Package ‘SEQTaRget’

September 17, 2025

Type Package

Title Sequential Trial Emulation

Version 1.0.0

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

RoxygenNote 7.3.2

Suggests rmarkdown, testthat (>= 3.0.0)

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

Config/testthat/edition 3

Depends R (>= 4.1)

URL `https://causalinference.github.io/SEQTaRget/`

VignetteBuilder knitr

NeedsCompilation no

Author Ryan O'Dea [aut, cre] (ORCID: <`https://orcid.org/0009-0000-0103-9546`>), Alejandro Szmulewicz [aut] (ORCID: <`https://orcid.org/0000-0002-2664-802X`>), Tom Palmer [aut] (ORCID: <`https://orcid.org/0000-0003-4655-4511`>, ROR: <`https://ror.org/0524sp257`>), Miguel Hernan [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 <`ryanodea@hsph.harvard.edu`>

Repository CRAN

Date/Publication 2025-09-17 07:50:10 UTC

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

list of fastglm objects

`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 numerator 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

list of hazard ratios

`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
`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

list of dataframes of survival values

`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 every followup

`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 4,139 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 zero, always zero 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

---

`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 zero, always zero 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 preform 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 preform
`options`	List: optional list of parameters from `SEQopts`
`verbose`	Logical: if TRUE, cats progress to console

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,
  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),
    fastglm.method = 2L,
    followup.class = FALSE,
    followup.include = TRUE,
    followup.max = Inf,
    followup.min = -Inf,
    followup.spline = FALSE,
    hazard = FALSE,
    indicator.baseline = "_bas",
    indicator.squared = "_sq",
    km.curves = FALSE,
    multinomial = FALSE,
    ncores = parallelly::availableCores() - 1,
    nthreads = data.table::getDTthreads(),
    numerator = NA,
    parallel = FALSE,
    plot.colors = c("#F8766D", "#00BFC4", "#555555"),
    plot.labels = NA,
    plot.subtitle = NA,
    plot.title = NA,
    plot.type = "survival",
    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,
    weight.eligible_cols = c(),
    weight.lower = -Inf,
    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
`bootstrap.sample`	Numeric: percentage of data to use when bootstrapping, should in [0, 1], default is 0.8
`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
`denominator`	String: denominator covariates to the right hand side of a to formula object
`deviation`	Logical: create switch based on deviation from column `deviation.col`
`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`
`deviation.excused_cols`	Character list: excused columns for deviation switches
`excused`	Logical: in the case of censoring, whether there is an excused condition
`excused.cols`	List: list of column names for treatment switch excuses - should be the same length, and ordered the same as `treat.level`
`fastglm.method`	Integer: decomposition method for fastglm (1-QR, 2-Cholesky, 3-LDLT, 4-QR.FPIV)
`followup.class`	Logical: treat followup as a class, e.g. expands every time to it's own indicator column
`followup.include`	Logical: whether or not to include 'followup' and 'followup_squared' in the outcome model
`followup.max`	Numeric: maximum time to expand about, default is Inf (no maximum)
`followup.min`	Numeric: minimum time to expand aboud, default is -Inf (no minimum)
`followup.spline`	Logical: treat followup as a cubic spline
`hazard`	Logical: hazard error calculation instead of survival estimation
`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
`multinomial`	Logical: whether to expect multilevel treatment values
`ncores`	Integer: number of cores to use in parallel processing, default is one less than system max
`nthreads`	Integer: number of threads to use for data.table processing
`numerator`	String: numerator covariates to the right hand side of a to formula object
`parallel`	Logical: define if the SEQuential process is run in parallel, default is FALSE
`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', 'risk', and 'inc' (in the case of censoring)
`seed`	Integer: starting seed
`selection.first_trial`	Logical: selects only the first eligible trial in the expanded dataset
`selection.prob`	Numeric: percent of total IDs to select for `selection.random`, should be bound [0, 1]
`selection.random`	Logical: randomly selects IDs with replacement to run analysis
`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
`trial.include`	Logical: whether or not to include 'trial' and 'trial_squared' in the outcome model
`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: weights truncated at lower end at this weight
`weight.lag_condition`	Logical: whether weights should be conditioned on treatment lag value
`weight.p99`	Logical: forces weight truncation at 1st and 99th percentile weights, will override provided `weight.upper` and `weight.lower`
`weight.preexpansion`	Logical: whether weighting should be done on pre-expanded data
`weight.upper`	Numeric: weights truncated at upper end at this weight
`weighted`	Logical: whether or not to preform 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 in minutes 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 preform expansion according to arguments passed to `options`
`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 preform
`options`	List: optional list of parameters from `SEQopts`
`verbose`	Logical: if TRUE, cats progress to console

Details

Implemention 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. SEQ 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

Examples

```
data <- SEQdata
model <- 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())
```

`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