

Package: survkit (via r-universe)

October 14, 2026

Title A Unified, Extensible Toolkit for Time-to-Event Analysis

Version 0.1.0

Description One clean interface over the established survival stack (parametric accelerated-failure-time models, Cox proportional hazards, competing risks, recurrent events, frailty, flexible splines and a non-parametric Kaplan-Meier baseline), driven by an extensible method registry so new estimators slot in without an API change. Fits return a single typed object with consistent prediction, summary and tidying methods. On top of the fits sit the value layer: cross-method triangulation that reads disagreement as a misspecification signal, restricted mean survival time and cumulative incidence, inverse-probability-of-censoring-weighted predictive scoring, a small-sample power gate that abstains when events are too few to identify the model, a registry conformance contract, and an optional emitter to an ensemble-manifest contract for composition with a wider analytics stack.

License MIT + file LICENSE

Encoding UTF-8

Language en-AU

Roxygen list(markdown = TRUE)

Depends R (>= 4.2.0)

Imports S7, stats, survival

Suggests flexsurv, cmprsk, coxme, flexsurvcure, icenReg, MASS, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://github.com/max578/survkit>

BugReports <https://github.com/max578/survkit/issues>

Config/roxygen2/version 8.1.0

Repository <https://max578.r-universe.dev>

Date/Publication 2026-10-09 22:31:17 UTC
RemoteUrl <https://github.com/max578/survkit>
RemoteRef main
RemoteSha 19e94eff878ba5dc3d8e0be25dcac72ef914dab8

Contents

as_survkit_manifest	2
is_survkit_refusal	3
predict.survkit_fit	4
survkit	5
survkit_contract	6
survkit_cuminc	7
survkit_curve	8
survkit_fit	9
survkit_manifest	10
survkit_methods	11
survkit_power	12
survkit_refusal	13
survkit_register	14
survkit_rmst	15
survkit_score	16
survkit_tidy	17
survkit_triangulate	18
survkit_validate_method	19
Index	21

as_survkit_manifest	<i>Emit an ensemble manifest from a fitted model</i>
---------------------	--

Description

Produce a draw ensemble for orchestra composition. Coefficients are drawn from their Gaussian sampling approximation (mean = estimate, covariance = vcov), and the result is shaped to the ensemble-manifest contract (a parameter draw frame with provenance) so a pesto_ensemble_manifest-compatible adapter can consume it.

Usage

```
as_survkit_manifest(  
  fit,  
  n_draws = 200L,  
  times = NULL,  
  newdata = NULL,  
  seed = NULL  
)
```

Arguments

<code>fit</code>	A survkit_fit .
<code>n_draws</code>	Integer number of draws (default 200L).
<code>times</code>	Optional numeric time grid for the survival outputs. NULL (default) emits a parameters-only manifest.
<code>newdata</code>	Optional single covariate profile (one row) for the survival outputs. Required for the per-draw survival ensemble.
<code>seed</code>	Optional integer RNG seed (recorded in the manifest).

Details

When `times` and a single-profile `newdata` are given and the fit is parametric, spline or cure, the `outputs` slot carries a genuine per-draw survival ensemble: the draws come from the backend's own sampling distribution and each draw's survival is evaluated through the fitted distribution function, so `params` and `outputs` share the same draws. For a Cox-family or non-parametric fit a per-draw survival ensemble would also need the baseline-hazard uncertainty, so `outputs` instead holds the maximum-likelihood survival curve marked with `draw = 0`.

Value

A [survkit_manifest](#) object.

See Also

[survkit_manifest](#) for the returned class, [survkit\(\)](#) for the fit.

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age, survival::lung, method = "weibull")
m <- as_survkit_manifest(fit, n_draws = 50L, seed = 1L)
nrow(m@params)
```

<code>is_survkit_refusal</code>	<i>Is an object a survkit power-gate refusal?</i>
---------------------------------	---

Description

Is an object a survkit power-gate refusal?

Usage

```
is_survkit_refusal(x)
```

Arguments

<code>x</code>	Any object.
----------------	-------------

Value

A single logical.

See Also

[survkit_refusal\(\)](#).

Other model-fitting: [survkit\(\)](#), [survkit_cuminc\(\)](#), [survkit_power\(\)](#), [survkit_refusal\(\)](#), [survkit_rmst\(\)](#), [survkit_score\(\)](#), [survkit_triangularize\(\)](#)

Examples

```
is_survkit_refusal(42)
```

predict.survkit_fit	<i>Predict from the raw backend fit</i>
---------------------	---

Description

Delegate to `stats::predict()` on the backend object underlying a [survkit_fit](#) (e.g. `predict.coxph()`, `predict.survreg()`), passing ... straight through. For a backend-independent survival, hazard or quantile prediction on a common time grid, use [survkit_curve\(\)](#) instead.

Arguments

object	A survkit_fit .
...	Passed to the backend's predict method.

Value

Whatever the backend's predict method returns.

See Also

[survkit_curve\(\)](#) for a backend-independent prediction grid.

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age + sex, survival::lung,
              method = "cox")
head(predict(fit, type = "lp"))
```

Description

One interface over the established survival stack. `method` selects an estimator from the registry ([survkit_methods\(\)](#)); the appropriate backend is checked, a small-sample power gate runs ([survkit_power\(\)](#)), and the fit is returned as a single typed object ([survkit_fit](#)) with consistent `print`, `summary`, `coef`, `vcov`, `predict` and [tidy\(\)](#) methods.

Usage

```
survkit(
  formula,
  data,
  method = "weibull",
  ...,
  check_power = TRUE,
  warn_underpowered = TRUE,
  on_underpowered = c("warn", "abstain")
)
```

Arguments

<code>formula</code>	A model formula, e.g. <code>survival::Surv(time, status) ~ x</code> .
<code>data</code>	A data frame.
<code>method</code>	Character registry key (default <code>"weibull"</code>); see survkit_methods() .
<code>...</code>	Passed to the backend estimator.
<code>check_power</code>	Logical; run the events-per-variable gate (default <code>TRUE</code>).
<code>warn_underpowered</code>	Logical; warn when the gate flags the model under-powered (default <code>TRUE</code>). Ignored when <code>on_underpowered = "abstain"</code> .
<code>on_underpowered</code>	One of <code>"warn"</code> (default) or <code>"abstain"</code> . <code>"warn"</code> preserves the historical behaviour: the model is fitted regardless, with a warning when <code>warn_underpowered</code> is <code>TRUE</code> . <code>"abstain"</code> declines to fit at all when the gate flags the model under-powered, returning a survkit_refusal() instead – the typed token an orchestra-level caller can recognise via <code>is_orchestra_decline()</code> .

Details

The left-hand side of `formula` is a `survival::Surv()` response. Extra arguments in `...` pass straight to the backend estimator, so any backend option remains reachable.

Value

A [survkit_fit](#) object, or a [survkit_refusal\(\)](#) when `on_underpowered = "abstain"` and the power gate is not satisfied.

See Also

[survkit_methods\(\)](#) for the registry, [survkit_power\(\)](#) for the gate.

Other model-fitting: [is_survkit_refusal\(\)](#), [survkit_cuminc\(\)](#), [survkit_power\(\)](#), [survkit_refusal\(\)](#), [survkit_rmst\(\)](#), [survkit_score\(\)](#), [survkit_triangulate\(\)](#)

Examples

```
df <- survival::lung
fit <- survkit(survival::Surv(time, status) ~ age + sex, df, method = "weibull")
fit
coef(fit)
```

survkit_contract

Describe the registry conformance contract

Description

Return the field-level contract that every registered `fit` closure must honour: the names of the fields a normalised fit result carries, so that `survkit()` and every downstream verb can rely on one shape. Use [survkit_validate_method\(\)](#) to check a specific method against it.

Usage

```
survkit_contract()
```

Value

A character vector of the required field names.

See Also

[survkit_validate_method\(\)](#), [survkit_register\(\)](#).

Other method-registry: [survkit_methods\(\)](#), [survkit_register\(\)](#), [survkit_validate_method\(\)](#)

Examples

```
survkit_contract()
```

survkit_cuminc

*Compute cumulative incidence for competing risks***Description**

The Aalen-Johansen estimator of the cumulative incidence function for each competing event, computed from a multi-state `survival::survfit()`. Unlike one minus a cause-specific Kaplan-Meier, this accounts for the competing events and so sums, across causes, to the overall failure probability.

Usage

```
survkit_cuminc(formula, data, ...)
```

Arguments

formula	A model formula whose response is a multi-state <code>survival::Surv(time, event)</code> with event a factor whose first level is censoring. The right-hand side is 1 or grouping variables.
data	A data frame.
...	Passed to <code>survival::survfit()</code> .

Value

A data frame with time, group (the stratum, or "all"), cause, cif (the cumulative incidence) and cif_se (its standard error).

See Also

[survkit\(\)](#) with `method = "fine_gray"` for a regression on the subdistribution hazard.

Other model-fitting: [is_survkit_refusal\(\)](#), [survkit\(\)](#), [survkit_power\(\)](#), [survkit_refusal\(\)](#), [survkit_rmst\(\)](#), [survkit_score\(\)](#), [survkit_triangulate\(\)](#)

Examples

```
mg <- survival::mgus2
mg$etime <- with(mg, ifelse(pstat == 1, ptime, futime))
mg$event <- with(mg, factor(ifelse(pstat == 1, "pcm",
                                ifelse(death == 1, "death", "censor")),
                          levels = c("censor", "pcm", "death")))
ci <- survkit_cuminc(survival::Surv(etime, event) ~ 1, mg)
head(ci)
```

survkit_curve

*Predict survival curves for a fit***Description**

Backend-aware predictions over a time grid for one or more covariate profiles. Parametric, spline and cure fits use the backend's own distribution function and support every type; Cox-family and non-parametric fits use `survival::survfit()` and support the survival curve.

Usage

```
survkit_curve(
  fit,
  newdata = NULL,
  times = NULL,
  type = c("survival", "hazard", "cumhaz", "quantile")
)
```

Arguments

<code>fit</code>	A survkit_fit .
<code>newdata</code>	A data frame of covariate profiles. The default is the backend's reference profile.
<code>times</code>	Numeric times at which to evaluate. The default is an automatic grid from the backend.
<code>type</code>	Character quantity to predict: "survival" (default), "hazard", "cumhaz" or "quantile". The last three need a parametric, spline or cure fit.

Value

A data frame with time, value (named for type when returned by the backend) and a profile index.

See Also

[survkit_tidy\(\)](#) for the coefficient table, [survkit_rmst\(\)](#) for the restricted mean.

Other fit-methods: [survkit_tidy\(\)](#)

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age, survival::lung, method = "weibull")
head(survkit_curve(fit, times = c(100, 300, 500)))
```

survkit_fit

*Represent a fitted survival model***Description**

An S7 object returned by `survkit()`. It normalises every backend's output to one shape: the method and kind, the raw backend fit (kept for backend-specific follow-up), the coefficient vector and covariance, the log-likelihood, the sample and event counts, and the small-sample power verdict.

Usage

```
survkit_fit(
  method = character(0),
  kind = character(0),
  backend = character(0),
  fit = NULL,
  coefficients = integer(0),
  vcov = NULL,
  loglik = NA_real_,
  n = integer(0),
  n_events = integer(0),
  ratio_type = NA_character_,
  aux_pars = character(0),
  power = list(),
  formula = NULL,
  call = NULL
)
```

Arguments

<code>method</code>	Character method name (a registry key, e.g. "weibull").
<code>kind</code>	Character method family: one of "parametric", "spline", "nonparametric", "cox", "frailty", "competing", "recurrent", "cure" or "interval".
<code>backend</code>	Character name of the package that produced the fit.
<code>fit</code>	The raw backend fit object.
<code>coefficients</code>	Named numeric coefficient vector.
<code>vcov</code>	Coefficient covariance matrix (or NULL).
<code>loglik</code>	Numeric log-likelihood (or NA).
<code>n</code>	Integer number of observations.
<code>n_events</code>	Integer number of events (non-censored).
<code>ratio_type</code>	Character code for how an exponentiated covariate coefficient reads: "HR" (hazard ratio), "TR" (time ratio), "OR" (odds ratio), "probit", or NA when the fit has no such coefficients.

aux_pars	Character names of the baseline distribution parameters (the auxiliary parameters the tidier keeps out of the ratio column).
power	A list power verdict from survkit_power() .
formula	The model formula.
call	The originating call.

Value

A survkit_fit S7 object.

See Also

[survkit\(\)](#) (constructs this object), [survkit_tidy\(\)](#), [survkit_curve\(\)](#) (consume it).

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age + sex, survival::lung,
               method = "cox")
fit@method
fit@n_events
```

survkit_manifest	<i>Represent an ensemble-manifest emission</i>
------------------	--

Description

A self-contained, manifest-compatible emission for orchestra composition: a draw ensemble of parameters and predicted survival, with provenance. Produced by [as_survkit_manifest\(\)](#); convertible to a pesto_ensemble_manifest when that contract is available (see the manifest emitter).

Usage

```
survkit_manifest(
  params = data.frame(),
  outputs = NULL,
  time_grid = numeric(0),
  method = character(0),
  emitter_version = character(0),
  seed = NA_integer_
)
```

Arguments

params	Data frame of parameter draws (n_draw rows).
outputs	Data frame of predicted survival draws on a time grid.
time_grid	Numeric vector of evaluation times.
method	Character method name.
emitter_version	Character package version that emitted the manifest.
seed	Integer RNG seed.

Value

A survkit_manifest S7 object.

See Also

[as_survkit_manifest\(\)](#), which constructs this object.

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age, survival::lung,
              method = "weibull")
man <- as_survkit_manifest(fit, n_draws = 50L)
man@method
```

survkit_methods	<i>List the registered methods</i>
-----------------	------------------------------------

Description

List the registered methods

Usage

```
survkit_methods()
```

Value

A data frame of method, kind, backend, available (is the backend installed) and note.

See Also

[survkit_register\(\)](#) to add a method.

Other method-registry: [survkit_contract\(\)](#), [survkit_register\(\)](#), [survkit_validate_method\(\)](#)

Examples

```
survkit_methods()
```

survkit_power	<i>Assess a fit's small-sample power (events per variable)</i>
---------------	--

Description

Assess whether a survival model is identifiable from the available events. Returns the event count, the predictor count, the events-per-variable ratio and a verdict. `survkit()` attaches this to every fit and warns when a model is under-powered.

Usage

```
survkit_power(formula, data, min_epv = 10)
```

Arguments

formula	A model formula with a <code>survival::Surv()</code> response.
data	A data frame.
min_epv	Numeric events-per-variable floor (default 10).

Details

The event count comes from the last column of the `survival::Surv()` response. When that column is a multi-state factor (competing risks) the event count cannot be read as a single 0/1 flag, so the verdict is "unknown" rather than a guess. For a counting-process response (recurrent events) the count is the number of event rows, not the number of distinct subjects.

Value

A list: `n`, `n_events`, `n_predictors`, `events_per_variable`, `min_epv`, `underpowered` (logical), `verdict` ("ok" / "underpowered" / "unknown") and `advice`.

References

Peduzzi P, Concato J, Kemper E, Holford TR, Feinstein AR. A simulation study of the number of events per variable in logistic regression analysis. *Journal of Clinical Epidemiology* 1996; 49(12): 1373-1379. doi:[10.1016/S08954356\(96\)002363](https://doi.org/10.1016/S08954356(96)002363)

See Also

`survkit()`, which runs this gate on every fit.

Other model-fitting: `is_survkit_refusal()`, `survkit()`, `survkit_cuminc()`, `survkit_refusal()`, `survkit_rmst()`, `survkit_score()`, `survkit_triangulate()`

Examples

```
df <- data.frame(time = rexp(40), status = rbinom(40, 1, 0.3),
                  x1 = rnorm(40), x2 = rnorm(40))
survkit_power(survival::Surv(time, status) ~ x1 + x2, df)$verdict
```

survkit_refusal	<i>A classed refusal from the small-sample power gate</i>
-----------------	---

Description

Construct the typed "will not fit" token `survkit()` returns when `on_underpowered = "abstain"` and the events-per-variable gate (`survkit_power()`) flags the model as under-powered. Not signalled as a condition – returned as an ordinary value, so a caller inspects it with `is_survkit_refusal()` rather than a `tryCatch()`.

Usage

```
survkit_refusal(power)
```

Arguments

`power` A power-gate result from `survkit_power()`, already flagged underpowered.

Value

A classed list with class `c("survkit_refusal", "orchestra_refusal", "error", "condition")` and elements `reason`, `n`, `n_events`, `n_predictors`, `events_per_variable`, `min_epv`, `underpowered` (always TRUE), `message`.

See Also

`survkit()`, `survkit_power()`, `is_survkit_refusal()`.

Other model-fitting: `is_survkit_refusal()`, `survkit()`, `survkit_cuminc()`, `survkit_power()`, `survkit_rmst()`, `survkit_score()`, `survkit_triangulate()`

Examples

```
df <- data.frame(time = rexp(20), status = rbinom(20, 1, 0.2),
                  x1 = rnorm(20), x2 = rnorm(20))
p <- survkit_power(survival::Surv(time, status) ~ x1 + x2, df)
r <- survkit_refusal(p)
is_survkit_refusal(r)
```

survkit_register	<i>Register a survival method</i>
------------------	-----------------------------------

Description

Add (or override) an estimator in the method registry. The fit closure receives (formula, data, ...) and must return a list with raw (the backend fit), coef, vcov, loglik, n, n_events, kind and backend – build it with the package's internal normaliser or by hand.

Usage

```
survkit_register(name, kind, backend, fit, note = "")
```

Arguments

name	Character key, e.g. "weibull".
kind	Character family: "parametric", "cox", "competing", "recurrent", "frailty", "spline", "cure", "interval".
backend	Character name of the backend package the fit closure needs (checked via requireNamespace() before the fit runs).
fit	A function (formula, data, ...) returning the normalised list.
note	Character one-line description (shown by survkit_methods()).

Value

Invisibly, the method name.

See Also

[survkit_methods\(\)](#) to list what is registered.

Other method-registry: [survkit_contract\(\)](#), [survkit_methods\(\)](#), [survkit_validate_method\(\)](#)

Examples

```
# register a trivial exponential alias backed by survival::survreg
survkit_register("exp_survreg", "parametric", "survival",
  fit = function(formula, data, ...) {
    raw <- survival::survreg(formula, data = data, dist = "exponential", ...)
    list(raw = raw, coef = stats::coef(raw), vcov = stats::vcov(raw),
      loglik = as.numeric(stats::logLik(raw)),
      n = nrow(data), n_events = NA_integer_,
      kind = "parametric", backend = "survival")
  })
"exp_survreg" %in% survkit_methods()$method
```

survkit_rmst

*Compute the restricted mean survival time***Description**

The restricted mean survival time (RMST) to a horizon τ : the area under the survival curve on $[0, \tau]$, a decision-relevant summary that stays defined when the median is not reached. Parametric, spline and cure fits use the backend's own RMST with its delta-method interval; Cox-family and non-parametric fits use the area under `survival::survfit()` with its standard error.

Usage

```
survkit_rmst(fit, tau, newdata = NULL)
```

Arguments

<code>fit</code>	A survfit_fit .
<code>tau</code>	Numeric horizon (a single positive time) the mean is restricted to.
<code>newdata</code>	A data frame of covariate profiles for the model-based fits. The default is the backend's reference profile.

Details

$$RMST(\tau) = \int_0^{\tau} S(t) dt$$

where $S(t)$ is the (model-based or Kaplan-Meier) survival function. Restricting the integral to $[0, \tau]$ keeps the summary defined and identifiable from the observed follow-up even when the survival curve never crosses one-half.

Value

A data frame with `tau`, `group` (or `profile`), `rmst`, `se`, `conf_low` and `conf_high`.

See Also

[survkit_curve\(\)](#) for the curve the mean integrates.

Other model-fitting: [is_survkit_refusal\(\)](#), [survkit\(\)](#), [survkit_cuminc\(\)](#), [survkit_power\(\)](#), [survkit_refusal\(\)](#), [survkit_score\(\)](#), [survkit_triangulate\(\)](#)

Examples

```
fit <- survkit(survival::Surv(time, status) ~ 1, survival::lung, method = "km")
survkit_rmst(fit, tau = 365)
```

survkit_score	<i>Score a fit's predictions</i>
---------------	----------------------------------

Description

Evaluate a fitted model's predictions at one or more horizons: the concordance index (discrimination) and the inverse-probability-of-censoring-weighted Brier score with its index of prediction accuracy (calibration and accuracy against a covariate-free Kaplan-Meier reference).

Usage

```
survkit_score(fit, data, times)
```

Arguments

fit	A survfit with a right-censored survival::Surv(time, status) response.
data	A data frame carrying the response and covariates to score on (in-sample if it is the training data, out-of-sample otherwise).
times	Numeric horizons at which to score.

Details

The inverse-probability-of-censoring-weighted Brier score at horizon t is

$$BS(t) = \frac{1}{n} \sum_{i=1}^n \left[\frac{(0 - \hat{S}(t | x_i))^2}{\hat{G}(T_i)} \mathbf{1}(T_i \leq t, \delta_i = 1) + \frac{(1 - \hat{S}(t | x_i))^2}{\hat{G}(t)} \mathbf{1}(T_i > t) \right]$$

where $\hat{S}(\cdot | x_i)$ is the predicted survival probability, $\hat{G}$ is the Kaplan-Meier estimator of the *censoring* survival function, and the index of prediction accuracy is $IPA(t) = 1 - BS(t)/BS_0(t)$ against a covariate-free reference BS_0 .

Value

A data frame with time, cindex (Harrell's C using predicted risk at that horizon), brier (IPCW Brier) and ipa (index of prediction accuracy).

References

Graf E, Schmoor C, Sauerbrei W, Schumacher M. Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine* 1999; 18(17-18): 2529-2545. doi:10.1002/(SICI)10970258(19990915/30)18:17/18<2529::AIDSIM274>3.0.CO;25

See Also

[survfit_curve\(\)](#) for the predictions scored here.

Other model-fitting: [is_survfit_refusal\(\)](#), [survfit\(\)](#), [survfit_cuminc\(\)](#), [survfit_power\(\)](#), [survfit_refusal\(\)](#), [survfit_rmst\(\)](#), [survfit_triangular\(\)](#)

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age + sex, survival::lung,
              method = "cox")
survkit_score(fit, survival::lung, times = c(200, 400, 600))
```

survkit_tidy

*Tidy a fitted model into a coefficient table***Description**

Return the coefficient table with standard errors, the exponentiated effect and Wald confidence intervals. Covariate rows and baseline distribution parameters are separated by a component column, and the ratio_type column records whether a covariate effect reads as a hazard ratio (proportional hazards) or a time ratio (accelerated failure time). Auxiliary parameters (a log-scale, a shape) are never exponentiated into the ratio column, so a parametric fit no longer reports a meaningless "ratio" for its scale.

Usage

```
survkit_tidy(fit, conf_level = 0.95)
```

Arguments

fit	A survfit .
conf_level	Confidence level for the Wald interval (default 0.95).

Value

A data frame with columns term, component ("covariate" / "auxiliary"), estimate, std_error, ratio_type, ratio, conf_low and conf_high. For auxiliary parameters ratio_type, ratio and the interval are NA.

See Also

[survkit_curve\(\)](#) for predicted survival.

Other fit-methods: [survkit_curve\(\)](#)

Examples

```
fit <- survkit(survival::Surv(time, status) ~ age + sex, survival::lung, method = "cox")
survkit_tidy(fit)
```

survkit_triangularate *Triangularate a survival question across methods*

Description

Fit the same data under several methods and compare them on the restricted mean survival time (and, if a landmark time is given, on survival at that time). The relative spread across methods is a model-misspecification signal: when the methods disagree by more than threshold, the fit is flagged.

Usage

```
survkit_triangularate(
  formula,
  data,
  methods = c("weibull", "cox", "km"),
  tau = NULL,
  time_star = NULL,
  newdata = NULL,
  threshold = 0.1,
  ...
)
```

Arguments

formula	A model formula with a <code>survival::Surv()</code> response.
data	A data frame.
methods	Character vector of registry keys to compare (default <code>c("weibull", "cox", "km")</code>). Methods whose backend is missing, or that fail to fit, are dropped with a warning.
tau	Numeric horizon for the restricted mean. The default is the 90th percentile of the observed times.
time_star	Optional landmark time; when given, survival at <code>time_star</code> is compared as well.
newdata	Optional single covariate profile for the model-based quantities.
threshold	Numeric relative-spread threshold above which the methods are flagged as disagreeing (default 0.1, i.e. 10 percent).
...	Passed to each <code>survkit()</code> fit.

Value

A `survkit_triangularation` list with the per-method quantities, the disagreement summary, and a logical `flagged`. It has a `print` method.

See Also

[survkit_rmst\(\)](#) for the quantity compared, [survkit\(\)](#) for a single fit.

Other model-fitting: [is_survkit_refusal\(\)](#), [survkit\(\)](#), [survkit_cuminc\(\)](#), [survkit_power\(\)](#), [survkit_refusal\(\)](#), [survkit_rmst\(\)](#), [survkit_score\(\)](#)

Examples

```
tri <- survkit_triangularate(survival::Surv(time, status) ~ age + sex,
                             survival::lung, methods = c("weibull", "cox", "km"),
                             tau = 500)

tri
```

survkit_validate_method

Validate a registered method against the contract

Description

Fit a registered method on the supplied data and check that its normalised result honours the registry contract ([survkit_contract\(\)](#)): every required field is present and of the right type. When the method's backend is not installed the check is skipped rather than failed.

Usage

```
survkit_validate_method(name, formula, data, ...)
```

Arguments

name	Character registry key to validate.
formula	A model formula appropriate for the method (competing-risks and recurrent methods need the matching multi-state or counting-process response).
data	A data frame to fit on.
...	Passed to the method's fit closure.

Value

A list with method, ok (TRUE / FALSE, or NA when skipped), missing (any absent fields), failed (fields failing their type check) and reason (why it was skipped, if it was).

See Also

[survkit_contract\(\)](#) for the contract, [survkit_register\(\)](#) to add a method.

Other method-registry: [survkit_contract\(\)](#), [survkit_methods\(\)](#), [survkit_register\(\)](#)

Examples

```
survkit_validate_method("cox", survival::Surv(time, status) ~ age,  
  survival::lung)
```

Index

- * **fit-methods**
 - survkit_curve, 8
 - survkit_tidy, 17
- * **method-registry**
 - survkit_contract, 6
 - survkit_methods, 11
 - survkit_register, 14
 - survkit_validate_method, 19
- * **model-fitting**
 - is_survkit_refusal, 3
 - survkit, 5
 - survkit_cuminc, 7
 - survkit_power, 12
 - survkit_refusal, 13
 - survkit_rmst, 15
 - survkit_score, 16
 - survkit_triangulate, 18
- as_survkit_manifest, 2
- as_survkit_manifest(), 10, 11
- is_survkit_refusal, 3
- is_survkit_refusal(), 6, 7, 12, 13, 15, 16, 19
- predict.survkit_fit, 4
- survkit, 5
- survkit(), 3, 4, 7, 9, 10, 12, 13, 15, 16, 18, 19
- survkit_contract, 6
- survkit_contract(), 11, 14, 19
- survkit_cuminc, 7
- survkit_cuminc(), 4, 6, 12, 13, 15, 16, 19
- survkit_curve, 8
- survkit_curve(), 4, 10, 15–17
- survkit_fit, 3–6, 8, 9, 15–17
- survkit_manifest, 3, 10
- survkit_methods, 11
- survkit_methods(), 5, 6, 14, 19
- survkit_power, 12
- survkit_power(), 4–7, 10, 13, 15, 16, 19
- survkit_refusal, 13
- survkit_refusal(), 4–7, 12, 15, 16, 19
- survkit_register, 14
- survkit_register(), 6, 11, 19
- survkit_rmst, 15
- survkit_rmst(), 4, 6–8, 12, 13, 16, 19
- survkit_score, 16
- survkit_score(), 4, 6, 7, 12, 13, 15, 19
- survkit_tidy, 17
- survkit_tidy(), 8, 10
- survkit_triangulate, 18
- survkit_triangulate(), 4, 6, 7, 12, 13, 15, 16
- survkit_validate_method, 19
- survkit_validate_method(), 6, 11, 14
- tidy(), 5