Package: SEQTaRget (via r-universe)

June 8, 2026

Type Package

Title Sequential Trial Emulation

Version 1.4.2.9002

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

Config/roxygen2/markdown TRUE

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Depends R (>= 4.1)

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

VignetteBuilder knitr

Config/pak/sysreqs libicu-dev

Repository https://causalinference.r-universe.dev

Date/Publication 2026-06-08 13:16:22 UTC

RemoteUrl https://github.com/causalinference/seqtarget

RemoteRef HEAD

RemoteSha ea65b42d44b6bcf146a57d9b0cdfba92e18a9673

RemoteSubdir SEQTaRget

Contents

`compevent` *Function to return competing event models from a SEQuential object*

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

List of diagnostic tables

`hazard_ratio` *Function to return hazard ratios from a SEQuential object*

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` *Function to print Kaplan-Meier curves*

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` *Function to return survival data from a SEQuential object*

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` *Retrieves Numerator Models from SEQuential object*

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 information at end of followup (risk ratios, risk differences and confidence intervals, if bootstrapped)

`risk_data` *Function to return risk information from a SEQuential object*

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

`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

`SEQdata` *Simulated observational example data for SEQuential*

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

---

`SEQestimate` *Estimate the (very rough) time to run SEQuential analysis on current machine*

Description

Estimate the (very rough) time to run SEQuential analysis on current machine

Usage

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

Arguments

`data`	data.frame or data.table, if not already expanded with `SEQexpand()`, will perform expansion according to arguments passed to either `params` or `...`
`id.col`	String: column name of the id column
`time.col`	String: column name of the time column
`eligible.col`	String: column name of the eligibility column
`treatment.col`	String: column name of the treatment column
`outcome.col`	String: column name of the outcome column
`time_varying.cols`	List: column names for time varying columns
`fixed.cols`	List: column names for fixed columns
`method`	String: method of analysis to perform
`options`	List: optional list of parameters from `SEQopts()`
`verbose`	Logical: if `TRUE`, cats progress to console, default is `TRUE`

Value

A list of (very rough) estimates for the time required for SEQuential containing:

- `modelTime` estimated time used when running models
- `expansionTime` estimated time used when expanding data
- `totalTime` sum of model and expansion time

`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),
  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

`bootstrap`	Logical: defines if `SEQuential()` should run bootstrapping, default is `FALSE`
`bootstrap.nboot`	Integer: number of bootstraps, default is `100`
`bootstrap.sample`	Numeric: percentage of data to use when bootstrapping, should be in [0, 1], default is `0.8`
`bootstrap.CI`	Numeric: defines the confidence interval after bootstrapping, default is `0.95` (95% CI)
`bootstrap.CI_method`	Character: selects which way to calculate bootstraps confidence intervals (`"se"`, `"percentile"`), default is `"se"`
`cense`	String: column name for additional censoring variable, e.g. loss-to-follow-up
`cense.denominator`	String: censoring denominator covariates to the right hand side of a formula object
`cense.eligible`	String: column name for indicator column defining which rows to use for censoring model
`cense.numerator`	String: censoring numerator covariates to the right hand side of a formula object
`compevent`	String: column name for competing event indicator
`covariates`	String: covariates to the right hand side of a formula object
`data.return`	Logical: whether to return the expanded dataframe with weighting information, default is `FALSE`
`denominator`	String: denominator covariates to the right hand side of a formula object
`deviation`	Logical: create switch based on deviation from column `deviation.col`, default is `FALSE`
`deviation.col`	Character: column name for deviation
`deviation.conditions`	Character list: RHS evaluations of the same length as `treat.levels`
`deviation.excused`	Logical: whether deviations should be excused by `deviation.excused_cols`, default is `FALSE`
`deviation.excused_cols`	Character list: excused columns for deviation switches
`excused`	Logical: in the case of censoring, whether there is an excused condition, default is `FALSE`

`excused.cols`	List: list of column names for treatment switch excuses - should be the same length, and ordered the same as `treat.level`
`expand.only`	Logical: if `TRUE`, `SEQuential()` returns the expanded `data.table` 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`
`fastglm.method`	Integer: decomposition method for fastglm (`0L`-column-pivoted QR, `1L`-unpivoted QR, `2L`-LLT Cholesky, `3L`-LDLT Cholesky), default is `2L`
`followup.class`	Logical: treat followup as a class, e.g. expands every time to it's own indicator column, default is `FALSE`
`followup.include`	Logical: whether or not to include 'followup' and 'followup_squared' in the outcome model, default is `TRUE`
`followup.max`	Numeric: maximum time to expand about, default is `Inf` (no maximum)
`followup.min`	Numeric: minimum follow-up time since trial enrollment to include, must be non-negative, default is `0`
`followup.spline`	Logical: treat followup as a natural cubic spline (`splines::ns()`), default is `FALSE`
`followup.spline.df`	Integer: degrees of freedom passed to `splines::ns()` when `followup.spline = TRUE`. With `df = k`, `ns()` places `k - 1` interior knots at quantiles of `followup`. Must be `>= 1`; `df = 1` is equivalent to a linear term and is generally not what you want. Default is `4` (3 interior knots).
`glm.package`	Character: package to use for fitting GLMs, either `"fastglm"` (default) or `"parglm"`. When `"parglm"` is selected the `nthreads` option controls the number of threads passed to `parglm::parglm.fit()`. 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.
`hazard`	Logical: hazard error calculation instead of survival estimation, default is `FALSE`
`indicator.baseline`	String: identifier for baseline variables in `covariates`, `numerator`, `denominator` - intended as an override
`indicator.squared`	String: identifier for squared variables in `covariates`, `numerator`, `denominator` - intended as an override
`km.curves`	Logical: Kaplan-Meier survival curve creation and data return, default is `FALSE`
`multinomial`	Logical: whether to expect multilevel treatment values, default is `FALSE`
`ncores`	Integer: number of cores to use in parallel processing, default is one less than system max, see `parallelly::availableCores()`
`nthreads`	Integer: number of threads to use for data.table processing, default is `data.table::getDTthreads()`
`numerator`	String: numerator covariates to the right hand side of a formula object
`parallel`	Logical: define if the SEQuential process is run in parallel, default is `FALSE`

`parglm.control`	A control object from `parglm::parglm.control()` to pass to `parglm::parglm.fit()`. Only used when `glm.package = "parglm"`. Defaults to `parglm::parglm.control(method = "FAST")`. If you encounter a `chol(): decomposition failed` error (e.g. with near-singular model matrices on large datasets), pass `parglm.control = parglm::parglm.control(method = "LAPACK")` to use the more numerically stable QR decomposition instead, or switch to using the fastglm backend.
`plot.colors`	Character: Colors for output plot if `km.curves = TRUE`, defaulted to ggplot2 defaults
`plot.labels`	Character: Color labels for output plot if `km.curves = TRUE` in order e.g. `c("risk.0", "risk.1")`
`plot.subtitle`	Character: Subtitle for output plot if `km.curves = TRUE`
`plot.title`	Character: Title for output plot if `km.curves = TRUE`
`plot.type`	Character: Type of plot to create if `km.curves = TRUE`, available options are `'survival'` (the default), `'risk'`, and `'inc'` (in the case of censoring)
`risk.times`	Numeric vector: follow-up times (in the data's follow-up units) at which to report risk difference and risk ratio when `km.curves = TRUE`. Each requested time is snapped to the latest available follow-up at or before it. The final follow-up time is always included. Default `NA` reports only the final follow-up time.
`seed`	Integer: starting seed
`selection.first_trial`	Logical: selects only the first eligible trial in the expanded dataset, default `FALSE`
`selection.prob`	Numeric: percent of total IDs to select for `selection.random`, should be bound [0, 1], default is `0.8`
`selection.random`	Logical: randomly selects IDs with replacement to run analysis, default `FALSE`
`subgroup`	Character: Column name to stratify outcome models on
`survival.max`	Numeric: maximum time for survival curves, default is `Inf` (no maximum)
`treat.level`	List: treatment levels to compare, default is `c(0, 1)`
`trial.include`	Logical: whether or not to include 'trial' and 'trial_squared' in the outcome model, default is `TRUE`
`visit`	String: column name for visit indicator variable, e.g. `"visit"`
`visit.denominator`	String: visit denominator covariates to the right hand side of a formula object
`visit.numerator`	String: visit numerator covariates to the right hand side of a formula object
`weight.eligible_cols`	List: list of column names for indicator columns defining which weights are eligible for weight models - in order of `treat.level`
`weight.lower`	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 `weight.statistics` and in the returned data (when `data.return = TRUE`) are the untruncated values.

`weight.lag_condition`	Logical: whether weights should be conditioned on treatment lag value, default `TRUE`
`weight.p99`	Logical: forces weight truncation at 1st and 99th percentile weights, will override provided `weight.upper` and `weight.lower`. The percentiles are taken from the untruncated weight distribution (as reported in `weight.statistics`), and as with `weight.lower`/`weight.upper` the truncation affects only the weights used to fit the outcome model.
`weight.preexpansion`	Logical: whether weighting should be done on pre-expanded data, default `TRUE`
`weight.upper`	Numeric: weights truncated at upper end at this weight, default is `Inf`. As with `weight.lower`, truncation affects only the weights used to fit the outcome model, not those reported in `weight.statistics` or the returned data.
`weighted`	Logical: whether or not to perform weighted analysis, default is `FALSE`

Value

An object of class 'SEQopts'

`SEQoutput-class` *An S4 class used to hold the outputs for the SEQuential process*

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

`weight.statistics` information from the weighting process, containing weight coefficients and weight statistics

`info` list of outcome and switch information (if applicable)

`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

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

# 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).
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 = "censoring",
           options = SEQopts(weighted = TRUE,
                             numerator = "sex",
                             denominator = "N + L + P + sex"))
```

`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