of the period (TRUE).

Value

A data.frame with one row per method/period and columns for mean, median, and the requested quantiles (in days, or percentage when `use_pc = TRUE`).

example_daystay	<i>Synthetic patient example: single day-stay admission</i>
-----------------	---

Description

A minimal example designed to illustrate the difference between the 'nights' and 'days' algorithms for a same-day (day-stay) admission. Under 'nights' the patient contributes 0 nights (DAOH = 30). Under 'days' the patient contributes 1 day (DAOH = 29).

Usage

```
example_daystay
```

Format

A list with two data.frames:

events One row: day-stay admission on the index date, no death.

index_dates One row: the index date for this patient.

Examples

```
data(example_daystay)
calc_daoh(example_daystay$events, example_daystay$index_dates,
  period = 30, method = "nights")
calc_daoh(example_daystay$events, example_daystay$index_dates,
  period = 30, method = "days")
```

example_death	<i>Synthetic patient example: multiple admissions and death</i>
---------------	---

Description

A patient with four hospital admissions who dies 8 days after the last discharge, within a 30-day follow-up period. Illustrates all seven DAOH variants.

Usage

```
example_death
```

Format

A list with two data.frames:

events Four rows of admissions plus date of death.

index_dates One index date (day of first admission).

Examples

```
data(example_death)
# All seven variants
for (meth in c("nights", "days", "exact")) {
  for (dm in c("midday", "midnight", "zero")) {
    res <- calc_daoh(example_death$events, example_death$index_dates,
                     period = 30, method = meth, death_method = dm)
    cat(meth, dm, "DAOH =", round(res$daoh, 2), "\n")
  }
}
```

example_population	<i>Synthetic population: 500 patients, mixed admission patterns</i>
--------------------	---

Description

A larger synthetic dataset suitable for demonstrating summary statistics, Bland-Altman analysis, and reclassification. Patients have between 1 and 5 admissions over a 365-day period, with a 5% mortality rate.

Usage

```
example_population
```

Format

A list with two data.frames:

events Hospital events with columns patientID, admission, discharge, dod.

index_dates One index date per patient.

Examples

```
data(example_population)
res_n <- calc_daoh(example_population$events, example_population$index_dates,
                   period = 90, method = "nights")
res_d <- calc_daoh(example_population$events, example_population$index_dates,
                   period = 90, method = "days")
bland_altman_daoh(res_n, res_d)
```

hospital_time	<i>Convert admission/discharge date pairs to numeric time intervals</i>
---------------	---

Description

Applies one of the three DAOH hospital-time algorithms to convert admission and discharge dates (or datetimes) into numeric time intervals expressed in fractional days from a reference origin.

Usage

```
hospital_time(
  admission_dates,
  discharge_dates,
  method = c("nights", "days", "exact"),
  origin = as.Date("1970-01-01")
)
```

Arguments

admission_dates	Date or POSIXct vector of admission dates/times.
discharge_dates	Date or POSIXct vector of discharge dates/times.
method	Character string: "nights", "days", or "exact".
origin	Date or POSIXct used as numeric zero. Defaults to "1970-01-01". All returned values are days since origin.

Value

A data.frame with columns `start` (numeric, days since origin) and `end` (numeric, days since origin) representing the hospital time interval under the chosen algorithm.

Mathematical definitions

Let a_i be the admission date and d_i the discharge date for event i , expressed as calendar dates (integers at midnight).

Nights algorithm

$$h_i = d_i - a_i$$

Equivalent to assuming both admission and discharge occur at 12:00 (noon). A same-day admission contributes 0 nights. This matches the conventional hospital "length of stay" metric.

Days algorithm

$$h_i = (d_i + 1) - a_i$$

Equivalent to assuming admission at 00:00 and discharge at 24:00 of the respective dates. A same-day admission contributes 1 day. For any admission, $h_i^{\text{days}} = h_i^{\text{nights}} + 1$, so the total difference across N merged episodes equals N days:

$$H^{\text{days}} - H^{\text{nights}} \approx N_{\text{episodes}}$$

Exact algorithm

$$h_i = t_i^{\text{discharge}} - t_i^{\text{admission}}$$

Uses recorded timestamps directly (in fractional days). Partial days are included.

Examples

```
# Same-day admission: nights=0, days=1
hospital_time(
  admission_dates = as.Date("2020-03-01"),
  discharge_dates = as.Date("2020-03-01"),
  method = "nights"
)
hospital_time(
  admission_dates = as.Date("2020-03-01"),
  discharge_dates = as.Date("2020-03-01"),
  method = "days"
)

# Two-night stay
hospital_time(
  admission_dates = as.Date("2020-03-01"),
  discharge_dates = as.Date("2020-03-03"),
  method = "nights" # 2 nights
)
```

load_example	Load a built-in example dataset
--------------	---------------------------------

Description

Reads one of the three synthetic example datasets bundled with the package as CSV files in `inst/extdata/`. This is the preferred way to access examples without needing to run the data-generation script.

Usage

```
load_example(name = c("daystay", "death", "population"))
```

Arguments

name	Character. One of: "daystay" Single day-stay admission: illustrates the 1-day difference between nights and days algorithms (30 vs 29 DAOH over 30 days). "death" Four admissions with in-period death: reproduces the Figure 1 worked example, covering all seven DAOH variants. "population" 500-patient synthetic cohort suitable for Bland-Altman, ICC, and reclassification demonstrations.
------	---

Value

A named list with elements `events` and `index_dates`, both data.frames ready to pass directly to `calc_daoh()`.

Examples

```
ex <- load_example("daystay")
calc_daoh(ex$events, ex$index_dates, period = 30, method = "nights")
calc_daoh(ex$events, ex$index_dates, period = 30, method = "days")
```

merge_intervals	<i>Merge overlapping or near-adjacent time intervals</i>
-----------------	--

Description

Given a set of intervals [start, end], merges those that overlap or are separated by less than gap time units. This is used to consolidate hospital admissions that are separated by short gaps (e.g., < 12 hours), treating them as a single continuous episode.

Usage

```
merge_intervals(starts, ends, gap = 0.5)
```

Arguments

<code>starts</code>	Numeric or POSIXct vector of interval start times.
<code>ends</code>	Numeric or POSIXct vector of interval end times (must be $\geq$ the corresponding start).
<code>gap</code>	Numeric. Intervals with a gap smaller than this value are merged. Units must match starts/ends. Default 0.5 (= 12 hours if times are in days).

Value

A data.frame with columns `start` and `end` representing the merged intervals, sorted by start time.

Examples

```
# Two admissions 10 hours apart -> merged into one
merge_intervals(
  starts = c(0, 1.5),
  ends   = c(1.0, 2.5),
  gap    = 0.5
)

# Two admissions 2 days apart -> kept separate
merge_intervals(
  starts = c(0, 3),
```

```
ends = c(1, 4),  
gap   = 0.5  
)
```

`plot_daoh_ba`*Plot a Bland-Altman comparison of two DAOH methods*

Description

Plot a Bland-Altman comparison of two DAOH methods

Usage

```
plot_daoh_ba(  
  ba_result,  
  method_a = "Method A",  
  method_b = "Method B",  
  use_hex = TRUE,  
  show_loa = FALSE  
)
```

Arguments

<code>ba_result</code>	Output of <code>bland_altman_daoh()</code> .
<code>method_a, method_b</code>	Character labels for the two methods.
<code>use_hex</code>	Logical. Use <code>geom_hex</code> for density (default TRUE, recommended for large datasets).
<code>show_loa</code>	Logical. When TRUE, the numeric mean-bias and 95% limits-of-agreement values are printed directly on the plot, next to their reference lines (in addition to the caption). Default FALSE.

Value

A `ggplot2` object.

plot_daoh_dist	<i>Plot the distribution of DAOH scores</i>
----------------	---

Description

Produces a histogram of DAOH scores with deaths overlaid in a contrasting colour. The y-axis is log-scaled to aid visualisation of the bimodal distribution.

Usage

```
plot_daoh_dist(
  result,
  log_y = TRUE,
  title = "DAOH distribution",
  bins = 50,
  use_pc = FALSE,
  ylab = "Count"
)
```

Arguments

result	data.frame (output of <code>calc_daoh()</code>).
log_y	Logical. Use log10 y-axis (default TRUE).
title	Character. Plot title.
bins	Integer. Number of histogram bins (default 50).
use_pc	Logical. Plot DAOH in days (daoh, the default) or as a percentage of the period (daohPC, when TRUE).
ylab	Character. y-axis label. Defaults to "Count" (the y-axis shows the absolute number of observations per bin).

Value

A ggplot2 object.

plot_daoh_reclassify	<i>Visualise quartile reclassification between two DAOH methods</i>
----------------------	---

Description

Plots a heatmap of the reclassification table (method A group vs method B group) with cell counts and an annotation of the overall reclassification rate.

Usage

```
plot_daoh_reclassify(  
  reclass_result,  
  method_a = "Method A",  
  method_b = "Method B"  
)
```

Arguments

`reclass_result` Output of `daoh_reclassify()`.
`method_a, method_b`
Character labels for the two methods.

Value

A ggplot2 object.

Index

- * **datasets**
 - example_daystay, [9](#)
 - example_death, [9](#)
 - example_population, [10](#)
- bland_altman_daoh, [2](#)
- bland_altman_daoh(), [14](#)
- calc_daoh, [3](#)
- calc_daoh(), [2](#), [5–8](#), [13](#), [15](#)
- daoh_icc, [5](#)
- daoh_reclassify, [6](#)
- daoh_reclassify(), [7](#), [8](#), [16](#)
- daoh_reclassify_centile, [7](#)
- daoh_summary, [8](#)
- dead_time(), [4](#)
- example_daystay, [9](#)
- example_death, [9](#)
- example_population, [10](#)
- hospital_time, [11](#)
- hospital_time(), [4](#)
- irr::icc(), [6](#)
- load_example, [12](#)
- merge_intervals, [13](#)
- merge_intervals(), [4](#)
- plot_daoh_ba, [14](#)
- plot_daoh_ba(), [2](#)
- plot_daoh_dist, [15](#)
- plot_daoh_reclassify, [15](#)