Data preparation for computing adherence to medication in AdhereR: overview of the event\_durations functions

Samuel S. Allemann, Dan Dediu & Alex L. Dima

2019-05-13

Adherence to medication is defined as the agreement between prescribed and actual medication use. Adherence metrics are often computed from electronic healthcare data (EHD) based on a single data source, such as pharmacy dispensing, or other administrative data. However, information about the medication prescribed and dispensed to a patient in a given time period may be found in different data sources, which become increasingly accessible for linking and thus for a more precise description of adherence patterns.

The core AdhereR functions estimate adherence based on durations for which medications have been prescribed and/or dispensed. However, medications might not be prescribed or dispensed for a specific duration in all circumstances. Many healthcare settings allow for multiple refills of prescriptions, and medications might be dispensed in fixed pre-packed quantities rather than from bulk. AdhereR can compute event durations based on prescribed and dispensed quantities.

Various events might affect supply durations after medications have been dispensed. Prescribed dosage might change between dispensing events, changing the original supply duration. Treatments may be interrupted and resumed at later times, for which existing supplies may or may not be used. In contrast, patients might not use their own supplies during certain periods (e.g., when hospitalised). AdhereR takes into account dosage changes, special periods, and treatment interruptions.

This vignette describes the functions compute\_event\_durations and its arguments. We also discuss the computation of adherence with the output of compute\_event\_durations, introducing the functions prune\_event\_durations, cover\_special\_periods and time\_to\_initiation. We use the provided example datasets included in the dataset to illustrate the various options and their impact on the calculated durations.

## Definitions

Throughout AdhereR, we use the same terms and definitions. For a complete list, you may refer to the [AdhereR: Adherence to Medications](https://cran.r-project.org/package=AdhereR/vignettes/AdhereR-overview.html) vignette. Here we reiterate a selection of those terms and describe additional terms relevant for the context of this function:

* *CMA* = continuous multiple-interval measures of medication availability/gaps, representing various indicators of the quality of implementation,
* *Medication event* = prescribing or dispensing record of a given medication for a given patient; usually includes the patient’s unique identifier, an event date, and a duration,
* *Duration* = number of days the quantity of supplied medication would last if used as recommended,
* *Quantity* = dose supplied at a medication event,
* *Daily dosage* = dose recommended to be taken daily,
* *Medication class* = classification performed by the researcher depending on study aims, e.g. based on therapeutic use, mechanism of action, chemical molecule or pharmaceutical formulation,
* *Dosage change* = adjustment of the dose recommended to be taken daily,
* *Initial prescription* = first prescription event recorded in the dataset,
* *Prescription renewal* = subsequent prescription events after initial prescription,
* *Prescription duration* = number of days for which the medication should be used as prescribed,
* *Treatment interruption* = stop of prescription for a period of time before prescription renewal,
* *Prescription episode* = period of prescribed use delimited by start and end date of prescribed use

## Input data

AdhereR is designed to use datasets that have already been extracted and prepared for processing, as described in the package vignette. To compute event durations, at least two separate datasets are required: *Dispensing events* and *Prescription events*. Additionally, periods for which medication is supplied but not documented in the dispensing dataset can be provided (e.g. *hospitalisation events*). Each of those datasets might require specific preparation steps to bring them into the format described below.

### Dispensing data

The minimum necessary dataset includes 4 variables for each dispensing event: *patient unique identifier*, *dispensing event date*, *medication type*, and *dispensed quantity*.

*Medication type* can include multiple columns, which allows to distinguish medications on multiple levels. For example, one might want to differentiate between dosage forms of the same substance (e.g., tablets and inhalers of corticosteroids) and include both columns (substance and form) to describe *medication type*. The *dispensed quantity* could be the number of units dispensed (e.g., tablets), or a total number of subunits contained in each dispensed unit (e.g., milligrams in tablets). If multiple dosage forms for the same substance exist, it is useful to calculate the subunits because the dispensed dosage per unit might not correspond to the prescribed dosage per unit. For example, a prescription for *40 mg Atorvastatin 1 tablet daily* might be dispensed as 80 mg tablets with the instruction to use half of a tablet per day.

For demonstration purposes, we included a sample dataset containing dispensing events (one per row) for 16 patients over a period of roughly 24 months (1794 events in total). Each row represents an individual dispensing record for a specific dose of a specific medication for a patient at a given date. Six variables are included in this dataset:

* patient unique identifier (ID),
* dispensing event date (DATE.DISP; from 1 July 2056 to 12 July 2058, in the “yyyy-mm-dd” ISO format),
* medication type
  + ATC code (ATC.CODE; 49 different codes according to the Anatomical Therapeutic Chemical Classification [ATC] System),
  + dosage unit (UNIT; 57% MG, 12% MICROG, 31% UI),
  + dosage form (FORM; 12% INHALATION VAPOUR, 3% INJECTION, 13% METERED INHALER, 72% ORAL FORM), and
* dispensed quantity (TOTAL.DOSE; median 20,000, range 10-120,000,000).

[Table 1](#Table-1) shows the first 10 rows of the dispensing events in the example dataset durcomp.dispensing.

**Table 1:** First 10 rows of example dispensing data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | DATE.DISP | ATC.CODE | UNIT | FORM | TOTAL.DOSE |
| 1 | 2057-01-14 | A02BC05 | MG | ORAL FORM | 1120 |
| 1 | 2057-03-07 | A02BC05 | MG | ORAL FORM | 280 |
| 1 | 2056-10-03 | A02BC05 | MG | ORAL FORM | 1120 |
| 1 | 2056-12-03 | A02BC05 | MG | ORAL FORM | 1120 |
| 1 | 2057-08-04 | A02BC05 | MG | ORAL FORM | 1120 |
| 1 | 2058-02-09 | A02BC05 | MG | ORAL FORM | 1120 |

### Prescribing data

The minimum necessary dataset includes 4 variables for each prescription event: *patient unique identifier*, *prescription event date*, *medication type*, and *prescribed daily dose*. A *visit number* and *prescription duration* are optional.

*Medication type* can include multiple columns, corresponding to the columns in the *dispensing dataset*. A duration will only be calculated if the information in all columns for the medication type are the same in dispensing and prescription events. Similar to the *dispensed quantity*, the *prescribed daily dose* could be the number of units prescribed per day (e.g., 2 tablets), or a total dosage to be taken daily (e.g., 40 mg). If a medication is prescribed for regular but not daily use, the dosage should be recalculated, e.g. in case of *70 mg once per week*, the *prescribed daily dose* should be 10 mg.

**IMPORTANT TO NOTE: It is assumed that the prescribed daily dose can be accomodated with the dispensed medication, for example by splitting tablets. This requires careful consideration and exploratory analysis of the dispensed and prescribed dosage forms and posologies.**

For demonstration purposes, we included a sample dataset containing prescription events (one per row) for 16 patients over a period of roughly 15 months (1502 events in total). Each row represents an individual prescription record for a specific dose of a specific medication for a patient at a given date. Eight variables are included in this dataset:

* patient unique identifier (ID),
* prescription event date (DATE.PRESC; from 15 September 2056 to 30 December 2057, in the “yyyy-mm-dd” ISO format),
* visit number (VISIT; median 5, range 0-16),
* medication type
  + ATC code (ATC.CODE; 43 different codes according to the Anatomical Therapeutic Chemical Classification [ATC] System),
  + dosage unit (UNIT; 50% MG, 10% MICROG, 40% UI),
  + dosage form (FORM; 18% INHALATION VAPOUR, 5% INJECTION, 13% METERED INHALER, 64% ORAL FORM),
* prescription duration (PRESC.DURATION; median 30, range 30-90 days, 1437 not provided (NA), and
* prescribed daily dose (DAILY.DOSE; median 600, range 0.07-8000000).

[Table 2](#Table-2) shows the first 10 rows of the prescription events in the example dataset durcomp.prescribing.

**Table 2:** First 10 rows of example prescribing data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | DATE.PRESC | VISIT | ATC.CODE | FORM | UNIT | PRESC.DURATION | DAILY.DOSE |
| 1 | 2056-12-08 | 0 | A09AA02 | ORAL FORM | UI | NA | 36000 |
| 1 | 2057-02-23 | 1 | A09AA02 | ORAL FORM | UI | NA | 86000 |
| 1 | 2057-03-03 | 2 | A09AA02 | ORAL FORM | UI | NA | 86000 |
| 1 | 2057-03-18 | 3 | A09AA02 | ORAL FORM | UI | NA | 86000 |
| 1 | 2057-09-01 | 4 | A09AA02 | ORAL FORM | UI | NA | 86000 |
| 1 | 2057-09-24 | 5 | A09AA02 | ORAL FORM | UI | NA | 86000 |

### Special periods (optional)

During certain periods, medication use may differ from what is expected based on the available data. Typical examples of such periods are hospitalisations, holidays, incarcerations, or similar. If available, these periods can be taken into account during computation of event durations. The minimum required variables are: *patient unique identifier*, *start date*, and *end date* of special periods. Optional columns are *type* (indicating the type of special period), *customised instructions* how to handle a specific period (explained below), and *medication class* (from those specified in dispensing and prescription datasets).

For demonstration purposes, AdhereR uses a sample dataset containing hospitalization periods (one per row) for 10 patients over a period of roughly 18 months (28 events in total). Each row represents an individual hospitalisation period of a patient for whom event durations should be calculated. All column names must match the format provided in this example:

* patient unique identifier (ID),
* start date (DATE.IN; from 15 September 2056 to 23 November 2057, in the “yyyy-mm-dd” ISO format), and
* end date (DATE.OUT; from 22 September 2056 to 24 December 2057, in the “yyyy-mm-dd” ISO format)

[Table 3](#Table-3) shows the first 10 rows of the hospitalisation events in the example dataset durcomp.hospitalisation.

**Table 3:** First 10 rows of example hospitalisation data

|  |  |  |
| --- | --- | --- |
| ID | DATE.IN | DATE.OUT |
| 1 | 2057-03-03 | 2057-03-06 |
| 1 | 2057-09-01 | 2057-09-04 |
| 3 | 2057-03-04 | 2057-03-17 |
| 3 | 2057-03-26 | 2057-05-01 |
| 3 | 2057-06-15 | 2057-06-22 |
| 3 | 2057-08-04 | 2057-08-12 |

## Function arguments

The function arguments for compute\_event\_durations are:

* disp.data: A *data.frame* or *data.table* containing the dispensing events (see above). Must contain, at a minimum, the patient unique ID, one medication identifier, the dispensing 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* or *data.table* containing the prescribing events (see above). 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* or *data.table* containing the information about special periods (see above). 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. Optional columns are TYPE (indicating the type of special situation), customized instructions how to handle a specific period (see special.periods.mapping), and any of those specified in medication.class.colnames.
* special.periods.mapping: 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 end of the special period. With *custom*, the mapping has to be included in *special.periods.data*. See details below.
* ID.colname: A *string*, the name of the column in disp.data, resc.data, and special.periods.data containing the unique patient ID.
* 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).
* 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).
* 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).
* 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.
* 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.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.
* 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.
* 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). See details below.
* 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 separatly. See details below.
* trt.interruption: Can be either of *“continue”*, *“discard”*, *“carryover”*, or a *string*. It indicates how to handle durations during treatment interruptions (see special.periods.mapping). If *string*, the name of the (*character*) column in *disp.data* containing the information (*“continue”*, *“discard”*, or *“carryover”*) for each medication class separatly. See details below.
* 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 separatly. Important if carryover should be considered later on. See details below.
* 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.

In the following paragraphs, we discuss the various options regarding special periods and treatment interruption, prescription start and renewal, and dosage changes.

### Special periods mapping and treatment interruptions

During special periods and treatment interruptions, medication use may differ from usual conditions. The argument special.periods.mapping tells AdhereR what to do during such periods. Similarly, trt.interruption specifies handling of treatment interruptions (periods without prescription). There are 3 options that can be set globally:

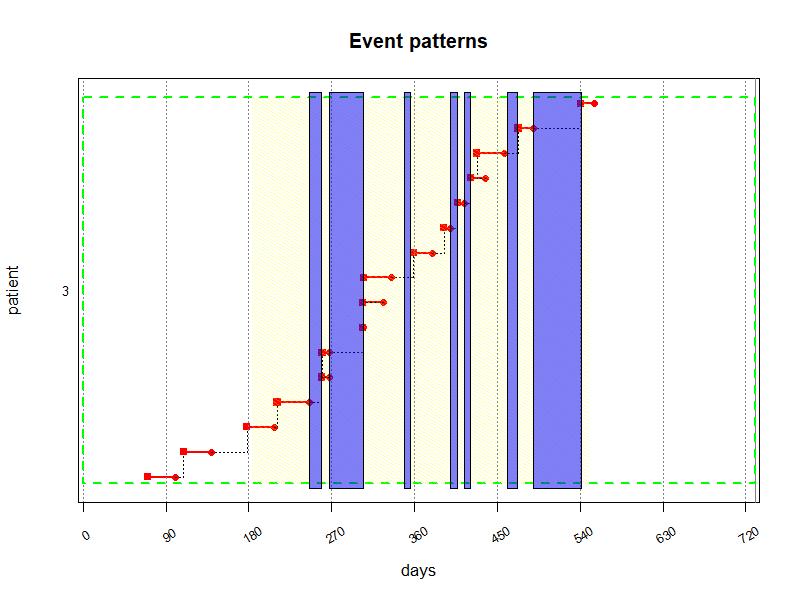
* *continue* has no impact on durations and dispensing start dates: Patients are expected to continue using the existing supply as initially prescribed,
* *discard* truncates supplies at the beginning of special period or treatment interruption and the remaining supply is discarded. This might be used if patients are asked to return unused medications after a limited treatment course (e.g. antibiotic treatments),
* *carryover* truncates supplies at the beginning of a special period or treatment interruption, but the remaining supply is carried over until the end of the interruption and a new event will be added for the remaining duration. This might be used if patients are hospitalized and receive medications from hospital wards but are expected to continue using their previously available supplies after discharge. Similarly, if patients have repeat prescriptions for short durations and are expected to use supplies from previous courses.

**IMPORTANT TO NOTE: When using this setting, the computed durations may need additional processing before CMA calculations (see examples below).**

In addition to the global options, both settings accept a column name in the dispensing dataset (for trt.interruption) or special periods dataset (for special.periods.mapping). The column can contain either of *continue*, *discard*, or *carryover* per medication type (for trt.interruption) or per special period type and/or medication type (for special.periods.mapping).

Special periods may occur during prescription episodes or treatment interruptions and different types of special periods may co-occur. Treatment interruptions are prioritized over other special periods: If a prescription ends and trt.interruption = 'carryover', a different setting in special.periods.mapping has no effect during periods of treatment interruption. However, if a special period of type *continue* overlaps with another special period of type *carryover*, the setting of the period with the later start date is used.

In [Figure 1](#Figure-1) below, the patient had frequent hospitalisation events (blue segments). By setting trt.interruptions = "carryover", supplies available before the start of hospitalisation are truncated and a new event is created on the day of discharge for the remaining supply.

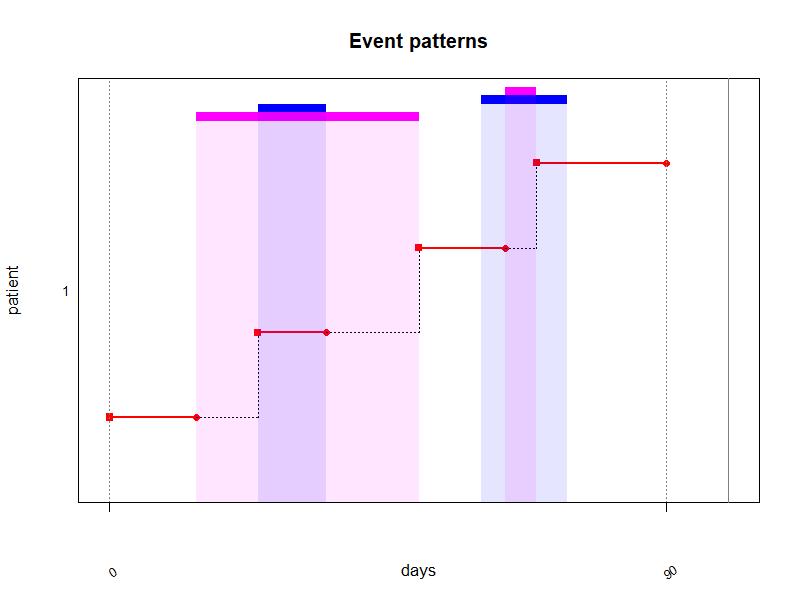


**Figure 1:** CMA0 calculated for a patient with trt.interruption = “carryover”

Let’s consider a hypothetical scenario with a lot of overlapping special periods.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | DATE.IN | DATE.OUT | TYPE | CUSTOM |
| 1 | 2000-01-15 | 2000-02-20 | HOSP | carryover |
| 1 | 2000-01-25 | 2000-02-05 | REHAB | continue |
| 1 | 2000-03-01 | 2000-03-15 | HOLIDAY | continue |
| 1 | 2000-03-05 | 2000-03-10 | HOSP | carryover |

[Figure 2](#Figure-2) shows a hypothetical patient with one prescription and one dispensing event for 60 days. By providing the above dataset as special.periods = "special\_episodes" and setting special.periods.mapping = "CUSTOM", the 60-day supply is truncated and restarted according to the most recent special period that hasn’t ended yet.



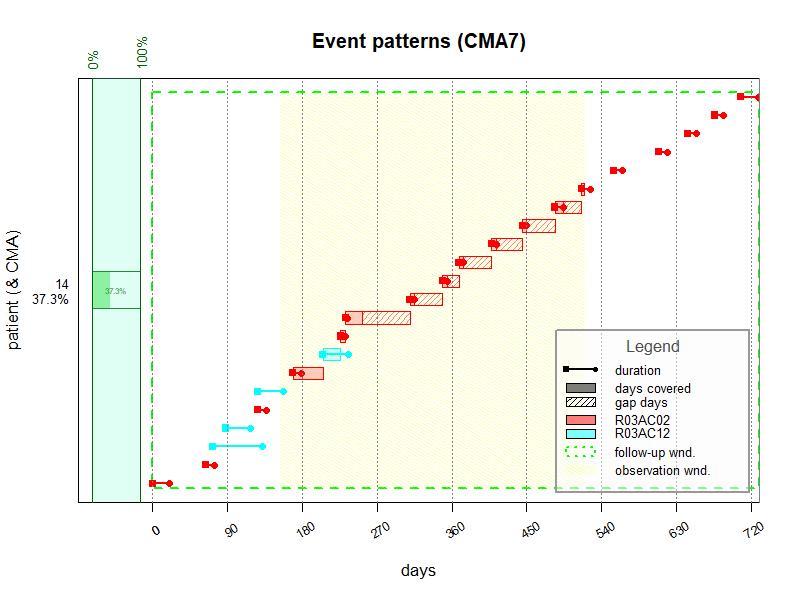
**Figure 2:** CMA0 calculated for a patient with different types of special periods and special.periods.mapping = “CUSTOM”. Bars at the top of the plot and shaded areas indicate special periods: Blue for ‘continue’ and Purple for ‘carryover’.

### Force initial prescription

If the dispensing dataset of a patient covers events with earlier dates than the first prescription events for this specific medication, force.init.presc = TRUE advances the date of the first prescription event to the date of the first dispensing event. For example, if prescribing data is only available during the observation window, but dispensing data covers a larger follow-up window, this setting allows the calculation of supply durations for carryover into the observation window.

**IMPORTANT TO NOTE: This only applies if the first prescription event is not limited in duration (as indicated in presc.duration.colname). This is to avoid assumptions about the true start date and duration of such prescriptions.**

In the example in [Figure 3](#Figure-3) below, the patient had dispensing events for Salbutamol (R03AC02) and Salmeterol (R03AC12) starting from 2056-07-31, but the first prescribing event for either of those was not before 2056-12-10. Because the prescriptions were not limited in duration, by setting force.init.presc = TRUE, durations for dispensing events before 2056-12-10 can be calculated.



**Figure 3:** CMA7 calculated for a patient with force.init.presc = TRUE

### Force prescription renewal

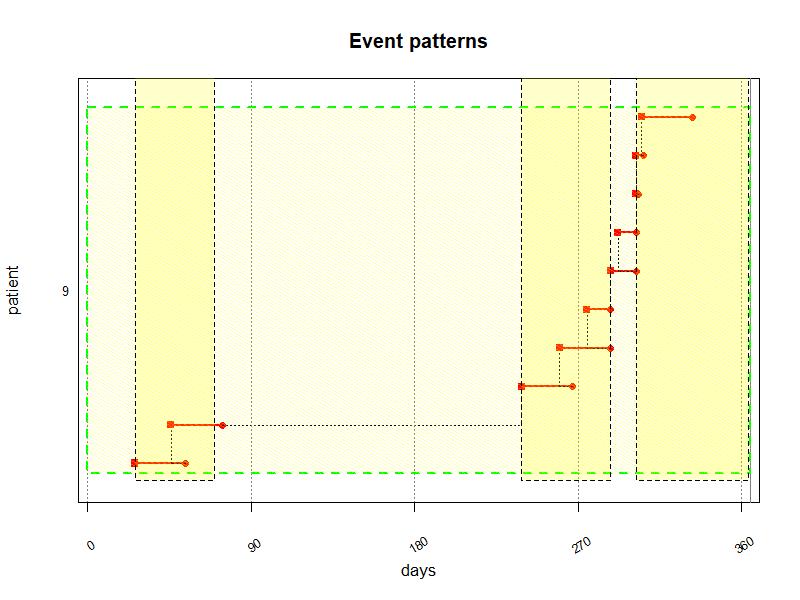
If a medication is not prescribed during any given visit for a patient, force.prescription.renew = TRUE will make sure that the prescription episode for this medication ends on the first visit without renewal. Alternatively, this can be set for each medication class separately by providing the name of a column containing the information in the dispensing dataset (logical, TRUE or FALSE).

If prescriptions are not routinely prescribed during all visits, force.prescription.renew should be set to FALSE. This can also be the case if prescription data covers multiple prescribers, because treatments prescribed by one prescriber might continue even when not prescribed during a subsequent visit to another prescriber.

In [Table 4](#Table-4) and [Figure 4](#Figure-4) below, the medication (a leukotriene receptor antagonist) was prescribed for a limited duration initially (during visit 2 and 3). Later, it was represcribed during visits 7, 8, 10, and 11, but not during visit 9. By setting force.presc.renew = TRUE, the prescription ends on the date of visit 9 and restarts on the date of visit 10.

**Table 4:** Prescription events for example patient.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | DATE.PRESC | VISIT | ATC.CODE | FORM | UNIT | PRESC.DURATION | DAILY.DOSE |
| 9 | 2057-01-27 | 2 | R03DC03 | ORAL FORM | MG | NA | 10 |
| 9 | 2057-02-10 | 3 | R03DC03 | ORAL FORM | MG | 30 | 10 |
| 9 | 2057-08-28 | 7 | R03DC03 | ORAL FORM | MG | NA | 10 |
| 9 | 2057-09-18 | 8 | R03DC03 | ORAL FORM | MG | NA | 10 |
| 9 | 2057-10-30 | 10 | R03DC03 | ORAL FORM | MG | NA | 10 |
| 9 | 2057-11-13 | 11 | R03DC03 | ORAL FORM | MG | NA | 10 |

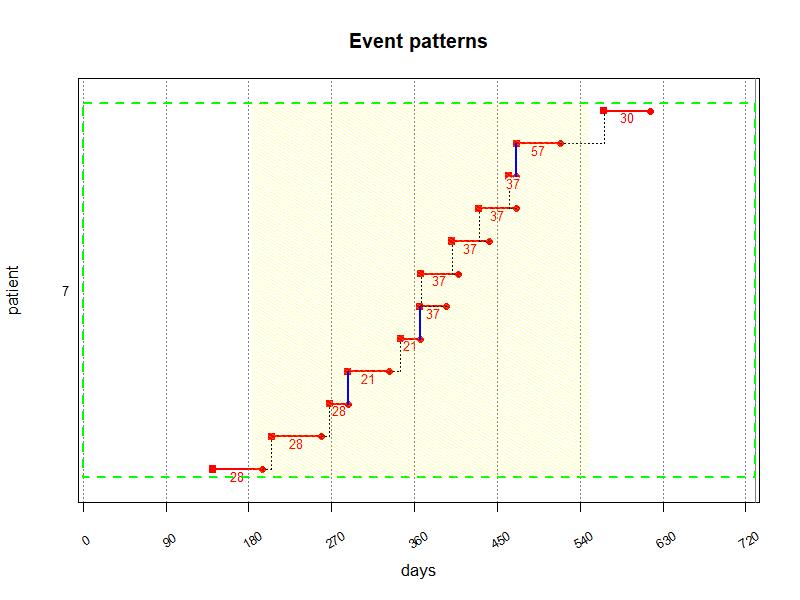


**Figure 4:** CMA0 calculated for a patient with force.presc.renew = TRUE

### Split on dosage change

If the dosage changes before the end of a supply duration, split.on.dosage.change = TRUE creates a new event on the day of dosage change and recalculates the duration for the remaining supply. If patients are expected to finish the previous supply with the original dose and starting with the new dosage recommendation from the next supply onward, split.on.dosage.change should be set to FALSE. Alternatively, this can be set for each medication class separately by providing the name of a column containing the information in the dispensing dataset (logical, TRUE or FALSE).

In [Figure 5](#Figure-5) below, the dosage for Insulin (NovoMix) changed while the patient still had an available supply. By setting split.on.dosage.change = TRUE, a new event is created on the day of dosage change (blue vertical lines).



**Figure 5:** CMA0 calculated for a patient with split.on.dosage.change = TRUE

## Output values

The output of compute\_event\_durations is a list containing all data required for CMA computations, plus additional information:

* 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.
  + disp.date.colname: the date of the dispensing event, as given by the disp.date.colnema parameter.
  + medication.class.colnames: the column(s) with classes/types/groups of medication, as given by the medication.class.colnames parameter.
  + total.dose.colname: the total dispensed quantity, 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/hospitalisations.
  + DURATION: the calculated duration of the supply, based on the total dispensed dose and the prescribed daily dose, starting from the DISP.START date.
  + 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 with the following columns:
  + ID.colname: the unique patient ID, as given by the ID.colname parameter.
  + DATE.IN: the start date of the special period
  + DATE.OUT: the end date of the special period
  + TYPE: optional, the type
  + CUSTOM: the special period mapping, either “carryover”, “continue”, or “discard”
  + SPECIAL.DURATION: the number of days between DATE.IN and DATE.OUT

In addition, the output contains all the arguments to the function call:

* special.periods.mapping
* ID.colname
* presc.date.colname
* disp.date.colname
* date.format
* medication.class.colnames
* total.dose.colname
* presc.daily.dose.colname
* presc.duration.colname
* visit.colname
* force.init.presc
* force.presc.renew
* trt.interruption
* split.on.dosage.change

# use compute\_event\_duration with the included example dataset and default settings  
durations <- compute\_event\_durations(disp.data = durcomp.dispensing, # the dispensing data  
 presc.data = durcomp.prescribing, # the prescribing data  
 special.periods.data = durcomp.hospitalisation, # the special periods data  
 ID.colname = "ID", # the unique Patient ID column in disp.data, presc.data, and special.periods.data  
 presc.date.colname = "DATE.PRESC", # the prescription date column  
 disp.date.colname = "DATE.DISP", # the dispensing date column  
 date.format = "%Y-%m-%d", # the date format  
 medication.class.colnames = c("ATC.CODE","UNIT", "FORM"), # the medication class columns  
 total.dose.colname = "TOTAL.DOSE", # the total dispensed dose column  
 presc.daily.dose.colname = "DAILY.DOSE", # the prescribed daily dose column  
 presc.duration.colname = "PRESC.DURATION", # the prescription duration column  
 visit.colname = "VISIT", # the prescribing event visit column  
 return.data.table = TRUE, # return data.table for better printing  
 progress.bar = FALSE) # don't display progress bar  
  
# display output  
durations

## $event\_durations  
## ID DATE.DISP ATC.CODE UNIT FORM TOTAL.DOSE DAILY.DOSE  
## 1: 1 2057-09-04 A02BC02 MG ORAL FORM 560 20  
## 2: 1 2058-06-03 A02BC02 MG ORAL FORM 560 20  
## 3: 1 2058-07-09 A02BC02 MG ORAL FORM 560 20  
## 4: 1 2056-08-05 A02BC05 MG ORAL FORM 1120 NA  
## 5: 1 2056-10-03 A02BC05 MG ORAL FORM 1120 NA  
## ---   
## 1443: 16 2058-06-17 R03AC12 MICROG METERED INHALER 3000 50  
## 1444: 16 2058-07-12 R03AC12 MICROG METERED INHALER 3000 50  
## 1445: 16 2057-11-02 R03AK06 MICROG METERED INHALER 30000 500  
## 1446: 16 2058-01-04 R03AK06 MICROG METERED INHALER 30000 500  
## 1447: 16 2058-04-20 R03AK06 MICROG METERED INHALER 60000 500  
## DISP.START DURATION episode.start episode.end SPECIAL.DURATION  
## 1: 2057-09-04 28 2057-09-01 <NA> 0  
## 2: 2058-06-03 28 2057-09-01 <NA> 0  
## 3: 2058-07-09 28 2057-09-01 <NA> 0  
## 4: <NA> NA <NA> <NA> NA  
## 5: <NA> NA <NA> <NA> NA  
## ---   
## 1443: 2058-06-17 60 2057-11-23 <NA> 0  
## 1444: 2058-07-12 60 2057-11-23 <NA> 0  
## 1445: 2057-11-02 60 2057-11-02 <NA> 0  
## 1446: 2058-01-04 60 2057-11-02 <NA> 0  
## 1447: 2058-04-20 120 2057-11-02 <NA> 0  
## tot.dosage.changes CARRYOVER.DURATION  
## 1: 0 NA  
## 2: 0 NA  
## 3: 0 NA  
## 4: NA NA  
## 5: NA NA  
## ---   
## 1443: 0 NA  
## 1444: 0 NA  
## 1445: 0 NA  
## 1446: 0 NA  
## 1447: 0 NA  
##   
## $prescription\_episodes  
## ID ATC.CODE UNIT FORM DAILY.DOSE episode.ID  
## 1: 1 A02BC02 MG ORAL FORM 20.000 1  
## 2: 1 A09AA02 UI ORAL FORM 36000.000 1  
## 3: 1 A09AA02 UI ORAL FORM 86000.000 2  
## 4: 1 R03AC12 MICROG METERED INHALER 100.000 1  
## 5: 2 A11CC05 UI ORAL FORM 1111.111 1  
## ---   
## 253: 16 J01MA02 MG ORAL FORM 1000.000 1  
## 254: 16 J02AC02 MG ORAL FORM 400.000 1  
## 255: 16 R03AC02 MG INHALATION VAPOUR 5.000 1  
## 256: 16 R03AC12 MICROG METERED INHALER 50.000 1  
## 257: 16 R03AK06 MICROG METERED INHALER 500.000 1  
## episode.start episode.duration episode.end  
## 1: 2057-09-01 NA <NA>  
## 2: 2056-12-08 77 2057-02-23  
## 3: 2057-02-23 NA <NA>  
## 4: 2057-02-23 NA <NA>  
## 5: 2056-09-22 NA <NA>  
## ---   
## 253: 2057-10-12 30 2057-11-11  
## 254: 2057-11-23 NA <NA>  
## 255: 2056-11-17 NA <NA>  
## 256: 2057-11-23 NA <NA>  
## 257: 2057-11-02 NA <NA>  
##   
## $special\_periods  
## ID DATE.IN DATE.OUT SPECIAL.DURATION  
## 1: 1 2057-03-03 2057-03-06 3  
## 2: 1 2057-09-01 2057-09-04 3  
## 3: 3 2057-03-04 2057-03-17 13  
## 4: 3 2057-03-26 2057-05-01 36  
## 5: 3 2057-06-15 2057-06-22 7  
## 6: 3 2057-08-04 2057-08-12 8  
## 7: 3 2057-08-19 2057-08-26 7  
## 8: 3 2057-10-05 2057-10-16 11  
## 9: 3 2057-11-02 2057-12-24 52  
## 10: 4 2057-07-15 2057-07-16 1  
## 11: 4 2057-11-23 2057-11-30 7  
## 12: 6 2056-09-15 2056-09-22 7  
## 13: 6 2057-03-11 2057-03-18 7  
## 14: 6 2057-10-14 2057-10-21 7  
## 15: 7 2057-03-12 2057-03-19 7  
## 16: 7 2057-05-14 2057-05-21 7  
## 17: 7 2057-07-17 2057-07-31 14  
## 18: 9 2057-04-09 2057-04-10 1  
## 19: 9 2057-05-25 2057-06-26 32  
## 20: 9 2057-09-02 2057-09-04 2  
## 21: 10 2057-03-27 2057-04-03 7  
## 22: 10 2057-07-13 2057-07-20 7  
## 23: 12 2057-07-07 2057-07-13 6  
## 24: 14 2057-01-16 2057-01-27 11  
## 25: 14 2057-02-11 2057-02-13 2  
## 26: 14 2057-02-24 2057-02-27 3  
## 27: 14 2057-10-26 2057-10-30 4  
## 28: 15 2057-09-07 2057-09-10 3  
## ID DATE.IN DATE.OUT SPECIAL.DURATION  
##   
## $special.periods.mapping  
## [1] "continue"  
##   
## $ID.colname  
## [1] "ID"  
##   
## $presc.date.colname  
## [1] "DATE.PRESC"  
##   
## $disp.date.colname  
## [1] "DATE.DISP"  
##   
## $date.format  
## [1] "%Y-%m-%d"  
##   
## $medication.class.colnames  
## [1] "ATC.CODE" "UNIT" "FORM"   
##   
## $total.dose.colname  
## [1] "TOTAL.DOSE"  
##   
## $presc.daily.dose.colname  
## [1] "DAILY.DOSE"  
##   
## $presc.duration.colname  
## [1] "PRESC.DURATION"  
##   
## $visit.colname  
## [1] "VISIT"  
##   
## $force.init.presc  
## [1] FALSE  
##   
## $force.presc.renew  
## [1] FALSE  
##   
## $trt.interruption  
## [1] "continue"  
##   
## $split.on.dosage.change  
## [1] TRUE

## Computing CMA with the output of compute\_event\_durations

In principle the output of compute\_event\_durations can be used for CMA computation as described in the main vignette. However, there are some specificities to consider.

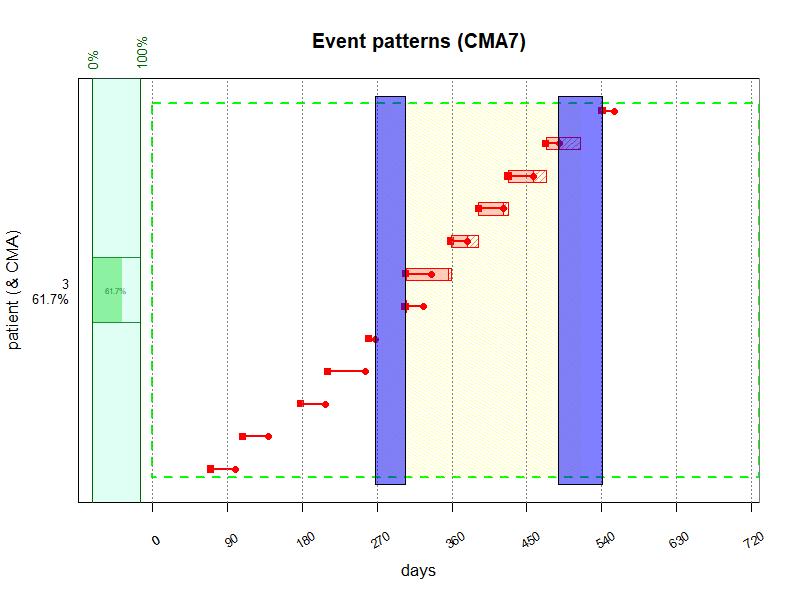
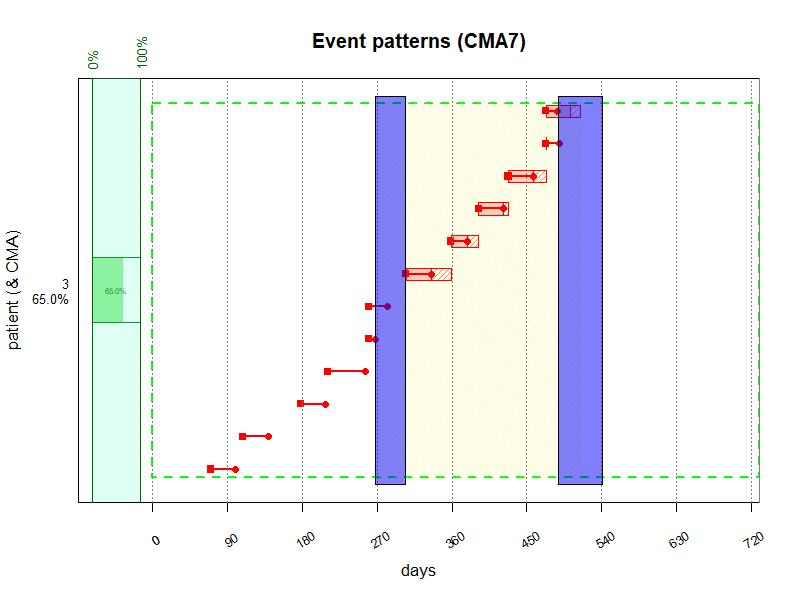
### Medication class

In compute\_event\_durations, multiple columns to specify medication classes can be provided. This is especially useful when different formulations or brands of the same medication need to be matched between dispensing and prescribing data. This way, polypharmacy regimens with multiple different treatments per patient can be processed. For CMA computations, only one column can be used for the medication type. If information from multiple columns should be considered, the content of these columns can be pasted together in a new column.

### Event date

The event\_durations dataset has two columns that can be used as event date: disp.date.colname, the original dispensing date in the dispensing dataset or DISP.START, which might differ from the former in case of dosage changes, treatment interruptions or special periods of type *carryover*. For CMA versions accounting for carryover, there will be no difference between the two choices, as long as the events take place within the observation period. If an event start gets pushed out of the observation period (e.g., because of carryover during a special period), this will affect CMA calculation. Generally, it is appropriate to use DISP.START as input for CMA calculations.

[Figure 6](#Figure-6) shows CMA7-plots of the same event\_durations, on top with event.date.colname = "DATE.DISP" and below with event.date.colname = "DISP.START". The observation window (OW) begins and ends during special periods of type *carryover* (blue areas). With event.date.colname = "DISP.START", part of a supply is pushed into the OW and another supply is partially pushed out of the OW.



### Prune event durations after carryover

Special periods and treatment interruptions of type *carryover* 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 do in fact discontinue treatments after the end of a special period or treatment interruption.

To detect whether new dispensing events occur shortly after the end of a special period or treatment interruption in spite of a remaining supply, AdhereR offers the prune\_event\_durations function. It accepts the raw list output of compute\_event\_durations and additional arguments to specify event durations that need to be removed:

* data: a *list*, the output of compute\_event\_durations,
* include: indicates whether to include special periods and/or treatment interruptions,
* medication.class.colnames: indicate columns in event\_durations to identify medication classes. Defaults to the columns used in compute\_event\_durations,
* days.within.out.date.1: event durations from before the special period or treatment interruptions are removed if there is a new dispensing event within the number of days after the end of a special period. If integer the number of days, if string the column name containing the number of days,
* days.within.out.date.2: event durations from before the special period are removed if there is *NO* new dispensing event within the number of days after the end of a special period. If integer the number of days, if string the column name containing the number of days,
* keep.all: *Logical*, should events be kept and marked for removal? If TRUE, a new column .prune.event will be added to the event\_durations, if FALSE these events will be removed.

The function output is the pruned event\_durations dataset.

### Consider special periods as covered

Without further processing of event\_durations for CMA computations, special periods will appear as gaps, possibly leading to underestimation of implementation or even assumption of discontinuation and non-persistence. To consider such periods as covered, they can be added to the event\_durations dataset, for example when it is assumed that hospitalized patients are adherent during the hospitalization period. This can be achieved by merging the special periods with the event\_durations dataset. This should be done after pruning with prune\_event\_durations.

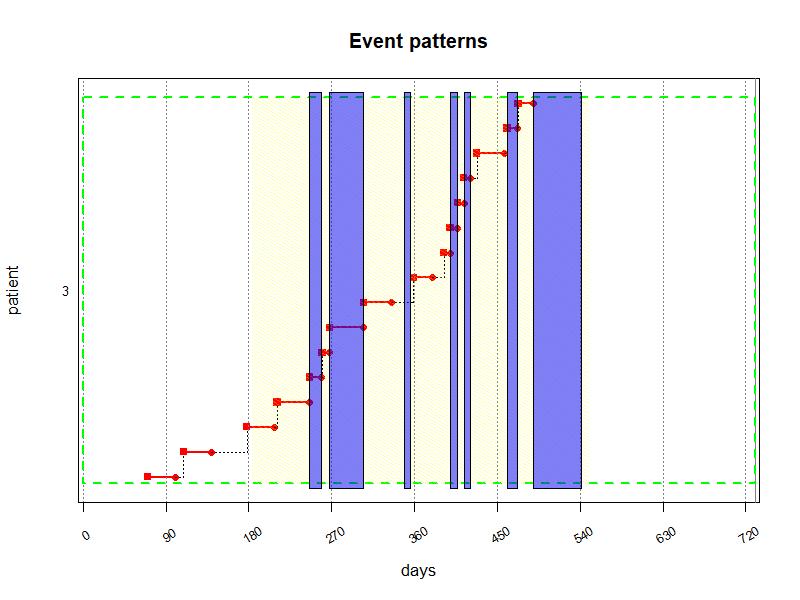
AdhereR offers the cover\_special\_periods function to identify special periods that are in proximity to already covered durations and adds an additional event for these durations. It requires at least the following arguments:

* events.data: a *data.frame* or *data.table* with the event durations,
* special.periods.data: a *data.frame* or *data.table* with the special periods,
* ID.colname: the unique patient ID,
* disp.start.colname: the start date of the medication event,
* duration.colname: the calculated duration for the medication event,
* medication.class.colnames: the column(s) with classes/types/groups of medication,
* 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.forma: a *string* with the date format.

The function output is the event.data dataset with the additional durations for special periods covered.

# select medication class of interest and compute event durations  
event\_durations\_list <- compute\_event\_durations(disp.data = durcomp.dispensing[ID == 3 & grepl("J01EE01", ATC.CODE)],  
 presc.data = durcomp.prescribing[ID == 3 & grepl("J01EE01", ATC.CODE)],  
 special.periods.data = durcomp.hospitalisation[ID == 3],  
 special.periods.mapping = "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",  
 suppress.warnings = FALSE,  
 return.data.table = TRUE,  
 progress.bar = FALSE)  
  
# prune dataset  
event\_durations <- prune\_event\_durations(event\_durations\_list,  
 include = c("special periods"), # only consider special periods  
 medication.class.colnames = "ATC.CODE",   
 days.within.out.date.1 = 7, # flag carryover durations if there are new events within 7 days after the end of special periods   
 days.within.out.date.2 = 30, # flag carryover durations if there are no new events within 30 days after the end of special periods   
 keep.all = FALSE # remove flagged events from dataset  
 )  
  
# cover special periods  
event\_durations\_covered <- cover\_special\_periods(events.data = event\_durations,  
 special.periods.data = event\_durations\_list$special\_periods,  
 ID.colname = "ID",  
 disp.start.colname = "DISP.START",  
 duration.colname = "DURATION",  
 medication.class.colnames = "ATC.CODE",  
 days.before = 7,  
 days.after = 7,  
 date.format = "%Y-%m-%d"  
 )

[Figure 7](#Figure-7) shows the same plot as [Figure 1](#Figure-1), but with pruned durations and added events for special durations.



**Figure 7:** CMA0 calculated for a patient with trt.interruption = “carryover”, carryover events pruned and special durations covered.

### Exclude special periods and treatment interruptions from CMA computations

During periods without prescriptions (treatment interruptions) or certain special episodes, it might not be meaninful to calculate a CMA at all. In these instances, it might be advisable to use CMA\_per\_episode with precomputed episodes, e.g. prescription episodes from the output of compute\_event\_durations. Precomputed episodes can be specified with the treat.epi parameter in CMA\_per\_episode. The episodes have to be a data.frame or data.table with the following columns:

* ID.colname: the patient ID,
* medication.class.colname: the medication class used for CMA computations,
* episode.ID: the episode unique ID (increasing sequentially),
* episode.start: the episode start date,
* episode.duration: the episode duration in days,
* episode.end: the episode end date.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | episode.ID | episode.start | end.episode.gap.days | episode.duration | episode.end | CMA |
| 3 | 1 | 2056-09-09 | 8.000000 | 159 | 2057-06-09 | 0.8113208 |
| 3 | 2 | 2057-06-09 | 46.000000 | 206 | 2058-01-01 | 0.5339806 |
| 5 | 1 | 2057-02-23 | 9.666667 | 30 | 2057-03-25 | 0.4444444 |
| 6 | 1 | 2057-01-16 | 4.000000 | 30 | 2057-02-15 | 0.8333333 |
| 6 | 2 | 2057-12-15 | NA | 17 | 2058-01-14 | 0.8823529 |
| 7 | 1 | 2057-12-10 | 8.000000 | 22 | 2058-01-09 | 0.4545455 |
| 9 | 1 | 2057-01-07 | NA | 30 | 2057-02-06 | NA |
| 9 | 2 | 2057-04-10 | 0.000000 | 30 | 2057-05-10 | 1.0000000 |
| 10 | 1 | 2057-01-06 | NA | 30 | 2057-02-05 | NA |
| 10 | 2 | 2057-04-21 | 0.000000 | 30 | 2057-05-21 | 1.0000000 |
| 10 | 3 | 2057-06-09 | 15.666667 | 30 | 2057-07-09 | 0.4444444 |
| 10 | 4 | 2057-08-21 | 0.000000 | 30 | 2057-09-20 | 0.1666667 |
| 12 | 1 | 2057-03-30 | 0.000000 | 30 | 2057-04-29 | 0.5000000 |
| 12 | 2 | 2057-07-07 | 20.000000 | 30 | 2057-08-06 | 0.3333333 |
| 13 | 1 | 2056-12-12 | 8.000000 | 10 | 2057-01-11 | 0.2000000 |
| 13 | 2 | 2057-04-23 | 53.000000 | 253 | 2058-01-01 | 0.2964427 |
| 14 | 1 | 2056-07-31 | 0.000000 | 258 | 2057-09-16 | 0.3643411 |
| 14 | 2 | 2057-09-16 | 2.000000 | 30 | 2057-10-16 | 0.9333333 |
| 16 | 1 | 2057-09-22 | 0.000000 | 48 | 2057-11-09 | 0.6875000 |
| 16 | 2 | 2057-11-09 | 3.000000 | 30 | 2057-12-09 | 0.9000000 |

CMA calculations for precomputed episodes do not necessarily reflect implementation. Delayed initiation and early discontinuation (non-persistence) may reduce CMA values. Non-persistence can be identified by looking at end.episode.gap.days in the output of CMA\_per\_episodes, which indicates how many days before the end of an episode are not covered.

Delayed initiation becomes evident when we calculate time to initiation for the same data. 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. It requires at least the following input:

* 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.
* 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).
* 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.

Its output is 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 start date of a prescription episode,
* first.disp: the date of the first dispensing event in an episode,
* time.to.initiation: the difference in days between the first dispensing date and the first prescription date per episode.

time\_init <- time\_to\_initiation(presc.data = prescription\_episodes,  
 disp.data = event\_durations,  
 ID.colname = "ID",  
 presc.start.colname = "episode.start",  
 disp.date.colname = "DATE.DISP",  
 medication.class.colnames = c("ATC.CODE"),  
 date.format = "%Y-%m-%d",  
 suppress.warnings = FALSE,  
 return.data.table = TRUE)  
  
knitr::kable(time\_init)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | ATC.CODE | episode.start | first.disp | time.to.initiation |
| 3 | J01EE01 | 2056-09-09 | 2056-09-09 | 0 |
| 3 | J01EE01 | 2057-06-09 | 2057-06-25 | 16 |
| 5 | J01EE01 | 2057-02-23 | 2057-03-02 | 7 |
| 6 | J01EE01 | 2057-01-16 | 2057-01-17 | 1 |
| 6 | J01EE01 | 2057-12-15 | 2057-12-16 | 1 |
| 7 | J01EE01 | 2057-12-10 | 2057-12-14 | 4 |
| 9 | J01EE01 | 2057-04-10 | NA | NA |
| 9 | J01EE01 | 2057-01-07 | 2057-03-19 | 71 |
| 10 | J01EE01 | 2057-01-06 | 2057-02-18 | 43 |
| 10 | J01EE01 | 2057-04-21 | 2057-04-24 | 3 |
| 10 | J01EE01 | 2057-06-09 | 2057-06-10 | 1 |
| 10 | J01EE01 | 2057-08-21 | 2057-09-15 | 25 |
| 11 | J01EE01 | 2057-10-19 | NA | NA |
| 12 | J01EE01 | 2057-07-07 | NA | NA |
| 12 | J01EE01 | 2057-03-30 | 2057-04-14 | 15 |
| 13 | J01EE01 | 2056-12-12 | 2056-12-24 | 12 |
| 13 | J01EE01 | 2057-04-23 | 2057-04-25 | 2 |
| 14 | J01EE01 | 2057-09-16 | NA | NA |
| 14 | J01EE01 | 2056-07-31 | 2056-07-31 | 0 |
| 16 | J01EE01 | 2057-09-22 | 2057-09-22 | 0 |
| 16 | J01EE01 | 2057-11-09 | 2057-11-09 | 0 |

## Conclusions

* functions presented here are compute\_event\_durations, prune\_event\_durations, cover\_special\_periods, and time\_to\_initiation to prepare data for calculation of TE, CMA, and eventually time\_to\_inititation
* the default use is when….
* Special cases are….
* Careful when using them for …