Package ‘CohortSurvival’

July 3, 2026

Title Estimate Survival from Common Data Model Cohorts

Version 1.1.2

Description Estimate survival using data mapped to the Observational Medical Outcomes Partnership common data model. Survival can be estimated based on user-defined study cohorts.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 8.0.0

Imports broom, cli, clock, dplyr, glue, omopgenerics (>= 1.1.0), PatientProfiles (>= 1.3.1), purrr, rlang, survival (>= 3.7.0), stats, stringr, tibble, tidyr

Suggests testthat (>= 3.0.0), CodelistGenerator, DBI, roxygen2, knitr, tictoc, rmarkdown, ggplot2, patchwork, cmprsk, duckdb, gt, flextable, scales, visOmopResults (>= 1.5.0), extrafont, CDMConnector (>= 2.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

URL `https://darwin-eu.github.io/CohortSurvival/`

BugReports `https://github.com/darwin-eu/CohortSurvival/issues`

NeedsCompilation no

Author Kim López-Güell [aut, cre] (ORCID: <`https://orcid.org/0000-0002-8462-8668`>), Edward Burn [aut] (ORCID: <`https://orcid.org/0000-0002-9286-1128`>), Martí Català [aut] (ORCID: <`https://orcid.org/0000-0003-3308-9905`>), Xintong Li [aut] (ORCID: <`https://orcid.org/0000-0002-6872-5804`>), Danielle Newby [aut] (ORCID: <`https://orcid.org/0000-0002-3001-1478`>), Nuria Mercade-Besora [aut] (ORCID: <`https://orcid.org/0009-0006-7948-3747`>)

Maintainer Kim López-Güell <`kim.lopez@spc.ox.ac.uk`>

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2026-07-03 06:20:02 UTC

Contents

`addCohortSurvival` *Add time and event status to a cohort table*

Description

Add the columns needed by standard survival modelling functions, such as `survival::Surv()`, to an OMOP cohort table. This is a lower-level helper: it creates `time` and `status` but does not fit a Kaplan-Meier curve or return a `summarised_result`.

Usage

```
addCohortSurvival(
  x,
  cdm,
  outcomeCohortTable,
  outcomeCohortId = 1,
  outcomeDateVariable = "cohort_start_date",
  outcomeWashout = Inf,
  censorOnCohortExit = FALSE,
  censorOnDate = NULL,
  followUpDays = Inf,
  name = NULL
)
```

Arguments

`x`	Cohort table to add survival information to.
`cdm`	CDM reference created by CDMConnector.
`outcomeCohortTable`	Name of the cohort table containing the outcome of interest.

`outcomeCohortId`	ID of event cohorts to include. Only one outcome (and so one ID) can be considered. It can either be a cohort_definition_id value or a cohort_name.
`outcomeDateVariable`	Variable containing date of outcome event. This is usually `"cohort_start_date"`.
`outcomeWashout`	Washout time in days for the outcome. If an individual has an outcome during the washout period before target cohort entry, `status` and `time` will be set to `NA`. Use `Inf` for any prior outcome and `0` for no pre-index washout.
`censorOnCohortExit`	If TRUE, an individual's follow up will be censored at their target cohort exit.
`censorOnDate`	If not NULL, an individual's follow up will be censored at the given date. This can be a scalar Date or the name of a date column in `x`.
`followUpDays`	Number of days to follow up individuals (lower bound 1, upper bound Inf). Follow-up is censored at this value.
`name`	Name of the new table, if NULL a temporary table is returned.

Details

`time` is the number of days from target cohort entry to the first applicable event or censoring date. Censoring can occur at the end of observation, at target cohort exit when `censorOnCohortExit = TRUE`, at `censorOnDate`, or at `followUpDays`. `status` is `1` for people with the outcome event and `0` for censored records. Records with an outcome in the washout window are kept in the table with `time` and `status` set to `NA`, so they can be removed by downstream analyses.

Value

A cohort table with two additional columns. The `time` column contains the number of days to event or censoring. The `status` column indicates whether the patient had the event (`1`) or was censored (`0`).

Examples

```
cdm <- mockMGUS2cdm()
cdm$mgus_diagnosis <- cdm$mgus_diagnosis |>
  addCohortSurvival(
    cdm = cdm,
    outcomeCohortTable = "death_cohort",
    outcomeCohortId = 1
  )

cdm$mgus_diagnosis |>
  dplyr::select(subject_id, cohort_start_date, time, status) |>
  dplyr::collect()
```

`asSurvivalResult` *Convert survival summarised results to a survival-specific format*

Description

Convert the long `omopgenerics::summarised_result` returned by `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()` into a wider `survival_result` object that is easier to inspect manually. The main object contains time-specific estimates when available. Event counts, summary statistics, and attrition are stored as attributes named `"events"`, `"summary"`, and `"attrition"`.

Usage

```
asSurvivalResult(result)
```

Arguments

`result`	A summarised_result object.

Details

The plotting and table functions in CohortSurvival accept both formats. The original `summarised_result` is usually preferable for exporting, binding with other omopgenerics results, and reporting through visOmopResults.

Value

A `survival_result` object.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(
  cdm = cdm,
  targetCohortTable = "mgus_diagnosis",
  targetCohortId = 1,
  outcomeCohortTable = "death_cohort",
  outcomeCohortId = 1,
  eventGap = 7
) |>
asSurvivalResult()
```

`availableSurvivalGrouping` *Variables that can be used for faceting and colouring survival plots*

Description

Variables that can be used for faceting and colouring survival plots

Usage

```
availableSurvivalGrouping(result, varying = FALSE)
```

Arguments

`result`	Survival results
`varying`	If FALSE (default), only variables with non-unique values will be returned, otherwise all available variables will be returned.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(cdm,
                                    targetCohortTable = "mgus_diagnosis",
                                    outcomeCohortTable = "death_cohort")
availableSurvivalGrouping(surv)
```

`estimateCompetingRiskSurvival` *Estimate cumulative incidence with a competing outcome*

Description

Estimate time-to-event probabilities for one or more target cohorts when an event of interest can be precluded by a competing outcome. The target cohort defines the population at risk and the index date for follow-up. The outcome cohort defines the event of interest and the competing outcome cohort defines the event that prevents the event of interest from subsequently occurring.

Usage

```
estimateCompetingRiskSurvival(
  cdm,
  targetCohortTable,
  outcomeCohortTable,
  competingOutcomeCohortTable,
  targetCohortId = NULL,
```

```
  outcomeCohortId = NULL,
  outcomeDateVariable = "cohort_start_date",
  outcomeWashout = Inf,
  competingOutcomeCohortId = NULL,
  competingOutcomeDateVariable = "cohort_start_date",
  competingOutcomeWashout = Inf,
  censorOnCohortExit = FALSE,
  censorOnDate = NULL,
  weight = NULL,
  followUpDays = Inf,
  strata = NULL,
  eventGap = 30,
  estimateGap = 1,
  restrictedMeanFollowUp = NULL,
  minimumSurvivalDays = 1
)
```

Arguments

`cdm`	A CDM reference created by CDMConnector.
`targetCohortTable`	Name of the cohort table containing the target cohorts. The table must be present in `cdm` and contain standard OMOP cohort columns.
`outcomeCohortTable`	Name of the cohort table containing the outcome of interest.
`competingOutcomeCohortTable`	Name of the cohort table containing the competing outcome.
`targetCohortId`	Target cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed. If `NULL`, all non-empty cohorts in `targetCohortTable` are used.
`outcomeCohortId`	Outcome cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed. If `NULL`, all outcome cohorts in `outcomeCohortTable` are used.
`outcomeDateVariable`	Variable containing the outcome event date. This is usually `"cohort_start_date"`, but another date column in the outcome cohort can be used.
`outcomeWashout`	Number of days before target cohort entry used to exclude people with a prior outcome. `Inf` excludes people with any prior outcome before index; `0` applies no pre-index washout.
`competingOutcomeCohortId`	Competing outcome cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed. If `NULL`, all competing outcome cohorts in `competingOutcomeCohortTable` are used.
`competingOutcomeDateVariable`	Variable containing the competing outcome event date.

`competingOutcomeWashout`	Number of days before target cohort entry used to exclude people with a prior competing outcome. `Inf` excludes people with any prior competing outcome before index; `0` applies no pre-index washout.
`censorOnCohortExit`	If TRUE, an individual's follow up will be censored at their target cohort exit date.
`censorOnDate`	If not NULL, an individual's follow up will be censored at the given date. This can be a scalar Date or the name of a date column in the target cohort table.
`weight`	If not NULL, the name of a numeric column in the target cohort table containing observation weights.
`followUpDays`	Number of days to follow up individuals (lower bound 1, upper bound Inf). Follow-up is censored at this value.
`strata`	A list of target cohort column names to stratify by. Each element can be one column name or a character vector of column names for a combined stratum, for example `list("sex", c("age_group", "sex"))`.
`eventGap`	Days between time points for which to report survival events, which are grouped into the specified intervals.
`estimateGap`	Days between time points for which to report survival estimates. First day will be day zero with risk estimates provided for times up to the end of follow-up, with a gap in days equivalent to estimateGap.
`restrictedMeanFollowUp`	Number of days of follow-up to use when calculating restricted mean summaries. See Details.
`minimumSurvivalDays`	Minimum number of days required for the main cohort to contribute to the analysis.

Details

The estimates from competing-risk analyses should be interpreted as cumulative incidence probabilities for the outcome and competing outcome, not as ordinary Kaplan-Meier survival probabilities. The returned object is an `omopgenerics::summarised_result` containing cumulative incidence estimates, event counts, summary statistics, and attrition. Use `asSurvivalResult()` for a wider, survival-specific view.

`restrictedMeanFollowUp` defines the time horizon used for the restricted mean summary. If `restrictedMeanFollowUp = NULL`, the horizon is left to the underlying survival summary. In stratified analyses, this can use a common maximum follow-up time across the fitted curves. A stratum with shorter observed follow-up may therefore have its last estimate carried forward and integrated beyond its own maximum follow-up. This means restricted mean summaries can be larger than the observed follow-up time for that stratum, and comparisons across strata may be misleading. Set a common clinically meaningful value that is supported by follow-up in all groups when restricted means will be compared across cohorts or strata. If the requested horizon is beyond the available follow-up for a curve, the restricted mean is reported as missing.

Value

An `omopgenerics::summarised_result` object with result types `survival_estimates`, `survival_events`, `survival_summary`, and `survival_attrition` when available.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateCompetingRiskSurvival(
  cdm = cdm,
  targetCohortTable = "mgus_diagnosis",
  targetCohortId = 1,
  outcomeCohortTable = "progression",
  outcomeCohortId = 1,
  competingOutcomeCohortTable = "death_cohort",
  competingOutcomeCohortId = 1,
  eventGap = 7
)
```

`estimateSingleEventSurvival`

Estimate survival for a single event of interest

Description

Estimate Kaplan-Meier survival for one or more target cohorts and outcome cohorts in an OMOP Common Data Model reference. The target cohort defines the population at risk and the index date for follow-up. The outcome cohort defines the event of interest.

Usage

```
estimateSingleEventSurvival(
  cdm,
  targetCohortTable,
  outcomeCohortTable,
  targetCohortId = NULL,
  outcomeCohortId = NULL,
  outcomeDateVariable = "cohort_start_date",
  outcomeWashout = Inf,
  censorOnCohortExit = FALSE,
  censorOnDate = NULL,
  weight = NULL,
  followUpDays = Inf,
  strata = NULL,
  eventGap = 30,
  estimateGap = 1,
  restrictedMeanFollowUp = NULL,
```

```
  minimumSurvivalDays = 1
)
```

Arguments

`cdm`	A CDM reference created by CDMConnector.
`targetCohortTable`	Name of the cohort table containing the target cohorts. The table must be present in `cdm` and contain standard OMOP cohort columns.
`outcomeCohortTable`	Name of the cohort table containing the outcome cohorts. The table must be present in `cdm` and contain standard OMOP cohort columns.
`targetCohortId`	Target cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed. If `NULL`, all non-empty cohorts in `targetCohortTable` are used.
`outcomeCohortId`	Outcome cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed. If `NULL`, all outcome cohorts in `outcomeCohortTable` are used.
`outcomeDateVariable`	Variable containing the outcome event date. This is usually `"cohort_start_date"`, but another date column in the outcome cohort can be used.
`outcomeWashout`	Number of days before target cohort entry used to exclude people with a prior outcome. `Inf` excludes people with any prior outcome before index; `0` applies no pre-index washout.
`censorOnCohortExit`	If TRUE, an individual's follow up will be censored at their target cohort exit date.
`censorOnDate`	If not NULL, an individual's follow up will be censored at the given date. This can be a scalar Date or the name of a date column in the target cohort table.
`weight`	If not NULL, the name of a numeric column in the target cohort table containing observation weights to use in the Kaplan-Meier estimation.
`followUpDays`	Number of days to follow up individuals (lower bound 1, upper bound Inf). Follow-up is censored at this value.
`strata`	A list of target cohort column names to stratify by. Each element can be one column name or a character vector of column names for a combined stratum, for example `list("sex", c("age_group", "sex"))`.
`eventGap`	Days between time points for which to report survival events, which are grouped into the specified intervals.
`estimateGap`	Days between time points for which to report survival estimates. First day will be day zero with risk estimates provided for times up to the end of follow-up, with a gap in days equivalent to estimateGap.
`restrictedMeanFollowUp`	Number of days of follow-up to use when calculating restricted mean survival. See Details.

`minimumSurvivalDays`
: Minimum number of days required for the main cohort to contribute to the analysis.

Details

The returned object is an `omopgenerics::summarised_result` containing survival estimates, event counts, summary statistics, and attrition. Use `asSurvivalResult()` when you want a wider, survival-specific view for manual inspection or downstream modelling.

`restrictedMeanFollowUp` defines the time horizon used for the restricted mean survival time. It is calculated as the area under the survival curve up to that horizon. If `restrictedMeanFollowUp = NULL`, the horizon is left to the underlying survival summary. In stratified analyses, this can use a common maximum follow-up time across the fitted curves. A stratum with shorter observed follow-up may therefore have its last survival estimate carried forward and integrated beyond its own maximum follow-up. This means restricted mean survival can be larger than the observed follow-up time for that stratum, and comparisons across strata may be misleading. Set a common clinically meaningful value that is supported by follow-up in all groups when restricted mean survival will be compared across cohorts or strata. If the requested horizon is beyond the available follow-up for a curve, the restricted mean is reported as missing.

Value

An `omopgenerics::summarised_result` object with result types `survival_estimates`, `survival_events`, `survival_summary`, and `survival_attrition` when available.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(
  cdm = cdm,
  targetCohortTable = "mgus_diagnosis",
  targetCohortId = 1,
  outcomeCohortTable = "death_cohort",
  outcomeCohortId = 1,
  eventGap = 7
)
```

`mockMGUS2cdm` *Create mock CDM reference with survival::mgus2 dataset*

Description

Create mock CDM reference with survival::mgus2 dataset

Usage

```
mockMGUS2cdm()
```

Value

CDM reference containing data from the survival::mgus2 dataset

Examples

```
cdm <- mockMGUS2cdm()
cdm$person
```

`optionsTableSurvival` *Additional arguments for the function tableSurvival()*

Description

It provides a list of allowed inputs for .option argument in tableSurvival and their given default value.

Usage

```
optionsTableSurvival()
```

Value

The default .options named list.

Examples

```
{
optionsTableSurvival()
}
```

`plotSurvival` *Plot survival or cumulative incidence results*

Description

Plot the time-specific estimates returned by `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()`. Single-event results are plotted as survival probabilities by default and can be displayed as cumulative failure with `cumulativeFailure = TRUE`. Competing-risk results are cumulative incidence estimates and therefore require `cumulativeFailure = TRUE`.

Usage

```
plotSurvival(
  result,
  ribbon = TRUE,
  facet = NULL,
  colour = NULL,
  cumulativeFailure = FALSE,
  riskTable = FALSE,
  riskInterval = 30,
  logLog = FALSE,
  timeScale = "days",
  type = NULL,
  style = NULL
)
```

Arguments

`result`	Survival results. A `summarised_result` or `survival_result` is accepted.
`ribbon`	If TRUE, add a ribbon using the confidence interval columns.
`facet`	Variables to use for facets.
`colour`	Variables to use for colours.
`cumulativeFailure`	Whether to plot the cumulative failure probability instead of the survival probability.
`riskTable`	Whether to print risk table below the plot.
`riskInterval`	Interval of time to print risk table below the plot. This should be compatible with the `eventGap` used when estimating the result.
`logLog`	If TRUE, the survival probabilities are transformed using the log-log formula.
`timeScale`	The scale of time in the x-axis. Can be "days", "months", or "years".
`type`	Character string specifying the desired plot type. See `visOmopResults::scatterPlot()` for supported types. If `type` = `NULL`, the default backend plot type is used.
`style`	A character string defining the visual theme to apply to the plot. You can set this to NULL to apply the standard ggplot2 default style, or provide a name for one of the package's pre-defined styles. Refer to the plotStyle() function for all available style pre-defined themes. For further customization, you can always modify the returned ggplot object directly.

Details

`facet` and `colour` should refer to columns available after converting the result with `asSurvivalResult()`, for example `target_cohort`, `outcome`, `competing_outcome`, `variable`, or strata columns such as `sex`.

Value

A plot of survival probabilities or cumulative incidence probabilities over time.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(cdm,
                                    targetCohortTable = "mgus_diagnosis",
                                    outcomeCohortTable = "death_cohort")
plotSurvival(surv)
```

`riskTable` *Table with survival events*

Description

`riskTable()` is kept for backwards compatibility. Use `tableSurvivalEvents()` in new code.

Usage

```
riskTable(
  x,
  eventGap = NULL,
  header = c("estimate"),
  type = "gt",
  groupColumn = NULL,
  hide = c("result_id", "estimate_type"),
  style = NULL,
  .options = list()
)
```

Arguments

`x`	Result from estimateSingleEventSurvival or estimateCompetingRiskSurvival.
`eventGap`	Event gap defining the times at which to report the risk table information. Must be one of the eventGap inputs used for the estimation function. If NULL, all available are reported.
`header`	A vector specifying the elements to include in the header. The order of elements matters, with the first being the topmost header. Elements in header can be: • Any of the columns returned by `tableColumns(result)` to create a header for these columns. • Any other input to create an overall header.
`type`	Character string specifying the desired output table format. See `tableType()` for supported table types. If `type = NULL`, global options (set via `setGlobalTableOptions()`) will be used if available; otherwise, a default `'gt'` table is created.

`groupColumn`	Columns to use as group labels, to see options use `tableColumns(result)`. By default, the name of the new group will be the tidy* column names separated by ";". To specify a custom group name, use a named list such as: list("newGroupName" = c("variable_name", "variable_level")). *tidy: The tidy format applied to column names replaces "_" with a space and converts to sentence case. Use `rename` to customise specific column names.
`hide`	Columns to drop from the output table. By default, `result_id` and `estimate_type` are always dropped.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style:** Use the name of a built-in style (e.g., `"darwin"`). See `tableStyle()` for available options. 2. **YAML file path:** Provide the path to an existing `.yml` file defining a new style. 3. **List of custom R code:** Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If `style = NULL`, the function will use global options (see `setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette on the package website.
`.options`	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A tibble containing the risk table information (n_risk, n_events, n_censor) for all times within the event gap specified.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(cdm,
                                    targetCohortTable = "mgus_diagnosis",
                                    outcomeCohortTable = "death_cohort")
riskTable(surv)
```

`tableSurvival` *Table with survival summary*

Description

Create a formatted table from the summary and, optionally, time-specific estimate result types returned by `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()`. For single-event analyses, time-specific rows are survival probabilities. For competing-risk analyses, they are cumulative incidence probabilities for the outcome and competing outcome.

Usage

```
tableSurvival(
  x,
  times = NULL,
  timeScale = "days",
  header = c("estimate"),
  estimates = c("median_survival", "restricted_mean_survival"),
  type = "gt",
  groupColumn = NULL,
  hide = c("result_id", "estimate_type"),
  style = NULL,
  .options = list()
)
```

Arguments

`x`	Result from `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()`. A `summarised_result` or `survival_result` is accepted.
`times`	Times at which to report estimates in the summary table. These must match available estimate times after applying `timeScale`; unavailable times are omitted with a message.
`timeScale`	Time unit to report survival in: days, months, or years.
`header`	A vector specifying the elements to include in the header. The order of elements matters, with the first being the topmost header. Elements in header can be: • Any of the columns returned by `tableColumns(result)` to create a header for these columns. • Any other input to create an overall header.
`estimates`	Character vector specifying which estimates to include in the table. Options include: "median_survival", "restricted_mean_survival", "q0_survival", "q05_survival", "q25_survival", "q75_survival", "q95_survival", "q100_survival". By default it includes c("median_survival", "restricted_mean_survival").
`type`	Character string specifying the desired output table format. See `tableType()` for supported table types. If `type = NULL`, global options (set via `setGlobalTableOptions()`) will be used if available; otherwise, a default `'gt'` table is created.
`groupColumn`	Columns to use as group labels, to see options use `tableColumns(result)`. By default, the name of the new group will be the tidy* column names separated by ";". To specify a custom group name, use a named list such as: list("newGroupName" = c("variable_name", "variable_level")). *tidy: The tidy format applied to column names replaces "_" with a space and converts to sentence case. Use `rename` to customise specific column names.
`hide`	Columns to drop from the output table. By default, `result_id` and `estimate_type` are always dropped.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways:

1. **Pre-defined style:** Use the name of a built-in style (e.g., `"darwin"`). See `tableStyle()` for available options.
2. **YAML file path:** Provide the path to an existing `.yml` file defining a new style.
3. **List of custom R code:** Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If `style = NULL`, the function will use global options (see `setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette on the package website.

`.options`	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Details

Restricted mean survival is taken from the estimation output. Its interpretation depends on the `restrictedMeanFollowUp` value used when the survival result was estimated; use a common value there when comparing restricted means across groups or strata.

Value

A formatted table containing a summary of observed survival or cumulative incidence in the required units.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(cdm,
                                    targetCohortTable = "mgus_diagnosis",
                                    outcomeCohortTable = "death_cohort")
tableSurvival(surv, times = c(50,100,365))
```

---

tableSurvivalAttrition

Display the attrition of a survival result in a visual table

Description

Display the attrition of a survival result in a visual table

Usage

```
tableSurvivalAttrition(
  result,
  type = "gt",
  header = "variable_name",
```

```
  groupColumn = c("cdm_name", "target_cohort", "variable_level"),
  hide = c("estimate_name"),
  style = NULL,
  .options = list()
)
```

Arguments

`result`	A summarised_result object obtained either from `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()`.
`type`	Character string specifying the desired output table format. See `tableType()` for supported table types. If `type = NULL`, global options (set via `setGlobalTableOptions()`) will be used if available; otherwise, a default `'gt'` table is created.
`header`	A vector specifying the elements to include in the header. The order of elements matters, with the first being the topmost header. Elements in header can be: • Any of the columns returned by `tableColumns(result)` to create a header for these columns. • Any other input to create an overall header.
`groupColumn`	Columns to use as group labels, to see options use `tableColumns(result)`. By default, the name of the new group will be the tidy* column names separated by ";". To specify a custom group name, use a named list such as: list("newGroupName" = c("variable_name", "variable_level")). *tidy: The tidy format applied to column names replaces "_" with a space and converts to sentence case. Use `rename` to customise specific column names.
`hide`	Columns to drop from the output table. By default, `result_id` and `estimate_type` are always dropped.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style:** Use the name of a built-in style (e.g., `"darwin"`). See `tableStyle()` for available options. 2. **YAML file path:** Provide the path to an existing `.yml` file defining a new style. 3. **List of custom R code:** Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If `style = NULL`, the function will use global options (see `setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette on the package website.
`.options`	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A visual table

Examples

```
library(CohortSurvival)

cdm <- mockMGUS2cdm()

surv <- estimateSingleEventSurvival(
  cdm = cdm,
  targetCohortTable = "mgus_diagnosis",
  outcomeCohortTable = "death_cohort"
)

tableSurvivalAttrition(surv)
```

`tableSurvivalEvents` *Table with survival events*

Description

Create a formatted table of the number at risk, number of events, and number censored by time interval. The available intervals are controlled by `eventGap` when the survival result is estimated.

Usage

```
tableSurvivalEvents(
  x,
  eventGap = NULL,
  header = c("estimate"),
  type = "gt",
  groupColumn = NULL,
  hide = c("result_id", "estimate_type"),
  style = NULL,
  .options = list()
)
```

Arguments

`x`	Result from estimateSingleEventSurvival or estimateCompetingRiskSurvival.
`eventGap`	Event gap defining the times at which to report the risk table information. Must be one of the eventGap inputs used for the estimation function. If NULL, all available are reported.
`header`	A vector specifying the elements to include in the header. The order of elements matters, with the first being the topmost header. Elements in header can be: • Any of the columns returned by `tableColumns(result)` to create a header for these columns. • Any other input to create an overall header.

`type`	Character string specifying the desired output table format. See `tableType()` for supported table types. If `type = NULL`, global options (set via `setGlobalTableOptions()`) will be used if available; otherwise, a default `'gt'` table is created.
`groupColumn`	Columns to use as group labels, to see options use `tableColumns(result)`. By default, the name of the new group will be the tidy* column names separated by ";". To specify a custom group name, use a named list such as: list("newGroupName" = c("variable_name", "variable_level")). *tidy: The tidy format applied to column names replaces "_" with a space and converts to sentence case. Use `rename` to customise specific column names.
`hide`	Columns to drop from the output table. By default, `result_id` and `estimate_type` are always dropped.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style:** Use the name of a built-in style (e.g., `"darwin"`). See `tableStyle()` for available options. 2. **YAML file path:** Provide the path to an existing `.yml` file defining a new style. 3. **List of custom R code:** Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If `style = NULL`, the function will use global options (see `setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette on the package website.
`.options`	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A tibble containing the risk table information (n_risk, n_events, n_censor) for all times within the event gap specified.

Examples

```
cdm <- mockMGUS2cdm()
surv <- estimateSingleEventSurvival(cdm,
                                    targetCohortTable = "mgus_diagnosis",
                                    outcomeCohortTable = "death_cohort")
tableSurvivalEvents(surv)
```

Index