

Package ‘SEQTaRget’

August 28, 2026

Type Package

Title Sequential Trial Emulation

Version 1.4.4

Description Implementation of sequential trial emulation for the analysis of observational databases. The 'SEQTaRget' software accommodates time-varying treatments and confounders, as well as binary and failure time outcomes. 'SEQTaRget' allows to compare both static and dynamic strategies, can be used to estimate observational analogs of intention-to-treat and per-protocol effects, and can adjust for potential selection bias induced by losses-to-follow-up. (Paper to come).

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests rmarkdown, testthat (>= 3.0.0)

Imports data.table, parglm, doFuture, doRNG, fastglm, future, future.apply, ggplot2, knitr, methods, stringr, survival, parallelly, utils

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.1.0

Config/testthat/edition 3

Depends R (>= 4.1)

URL <https://causalinference.github.io/SEQTaRget/>,
<https://github.com/CausalInference/SEQTaRget>

VignetteBuilder knitr

NeedsCompilation no

Author Ryan O'Dea [aut, cre] (ORCID: <<https://orcid.org/0009-0000-0103-9546>>),
Tom Palmer [aut] (ORCID: <<https://orcid.org/0000-0003-4655-4511>>, ROR:
<<https://ror.org/0524sp257>>),
Paul Madley-Dowd [aut] (ORCID: <<https://orcid.org/0000-0003-2932-9486>>),
Alejandro Szmulewicz [aut] (ORCID:

<<https://orcid.org/0000-0002-2664-802X>>),
Miguel A. Hernán [aut] (ORCID: <<https://orcid.org/0000-0003-1619-8456>>),
The President and Fellows of Harvard College [cph] (ROR:
<<https://ror.org/03vek6s52>>)

Maintainer Ryan O'Dea <ryan.odea@psi.ch>

Repository CRAN

Date/Publication 2026-08-28 07:50:02 UTC

Contents

compevent	2
covariates	3
denominator	3
diagnostics	4
end_of_fup	5
hazard_ratio	5
km_curve	6
km_data	7
numerator	7
outcome	8
risk_comparison	8
risk_data	9
SEQdata	9
SEQdata.LTFU	10
SEQdata.multitreatment	10
SEQopts	11
SEQoutput-class	17
SEquential	18
SEQ_data	20
show,SEQoutput-method	21

Index	22
--------------	-----------

compevent	<i>Function to return competing event models from a SEquential object</i>
-----------	---

Description

Function to return competing event models from a SEquential object

Usage

compevent(object)

Arguments

object SEQoutput object

Value

A fastglm object, or a named list of fastglm objects when subgroups are specified

covariates

Retrieves Outcome, Numerator, and Denominator Covariates

Description

Retrieves Outcome, Numerator, and Denominator Covariates

Usage

```
covariates(object)
```

Arguments

object object of class SEQoutput

Value

List of SEQuential covariates

denominator

Retrieves Denominator Models from SEQuential object

Description

Retrieves Denominator Models from SEQuential object

Usage

```
denominator(object)
```

Arguments

object object of class SEQoutput

Value

List of both denominator models

diagnostics

Function to return diagnostic tables from a SEquential object

Description

Function to return diagnostic tables from a SEquential object

Usage

```
diagnostics(object)
```

Arguments

object SEQoutput object

Value

A named list of diagnostic tables, each broken down by baseline treatment arm. The "unique" and "non-unique" variants count different things:

- `outcome.unique / outcome.nonunique`: distinct subjects who had the outcome vs. the total number of outcome events. These coincide for a one-time (terminal) outcome, since each subject contributes at most one event row. Both are NA for a continuous end-of-follow-up outcome, which has no events to count.
- `eof.unique / eof.nonunique`: present only when `end_of_fup = TRUE`, accounting for every trial-period at the end-of-follow-up time across four mutually exclusive categories - measured At k, measured In window, Excluded (outside window) and Excluded (no measurement) - against the Eligible total. The non-unique (trial-period) counts partition Eligible, and At k plus In window equals the trial-periods contributing to the estimate. The unique (subject) counts need not sum to Eligible, since one subject can fall into different categories for different trials. For a continuous outcome `eof.summary` additionally reports the N, mean and SD of the raw analysed measurements per arm, standing in for the suppressed outcome count tables.
- `followup.unique / followup.nonunique`: distinct subjects contributing follow-up vs. the total number of person-time intervals (expanded rows). The non-unique count is much larger because each subject contributes one row per follow-up period; it is the denominator that, with `outcome.nonunique`, gives the per-arm event rate.

end_of_fup	<i>Extract the end-of-follow-up outcome estimates</i>
------------	---

Description

Available when `SEquential()` was run with `end_of_fup = TRUE`. The estimate in each arm is the weighted average of the outcome measured at `end_of_fup.time`, weighted by the period-trial-specific weight at the time the measurement was taken.

Usage

```
end_of_fup(object)
```

Arguments

object	SEQoutput object
--------	------------------

Value

A named list, one element per subgroup, each a list of two data.tables:

- **estimates:** the weighted proportion (binary) or mean (continuous) in each baseline treatment arm, with its bootstrap confidence interval, the trial-periods eligible in each arm, partitioned into those analysed, those censored for want of a measurement in the window - measured at some point but not within $[k - \text{window}, k + \text{window}]$ - and those never measured at all, with the censoring also as a percentage of the eligible total. Subjects counts the distinct contributing subjects.
- **comparison:** the pairwise between-arm difference - in proportions for a binary outcome, in means for a continuous one - with its bootstrap standard error and confidence interval, paired by iteration so the interval accounts for the correlation between arms. For a binary outcome the ratio of proportions is also given, with a confidence interval computed on the log scale and a $\log(\text{Ratio})$ SE for inverse-variance pooling. A ratio is not reported for a continuous outcome, where the outcome need not be bounded away from zero.

hazard_ratio	<i>Function to return hazard ratios from a SEquential object</i>
--------------	--

Description

Function to return hazard ratios from a SEquential object

Usage

```
hazard_ratio(object)
```

Arguments

object SEQoutput object

Value

A named vector of hazard ratios, or a named list of vectors when subgroups are specified

km_curve	<i>Function to print Kaplan-Meier curves</i>
----------	--

Description

Function to print Kaplan-Meier curves

Usage

```
km_curve(
  object,
  plot.type = "survival",
  plot.title,
  plot.subtitle,
  plot.labels,
  plot.colors
)
```

Arguments

object SEQoutput object to plot

plot.type character: type of plot to print; one of: "survival" (default), "risk", "inc"

plot.title character: defines the title of the plot

plot.subtitle character: plot subtitle

plot.labels length 2 character: plot labels

plot.colors length 2 character: plot colors

Value

ggplot object of plot plot.type

km_data	<i>Function to return survival data from a SEquential object</i>
---------	--

Description

Function to return survival data from a SEquential object

Usage

```
km_data(object)
```

Arguments

object	SEQoutput object
--------	------------------

Value

A data frame of survival values, or a named list of data frames when subgroups are specified

numerator	<i>Retrieves Numerator Models from SEquential object</i>
-----------	--

Description

Retrieves Numerator Models from SEquential object

Usage

```
numerator(object)
```

Arguments

object	object of class SEQoutput
--------	---------------------------

Value

List of both numerator models

outcome

Retrieves Outcome Models from SEquential object

Description

Retrieves Outcome Models from SEquential object

Usage

```
outcome(object)
```

Arguments

object object of class SEQoutput

Value

List of all outcome models

risk_comparison

Function to return risk information from a SEquential object

Description

Function to return risk information from a SEquential object

Usage

```
risk_comparison(object)
```

Arguments

object SEQoutput object

Value

A data frame of risk comparison information at the reported follow-up time(s): risk ratios and risk differences, and when bootstrapped their confidence intervals and standard errors. The bootstrap standard errors are reported regardless of bootstrap.CI_method: RD SE is the standard error of the risk difference (natural scale) and log(RR) SE is the standard error of the log risk ratio. For an inverse-variance-weighted meta-analysis across samples, pool Risk Difference with RD SE, and log(Risk Ratio) with log(RR) SE (then exponentiate the pooled ratio).

risk_data	<i>Function to return risk information from a SEquential object</i>
-----------	---

Description

Function to return risk information from a SEquential object

Usage

```
risk_data(object)
```

Arguments

object SEQoutput object

Value

A data table of risk information at the end of followup

SEQdata	<i>Simulated observational example data for SEquential</i>
---------	--

Description

Simulated observational example data for [SEquential\(\)](#)

Usage

```
SEQdata
```

Format

A data frame with 12,180 rows and 11 columns:

ID Integer: Unique ID emulating individual patients

time Integer: Time of observation, always begins at 0, max time of 59. Should be continuous

eligible Binary: eligibility criteria for timepoints

outcome Binary: If an outcome is observed at this time point

tx_init Binary: If treatment is observed at this time point

sex Binary: Sex of the emulated patient

N Numeric: Normal random variable from N(10,5)

L Numeric: 4% continuously increase from U(0, 1)

P Numeric: 2% continuously decrease from U(9, 10)

excusedOne Binary: Once one, always one variable emulating an excuse for treatment switch

excusedZero Binary: Once one, always one variable emulating an excuse for treatment switch

 SEQdata.LTFU

Simulated lost-to-followup example data for [SEquential\(\)](#)

Description

Simulated lost-to-followup example data for [SEquential\(\)](#)

Usage

SEQdata.LTFU

Format

A dataframe with 54,687 rows and 13 columns:

ID Integer: Unique ID emulating individual patients

time Integer: Time of observation, always begins at 0, max time of 59; however, if lost-to-followup, time is truncated at a random point

eligible Binary: eligibility criteria for timepoints

outcome Binary: If an outcome is observed at this time point

tx_init Binary: If treatment is observed at this time point

sex Binary: Sex of the emulated patient

N Numeric: Normal random variable from $N(10,5)$

L Numeric: 4% continuously increase from $U(0, 1)$

P Numeric: 2% continuously decrease from $U(9, 10)$

excusedOne Binary: Once one, always one variable emulating an excuse for treatment switch

excusedZero Binary: Once one, always one variable emulating an excuse for treatment switch

LTFU Binary: Flag for losing a simulated ID to followup, if 1 there are no more records of the ID afterwards

eligible_cense Binary: emulates columns which are eligible to entering into censoring models (e.g. if you want to limit columns for the LTFU model)

 SEQdata.multitreatment

Simulated multitreatment example data for [SEquential\(\)](#) multinomial models

Description

Simulated multitreatment example data for [SEquential\(\)](#) multinomial models

Usage

```
SEQdata.multitreatment
```

Format

A dataframe with 5,976 rows and 11 columns:

ID Integer: Unique ID emulating individual patients

time Integer: Time of observation, always begins at 0, max time of 59; however, if lost-to-followup, time is truncated at a random point

eligible Binary: eligibility criteria for timepoints

outcome Binary: If an outcome is observed at this time point

tx_init Integer: Which treatment is observed at this time point

sex Binary: Sex of the emulated patient

N Numeric: Normal random variable from $N(10,5)$

L Numeric: 4% continuously increase from $U(0, 1)$

P Numeric: 2% continuously decrease from $U(9, 10)$

excusedOne Binary: Once one, always one variable emulating an excuse for treatment switch

excusedZero Binary: Once one, always one variable emulating an excuse for treatment switch

 SEQopts

Parameter Builder for SEquential Model and Estimates

Description

Parameter Builder for SEquential Model and Estimates

Usage

```
SEQopts(
  bootstrap = FALSE,
  bootstrap.nboot = 100,
  bootstrap.sample = 0.8,
  bootstrap.CI = 0.95,
  bootstrap.CI_method = "se",
  cense = NA,
  cense.denominator = NA,
  cense.eligible = NA,
  cense.numerator = NA,
  compevent = NA,
  covariates = NA,
  data.return = FALSE,
  denominator = NA,
  deviation = FALSE,
```

```
deviation.col = NA,  
deviation.conditions = c(NA, NA),  
deviation.excused = FALSE,  
deviation.excused_cols = c(NA, NA),  
end_of_fup = FALSE,  
end_of_fup.time = NA,  
end_of_fup.type = "binary",  
end_of_fup.window = 0,  
excused = FALSE,  
excused.cols = c(NA, NA),  
expand.only = FALSE,  
fastglm.method = 2L,  
followup.class = FALSE,  
followup.include = TRUE,  
followup.max = Inf,  
followup.min = 0,  
followup.spline = FALSE,  
followup.spline.df = 4L,  
glm.package = "fastglm",  
hazard = FALSE,  
indicator.baseline = "_bas",  
indicator.squared = "_sq",  
km.curves = FALSE,  
multinomial = FALSE,  
ncores = availableCores(omit = 1L),  
nthreads = getDTthreads(),  
numerator = NA,  
parallel = FALSE,  
parglm.control = NULL,  
plot.colors = c("#F8766D", "#00BFC4", "#555555"),  
plot.labels = NA,  
plot.subtitle = NA,  
plot.title = NA,  
plot.type = "survival",  
risk.times = NA,  
seed = NULL,  
selection.first_trial = FALSE,  
selection.prob = 0.8,  
selection.random = FALSE,  
subgroup = NA,  
survival.max = Inf,  
treat.level = c(0, 1),  
trial.include = TRUE,  
visit = NA,  
visit.denominator = NA,  
visit.numerator = NA,  
weight.eligible_cols = c(),  
weight.lower = 0,
```

```

weight.lag_condition = TRUE,
weight.p99 = FALSE,
weight.preexpansion = TRUE,
weight.upper = Inf,
weighted = FALSE
)

```

Arguments

<code>bootstrap</code>	Logical: defines if SEquential() should run bootstrapping, default is FALSE
<code>bootstrap.nboot</code>	Integer: number of bootstraps, default is 100
<code>bootstrap.sample</code>	Numeric: percentage of data to use when bootstrapping, should be in [0, 1], default is 0.8
<code>bootstrap.CI</code>	Numeric: defines the confidence interval after bootstrapping, default is 0.95 (95% CI)
<code>bootstrap.CI_method</code>	Character: selects which way to calculate bootstraps confidence intervals ("se", "percentile"), default is "se"
<code>cense</code>	String: column name for additional censoring variable, e.g. loss-to-follow-up
<code>cense.denominator</code>	String: censoring denominator covariates to the right hand side of a formula object
<code>cense.eligible</code>	String: column name for indicator column defining which rows to use for censoring model
<code>cense.numerator</code>	String: censoring numerator covariates to the right hand side of a formula object
<code>compevent</code>	String: column name for competing event indicator
<code>covariates</code>	String: covariates to the right hand side of a formula object
<code>data.return</code>	Logical: whether to return the expanded dataframe with weighting information, default is FALSE
<code>denominator</code>	String or character vector: denominator covariates to the right hand side of a formula object. A single string fits the same model in every treatment arm. A vector with one formula per <code>treat.level</code> (in <code>treat.level</code> order) fits a separate denominator model, with its own covariates, in each arm; this is only supported for post-expansion weights (<code>weight.preexpansion = FALSE</code>)
<code>deviation</code>	Logical: create switch based on deviation from column <code>deviation.col</code> , default is FALSE
<code>deviation.col</code>	Character: column name for deviation
<code>deviation.conditions</code>	Character list: RHS evaluations of the same length as <code>treat.levels</code>
<code>deviation.excused</code>	Logical: whether deviations should be excused by <code>deviation.excused_cols</code> , default is FALSE

<code>deviation.excused_cols</code>	Character list: excused columns for deviation switches
<code>end_of_fup</code>	Logical: estimate an end-of-follow-up outcome - one measured at a single follow-up time rather than as a time-to-event - instead of fitting a survival outcome model, default is FALSE. The estimate is the weighted average of the outcome within each baseline treatment arm, weighted by the period-trial-specific weight at the time the outcome is taken. Incompatible with <code>km.curves</code> and <code>hazard</code>
<code>end_of_fup.time</code>	Numeric: the follow-up time <i>k</i> (in follow-up periods since trial enrollment) at which the end-of-follow-up outcome is evaluated. Required when <code>end_of_fup = TRUE</code>
<code>end_of_fup.type</code>	String: type of end-of-follow-up outcome, either 'binary' (the default, giving the weighted proportion in each arm) or 'continuous' (giving the weighted mean)
<code>end_of_fup.window</code>	Numeric: half-width of the window used when a trial-period has no outcome measurement at exactly <code>end_of_fup.time</code> , default is 0 (no window). Those trial-periods fall back to the measurement nearest to <i>k</i> within [<i>k</i> - window, <i>k</i> + window] (ties, i.e. measurements equally far either side of <i>k</i> , are broken toward the later measurement, so that at least <i>k</i> of follow-up has elapsed); any with no measurement anywhere in the window are censored, i.e. excluded from the average
<code>excused</code>	Logical: in the case of censoring, whether there is an excused condition, default is FALSE
<code>excused.cols</code>	List: list of column names for treatment switch excuses - should be the same length, and ordered the same as <code>treat.level</code>
<code>expand.only</code>	Logical: if TRUE, <code>SEQential()</code> returns the expanded <code>data.table</code> immediately after expansion and skips weighting, outcome modelling and survival/risk steps. Useful when you only need the expanded dataset (e.g. to inspect or store separately). Default is FALSE
<code>fastglm.method</code>	Integer: decomposition method for fastglm (0L-column-pivoted QR, 1L-unpivoted QR, 2L-LLT Cholesky, 3L-LDLT Cholesky, 4L-full-pivoted QR, 5L-Bidiagonal Divide and Conquer SVD), default is 2L
<code>followup.class</code>	Logical: treat followup as a class, e.g. expands every time to it's own indicator column, default is FALSE
<code>followup.include</code>	Logical: whether or not to include 'followup' and 'followup_squared' in the outcome model, default is TRUE
<code>followup.max</code>	Numeric: maximum time to expand about, default is Inf (no maximum)
<code>followup.min</code>	Numeric: minimum follow-up time since trial enrollment to include, must be non-negative, default is 0
<code>followup.spline</code>	Logical: treat followup as a natural cubic spline (<code>splines::ns()</code>), default is FALSE

<code>followup.spline.df</code>	Integer: degrees of freedom passed to <code>splines::ns()</code> when <code>followup.spline = TRUE</code> . With <code>df = k</code> , <code>ns()</code> places $k - 1$ interior knots at quantiles of <code>followup</code> . Must be ≥ 1 ; <code>df = 1</code> is equivalent to a linear term and is generally not what you want. Default is 4 (3 interior knots).
<code>glm.package</code>	Character: package to use for fitting GLMs, either "fastglm" (default) or "parglm". When "parglm" is selected the <code>nthreads</code> option controls the number of threads passed to <code>parglm::parglm.fit()</code> . For most realistic SEQTaRget workloads (expanded datasets up to approximately a few million rows) "fastglm" is faster; "parglm" may help only on substantially larger datasets where the parallel chunking outweighs its setup overhead. Note that when <code>bootstrap = TRUE</code> only the main fit uses "parglm": the bootstrap refits always use "fastglm", warm-started from the main fit's coefficients, which is faster per resample than parglm's per-fit thread setup.
<code>hazard</code>	Logical: hazard error calculation instead of survival estimation, default is FALSE
<code>indicator.baseline</code>	String: identifier for baseline variables in covariates, numerator, denominator - intended as an override
<code>indicator.squared</code>	String: identifier for squared variables in covariates, numerator, denominator - intended as an override
<code>km.curves</code>	Logical: Kaplan-Meier survival curve creation and data return, default is FALSE
<code>multinomial</code>	Logical: whether to expect multilevel treatment values, default is FALSE
<code>ncores</code>	Integer: number of cores to use in parallel processing, default is one less than system max, see parallelly::availableCores()
<code>nthreads</code>	Integer: number of threads to use for data.table processing, default is data.table::getDTthreads()
<code>numerator</code>	String or character vector: numerator covariates to the right hand side of a formula object. A single string fits the same model in every treatment arm. A vector with one formula per <code>treat.level</code> (in <code>treat.level</code> order) fits a separate numerator model, with its own covariates, in each arm; this is only supported for post-expansion weights (<code>weight.preexpansion = FALSE</code>)
<code>parallel</code>	Logical: define if the SEquential process is run in parallel, default is FALSE
<code>parglm.control</code>	A control object from parglm::parglm.control() to pass to <code>parglm::parglm.fit()</code> . Only used when <code>glm.package = "parglm"</code> . Defaults to <code>parglm::parglm.control(method = "FAST")</code> . If you encounter a <code>chol(): decomposition failed</code> error (e.g. with near-singular model matrices on large datasets), pass <code>parglm.control = parglm::parglm.control(method = "LAPACK")</code> to use the more numerically stable QR decomposition instead, or switch to using the fastglm backend.
<code>plot.colors</code>	Character: Colors for output plot if <code>km.curves = TRUE</code> , defaulted to ggplot2 defaults
<code>plot.labels</code>	Character: Color labels for output plot if <code>km.curves = TRUE</code> in order e.g. <code>c("risk.0", "risk.1")</code>
<code>plot.subtitle</code>	Character: Subtitle for output plot if <code>km.curves = TRUE</code>
<code>plot.title</code>	Character: Title for output plot if <code>km.curves = TRUE</code>

<code>plot.type</code>	Character: Type of plot to create if <code>km.curves = TRUE</code> , available options are 'survival' (the default), 'risk', and 'inc' (in the case of censoring)
<code>risk.times</code>	Numeric vector: follow-up times (in the data's follow-up units) at which to report risk difference and risk ratio when <code>km.curves = TRUE</code> . Each requested time is snapped to the latest available follow-up at or before it. The final follow-up time is always included; because the follow-up grid is zero-indexed (followup runs <code>0:survival.max</code>), this final time is <code>survival.max + 1</code> , so e.g. <code>survival.max = 120</code> reports a row at 121. Default NA reports only the final follow-up time.
<code>seed</code>	Integer: starting seed; the default NULL draws a single random integer when <code>SEQopts()</code> is called, so set this explicitly for reproducible bootstrap results
<code>selection.first_trial</code>	Logical: selects only the first eligible trial in the expanded dataset, default FALSE
<code>selection.prob</code>	Numeric: percent of total IDs to select for <code>selection.random</code> , should be bound <code>[0, 1]</code> , default is 0.8
<code>selection.random</code>	Logical: randomly selects IDs with replacement to run analysis, default FALSE
<code>subgroup</code>	Character: Column name to stratify outcome models on
<code>survival.max</code>	Numeric: maximum time for survival curves, default is Inf (no maximum)
<code>treat.level</code>	List: treatment levels to compare, default is <code>c(0, 1)</code>
<code>trial.include</code>	Logical: whether or not to include 'trial' and 'trial_squared' in the outcome model, default is TRUE
<code>visit</code>	String: column name for visit indicator variable, e.g. "visit"
<code>visit.denominator</code>	String: visit denominator covariates to the right hand side of a formula object
<code>visit.numerator</code>	String: visit numerator covariates to the right hand side of a formula object
<code>weight.eligible_cols</code>	List: list of column names for indicator columns defining which weights are eligible for weight models - in order of <code>treat.level</code>
<code>weight.lower</code>	Numeric: IPCW weights truncated at this lower bound, must be non-negative, default is 0. Truncation is applied only to the weights used to fit the outcome model; the weights reported in <code>weight.statistics</code> and in the returned data (when <code>data.return = TRUE</code>) are the untruncated values.
<code>weight.lag_condition</code>	Logical: whether weights should be conditioned on treatment lag value, default TRUE
<code>weight.p99</code>	Logical: forces weight truncation at 1st and 99th percentile weights, will override provided <code>weight.upper</code> and <code>weight.lower</code> . The percentiles are taken from the untruncated weight distribution (as reported in <code>weight.statistics</code>), and as with <code>weight.lower/weight.upper</code> the truncation affects only the weights used to fit the outcome model.
<code>weight.preexpansion</code>	Logical: whether weighting should be done on pre-expanded data, default TRUE

<code>weight.upper</code>	Numeric: weights truncated at upper end at this weight, default is Inf. As with <code>weight.lower</code> , truncation affects only the weights used to fit the outcome model, not those reported in <code>weight.statistics</code> or the returned data.
<code>weighted</code>	Logical: whether or not to perform weighted analysis, default is FALSE

Value

An object of class 'SEQopts'

SEQoutput-class	<i>An S4 class used to hold the outputs for the SEQuential process</i>
-----------------	--

Description

An S4 class used to hold the outputs for the SEQuential process

Slots

`params` SEQparams object
`outcome` outcome covariates
`numerator` numerator covariates
`denominator` denominator covariates
`outcome.model` list of length `bootstrap.nboot` containing outcome coefficients
`hazard` hazard ratio
`survival.curve` ggplot object for the survival curves
`survival.data` data.table of survival data
`risk.difference` risk difference calculated from survival data
`risk.ratio` risk ratio calculated from survival data
`time` time used for the SEQuential process
`eof.data` end-of-follow-up estimates per treatment arm when `end_of_fup = TRUE`, empty otherwise. See [end_of_fup\(\)](#)
`eof.comparison` pairwise between-arm contrasts of the end-of-follow-up estimates when `end_of_fup = TRUE`, empty otherwise
`weight.statistics` information from the weighting process, containing weight coefficients and weight statistics
`info` list of diagnostic tables (outcome, follow-up, switch, and competing-event counts where applicable), each split by baseline treatment arm. The "unique" tables count distinct subjects; the "non-unique" tables count rows: total outcome events for the outcome tables, and total person-time intervals for the follow-up tables. The outcome tables are NA for a continuous end-of-follow-up outcome, which has no events to count. See [diagnostics\(\)](#).
`ce.model` list of competing event models if `compevent` is specified, NA otherwise

SEquential

*SEquential trial emulation***Description**

SEquential is an all-in-one API to SEquential analysis, returning a SEQoutput object of results. More specific examples can be found on pages at <https://causalinference.github.io/SEQTaRget/>

Usage

```
SEquential(
  data,
  id.col,
  time.col,
  eligible.col,
  treatment.col,
  outcome.col,
  time_varying.cols = list(),
  fixed.cols = list(),
  method,
  options,
  verbose = TRUE
)
```

Arguments

<code>data</code>	data.frame or data.table, will perform expansion according to arguments passed through the options argument
<code>id.col</code>	String: column name of the id column
<code>time.col</code>	String: column name of the time column
<code>eligible.col</code>	String: column name of the eligibility column
<code>treatment.col</code>	String: column name of the treatment column
<code>outcome.col</code>	String: column name of the outcome column
<code>time_varying.cols</code>	List: column names for time varying columns
<code>fixed.cols</code>	List: column names for fixed columns
<code>method</code>	String: method of analysis to perform; should be one of "ITT", "dose-response", or "censoring"
<code>options</code>	List: optional list of parameters from SEQopts()
<code>verbose</code>	Logical: if TRUE, cats progress to console, default is TRUE

Details

Implementation of sequential trial emulation for the analysis of observational databases. The SEquential software accommodates time-varying treatments and confounders, as well as binary and failure time outcomes. SEquential allows to compare both static and dynamic strategies, can be used to estimate observational analogs of intention-to-treat and per-protocol effects, and can adjust for potential selection bias induced by losses-to-follow-up.

Value

An S4 object of class SEQoutput. If `options = SEQopts(expand.only = TRUE)`, returns the expanded `data.table` directly, with analysis steps skipped.

Examples

```
data <- SEQdata

# Intention-to-treat (ITT) effect: subjects are assigned to the treatment
# arm defined by their baseline treatment and followed regardless of any later
# treatment changes, so no weighting is required.
SEQquential(data, id.col = "ID",
             time.col = "time",
             eligible.col = "eligible",
             treatment.col = "tx_init",
             outcome.col = "outcome",
             time_varying.cols = c("N", "L", "P"),
             fixed.cols = "sex",
             method = "ITT",
             options = SEQopts())

# Per-protocol effect via artificial censoring: subjects are censored when they
# deviate from their assigned strategy, and inverse-probability-of-censoring
# weights adjust for the resulting selection bias. The denominator models the
# probability of remaining uncensored given the time-varying confounders, while
# the numerator uses only the baseline covariates to stabilize the weights (so
# the two formulas must differ - identical formulas give weights of 1).
SEQquential(data, id.col = "ID",
             time.col = "time",
             eligible.col = "eligible",
             treatment.col = "tx_init",
             outcome.col = "outcome",
             time_varying.cols = c("N", "L", "P"),
             fixed.cols = "sex",
             method = "censoring",
             options = SEQopts(weighted = TRUE,
                               numerator = "sex",
                               denominator = "N + L + P + sex"))

# End-of-follow-up outcome: rather than a time-to-event, the outcome is read
# once per trial-period at follow-up time 12 and averaged within each baseline
# arm, weighted by the weight at that time. Trial-periods with no measurement
# at exactly 12 fall back to the nearest one within 12 +/- 3; any with none
```

```

# in that window are censored out of the average. Use end_of_fup.type =
# "continuous" for a continuous outcome, which is reported as a mean.
eof <- SEquential(data, id.col = "ID",
  time.col = "time",
  eligible.col = "eligible",
  treatment.col = "tx_init",
  outcome.col = "outcome",
  time_varying.cols = c("N", "L", "P"),
  fixed.cols = "sex",
  method = "ITT",
  options = SEQopts(end_of_fup = TRUE,
    end_of_fup.time = 12,
    end_of_fup.type = "binary",
    end_of_fup.window = 3,
    bootstrap = TRUE,
    bootstrap.nboot = 5))

end_of_fup(eof)

```

SEQ_data

Function to return the internal data from a SEquential object

Description

Function to return the internal data from a SEquential object

Usage

```
SEQ_data(object)
```

Arguments

object SEQoutput object

Value

data.table

show,SEQoutput-method *Show method for S4 object - SEQoutput.*

Description

Show method for S4 object - SEQoutput.

Usage

```
## S4 method for signature 'SEQoutput'  
show(object)
```

Arguments

object A SEQoutput object - usually generated from [SEquential\(\)](#)

Value

No return value, sends information about SEQoutput to the console

Index

* datasets

SEQdata, [9](#)

SEQdata.LTFU, [10](#)

SEQdata.multitreatment, [10](#)

compevent, [2](#)

covariates, [3](#)

data.table::getDTthreads(), [15](#)

denominator, [3](#)

diagnostics, [4](#)

diagnostics(), [17](#)

end_of_fup, [5](#)

end_of_fup(), [17](#)

hazard_ratio, [5](#)

km_curve, [6](#)

km_data, [7](#)

numerator, [7](#)

outcome, [8](#)

parallelly::availableCores(), [15](#)

parglm::parglm.control(), [15](#)

risk_comparison, [8](#)

risk_data, [9](#)

SEQ_data, [20](#)

SEQdata, [9](#)

SEQdata.LTFU, [10](#)

SEQdata.multitreatment, [10](#)

SEQopts, [11](#)

SEQopts(), [18](#)

SEQoutput-class, [17](#)

SEquential, [18](#)

SEquential(), [5](#), [9](#), [10](#), [14](#), [21](#)

show, SEQoutput-method, [21](#)