Package ‘AdhereR’

June 11, 2019

Type Package
Title Adherence to Medications
Version 0.5
Author Dan Dediu [aut, cre],
       Alexandra Dima [aut],
       Samuel Allemann [aut]
Maintainer Dan Dediu <ddediu@gmail.com>
Description Computation of adherence to medications from Electronic Health care
   Data and visualization of individual medication histories and adherence
   patterns. The package implements a set of S3 classes and
   functions consistent with current adherence guidelines and definitions.
   It allows the computation of different measures of
   adherence (as defined in the literature, but also several original ones),
   their publication-quality plotting,
   the estimation of event duration and time to initiation,
   the interactive exploration of patient medication history and
   the real-time estimation of adherence given various parameter settings.
   It scales from very small datasets stored in flat CSV files to very large
   databases and from single-thread processing on mid-range consumer
   laptops to parallel processing on large heterogeneous computing clusters.
   It exposes a standardized interface allowing it to be used from other
   programming languages and platforms, such as Python.
URL https://github.com/ddediu/AdhereR
License GPL (>= 2)
LazyData TRUE
RoxygenNote 6.1.1
Imports lubridate (>= 1.5), parallel (>= 3.0), data.table (>= 1.9),
       manipulate (>= 1.0), shiny (>= 1.0), shinyWidgets (>= 0.4.4),
       shinyjs (>= 1.0), V8 (>= 1.5), colourpicker (>= 1.0),
       viridisLite(>= 0.3), highlight (>= 0.4), clipr (>= 0.4), knitr
       (>= 1.20), readODS (>= 1.6), readxl (>= 1.2), haven (>= 2.0),
       DBI (>= 1.0), RMariaDB (>= 1.0.5), RSQLite (>= 2.1), scales (>=
       1.0)
R topics documented:
callAdhereR .................................................. 3
CMA0 ............................................................ 4
CMA1 ............................................................ 7
CMA2 ............................................................ 12
CMA5 ............................................................ 16
CMA6 ............................................................ 21
CMA7 ............................................................ 25
CMA8 ............................................................ 29
CMA9 ............................................................ 34
CMA_per_episode ........................................... 39
CMA_sliding_window ...................................... 44
compute.event.int.gaps .................................. 48
compute.treatment.episodes ............................ 52
compute_event_durations ................................. 55
cover_special_periods .................................. 60
durcomp.dispensing ...................................... 62
durcomp.hospitalisation ................................ 63
durcomp.prescribing ...................................... 64
getCallerWrapperLocation ............................... 65
getCMA ......................................................... 65
med.events .................................................... 66
plot.CMA0 ..................................................... 67
plot.CMA1 ..................................................... 70
plot.CMA_per_episode .................................. 74
plot_interactive_cma .................................... 78
print.CMA0 ................................................... 81
prune_event_durations .................................. 83
time_to_initiation ....................................... 85

Index 87
Description

The function encapsulating all the logics that allows AdhereR to be called from any platform using the generic shell mechanism.

Usage

callAdhereR(shared.data.directory)

Arguments

shared.data.directory

A string containing the path to the directory where all the exchanged (shared) data (both input and output) is. AdhereR needs read and write access to this directory.

Details

In most cases this should not be done directly by the user, but instead used by an appropriate wrapper on the client platform. It allows transparent use of AdhereR from virtually any platform or programming language for which an appropriate wrapper is provided. For more details see the vignette describing the included reference Python 3 wrapper.

Value

This function displays any messages to the console, tries to also write them to the AdhereR-results.txt file in the shared.data.directory directory, and, when finished, forces R to quit with a given shell error code:

• 0 The processing ended without major errors;
• 1 General error (hopefully there are messages in the AdhereR-results.txt file);
• 10 The directory shared.data.directory does not exit;
• 11 AdhereR does not have read access to the shared.data.directory directory;
• 12 AdhereR does not have write access to the shared.data.directory directory;
• 13 issues with the parameters file parameters.log;
• 14 issues with the data file dataset.csv;
• 15 plotting issues;
• 16 interactive plotting issues;
• 17 issues exporting the results.
Description

Constructs a basic CMA (continuous multiple-interval measures of medication availability/gaps) object.

Usage

CMA0(data = NULL, ID.colname = NA, event.date.colname = NA, event.duration.colname = NA, event.daily.dose.colname = NA, medication.class.colname = NA, medication.groups = NULL, carryover.within.obs.window = NA, carryover.into.obs.window = NA, carry.only.for.the.same.medication = NA, consider.dosage.change = NA, followup.window.start = 0, followup.window.start.unit = c("days", "weeks", "months", "years")[], followup.window.duration = 365 * 2, followup.window.duration.unit = c("days", "weeks", "months", "years")[], observation.window.start = 0, observation.window.start.unit = c("days", "weeks", "months", "years")[], observation.window.duration = 365 * 2, observation.window.duration.unit = c("days", "weeks", "months", "years")[], date.format = "%d/%m/%Y", summary = "Base CMA object", suppress.warnings = FALSE, arguments.that.should.not.be-defined = NULL, ...)

Arguments

data  A data.frame containing the medication events (prescribing or dispensing) used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

ID.colname  A string, the name of the column in data containing the unique patient ID, or NA if not defined.

event.date.colname  A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter), or NA if not defined.

event.duration.colname  A string, the name of the column in data containing the event duration (in days), or NA if not defined.

event.daily.dose.colname  A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

medication.class.colname  A string, the name of the column in data containing the classes/types/groups of medication, or NA if not defined.
A list of vectors of medication class names; if (some of) these vectors are named, these names will be used the names of the classes, otherwise automatic names will be generated by concatenating their contents separated by "+". One example could be, list(c("A", "B"), "G2"=c("C", "D", "E")). Class names that are not included in the list are considered to be their own group. If NULL (the default), there's a single group containing all the medications.

carryover.within.obs.window
Logical, if TRUE consider the carry-over within the observation window, or NA if not defined.

carryover.into.obs.window
Logical, if TRUE consider the carry-over from before the starting date of the observation window, or NA if not defined.

carry.only.for.same.medication
Logical, if TRUE the carry-over applies only across medications of the same type, or NA if not defined.

consider.dosage.change
Logical, if TRUE the carry-over is adjusted to reflect changes in dosage, or NA if not defined.

followup.window.start
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

followup.window.start.unit
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit
the definition of the observation window (see the follow-up window parameters above for details).

date.format
A string giving the format of the dates used in the data and the other parameters; see the format parameters of the \texttt{as.Date} function for details (NB, this concerns only the dates given as strings and not as Date objects).

summary
Metadata as a string, briefly describing this CMA.
suppressWarnings

Logical, if TRUE don't show any warnings.

arguments.that.should.not.be_defined

a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

... other possible parameters

Details

In most cases this should not be done directly by the user, but it is used internally by the other CMAs.

Value

An S3 object of class CMA0 with the following fields:

- data The actual event (prescribing or dispensing) data, as given by the data parameter.
- ID.colname the name of the column in data containing the unique patient ID, as given by the ID.colname parameter.
- event.date.colname the name of the column in data containing the start date of the event (in the format given in the date.format parameter), as given by the event.date.colname parameter.
- event.duration.colname the name of the column in data containing the event duration (in days), as given by the event.duration.colname parameter.
- event.daily.dose.colname the name of the column in data containing the prescribed daily dose, as given by the event.daily.dose.colname parameter.
- medication.class.colname the name of the column in data containing the classes/types/groups of medication, as given by the medication.class.colname parameter.
- carryover.within.obs.window whether to consider the carry-over within the observation window, as given by the carryover.within.obs.window parameter.
- carryover.into.obs.window whether to consider the carry-over from before the starting date of the observation window, as given by the carryover.into.obs.window parameter.
- carry.only.for.same.medication whether the carry-over applies only across medication of the same type, as given by the carry.only.for.same.medication parameter.
- consider.dosage.change whether the carry-over is adjusted to reflect changes in dosage, as given by the consider.dosage.change parameter.
- followup.window.start the beginning of the follow-up window, as given by the followup.window.start parameter.
- followup.window.start.unit the time unit of the followup.window.start, as given by the followup.window.start.unit parameter.
- followup.window.duration the duration of the follow-up window, as given by the followup.window.duration parameter.
- followup.window.duration.unit the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.
• observation.window.start the beginning of the observation window, as given by the observation.window.start parameter.

• observation.window.start.unit the time unit of the observation.window.start, as given by the observation.window.start.unit parameter.

• observation.window.duration the duration of the observation window, as given by the observation.window.duration parameter.

• observation.window.duration.unit the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.

• date.format the format of the dates, as given by the date.format parameter.

• summary the metadata, as given by the summary parameter.

Examples

```r
library(medevents)

cma0 <- CMA0(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  followup.window.start=0,
  followup.window.start.unit="days",
  followup.window.duration=2*365,
  followup.window.duration.unit="days",
  observation.window.start=30,
  observation.window.start.unit="days",
  observation.window.duration=365,
  observation.window.duration.unit="days",
  date.format="%m/%d/%y",
  summary="Base CMA");
```

CMA1 and CMA3 constructors.

Description

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 1 or type 3 object.

Usage

```r
CMA1(data = NULL, ID.colname = NA, event.date.colname = NA, 
  event.duration.colname = NA, followup.window.start = 0, 
  followup.window.start.unit = c("days", "weeks", "months", "years") [1],
  followup.window.duration = 365 * 2,
  followup.window.duration.unit = c("days", "weeks", "months", 
  "years") [1], observation.window.start = 0, 
  observation.window.start.unit = c("days", "weeks", "months"),
```

```
"years")[1], observation.window.duration = 365 * 2,
observation.window.duration.unit = c("days", "weeks", "months",
"years")[1], date.format = "%m/%d/%Y", summary = NA,
event.interval.colname = "event.interval",
gap.days.colname = "gap.days",
force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none",
"multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[1],
parallel.threads = "auto", suppress.warnings = FALSE,
arguments.that.should.not.be.defined = c(carryover.within.obs.window =
FALSE, carryover.into.obs.window = FALSE, carry.only.for.same.medication
= FALSE, consider.dosage.change = FALSE), ...)

CMA3(data = NULL, ID.colname = NA, event.date.colname = NA,
event.duration.colname = NA, followup.window.start = 0,
followup.window.start.unit = c("days", "weeks", "months", "years")[1],
followup.window.duration = 365 * 2,
followup.window.duration.unit = c("days", "weeks", "months",
"years")[1], observation.window.start = 0,
observation.window.start.unit = c("days", "weeks", "months",
"years")[1], observation.window.duration = 365 * 2,
observation.window.duration.unit = c("days", "weeks", "months",
"years")[1], date.format = "%m/%d/%Y", summary = NA,
event.interval.colname = "event.interval",
gap.days.colname = "gap.days",
force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none",
"multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[1],
parallel.threads = "auto", suppress.warnings = FALSE,
arguments.that.should.not.be.defined = c(carryover.within.obs.window =
FALSE, carryover.into.obs.window = FALSE, carry.only.for.same.medication
= FALSE, consider.dosage.change = FALSE), ...)

Arguments

- **data**
  A `data.frame` containing the events used to compute the CMA. Must contain,
at a minimum, the patient unique ID, the event date and duration, and might
also contain the daily dosage and medication type (the actual column names are
defined in the following four parameters).

- **id.colname**
  A `string`, the name of the column in `data` containing the unique patient ID; must
be present.

- **event.date.colname**
  A `string`, the name of the column in `data` containing the start date of the event
(in the format given in the `date.format` parameter); must be present.

- **event.duration.colname**
  A `string`, the name of the column in `data` containing the event duration (in days);
must be present.

- **followup.window.start**
  If a Date object, it represents the actual start date of the follow-up window; if a
`string` it is the name of the column in `data` containing the start date of the follow-
up window either as the numbers of \texttt{followup.window.start.unit} units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type \texttt{Date}); if a \texttt{number} it is the number of time units defined in the \texttt{followup.window.start.unit} parameter after the begin of the participant’s first event; or NA if not defined.

\texttt{followup.window.start.unit}

can be either \textit{"days"}, \textit{"weeks"}, \textit{"months"} or \textit{"years"}, and represents the time units that \texttt{followup.window.start} refers to (when a number), or NA if not defined.

\texttt{followup.window.duration}

either a \texttt{number} representing the duration of the follow-up window in the time units given in \texttt{followup.window.duration.unit}, or a \texttt{string} giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

\texttt{followup.window.duration.unit}

can be either \textit{"days"}, \textit{"weeks"}, \textit{"months"} or \textit{"years"}, and represents the time units that \texttt{followup.window.duration} refers to, or NA if not defined.

\texttt{observation.window.start}, \texttt{observation.window.start.unit}, \texttt{observation.window.duration}, \texttt{observation.window.duration.unit}

the definition of the observation window (see the follow-up window parameters above for details).

\texttt{date.format}

A \texttt{string} giving the format of the dates used in the data and the other parameters; see the \texttt{format} parameters of the \texttt{as.Date} function for details (NB, this concerns only the dates given as strings and not as \texttt{Date} objects).

\texttt{summary}

Metadata as a \texttt{string}, briefly describing this CMA.

\texttt{event.interval.colname}

A \texttt{string}, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value \texttt{"event.interval"} should be changed only if there is a naming conflict with a pre-existing \texttt{"event.interval"} column in \texttt{event.info}.

\texttt{gap.days.colname}

A \texttt{string}, the name of a newly-created column storing the number of days when medication was not available (i.e., the \texttt{"gap days"}); the default value \texttt{"gap.days"} should be changed only if there is a naming conflict with a pre-existing \texttt{"gap.days"} column in \texttt{event.info}.

\texttt{force.NA.CMA.for.failed.patients}

\texttt{Logical} describing how the patients for which the CMA estimation fails are treated: if \texttt{TRUE} they are returned with an NA CMA estimate, while for \texttt{FALSE} they are omitted.

\texttt{parallel.backend}

Can be \texttt{"none"} (the default) for single-threaded execution, \texttt{"multicore"} (using \texttt{mclapply} in package \texttt{parallel}) for multicore processing (NB. not currently implemented on MS Windows and automatically falls back on \texttt{"snow"} on this platform), or \texttt{"snow"}, \texttt{"snow(SOCK)"} (equivalent to \texttt{"snow"}), \texttt{"snow(MPI)"} or \texttt{"snow(NWS)"} specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package
snow for details; the last two require packages \texttt{Rmpi} and \texttt{nws}, respectively, not automatically installed with \texttt{AdhereR}).

\texttt{parallel.threads}

Can be "auto" (for \texttt{parallel.backend} == "multicore"), defaults to the number of cores in the system as given by \texttt{options("cores")}, while for \texttt{parallel.backend} == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for \texttt{parallel.backend} == "snow" (see the documentation of package \texttt{snow} for details).

\texttt{suppress.warnings}

Logical, if \texttt{TRUE} don't show any warnings.

\texttt{arguments.that.should.not.be.defined}

a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

\ldots

other possible parameters

\section*{Details}

\texttt{CMA1} considers the total number of days with medication supplied in all medication events in the observation window, excluding the last event. \texttt{CMA3} is identical to \texttt{CMA1} except that it is capped at 100%.

The formula is

\[
\frac{\text{number of dayssupplyexcludinglast}}{\text{firsttolastevent}}
\]

Thus, the durations of all events are added up, possibly resulting in an CMA estimate (much) bigger than 1.0 (100%).

\texttt{CMA2} and \texttt{CMA1} differ in the inclusion or not of the last event.

\section*{Value}

An S3 object of class \texttt{CMA1} (derived from \texttt{CMA0}) with the following fields:

- \texttt{data} The actual event data, as given by the \texttt{data} parameter.
- \texttt{ID.colname} the name of the column in \texttt{data} containing the unique patient ID, as given by the \texttt{ID.colname} parameter.
- \texttt{event.date.colname} the name of the column in \texttt{data} containing the start date of the event (in the format given in the \texttt{date.format} parameter), as given by the \texttt{event.date.colname} parameter.
- \texttt{event.duration.colname} the name of the column in \texttt{data} containing the event duration (in days), as given by the \texttt{event.duration.colname} parameter.
- \texttt{event.daily.dose.colname} the name of the column in \texttt{data} containing the prescribed daily dose, as given by the \texttt{event.daily.dose.colname} parameter.
- \texttt{medication.class.colname} the name of the column in \texttt{data} containing the classes/types/groups of medication, as given by the \texttt{medication.class.colname} parameter.
- \texttt{followup.window.start} the beginning of the follow-up window, as given by the \texttt{followup.window.start} parameter.
followup.window.start the time unit of the followup.window.start, as given by the followup.window.start.unit parameter.

followup.window.duration the duration of the follow-up window, as given by the followup.window.duration parameter.

followup.window.duration.unit the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.

observation.window.start the beginning of the observation window, as given by the observation.window.start parameter.

observation.window.start.unit the time unit of the observation.window.start, as given by the observation.window.start.unit parameter.

observation.window.duration the duration of the observation window, as given by the observation.window.duration parameter.

observation.window.duration.unit the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.

date.format the format of the dates, as given by the date.format parameter.

summary the metadata, as given by the summary parameter.

event.info the data.frame containing the event info (irrelevant for most users; see compute.event.int.gaps for details).

CMA the data.frame containing the actual CMA estimates for each participant (the ID.colname column).

See Also

CMAs 1 to 8 are described in:


Examples

cma1 <- CMA1(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);

cma3 <- CMA3(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
CMA2 and CMA4 constructors.

Description

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 2 or type 4 object.

Usage

```r
CMA2(data = NULL, ID.colname = NA, event.date.colname = NA,
    event.duration.colname = NA, followup.window.start = 0, 
    followup.window.start.unit = c("days", "weeks", "months", "years")[[1]], 
    followup.window.duration = 365 * 2, 
    followup.window.duration.unit = c("days", "weeks", "months", 
    "years")[[1]], observation.window.start = 0, 
    observation.window.duration = 365 * 2, 
    observation.window.duration.unit = c("days", "weeks", "months", 
    "years")[[1]], date.format = "%m/%d/%Y", summary = NA, 
    event.interval.colname = "event.interval", 
    gap.days.colname = "gap.days", 
    force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", 
    "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)"), 
    parallel.threads = "auto", suppress.warnings = FALSE, 
    arguments.that.should.not.be.defined = c(carryover.within.obs.window = 
    FALSE, carryover.into.obs.window = FALSE, carry.only.for.same.medication 
    = FALSE, consider.dosage.change = FALSE), ...)
```

```r
CMA4(data = NULL, ID.colname = NA, event.date.colname = NA, 
    event.duration.colname = NA, followup.window.start = 0, 
    followup.window.start.unit = c("days", "weeks", "months", "years")[[1]], 
    followup.window.duration = 365 * 2, 
    followup.window.duration.unit = c("days", "weeks", "months", 
    "years")[[1]], observation.window.start = 0, 
    observation.window.duration = 365 * 2, 
    observation.window.duration.unit = c("days", "weeks", "months", 
    "years")[[1]], date.format = "%m/%d/%Y", summary = NA, 
    event.interval.colname = "event.interval", 
    gap.days.colname = "gap.days", 
    force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", 
    "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)"), 
    parallel.threads = "auto", suppress.warnings = FALSE, 
    arguments.that.should.not.be.defined = c(carryover.within.obs.window = 
    FALSE, carryover.into.obs.window = FALSE, carry.only.for.same.medication 
    = FALSE, consider.dosage.change = FALSE), ...)
```
Arguments

data A data.frame containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

ID.colname A string, the name of the column in data containing the unique patient ID; must be present.

event.date.colname A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter); must be present.

event.duration.colname A string, the name of the column in data containing the event duration (in days); must be present.

followup.window.start If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

followup.window.start.unit can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit the definition of the observation window (see the follow-up window parameters above for details).

date.format A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

summary Metadata as a string, briefly describing this CMA.

event.interval.colname A string, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in event.info.
gap.days.colname

A string, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in event.info.

force.NA.CMA.for.failed.patients

Logical describing how the patients for which the CMA estimation fails are treated: if TRUE they are returned with an NA CMA estimate, while for FALSE they are omitted.

parallel.backend

Can be "none" (the default) for single-threaded execution, "multicore" (using mclapply in package parallel) for multicore processing (NB. not currently implemented on MS Windows and automatically falls back on "snow" on this platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or "snow(NWS)" specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package snow for details; the last two require packages Rmpi and nws, respectively, not automatically installed with AdhereR).

parallel.threads

Can be "auto" (for parallel.backend == "multicore", defaults to the number of cores in the system as given by options("cores"), while for parallel.backend == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for parallel.backend == "snow" (see the documentation of package snow for details).

suppress.warnings

Logical, if TRUE don’t show any warnings.

arguments.that.should.not.be.defined

a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

... other possible parameters

Details

CMA2 considers the total number of days with medication supplied in all medication events in the observation window, including the last event. CMA4 is identical to CMA2 except that it is capped at 100%.

The formula is

\[
\frac{\text{number of dayssupply including last event}}{\text{first to last event}}
\]

Thus, the durations of all events are added up, possibly resulting in an CMA estimate (much) bigger than 1.0 (100%)

CMA2 and CMA1 differ in the inclusion or not of the last event.

Value

An S3 object of class CMA2 (derived from CMA0) with the following fields:
• data The actual event data, as given by the data parameter.

• ID.colname the name of the column in data containing the unique patient ID, as given by the ID.colname parameter.

• event.date.colname the name of the column in data containing the start date of the event (in the format given in the date.format parameter), as given by the event.date.colname parameter.

• event.duration.colname the name of the column in data containing the event duration (in days), as given by the event.duration.colname parameter.

• event.daily.dose.colname the name of the column in data containing the prescribed daily dose, as given by the event.daily.dose.colname parameter.

• medication.class.colname the name of the column in data containing the classes/types/groups of medication, as given by the medication.class.colname parameter.

• followup.window.start the beginning of the follow-up window, as given by the followup.window.start parameter.

• followup.window.unit the time unit of the followup.window.start, as given by the followup.window.unit parameter.

• followup.window.duration the duration of the follow-up window, as given by the followup.window.duration parameter.

• followup.window.unit the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.

• observation.window.start the beginning of the observation window, as given by the observation.window.start parameter.

• observation.window.unit the time unit of the observation.window.start, as given by the observation.window.unit parameter.

• observation.window.duration the duration of the observation window, as given by the observation.window.duration parameter.

• observation.window.unit the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.

• date.format the format of the dates, as given by the date.format parameter.

• summary the metadata, as given by the summary parameter.

• event.info the data.frame containing the event info (irrelevant for most users; see compute.event.intNgaps for details).

• CMA the data.frame containing the actual CMA estimates for each participant (the ID.colname column).

See Also

CMAs 1 to 8 are defined in:

Examples

```r
## Not run:
cma2 <- CMA5(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
); cma4 <- CMA5(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
## End(Not run)
```

### CMA5

**CMA5 constructor.**

**Description**

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 5 object.

**Usage**

```r
CMA5(data = NULL, ID.colname = NA, event.date.colname = NA,
  event.duration.colname = NA, event.daily.dose.colname = NA,
  medication.class.colname = NA,
  carry.only.for.same.medication = FALSE,
  consider.dosage.change = FALSE, followup.window.start = 0,
  followup.window.duration = 365 * 2,
  followup.window.duration.unit = c("days", "weeks", "months",
  "years")[1],
  observation.window.start = 0,
  observation.window.duration = 365 * 2,
  observation.window.duration.unit = c("days", "weeks", "months",
  "years")[1],
  date.format = "%m/%d/%Y", summary = NA,
  event.interval.colname = "event.interval",
  gap.days.colname = "gap.days",
  force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none",
```
CMA5

"multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)"
[1],
parallel.threads = "auto", suppress.warnings = FALSE,
arguments.that.should.not.beDEFINED = c(carriOver.within.obs.window =
TRUE, carryover.into.obs.window = FALSE), ...

Arguments

data A data.frame containing the medication events used to compute the CMA.
Must contain, at a minimum, the patient unique ID, the event date and duration,
and might also contain the daily dosage and medication type (the actual column
names are defined in the following four parameters).

id.colname A string, the name of the column in data containing the unique patient ID; must
be present.

event.date.colname A string, the name of the column in data containing the start date of the event
(in the format given in the date.format parameter); must be present.

event.duration.colname A string, the name of the column in data containing the event duration (in days);
must be present.

event.daily.dose.colname A string, the name of the column in data containing the prescribed daily dose,
or NA if not defined.

medication.class.colname
A string, the name of the column in data containing the medication type, or NA
if not defined.

carry.only.for.same.medication Logical, if TRUE, the carry-over applies only across medication of the same type.

consider.dosage.change Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.

followup.window.start If a Date object, it represents the actual start date of the follow-up window; if a
string it is the name of the column in data containing the start date of the follow-
up window either as the numbers of followup.window.start.unit units after
the first event (the column must be of type numeric) or as actual dates (in which
case the column must be of type Date); if a number it is the number of time units
defined in the followup.window.start.unit parameter after the begin of the
participant's first event; or NA if not defined.

followup.window.start.unit
can be either "days", "weeks", "months" or "years", and represents the time units
that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
either a number representing the duration of the follow-up window in the time
units given in followup.window.duration.unit, or a string giving the col-
umn containing these numbers. Should represent a period for which relevant
medication events are recorded accurately (e.g. not extend after end of relevant
treatment, loss-to-follow-up or change to a health care provider not covered by
the database).
followup.window.duration.unit
  can be either "days", "weeks", "months" or "years", and represents the time units
  that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit
  the definition of the observation window (see the follow-up window parameters
  above for details).

date.format
  A string giving the format of the dates used in the data and the other param-
  eters; see the format parameters of the as.Date function for details (NB, this
  concerns only the dates given as strings and not as Date objects).

summary
  Metadata as a string, briefly describing this CMA.

event.interval.colname
  A string, the name of a newly-created column storing the number of days be-
  tween the start of the current event and the start of the next one; the default
  value "event.interval" should be changed only if there is a naming conflict with a
  pre-existing "event.interval" column in event.info.

gap.days.colname
  A string, the name of a newly-created column storing the number of days when
  medication was not available (i.e., the "gap days"); the default value "gap.days"
  should be changed only if there is a naming conflict with a pre-existing "gap.days"
  column in event.info.

force.NA.CMA.for.failed.patients
  Logical describing how the patients for which the CMA estimation fails are
  treated: if TRUE they are returned with an NA CMA estimate, while for FALSE
  they are omitted.

parallel.backend
  Can be "none" (the default) for single-threaded execution, "multicore" (using
  mclapply in package parallel) for multicore processing (NB, not currently
  implemented on MS Windows and automatically falls back on "snow" on this
  platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
  "snow(NWS)" specifying various types of SNOW clusters (can be on the local
  machine or more complex setups – please see the documentation of package
  snow for details; the last two require packages Rmpi and nws, respectively, not
  automatically installed with AdhereR).

parallel.threads
  Can be "auto" (for parallel.backend == "multicore", defaults to the number of
  cores in the system as given by options("cores")), while for parallel.backend
  == "snow", defaults to 2), a strictly positive integer specifying the number of
  parallel threads, or a more complex specification of the SNOW cluster nodes
  for parallel.backend == "snow" (see the documentation of package snow for
details).

suppress.warnings
  Logical, if TRUE don't show any warnings.

arguments.that.should.not.be_defined
  a list of argument names and pre-defined values for which a warning should be
  thrown if passed to the function.

... other possible parameters
Details

CMA5 assumes that, within the observation window, the medication is used as prescribed and new medication is "banked" until needed (oversupply from previous events is used first, followed new medication supply). It computes days of theoretical use by extracting the total number of gap days from the total time interval between the first and the last event, accounting for carry over for all medication events within the observation window. Thus, it accounts for timing within the observation window, and excludes the remaining supply at the start of the last event within the observation window.

The formula is

\[
\frac{\text{number of days of theoretical use}}{\text{first-to-last event}}
\]

Observations:

- the `carry.only.for.same.medication` parameter controls the transmission of carry-over across medication changes, producing a "standard" CMA5 (default value is FALSE), and an "alternative" CMA5b, respectively;
- the `consider.dosage.change` parameter controls if dosage changes are taken into account, i.e. if set as TRUE and a new medication event has a different daily dosage recommendation, carry-over is recomputed assuming medication use according to the new prescribed dosage (default value is FALSE).

Value

An S3 object of class CMA5 (derived from CMA0) with the following fields:

- `data` The actual event data, as given by the `data` parameter.
- `id.colname` the name of the column in `data` containing the unique patient ID, as given by the `id.colname` parameter.
- `event.date.colname` the name of the column in `data` containing the start date of the event (in the format given in the `date.format` parameter), as given by the `event.date.colname` parameter.
- `event.duration.colname` the name of the column in `data` containing the event duration (in days), as given by the `event.duration.colname` parameter.
- `event.daily.dose.colname` the name of the column in `data` containing the prescribed daily dose, as given by the `event.daily.dose.colname` parameter.
- `medication.class.colname` the name of the column in `data` containing the classes/types/groups of medication, as given by the `medication.class.colname` parameter.
- `carry.only.for.same.medication` whether the carry-over applies only across medication of the same type, as given by the `carry.only.for.same.medication` parameter.
- `consider.dosage.change` whether the carry-over is adjusted to reflect changes in dosage, as given by the `consider.dosage.change` parameter.
- `followup.window.start` the beginning of the follow-up window, as given by the `followup.window.start` parameter.
- `followup.window.start.unit` the time unit of the `followup.window.start`, as given by the `followup.window.start.unit` parameter.
• followup.window.duration the duration of the follow-up window, as given by the followup.window.duration parameter.

• followup.window.duration.unit the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.

• observation.window.start the beginning of the observation window, as given by the observation.window.start parameter.

• observation.window.start.unit the time unit of the observation.window.start, as given by the observation.window.start.unit parameter.

• observation.window.duration the duration of the observation window, as given by the observation.window.duration parameter.

• observation.window.duration.unit the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.

• date.format the format of the dates, as given by the date.format parameter.

• summary the metadata, as given by the summary parameter.

• event.info the data.frame containing the event info (irrelevant for most users; see compute.event.int.gaps for details).

• cma the data.frame containing the actual CMA estimates for each participant (the ID.colname column).

See Also

CMAs 1 to 8 are defined in:


Examples

cma5 <- CMA5(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
CMA6 constructor.

Description

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 6 object.

Usage

```r
CMA6(data = NULL, ID.colname = NA, event.date.colname = NA,
     event.duration.colname = NA, event.daily.dose.colname = NA,
     medication.class.colname = NA,
     carry.only.for.same.medication = FALSE,
     consider.dosage.change = FALSE, followup.window.start = 0,
     followup.window.start.unit = c("days", "weeks", "months", "years")[1],
     followup.window.duration = 365 * 2,
     followup.window.duration.unit = c("days", "weeks", "months", "years")[1],
     observation.window.start = 0,
     observation.window.start.unit = c("days", "weeks", "months", "years")[1],
     observation.window.duration = 365 * 2,
     observation.window.duration.unit = c("days", "weeks", "months", "years")[1],
     date.format = "%m/%d/%Y", summary = NA,
     event.interval.colname = "event.interval",
     gap.days.colname = "gap.days",
     force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none",
     "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[1],
     parallel.threads = "auto", suppress.warnings = FALSE,
     arguments.that.should.not.beDefined = c(carryover.within.obs.window =
     TRUE, carryover.into.obs.window = FALSE), ...)```

Arguments

- **data**
  
  A `data.frame` containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

- **id.colname**
  
  A `string`, the name of the column in `data` containing the unique patient ID; must be present.

- **event.date.colname**
  
  A `string`, the name of the column in `data` containing the start date of the event (in the format given in the `date.format` parameter); must be present.

- **event.duration.colname**
  
  A `string`, the name of the column in `data` containing the event duration (in days); must be present.
event.daily.dose.colname
A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

medication.class.colname
A string, the name of the column in data containing the medication type, or NA if not defined.

carry.only.for.same.medication
Logical, if TRUE, the carry-over applies only across medication of the same type.

consider.dosage.change
Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.

followup.window.start
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

followup.window.start.unit
can be either “days”, “weeks”, “months” or “years”, and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit
can be either “days”, “weeks”, “months” or “years”, and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit
the definition of the observation window (see the follow-up window parameters above for details).

date.format
A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

summary
Metadata as a string, briefly describing this CMA.

event.interval.colname
A string, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in event.info.

gap.days.colname
A string, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in event.info.
force.NA.CMA.for.failed.patients

Logical describing how the patients for which the CMA estimation fails are treated: if TRUE they are returned with an NA CMA estimate, while for FALSE they are omitted.

parallel.backend

Can be "none" (the default) for single-threaded execution, "multicore" (using mclapply in package parallel) for multicore processing (NB. not currently implemented on MS Windows and automatically falls back on "snow" on this platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or "snow(NWS)" specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package snow for details; the last two require packages Rmpi and nws, respectively, not automatically installed with AdhereR).

parallel.threads

Can be "auto" (for parallel.backend == "multicore", defaults to the number of cores in the system as given by options("cores"), while for parallel.backend == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for parallel.backend == "snow" (see the documentation of package snow for details).

suppress.warnings

Logical, if TRUE don't show any warnings.

arguments.that.should.not.bedefined

a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

other possible parameters

Details

CMA6 assumes that, within the observation window, the medication is used as prescribed and new medication is "banked" until needed (oversupply from previous events is used first, followed new medication supply). It computes days of theoretical use by extracting the total number of gap days from the total time interval between the first event and the end of the observation window, accounting for carry over for all medication events within the observation window. Thus, it accounts for timing within the observation window, and excludes the remaining supply at the end of the observation window.

The formula is

\[
\frac{\text{number of days of theoretical use}}{\text{first event to end of observation window}}
\]

Observations:

- the carry.only.for.same.medication parameter controls the transmission of carry-over across medication changes, producing a "standard" CMA6 (default value is FALSE), and an "alternative" CMA6b, respectively;
- the consider.dosage.change parameter controls if dosage changes are taken into account, i.e. if set as TRUE and a new medication event has a different daily dosage recommendation, carry-over is recomputed assuming medication use according to the new prescribed dosage (default value is FALSE).
Value

An S3 object of class CMA6 (derived from CMA0) with the following fields:

- **data** The actual event data, as given by the data parameter.
- **id. colname** the name of the column in data containing the unique patient ID, as given by the ID. colname parameter.
- **event. date. colname** the name of the column in data containing the start date of the event (in the format given in the date. format parameter), as given by the event. date. colname parameter.
- **event. duration. colname** the name of the column in data containing the event duration (in days), as given by the event. duration. colname parameter.
- **event. daily. dose. colname** the name of the column in data containing the prescribed daily dose, as given by the event. daily. dose. colname parameter.
- **medication. class. colname** the name of the column in data containing the classes/types/groups of medication, as given by the medication. class. colname parameter.
- **carry. only. for. same. medication** whether the carry-over applies only across medication of the same type, as given by the carry. only. for. same. medication parameter.
- **consider. dosage. change** whether the carry-over is adjusted to reflect changes in dosage, as given by the consider. dosage. change parameter.
- **followup. window. start** the beginning of the follow-up window, as given by the followup. window. start parameter.
- **followup. window. start. unit** the time unit of the followup. window. start, as given by the followup. window. start. unit parameter.
- **followup. window. duration** the duration of the follow-up window, as given by the followup. window. duration parameter.
- **followup. window. duration. unit** the time unit of the followup. window. duration, as given by the followup. window. duration. unit parameter.
- **observation. window. start** the beginning of the observation window, as given by the observation. window. start parameter.
- **observation. window. start. unit** the time unit of the observation. window. start, as given by the observation. window. start. unit parameter.
- **observation. window. duration** the duration of the observation window, as given by the observation. window. duration parameter.
- **observation. window. duration. unit** the time unit of the observation. window. duration, as given by the observation. window. duration. unit parameter.
- **date. format** the format of the dates, as given by the date. format parameter.
- **summary** the metadata, as given by the summary parameter.
- **event. info** the data. frame containing the event info (irrelevant for most users; see compute. event. int. Ngaps for details).
- **cma** the data. frame containing the actual CMA estimates for each participant (the ID. colname column).
See Also

CMAs 1 to 8 are defined in:

Examples

cma6 <- CMA6(data=med.events,  
  ID.colname="PATIENT_ID",  
  event.date.colname="DATE",  
  event.duration.colname="DURATION",  
  event.daily.dose.colname="PERDAY",  
  medication.class.colname="CATEGORY",  
  carry.only.for.same.medication=FALSE,  
  consider.dosage.change=FALSE,  
  followup.window.start=30,  
  observation.window.start=30,  
  observation.window.duration=365,  
  date.format="%m/%d/%Y"
);

---

**CMA7 constructor.**

**Description**

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 7 object.

**Usage**

CMA7(data = NULL, ID.colname = NA, event.date.colname = NA,  
  event.duration.colname = NA, event.daily.dose.colname = NA,  
  medication.class.colname = NA,  
  carry.only.for.same.medication = FALSE,  
  consider.dosage.change = FALSE, followup.window.start = 0,  
  followup.window.unit = c("days", "weeks", "months", "years")[1],  
  followup.window.duration = 365 * 2,  
  observation.window.unit = c("days", "weeks", "months", "years")[1], observation.window.duration = 365 * 2,  
  observation.window.duration.unit = c("days", "weeks", "months", "years")[1], date.format = "%m/%d/%Y", summary = NA,  
  event.interval.colname = "event.interval",  
  gap.days.colname = "gap.days",  
  event.interval.unit = c("days", "weeks", "months", "years")[1],  
  observation.interval.duration = 365 * 2,  
  observation.interval.duration.unit = c("days", "weeks", "months", "years")[1],  
  date.format = "%m/%d/%Y",  
  summary = NA)

---
force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[1], parallel.threads = "auto", suppress.warnings = FALSE, arguments.that.should.not.bedefined = c(carryover.within.obs.window = TRUE, carryover.into.obs.window = TRUE), ...)

**Arguments**

**data**
A data.frame containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

**id.colname**
A string, the name of the column in data containing the unique patient ID; must be present.

**event.date.colname**
A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter); must be present.

**event.duration.colname**
A string, the name of the column in data containing the event duration (in days); must be present.

**event.daily.dose.colname**
A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

**medication.class.colname**
A string, the name of the column in data containing the medication type, or NA if not defined.

**carry.only.for.same.medication**
Logical, if TRUE, the carry-over applies only across medication of the same type.

**consider.dosage.change**
Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.

**followup.window.start**
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

**followup.window.start.unit**
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

**followup.window.duration**
either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).
followup.window.duration.unit

- can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit

- the definition of the observation window (see the follow-up window parameters above for details).

date.format

- A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

summary

- Metadata as a string, briefly describing this CMA.

event.interval.colname

- A string, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in event.info.

gap.days.colname

- A string, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in event.info.

force.NA.CMA.for.failed.patients

- Logical describing how the patients for which the CMA estimation fails are treated: if TRUE they are returned with an NA CMA estimate, while for FALSE they are omitted.

parallel.backend

- Can be "none" (the default) for single-threaded execution, "multicore" (using mclapply in package parallel) for multicore processing (NB, not currently implemented on MS Windows and automatically falls back on "snow" on this platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or "snow(NWS)" specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package snow for details; the last two require packages Rmpi and nws, respectively, not automatically installed with AdhereR).

parallel.threads

- Can be "auto" (for parallel.backend == "multicore", defaults to the number of cores in the system as given by options("cores"), while for parallel.backend == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for parallel.backend == "snow" (see the documentation of package snow for details).

suppress.warnings

- Logical, if TRUE don’t show any warnings.

arguments.that.should.not.be.declared

- a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

other possible parameters
Details

CMA7 assumes that, within and before the observation window, the medication is used as prescribed and new medication is "banked" until needed (oversupply from previous events is used first, followed new medication supply). It computes days of theoretical use by extracting the total number of gap days from the total time interval between the start and the end of the observation window, accounting for carry over for all medication events within and before the observation window. All medication events in the follow up window before observation window are considered for carry-over calculation. Thus, it accounts for timing within and before the observation window, and excludes the remaining supply at the end of the observation window.

The formula is

\[
\frac{\text{number of days of theoretical use}}{\text{start to end of observation window}}
\]

Observations:

- the `carry.only.for.same.medication` parameter controls the transmission of carry-over across medication changes, producing a "standard" CMA7 (default value is FALSE), and an "alternative" CMA7b, respectively;
- the `consider.dosage.change` parameter controls if dosage changes are taken into account, i.e. if set as TRUE and a new medication event has a different daily dosage recommendation, carry-over is recomputed assuming medication use according to the new prescribed dosage (default value is FALSE).

Value

An S3 object of class `CMA7` (derived from `CMA0`) with the following fields:

- `data` The actual event data, as given by the `data` parameter.
- `id.colname` The name of the column in `data` containing the unique patient ID, as given by the `id.colname` parameter.
- `event.date.colname` The name of the column in `data` containing the start date of the event (in the format given in the `date.format` parameter), as given by the `event.date.colname` parameter.
- `event.duration.colname` The name of the column in `data` containing the event duration (in days), as given by the `event.duration.colname` parameter.
- `event.daily.dose.colname` The name of the column in `data` containing the prescribed daily dose, as given by the `event.daily.dose.colname` parameter.
- `medication.class.colname` The name of the column in `data` containing the classes/types/groups of medication, as given by the `medication.class.colname` parameter.
- `carry.only.for.same.medication` whether the carry-over applies only across medication of the same type, as given by the `carry.only.for.same.medication` parameter.
- `consider.dosage.change` whether the carry-over is adjusted to reflect changes in dosage, as given by the `consider.dosage.change` parameter.
- `followup.window.start` the beginning of the follow-up window, as given by the `followup.window.start` parameter.
• followup.window.start.unit the time unit of the followup.window.start, as given by the followup.window.start.unit parameter.
• followup.window.duration the duration of the follow-up window, as given by the followup.window.duration parameter.
• followup.window.duration.unit the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.
• observation.window.start the beginning of the observation window, as given by the observation.window.start parameter.
• observation.window.start.unit the time unit of the observation.window.start, as given by the observation.window.start.unit parameter.
• observation.window.duration the duration of the observation window, as given by the observation.window.duration parameter.
• observation.window.duration.unit the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.
• date.format the format of the dates, as given by the date.format parameter.
• summary the metadata, as given by the summary parameter.
• event.info the data.frame containing the event info (irrelevant for most users; see compute.event.int.gaps for details).
• cma the data.frame containing the actual CMA estimates for each participant (the ID.colname column).

Examples

```r
cma8 <- CMA8(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
```

Description

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 8 object.
Usage

CMA8(data = NULL, ID.colname = NA, event.date.colname = NA, 
  event.duration.colname = NA, event.daily.dose.colname = NA, 
  medication.class.colname = NA, 
  carry.only.for.same.medication = FALSE, 
  consider.dosage.change = FALSE, followup.window.start = 0, 
  followup.window.start = c("days", "weeks", "months", "years")[1], 
  followup.window.duration = 365 * 2, 
  followup.window.duration = c("days", "weeks", "months", 
    "years")[1], observation.window.start = 0, 
  observation.window.duration = 365 * 2, 
  observation.window.duration = c("days", "weeks", "months", 
    "years")[1], date.format = "%m/%d/%Y", summary = NA, 
  event.interval.colname = "event.interval", 
  gap.days.colname = "gap.days", 
  force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", 
    "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[1], 
  parallel.threads = "auto", suppress.warnings = FALSE, 
  arguments.that.should.not.be.defined = c(carryover.within.obs.window = 
    TRUE, carryover.into.obs.window = TRUE), ...)

Arguments

data
A data.frame containing the events used to compute the CMA. Must contain,
at a minimum, the patient unique ID, the event date and duration, and might
also contain the daily dosage and medication type (the actual column names are
defined in the following four parameters).

ID.colname
A string, the name of the column in data containing the unique patient ID; must
be present.

event.date.colname
A string, the name of the column in data containing the start date of the event
(in the format given in the date.format parameter); must be present.

event.duration.colname
A string, the name of the column in data containing the event duration (in days);
must be present.

event.daily.dose.colname
A string, the name of the column in data containing the prescribed daily dose,
or NA if not defined.

medication.class.colname
A string, the name of the column in data containing the medication type, or NA
if not defined.

carry.only.for.same.medication
Logical, if TRUE, the carry-over applies only across medication of the same type.

consider.dosage.change
Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.
followup.window.start

If a `Date` object, it represents the actual start date of the follow-up window; if a `string` it is the name of the column in data containing the start date of the follow-up window either as the numbers of `followup.window.start.unit` units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type `Date`); if a `number` it is the number of time units defined in the `followup.window.start.unit` parameter after the begin of the participant's first event; or NA if not defined.

followup.window.start.unit

can be either "days", "weeks", "months" or "years", and represents the time units that `followup.window.start` refers to (when a number), or NA if not defined.

followup.window.duration

either a `number` representing the duration of the follow-up window in the time units given in `followup.window.duration.unit`, or a `string` giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit

can be either "days", "weeks", "months" or "years", and represents the time units that `followup.window.duration` refers to, or NA if not defined.

observation.window.start, observation.window.duration, observation.window.duration.unit

the definition of the observation window (see the follow-up window parameters above for details).

date.format

A `string` giving the format of the dates used in the data and the other parameters; see the `format` parameters of the `as.Date` function for details (NB, this concerns only the dates given as strings and not as `Date` objects).

summary

Metadata as a `string`, briefly describing this CMA.

event.interval.colname

A `string`, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in `event.info`.

gap.days.colname

A `string`, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in `event.info`.

force.NA.CMA.for.failed.patients

`Logical` describing how the patients for which the CMA estimation fails are treated: if TRUE they are returned with an NA CMA estimate, while for FALSE they are omitted.

parallel.backend

Can be "none" (the default) for single-threaded execution, "multicore" (using `mclapply` in package `parallel`) for multicore processing (NB. not currently implemented on MS Windows and automatically falls back on "snow" on this platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
“snow(NWS)” specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package snow for details; the last two require packages rmpi and nws, respectively, not automatically installed with AdhereR).

parallel.threads

Can be “auto” (for parallel.backend == "multicore", defaults to the number of cores in the system as given by options("cores"), while for parallel.backend == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for parallel.backend == "snow" (see the documentation of package snow for details).

suppress.warnings

Logical, if TRUE don’t show any warnings.

arguments.that.should.not.be_defined

a list of argument names and pre-defined values for which a warning should be thrown if passed to the function.

... other possible parameters

Details

CMA8 is similar to CMA6 in that it assumes that, within the observation window, the medication is used as prescribed and new medication is "banked" until needed (oversupply from previous events is used first, followed new medication supply). Unlike CMA6 it accounts for carry-over from before the window - but in a different way from CMA7: by adding a time lag at the start of the observation window equal to the duration of carry-over from before. It is designed for situations when an event with a hypothesized causal effect on adherence happens at the start of the observation window (e.g. enrolment in an intervention study); in this case, it may be that the existing supply is not part of the relationship under study (e.g. it delays the actual start of the study for that participant) and needs to be excluded by shortening the time interval examined. The end of the observation window remains the same. Thus, CMA8 computes days of theoretical use by extracting the total number of gap days from the total time interval between the lagged start and the end of the observation window, accounting for carry over for all medication events within the observation window. All medication events in the follow up window before observation window are considered for carry-over calculation. Thus, as CMA7, it accounts for timing within the observation window, as well as before (different adjustment than CMA7), and excludes the remaining supply at the end of the observation window.

The formula is

\[
\frac{\text{number of days of theoretical use}}{\text{lagged start to end of observation window}}
\]

Observations:

- the carry.only.for.same.medication parameter controls the transmission of carry-over across medication changes, producing a "standard" CMA8 (default value is FALSE), and an "alternative" CMA8b, respectively;
- the consider.dosage.change parameter controls if dosage changes are taken into account, i.e. if set as TRUE and a new medication event has a different daily dosage recommendation, carry-over is recomputed assuming medication use according to the new prescribed dosage (default value is FALSE).
Value

An S3 object of class CMA8 (derived from CMA0) with the following fields:

- **data** The actual event data, as given by the data parameter.
- **ID.colname** the name of the column in data containing the unique patient ID, as given by the ID.colname parameter.
- **event.date.colname** the name of the column in data containing the start date of the event (in the format given in the date.format parameter), as given by the event.date.colname parameter.
- **event.duration.colname** the name of the column in data containing the event duration (in days), as given by the event.duration.colname parameter.
- **event.daily.dose.colname** the name of the column in data containing the prescribed daily dose, as given by the event.daily.dose.colname parameter.
- **medication.class.colname** the name of the column in data containing the classes/types/groups of medication, as given by the medication.class.colname parameter.
- **carry.only.for.same.medication** whether the carry-over applies only across medication of the same type, as given by the carry.only.for.same.medication parameter.
- **consider.dosage.change** whether the carry-over is adjusted to reflect changes in dosage, as given by the consider.dosage.change parameter.
- **followup.window.start** the beginning of the follow-up window, as given by the followup.window.start parameter.
- **followup.window.start.unit** the time unit of the followup.window.start, as given by the followup.window.start.unit parameter.
- **followup.window.duration** the duration of the follow-up window, as given by the followup.window.duration parameter.
- **followup.window.duration.unit** the time unit of the followup.window.duration, as given by the followup.window.duration.unit parameter.
- **observation.window.start** the beginning of the observation window, as given by the observation.window.start parameter.
- **observation.window.start.unit** the time unit of the observation.window.start, as given by the observation.window.start.unit parameter.
- **observation.window.duration** the duration of the observation window, as given by the observation.window.duration parameter.
- **observation.window.duration.unit** the time unit of the observation.window.duration, as given by the observation.window.duration.unit parameter.
- **date.format** the format of the dates, as given by the date.format parameter.
- **summary** the metadata, as given by the summary parameter.
- **event.info** the data.frame containing the event info (irrelevant for most users; see `compute.event.int.gaps` for details).
- **CMA** the data.frame containing the actual CMA estimates for each participant (the ID.colname column).
See Also

CMAs 1 to 8 are defined in:

Examples

```r
cma8 <- CMA8(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
```

---

CMA9

CMA9 constructor.

Description

Constructs a CMA (continuous multiple-interval measures of medication availability/gaps) type 9 object.

Usage

```r
CMA9(data = NULL, ID.colname = NA, event.date.colname = NA, event.duration.colname = NA, event.daily.dose.colname = NA, medication.class.colname = NA, carry.only.for.same.medication = FALSE, consider.dosage.change = FALSE, followup.window.start = 0, followup.window.start.unit = c("days", "weeks", "months", "years")[1], followup.window.duration = 365 * 2, followup.window.duration.unit = c("days", "weeks", "months", "years")[1], observation.window.start = 0, observation.window.start.unit = c("days", "weeks", "months", "years")[1], observation.window.duration = 365 * 2, observation.window.duration.unit = c("days", "weeks", "months", "years")[1], date.format = "%m/%d/%Y", summary = NA, event.interval.colname = "event.interval", gap.days.colname = "gap.days",
```
```r
force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)"))[1], parallel.threads = "auto", suppress.warnings = FALSE,
arguments.that.should.not.be_defined = c(carriage.within.obs.window = TRUE, carriage.into.obs.window = TRUE), ...
```

Arguments

**data**
A data.frame containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

**id.colname**
A string, the name of the column in data containing the unique patient ID; must be present.

**event.date.colname**
A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter); must be present.

**event.duration.colname**
A string, the name of the column in data containing the event duration (in days); must be present.

**event.daily.dose.colname**
A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

**medication.class.colname**
A string, the name of the column in data containing the medication type, or NA if not defined.

**carry.only.for.same.medication**
Logical, if TRUE, the carry-over applies only across medication of the same type.

**consider.dosage.change**
Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.

**followup.window.start**
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

**followup.window.start.unit**
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

**followup.window.duration**
either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).
followup.window.duration.unit
   can be either "days", "weeks", "months" or "years", and represents the time units
that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.end
   the definition of the observation window (see the follow-up window parameters
above for details).

date.format
   A string giving the format of the dates used in the data and the other param-
eters; see the format parameters of the as.Date function for details (NB, this
concerns only the dates given as strings and not as Date objects).

summary
   Metadata as a string, briefly describing this CMA.

event.interval.colname
   A string, the name of a newly-created column storing the number of days be-
tween the start of the current event and the start of the next one; the default
value "event.interval" should be changed only if there is a naming conflict with
a pre-existing "event.interval" column in event.info.

gap.days.colname
   A string, the name of a newly-created column storing the number of days when
medication was not available (i.e., the "gap days"); the default value "gap.days"
should be changed only if there is a naming conflict with a pre-existing "gap.days"
column in event.info.

force.NA.CMA.for.failed.patients
   Logical describing how the patients for which the CMA estimation fails are
treated: if TRUE they are returned with an NA CMA estimate, while for FALSE
they are omitted.

parallel.backend
   Can be "none" (the default) for single-threaded execution, "multicore" (using
mclapply in package parallel) for multicore processing (NB. not currently
implemented on MS Windows and automatically falls back on "snow" on this
platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
"snow(NWS)" specifying various types of SNOW clusters (can be on the local
machine or more complex setups – please see the documentation of package
snow for details; the last two require packages rmpi and nws, respectively, not
automatically installed with AdhereR).

parallel.threads
   Can be "auto" (for parallel.backend == "multicore", defaults to the number of
cores in the system as given by options("cores"), while for parallel.backend
== "snow", defaults to 2), a strictly positive integer specifying the number of
parallel threads, or a more complex specification of the SNOW cluster nodes
for parallel.backend == "snow" (see the documentation of package snow for
details).

suppress.warnings
   Logical, if TRUE don’t show any warnings.

arguments.that.should.not.be.defined
   a list of argument names and pre-defined values for which a warning should be
thrown if passed to the function.

... other possible parameters
Details

CMA9 is similar to CMA7 and CMA8 in that it accounts for carry-over within and before the observation window assuming that new medication is "banked" until needed (oversupply from previous events is used first, followed new medication supply). Yet, unlike these previous CMA8s, it does not assume the medication is used as prescribed; in longitudinal studies with multiple CMA measures, this assumption may introduce additional variation in CMA estimates depending on when the observation window starts in relation to the previous medication event. A shorter time distance from the previous event (and longer to the first event in the observation window) results in higher values even if the number of gap days is the same, and it may also be that the patient has had a similar use pattern for that time interval, rather than perfect adherence followed by no medication use. CMA9 applies a different adjustment: it computes a ratio of days supply over each interval between two prescriptions and considers this applies for each day of that interval, up to 100% (moving oversupply to the next event interval). All medication events in the follow up window before observation window are considered for carry-over calculation. The last interval ends at the end of the follow-up window. Thus, it accounts for timing within the observation window, as well as before (but differently from CMA7 and CMA8), and excludes the remaining supply at the end of the observation window, if any.

The formula is

\[
\frac{\text{number of days in the observation window, each weighted by the ratio of days supply applicable to the interval}}{\text{start to end of observation window}}
\]

Observations:

- the carry.only.for.same.medication parameter controls the transmission of carry-over across medication changes, producing a "standard" CMA7 (default value is FALSE), and an "alternative" CMA7b, respectively;
- the consider.dosage.change parameter controls if dosage changes are taken into account, i.e. if set as TRUE and a new medication event has a different daily dosage recommendation, carry-over is recomputed assuming medication use according to the new prescribed dosage (default value is FALSE).

Value

An S3 object of class CMA9 (derived from CMA8) with the following fields:

- data: The actual event data, as given by the data parameter.
- ID.colname: the name of the column in data containing the unique patient ID, as given by the ID.colname parameter.
- event.date.colname: the name of the column in data containing the start date of the event (in the format given in the date.format parameter), as given by the event.date.colname parameter.
- event.duration.colname: the name of the column in data containing the event duration (in days), as given by the event.duration.colname parameter.
- event.daily.dose.colname: the name of the column in data containing the prescribed daily dose, as given by the event.daily.dose.colname parameter.
- medication.class.colname: the name of the column in data containing the classes/types/groups of medication, as given by the medication.class.colname parameter.
• `carry.only.for.same.medication` whether the carry-over applies only across medication of the same type, as given by the `carry.only.for.same.medication` parameter.

• `consider.dosage.change` whether the carry-over is adjusted to reflect changes in dosage, as given by the `consider.dosage.change` parameter.

• `followup.window.start` the beginning of the follow-up window, as given by the `followup.window.start` parameter.

• `followup.window.start.unit` the time unit of the `followup.window.start`, as given by the `followup.window.start.unit` parameter.

• `followup.window.duration` the duration of the follow-up window, as given by the `followup.window.duration` parameter.

• `followup.window.duration.unit` the time unit of the `followup.window.duration`, as given by the `followup.window.duration.unit` parameter.

• `observation.window.start` the beginning of the observation window, as given by the `observation.window.start` parameter.

• `observation.window.start.unit` the time unit of the `observation.window.start`, as given by the `observation.window.start.unit` parameter.

• `observation.window.duration` the duration of the observation window, as given by the `observation.window.duration` parameter.

• `observation.window.duration.unit` the time unit of the `observation.window.duration`, as given by the `observation.window.duration.unit` parameter.

• `date.format` the format of the dates, as given by the `date.format` parameter.

• `summary` the metadata, as given by the `summary` parameter.

• `event.info` the data frame containing the event info (irrelevant for most users; see `compute.event.int.gaps` for details).

• `cma` the data frame containing the actual CMA estimates for each participant (the ID.colname column).

Examples

```r
  cma9 <- CMA9(data=med.events, 
               ID.colname="PATIENT_ID", 
               event.date.colname="DATE", 
               event.duration.colname="DURATION", 
               event.daily.dose.colname="PERDAY", 
               medication.class.colname="CATEGORY", 
               carry.only.for.same.medication=FALSE, 
               consider.dosage.change=FALSE, 
               followup.window.start=30, 
               observation.window.start=30, 
               observation.window.duration=365, 
               date.format="%m/%d/%Y"
  );
```
Description

Applies a given CMA to each treatment episode and constructs a CMA_per_episode object.

Usage

CMA_per_episode(CMA.to.apply, data, treat.epi = NULL, id.colname = NA,
  event.date.colname = NA, event.duration.colname = NA,
  event.daily.dose.colname = NA, medication.class.colname = NA,
  carry.only.for.same.medication = NA, consider.dosage.change = NA,
  medication.change.means.new.treatment.episode = TRUE,
  dosage.change.means.new.treatment.episode = FALSE,
  maximum.permissible.gap = 180,
  maximum.permissible.gap.unit = c("days", "weeks", "months", "years",
    "percent")[[1]],
  followup.window.start = 0,
  followup.window.start.unit = c("days", "weeks", "months", "years")[[1]],
  followup.window.duration = 365 * 2,
  followup.window.duration.unit = c("days", "weeks", "months", "years")[[1]],
  observation.window.start = 0,
  observation.window.start.unit = c("days", "weeks", "months", "years")[[1]],
  observation.window.duration = 365 * 2,
  observation.window.duration.unit = c("days", "weeks", "months", "years")[[1]],
  date.format = "%m/%d/%Y",
  summary = "CMA per treatment episode",
  event.interval.colname = "event.interval",
  gap.days.colname = "gap.days",
  force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none",
    "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[[1]],
  parallel.threads = "auto", suppress.warnings = FALSE, ...
)

Arguments

CMA.to.apply A string giving the name of the CMA function (1 to 9) that will be computed for each treatment episode.

data A data.frame containing the events (prescribing or dispensing) used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

treat.epi A data.frame containing the treatment episodes. Must contain the patient ID (as given in id.colname), the episode unique ID (increasing sequentially, episode.ID), the episode start date (episode.start), the episode duration in days (episode.duration), and the episode end date (episode.end).
**ID.colname**  
A string, the name of the column in data containing the unique patient ID; must be present.

**event.date.colname**  
A string, the name of the column in data containing the start date of the event (in the format given in the `date.format` parameter); must be present.

**event.duration.colname**  
A string, the name of the column in data containing the event duration (in days); must be present.

**event.daily.dose.colname**  
A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

**medication.class.colname**  
A string, the name of the column in data containing the medication type, or NA if not defined.

**carry.only.for.same.medication**  
Logical, if TRUE, the carry-over applies only across medication of the same type; valid only for CMAs 5 to 9, in which case it is coupled (i.e., the same value is used for computing the treatment episodes and the CMA on each treatment episode).

**consider.dosage.change**  
Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage; valid only for CMAs 5 to 9, in which case it is coupled (i.e., the same value is used for computing the treatment episodes and the CMA on each treatment episode).

**medication.change.means.new.treatment.episode**  
Logical, should a change in medication automatically start a new treatment episode?

**dosage.change.means.new.treatment.episode**  
Logical, should a change in dosage automatically start a new treatment episode?

**maximum.permissible.gap**  
The number of units given by `maximum.permissible.gap.unit` representing the maximum duration of permissible gaps between treatment episodes (can also be a percent, see `maximum.permissible.gap.unit` for details).

**maximum.permissible.gap.unit**  
Can be either "days", "weeks", "months", "years" or "percent", and represents the time units that `maximum.permissible.gap` refers to; if percent, then `maximum.permissible.gap` is interpreted as a percent (can be greater than 100%) of the duration of the current prescription.

**followup.window.start**  
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of `followup.window.start.unit` units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the `followup.window.start.unit` parameter after the begin of the participant’s first event; or NA if not defined.
followup.window.start.unit
can be either "days", "weeks", "months" or "years", and represents the time units
that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
either a number representing the duration of the follow-up window in the time
units given in followup.window.duration.unit, or a string giving the column
containing these numbers. Should represent a period for which relevant
medication events are recorded accurately (e.g. not extend after end of relevant
treatment, loss-to-follow-up or change to a health care provider not covered by
the database).

followup.window.duration.unit
can be either "days", "weeks", "months" or "years", and represents the time units
that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit
the definition of the observation window (see the follow-up window parameters
above for details).

date.format
A string giving the format of the dates used in the data and the other param-
eters; see the format parameters of the as.Date function for details (NB, this
concerns only the dates given as strings and not as Date objects).

summary
Metadata as a string, briefly describing this CMA.

event.interval.colname
A string, the name of a newly-created column storing the number of days be-
tween the start of the current event and the start of the next one; the default
value "event.interval" should be changed only if there is a naming conflict with
a pre-existing "event.interval" column in event.info.

gap.days.colname
A string, the name of a newly-created column storing the number of days when
medication was not available (i.e., the "gap days"); the default value "gap.days"
should be changed only if there is a naming conflict with a pre-existing "gap.days"
column in event.info.

force.NA.CMA.for.failed.patients
Logical describing how the patients for which the CMA estimation fails are
treated: if TRUE they are returned with an NA CMA estimate, while for FALSE
they are omitted.

parallel.backend
Can be "none" (the default) for single-threaded execution, "multicore" (using
mclapply in package parallel) for multicore processing (NB. not currently
implemented on MS Windows and automatically falls back on "snow" on this
platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
"snow(NWS)" specifying various types of SNOW clusters (can be on the local
machine or more complex setups – please see the documentation of package
snow for details; the last two require packages Rmpi and nws, respectively, not
automatically installed with AdhereR).

parallel.threads
Can be "auto" (for parallel.backend == "multicore", defaults to the number of
cores in the system as given by options("cores"), while for parallel.backend
== "snow", defaults to 2), a strictly positive integer specifying the number of
parallel threads, or a more complex specification of the SNOW cluster nodes
for parallel.backend == "snow" (see the documentation of package snow for
details).

suppress.warnings
Logical, if TRUE don’t show any warnings.

... other possible parameters

Details

CMA_per_episode first identifies the treatment episodes for the whole follo-up window (using the
compute.treatment.episodes function), and then computes the given "simple" CMA for each
treatment episode that intersects with the observation window. NB: the CMA is computed for the
period of the episode that is part of the observations window; thus, if an episode starts earlier or
ends later than the observation window, CMA will be computed for a section of that episode. Thus,
as opposed to the "simple" CMAs 1 to 9, it returns a set of CMAs, with possibly more than one
element.

It is highly similar to CMA_sliding_window which computes a CMA for a set of sliding windows.

Value

An S3 object of class CMA_per_episode with the following fields:

• data The actual event data, as given by the data parameter.
• ID.colname the name of the column in data containing the unique patient ID, as given by the
  ID.colname parameter.
• event.date.colname the name of the column in data containing the start date of the event
  (in the format given in the date.format parameter), as given by the event.date.colname
  parameter.
• event.duration.colname the name of the column in data containing the event duration (in
days), as given by the event.duration.colname parameter.
• event.daily.dose.colname the name of the column in data containing the prescribed daily
dose, as given by the event.daily.dose.colname parameter.
• medication.class.colname the name of the column in data containing the classes/types/groups
  of medication, as given by the medication.class.colname parameter.
• carry.only.for.same.medication whether the carry-over applies only across medication
  of the same type, as given by the carry.only.for.same.medication parameter.
• consider.dosage.change whether the carry-over is adjusted to reflect changes in dosage, as
given by the consider.dosage.change parameter.
• followup.window.start the beginning of the follow-up window, as given by the followup.window.start
  parameter.
• followup.window.start.unit the time unit of the followup.window.start, as given by
  the followup.window.start.unit parameter.
• followup.window.duration the duration of the follow-up window, as given by the followup.window.duration
  parameter.
• followup.window.duration.unit the time unit of the followup.window.duration, as given
  by the followup.window.duration.unit parameter.
• observation_window.start the beginning of the observation window, as given by the observation_window.start parameter.
• observation_window.unit the time unit of the observation_window.start, as given by the observation_window.start.unit parameter.
• observation_window.duration the duration of the observation window, as given by the observation_window.duration parameter.
• observation_window.duration.unit the time unit of the observation_window.duration, as given by the observation_window.duration.unit parameter.
• date.format the format of the dates, as given by the date.format parameter.
• summary the metadata, as given by the summary parameter.
• event.info the data.frame containing the event info (irrelevant for most users; see compute.event.int.gaps for details).
• computed.CMA the class name of the computed CMA.
• cma the data.frame containing the actual CMA estimates for each participant (the ID.colname column) and treatment episode, with columns:
  – ID.colname the patient ID as given by the ID.colname parameter.
  – episode.ID the unique treatment episode ID (within patients).
  – episode.start the treatment episode’s start date (as a Date object).
  – end.episode.gap.days the corresponding gap days of the last event in this episode.
  – episode.duration the treatment episode’s duration in days.
  – episode.end the treatment episode’s end date (as a Date object).
  – CMA the treatment episode’s estimated CMA.

See Also

CMA_sliding_window is very similar, computing a "simple" CMA for each of a set of same-size sliding windows. The "simple" CMAs that can be computed comprise CMA1, CMA2, CMA3, CMA4, CMA5, CMA6, CMA7, CMA8, CMA9, as well as user-defined classes derived from CMA0 that have a CMA component giving the estimated CMA per patient as a data.frame.

Examples

```r
## Not run:
cmaE <- CMA_per_episode(CMA="CMA1",
data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=0,
  observation.window.start=0,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
```

## End(Not run)
CMA_sliding_window (CMA.to.apply, data, ID.colname = NA, 
  event.date.colname = NA, event.duration.colname = NA, 
  event.daily.dose.colname = NA, medication.class.colname = NA, 
  carry.only.for.same.medication = NA, consider.dosage.change = NA, 
  followup.window.start = 0, followup.window.start.unit = c("days", 
  "weeks", "months", "years")[[1]], followup.window.duration = 365 * 2, 
  followup.window.duration.unit = c("days", "weeks", "months", 
  "years")[[1]], observation.window.start = 0, 
  observation.window.start.unit = c("days", "weeks", "months", 
  "years")[[1]], observation.window.duration = 365 * 2, 
  observation.window.duration.unit = c("days", "weeks", "months", 
  "years")[[1]], sliding.window.start = 0, 
  sliding.window.start.unit = c("days", "weeks", "months", "years")[[1]], 
  sliding.window.duration = 90, 
  sliding.window.duration.unit = c("days", "weeks", "months", "years")[[1]], 
  sliding.window.step.duration = 30, 
  sliding.window.step.unit = c("days", "weeks", "months", "years")[[1]], 
  sliding.window.no.steps = NA, date.format = "%m/%d/%y", 
  summary = "CMA per sliding window", 
  event.interval.colname = "event.interval", 
  gap.days.colname = "gap.days", 
  force.NA.CMA.for.failed.patients = TRUE, parallel.backend = c("none", 
  "multicore", "snow", "snow(SOCK)", "snow(MPI)", "snow(NWS)")[[1]], 
  parallel.threads = "auto", suppress.warnings = FALSE, ...
)

**Arguments**

- **CMA.to.apply**  
  A string giving the name of the CMA function (1 to 9) that will be computed for each treatment episode.

- **data**  
  A data.frame containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters).

- **ID.colname**  
  A string, the name of the column in data containing the unique patient ID; must be present.

- **event.date.colname**  
  A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter); must be present.
event.duration.colname

A string, the name of the column in data containing the event duration (in days); must be present.

event.daily.dose.colname

A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

medication.class.colname

A string, the name of the column in data containing the medication type, or NA if not defined.

carry.only.for.same.medication

Logical, if TRUE, the carry-over applies only across medication of the same type.

consider.dosage.change

Logical, if TRUE, the carry-over is adjusted to also reflect changes in dosage.

followup.window.start

If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant’s first event; or NA if not defined.

followup.window.start.unit

can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration

either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit

can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit

the definition of the observation window (see the follow-up window parameters above for details).

sliding.window.start, sliding.window.start.unit, sliding.window.duration, sliding.window.duration.unit

the definition of the first sliding window (see the follow-up window parameters above for details).

sliding.window.step.duration, sliding.window.step.unit

if not missing (NA), these give the step (or "jump") to the right of the sliding window in time units.

sliding.window.no.steps

a integer specifying the desired number of sliding windows to cover the observation window (if possible); trumps sliding.window.step.duration and sliding.window.step.unit.
 Metadata as a string, briefly describing this CMA.

event.interval.colname

A string, the name of a newly-created column storing the number of days between the start of the current event and the start of the next one; the default value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in event.info.

gap.days.colname

A string, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in event.info.

force.NA.CMA.for.failed.patients

Logical describing how the patients for which the CMA estimation fails are treated: if TRUE they are returned with an NA CMA estimate, while for FALSE they are omitted.

parallel.backend

Can be "none" (the default) for single-threaded execution, "multicore" (using mclapply in package parallel) for multicore processing (NB. not currently implemented on MS Windows and automatically falls back on "snow" on this platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or "snow(NWS)" specifying various types of SNOW clusters (can be on the local machine or more complex setups – please see the documentation of package snow for details; the last two require packages Rmpi and nws, respectively, not automatically installed with AdhereR).

parallel.threads

Can be "auto" (for parallel.backend == "multicore", defaults to the number of cores in the system as given by options("cores"), while for parallel.backend == "snow", defaults to 2), a strictly positive integer specifying the number of parallel threads, or a more complex specification of the SNOW cluster nodes for parallel.backend == "snow" (see the documentation of package snow for details).

suppress.warnings

Logical, if TRUE don’t show any warnings.

... other possible parameters

Details

CMA_sliding_window first computes a set of fixed-size (possibly partly overlapping) sliding windows, each sliding to the right by a fixed timelag, and then, for each of them, it computes the given "simple" CMA. Thus, as opposed to the "simple" CMAs 1 to 9, it returns a set of CMAs, with possibly more than one element.

It is highly similar to CMA_per_episode which computes a CMA for a set of treatment episodes.
An S3 object of class `CMA_sliding_window` with the following fields:

- **data** The actual event data, as given by the `data` parameter.
- **id.colname** the name of the column in data containing the unique patient ID, as given by the `id.colname` parameter.
- **event.date.colname** the name of the column in data containing the start date of the event (in the format given in the `date.format` parameter), as given by the `event.date.colname` parameter.
- **event.duration.colname** the name of the column in data containing the event duration (in days), as given by the `event.duration.colname` parameter.
- **event.daily.dose.colname** the name of the column in data containing the prescribed daily dose, as given by the `event.daily.dose.colname` parameter.
- **medication.class.colname** the name of the column in data containing the classes/types/groups of medication, as given by the `medication.class.colname` parameter.
- **carry.only.for.same.medication** whether the carry-over applies only across medication of the same type, as given by the `carry.only.for.same.medication` parameter.
- **consider.dosage.change** whether the carry-over is adjusted to reflect changes in dosage, as given by the `consider.dosage.change` parameter.
- **followup.window.start** the beginning of the follow-up window, as given by the `followup.window.start` parameter.
- **followup.window.start.unit** the time unit of the `followup.window.start`, as given by the `followup.window.start.unit` parameter.
- **followup.window.duration** the duration of the follow-up window, as given by the `followup.window.duration` parameter.
- **followup.window.duration.unit** the time unit of the `followup.window.duration`, as given by the `followup.window.duration.unit` parameter.
- **observation.window.start** the beginning of the observation window, as given by the `observation.window.start` parameter.
- **observation.window.start.unit** the time unit of the `observation.window.start`, as given by the `observation.window.start.unit` parameter.
- **observation.window.duration** the duration of the observation window, as given by the `observation.window.duration` parameter.
- **observation.window.duration.unit** the time unit of the `observation.window.duration`, as given by the `observation.window.duration.unit` parameter.
- **date.format** the format of the dates, as given by the `date.format` parameter.
- **summary** the metadata, as given by the `summary` parameter.
- **event.info** the data frame containing the event info (irrelevant for most users; see `compute.event.int.gaps` for details).
- **computed.CMA** the class name of the computed CMA.
- **cma** the data frame containing the actual CMA estimates for each participant (the `id.colname` column) and sliding window, with columns:
compute.event.int.gaps

- ID.colname the patient ID as given by the ID.colname parameter.
- window.ID the unique window ID (within patients).
- window.start the window’s start date (as a Date object).
- window.end the window’s end date (as a Date object).
- CMA the window’s estimated CMA.

See Also

CMA_per_episode is very similar, computing a "simple" CMA for each of the treatment episodes. The "simple" CMAs that can be computed comprise CMA1, CMA2, CMA3, CMA4, CMA5, CMA6, CMA7, CMA8, CMA9, as well as user-defined classes derived from CMA0 that have a CMA component giving the estimated CMA per patient as a data.frame.

Examples

```r
## Not run:
cmaW <- CMA_sliding_window(CMA="CMA1",
data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
carry.only.for.same.medication=FALSE,
consider.dosage.change=FALSE,
followup.window.start=0,
obervation.window.start=0,
obervation.window.duration=365,
sliding.window.start=0,
sliding.window.start.unit="days",
sliding.window.duration=90,
sliding.window.duration.unit="days",
sliding.window.step.duration=7,
sliding.window.step.unit="days",
sliding.window.no.steps=NA,
date.format="%m/%d/%Y"
);
## End(Not run)
```

---

compute.event.int.gaps

*Gap Days and Event (prescribing or dispensing) Intervals.*

Description

For a given event (prescribing or dispensing) database, compute the gap days and event intervals in various scenarious.
Usage

compute.event.int.gaps(data, ID.colname = NA, event.date.colname = NA, 
                      event.duration.colname = NA, event.daily.dose.colname = NA, 
                      medication.class.colname = NA, 
                      event.interval.colname = "event.interval", 
                      gap.days.colname = "gap.days", carryover.within.obs.window = FALSE, 
                      carryover.into.obs.window = FALSE, 
                      carry.only.for.same.medication = FALSE, 
                      consider.dosage.change = FALSE, followup.window.start = 0, 
                      followup.window.start.unit = c("days", "weeks", "months", "years")[1], 
                      followup.window.duration = 365 * 2, 
                      followup.window.duration.unit = c("days", "weeks", "months", 
                      "years")[1], observation.window.start = 0, 
                      observation.window.start.unit = c("days", "weeks", "months", 
                      "years")[1], observation.window.duration = 365 * 2, 
                      observation.window.duration.unit = c("days", "weeks", "months", 
                      "years")[1], date.format = "%d/%m/%Y", 
                      keep.window.start.end.dates = FALSE, 
                      remove.events.outside.followup.window = TRUE, 
                      keep.event.interval.for.all.events = FALSE, 
                      parallel.backend = c("none", "multicore", "snow", "snow(SOCK)", 
                      "snow(MPI)", "snow(NWS)")[1], parallel.threads = "auto", 
                      suppress.warnings = FALSE, return.data.table = FALSE, ...)

Arguments

data A data.frame containing the events used to compute the CMA. Must contain, 
at a minimum, the patient unique ID, the event date and duration, and might also 
contain the daily dosage and medication type (the actual column names are 
defined in the following four parameters); the CMA constructors call this parameter 
data.

ID.colname A string, the name of the column in data containing the unique patient ID; must 
be present.

event.date.colname A string, the name of the column in data containing the start date of the event 
(in the format given in the date.format parameter); must be present.

event.duration.colname A string, the name of the column in data containing the prescribed daily dose, 
or NA if not defined.

medication.class.colname A string, the name of the column in data containing the classes/types/groups of 
medication, or NA if not defined.

event.interval.colname A string, the name of a newly-created column storing the number of days be- 
tween the start of the current event and the start of the next one; the default
value "event.interval" should be changed only if there is a naming conflict with a pre-existing "event.interval" column in event.info.

gap.days.colname
A string, the name of a newly-created column storing the number of days when medication was not available (i.e., the "gap days"); the default value "gap.days" should be changed only if there is a naming conflict with a pre-existing "gap.days" column in event.info.

carryover.within.obs.window
Logical, if TRUE consider the carry-over within the observation window, or NA if not defined.

carryover.into.obs.window
Logical, if TRUE consider the carry-over from before the starting date of the observation window, or NA if not defined.

carry.only.for.same.medication
Logical, if TRUE the carry-over applies only across medication of the same type, or NA if not defined.

consider.dosage.change
Logical, if TRUE the carry-over is adjusted to reflect changes in dosage, or NA if not defined.

followup.window.start
If a Date object, it represents the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window either as the numbers of followup.window.start.unit units after the first event (the column must be of type numeric) or as actual dates (in which case the column must be of type Date); if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event.

followup.window.start.unit
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
either a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or a string giving the column containing these numbers. Should represent a period for which relevant medication events are recorded accurately (e.g. not extend after end of relevant treatment, loss-to-follow-up or change to a health care provider not covered by the database).

followup.window.duration.unit
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.duration refers to, or NA if not defined.

observation.window.start, observation.window.start.unit, observation.window.duration, observation.window.duration.unit
the definition of the observation window (see the follow-up window parameters above for details).

date.format
A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).
keep.window.start.end.dates
  Logical, should the computed start and end dates of the windows be kept?

remove.events.outside.followup.window
  Logical, should the events that fall outside the follow-up window be removed
  from the results?

keep.event.interval.for.all.events
  Logical, should the computed event intervals be kept for all events, or NA’ed
  for those outside the OW?

parallel.backend
  Can be "none" (the default) for single-threaded execution, "multicore" (using
  mclapply in package parallel) for multicore processing (NB. not currently
  implemented on MS Windows and automatically falls back on "snow" on this
  platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
  "snow(NWS)" specifying various types of SNOW clusters (can be on the local
  machine or more complex setups – please see the documentation of package
  snow for details; the last two require packages Rmpi and nws, respectively, not
  automatically installed with AdhereR).

parallel.threads
  Can be "auto" (for parallel.backend == "multicore", defaults to the number of
  cores in the system as given by options("cores"), while for parallel.backend
  == "snow", defaults to 2), a strictly positive integer specifying the number of
  parallel threads, or a more complex specification of the SNOW cluster nodes
  for parallel.backend == "snow" (see the documentation of package snow for
  details).

suppress.warnings
  Logical, if TRUE don’t show any warnings.

return.data.table
  Logical, if TRUE return a data.table object, otherwise a data.frame.

... extra arguments.

Details

This should in general not be called directly by the user, but is provided as a basis for the extension
  to new CMAs.

Value

A data.frame or data.table extending the event.info parameter with:

- event.interval Or any other name given in event.interval.colname, containing the
  number of days between the start of the current event and the start of the next one.
- gap.days Or any other name given in gap.days.colname, containing the number of days
  when medication was not available for the current event (i.e., the "gap days").
- .FU.START.DATE, .FU.END.DATE if kept, the actual start and end dates of the follow-up win-
  dow (after adjustments due to the various parameters).
- .OBS.START.DATE, .OBS.END.DATE if kept, the actual start and end dates of the observation
  window (after adjustments due to the various parameters).
compute.treatment.episodes

Compute Treatment Episodes.

Description

For a given event (prescribing or dispensing) database, compute the treatment episodes for each patient in various scenarios.

Usage

```r
compute.treatment.episodes(data, ID.colname = NA,
 event.date.colname = NA, event.duration.colname = NA,
 event.daily.dose.colname = NA, medication.class.colname = NA,
 carryover.within.obs.window = TRUE,
 carry.only.for.same.medication = TRUE, consider.dosage.change = TRUE,
 medication.change.means.new.treatment.episode = TRUE,
 dosage.change.means.new.treatment.episode = FALSE,
 maximum.permissible.gap = 90,
 maximum.permissible.gap.unit = c("days", "weeks", "months", "years",
 "percent")[1], followup.window.start = 0,
 followup.window.start.unit = c("days", "weeks", "months", "years")[1],
 followup.window.duration = 365 * 2,
 followup.window.duration.unit = c("days", "weeks", "months", "years")[1],
 event.interval.colname = "event.interval",
 gap.days.colname = "gap.days", date.format = "%m/%d/%Y",
 parallel.backend = c("none", "multicore", "snow", "snow(SOCK)",
 "snow(MPI)", "snow(NWS)")[1], parallel.threads = "auto",
 suppress.warnings = FALSE, return.data.table = FALSE, ...)
```

Arguments

data

A data.frame containing the events used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters); the CMA constructors call this parameter data.
ID.colname A string, the name of the column in data containing the unique patient ID, or NA if not defined.

event.date.colname
A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter), or NA if not defined.

event.duration.colname
A string, the name of the column in data containing the event duration (in days), or NA if not defined.

event.daily.dose.colname
A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

medication.class.colname
A string, the name of the column in data containing the classes/types/groups of medication, or NA if not defined.

carryover.within.obs.window
Logical, if TRUE consider the carry-over within the observation window, or NA if not defined.

carry.only.for.same.medication
Logical, if TRUE the carry-over applies only across medication of the same type, or NA if not defined.

consider.dosage.change
Logical, if TRUE the carry-over is adjusted to reflect changes in dosage, or NA if not defined.

medication.change.means.new.treatment.episode
Logical, should a change in medication automatically start a new treatment episode?

dosage.change.means.new.treatment.episode
Logical, should a change in dosage automatically start a new treatment episode?

maximum.permissible.gap
The number of units given by maximum.permissible.gap.unit representing the maximum duration of permissible gaps between treatment episodes (can also be a percent, see maximum.permissible.gap.unit for details).

maximum.permissible.gap.unit
can be either "days", "weeks", "months" or "percent", and represents the time units that maximum.permissible.gap refers to; if percent, then maximum.permissible.gap is interpreted as a percent (can be greater than 100%) of the duration of the current prescription.

followup.window.start
If a Date object it is the actual start date of the follow-up window; if a string it is the name of the column in data containing the start date of the follow-up window; if a number it is the number of time units defined in the followup.window.start.unit parameter after the begin of the participant's first event; or NA if not defined.

followup.window.start.unit
can be either "days", "weeks", "months" or "years", and represents the time units that followup.window.start refers to (when a number), or NA if not defined.

followup.window.duration
a number representing the duration of the follow-up window in the time units given in followup.window.duration.unit, or NA if not defined.
followup.window.duration.unit
  can be either "days", "weeks", "months" or "years", and represents the time units
  that followup.window.duration refers to, or NA if not defined.

event.interval.colname
  A string, the name of a newly-created column storing the number of days be-
  tween the start of the current event and the start of the next one; the default
  value "event.interval" should be changed only if there is a naming conflict with
  a pre-existing "event.interval" column in event.info.

gap.days.colname
  A string, the name of a newly-created column storing the number of days when
  medication was not available (i.e., the "gap days"); the default value "gap.days"
  should be changed only if there is a naming conflict with a pre-existing "gap.days"
  column in event.info.

date.format
  A string giving the format of the dates used in the data and the other param-
  eters; see the format parameters of the as.Date function for details (NB, this
  concerns only the dates given as strings and not as Date objects).

parallel.backend
  Can be "none" (the default) for single-threaded execution, "multicore" (using
  mclapply in package parallel) for multicore processing (NB, not currently
  implemented on MS Windows and automatically falls back on "snow" on this
  platform), or "snow", "snow(SOCK)" (equivalent to "snow"), "snow(MPI)" or
  "snow(NWS)" specifying various types of SNOW clusters (can be on the local
  machine or more complex setups – please see the documentation of package
  snow for details; the last two require packages Rmpi and nws, respectively, not
  automatically installed with AdhereR).

parallel.threads
  Can be "auto" (for parallel.backend == "multicore", defaults to the number of
  cores in the system as given by options("cores"), while for parallel.backend
  == "snow", defaults to 2), a strictly positive integer specifying the number of
  parallel threads, or a more complex specification of the SNOW cluster nodes
  for parallel.backend == "snow" (see the documentation of package snow for
  details).

suppress.warnings
 Logical, if TRUE don’t show any warnings.

return.data.table
  Logical, if TRUE return a data.table object, otherwise a data.frame.

... extra arguments.

Details

This should in general not be called directly by the user, but is provided as a basis for the extension

to new CMAs.

For the last treatment episode, the gap is considered only when longer than the maximum permissi-

ble gap. Please note the following:

  • episode starts at first medication event for a particular medication,
• episode ends on the day when the last supply of that medication finished or if a period longer than the permissible gap preceded the next medication event, or at the end of the FUW,

• end episode gap days represents either the number of days after the end of the treatment episode (if medication changed, or if a period longer than the permissible gap preceded the next medication event) or at the end of (and within) the episode, i.e. the number of days after the last supply finished (if no other medication event followed until the end of the FUW),

• the duration of the episode is the interval between the episode start and episode end (and may include the gap days at the end, in the latter condition described above),

• the number of gap days after the end of the episode can be computed as all values larger than the permissible gap and 0 otherwise,

• if medication change starts new episode, then previous episode ends when the last supply is finished (irrespective of the length of gap compared to a maximum permissible gap); any days before the date of the new medication supply are considered a gap; this maintains consistency with gaps between episodes (whether they are constructed based on the maximum permissible gap rule or the medication change rule).

Value

A `data.frame` or `data.table` with the following columns (or NULL if no treatment episodes could be computed):

• `patid` the patient ID.
• `episode.id` the episode unique ID (increasing sequentially).
• `episode.start` the episode start date.
• `end.episode.gap.days` the corresponding gap days of the last event in this episode.
• `episode.duration` the episode duration in days.
• `episode.end` the episode end date.

---

`compute_event_durations`

Computation of event durations.

Description

Computes event durations based on dispensing, prescription, and other data (e.g. hospitalization data) and returns a `data.frame` which can be used with the CMA constructors in `AdhereR`.

Usage

```r
compute_event_durations(disp.data = NULL, presc.data = NULL, special.periods.data = NULL, ID.colname, medication.class.colnames, disp.date.colname, total.dose.colname, presc.date.colname, presc.daily.dose.colname, presc.duration.colname, visit.colname, split.on.dosage.change = TRUE, force.init.presc = FALSE, force.presc.renew = FALSE, trt.interruption = c("continue"),
```

```r
```
"discard", "carryover")][1], special.periods.method = trt.interruption,
date.format = "%d.%m.%Y", suppress.warnings = FALSE,
return.data.table = FALSE, progress.bar = TRUE, ...)

Arguments

disp.data  A data.frame or data.table containing the dispensing events. Must contain,
at a minimum, the patient unique ID, one medication identifier, the dispens-
ing date, and total dispensed dose, and might also contain additional columns
to identify and group medications (the actual column names are defined in the
medication.class.colnames parameter).

presc.data  A data.frame containing the prescribing events. Must contain, at a minimum,
the same unique patient ID and medication identifier(s) as the dispensing data,
the prescription date, the daily prescribed dose, and the prescription duration.
Optionally, it might also contain a visit number.

special.periods.data  Optional, NULL or a data.frame containing the information about special peri-
periods (e.g., hospitalizations or other situations where medication use may differ,
e.g. during incarcerations or holidays). Must contain the same unique patient
ID as dispensing and prescription data, the start and end dates of the special pe-
riods with the exact column names DATE.IN and DATE.OUT. Optional columns
are TYPE (indicating the type of special situation), customized instructions how
to handle a specific period (see special.periods.method), and any of those
specified in medication.class.colnames.

ID.colname  A string, the name of the column in disp.data, presc.data, and special.periods.data
containing the unique patient ID.

medication.class.colnames  A Vector of strings, the name(s) of the column(s) in disp.data and presc.data
containing the classes/types/groups of medication.

disp.date.colname  A string, the name of the column in disp.data containing the dispensing date
(in the format given in the date.format parameter).

total.dose.colname  A string, the name of the column in disp.data containing the total dispensed
dose as numeric (e.g. 500 for 10 tablets of 50 mg).

presc.date.colname  A string, the name of the column in presc.data containing the prescription
date (in the format given in the date.format parameter).

presc.daily.dose.colname  A string, the name of the column in presc.data containing the daily prescribed
dose as numeric (e.g. 50 for 50 mg once per day, or 25 for 50 mg once ever 2
days).

presc.duration.colname  A string, the name of the column in presc.data containing the duration of the
prescription as numeric or NA if duration is unknown.
compute_event_durations

visit.colname  A string, the name of the column in presc.data containing the number of the visit or a new column name if the prescribing data does not contain such a column.

split.on.dosage.change
Logical or string. If TRUE split the dispensing event on days with dosage change and create a new event with the new dosage for the remaining supply. If string, the name of the column containing the Logical in disp.data for each medication class separately. Important if carryover should be considered later on.

force.init.presc
Logical. If TRUE advance the date of the first prescription event to the date of the first dispensing event, if the first prescription event is after the first dispensing event for a specific medication. Only if the first prescription event is not limited in duration (as indicated in the presc.duration.colname).

force.presc.renew
Logical or string. If TRUE require a new prescription for all medications for every prescription event (visit), otherwise prescriptions end on the first visit without renewal. If string, the name of the column in disp.data containing the Logical for each medication class separately.

trt.interruption
  can be either of "continue", "discard", "carryover", or a string. It indicates how to handle durations during treatment interruptions (see special.periods.method). If string, the name of the (character) column in disp.data containing the information ("continue", "discard", or "carryover") for each medication class separately.

special.periods.method
  can be either of continue, discard, carryover, or custom. It indicates how to handle durations during special periods. With continue, special periods have no effect on durations and event start dates. With discard, durations are truncated at the beginning of special periods and the remaining quantity is discarded. With carryover, durations are truncated at the beginning of a special period and a new event with the remaining duration is created after the end of the special period. With custom, the mapping has to be included in special.periods.data.

date.format
  A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

suppress.warnings
  Logical, if TRUE don’t show any warnings.

return.data.table
  Logical, if TRUE return a data.table object, otherwise a data.frame.

progress.bar
  Logical, if TRUE show a progress bar.

...  other possible parameters.

Details

Computation of CMAs requires a supply duration for medications dispensed to patients. If medications are not supplied for fixed durations but as a quantity that may last for various durations based
on the prescribed dose, the supply duration has to be calculated based on dispensed and prescribed doses. Treatments may be interrupted and resumed at later times, for which existing supplies may or may not be taken into account. Patients may be hospitalized or incarcerated, and may not use their own supplies during these periods. This function calculates supply durations, taking into account the aforementioned situations and providing various parameters for flexible adjustments.

**Value**

A list with the following elements:

- **event_durations**: A data.table or data.frame with the following columns:
  - `ID.colname` the unique patient ID, as given by the `ID.colname` parameter.
  - `medication.class.colnames` the column(s) with classes/types/groups of medication, as given by the `medication.class.colnames` parameter.
  - `disp.date.colname` the date of the dispensing event, as given by the `disp.date.colname` parameter.
  - `total.dose.colname` the total dispensed dose, as given by the `total.dose.colname` parameter.
  - `presc.daily.dose.colname` the prescribed daily dose, as given by the `presc.daily.dose.colname` parameter.
  - `DISP.START` the start date of the dispensing event, either the same as in `disp.date.colname` or a later date in case of dosage changes or treatment interruptions/hospitalizations.
  - `DURATION` the calculated duration of the supply, based on the total dispensed dose and the prescribed daily dose, starting from the `DISP.START` date.
  - `episode.start`: the start date of the current prescription episode.
  - `episode.end`: the end date of the current prescription episode. Can be before the start date of the dispensing event if dispensed during a treatment interruption.
  - `SPECIAL.DURATION` the number of days during the current duration affected by special durations or treatment interruptions of type "continue".
  - `CARRYOVER.DURATION` the number of days after the current duration affected by special durations or treatment interruptions of type "carryover".
  - `tot.presc.interruptions` the total number of prescription interruptions per patient for a specific medication.
  - `tot.dosage.changes` the total number of dosage changes per patient for a specific medication.

- **prescription_episodes**: A data.table or data.frame with the following columns:
  - `ID.colname`: the unique patient ID, as given by the `ID.colname` parameter.
  - `medication.class.colnames`: the column(s) with classes/types/groups of medication, as given by the `medication.class.colnames` parameter.
  - `presc.daily.dose.colname`: the prescribed daily dose, as given by the `presc.daily.dose.colname` parameter.
  - `episode.start`: the start date of the prescription episode.
  - `episode.duration`: the duration of the prescription episode in days.
  - `episode.end`: the end date of the prescription episode.

- **special_periods**: A data.table or data.frame, the `special.periods.data` with an additional column `SPECIAL.DURATION`: the number of days between `DATE.IN` and `DATE.OUT`
• ID.colname the name of the columns containing the unique patient ID, as given by the ID.colname parameter.

• medication.class.colnames the name(s) of the column(s) in disp. data and presc. data containing the classes/types/groups of medication, as given by the medication.class.colnames parameter.

• disp.date.colname the name of the column in disp. data containing the dispensing date, as given in the disp.date.colname parameter.

• total.dose.colname the name of the column in disp. data containing the total dispensed dose, as given by the total.dose.colname parameter.

• presc.date.colname the name of the column in presc. data containing the prescription date, as given in the presc.date.colname parameter.

• presc.daily.dose.colname the name of the column in presc. data containing the daily prescribed dose, as given by the presc.daily.dose.colname parameter.

• presc.duration.colname the name of the column in presc. data containing the duration of the prescription, as given by the presc.duration.colname parameter.

• visit.colname the name of the column containing the number of the visit, as given by the visit.colname parameter

• split.on.dosage.change whether to split the dispensing event on days with dosage changes and create a new event with the new dosage for the remaining supply, as given by the split.on.dosage.change parameter.

• force.init.presc whether the date of the first prescription event was set back to the date of the first dispensing event, when the first prescription event was after the first dispensing event for a specific medication, as given by the force.init.presc parameter.

• force.presc.renew whether a new prescription was required for all medications for every prescription event (visit), as given by the force.presc.renew parameter.

• trt.interruption how durations during treatment interruptions were handled, as given by the trt.interruption parameter.

• special.periods.method as given by the special.periods.method parameter.

• date.format the format of the dates, as given by the date.format parameter.

Examples

```r
## Not run:
event_durations <- compute_event_durations(disp.data = durcomp.dispensing,
presc.data = durcomp.prescribing,
special.periods.data = durcomp.hospitalisation,
ID.colname = "ID",
presc.date.colname = "DATE.PRESC",
disp.date.colname = "DATE.DISP",
medication.class.colnames = c("ATC.CODE", "UNIT", "FORM"),
total.dose.colname = "TOTAL.DOSE",
presc.daily.dose.colname = "DAILY.DOSE",
presc.duration.colname = "PRESC.DURATION",
visit.colname = "VISIT",
split.on.dosage.change = TRUE,
```

force.init.presc = TRUE,
force.presc.renew = TRUE,
trt.interruption = "continue",
special.periods.method = "continue",
date.format = "%Y-%m-%d",
suppress.warnings = FALSE,
return.data.table = TRUE);

## End(Not run)

cover_special_periods  Cover special periods.

Description
Identifies special periods that are in proximity to already covered durations and adds additional
events for these durations.

Usage
cover_special_periods(events.data, special.periods.data, ID.colname,
medication.class.colnames, disp.start.colname, duration.colname,
days.before, days.after, date.format, suppress.warnings = FALSE,
return.data.table = FALSE, ...)

Arguments
events.data  A data.frame or data.table with the event durations.
special.periods.data  A data.frame or data.table containing the information about special peri-
ods (e.g., hospitalizations or other situations where medication use may differ,
e.g. during incarcerations or holidays). Must contain the same unique patient ID
as dispensing and prescription data, the start and end dates of the special periods
with the exact column names DATE.IN and DATE.OUT.
ID.colname  A string, the name of the column in events.data and special.periods.data
containing the unique patient ID.
medication.class.colnames  A vector of strings, the name(s) of the column(s) in the events.data identify
medication classes.
disp.start.colname  A string, the name of the column in events.data containing the event start date
(in the format given in the date.format parameter).
duration.colname  A string, the name of the column in events.data containing the duration of the
medication event.
days.before  an integer, the number of days before the start of a special period within which
an event duration must end to consider the special period as covered.
days.after  an integer, the number of days after a special period within which an event duration must start to consider the special period as covered.

date.format  A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

suppress.warnings  Logical, if TRUE don’t show any warnings.

return.data.table  Logical, if TRUE return a data.table object, otherwise a data.frame.

...  other possible parameters.

Details

Special periods may appear as gaps, possibly leading to underestimation of implementation or even assumption of discontinuation and non-persistence. To consider such periods as covered, this function adds additional durations, for example when it is assumed that hospitalized patients are adherent during the hospitalization period. This function should be used after pruning with prune_event_durations.

Value

A data.frame or data.table, the events.data with the additional durations for special periods covered.

Examples

```r
## Not run:
# select medication class of interest and compute event durations
disp_data <- durcomp.dispensing[ID == 3 & grepl("J01EE01", ATC.CODE)]
presc_data <- durcomp.prescribing[ID == 3 & grepl("J01EE01", ATC.CODE)]

event_durations_list <- compute_event_durations(disp.data = disp_data,
                                               presc.data = presc_data,
                                               special.periods.data = durcomp.hospitalisation,
                                               special.periods.method = "carryover",
                                               ID.colname = "ID",
                                               presc.date.colname = "DATE.PRESC",
                                               disp.date.colname = "DATE.DISP",
                                               date.format = "%Y-%m-%d",
                                               medication.class.colnames = c("ATC.CODE",
                                                                              "UNIT",
                                                                              "FORM"),
                                               total.dose.colname = "TOTAL.DOSE",
                                               presc.daily.dose.colname = "DAILY.DOSE",
                                               presc.duration.colname = "PRESC.DURATION",
                                               visit.colname = "VISIT",
                                               force.init.presc = TRUE,
                                               force.presc.renew = TRUE,
                                               split.on.dosage.change = TRUE,
                                               trt.interruption = "carryover",
```

Example dispensing events for 16 patients.

**Description**

A sample dataset containing dispensing events (one per row) for 16 patients over a period of roughly 24 months (1794 events in total). This is the appropriate format to compute event durations with the `compute_event_durations` function. Each row represents an individual dispensing record for a specific dose of a specific medication for a patient at a given date. More than one column to group medications can be supplied (such as ATC code, Form and Unit).

**Usage**

`durcomp.dispensing`

**Format**

A data frame with 1794 rows and 6 variables:

- **ID** integer here; patient unique identifier. Can also be string.
- **DATE.DISP** Date here: the dispensing event date, by default in the yyyy-mm-dd format. Can also be string.
ATC.CODE character; the medication type, according to the WHO ATC classification system. This can be a researcher-defined classification depending on study aims (e.g., based on therapeutic use, mechanism of action, chemical molecule, or pharmaceutical formulation). The compute_event_durations function will match prescribed medication to dispensed medications based on this variable.

UNIT integer; the unit of the dispensed dose. This is optional and can be used as a separate variable to match between prescription and dispensing events.

FORM character; the galenic form of the dispensed preparation. This is optional and can be used as a separate variable to match between prescription and dispensing events.

TOTAL.DOSE numeric; the total dispensed dose supplied at this medication event (e.g., 5000 for 10 tables of 500 mg).

durcomp.hospitalisation

Example special periods for 10 patients.

---

Description

A sample dataset containing special periods (one per row) for 10 patients over a period of roughly 18 months (28 events in total). This is the appropriate format to compute event durations with the compute_event_durations function. Each row represents an individual special period of type “hospitalization” of a patient for whom event durations should be calculated. Besides hospitalizations, this could cover other situations where medication use may differ, e.g. during incarcerations or holidays. All column names must match the format provided in this example.

Usage

durcomp.hospitalisation

Format

A data frame with 28 rows and 3 variables:

ID Integer here; patient unique identifier. Can also be string.

DATE.IN Date here; the start of the hospitalization period, by default in the yyyy-mm-dd format. Can also be string.

DATE.OUT Date: the end of the hospitalization period, by default in the yyyy-mm-dd format. Can also be string.
Example prescription events for 16 patients.

**Description**

A sample dataset containing prescription events (one per row) for 16 patients over a period of roughly 15 months (1502 events in total). This is the appropriate format to compute event durations with the `compute_event_durations` function. Each row represents an individual prescription record for a specific dose of a specific medication for a patient at a given date. Visit number and Duration are optional, and more than one column to group medications can be supplied (such as ATC Code, Form or Unit).

**Usage**

durcomp.prescribing

**Format**

A data table with 1502 rows and 8 variables:

**ID** integer here; patient unique identifier. Can also be string.

**DATE PRESC** Date here; the prescription event date, by default in the yyyy-mm-dd format. Can also be string.

**VISIT** integer; the consecutive number of the prescription instances. This column is optional and will be generated internally when not supplied. It is used to identify treatment interruptions.

**ATC.CODE** character; the medication type, according to the WHO ATC classification system. This can be a researcher-defined classification depending on study aims (e.g., based on therapeutic use, mechanism of action, chemical molecule, or pharmaceutical formulation). The `compute_event_durations` function will match prescribed medication to dispensed medications based on this variable.

**FORM** character; the galenic form of the prescribed preparation. This is optional and can be used as a separate variable to match between prescription and dispensing events.

**UNIT** integer; the unit of the prescribed dose. This is optional and can be used as a separate variable to match between prescription and dispensing events.

**PRESC.DURATION** numeric; the duration (in days) for which the prescription is intended. Can be NA if the prescription is continuous without a fixed end date.

**DAILY.DOSE** numeric; the daily dose prescribed during this event (e.g., 50 for 1 tablet of 50 mg per day or 25 for 1 tablet of 50 mg every two days).
getCallerWrapperLocation

This function returns the full path to where the various wrappers that can call \texttt{AdhereR} are installed.

Usage

```r
getCallerWrapperLocation(callig.platform = c("python3")[1],
full.path = FALSE)
```

Arguments

- `callig.platform`  
  A string specifying the desired wrapper. Currently it can be "python3".
- `full.path`  
  A logical specifying if the returned path should also include the wrapper's main file name.

Details

In most cases, these wrappers are one or more files in the calling language that may be directly used as such. For more details see the vignette describing the included reference Python 3 wrapper.

Value

The full path to the requested wrapper or NULL if none exists.

getcma

Access the actual CMA estimate from a CMA object.

Description

Retrieve the actual CMA estimate(s) encapsulated in a simple, per episode, or sliding window CMA object.

Usage

```r
getcma(x)
```

Arguments

- `x`  
  a CMA object.
med.events

Value

A `data.frame` containing the CMA estimate(s).

Examples

```r
cma1 <- CMA1(data=med.events,
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
getCMA(cma1);
## Not run:
cmaE <- CMA_per_episode(CMA="CMA1",
  data=med.events,
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=0,
  observation.window.start=0,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
getCMA(cmaE);
## End(Not run)
```

Description

An artificial dataset containing medication events (one per row) for 100 patients (1080 events in total). This is the dataset format appropriate for medication adherence analyses performed with the R package AdhereR. Medication events represent individual records of prescribing or dispensing a specific medication for a patient at a given date. Dosage and medication type is optional (only needed if calculation of adherence or persistence takes into account changes in dosage and type of medication).

Usage

`med.events`
Format

A data frame with 1080 rows and 5 variables:

PATIENT_ID integer here; patient unique identifier. Can also be string.

DATE character; the medication event date, by default in the mm/dd/yyyy format. It may represent
a prescribing or dispensing date.

PERDAY integer; the daily dosage prescribed for the medication supplied at this medication event
(i.e. how many doses should be taken daily according to the prescription). This column is
optional, as it is not considered in all functions but may be relevant for specific research or
clinical contexts. All values should be > 0.

CATEGORY character; the medication type, here two placeholder labels, 'medA' and 'medB'.
This is a researcher-defined classification depending on study aims (e.g., based on therapeutic
use, mechanism of action, chemical molecule, or pharmaceutical formulation). This column
is optional, as it is not considered in all functions but may be relevant for specific research or
clinical contexts.

DURATION integer; the medication event duration in days (i.e. how many days the mediation
supplied would last if used as prescribed); may be available in the extraction or computed
based on quantity supplied (the number of doses prescribed or dispensed on that occasion)
and daily dosage. All values should be > 0.

plot.CMA0

Plot CMA0 objects.

Description

Plots the events (prescribing or dispensing) data encapsulated in a basic CMA0 object.

Usage

```r
## S3 method for class 'CMA0'
plot(x, ..., patients.to.plot = NULL, duration = NA,
     align.all.patients = FALSE, show.period = c("dates", "days")[2],
     period.in.days = 90, show.legend = TRUE, legend.x = "right",
     legend.y = "bottom", legend.bkg.opacity = 0.5, legend.cex = 0.75,
     legend.cex.title = 1, cex = 1, cex.axis = 0.75, cex.lab = 1,
     col.cats = rainbow, unspecified.category.label = "drug",
     lty.event = "solid", lwd.event = 2, pch.start.event = 15,
     pch.end.event = 16, print.dose = FALSE, cex.dose = 0.75,
     print.dose.outline.col = "white", print.dose.centered = FALSE,
     plot.dose = FALSE, lwd.event.max.dose = 8,
     plot.dose.lwd.across.medication.classes = FALSE,
     col.continuation = "black", lty.continuation = "dotted",
     lwd.continuation = 1, col.na = "lightgray",
     highlight.followup.window = TRUE, followup.window.col = "green",
     highlight.observation.window = TRUE,
```

observation.window.col = "yellow", observation.window.density = 35,
observeration.window.angle = -30, observation.window.opacity = 0.3,
bw.plot = FALSE, print.CMA = TRUE, plot.CMA = TRUE,
CMA.plot.ratio = 0.1, min.plot.size.in.characters.horiz = 10,
min.plot.size.in.characters.vert = 0.5, max.patients.to.plot = 100)

Arguments

x A CMA or derived object, representing the CMA to plot

... other possible parameters

patients.to.plot A vector of strings containing the list of patient IDs to plot (a subset of those in the cma object), or NULL for all

duration A number, the total duration (in days) of the whole period to plot; in NA it is automatically determined from the event data such that the whole dataset fits.

align.all.patients Logical, should all patients be aligned (i.e., the actual dates are discarded and all plots are relative to the earliest date)?

show.period A string, if "dates" show the actual dates at the regular grid intervals, while for "days" (the default) shows the days since the beginning; if align.all.patients == TRUE, show.period is taken as "days".

period.in.days The number of days at which the regular grid is drawn (or 0 for no grid).

show.legend Logical, should the legend be drawn?

legend.x The position of the legend on the x axis; can be "left", "right" (default), or a numeric value.

legend.y The position of the legend on the y axis; can be "bottom" (default), "top", or a numeric value.

legend.bkg.opacity A number between 0.0 and 1.0 specifying the opacity of the legend background.

cex, cex.axis, cex.lab, legend.cex, legend.cex.title

numeric values specifying the cex of the various types of text.

col.cats A color or a function that specifies the single colour or the colour palette used to plot the different medication; by default rainbow, but we recommend, whenever possible, a colorblind-friendly palette such as viridis or colorblind_pal.

unspecified.category.label A string giving the name of the unspecified (generic) medication category.

lty.event, lwd.event, pch.start.event, pch.end.event

The style of the event (line style, width, and start and end symbols).

print.dose Logical, should the daily dose be printed as text?

lty.event Numeric, if daily dose is printed, what text size to use?

print.dose.outline.col

If NA, don’t print dose text with outline, otherwise a color name/code for the outline.

print.dose.centered Logical, print the daily dose centered on the segment or slightly below it?
plot.dose  Logical, should the daily dose be indicated through segment width?

lwd.event.max.dose
    Numeric, the segment width corresponding to the maximum daily dose (must be 
    >= lwd.event but not too big either).

plot.dose.lwd.across.medication.classes
    Logical, if TRUE, the line width of the even is scaled relative to all medication 
    classes (i.e., relative to the global minimum and maximum doses), otherwise it 
    is scale relative only to its medication class.

col.continuation, lty.continuation, lwd.continuation
    The style of the "continuation" lines connecting consecutive events (colour, line 
    style and width).

col.na
    The colour used for missing event data.

highlight.followup.window
    Logical, should the follow-up window be plotted?

followup.window.col
    The follow-up window's colour.

highlight.observation.window
    Logical, should the observation window be plotted?

observation.window.col, observation.window.density, observation.window.angle, observation.window.opacity
    Attributes of the observation window (colour, shading density, angle and opacity).

bw.plot
    Logical, should the plot use grayscale only (i.e., the gray.colors function)?

print.CMA
    Logical, should the CMA values be printed?

plot.CMA
    Logical, should the CMA values be represented graphically?

CMA.plot.ratio
    A number, the proportion of the total horizontal plot space to be allocated to the 
    CMA plot. Numeric, the minimum size of the plotting surface in characters; horiz- 
    ontally (min.plot.size.in.characters.horiz) refers to the the whole duration of 
    the events to plot; vertically (min.plot.size.in.characters.vert) refers to a single 
    event.

min.plot.size.in.characters.horiz, min.plot.size.in.characters.vert
    Numeric, the minimum size of the plotting surface in characters; horizontally 
    (min.plot.size.in.characters.horiz) refers to the the whole duration of the events 
    to plot; vertically (min.plot.size.in.characters.vert) refers to a single event.

max.patients.to.plot
    Numeric, the maximum patients to attempt to plot.

Details

The x-axis represents time (either in days since the earliest date or as actual dates), with consecutive 
events represented as ascending on the y-axis.

Each event is represented as a segment with style lty.event and line width lwd.event starting 
with a pch.start.event and ending with a pch.end.event character, coloured with a unique 
color as given by col.cats, extending from its start date until its end date. Consecutive events 
are thus represented on consecutive levels of the y-axis and are connected by a "continuation" line 
with col.continuation colour, lty.continuation style and lwd.continuation width; these
continuation lines are purely visual guides helping to perceive the sequence of events, and carry no information about the availability of medication in this interval.

When several patients are displayed on the same plot, they are organized vertically, and alternating bands (white and gray) help distinguish consecutive patients. Implicitly, all patients contained in the `cma` object will be plotted, but the `patients.to.plot` parameter allows the selection of a subset of patients.

**Examples**

```r

cma0 <- CMA0(data=med.events,
    ID.colname="PATIENT_ID",
    event.date.colname="DATE",
    event.duration.colname="DURATION",
    event.daily.dose.colname="PERDAY",
    medication.class.colname="CATEGORY",
    followup.window.start=0,
    followup.window.start.unit="days",
    followup.window.duration=2*365,
    followup.window.duration.unit="days",
    observation.window.start=30,
    observation.window.start.unit="days",
    observation.window.duration=365,
    observation.window.duration.unit="days",
    date.format="%m/%d/%y",
    summary="Base CMA")

plot(cma0, patients.to.plot=c("1","2"));
```

---

**plot.CMA1**

Plot `CMA0-derived objects`.

**Description**

Plots the event data and estimated CMA encapsulated in objects derived from `CMA0`.

**Usage**

```r
#
```

```r
plot(x, ..., patients.to.plot = NULL, duration = NA,
    align.all.patients = FALSE, align.first.event.at.zero = TRUE,
    show.period = c("dates", "days")[2], period.in.days = 90,
    show.legend = TRUE, legend.x = "right", legend.y = "bottom",
    legend.bkg.opacity = 0.5, legend.cex = 0.75, legend.cex.title = 1,
    cex = 1, cex.axis = 0.75, cex.lab = 1, show.cma = TRUE,
    col.cats = rainbow, unspecified.category.label = "drug",
    lty.event = "solid", lwd.event = 2, pch.start.event = 15,
    pch.end.event = 16, show.event.intervals = TRUE,
    col.na = "lightgray", print.CMA = TRUE, CMA.cex = 0.5,
    plot.CMA = TRUE, CMA.plot.ratio = 0.1, CMA.plot.col = "lightgreen",
```
Arguments

\[
x\quad \text{A CMA object or derived object, representing the CMA to plot}
\]

... other possible parameters

\[
\text{patients.to.plot}\quad A \text{vector of strings containing the list of patient IDs to plot (a subset of those in the cma object), or NULL for all patients.}
\]

\[
\text{duration}\quad A \text{number, the total duration (in days) of the whole period to plot; in NA it is automatically determined from the event data such that the whole dataset fits.}
\]

\[
\text{align.all.patients}\quad \text{Logical, should all patients be aligned (i.e., the actual dates are discarded and}
\]
align.first.event.at.zero
  Logical, should the first event be placed at the origin of the time axis (at 0)?

show.period
  A string, if "dates" show the actual dates at the regular grid intervals, while for "days" (the default) shows the days since the beginning; if align.all.patients == TRUE, show.period is taken as "days".

period.in.days
  The number of days at which the regular grid is drawn (or 0 for no grid).

show.legend
  Logical, should the legend be drawn?

legend.x
  The position of the legend on the x axis; can be "left", "right" (default), or a numeric value.

legend.y
  The position of the legend on the y axis; can be "bottom" (default), "top", or a numeric value.

legend.bkg.opacity
  A number between 0.0 and 1.0 specifying the opacity of the legend background.

cex, cex.axis, cex.lab, legend.cex, legend.cex.title, CMA.cex
  numeric values specifying the cex of the various types of text.

show.cma
  Logical, should the CMA type be shown in the title?

col.cats
  A color or a function that specifies the single colour or the colour palette used to plot the different medication; by default rainbow, but we recommend, whenever possible, a colorblind-friendly palette such as viridis or colorblind_pal.

unspecified.category.label
  A string giving the name of the unspecified (generic) medication category.

lty.event, lwd.event, pch.start.event, pch.end.event
  The style of the event (line style, width, and start and end symbols).

show.event.intervals
  Logical, should the actual event intervals be shown?

col.na
  The colour used for missing event data.

print.CMA
  Logical, should the CMA values be printed?

plot.CMA
  Logical, should the CMA values be represented graphically?

CMA.plot.ratio
  A number, the proportion of the total horizontal plot space to be allocated to the CMA plot.

CMA.plot.col, CMA.plot.border, CMA.plot.bkg, CMA.plot.text
  Strings giving the colours of the various components of the CMA plot.

highlight.followup.window
  Logical, should the follow-up window be plotted?

followup.window.col
  The follow-up window’s colour.

highlight.observation.window
  Logical, should the observation window be plotted?

observation.window.col, observation.window.density, observation.window.angle, observation.window.opacity
  Attributes of the observation window (colour, shading density, angle and opacity).
show.real.obs.window.start, real.obs.window.density, real.obs.window.angle

For some CMAs, the observation window might be adjusted, in which case should it be plotted and with that attributes?

print.dose Logical, should the daily dose be printed as text?
cex.dose Numeric, if daily dose is printed, what text size to use?
print.dose.outline.col

If NA, don’t print dose text with outline, otherwise a color name/code for the outline.

print.dose.centered Logical, print the daily dose centered on the segment or slightly below it?
plot.dose Logical, should the daily dose be indicated through segment width?
lwd.event.max.dose

Numeric, the segment width corresponding to the maximum daily dose (must be >= lwd.event but not too big either).

plot.dose.lwd.across.medication.classes Logical, if TRUE, the line width of the even is scaled relative to all medication classes (i.e., relative to the global minimum and maximum doses), otherwise it is scale relative only to its medication class.

bw.plot Logical, should the plot use grayscale only (i.e., the gray.colors function)?
min.plot.size.in.characters.horiz, min.plot.size.in.characters.vert

Numeric, the minimum size of the plotting surface in characters; horizontally (min.plot.size.in.characters.horiz) refers to the the whole duration of the events to plot; vertically (min.plot.size.in.characters.vert) refers to a single event.

max.patients.to.plot

Numeric, the maximum patients to attempt to plot.

Details

Please note that this function plots objects inheriting from CMA0 but not objects of type CMA0 itself (these are plotted by plot.CMA0).

The x-axis represents time (either in days since the earliest date or as actual dates), with consecutive events represented as ascending on the y-axis.

Each event is represented as a segment with style lty.event and line width lwd.event starting with a pch.start.event and ending with a pch.end.event character, coloured with a unique color as given by col.cats, extending from its start date until its end date. Superimposed on these are shown the event intervals and gap days as estimated by the particular CMA method, more precisely plotting the start and end of the available events as solid filled-in rectangles, and the event gaps as shaded rectangles.

The follow-up and the observation windows are plotted as an empty rectangle and as shaded rectangle, respectively (for some CMAs the observation window might be adjusted in which case the adjustment may also be plotted using a different shading).

The CMA estimates can be visually represented as well in the left side of the figure using bars (sometimes the estimates can go above 100%, in which case the maximum possible bar filling is adjusted to reflect this).

When several patients are displayed on the same plot, they are organized vertically, and alternating bands (white and gray) help distinguish consecutive patients. Implicitely, all patients contained in
the cma object will be plotted, but the patients.to.plot parameter allows the selection of a subset of patients.

Finally, the y-axis shows the patient ID and possibly the CMA estimate as well.

Examples

cma1 <- CMA1(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
plot(cma1, patients.to.plot=c("1","2"));

plot.CMA_per_episode  Plot CMA_per_episode and CMA_sliding_window objects.

Description

Plots the event data and the estimated CMA per treatment episode and sliding window, respectively.

Usage

```r
## S3 method for class 'CMA_per_episode'
plot(x, ..., patients.to.plot = NULL,
  duration = NA, align.all.patients = FALSE,
  align.first.event.at.zero = TRUE, show.period = c("dates",
  "days")][2], period.in.days = 90, show.legend = TRUE,
  legend.x = "right", legend.y = "bottom", legend.bkg.opacity = 0.5,
  cex = 1, cex.axis = 0.75, cex.lab = 1, show.cma = TRUE,
  col.cats = rainbow, unspecified.category.label = "drug",
  lty.event = "solid", lwd.event = 2, pch.start.event = 15,
  pch.end.event = 16, col.na = "lightgray",
  col.continuation = "black", lty.continuation = "dotted",
  lwd.continuation = 1, print.CMA = TRUE, CMA.cex = 0.5,
  plot.CMA = TRUE, plot.CMA.as.histogram = TRUE,
  CMA.plot.ratio = 0.1, CMA.plot.col = "lightgreen",
  CMA.plot.border = "darkgreen", CMA.plot.bkg = "aquamarine",
  CMA.plot.text = CMA.plot.border, highlight.followup.window = TRUE,
  followup.window.col = "green", highlight.observation.window = TRUE,
  observation.window.col = "yellow", observation.window.density = 35,
  observation.window.angle = -30, show.real.obs.window.start = TRUE,
  real.obs.window.density = 35, real.obs.window.angle = 30,
  bw.plot = FALSE)
```
## S3 method for class 'CMA_sliding_window'

plot(...)

### Arguments

- **x**: A CMA or derived object, representing the CMA to plot
- **...**: other parameters (to be passed to the estimation and plotting of the simple CMA)
- **patients.to.plot**: A vector of *strings* containing the list of patient IDs to plot (a subset of those in the cma object), or NULL for all
- **duration**: A *number*, the total duration (in days) of the whole period to plot; in NA it is automatically determined from the event data such that the whole dataset fits.
- **align.all.patients**: *Logical*, should all patients be aligned (i.e., the actual dates are discarded and all plots are relative to the earliest date)?
- **align.first.event.at.zero**: *Logical*, should the first event be placed at the origin of the time axis (at 0)?
- **show.period**: A *string*, if "dates" show the actual dates at the regular grid intervals, while for "days" (the default) shows the days since the beginning; if `align.all.patients` == TRUE, `show.period` is taken as "days".
- **period.in.days**: The *number* of days at which the regular grid is drawn (or 0 for no grid).
- **show.legend**: *Logical*, should the legend be drawn?
- **legend.x**: The position of the legend on the x axis; can be "left", "right" (default), or a *numeric* value.
- **legend.y**: The position of the legend on the y axis; can be "bottom" (default), "top", or a *numeric* value.
- **legend.bkg.opacity**: A *number* between 0.0 and 1.0 specifying the opacity of the legend background.
- **cex, cex.axis, cex.lab**: *numeric* values specifying the cex of the various types of text.
- **show.cma**: *Logical*, should the CMA type be shown in the title?
- **col.cats**: A *color* or a *function* that specifies the single colour or the colour palette used to plot the different medication; by default `rainbow`, but we recommend, whenever possible, a colorblind-friendly palette such as `viridis` or `colorblind_pal`.
- **unspecified.category.label**: A *string* giving the name of the unspecified (generic) medication category.
- **lty.event, lwd.event, pch.start.event, pch.end.event**: The style of the event (line style, width, and start and end symbols).
- **col.na**: The colour used for missing event data.
- **col.continuation, lty.continuation, lwd.continuation**: The color, style and width of the continuation lines connecting consecutive events.
- **print.CMA**: *Logical*, should the CMA values be printed?
- **CMA.cex**: ... and, if printed, what cex (*numeric*) to use?
Logical, should the CMA values be represented graphically?

plot.CMA.as.histogram

Logical, should the CMA plot be a histogram or a (truncated) density plot?

Please note that it is TRUE by default for CMA_per_episode and FALSE for CMA_sliding_window, because usually there are more sliding windows than episodes. Also, the density estimate cannot be estimated for less than three different values.

CMA.plot.ratio

A number, the proportion of the total horizontal plot space to be allocated to the CMA plot.

CMA.plot.col, CMA.plot.border, CMA.plot.bkg, CMA.plot.text

Strings giving the colours of the various components of the CMA plot.

highlight.followup.window

Logical, should the follow-up window be plotted?

followup.window.col

The follow-up window colour.

highlight.observation.window

Logical, should the observation window be plotted?

observation.window.col, observation.window.density, observation.window.angle

Attributes of the observation window (colour, shading density and angle).

show.real.obs.window.start, real.obs.window.density, real.obs.window.angle

For some CMAs, the observation window might be adjusted, in which case should it be plotted and with what attributes?

bw.plot

Logical, should the plot use grayscale only (i.e., the gray.colors function)?

Details

The x-axis represents time (either in days since the earliest date or as actual dates), with consecutive events represented as ascending on the y-axis.

Each event is represented as a segment with style lty.event and line width lwd.event starting with a pch.start.event and ending with a pch.end.event character, coloured with a unique color as given by col.cats, extending from its start date until its end date. Consecutive events are thus represented on consecutive levels of the y-axis and are connected by a "continuation" line with col.continuation colour, lty.continuation style and lwd.continuation width; these continuation lines are purely visual guides helping to perceive the sequence of events, and carry no information about the availability of medicine in this interval.

Above these, the treatment episodes or the sliding windows are represented in a stacked manner from the earliest (left, bottom of the stack) to the latest (right, top of the stack), each showing the CMA as percent fill (capped at 100% even if CMA values may be higher) and also as text.

The follow-up and the observation windows are plotted as empty an rectangle and as shaded rectangle, respectively (for some CMAs the observation window might be adjusted in which case the adjustment may also be plotted using a different shading).

The kernel density ("smoothed histogram") of the CMA estimates across treatment episodes/sliding windows (if more than 2) can be visually represented as well in the left side of the figure (NB, their horizontal scales may be different across patients).

When several patients are displayed on the same plot, they are organized vertically, and alternating bands (white and gray) help distinguish consecutive patients. Implicitly, all patients contained in
the cma object will be plotted, but the patients.to.plot parameter allows the selection of a subset of patients.

Finally, the y-axis shows the patient ID and possibly the CMA estimate as well.

Any not explicitly defined arguments are passed to the simple CMA estimation and plotting function; therefore, for more info about possible estimation parameters please see the help for the appropriate simple CMA, and for possible aesthetic tweaks, please see the help for their plotting.

See Also

See the simple CMA estimation \texttt{CMA1} to \texttt{CMA9} and plotting \texttt{plot.CMA1} functions for extra parameters.

Examples

\begin{verbatim}
## Not run:
cmaW <- CMA_sliding_window(CMA=CMA1,
  data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=0,
  observation.window.start=0,
  observation.window.duration=365,
  sliding.window.start=0,
  sliding.window.start.unit="days",
  sliding.window.duration=90,
  sliding.window.duratio...[end of line]
  date.format="%m/%d/%Y"
);
plot(cmaW, patients.to.plot=c("1","2"));
cmaE <- CMA_per_episode(CMA=CMA1,
  data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  carry.only.for.same.medication=FALSE,
  consider.dosage.change=FALSE,
  followup.window.start=0,
  observation.window.start=0,
  observation.window.duration=365,
  date.format="%m/%d/%Y"
);
\end{verbatim}
plot_interactive_cma

Interactive exploration CMA computation.

Description

Interactively plot a given patient’s data, allowing the real-time exploration of the various CMAs and their parameters. It can use RStudio’s `manipulate` library or Shiny.

Usage

```r
plot_interactive_cma(data = NULL, ID = NULL, cma.class = c("simple", "per episode", "sliding window")[[1]], print.full.params = FALSE,
  ID.colname = NA, event.date.colname = NA,
  event.duration.colname = NA, event.daily.dose.colname = NA,
  medication.class.colname = NA, date.format = "%m/%d/%Y",
  followup.window.start.max = 5 * 365, followup.window.duration.max = 5
  * 365, observation.window.start.max = followup.window.start.max,
  observation.window.duration.max = followup.window.duration.max,
  align.all.patients = FALSE, align.first.event.at.zero = TRUE,
  maximum.permissible.gap.max = 2 * 365,
  sliding.window.start.max = followup.window.start.max,
  sliding.window.duration.max = 2 * 365,
  sliding.window.step.duration.max = 2 * 365, backend = c("shiny",
  "rstudio")[[1]], use.system.browser = FALSE,
  get.colnames.fnc = function(d) names(d),
  get.patients.fnc = function(d, idcol) unique(d[[idcol]])],
  get.data.for.patients.fnc = function(patientid, d, idcol, cols = NA,
  maxrows = NA) d[[idcol]] %in% patientid, ...)
```

Arguments

- `data`: Usually a `data.frame` containing the events (prescribing or dispensing) used to compute the CMA. Must contain, at a minimum, the patient unique ID, the event date and duration, and might also contain the daily dosage and medication type (the actual column names are defined in the following four parameters). Alternatively, this can be any other data source (for example, a connection to a database), in which case the user must redefine the arguments `get.colnames.fnc`, `get.patients.fnc`, and `get.data.for.patients.fnc` appropriately. Currently, this works only when using Shiny for interactive rendering. For a working example, please see the vignette describing the interfacing with databases.

- `ID`: The ID (as given in the `ID.colname` column) of the patient whose data to interactively plot (if absent, pick the first one); please note that this an be interactively selected during plotting.
The type of CMAs to plot; can be "simple" (CMA0 to CMA9), "per episode", or "sliding window".

A logical specifying if the values of all the parameters used to generate the current plot should be printed in the console (if TRUE, it can generate extremely verbose output!).

A string, the name of the column in data containing the unique patient ID, or NA if not defined.

A string, the name of the column in data containing the start date of the event (in the format given in the date.format parameter), or NA if not defined.

A string, the name of the column in data containing the event duration (in days), or NA if not defined.

A string, the name of the column in data containing the prescribed daily dose, or NA if not defined.

A string, the name of the column in data containing the classes/types/groups of medication, or NA if not defined.

A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).

The maximum number of days when the follow-up window can start.

The maximum duration of the follow-up window in days.

The maximum number of days when the observation window can start.

The maximum duration of the observation window in days.

Should the patients be aligned?

Should the first event be put at zero?

The maximum permissible gap in days.

The maximum number of days when the sliding windows can start.

The maximum duration of the sliding windows in days.

The maximum sliding window step in days.

The plotting backend to use; "shiny" (the default) tries to use the Shiny framework, while "rstudio" uses the manipulate RStudio capability.
For shiny, use the system browser?

get.colnames.fnc

A function taking as parameter the data source and returning the column names. Must be overridden when the data source is not derived from a `data.frame`.

get.patients.fnc

A function taking as parameter the data source and the patient ID column name, and returns the list of all patient IDs. Must be overridden when the data source is not derived from a `data.frame`.

get.data.for.patients.fnc

A function taking as parameter a (set of) patient ID(s), the data source, and the patient ID column name, and returns the list of all patient IDs. Must be overridden when the data source is not derived from a `data.frame`.

... Extra arguments.

Details

The `manipulate` is kept for backward compatibility only, as it is much more limited than Shiny and will receive no new development in the future. Shiny currently allows the use of any other data source besides a default (and usual) `data.frame` (or derived), such a connection to an SQL database. In this case, the user must redefine the three argument functions `get.colnames.fnc`, `get.patients.fnc` and `get.data.for.patients.fnc` which collectively define an interface for listing the column names, all the patient IDs, and for retrieving the actual data for a (set of) patient ID(s). A fully worked example is described in the vignette detailing the access to standard databases storing the patient information. For more info please see the online vignette `https://htmlpreview.github.io/?https://github.com/ddedu/AdhereR/blob/master/online-only-doc/adherer_interactive_plots/adherer_interactive_plots.html`.

Value

Nothing

See Also


Examples

```r
## Not run:
plot_interactive_cma(med.events,
  ID.colname=“PATIENT_ID”,
  event.date.colname=“DATE”,
  event.duration.colname=“DURATION”,
  event.daily.dose.colname=“PERDAY”,
  medication.class.colname=“CATEGORY”);
## End(Not run)
```
**Description**

Prints and summarizes a basic CMA0, or derived, object.

**Usage**

```r
## S3 method for class 'CMA0'
print(x, ..., inline = FALSE, format = c("text", "latex", "markdown"), print.params = TRUE, print.data = TRUE, exclude.params = c("event.info"), skip.header = FALSE, cma.type = class(cma)[1])

## S3 method for class 'CMA1'
print(...)

## S3 method for class 'CMA2'
print(...)

## S3 method for class 'CMA3'
print(...)

## S3 method for class 'CMA4'
print(...)

## S3 method for class 'CMA5'
print(...)

## S3 method for class 'CMA6'
print(...)

## S3 method for class 'CMA7'
print(...)

## S3 method for class 'CMA8'
print(...)

## S3 method for class 'CMA9'
print(...)

## S3 method for class 'CMA_per_episode'
print(x, ..., inline = FALSE, format = c("text", "latex", "markdown"), print.params = TRUE, print.data = TRUE, exclude.params = c("event.info"), skip.header = FALSE, cma.type = class(cma)[1])
```
## S3 method for class 'CMA_sliding_window'
print(...)

Arguments

- **x**: A CMA or derived object, representing the CMA to print.
- **...**: other possible parameters
- **inline**: *Logical*, should print inside a line of text or as a separate, extended object?
- **format**: A *string*, the type of output: plain text ("text"; default), LaTeX ("latex") or R Markdown ("markdown").
- **print.params**: *Logical*, should print the parameters?
- **print.data**: *Logical*, should print a summary of the data?
- **exclude.params**: A vector of *strings*, the names of the object fields to exclude from printing (usually, internal information irrelevant to the end-user).
- **skip.header**: *Logical*, should the header be printed?
- **cma.type**: A *string*, used to override the reported object’s class.

Details

Can produce output for the console (text), R Markdown or LaTeX, showing various types of information.

Examples

```r
# Assume 'med.events' is defined

cma0 <- CMA0(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  event.daily.dose.colname="PERDAY",
  medication.class.colname="CATEGORY",
  followup.window.start=0,
  followup.window.start.unit="days",
  followup.window.duration=2*365,
  followup.window.duration.unit="days",
  observation.window.start=30,
  observation.window.start.unit="days",
  observation.window.duration=365,
  observation.window.duration.unit="days",
  date.format="%m/%d/%Y",
  summary="Base CMA")

cma0;
print(cma0, format="markdown");
cma1 <- CMA1(data=med.events,
  ID.colname="PATIENT_ID",
  event.date.colname="DATE",
  event.duration.colname="DURATION",
  followup.window.start=30,
  observation.window.start=30,
```
**prune_event_durations**  

Prune event durations.

**Description**

Flags or removes leftover supply durations after dosage changes, the end of a special period, or treatment interruption. The function accepts the raw list output of `compute_event_durations` and additional arguments to specify event durations that need to be removed.

**Usage**

```r
prune_event_durations(data, include = c("special periods", "treatment interruptions", "dosage changes"), 
medication.class.colnames = data$medication.class.colnames, 
days.within.out.date.1, days.within.out.date.2, keep.all = TRUE, 
suppress.warnings = FALSE, return.data.table = FALSE, ...)
```

**Arguments**

- `data` A list, the output of `compute_event_durations`.
- `include` A vector of strings indicating whether to include dosage changes, special periods, and/or treatment interruptions.
- `medication.class.colnames` A vector of strings, the name(s) of the column(s) in the `event_durations` element of `data` to identify medication classes. Defaults to the columns used in `compute_event_durations`.
- `days.within.out.date.1` event durations from before the dosage change, special period, or treatment interruption are removed if there is a new dispensing event within the number of days specified as integer after the dosage change or end of the special period/treatment interruption.
- `days.within.out.date.2` event durations from before dosage change, special period, or treatment interruption are removed if there is NO new dispensing event within the number of days specified as integer after the dosage change or end of the special period/treatment interruption.
- `keep.all` Logical, should events be kept and marked for removal? If TRUE, a new column `.prune.event` will be added to `event_durations`, if FALSE the events will be removed from the output.
- `suppress.warnings` Logical, if TRUE don't show any warnings.
prune_event_durations

return.data.table

Logical, if TRUE return a data.table object, otherwise a data.frame.

... other possible parameters.

Details

Dosage changes, special periods, and treatment interruptions may lead to overestimation of implementation, e.g. if patients get a refill after discharge from hospital and don’t continue to use their previous supply. Likewise, it may also lead to overestimation of persistence, e.g. when patients discontinue treatments after the end of a special period or treatment interruption.

Value

A data.frame or data.table, the pruned event_durations.

Examples

```r
## Not run:
# select medication class of interest and compute event durations

disp_data <- durcomp.dispensing[ID == 3 & grepl("J01EE01", ATC.CODE)]
presc_data <- durcomp.prescribing[ID == 3 & grepl("J01EE01", ATC.CODE)]

# compute event durations
event_durations_list <- compute_event_durations(disp.data = disp_data,
                                              presc.data = presc_data,
                                              special.periods.data = durcomp.hospitalisation,
                                              ID.colname = "ID",
                                              presc.date.colname = "DATE.PRESC",
                                              disp.date.colname = "DATE.DISP",
                                              date.format = "%Y-%m-%d",
                                              medication.class.colnames = c("ATC.CODE",
                                                              "UNIT",
                                                              "FORM"),
                                              total.dose.colname = "TOTAL.DOSE",
                                              presc.daily.dose.colname = "DAILY.DOSE",
                                              presc.duration.colname = "PRESC.DURATION",
                                              visit.colname = "VISIT",
                                              force.init.presc = TRUE,
                                              force.presc.renew = TRUE,
                                              split.on.dosage.change = TRUE,
                                              trt.interruption = "carryover",
                                              special.periods.method = "carryover",
                                              suppress.warnings = FALSE,
                                              return.data.table = TRUE,
                                              progress.bar = FALSE)

# prune event durations
event_durations <- prune_event_durations(event_durations_list,
                                          include = c("special periods"),
                                          medication.class.colnames = "ATC.CODE",
                                          days.within.out.date.1 = 7,
                                          days.within.out.date.2 = 14,
                                          unique.periods = TRUE,
                                          return.data.table = TRUE,
                                          progress.bar = FALSE)
```

Computation of initiation times.

Description

Computes the time between the start of a prescription episode and the first dispensing event for each medication class.

Usage

time_to_initiation(presc.data = NULL, disp.data = NULL,
   ID.colname = NA, medication.class.colnames = NA,
   presc.start.colname = NA, disp.date.colname = NA,
   date.format = "%d.%m.%Y", suppress.warnings = FALSE,
   return.data.table = FALSE, ...)

Arguments

- **presc.data**: A data.frame or data.table containing the prescription episodes. Must contain, at a minimum, the patient unique ID, one medication identifier, and the start date of the prescription episode, and might also contain additional columns to identify and group medications (the actual column names are defined in the medication.class.colnames parameter).

- **disp.data**: A data.frame or data.table containing the dispensing events. Must contain, at a minimum, the patient unique ID, one medication identifier, the dispensing date, and might also contain additional columns to identify and group medications (the actual column names are defined in the medication.class.colnames parameter).

- **ID.colname**: A string, the name of the column in presc.data and disp.data containing the unique patient ID, or NA if not defined.

- **medication.class.colnames**: A vector of strings, the name(s) of the column(s) in data containing the classes/types/groups of medication, or NA if not defined.

- **presc.start.colname**: A string, the name of the column in presc.data containing the prescription date (in the format given in the date.format parameter), or NA if not defined.

- **disp.date.colname**: A string, the name of the column in disp.data containing the dispensing date (in the format given in the date.format parameter), or NA if not defined.

- **date.format**: A string giving the format of the dates used in the data and the other parameters; see the format parameters of the as.Date function for details (NB, this concerns only the dates given as strings and not as Date objects).
suppress.warnings
  Logical, if TRUE don’t show any warnings.
return.data.table
  Logical, if TRUE return a data.table object, otherwise a data.frame.

... other possible parameters

Details

The period between the start of a prescription episode and the first dose administration may impact health outcomes differently than omitting doses once on treatment or interrupting medication for longer periods of time. Primary non-adherence (not acquiring the first prescription) or delayed initiation may have a negative impact on health outcomes. The function `time_to_initiation` calculates the time between the start of a prescription episode and the first dispensing event, taking into account multiple variables to differentiate between treatments.

Value

A data.frame or data.table with the following columns:

- **ID.colname** the unique patient ID, as given by the `ID.colname` parameter.
- **medication.class.colnames** the column(s) with classes/types/groups of medication, as given by the `medication.class.colnames` parameter.
- **episode.start** the date of the first prescription event.
- **first.disp** the date of the first dispensing event.
- **time.to.initiation** the difference in days between the first dispensing date and the first prescription date.

Examples

```r
# Set parameters
durcomp.prescribing = "PREScribing",
durcomp.dispensing = "Dispensing",
suppress.warnings = FALSE,
return.data.table = TRUE;

time_init <- time_to_initiation(presc.data = durcomp.prescribing,
  disp.data = durcomp.dispensing,
  ID.colname = "ID",
  medication.class.colnames = c("ATC.CODE", "FORM", "UNIT"),
  presc.start.colname = "DATE.PRESC",
  disp.date.colname = "DATE.DISP",
  date.format = "%Y-%m-%d",
  suppress.warnings = FALSE,
  return.data.table = TRUE);
```
Index

*Topic datasets
  durcomp.dispensing, 62
  durcomp.hospitalisation, 63
  durcomp.prescribing, 64
  med.events, 66

as.Date, 5, 9, 13, 18, 22, 27, 31, 36, 41, 46, 50, 54, 57, 61, 79, 85

callAdhereR, 3
CMA0, 4, 43, 48
CMA1, 7, 14, 43, 48, 77
CMA2, 10, 12, 43, 48
CMA3, 43, 48
CMA3 (CMA1), 7
CMA4, 43, 48
CMA4 (CMA2), 12
CMA5, 16, 43, 48
CMA6, 21, 43, 48
CMA7, 25, 43, 48
CMA8, 29, 43, 48
CMA9, 34, 43, 48, 77
CMA_per_episode, 39, 46, 48
CMA_sliding_window, 42, 43, 44
compute.event.int.gaps, 11, 15, 20, 24, 29, 33, 38, 43, 47, 48
compute.treatment.episodes, 42, 52
compute_event_durations, 55
cover_special_periods, 60

durcomp.dispensing, 62
durcomp.hospitalisation, 63
durcomp.prescribing, 64

getCallerWrapperLocation, 65
getCMA, 65
gray.colors, 69, 73, 76

med.events, 66

plot.CMA0, 67, 73

plot.CMA1, 70, 77
plot.CMA2 (plot.CMA1), 70
plot.CMA3 (plot.CMA1), 70
plot.CMA4 (plot.CMA1), 70
plot.CMA5 (plot.CMA1), 70
plot.CMA6 (plot.CMA1), 70
plot.CMA7 (plot.CMA1), 70
plot.CMA8 (plot.CMA1), 70
plot.CMA9 (plot.CMA1), 70
plot.CMA_per_episode, 74
plot.CMA_sliding_window
  (plot.CMA_per_episode), 74
plot_interactive_cma, 78
print.CMA0, 81
print.CMA1 (print.CMA0), 81
print.CMA2 (print.CMA0), 81
print.CMA3 (print.CMA0), 81
print.CMA4 (print.CMA0), 81
print.CMA5 (print.CMA0), 81
print.CMA6 (print.CMA0), 81
print.CMA7 (print.CMA0), 81
print.CMA8 (print.CMA0), 81
print.CMA9 (print.CMA0), 81
print.CMA_per_episode (print.CMA0), 81
print.CMA_sliding_window (print.CMA0), 81

prune_event_durations, 83
time_to_initiation, 85