Package 'rxode2et'

January 30, 2024

Title Event Table Functions for 'rxode2'

Version 2.0.12

Description Provides the event table and support functions needed for
'rxode2' (Wang, Hallow and James (2016) <doi:10.1002/psp4.12052>) and 'nlmixr2'
(Fidler et al (2019) <doi:10.1002/psp4.12445>).
This split will reduce computational burden of recompiling 'rxode2'.

License GPL (>= 3)

URL https://nlmixr2.github.io/rxode2et/,
https://github.com/nlmixr2/rxode2et/

BugReports https://github.com/nlmixr2/rxode2et/issues/

Depends R (>= 4.0.0)

Imports stats, utils, methods, Rcpp, checkmate, rxode2random (>=
2.0.13), rxode2parse (>= 2.0.18), cli, crayon, magrittr, lotri

Suggests testthat (>= 3.0.0), units, tibble, data.table, dplyr,
pillar, nlmixr2data, qs

LinkingTo rxode2random, rxode2parse, Rcpp

Biarch true

Config/testthat/edition 3

Encoding UTF-8

Language en-US

NeedsCompilation yes

RoxygenNote 7.3.1

Author Matthew L. Fidler [aut, cre] (<https://orcid.org/0000-0001-8538-6691>),
Wenping Wang [ctb],
Fuji Goro [ctb],
Morwenn [ctb],
Igor Kushnir [ctb],
Omar Elashkar [ctb]

Maintainer Matthew L. Fidler <matthew.fidler@gmail.com>

Repository CRAN

Date/Publication 2024-01-30 15:20:02 UTC

R topics documented:

.collectWarnings . . . 2
add.dosing . . . 3
add.sampling . . . 5
as.et . . . 7
et . . . 7
etExpand . . . 12
etRbind . . . 12
etRep . . . 15
etSeq . . . 17
eventTable . . . 20
is.rxStackData . . . 22
rxCbindStudyIndividual . . . 22
rxEtDispatchSolve . . . 24
rxEvid . . . 24
rxRateDur . . . 25
rxStack . . . 26
toTrialDuration . . . 27

Index **28**

`.collectWarnings` *Collect warnings and just warn once.*

Description

Collect warnings and just warn once.

Usage

```
.collectWarnings(expr, lst = FALSE)
```

Arguments

`expr`	R expression
`lst`	When `TRUE` return a list with list(object,warnings) instead of issuing the warnings. Otherwise, when `FALSE` issue the warnings and return the object.

Value

The value of the expression or a list with the value of the expression and a list of warning messages

Author(s)

Matthew L. Fidler

add.dosing *Add dosing to eventTable*

Description

This adds a dosing event to the event table. This is provided for piping syntax through magrittr. It can also be accessed by `eventTable$add.dosing(...)`

Usage

```
add.dosing(
  eventTable,
  dose,
  nbr.doses = 1L,
  dosing.interval = 24,
  dosing.to = 1L,
  rate = NULL,
  amount.units = NA_character_,
  start.time = 0,
  do.sampling = FALSE,
  time.units = NA_character_,
  ...
)
```

Arguments

`eventTable`	eventTable object; When accessed from object it would be `eventTable$`
`dose`	numeric scalar, dose amount in `amount.units`;
`nbr.doses`	integer, number of doses;
`dosing.interval`	required numeric scalar, time between doses in `time.units`, defaults to 24 of `time.units="hours"`;
`dosing.to`	integer, compartment the dose goes into (first compartment by default);
`rate`	for infusions, the rate of infusion (default is `NULL`, for bolus dosing;
`amount.units`	optional string indicating the dosing units. Defaults to `NA` to indicate as per the original `EventTable` definition.
`start.time`	required dosing start time;
`do.sampling`	logical, should observation sampling records be added at the dosing times? Defaults to `FALSE`.
`time.units`	optional string indicating the time units. Defaults to `"hours"` to indicate as per the original `EventTable` definition.
`...`	Other parameters passed to `et()`.

Value

eventTable with updated dosing (note the event table will be updated anyway)

Author(s)

Matthew L. Fidler

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

eventTable, add.sampling, add.dosing, et, etRep, etRbind

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)
```

```
## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function

qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

`add.sampling` *Add sampling to eventTable*

Description

This adds a dosing event to the event table. This is provided for piping syntax through magrittr. It can also be accessed by `eventTable$add.sampling()`

Usage

```
add.sampling(eventTable, time, time.units = NA)
```

Arguments

`eventTable`	An eventTable object. When accessed from object it would be `eventTable$`
`time`	a vector of time values (in `time.units`).
`time.units`	an optional string specifying the time units. Defaults to the units specified when the `EventTable` was initialized.

Value

eventTable with updated sampling. (Note the event table will be updated even if you don't reassign the eventTable)

Author(s)

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

eventTable, add.sampling, add.dosing, et, etRep, etRbind

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)


## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function
```

```
qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

`as.et` *Coerce object to data.frame*

Description

Coerce object to data.frame

Usage

```
as.et(x, ...)

## Default S3 method:
as.et(x, ...)
```

Arguments

`x`	Object to coerce to et.
`...`	Other parameters

Value

An event table

`et` *Event Table Function*

Description

Event Table Function

Usage

```
et(x, ..., envir = parent.frame())

## S3 method for class 'rxode2'
et(x, ..., envir = parent.frame())

## S3 method for class '`function`'
et(x, ..., envir = parent.frame())

## S3 method for class 'rxUi'
et(x, ..., envir = parent.frame())

## S3 method for class 'rxSolve'
et(x, ..., envir = parent.frame())

## S3 method for class 'rxParams'
et(x, ..., envir = parent.frame())

## Default S3 method:
et(
  x,
  ...,
  time,
  amt,
  evid,
  cmt,
  ii,
  addl,
  ss,
  rate,
  dur,
  until,
  id,
  amountUnits,
  timeUnits,
  addSampling,
  envir = parent.frame(),
  by = NULL,
  length.out = NULL
)
```

Arguments

x	This is the first argument supplied to the event table. This is named to allow et to be used in a pipe-line with arbitrary objects.
...	Times or event tables. They can also be one of the named arguments below.
envir	the environment in which expr is to be evaluated. May also be NULL, a list, a data frame, a pairlist or an integer as specified to sys.call.

`time` Time is the time of the dose or the sampling times. This can also be unspecified and is determined by the object type (list or numeric/integer).

`amt` Amount of the dose. If specified, this assumes a dosing record, instead of a sampling record.

`evid` Event ID; This can be:

Numeric Value	Description
0	An observation. This can also be specified as `evid=obs`
1	A dose observation. This can also be specified as `evid=dose`
2	A non-dose event. This can also be specified as `evid=other`
3	A reset event. This can also be specified as `evid=reset`.
4	Dose and reset event. This can also be specified as `evid=doseReset` or `evid=resetDose`

Note a reset event resets all the compartment values to zero and turns off all infusions.

`cmt` Compartment name or number. If a number, this is an integer starting at 1. Negative compartments turn off a compartment. If the compartment is a name, the compartment name is changed to the correct state/compartment number before running the simulation. For a compartment named "-cmt" the compartment is turned off.

```
Can also specify `cmt` as `dosing.to`,
`dose.to`, `doseTo`, `dosingTo`, and
`state`.
```

`ii` When specifying a dose, this is the inter-dose interval for `ss`, `addl` and `until` options (described below).

`addl` The number of additional doses at a inter-dose interval after one dose.

`ss` Steady state flag; It can be one of:

Value	Description
0	This dose is not a steady state dose
1	This dose is a steady state dose with the between/inter-dose interval of `ii`
2	Superposition steady state

When `ss=2` the steady state dose that uses the super-position principle to allow more complex steady states, like 10 mg in the morning and 20 mg at night, or dosing at 8 am 12 pm and 8 pm instead of every 12 hours. Since it uses the super positioning principle, it only makes sense when you know the kinetics are linear.

All other values of `SS` are currently invalid.

`rate` When positive, this is the rate of infusion. Otherwise:

Value	Description
0	No infusion is on this record

-1 Modeled rate (in rxode2:`rate(cmt) =`); Can be `et(rate=model)`.

-2 Modeled duration (in rxode2: `dur(cmt) =`); Can be`et(dur=model)` or `et(rate=dur)`.

When a modeled bioavailability is applied to positive rates (`rate` > 0), the duration of infusion is changed. This is because the data specify the rate and amount, the only think that modeled bioavailability can affect is duration.

If instead you want the modeled bioavailability to increase the rate of infusion instead of the duration of infusion, specify the `dur` instead or model the duration with `rate=2`.

`dur` Duration of infusion. When `amt` and `dur` are specified the rate is calculated from the two data items. When `dur` is specified instead of `rate`, the bioavailability changes will increase rate instead of duration.

`until` This is the time until the dosing should end. It can be an easier way to figure out how many additional doses are needed over your sampling period.

`id` A integer vector of IDs to add or remove from the event table. If the event table is identical for each ID, then you may expand it to include all the IDs in this vector. All the negative IDs in this vector will be removed.

`amountUnits` The units for the dosing records (`amt`)

`timeUnits` The units for the time records (`time`)

`addSampling` This is a boolean indicating if a sampling time should be added at the same time as a dosing time. By default this is `FALSE`.

`by` number: increment of the sequence.

`length.out` desired length of the sequence. A non-negative number, which for `seq` and `seq.int` will be rounded up if fractional.

Value

A new event table

Author(s)

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

`eventTable`, `add.sampling`, `add.dosing`, `et`, `etRep`, `etRbind`

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)


## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function

qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

etExpand *Expand additional doses*

Description

Expand additional doses

Usage

```
etExpand(et)
```

Arguments

et	Event table to expand additional doses for.

Value

New event table with addl doses expanded

Author(s)

Matthew Fidler

Examples

```
ev <- et(amt = 3, ii = 24, until = 240)
print(ev)
etExpand(ev) # expands event table, but doesn't modify it

print(ev)

ev$expand() ## Expands the current event table and saves it in ev
```

etRbind *Combining event tables*

Description

Combining event tables

Usage

```
etRbind(
  ...,
  samples = c("use", "clear"),
  waitII = c("smart", "+ii"),
  id = c("merge", "unique")
)

## S3 method for class 'rxEt'
rbind(..., deparse.level = 1)
```

Arguments

`...`	The event tables and optionally time between event tables, called waiting times in this help document.
`samples`	How to handle samples when repeating an event table. The options are: • `"clear"` Clear sampling records before combining the datasets • `"use"` Use the sampling records when combining the datasets
`waitII`	This determines how waiting times between events are handled. The options are: • `"smart"` This "smart" handling of waiting times is the default option. In this case, if the waiting time is above the last observed inter-dose interval in the first combined event table, then the actual time between doses is given by the wait time. If it is smaller than the last observed inter-dose interval, the time between event tables is given by the inter-dose interval + the waiting time between event tables. • `"+ii"` In this case, the wait time is added to the inter-dose interval no matter the length of the wait time or inter-dose interval
`id`	This is how rbind will handle IDs. There are two different types of options: • `merge` with `id="merge"`, the IDs are merged together, overlapping IDs would be merged into a single event table. • `unique` with `id="unique"`, the IDs will be renumbered so that the IDs in all the event tables are not overlapping.
`deparse.level`	The `deparse.level` of a traditional `rbind` is ignored.

Value

An event table

Author(s)

Matthew L Fidler

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

eventTable, add.sampling, add.dosing, et, etRep, etRbind

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)


## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function
```

```
qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

---

etRep *Repeat an rxode2 event table*

Description

Repeat an rxode2 event table

Usage

```
etRep(
  x,
  times = 1,
  length.out = NA,
  each = NA,
  n = NULL,
  wait = 0,
  id = integer(0),
  samples = c("clear", "use"),
  waitII = c("smart", "+ii"),
  ii = 24
)

## S3 method for class 'rxEt'
rep(x, ...)
```

Arguments

x	An rxode2 event table
times	Number of times to repeat the event table
length.out	Invalid with rxode2 event tables, will throw an error if used.
each	Invalid with rxode2 event tables, will throw an error if used.
n	The number of times to repeat the event table. Overrides times.
wait	Waiting time between each repeated event table. By default there is no waiting, or wait=0
id	A integer vector of IDs to add or remove from the event table. If the event table is identical for each ID, then you may expand it to include all the IDs in this vector. All the negative IDs in this vector will be removed.

samples — How to handle samples when repeating an event table. The options are:

- "clear" Clear sampling records before combining the datasets
- "use" Use the sampling records when combining the datasets

waitII — This determines how waiting times between events are handled. The options are:

- "smart" This "smart" handling of waiting times is the default option. In this case, if the waiting time is above the last observed inter-dose interval in the first combined event table, then the actual time between doses is given by the wait time. If it is smaller than the last observed inter-dose interval, the time between event tables is given by the inter-dose interval + the waiting time between event tables.
- "+ii" In this case, the wait time is added to the inter-dose interval no matter the length of the wait time or inter-dose interval

ii — When specifying a dose, this is the inter-dose interval for ss, addl and until options (described below).

... — Times or event tables. They can also be one of the named arguments below.

Value

An event table

Author(s)

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

eventTable, add.sampling, add.dosing, et, etRep, etRbind

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
```

```
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)


## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function

qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

---

etSeq *Sequence of event tables*

Description

This combines a sequence of event tables.

Usage

```
etSeq(..., samples = c("clear", "use"), waitII = c("smart", "+ii"), ii = 24)

## S3 method for class 'rxEt'
seq(...)
```

Arguments

`...`	The event tables and optionally time between event tables, called waiting times in this help document.
`samples`	How to handle samples when repeating an event table. The options are: • `"clear"` Clear sampling records before combining the datasets • `"use"` Use the sampling records when combining the datasets
`waitII`	This determines how waiting times between events are handled. The options are: • `"smart"` This "smart" handling of waiting times is the default option. In this case, if the waiting time is above the last observed inter-dose interval in the first combined event table, then the actual time between doses is given by the wait time. If it is smaller than the last observed inter-dose interval, the time between event tables is given by the inter-dose interval + the waiting time between event tables. • `"+ii"` In this case, the wait time is added to the inter-dose interval no matter the length of the wait time or inter-dose interval
`ii`	If there was no inter-dose intervals found in the event table, assume that the interdose interval is given by this `ii` value. By default this is `24`.

Details

This sequences all the event tables in added in the argument list `...`. By default when combining the event tables the offset is at least by the last inter-dose interval in the prior event table (or `ii`). If you separate any of the event tables by a number, the event tables will be separated at least the wait time defined by that number or the last inter-dose interval.

Value

An event table

Author(s)

Matthew L Fidler, Wenping Wang

References

Wang W, Hallow K, James D (2015). "A Tutorial on rxode2: Simulating Differential Equation Pharmacometric Models in R." CPT: Pharmacometrics and Systems Pharmacology, 5(1), 3-10. ISSN 2163-8306, <URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728294/>.

See Also

`eventTable`, `add.sampling`, `add.dosing`, `et`, `etRep`, `etRbind`

Examples

```
library(units)

## These are making the more complex regimens of the rxode2 tutorial

## bid for 5 days
bid <- et(timeUnits="hr") %>%
       et(amt=10000,ii=12,until=set_units(5, "days"))

## qd for 5 days
qd <- et(timeUnits="hr") %>%
      et(amt=20000,ii=24,until=set_units(5, "days"))

## bid for 5 days followed by qd for 5 days

et <- seq(bid,qd) %>% et(seq(0,11*24,length.out=100));


## Now Infusion for 5 days followed by oral for 5 days

##  note you can dose to a named compartment instead of using the compartment number
infusion <- et(timeUnits = "hr") %>%
      et(amt=10000, rate=5000, ii=24, until=set_units(5, "days"), cmt="centr")


qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(5, "days"), cmt="depot")

et <- seq(infusion,qd)


## 2wk-on, 1wk-off

qd <- et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- seq(qd, set_units(1,"weeks"), qd) %>%
     add.sampling(set_units(seq(0, 5.5,by=0.005),weeks))


## You can also repeat the cycle easily with the rep function

qd <-et(timeUnits = "hr") %>% et(amt=10000, ii=24, until=set_units(2, "weeks"), cmt="depot")

et <- etRep(qd, times=4, wait=set_units(1,"weeks")) %>%
     add.sampling(set_units(seq(0, 12.5,by=0.005),weeks))
```

`eventTable` *Create an event table object*

Description

Initializes an object of class 'EventTable' with methods for adding and querying dosing and observation records

Usage

```
eventTable(amount.units = NA, time.units = NA)
```

Arguments

`amount.units`	string denoting the amount dosing units, e.g., "mg", "ug". Default to `NA` to denote unspecified units. It could also be a solved rxode2 object. In that case, eventTable(obj) returns the eventTable that was used to solve the rxode2 object.
`time.units`	string denoting the time units, e.g., "hours", "days". Default to `"hours"`.
	An `eventTable` is an object that consists of a data.frame storing ordered time-stamped events of an (unspecified) PK/PD dynamic system, units (strings) for dosing and time records, plus a list of functions to add and extract event records.
	Currently, events can be of two types: dosing events that represent inputs to the system and sampling time events that represent observations of the system with 'amount.units' and 'time.units', respectively.

Value

A modified data.frame with the following accessible functions:

- `get.EventTable()` returns the current event table
- `add.dosing()` adds dosing records to the event table.
- `get.dosing()` returns a data.frame of dosing records.
- `clear.dosing()` clears or deletes all dosing from event table
- 'add.sampling()` adds sampling time observation records to the event table.
- `get.sampling()`returns a data.frame of sampled observation records.
- `clear.sampling()` removes all sampling from event table.
- `get.obs.rec()` returns a logical vector indicating whether each event record represents an observation or not.
- `get.nobs()` returns the number of observation (not dosing) records.
- `get.units()` returns a two-element character vector with the dosing and time units, respectively
- `copy()` makes a copy of the current event table. To create a copy of an event table object use `qd2 <- qd$copy()`
- `expand()` Expands the event table for multi-subject solving. This is done by `qd$expand(400)` for a 400 subject data expansion

Author(s)

Matthew Fidler, Melissa Hallow and Wenping Wang

See Also

et()

Examples

```
# create dosing and observation (sampling) events
# QD 50mg dosing, 5 days followed by 25mg 5 days
#
qd <- eventTable(amount.units = "mg", time.units = "days")
#
qd$add.dosing(dose = 50, nbr.doses = 5, dosing.interval = 1, do.sampling = FALSE)
#
# sample the system's drug amounts hourly the first day, then every 12 hours
# for the next 4 days
qd$add.sampling(seq(from = 0, to = 1, by = 1 / 24))
qd$add.sampling(seq(from = 1, to = 5, by = 12 / 24))
#
# print(qd$get.dosing())     # table of dosing records
print(qd$get.nobs()) # number of observation (not dosing) records
#
# BID dosing, 5 days
bid <- eventTable("mg", "days") # only dosing
bid$add.dosing(
  dose = 10000, nbr.doses = 2 * 5,
  dosing.interval = 12, do.sampling = FALSE
)
#
# Use the copy() method to create a copy (clone) of an existing
# event table (simple assignments just create a new reference to
# the same event table object (closure)).
#
bid.ext <- bid$copy() # three-day extension for a 2nd cohort
bid.ext$add.dosing(
  dose = 5000, nbr.doses = 2 * 3,
  start.time = 120, dosing.interval = 12, do.sampling = FALSE
)

# You can also use the Piping operator to create a table

qd2 <- eventTable(amount.units = "mg", time.units = "days") %>%
  add.dosing(dose = 50, nbr.doses = 5, dosing.interval = 1, do.sampling = FALSE) %>%
  add.sampling(seq(from = 0, to = 1, by = 1 / 24)) %>%
  add.sampling(seq(from = 1, to = 5, by = 12 / 24))
# print(qd2$get.dosing())     # table of dosing records
print(qd2$get.nobs()) # number of observation (not dosing) records

# Note that piping with %>% will update the original table.
```

```
qd3 <- qd2 %>% add.sampling(seq(from = 5, to = 10, by = 6 / 24))
print(qd2$get.nobs())
print(qd3$get.nobs())
```

`is.rxStackData` *Return if the object can be stacked*

Description

Return if the object can be stacked

Usage

```
is.rxStackData(object)
```

Arguments

`object`	object to test if it can be stacked

Value

boolean to tell if an object can be stacked using rxode2

Author(s)

Matthew L. Fidler

Examples

```
is.rxStackData(NULL)
```

`rxCbindStudyIndividual` *Bind the study parameters and individual parameters*

Description

Bind the study parameters and individual parameters

Usage

```
rxCbindStudyIndividual(studyParameters, individualParameters)
```

Arguments

`studyParameters`
: These are the study parameters, often can be generated by sampling from a population. This can be either a matrix or a data frame

`individualParameters`
: A data frame of individual parameters

Value

Data frame that can be used in rxode2 simulations

Author(s)

Matthew Fidler

Examples

```
# Function for coverting coefficient of covariance into a variance
lognCv <- function(x){log((x/100)^2+1)}

set.seed(32)

nSub  <-  100
nStud  <-  10

#define theta
theta <- c(lka=log(0.5), # log ka
          lCl=log(5), # log Cl
          lV=log(300) # log V
          )

#define theta Matrix
thetaMat <- lotri(lCl ~ lognCv(5),
                  lV  ~ lognCv(5),
                  lka ~ lognCv(5))


nev <- nSub*nStud

ev1 <- data.frame(COV1=rnorm(nev,50,30),COV2=rnorm(nev,75,10),
                  COV3=sample(c(1.0,2.0),nev,replace=TRUE))


tmat <-rxRmvn(nStud, theta[dimnames(thetaMat)[[1]]], thetaMat)

rxCbindStudyIndividual(tmat, ev1)
```

rxEtDispatchSolve *Dispatch solve to 'rxode2' solve*

Description

Dispatch solve to 'rxode2' solve

Usage

```
rxEtDispatchSolve(x, ...)

## Default S3 method:
rxEtDispatchSolve(x, ...)
```

Arguments

x	rxode2 solve dispatch object
...	other arguments

Value

if 'rxode2' is loaded, a solved object, otherwise an error

Author(s)

Matthew L. Fidler

rxEvid *EVID formatting for tibble and other places.*

Description

This is to make an EVID more readable by non pharmacometricians. It displays what each means and allows it to be displayed in a tibble.

Usage

```
rxEvid(x)

as.rxEvid(x)

## S3 method for class 'rxEvid'
c(x, ...)

## S3 method for class 'rxEvid'
```

```
x[...]

## S3 method for class 'rxEvid'
as.character(x, ...)

## S3 method for class 'rxEvid'
x[[...]]

## S3 method for class 'rxRateDur'
c(x, ...)

## S3 method for class 'rxEvid'
format(x, ...)

## S3 method for class 'rxRateDur'
format(x, ...)

## S3 method for class 'rxEvid'
print(x, ...)
```

Arguments

x	Item to be converted to a rxode2 EVID specification.
...	Other parameters

Value

rxEvid specification

Examples

```
rxEvid(1:7)
```

`rxRateDur` *Creates a rxRateDur object*

Description

This is primarily to display information about rate

Usage

```
rxRateDur(x)

## S3 method for class 'rxRateDur'
x[...]
```

```
as.rxRateDur(x)

## S3 method for class 'rxRateDur'
as.character(x, ...)

## S3 method for class 'rxRateDur'
x[[...]]
```

Arguments

x	rxRateDur data
...	Other parameters

Value

rxRateDur object

`rxStack` *Stack a solved object for things like default ggplot2 plot*

Description

Stack a solved object for things like default ggplot2 plot

Usage

```
rxStack(data, vars = NULL, doSim = TRUE)
```

Arguments

data	is a rxode2 object to be stacked.
vars	Variables to include in stacked data; By default this is all the variables when vars is NULL. When vars is `sim` and comes from a `rxode2` ui simulation with multiple endpoints (ie it has a `CMT` in the simulation), it will rework the data as if it was stacked based the value based on the compartments in the multiple endpoint model. When the vars is `sim.endpoint1` it will subset the stack to endpoint1, you can also have ‘c("sim.endpoint1", "sim.endpoint2") and the "stack" will subset to endpoint1 and endpoint2. When you specify the `sim` type variables they have to be all prefixed with `sim` otherwise, the stack will not treat them differently.
doSim	boolean that determines if the "sim" variable in a `rxSolve` dataset is actually "stacking" based on the endpoint (`TRUE`) or simply treating `sim` as a variable.

Value

Stacked data with `value` and `trt`, where value is the values and `trt` is the state and `lhs` variables.

Author(s)

Matthew Fidler

`toTrialDuration` *Convert event data to trial duration data A helper function to create a custom event table. The observation time will start from the first event time (baseline) and end at trial duration. The interval is the spacing between each observation.*

Description

Convert event data to trial duration data A helper function to create a custom event table. The observation time will start from the first event time (baseline) and end at trial duration. The interval is the spacing between each observation.

Usage

```
toTrialDuration(ev, trialEnd, interval, writeDir = NULL)
```

Arguments

`ev`	event data
`trialEnd`	extend trial duration. Must be same time unit as event data
`interval`	observation interval. Must be same time unit as event data
`writeDir`	if not NULL, write the output to a csv file

Author(s)

Omar Elashkar

Examples

```
# Create event table with unique time for each ID
ev = et(data.frame(id = rep(1:10, 3),  time = runif(min = 10, max = 20, n = 30)))

# select the duration and spacing interval (assuming time is in years)
toTrialDuration(ev, trialEnd = 1.5, interval = 0.2)
```

Index

∗ **Nonlinear regression**
    eventTable, 20
∗ **Pharmacodynamics (PD)**
    eventTable, 20
∗ **Pharmacokinetics (PK)**
    eventTable, 20
∗ **data**
    eventTable, 20
∗ **models**
    eventTable, 20
∗ **ordinary differential equations**
    eventTable, 20
.collectWarnings, 2
[.rxEvid *(*rxEvid*)*, 24
[.rxRateDur *(*rxRateDur*)*, 25
[[.rxEvid *(*rxEvid*)*, 24
[[.rxRateDur *(*rxRateDur*)*, 25

add.dosing, 3, *4*, *6*, *10*, *14*, *16*, *18*
add.dosing(), *20*
add.sampling, *4*, 5, *6*, *10*, *14*, *16*, *18*
add.sampling(), *20*
as.character.rxEvid *(*rxEvid*)*, 24
as.character.rxRateDur *(*rxRateDur*)*, 25
as.et, 7
as.rxEvid *(*rxEvid*)*, 24
as.rxRateDur *(*rxRateDur*)*, 25

c.rxEvid *(*rxEvid*)*, 24
c.rxRateDur *(*rxEvid*)*, 24

environment, *8*
et, *4*, *6*, 7, *10*, *14*, *16*, *18*
et(), *3*, *21*
etExpand, 12
etRbind, *4*, *6*, *10*, 12, *14*, *16*, *18*
etRep, *4*, *6*, *10*, *14*, 15, *16*, *18*
etSeq, 17
eventTable, *4*, *6*, *10*, *14*, *16*, *18*, 20

format.rxEvid *(*rxEvid*)*, 24
format.rxRateDur *(*rxEvid*)*, 24

is.rxStackData, 22

print.rxEvid *(*rxEvid*)*, 24

rbind.rxEt *(*etRbind*)*, 12
rep.rxEt *(*etRep*)*, 15
rxCbindStudyIndividual, 22
rxEtDispatchSolve, 24
rxEvid, 24
rxRateDur, 25
rxStack, 26

seq.rxEt *(*etSeq*)*, 17
sys.call, *8*

toTrialDuration, 27