Package: weightedsurv (via r-universe)

November 21, 2026

Title Survival Analysis with Subject-Specific (Case Weights) and Time-Dependent Weighting

Version 0.2.0

Description Provides survival analysis functions with support for time-dependent and subject-specific (e.g., propensity score) weighting. Implements weighted estimation for Cox models, Kaplan-Meier survival curves, and treatment differences with point-wise and simultaneous confidence bands. Includes restricted mean survival time (RMST) comparisons evaluated across all potential truncation times with both point-wise and simultaneous confidence bands. See Cole, S. R. & Hernán, M. A. (2004) <doi:10.1016/j.cmpb.2003.10.004> for methodological background.

License MIT + file LICENSE

URL https://larry-leon.github.io/weightedsurv/, https://github.com/larry-leon/weightedsurv

BugReports https://github.com/larry-leon/weightedsurv/issues

Encoding UTF-8

Imports ggplot2, graphics, grDevices, rlang, stats, survival, utils

Suggests adjustedCurves, data.table, DiagrammeR, dplyr, gt, knitr, pammtools, rmarkdown, testthat (>= 3.1.7)

Config/Needs/website doFuture, foreach, future, future.callr, gsDesign2, htmltools, parallelly, simtrial, tibble, tictoc, tidyr, katex

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Larry Leon [aut, cre]

Maintainer Larry Leon <larry.leon.05@post.harvard.edu>

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

Date/Publication 2026-11-20 22:10:02 UTC

RemoteUrl https://github.com/cran/weightedsurv

RemoteRef HEAD

RemoteSha 00feaf89393496c3d455deb9ce90aa7b9199a816

Contents

add_legends . . . 3
add_median_annotation . . . 4
add_risk_table . . . 5
calculate_risk_event_counts . . . 6
check_km_curve . . . 7
check_results . . . 7
ci_cox . . . 9
count_weighted . . . 10
cox_rhogamma . . . 10
cox_rhogamma_resample . . . 15
cox_score_rhogamma . . . 17
create_baseline_table . . . 18
cumulative_rmst_bands . . . 20
df_counting . . . 24
extract_and_calc_weights . . . 32
extract_group_data . . . 32
find_cox_root . . . 33
format_pval . . . 33
get_censoring_and_events . . . 34
get_dfcounting . . . 34
get_event_risk_matrices . . . 35
get_riskpoints . . . 35
get_validated_weights . . . 36
get_weights . . . 36
KM_diff . . . 37
KM_estimates . . . 42
KM_plot_2sample_weighted_counting . . . 43
km_quantile . . . 46
km_quantile_table . . . 47
kmq_calculations . . . 48
N_rhogamma . . . 48
plot_km . . . 49
plot_km_confint_polygon . . . 50
plot_km_curves_counting . . . 50
plot_weight_schemes . . . 52
plot_weighted_km . . . 53
plotKM.band_subgroups . . . 54
ps_weights . . . 59

resampling_survival 61
risk_weighted 62
safe_run 63
score_calculation 63
validate_input 64
validate_scheme_params 64
wlr_cumulative 65
wlr_dhat_estimates 68
wt.rg.S 72

Index **75**

`add_legends` *Add legends to KM plot*

Description

Adds legends for Cox model, log-rank test, and arms to a Kaplan-Meier plot.

Usage

```
add_legends(
  dfcount,
  show.cox,
  cox.cex,
  put.legend.cox,
  show.logrank,
  logrank.cex,
  put.legend.lr,
  show_arm_legend,
  arms,
  col.1,
  col.0,
  ltys,
  lwds,
  arm.cex,
  put.legend.arms
)
```

Arguments

`dfcount`	List with results, typically output from a survival analysis function. Should contain elements such as `cox_results`, `z.score`, and `zlogrank_text`.
`show.cox`	Logical; show Cox legend.
`cox.cex`	Numeric; Cox legend size.
`put.legend.cox`	Character; Cox legend position (e.g., "topright").
`show.logrank`	Logical; show logrank legend.

logrank.cex	Numeric; logrank legend size.
put.legend.lr	Character; logrank legend position (e.g., "topleft").
show_arm_legend	Logical; show arm legend.
arms	Character vector of arm labels.
col.1	Color for group 1.
col.0	Color for group 0.
ltys	Line types.
lwds	Line widths.
arm.cex	Numeric; arm legend size.
put.legend.arms	Character; arm legend position (e.g., "left").

Value

Invisibly returns NULL. Used for plotting side effects.

---

add_median_annotation *Add median annotation to KM plot*

Description

Adds median survival annotation to a Kaplan-Meier plot.

Usage

```
add_median_annotation(
  medians_df,
  med.digits,
  med.cex,
  med.font,
  xmed.fraction,
  ymed.offset
)
```

Arguments

medians_df	Data frame with quantile results. Should contain columns quantile, lower, upper, and group.
med.digits	Integer; number of digits to display for median and confidence interval.
med.cex	Numeric; text size for median annotation.
med.font	Integer; font for median annotation.
xmed.fraction	Numeric; fraction of the x-axis for annotation placement (e.g., 0.8 for 80\% to the right).
ymed.offset	Numeric; offset from the top of the plot for annotation placement.

Value

Invisibly returns NULL. Used for plotting side effects.

`add_risk_table` *Add risk table annotation to KM plot*

Description

Adds risk set counts for two groups to a Kaplan-Meier plot.

Usage

```
add_risk_table(
  risk.points,
  rpoints0,
  rpoints1,
  col.0,
  col.1,
  risk.cex,
  ymin,
  risk_offset,
  risk_delta,
  y.risk0,
  y.risk1
)
```

Arguments

`risk.points`	Numeric vector of risk time points.
`rpoints0`	Numeric vector of risk set counts for group 0.
`rpoints1`	Numeric vector of risk set counts for group 1.
`col.0`	Color for group 0.
`col.1`	Color for group 1.
`risk.cex`	Numeric; text size for risk table.
`ymin`	Numeric; minimum y value.
`risk_offset`	Numeric; offset for risk table.
`risk_delta`	Numeric; delta for risk table.
`y.risk0`	Numeric; y position for group 0 risk table.
`y.risk1`	Numeric; y position for group 1 risk table.

Value

Invisibly returns NULL. Used for plotting side effects.

`calculate_risk_event_counts` *Calculate risk set and event counts at time points*

Description

Calculates risk set and event counts for a group at specified time points, with variance estimation. Passed to `KM_estimates`, these give the Kaplan-Meier estimate only when `at_points` holds every event time of the group.

Usage

```
calculate_risk_event_counts(U, D, W, at_points, draws = 0, seedstart = 816951)
```

Arguments

`U`	Numeric vector of times for group.
`D`	Numeric vector of event indicators for group.
`W`	Numeric vector of weights for group.
`at_points`	Numeric vector of time points.
`draws`	Number of draws for variance estimation (default 0).
`seedstart`	Random seed for draws (default 816951).

Details

`sig2w_multiplier` is, with no weights or all weights equal, the per-time-point Greenwood term $dN(s)/[Y(s)\{Y(s)-dN(s)\}]$ (0 where $Y(s)=0$ or $Y(s)-dN(s)\le 10^{-10}Y(s)$). With weights that are not all equal it is the successive differences, along `at_points`, of $\sum_i w_i^2\psi_i(t)^2$ (the robust variance below divided by $\hat{S}(t)^2$), or, when `draws > 0`, of the variance over draws of $\sum_i G_i w_i\psi_i(t)$; individual differences may be negative, only their cumulative sum is meaningful.

Value

List with ybar (risk set counts), nbar (event counts), sig2w_multiplier (variance term: its cumulative sum times $\hat{S}(t)^2$ is the variance of the Kaplan-Meier estimate, see `KM_estimates`).

Variance of the Kaplan-Meier estimate

With no weights, or with all weights equal, the variance of a Kaplan-Meier estimate is Greenwood's formula, and these results are unchanged. When the weights are not all equal it is the robust (infinitesimal-jackknife) variance

$$\widehat{Var}[\hat{S}(t)] = \hat{S}(t)^2 \sum_i w_i^2 \psi_i(t)^2,$$

$$\psi_i(t) = \frac{D_i 1(U_i \le t)}{Y_w(U_i) - dN_w(U_i)} - \sum_{s \le \min(t,U_i)} \frac{dN_w(s)}{Y_w(s)\{Y_w(s) - dN_w(s)\}},$$

where U_i, D_i and w_i are subject i's time, event indicator and weight, $Y_w(s)$ is the weighted number at risk, $dN_w(s)$ the weighted number of events at s, the sum runs over event times, and a term is 0 where $Y_w(s) - dN_w(s) \leq 10^{-10} Y_w(s)$. It equals the standard error of `survival::survfit(weights = w, robust = TRUE)` (on the time-fixed data), reduces to Greenwood's formula at equal weights, and does not depend on the scale of the weights. With `draws > 0` and unequal weights the variance is instead the variance over resampling draws of $\sum_i G_i w_i \psi_i(t)$ (standard-normal multipliers G_i), which estimates the same quantity; resampled curves for weighted data are built from the same per-subject contributions (see `resampling_survival`). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the standard error (section "Estimated weights" of `df_counting`). When the weights are extreme (low effective sample size) the standard error can be too small under either option.

`check_km_curve` *Check KM curve for validity*

Description

Checks that a Kaplan-Meier curve is valid (values in [0,1], non-increasing).

Usage

```
check_km_curve(S.KM, group_name = "Group", stop_on_error = TRUE)
```

Arguments

`S.KM`	Numeric vector of survival probabilities.
`group_name`	Character; name of the group.
`stop_on_error`	Logical; whether to stop on error (default TRUE).

Value

None. Stops or warns if invalid.

`check_results` *Check and Compare Statistical Test Results*

Description

Calculates and compares three equivalent test statistics from weighted survival analysis: the squared standardized weighted log-rank statistic, the log-rank chi-squared statistic, and the squared Cox model z-score. These should be approximately equal under correct implementation.

Usage

```
check_results(dfcount, verbose = TRUE)
```

Arguments

dfcount	A list or data frame from df_counting containing: **lr** Weighted log-rank statistic **sig2_lr** Variance of the weighted log-rank statistic **z.score** Standardized z-score from Cox model **logrank_results** List containing chisq, the log-rank chi-squared statistic
verbose	Logical; if TRUE, prints the comparison table to the console. Default: TRUE.

Details

This function serves as a diagnostic tool to verify computational consistency. The three statistics should be numerically equivalent (within rounding error):

$$(lr/\sqrt{sig2_lr})^2 \approx logrank_chisq \approx z.score^2$$

Discrepancies between these values may indicate:

- Numerical instability in variance estimation
- Incorrect weighting scheme application
- Data processing errors

Value

A data frame with one row and three columns, returned invisibly:

zlr_sq Squared standardized weighted log-rank: $(lr/\sqrt{sig2_lr})^2$

logrank_chisq Chi-squared statistic from log-rank test

zCox_sq Squared z-score from Cox model: $z.score^2$

Note

This function is primarily used for package development and validation. End users typically don't need to call it directly.

See Also

df_counting for generating the input object

Examples

```
# After running df_counting
library(survival)
data(veteran)
veteran$treat <- as.numeric(veteran$trt) - 1

result <- df_counting(
  df = veteran,
  tte.name = "time",
```

```
  event.name = "status",
  treat.name = "treat"
)

# Check consistency of test statistics
check_results(result)

# Store results without printing
stats_comparison <- check_results(result, verbose = FALSE)
print(stats_comparison)


# Simple example with constructed data
dfcount_example <- list(
  lr = 2.5,
  sig2_lr = 1.0,
  z.score = 2.5,
  logrank_results = list(chisq = 6.25)
)
check_results(dfcount_example)
```

---

ci_cox *Confidence interval for Cox model estimate*

Description

Calculates the confidence interval for a Cox model log hazard ratio estimate.

Usage

```
ci_cox(bhat, sig_bhat, alpha = 0.05, verbose = FALSE)
```

Arguments

bhat	Estimated log hazard ratio.
sig_bhat	Standard error of the estimate.
alpha	Significance level (default: 0.05).
verbose	Logical; if TRUE, prints interval.

Value

Data frame with log hazard ratio, standard error, hazard ratio, lower and upper confidence limits.

count_weighted *Weighted counting process*

Description

Computes the weighted count of events up to a specified time.

Usage

```
count_weighted(x, y, w = rep(1, length(y)))
```

Arguments

x	Time point.
y	Vector of event/censoring times.
w	Weights (default 1).

Value

Weighted count of events up to time x.

cox_rhogamma *Weighted Cox Model with Rho-Gamma Weights*

Description

Fits a weighted Cox proportional hazards model using flexible time-dependent weights (e.g., Fleming-Harrington, Magirr-Burman). Supports resampling-based inference for variance estimation and bias correction.

Usage

```
cox_rhogamma(
  dfcount,
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 0.5),
  draws = 0,
  alpha = 0.05,
  verbose = FALSE,
  lr.digits = 4
)
```

Arguments

dfcount	List; output from df_counting containing counting process data including time, delta, z, w_hat, survP_all, survG_all, etc.
scheme	Character; weighting scheme. See wt.rg.S for options. Default: "fh".
scheme_params	List; parameters for the selected scheme. Default: list(rho = 0, gamma = 0.5). Required parameters depend on scheme: • "fh": rho, gamma • "MB": mb_tstar • "custom_time": t.tau, w0.tau, w1.tau
draws	Integer; number of resampling draws for variance estimation and bias correction. If 0, only asymptotic inference is performed. Default: 0.
alpha	Numeric; significance level for confidence intervals. Default: 0.05.
verbose	Logical; whether to print detailed output. Default: FALSE.
lr.digits	Integer; number of decimal places for formatted output. Default: 4.

Details

This function solves the weighted Cox partial likelihood score equation: U(beta) = sum_i w_i K_i (dN_0/Y_0 - dN_1/Y_1) = 0

where K_i are time-dependent weights and Y_j, dN_j are risk sets and event counts for group j.

When draws > 0, the function performs resampling to:

- Estimate finite-sample variance (more accurate than asymptotic)
- Compute bias correction for $\hat{\beta}$
- Provide improved confidence intervals for small samples

The score test at $\beta = 0$ corresponds to the weighted log-rank test.

With subject weights that are not all equal (dfcount$w_hat), the variance of the score at 0 and at $\hat{\beta}$ is the sum of squared per-subject score contributions (section below), so fit$sig2_score, fit$z.score, fit$sig_bhat_asy, hr_ci_asy, zlogrank_text and cox_text_asy differ from versions before 0.2.0; with draws > 0 the resampled scores are built from the same contributions (cox_rhogamma_resample), so the resampling standard error estimates the same quantity as sig_bhat_asy. The estimate bhat, the scores u.beta and u.zero and the time weights wt_rg do not change. fit$z.score equals df_counting()$z.score for the same scheme. With no weights or all weights equal, every result is as before.

When dfcount was made with weight_variance = "estimated" (df_counting with a ps_weights object), the estimation of the weights is carried into fit$sig2_score, fit$z.score, z.score, fit$sig_bhat_asy, hr_ci_asy, zlogrank_text and cox_text_asy, and with draws > 0 into the resampled scores and everything computed from them (section "Estimated weights"). bhat, the scores, the information and the time weights are unchanged. The result's element weight_variance says which option was used. A direct call to cox_score_rhogamma is not affected.

Value

A list containing:

fit List with fitted model components:

- `bhat`: Estimated log hazard ratio
- `sig_bhat_asy`: Asymptotic standard error
- `u.zero`: Score statistic at beta=0 (log-rank)
- `z.score`: Standardized score statistic
- `sig2_score`: Variance of score statistic
- `wt_rg`: Vector of time-dependent weights
- `bhat_debiased`: Bias-corrected estimate (if draws > 0)
- `sig_bhat_star`: Resampling-based standard error (if draws > 0)

hr_ci_asy Data frame with asymptotic HR and CI

hr_ci_star Data frame with resampling-based HR and CI (if draws > 0)

cox_text_asy Formatted string with HR and asymptotic CI

cox_text_star Formatted string with HR and resampling CI (if draws > 0)

z.score_debiased Bias-corrected score statistic (if draws > 0)

zlogrank_text Formatted log-rank test result

weight_variance `"known"`, or `"estimated"` when `dfcount` was made with `weight_variance = "estimated"`

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i\Big(D_i K(U_i)\{Z_i - \bar{z}(U_i,\beta)\} - e^{\beta Z_i}\sum_{s\le U_i} K(s)\{Z_i - \bar{z}(s,\beta)\}d\hat{\Lambda}(s,\beta)\Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s,\beta) = \sum_j a_j 1(U_j \ge s)e^{\beta Z_j}$, $\bar{z}(s,\beta) = \sum_j a_j 1(U_j \ge s)Z_j e^{\beta Z_j}/S_0(s,\beta)$ and $d\hat{\Lambda}(s,\beta) = \sum_j a_j D_j 1(U_j = s)/S_0(s,\beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2/i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t) \, dw_k, \qquad \Phi_i(t) = 1(i \in g) \, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{ \hat{S}_1(t) \Phi_i^1(t) - \hat{S}_0(t) \Phi_i^0(t) \}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t) \sum_{i=1}^{n} G_i \Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j} \, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i \psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i \{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j \{c_j(t)/a_j\} dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j) dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)
- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With `"estimated"` the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (`df_counting()$ess`) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

Note

The treatment variable in `dfcount` must be coded as 0=control, 1=experimental.

References

Magirr, D. and Burman, C. F. (2019). Modestly weighted logrank tests. Statistics in Medicine, 38(20), 3782-3790.

See Also

df_counting for preprocessing wt.rg.S for weighting schemes cox_score_rhogamma for score function

Other survival_analysis: KM_diff(), df_counting(), ps_weights(), wt.rg.S()

Other weighted_tests: df_counting(), wt.rg.S()

Examples

```
# First get counting process data
library(survival)
str(veteran)
veteran$treat <- as.numeric(veteran$trt) - 1

dfcount <- df_counting(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat"
)

# Fit weighted Cox model with FH(0,0.5) weights
fit <- cox_rhogamma(
  dfcount = dfcount,
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 0.5),
  draws = 1000,
  verbose = TRUE
)

print(fit$cox_text_star)  # Resampling-based CI
print(fit$zlogrank_text)  # Weighted log-rank test

# Compare asymptotic and resampling CIs
print(fit$hr_ci_asy)
print(fit$hr_ci_star)
```

---

cox_rhogamma_resample *Resampling for Weighted Cox Model (rho, gamma)*

Description

Performs resampling to estimate uncertainty for the weighted Cox model (rho, gamma).

Usage

```
cox_rhogamma_resample(
  fit_rhogamma,
```

```
  i_bhat,
  K_wt_rg,
  i_zero,
  K_zero,
  G1.draws = NULL,
  G0.draws = NULL,
  draws = 100,
  seedstart = 8316951
)
```

Arguments

`fit_rhogamma`	List with fitted Cox model results.
`i_bhat`	Information at estimated beta.
`K_wt_rg`	Weights at estimated beta.
`i_zero`	Information at beta=0.
`K_zero`	Weights at beta=0.
`G1.draws`	Optional: pre-generated random draws for groups.
`G0.draws`	Optional: pre-generated random draws for groups.
`draws`	Number of resampling iterations (default: 100).
`seedstart`	Random seed for reproducibility (default: 8316951).

Details

The multipliers are standard normal, one row per subject (`G0.draws` for controls, `G1.draws` for treated, in the order of `fit_rhogamma$time`), drawn under `set.seed(seedstart)`; the caller's random-number stream is restored afterwards. With `w_hat` absent or all equal, the multipliers perturb the events only, as before, and these results are unchanged. When `w_hat` is not all equal, each resampled score is built from the per-subject score contributions described below: `score_star` is $-\sum_i G_i \psi_i(\hat{\beta})$ and `score_star_null` is $-\sum_i G_i \psi_i(0)$, so their variance over draws is the variance of the score used for `sig_bhat_asy` and the log-rank z-score (`K_wt_rg` and `K_zero` are then not used). `bhat_center_star`, `Z_bstar` and `sig_bhat_star` are formed from `score_star` as before. When `fit_rhogamma` comes from a `df_counting` result made with `weight_variance = "estimated"` (it then holds `ew_parts`), the contributions carry the estimation of the weights, $\psi_i + IF_i' B$ (section "Estimated weights" of `df_counting`).

Value

List with resampling results (score, beta, standard error, etc.).

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i\Big(D_i K(U_i)\{Z_i - \bar{z}(U_i,\beta)\} - e^{\beta Z_i}\sum_{s\le U_i} K(s)\{Z_i - \bar{z}(s,\beta)\}d\hat{\Lambda}(s,\beta)\Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s,\beta) = \sum_j a_j 1(U_j \ge s)e^{\beta Z_j}$, $\bar{z}(s,\beta) = \sum_j a_j 1(U_j \ge s)Z_j e^{\beta Z_j}/S_0(s,\beta)$ and $d\hat{\Lambda}(s,\beta) = \sum_j a_j D_j 1(U_j = s)/S_0(s,\beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2/i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

`cox_score_rhogamma` *Cox score with rho-gamma weights*

Description

Calculates the Cox score statistic for weighted log-rank tests.

Usage

```
cox_score_rhogamma(
  beta,
  time,
  delta,
  z,
  w_hat = rep(1, length(time)),
  wt_rg = rep(1, length(time)),
  score_only = TRUE
)
```

Arguments

`beta`	Log hazard ratio parameter.
`time`	Numeric vector of event times.
`delta`	Numeric vector of event indicators.
`z`	Numeric vector of group indicators.
`w_hat`	Numeric vector of weights.
`wt_rg`	Numeric vector of rho-gamma weights.
`score_only`	Logical; if TRUE, returns only the score.

Details

With `w_hat` absent or all equal, `sig2_score` and `sig2_beta_asy` are the formulas used before (`score_calculation`) and are unchanged. When `w_hat` is not all equal, `sig2_score` is the sum of squared per-subject score contributions $\sum_i \psi_i(\beta)^2$ described below and `sig2_beta_asy` is that sum over `i_bhat^2`. The score, `K_wt_rg` and `i_bhat` are unchanged. `time` is taken to be sorted, as `df_counting` returns it; the time weight of a tied time is the one at its first observation.

Value

Numeric value of the score or, with `score_only = FALSE`, a list with `score`, `sig2_score` (variance of the score at `beta`), `sig2_beta_asy` (`sig2_score / i_bhat^2`), `K_wt_rg` and `i_bhat` (the information).

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i\Big(D_i K(U_i)\{Z_i - \bar{z}(U_i,\beta)\} - e^{\beta Z_i}\sum_{s\le U_i} K(s)\{Z_i - \bar{z}(s,\beta)\}d\hat{\Lambda}(s,\beta)\Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s,\beta) = \sum_j a_j 1(U_j \ge s)e^{\beta Z_j}$, $\bar{z}(s,\beta) = \sum_j a_j 1(U_j \ge s)Z_j e^{\beta Z_j}/S_0(s,\beta)$ and $d\hat{\Lambda}(s,\beta) = \sum_j a_j D_j 1(U_j = s)/S_0(s,\beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2/i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

`create_baseline_table` *Create Baseline Characteristics Table by Treatment Arm*

Description

This function generates publication-ready baseline characteristic tables commonly used in clinical trials and observational studies. It calculates summary statistics, p-values, and standardized mean differences for continuous, categorical, and binary variables.

Usage

```
create_baseline_table(
  data,
  treat_var = "treat",
  vars_continuous = NULL,
  vars_categorical = NULL,
  vars_binary = NULL,
  var_labels = NULL,
  digits = 1,
  show_pvalue = TRUE,
  show_smd = TRUE,
  show_missing = TRUE
)
```

Arguments

data	Data frame containing baseline variables
treat_var	Name of treatment variable (default: "treat")
vars_continuous	Character vector of continuous variable names
vars_categorical	Character vector of categorical variable names
vars_binary	Character vector of binary variable names
var_labels	Named vector for variable labels (e.g., c(age = "Age (years)"))
digits	Number of decimal places for continuous variables (default: 1)
show_pvalue	Logical, whether to show p-values (default: TRUE)
show_smd	Logical, whether to show standardized mean differences (default: TRUE)
show_missing	Logical, whether to show missing data counts (default: TRUE)

Value

A gt table object (if gt package is available) or data frame

Examples

```
# Create sample data
set.seed(123)
n <- 500
sample_data <- data.frame(
  treat = rbinom(n, 1, 0.5),
  age = rnorm(n, mean = 55, sd = 10),
  stage = sample(c("I", "II", "III", "IV"), n, replace = TRUE),
  sex = rbinom(n, 1, 0.45),
  smoking = rbinom(n, 1, 0.3)
)

# Create table
```

```
table <- create_baseline_table(
  data = sample_data,
  treat_var = "treat",
  vars_continuous = "age",
  vars_categorical = "stage",
  vars_binary = c("sex", "smoking"),
  var_labels = c(
    age = "Age (years)",
    stage = "Disease Stage",
    sex = "Female",
    smoking = "Current Smoker"
  )
)
```

`cumulative_rmst_bands` *Cumulative RMST bands for survival curves*

Description

Calculates cumulative Restricted Mean Survival Time (RMST) and confidence bands for survival curves using resampling. Optionally plots the cumulative RMST curve, pointwise confidence intervals, and simultaneous confidence bands.

Usage

```
cumulative_rmst_bands(
  df,
  fit,
  tte.name,
  event.name,
  treat.name,
  weight.name = NULL,
  draws_sb = 1000,
  xlab = "months",
  ylim_pad = 0.5,
  rmst_max_legend = "left",
  rmst_max_cex = 0.7,
  plot = TRUE,
  mean_rule = c("trapezoid", "rectangle"),
  weight_variance = c("known", "estimated"),
  weight_model = NULL
)
```

Arguments

df	Data frame containing survival data.
fit	Survival fit object (output from KM_diff).
tte.name	Name of time-to-event variable in df.
event.name	Name of event indicator variable in df.
treat.name	Name of treatment group variable in df.
weight.name	Optional name of weights variable in df.
draws_sb	Number of resampling draws for simultaneous bands (default: 1000).
xlab	Label for x-axis (default: "months").
ylim_pad	Padding for y-axis limits (default: 0.5).
rmst_max_legend	Position for RMST legend (default: "left").
rmst_max_cex	Text size for RMST legend (default: 0.7).
plot	Logical; if TRUE, plot the results. Default is TRUE.
mean_rule	Character; how the survival difference is integrated over the time grid of fit (fit$at_points). "trapezoid" (default) averages the difference at consecutive grid points: $R(t_j) = \sum_{k \le j} \{d(t_{k-1}) + d(t_k)\}/2\,(t_k - t_{k-1})$. "rectangle" is the left-rectangle sum $R(t_j) = \sum_{k \le j} d(t_{k-1})(t_k - t_{k-1})$, the exact area under the step function. In both, $R = 0$ at the first grid point. "rectangle" equals the difference of survfit()'s restricted means (summary(fit, rmean = tau)) when the time grid holds every event time, which it does for the default KM_diff() grid; it does not when the fit was made with modify_tau = TRUE. The rule applies to the estimate and to both sets of resampled curves; nothing else changes.
weight_variance	How the weights are treated in standard errors, intervals, bands and tests. "known" (default): as fixed, known numbers. "estimated": as estimated by the logistic propensity model given in weight_model, whose estimation is carried into the inference. Estimates never change. See the section "Estimated weights".
weight_model	With weight_variance = "estimated", the object returned by ps_weights for the rows of df, in the same order; its weights must be the column weight.name. Ignored with "known".

Details

The pointwise variance of the cumulative RMST difference and the simultaneous band come from the resampled survival-difference curves of fit (fit$dhat_star) and of a second KM_diff call with seedstart = 99999. With weights that are not all equal, those resampled curves are built from the per-subject contributions of the robust variance (see resampling_survival), so the RMST variance and bands are the robust ones; with no weights or all weights equal they are unchanged. By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, weight_variance = "estimated" (with the same option in the KM_diff() call that made fit; a fit made with the other option is an error) carries their estimation into both sets of resampled curves, and so into the variance, the intervals and the band (section "Estimated weights"). With extreme weights (low effective sample size) the intervals can be too narrow under either option.

Value

A list with elements:

`at_points`	Time points used for RMST calculation
`rmst_time`	Cumulative RMST estimates
`sig2_rmst_time`	Variance of RMST estimates
`rmst_time_lower`	Pointwise lower confidence interval
`rmst_time_upper`	Pointwise upper confidence interval
`rmst_maxtau_ci`	RMST and CI at maximum time
`rmst_text`	Text summary for legend
`c_alpha_band`	Critical value for simultaneous band
`rmst_time_sb_lower`	Simultaneous band lower bound
`rmst_time_sb_upper`	Simultaneous band upper bound
`mean_rule`	The integration rule used
`weight_variance`	`"known"` or `"estimated"`: how the weights were treated

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t)\, dw_k, \qquad \Phi_i(t) = 1(i \in g)\, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{\hat{S}_1(t)\Phi_i^1(t) - \hat{S}_0(t)\Phi_i^0(t)\}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t) \sum_{i=1}^{n} G_i \Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j}\, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i \psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i \{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j \{c_j(t)/a_j\} dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j) dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)

- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With `"estimated"` the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (`df_counting()$ess`) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

`df_counting` *Weighted and Stratified Survival Analysis*

Description

Performs comprehensive weighted and/or stratified survival analysis, including Cox proportional hazards model, logrank/Fleming-Harrington tests, and calculation of risk/event sets, Kaplan-Meier curves, quantiles, and variance estimates.

Usage

```
df_counting(
  df,
  tte.name = "tte",
  event.name = "event",
  treat.name = "treat",
  weight.name = NULL,
  strata.name = NULL,
  arms = c("treat", "control"),
  time.zero = 0,
  tpoints.add = c(0),
  by.risk = 6,
  time.zero.label = 0,
  risk.add = NULL,
  get.cox = TRUE,
  cox.digits = 2,
  lr.digits = 2,
  cox.eps = 0.001,
  lr.eps = 0.001,
  verbose = FALSE,
```

```
  qprob = 0.5,
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 0),
  conf_level = 0.95,
  check.KM = TRUE,
  check.seKM = FALSE,
  draws = 0,
  seedstart = 8316951,
  stop.onerror = FALSE,
  censoring_allmarks = TRUE,
  weight_variance = c("known", "estimated"),
  weight_model = NULL
)
```

Arguments

`df`	Data frame containing survival data.
`tte.name`	Character; name of the time-to-event column in `df`.
`event.name`	Character; name of the event indicator column in `df` (1=event, 0=censored). Logical (TRUE = event) and 1/2 coding (2 = event, as `survival::Surv` reads it) are also accepted.
`treat.name`	Character; name of the treatment/group column in `df` (must be coded as 0=control, 1=experimental).
`weight.name`	Character or NULL; name of the weights column in `df`. If NULL, equal weights are used.
`strata.name`	Character or NULL; name of the strata column in `df` for stratified analysis.
`arms`	Character vector of length 2; group labels. Default: `c("treat","control")`.
`time.zero`	Numeric; time value to use as zero. Default: 0.
`tpoints.add`	Numeric vector; additional time points to include in calculations. Default: `c(0)`.
`by.risk`	Numeric; interval for risk set time points. Default: 6.
`time.zero.label`	Numeric; label for time zero in output. Default: 0.0.
`risk.add`	Numeric vector or NULL; additional specific risk points to include.
`get.cox`	Logical; whether to fit Cox proportional hazards model. Default: TRUE.
`cox.digits`	Integer; number of decimal places for Cox output formatting. Default: 2.
`lr.digits`	Integer; number of decimal places for logrank output formatting. Default: 2.
`cox.eps`	Numeric; threshold for Cox p-value formatting (values below shown as "<eps"). Default: 0.001.
`lr.eps`	Numeric; threshold for logrank p-value formatting. Default: 0.001.
`verbose`	Logical; whether to print warnings and diagnostic messages. Default: FALSE.
`qprob`	Numeric in (0,1); survival level for the KM quantile table: the quantile reported is the time at which survival drops to `qprob`, i.e. `quantile(survfit(...), probs = 1 - qprob)`. Default: 0.5 (median).

scheme	Character; weighting scheme for logrank/Fleming-Harrington test. Options: "fh", "schemper", "XO", "MB", "custom_time", "fh_exp1", "fh_exp2". Default: "fh".
scheme_params	List; parameters for the selected weighting scheme. Default: `list(rho = 0, gamma = 0)`. • For "fh": `rho` and `gamma` (Fleming-Harrington parameters) • For "MB": `mb_tstar` (cutoff time) • For "custom_time": `t.tau`, `w0.tau`, `w1.tau`
conf_level	Numeric in (0,1); confidence level for quantile intervals. Default: 0.95.
check.KM	Logical; whether to check KM curve validity against `survival::survfit`. Default: TRUE.
check.seKM	Logical; whether to check KM standard error estimates. Default: FALSE.
draws	Integer; number of draws for resampling-based variance estimation. Default: 0 (no resampling).
seedstart	Integer; random seed for reproducible resampling. Default: 8316951.
stop.onerror	Logical; whether to stop execution on errors (TRUE) or issue warnings (FALSE). Default: FALSE.
censoring_allmarks	Logical; if FALSE, removes events from censored time points. Default: TRUE.
weight_variance	How the weights are treated in standard errors, intervals, bands and tests. `"known"` (default): as fixed, known numbers. `"estimated"`: as estimated by the logistic propensity model given in `weight_model`, whose estimation is carried into the inference. Estimates never change. See the section "Estimated weights".
weight_model	With `weight_variance = "estimated"`, the object returned by `ps_weights` for the rows of `df`, in the same order; its weights must be the column `weight.name`. Ignored with `"known"`.

Details

This function implements a comprehensive survival analysis framework supporting:

- Weighted observations via `weight.name`
- Stratified analysis via `strata.name`
- Multiple weighting schemes for log-rank tests
- Resampling-based variance estimation
- Automatic validation against `survival` package results

The function performs time-fixing using `survival::aeqSurv` to handle tied event times.

Input checks: missing values (in any named column, including `strata.name`), non-finite or negative times, negative weights and an empty arm are errors; rows are never dropped silently. Zero weights are allowed: those rows contribute nothing, and the Cox model and log-rank test are fitted on the positive-weight rows. Zero-weight rows do not affect which variance formula runs either: whether the weights are "all equal" is decided on the positive weights, so a 0/1 weight column gives the

results of the positive-weight rows analysed without weights. A stratum in which one arm is empty contributes nothing to the stratified statistics (that arm's stratum curve is NA). On degenerate data (an arm without events, no events at all, one subject per arm) a result is still returned; quantities that cannot be computed are NA, with one warning that says why. For stratified analyses, stratum-specific estimates are computed and combined appropriately.

Value

A list with the following components:

cox_results List with the results of `survival::coxph()`: `cox_fit`, `hr`, `hr_ci`, `pval`, `score`, `cox_text` and `se`, the standard error of the log hazard ratio behind `hr_ci` and `pval` (the robust one of `coxph(robust = TRUE)`; NA when the model could not be fitted). `score` is the score test of the model: with no weights, or with all positive weights equal, `summary(cox_fit)$sctest`; with subject weights that are not all equal, the robust score test `summary(cox_fit)$robscore` (the model-based one reads the weights as counts).

logrank_results List with `chisq`, `pval`, `logrank_text` and `method`. With no weights, or with all positive weights equal, the test of `survival::survdiff()` (`method = "survdiff"`; `rho` from the scheme when it is `"fh"`). With subject weights that are not all equal `survdiff()` has no counterpart, and the result is the robust score test at 0 of the weighted Cox model (`coxph(weights = , init = 0, iter.max = 0, robust = TRUE, ties = "breslow")` on the positive-weight rows, strata included; `method = "coxph_robust_score"`): the `rho = 0` test whatever the scheme. Its `chisq` equals `z.score^2` (`z.score_stratified^2`) for Fleming-Harrington (0, 0) weights, so `check_results` agrees on weighted data.

z.score Standardized weighted log-rank test statistic (with subject weights that are not all equal, standardized by the score-residual variance; see the section on subject weights)

at_points Vector of all time points used in calculations

surv0, surv1 Kaplan-Meier survival estimates for control and treatment groups

sig2_surv0, sig2_surv1 Variance estimates for survival curves

survP Pooled survival estimates

survG Censoring distribution estimates

quantile_results Data frame with median survival and confidence intervals by group

lr, sig2_lr Weighted log-rank statistic and its variance (`get_score` holds the same as returned by `wlr_cumulative`; `zlogrank_text` is the one-sided p-value of `z.score`)

riskpoints0, riskpoints1 Risk set counts at specified time points

z.score_stratified Stratified z-score (if stratified analysis), with `lr_stratified` and `sig2_lr_stratified`

ess Data frame with one row per arm, in the order of `quantile_results` (`arms[1]`, the treated arm, then `arms[2]`): `group`; `n`, the rows of the arm with positive weight; `ess`, the effective sample size $(\sum w)^2/\sum w^2$ over those rows; and `ratio = ess / n`. Without weights, and with constant weights, `ess = n`. A ratio near 1 means the weights are even; a low ratio means that a few subjects carry most of the weight, and standard errors, intervals and tests can then be too optimistic, whether the weights are treated as known or as estimated. No warning or message depends on it.

weight_variance `"known"` or `"estimated"`: how the weights were treated.

ew_parts Only with `weight_variance = "estimated"`: list with `IF` and `dw` of `weight_model`, rows in the order of `time`; read by `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates`.

Variance of the Kaplan-Meier estimate

With no weights, or with all weights equal, the variance of a Kaplan-Meier estimate is Greenwood's formula, and these results are unchanged. When the weights are not all equal it is the robust (infinitesimal-jackknife) variance

$$\widehat{Var}[\hat{S}(t)] = \hat{S}(t)^2 \sum_i w_i^2 \psi_i(t)^2,$$

$$\psi_i(t) = \frac{D_i 1(U_i \le t)}{Y_w(U_i) - dN_w(U_i)} - \sum_{s \le \min(t, U_i)} \frac{dN_w(s)}{Y_w(s)\{Y_w(s) - dN_w(s)\}},$$

where U_i, D_i and w_i are subject i's time, event indicator and weight, $Y_w(s)$ is the weighted number at risk, $dN_w(s)$ the weighted number of events at s, the sum runs over event times, and a term is 0 where $Y_w(s) - dN_w(s) \le 10^{-10} Y_w(s)$. It equals the standard error of `survival::survfit(weights = w, robust = TRUE)` (on the time-fixed data), reduces to Greenwood's formula at equal weights, and does not depend on the scale of the weights. With `draws > 0` and unequal weights the variance is instead the variance over resampling draws of $\sum_i G_i w_i \psi_i(t)$ (standard-normal multipliers G_i), which estimates the same quantity; resampled curves for weighted data are built from the same per-subject contributions (see `resampling_survival`). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the standard error (section "Estimated weights" of `df_counting`). When the weights are extreme (low effective sample size) the standard error can be too small under either option.

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i \Big(D_i K(U_i)\{Z_i - \bar{z}(U_i, \beta)\} - e^{\beta Z_i} \sum_{s \le U_i} K(s)\{Z_i - \bar{z}(s, \beta)\} d\hat{\Lambda}(s, \beta) \Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s, \beta) = \sum_j a_j 1(U_j \ge s) e^{\beta Z_j}$, $\bar{z}(s, \beta) = \sum_j a_j 1(U_j \ge s) Z_j e^{\beta Z_j} / S_0(s, \beta)$ and $d\hat{\Lambda}(s, \beta) = \sum_j a_j D_j 1(U_j = s) / S_0(s, \beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2 / i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t) \, dw_k, \qquad \Phi_i(t) = 1(i \in g) \, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{ \hat{S}_1(t) \Phi_i^1(t) - \hat{S}_0(t) \Phi_i^0(t) \}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t) \sum_{i=1}^{n} G_i \Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j} \, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i \psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i \{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j \{c_j(t)/a_j\} dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j) dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)
- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With `"estimated"` the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (`df_counting()$ess`) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

Weighting Schemes

fh Fleming-Harrington: w(t) = S(t)^rho * (1-S(t))^gamma

MB Magirr-Burman: w(t) = 1/max(S(t), S(t*))

schemper Schemper: w(t) = S(t)/G(t) where G is the censoring distribution

XO Xu-O'Quigley: w(t) = S(t)/Y(t) where Y is risk set size

References

Fleming, T. R. and Harrington, D. P. (1991). Counting Processes and Survival Analysis. Wiley.

Magirr, D. and Burman, C. F. (2019). Modestly weighted logrank tests. Statistics in Medicine, 38(20), 3782-3790.

Leon, L. F., Cai, T. and Wei, L. J. (2009). Robust inferences for covariate effects on survival time with censored linear regression models. Statistics in Biosciences, 1, 50-64. doi:10.1007/s12561-00990025

See Also

coxph, survdiff, survfit cox_rhogamma for weighted Cox models KM_diff for Kaplan-Meier differences

Other survival_analysis: KM_diff(), cox_rhogamma(), ps_weights(), wt.rg.S()

Other weighted_tests: cox_rhogamma(), wt.rg.S()

Examples

```
# Basic survival analysis
library(survival)
str(veteran)
veteran$treat <- as.numeric(veteran$trt) - 1

result <- df_counting(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat",
  arms = c("Treatment", "Control")
)

# Print results
print(result$cox_results$cox_text)
print(result$zlogrank_text)

# Fleming-Harrington (0,1) weights (emphasizing late differences)
result_fh <- df_counting(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat",
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 1)
)

# Stratified analysis
result_strat <- df_counting(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat",
```

```
  strata.name = "celltype"
)
```

`extract_and_calc_weights` *Extract and calculate weights for multiple schemes*

Description

Extracts and calculates weights for multiple schemes and returns a combined data frame.

Usage

```
extract_and_calc_weights(atpoints, S.pool, weights_spec_list)
```

Arguments

`atpoints`	Numeric vector of time points.
`S.pool`	Numeric vector of pooled survival probabilities.
`weights_spec_list`	List of weighting scheme specifications.

Value

Data frame with weights for each scheme.

`extract_group_data` *Extract time, event, and weight data for a group*

Description

Extracts time, event, and weight vectors for a specified group.

Usage

```
extract_group_data(time, delta, wgt, z, group = 1)
```

Arguments

`time`	Numeric vector of times.
`delta`	Numeric vector of event indicators (1=event, 0=censored).
`wgt`	Numeric vector of weights.
`z`	Numeric vector of group indicators.
`group`	Value of group to extract (default 1).

Value

List with U (times), D (events), W (weights).

`find_cox_root` *Root-finding for Cox score function*

Description

Finds the root of the Cox model score equation for weighted log-rank statistics.

Usage

```
find_cox_root(time, delta, z, w_hat, wt_rg)
```

Arguments

`time`	Numeric vector of event times.
`delta`	Numeric vector of event indicators.
`z`	Numeric vector of group indicators.
`w_hat`	Numeric vector of weights.
`wt_rg`	Numeric vector of rho-gamma weights.

Value

List with root and additional information, or NA if not found.

`format_pval` *Format p-value for display*

Description

Formats a p-value for display, showing "<eps" for small values.

Usage

```
format_pval(pval, eps = 0.001, digits = 3)
```

Arguments

`pval`	Numeric p-value.
`eps`	Threshold for small p-values.
`digits`	Number of digits to display.

Value

Formatted p-value as character.

get_censoring_and_events *Get censoring and event times and their indices*

Description

Extracts censoring and event times and their indices for a group at specified time points.

Usage

```
get_censoring_and_events(time, delta, z, group, censoring_allmarks, at_points)
```

Arguments

time	Numeric vector of times.
delta	Numeric vector of event indicators.
z	Numeric vector of group indicators.
group	Value of group to extract.
censoring_allmarks	Logical; if FALSE, remove events from censored.
at_points	Numeric vector of time points.

Value

List with cens (censored times), ev (event times), idx_cens, idx_ev, idx_ev_full.

get_dfcounting *Get df_counting results*

Description

Wrapper for df_counting, safely executes and returns results for survival analysis.

Usage

```
get_dfcounting(...)
```

Arguments

...	Arguments passed to df_counting.

Value

Result from df_counting or NULL if error.

See Also

df_counting

`get_event_risk_matrices` *Event and Risk Matrices for Survival Analysis*

Description

Constructs matrices indicating event and risk status for each subject at specified time points.

Usage

```
get_event_risk_matrices(U, at.points)
```

Arguments

`U`	Vector of observed times (e.g., time-to-event).
`at.points`	Vector of time points at which to evaluate events and risk.

Value

A list with event and risk matrices.

`get_riskpoints` *Get risk set counts at specified risk points*

Description

Returns risk set counts at specified risk points.

Usage

```
get_riskpoints(ybar, risk_points, at_points)
```

Arguments

`ybar`	Numeric vector of risk set counts.
`risk_points`	Numeric vector of risk points.
`at_points`	Numeric vector of time points.

Value

Numeric vector of risk set counts at risk points.

get_validated_weights *Get validated weights for a data frame*

Description

Validates and returns weights for a data frame according to specified schemes.

Usage

```
get_validated_weights(
  df_weights,
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 0),
  details = FALSE
)
```

Arguments

df_weights	Data frame containing weights and related data.
scheme	Character string specifying the weighting scheme.
scheme_params	List of parameters for the scheme.
details	Logical; if TRUE, returns detailed output.

Value

Numeric vector or list of validated weights.

get_weights *Get weights for a weighting scheme*

Description

Calculates weights for a specified scheme at given time points.

Usage

```
get_weights(scheme, scheme_params, S.pool, tpoints)
```

Arguments

scheme	Character string specifying the weighting scheme.
scheme_params	List of parameters for the scheme.
S.pool	Numeric vector of pooled survival probabilities.
tpoints	Numeric vector of time points.

Value

Numeric vector of weights.

`KM_diff` *Kaplan-Meier Difference Between Groups*

Description

Calculates the difference in Kaplan-Meier curves between two groups, with confidence intervals and optional resampling-based simultaneous confidence bands.

Usage

```
KM_diff(
  df,
  tte.name = "tte",
  event.name = "event",
  treat.name = "treat",
  weight.name = NULL,
  at_points = sort(df[[tte.name]]),
  alpha = 0.05,
  seedstart = 8316951,
  draws = 0,
  risk.points = NULL,
  draws.band = 0,
  tau.seq = 0.25,
  qtau = 0.025,
  show_resamples = FALSE,
  modify_tau = FALSE,
  weight_variance = c("known", "estimated"),
  weight_model = NULL
)
```

Arguments

`df`	Data frame containing survival data.
`tte.name`	Character; name of time-to-event variable in `df`.
`event.name`	Character; name of event indicator variable in `df` (1=event, 0=censored). Logical (TRUE = event) and 1/2 coding (2 = event, as `survival::Surv` reads it) are also accepted.
`treat.name`	Character; name of treatment group variable in `df` (0=control, 1=treatment).
`weight.name`	Character or NULL; name of weights variable in `df`.
`at_points`	Numeric vector; time points at which the curves are returned: finite and non-decreasing (duplicates are allowed and returned as given). The Kaplan-Meier value at each point is right-continuous, whether or not the points include the event times. Default: the sorted observed times.

alpha	Numeric; significance level for the pointwise intervals and the simultaneous band (the band's critical value is the 1 - alpha quantile of the resampled suprema). Default: 0.05.
seedstart	Integer; random seed for reproducibility. Default: 8316951.
draws	Integer; number of draws for pointwise variance estimation. Default: 0.
risk.points	Numeric vector; time points for risk table display.
draws.band	Integer; number of draws for simultaneous confidence bands. Default: 0.
tau.seq	Numeric; step size for tau sequence when modify_tau=TRUE. Default: 0.25.
qtau	Numeric; quantile for tau range restriction. Default: 0.025.
show_resamples	Logical; whether to plot the first 20 standardised resampled difference curves (only when draws.band > 0). Default: FALSE.
modify_tau	Logical; whether to restrict time range for simultaneous bands. Default: FALSE.
weight_variance	How the weights are treated in standard errors, intervals, bands and tests. "known" (default): as fixed, known numbers. "estimated": as estimated by the logistic propensity model given in weight_model, whose estimation is carried into the inference. Estimates never change. See the section "Estimated weights".
weight_model	With weight_variance = "estimated", the object returned by ps_weights for the rows of df, in the same order; its weights must be the column weight.name. Ignored with "known".

Details

This function computes the difference in Kaplan-Meier survival curves, delta(t) = S_1(t) - S_0(t),

along with variance estimates and confidence intervals.

When draws.band > 0, simultaneous confidence bands are constructed using the supremum distribution of the standardized difference process. These bands maintain the specified coverage probability across all time points simultaneously.

The pointwise variance of each arm's curve is Greenwood's formula for unweighted data (or all weights equal) and the robust variance for weights that are not all equal (see the section below), or resampling-based when draws > 0 and the weights are not all equal.

For weights that are not all equal, the resampled curves (surv0_star, surv1_star, and from them dhat_star, Zdhat_star, the simultaneous band and cumulative_rmst_bands) are built from the same per-subject contributions as the robust variance (resampling_survival), so the variance across draws matches the analytic variance. With no weights or all weights equal they are unchanged.

Value

A list containing:

at_points Time points used in calculations

surv0, surv1 Survival estimates for control and treatment groups

sig2_surv0, sig2_surv1 Variance estimates for survival curves

dhat Survival difference (S1 - S0) at each time point

sig2_dhat Variance of survival difference

lower, upper Pointwise confidence limits (1 - alpha/2)

sb_lower, sb_upper Simultaneous band limits (if draws.band > 0)

c_alpha_band Critical value for simultaneous band (if draws.band > 0)

dhat_star Matrix of resampled differences (if draws.band > 0)

Zdhat_star Standardized resampled differences (if draws.band > 0)

weight_variance `"known"` or `"estimated"`: how the weights were treated

Variance of the Kaplan-Meier estimate

With no weights, or with all weights equal, the variance of a Kaplan-Meier estimate is Greenwood's formula, and these results are unchanged. When the weights are not all equal it is the robust (infinitesimal-jackknife) variance

$$\widehat{Var}[\hat{S}(t)] = \hat{S}(t)^2 \sum_i w_i^2 \psi_i(t)^2,$$

$$\psi_i(t) = \frac{D_i 1(U_i \le t)}{Y_w(U_i) - dN_w(U_i)} - \sum_{s \le \min(t, U_i)} \frac{dN_w(s)}{Y_w(s)\{Y_w(s) - dN_w(s)\}},$$

where U_i, D_i and w_i are subject i's time, event indicator and weight, $Y_w(s)$ is the weighted number at risk, $dN_w(s)$ the weighted number of events at s, the sum runs over event times, and a term is 0 where $Y_w(s) - dN_w(s) \le 10^{-10} Y_w(s)$. It equals the standard error of `survival::survfit(weights = w, robust = TRUE)` (on the time-fixed data), reduces to Greenwood's formula at equal weights, and does not depend on the scale of the weights. With `draws > 0` and unequal weights the variance is instead the variance over resampling draws of $\sum_i G_i w_i \psi_i(t)$ (standard-normal multipliers G_i), which estimates the same quantity; resampled curves for weighted data are built from the same per-subject contributions (see `resampling_survival`). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the standard error (section "Estimated weights" of `df_counting`). When the weights are extreme (low effective sample size) the standard error can be too small under either option.

Confidence Intervals vs Bands

- Pointwise CIs (`lower`, `upper`): Cover the true difference at each time point with probability 1-alpha
- Simultaneous bands (`sb_lower`, `sb_upper`): Cover the entire difference curve with probability 1-alpha

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i\,(A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i\,(A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t)\, dw_k, \qquad \Phi_i(t) = 1(i \in g)\, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{\hat{S}_1(t)\Phi_i^1(t) - \hat{S}_0(t)\Phi_i^0(t)\}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t) \sum_{i=1}^{n} G_i \Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j}\, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i \psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i \{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j \{c_j(t)/a_j\} dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for

tzero, and the covariance uses both adjusted contributions. The outputs of df_counting() that come from the survival package (logrank_results, cox_results$score, $se, $hr_ci, $pval) carry the same adjustment, applied to coxph()'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j) dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with ps_weights, put its weights in a column of df, and pass weight_variance = "estimated" and weight_model to df_counting(), KM_diff, cumulative_rmst_bands or plotKM.band_subgroups. cox_rhogamma, wlr_cumulative and wlr_dhat_estimates take the option from the df_counting() result they are given (its elements weight_variance and ew_parts). The rows of df, the treatment column and the weight column must be those of weight_model; anything else is an error.

Scope and limits of this version.

- A logistic glm for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use "known".
- A subgroup analysis that reuses weights from a model fitted on the full sample must use "known": the model's rows are then not the rows analysed. (The subgroup curves of plotKM.band_subgroups() are point estimates and are not affected.)
- draws > 0 (the resampling-based pointwise variance) is not available with "estimated"; draws.band and the resampling of cox_rhogamma() are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With "estimated" the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (df_counting()$ess) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

Note

Treatment must be coded as 0=control, 1=experimental, and both arms must have observations. Missing values, non-finite or negative times and negative weights are errors (rows are never dropped silently); zero weights are allowed and those rows contribute nothing: they do not affect which variance formula runs either (whether the weights are all equal is decided on the positive weights).

With weight_variance = "estimated" the variance of the difference is not the sum of the two arm variances, and draws must be 0.

See Also

df_counting for full survival analysis plotKM.band_subgroups for visualization cumulative_rmst_bands for RMST analysis

Other survival_analysis: cox_rhogamma(), df_counting(), ps_weights(), wt.rg.S()

Examples

```
library(survival)
str(veteran)
veteran$treat <- as.numeric(veteran$trt) - 1

# Basic KM difference
result <- KM_diff(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat"
)

# Plot the difference
plot(result$at_points, result$dhat, type = "s",
     xlab = "Time", ylab = "Survival Difference")

# With simultaneous confidence bands
result_band <- KM_diff(
  df = veteran,
  tte.name = "time",
  event.name = "status",
  treat.name = "treat",
  draws.band = 1000,
  modify_tau = TRUE,
  show_resamples = TRUE
)
```

KM_estimates *Kaplan-Meier Survival Estimates and Variance*

Description

Computes Kaplan-Meier survival estimates and their variances given risk and event counts. Survival is returned in [0, 1]; when everything still at risk fails at a time point, survival is exactly 0 from there on and so is its variance.

Usage

```
KM_estimates(ybar, nbar, sig2w_multiplier = NULL)
```

Arguments

ybar	Vector of risk set sizes at each time point.
nbar	Vector of cumulative event counts at each time point.
sig2w_multiplier	Optional vector for variance calculation. If NULL, calculated internally.

Details

The result is the Kaplan-Meier estimate only when the time points behind `ybar` and `nbar` include every event time of the group (see `KM_diff`, which arranges this for any `at_points`).

Value

List with survival estimates and variances.

`KM_plot_2sample_weighted_counting` *Plot Weighted Kaplan-Meier Curves for Two Samples (Counting Process Format)*

Description

Plots Kaplan-Meier survival curves for two groups using precomputed risk/event counts and survival estimates. Optionally displays confidence intervals, risk tables, median survival annotations, and statistical test results.

Usage

```
KM_plot_2sample_weighted_counting(
  dfcount,
  show.cox = TRUE,
  cox.cex = 0.725,
  show.logrank = FALSE,
  logrank.cex = 0.725,
  cox.eps = 0.001,
  lr.eps = 0.001,
  show_arm_legend = TRUE,
  arms = c("treat", "control"),
  put.legend.arms = "left",
  stop.onerror = TRUE,
  check.KM = TRUE,
  put.legend.cox = "topright",
  put.legend.lr = "topleft",
  lr.digits = 2,
  cox.digits = 2,
  tpoints.add = c(0),
  by.risk = NULL,
  Xlab = "time",
  Ylab = "proportion surviving",
  col.0 = "black",
  col.1 = "blue",
  show.med = TRUE,
  med.digits = 2,
  med.font = 4,
```

```
  conf.int = FALSE,
  conf_level = 0.95,
  choose_ylim = FALSE,
  arm.cex = 0.7,
  quant = 0.5,
  med.cex = 0.725,
  ymed.offset = 0.1,
  xmed.fraction = 0.8,
  risk.cex = 0.725,
  ltys = c(1, 1),
  lwds = c(1, 1),
  censor.mark.all = TRUE,
  censor.cex = 0.5,
  show.ticks = TRUE,
  risk.set = TRUE,
  ymin = 0,
  ymax = 1,
  ymin.del = 0.035,
  ymin2 = NULL,
  risk_offset = 0.125,
  risk_delta = 0.05,
  y.risk0 = NULL,
  show.Y.axis = TRUE,
  cex_Yaxis = 1,
  y.risk1 = NULL,
  add.segment = FALSE,
  risk.add = NULL,
  xmin = 0,
  xmax = NULL,
  x.truncate = NULL,
  time.zero = 0,
  prob.points = NULL
)
```

Arguments

dfcount	List containing precomputed survival data.
show.cox	Logical; show Cox model results.
cox.cex	Numeric; text size for Cox annotation.
show.logrank	Logical; show log-rank test results.
logrank.cex	Numeric; text size for log-rank annotation.
cox.eps	Numeric; small values for Cox calculations.
lr.eps	Numeric; small values for log-rank calculations.
show_arm_legend	Logical; show arm legend.
arms	Character vector of arm labels.

put.legend.arms	Character; legend positions.
stop.onerror	Logical; stop on KM curve errors.
check.KM	Logical; check KM curve validity.
put.legend.cox	Character; legend positions.
put.legend.lr	Character; legend positions.
lr.digits	Integer; digits for test results.
cox.digits	Integer; digits for test results.
tpoints.add	Numeric vector; additional time points to include (default: c(0)).
by.risk	Numeric; interval for risk table time points.
Xlab	Character; axis labels.
Ylab	Character; axis labels.
col.0	Color for control curve.
col.1	Color for treatment curve.
show.med	Logical; annotate median survival.
med.digits	Median annotation settings.
med.font	Median annotation settings.
conf.int	Logical; plot confidence intervals.
conf_level	Numeric; confidence level for intervals.
choose_ylim	Logical; auto-select y-axis limits.
arm.cex	Numeric; text size for arm legend.
quant	Numeric; quantile for annotation.
med.cex	Median annotation settings.
ymed.offset	Median annotation settings.
xmed.fraction	Median annotation settings.
risk.cex	Numeric; text size for risk table.
ltys	Integer; line types for curves.
lwds	Integer; line widths for curves.
censor.mark.all	Logical; mark all censored times.
censor.cex	Numeric; size of censor marks.
show.ticks	Logical; show axis ticks.
risk.set	Logical; display risk table.
ymin	Additional graphical and calculation parameters.
ymax	Additional graphical and calculation parameters.
ymin.del	Additional graphical and calculation parameters.
ymin2	Additional graphical and calculation parameters.
risk_offset	Additional graphical and calculation parameters.

risk_delta	Additional graphical and calculation parameters.
y.risk0	Additional graphical and calculation parameters.
show.Y.axis	Additional graphical and calculation parameters.
cex_Yaxis	Additional graphical and calculation parameters.
y.risk1	Additional graphical and calculation parameters.
add.segment	Additional graphical and calculation parameters.
risk.add	Additional graphical and calculation parameters.
xmin	Additional graphical and calculation parameters.
xmax	Additional graphical and calculation parameters.
x.truncate	Additional graphical and calculation parameters.
time.zero	Numeric; time at which the curves start at S = 1 (default: 0).
prob.points	Numeric vector; probability points for additional annotations (default: NULL).

Value

Invisibly returns NULL. Used for plotting side effects.

`km_quantile` *Kaplan-Meier quantile and confidence interval*

Description

Calculates the quantile and confidence interval for a Kaplan-Meier curve. The quantile is the time at which survival drops to `qprob` (see `kmq_calculations`); the limits are the same quantity read from the log-transform pointwise limits of the curve (upper limit capped at 1, NA where S = 0), as `survival::quantile.survfit` does with `conf.type = "log"`.

Usage

```
km_quantile(
  time_points,
  survival_probs,
  se_probs = NULL,
  qprob = 0.5,
  type = c("midpoint", "min"),
  conf_level = 0.95
)
```

Arguments

time_points	Vector of time points.
survival_probs	Vector of survival probabilities.
se_probs	Standard errors of survival probabilities.
qprob	Survival level (default 0.5): the quantile is the time at which S drops to `qprob`.
type	Calculation type (midpoint or min).
conf_level	Confidence level (default 0.95).

Value

List with quantile and confidence interval.

`km_quantile_table` *Table of KM quantiles for two groups*

Description

Returns a data frame of quantiles and confidence intervals for two groups.

Usage

```
km_quantile_table(
  time_points,
  surv0,
  se0,
  surv1,
  se1,
  arms = c("treat", "control"),
  qprob = 0.5,
  type = c("midpoint", "min"),
  conf_level = 0.95
)
```

Arguments

`time_points`	Vector of time points.
`surv0`	Survival probabilities for group 0.
`se0`	Standard errors for group 0.
`surv1`	Survival probabilities for group 1.
`se1`	Standard errors for group 1.
`arms`	Group labels.
`qprob`	Quantile probability.
`type`	Calculation type.
`conf_level`	Confidence level.

Value

Data frame of quantiles and CIs for each group.

kmq_calculations *Kaplan-Meier quantile calculation*

Description

Calculates the quantile time for a Kaplan-Meier curve: the time at which survival drops to qprob (qprob = 0.5 is the median; qprob = 0.25 is the time at which 25\

Usage

```
kmq_calculations(time_points, survival_probs, qprob = 0.5, type = "midpoint")
```

Arguments

time_points	Vector of time points (non-decreasing).
survival_probs	Vector of survival probabilities at time_points.
qprob	Survival level (default 0.5): the quantile is the time at which S drops to qprob.
type	Calculation type (midpoint or min).

Details

With type = "midpoint" the rule of survival::quantile.survfit is used: the first time at which S(t) <= qprob (comparisons within sqrt(.Machine$double.eps)); when S(t) equals qprob on an interval, the midpoint of that interval; when that plateau runs to the last time point, the midpoint between its start and the last time point. NA when the curve never reaches qprob. With type = "min", the first time at which S(t) <= qprob. The curve is taken to start at S = 1 at time 0 and to end at the last element of time_points; values may be NA (they are skipped).

Value

Estimated quantile time.

N_rhogamma *Weighted event count for Cox model*

Description

Calculates the weighted event count for the Cox model with rho-gamma weights.

Usage

```
N_rhogamma(x, error, delta, weight = 1)
```

Arguments

x	Numeric vector of time points.
error	Numeric vector of error terms.
delta	Numeric vector of event indicators.
weight	Numeric vector of weights (default: 1).

Value

Numeric value of weighted event count.

---

plot_km *Plot Kaplan-Meier curves*

Description

Plots Kaplan-Meier survival curves for groups in the data.

Usage

```
plot_km(df, tte.name, event.name, treat.name, weights = NULL, ...)
```

Arguments

df	Data frame containing survival data.
tte.name	Name of time-to-event column.
event.name	Name of event indicator column.
treat.name	Name of treatment/group column.
weights	Optional; name of weights column.
...	Additional arguments passed to plot().

Value

Kaplan-Meier fit object (invisible).

plot_km_confint_polygon

Plot confidence interval polygon for KM curve

Description

Plots a shaded polygon representing the confidence interval for a Kaplan-Meier survival curve. Limits use the log transform; the upper limit is capped at 1 and, where survival is 0, the band closes onto the curve.

Usage

```
plot_km_confint_polygon(x, surv, se, conf_level, col)
```

Arguments

x	Numeric vector of time points.
surv	Numeric vector of survival probabilities.
se	Numeric vector of standard errors of survival probabilities.
conf_level	Numeric; confidence level for interval (default 0.95).
col	Color for the polygon.

Value

Invisibly returns NULL. Used for plotting side effects.

plot_km_curves_counting

Plot KM curves for two groups with optional confidence intervals and censoring marks

Description

Plots Kaplan-Meier survival curves for two groups, with options for confidence intervals and censoring marks. Each curve is drawn as the right-continuous step function from `time.zero` (where S = 1) to the group's last observed time; the confidence band (log transform, upper limit capped at 1, closing onto the curve where S = 0) is a step band over the same range. The two bands are filled with semi-transparent grey (control) and light blue (treated), so each stays visible where they overlap.

Usage

```
plot_km_curves_counting(
  at_points,
  S0.KM,
  idx0,
  idv0,
  S1.KM,
  idx1,
  idv1,
  col.0,
  col.1,
  ltys,
  lwds,
  Xlab,
  Ylab,
  ylim,
  xlim,
  show.ticks = FALSE,
  cens0 = NULL,
  risk.points,
  risk.points.label,
  cens1 = NULL,
  se0.KM = NULL,
  se1.KM = NULL,
  conf.int = FALSE,
  conf_level = 0.95,
  censor.cex = 1,
  time.zero = 0,
  tpoints.add = c(0),
  ...
)
```

Arguments

at_points	Numeric vector of time points for plotting.
S0.KM	Numeric vector of survival probabilities for group 0.
idx0	Indices for censoring in group 0.
idv0	Indices for events in group 0.
S1.KM	Numeric vector of survival probabilities for group 1.
idx1	Indices for censoring in group 1.
idv1	Indices for events in group 1.
col.0	Color for group 0.
col.1	Color for group 1.
ltys	Line types for groups.
lwds	Line widths for groups.

Xlab	X-axis label.
Ylab	Y-axis label.
ylim	Y-axis limits.
xlim	X-axis limits.
show.ticks	Logical; show censoring marks (default FALSE).
cens0	Numeric vector of censoring times for group 0.
risk.points	Numeric vector of risk time points.
risk.points.label	Character vector of labels for risk time points.
cens1	Numeric vector of censoring times for group 1.
se0.KM	Numeric vector of standard errors for group 0.
se1.KM	Numeric vector of standard errors for group 1.
conf.int	Logical; show confidence intervals (default FALSE).
conf_level	Confidence level (default 0.95).
censor.cex	Numeric; censoring mark size (default 1.0).
time.zero	Numeric; time at which the curves start at S = 1 (default 0).
tpoints.add	Numeric vector; additional time points to include (default c(0)).
...	Additional arguments to plot.

Value

Invisibly returns NULL. Used for plotting side effects.

`plot_weight_schemes` *Plot weight schemes for survival analysis*

Description

Plots the weights for different schemes over time using ggplot2.

Usage

```
plot_weight_schemes(
  dfcount,
  tte.name = "time_months",
  event.name = "status",
  treat.name = "treat",
  arms = c("treat", "control"),
 weights_spec_list = list(`MB(12)` = list(scheme = "MB", mb_tstar = 12), `MB(6)` =
  list(scheme = "MB", mb_tstar = 6), `FH(0,1)` = list(scheme = "fh", rho = 0, gamma =
  1), `FH(0.5,0.5)` = list(scheme = "fh", rho = 0.5, gamma = 0.5), custom_time =
   list(scheme = "custom_time", t.tau = 25, w0.tau = 1, w1.tau = 1.5), FHexp2 =
```

```
    list(scheme = "fh_exp2")),
  custom_colors = c(`FH(0,1)` = "grey", FHexp2 = "black", `MB(12)` = "#1b9e77", `MB(6)` =
    "#d95f02", `FH(0.5,0.5)` = "#7570b3", custom_time = "green"),
  custom_sizes = c(`FH(0,1)` = 2, FHexp2 = 1, `MB(12)` = 1, `MB(6)` = 1, `FH(0.5,0.5)` =
    1, custom_time = 1),
  transform_fh = FALSE,
  rescheme_fhexp2 = TRUE
)
```

Arguments

dfcount	Data frame containing counting process results, including time points and survival probabilities.
tte.name	Name of the time-to-event variable (default: 'time_months').
event.name	Name of the event indicator variable (default: 'status').
treat.name	Name of the treatment group variable (default: 'treat').
arms	Character vector of group names (default: c('treat', 'control')).
weights_spec_list	List of weighting scheme specifications.
custom_colors	Named character vector of colors for each scheme.
custom_sizes	Named numeric vector of line sizes for each scheme.
transform_fh	Logical; whether to transform FH weights (default: FALSE).
rescheme_fhexp2	Logical; whether to rescheme FHexp2 and custom_time (default: TRUE).

Details

This function visualizes the weights used in various survival analysis schemes (e.g., FH, MB, custom) using ggplot2. Facets and colors are customizable.

Value

A ggplot object showing the weight schemes over time.

---

plot_weighted_km *Plot weighted Kaplan-Meier curves*

Description

Plots weighted Kaplan-Meier curves using a custom function.

Usage

```
plot_weighted_km(dfcount, ...)
```

Arguments

dfcount	Result object from df_counting.
...	Additional arguments passed to KM_plot_2sample_weighted_counting.

Value

None. Plots the curves.

---

plotKM.band_subgroups *Plot Kaplan-Meier Survival Difference Curves with Subgroups and Confidence Bands*

Description

Plots the difference in Kaplan-Meier survival curves between two groups, optionally including simultaneous confidence bands and subgroup curves. Also displays risk tables for the overall population and specified subgroups.

Usage

```
plotKM.band_subgroups(
  df,
  tte.name,
  event.name,
  treat.name,
  weight.name = NULL,
  sg_labels = NULL,
  ltype = "s",
  lty = 1,
  draws = 0,
  lwd = 2,
  sg_colors = NULL,
  color = "lightgrey",
  ymax.pad = 0.01,
  ymin.pad = -0.01,
  taus = c(-Inf, Inf),
  yseq_length = 5,
  cex_Yaxis = 0.8,
  risk_cex = 0.8,
  by.risk = 6,
  risk.add = NULL,
  xmax = NULL,
  ymin = NULL,
  ymax = NULL,
  ymin.del = 0.035,
  y.risk1 = NULL,
```

```
  y.risk2 = NULL,
  ymin2 = NULL,
  risk_offset = NULL,
  risk.pad = 0.01,
  risk_delta = 0.0275,
  tau_add = NULL,
  time.zero.pad = 0,
  time.zero.label = 0,
  xlabel = NULL,
  ylabel = NULL,
  Maxtau = NULL,
  seedstart = 8316951,
  ylim = NULL,
  draws.band = 1000,
  qtau = 0.025,
  show_resamples = FALSE,
  modify_tau = FALSE,
  weight_variance = c("known", "estimated"),
  weight_model = NULL
)
```

Arguments

df	Data frame containing survival data.
tte.name	Name of the time-to-event column.
event.name	Name of the event indicator column (0/1; logical and 1/2 coding as in `survival::Surv` are also accepted).
treat.name	Name of the treatment group column (0/1).
weight.name	Optional name of the weights column.
sg_labels	Character vector of subgroup definitions (as logical expressions).
ltype	Line type for curves (default: "s").
lty	Line style for curves (default: 1).
draws	Number of draws for the resampling-based pointwise variance, passed to `KM_diff` (default: 0). With the default the pointwise standard error, the pointwise limits and the standardisation of the simultaneous band use the analytic variance (Greenwood's formula; the robust variance when the weights are not all equal). `draws > 0` only has an effect when the weights are not all equal: the variance is then the variance over that many resampling draws. The default was 20 before version 0.2.0.
lwd	Line width for curves (default: 2).
sg_colors	Colors for subgroup curves.
color	Color for confidence band polygon (default: "lightgrey").
ymax.pad	Padding for y-axis limits.
ymin.pad	Padding for y-axis limits.
taus	Vector for time truncation (default: c(-Inf, Inf)).

yseq_length	Number of y-axis ticks (default: 5).
cex_Yaxis	Text size for axis.
risk_cex	Text size for risk table.
by.risk	Interval for risk table time points (default: 6).
risk.add	Additional time points for risk table.
xmax	Additional graphical and calculation parameters.
ymin	Additional graphical and calculation parameters.
ymax	Additional graphical and calculation parameters.
ymin.del	Additional graphical and calculation parameters.
y.risk1	Additional graphical and calculation parameters.
y.risk2	Additional graphical and calculation parameters.
ymin2	Additional graphical and calculation parameters.
risk_offset	Additional graphical and calculation parameters.
risk.pad	Additional graphical and calculation parameters.
risk_delta	Additional graphical and calculation parameters.
tau_add	Additional graphical and calculation parameters.
time.zero.pad	Additional graphical and calculation parameters.
time.zero.label	Additional graphical and calculation parameters.
xlabel	Additional graphical and calculation parameters.
ylabel	Additional graphical and calculation parameters.
Maxtau	Upper end of the time range. By default the smaller of the two arms' last event times; it must be supplied when an arm has no events.
seedstart	Additional graphical and calculation parameters.
ylim	Additional graphical and calculation parameters.
draws.band	Number of draws for the simultaneous confidence band (default: 1000); `0` draws the pointwise limits only. The band's critical value is a quantile of the suprema of that many resampled curves. The default was 20 before version 0.2.0, which left the critical value to 20 draws; a call that supplies `draws.band` is unaffected.
qtau	Quantile for time range in simultaneous bands (default: 0.025).
show_resamples	Logical; whether to plot resampled curves (default: FALSE).
modify_tau	Logical; restrict time range for bands.
weight_variance	How the weights are treated in standard errors, intervals, bands and tests. `"known"` (default): as fixed, known numbers. `"estimated"`: as estimated by the logistic propensity model given in `weight_model`, whose estimation is carried into the inference. Estimates never change. See the section "Estimated weights".
weight_model	With `weight_variance = "estimated"`, the object returned by `ps_weights` for the rows of `df`, in the same order; its weights must be the column `weight.name`. Ignored with `"known"`.

Details

The pointwise standard error of the difference, the pointwise limits and the standardisation of the simultaneous band come from KM_diff with the draws and draws.band given here. Since version 0.2.0 the default is draws = 0: the pointwise variance is the analytic one (Greenwood's formula, or the robust variance when the weights are not all equal). Before, the default draws = 20 made the pointwise standard error of a weighted plot, and with it the band, a 20-draw resampling variance. Plots without weights, or with all weights equal, are numerically unaffected by draws. Since version 0.2.0 the default of draws.band is 1000 (it was 20), so the critical value of the simultaneous band of a call that does not pass draws.band rests on 1000 resampled curves. Zero-weight rows contribute nothing and do not affect which variance formula runs.

weight_variance and weight_model are passed to the overall KM_diff() call (pointwise limits and band). The subgroup curves are point estimates and are computed as before.

Value

(Invisible) list containing KM_diff results, time points, subgroup curves, risk tables, and confidence intervals.

Estimated weights

Propensity-score weights are estimated from the data. weight_variance says how standard errors, intervals, bands and tests treat them.

"known" **(default)** The weights are fixed, known numbers: the formulas of the two sections above. weight_model is ignored.

"estimated" The weights were estimated by a logistic regression for the treatment, given as weight_model (the object returned by ps_weights), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t) \, dw_k, \qquad \Phi_i(t) = 1(i \in g) \, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{\hat{S}_1(t)\Phi_i^1(t) - \hat{S}_0(t)\Phi_i^0(t)\}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t)\sum_{i=1}^{n} G_i\Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j}\, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i\psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i\{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j\{c_j(t)/a_j\}dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j)dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)
- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.

- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With `"estimated"` the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (`df_counting()$ess`) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

`ps_weights` *Propensity-score weights with the pieces needed to carry their estimation into the inference*

Description

Fits (or takes) a logistic regression for a binary treatment and returns inverse-probability weights together with the influence of the fitted coefficients and the derivative of each weight in those coefficients. Passed as `weight_model` to `df_counting()`, `KM_diff()`, `cumulative_rmst_bands()` or `plotKM.band_subgroups()` with `weight_variance = "estimated"`, the object lets standard errors, intervals, bands and tests account for the fact that the weights were estimated (see the section "Estimated weights" of `df_counting()`).

Usage

```
ps_weights(
  formula = NULL,
  data = NULL,
  type = c("ipw", "stabilized"),
  model = NULL
)

## S3 method for class 'ps_weights'
print(x, digits = 4, ...)
```

Arguments

`formula`	A formula for the treatment model, treatment (coded 0/1) on the left-hand side. Used with `data`; fitted by `stats::glm(formula, family = stats::binomial(), data = data)` with the default control, so that weights computed by hand from the same call agree.
`data`	Data frame holding the variables of `formula`: exactly the rows that will be analysed.

type	"ipw" (default): 1 / e for treated subjects and 1 / (1 - e) for controls, with e the fitted probability of treatment. "stabilized": those weights multiplied by the proportion treated (treated subjects) and the proportion not treated (controls).
model	An existing logistic glm fit, instead of formula and data. Give either formula and data, or model, not both.
x	A "ps_weights" object.
digits	Number of significant digits to print.
...	Not used.

Details

With x_i the row of the model matrix of subject i, A_i the treatment and e_i the fitted probability,

$$IF_i = \Big\{ \sum_j x_j x_j' e_j (1 - e_j) \Big\}^{-1} x_i (A_i - e_i)$$

is the influence of subject i on the fitted coefficients (the estimate minus its limit is approximately $\sum_i IF_i$), and the derivative of the weight of subject i in the coefficients is $dw_i = -\{(1-e_i)/e_i\}x_i$ for a treated subject and $dw_i = \{e_i/(1-e_i)\}x_i$ for a control. With type = "stabilized" the weights and dw are those of "ipw" multiplied by the arm proportion, which is treated as a fixed number; IF is the same.

The following are errors, each naming the problem: a model that is not a glm; a family other than binomial or a link other than logit; prior weights other than 1 or an offset; a fit that did not converge; a fitted probability within 1e-10 of 0 or 1; rows dropped for missing values; a response that does not take both values 0 and 1.

The weights are never truncated or rescaled. The rows of weights, treat, ps, IF and dw are in the order of the rows of the data the model was fitted on.

Value

An object of class "ps_weights", a list with

weights The weights, one per row.

type "ipw" or "stabilized".

treat The treatment (0/1), one per row.

ps The fitted probabilities of treatment.

IF Matrix, one row per subject and one column per estimated coefficient (aliased columns dropped): the influence of each subject on the coefficients.

dw Matrix of the same shape: the derivative of each weight in the coefficients.

model The glm fit.

call The call.

print() shows the type, the number of rows and of coefficients, and by arm the size, the range of the weights and the effective sample size $(\sum w)^2 / \sum w^2$.

See Also

df_counting() (section "Estimated weights"), KM_diff(), cox_rhogamma()

Other survival_analysis: KM_diff(), cox_rhogamma(), df_counting(), wt.rg.S()

Examples

```
library(survival)
gbsg$tte <- gbsg$rfstime / 30.4375
pw <- ps_weights(hormon ~ age + meno + size + grade + nodes + pgr + er, data = gbsg,
                 type = "stabilized")
pw
gbsg$sw <- pw$weights

# weights treated as known (the default) and with their estimation carried through
known <- KM_diff(gbsg, "tte", "status", "hormon", weight.name = "sw")
est <- KM_diff(gbsg, "tte", "status", "hormon", weight.name = "sw",
               weight_variance = "estimated", weight_model = pw)
identical(known$dhat, est$dhat)                      # estimates do not change
summary(sqrt(est$sig2_dhat / known$sig2_dhat))       # standard errors do

dc <- df_counting(gbsg, "tte", "status", "hormon", weight.name = "sw",
                  weight_variance = "estimated", weight_model = pw)
dc$ess                                               # effective sample size by arm
cox_rhogamma(dc, scheme = "fh", scheme_params = list(rho = 0, gamma = 0))$hr_ci_asy
```

---

resampling_survival *Resampling Survival Curves for Confidence Bands*

Description

Performs resampling to generate survival curves for a group, used for constructing confidence bands.

Usage

```
resampling_survival(U, W, D, at.points, draws.band, surv, G_draws)
```

Arguments

U	Vector of observed times.
W	Vector of weights.
D	Vector of event indicators (0/1).
at.points	Vector of time points for evaluation.
draws.band	Number of resampling draws.
surv	Vector of survival estimates.
G_draws	Matrix of random draws for resampling.

Details

Each column is one resampled deviation $\hat{S}^*(t) - \hat{S}(t)$ at `at.points`, for one column of multipliers `G_draws` (one row per subject). With no weights or all weights equal, the multipliers perturb the events only: $-\hat{S}(t) \sum_{s \le t} \sum_i G_i w_i dN_i(s)/Y_w(s)$; these results are unchanged. With weights that are not all equal, the deviation is built from the per-subject influence contributions $\psi_i(t)$ of the robust variance (see `KM_diff`, section "Variance of the Kaplan-Meier estimate"): $-\hat{S}(t) \sum_i G_i w_i \psi_i(t)$, so its variance over draws is the robust variance $\hat{S}(t)^2 \sum_i w_i^2 \psi_i(t)^2$, which equals that of `survival::survfit(weights = w, robust = TRUE)`. The weights are treated as fixed and known. Both forms need `at.points` to hold every event time of the group (`KM_diff` arranges this).

Value

Matrix of resampled survival curves.

`risk_weighted` *Weighted risk set*

Description

Computes the weighted number at risk at a specified time.

Usage

```
risk_weighted(x, y, w = rep(1, length(y)))
```

Arguments

`x`	Time point.
`y`	Vector of event/censoring times.
`w`	Weights (default 1).

Value

Weighted number at risk at time `x`.

safe_run *Safe execution wrapper*

Description

Executes an R expression safely, returning NULL and printing an error message if an error occurs.

Usage

```
safe_run(expr)
```

Arguments

expr	An R expression to evaluate.

Value

The result of expr, or NULL if an error occurs.

score_calculation *Score calculation for weighted Cox model*

Description

Calculates the score and variance for the weighted Cox model.

Usage

```
score_calculation(ybar1, ybar0, dN1, dN0, wt_rg)
```

Arguments

ybar1	Numeric vector of event counts for group 1.
ybar0	Numeric vector of event counts for group 0.
dN1	Numeric vector of event increments for group 1.
dN0	Numeric vector of event increments for group 0.
wt_rg	Numeric vector of rho-gamma weights.

Value

List with score, variance, information, and weights.

validate_input *Validate required columns in a data frame*

Description

Checks that all required columns are present in a data frame.

Usage

```
validate_input(df, required_cols)
```

Arguments

df	Data frame to check.
required_cols	Character vector of required column names.

Value

NULL if all columns present, otherwise error.

validate_scheme_params *Validate weighting scheme parameters*

Description

Checks and validates the parameters for a given weighting scheme.

Usage

```
validate_scheme_params(scheme, scheme_params, S.pool)
```

Arguments

scheme	Character string specifying the weighting scheme.
scheme_params	List of parameters for the scheme.
S.pool	Numeric vector of pooled survival probabilities.

Value

Logical indicating if parameters are valid, or stops with error.

`wlr_cumulative` *Weighted log-rank cumulative statistics*

Description

Calculates cumulative weighted log-rank statistics for survival data.

Usage

```
wlr_cumulative(
  df_weights,
  scheme,
  scheme_params = list(rho = 0, gamma = 0),
  return_cumulative = FALSE
)
```

Arguments

`df_weights`	Data frame with the time grid, risk and event counts and pooled survival (`at_points`, `nbar0`, `nbar1`, `ybar0`, `ybar1`, `survP`, and `survG` for Schemper weights), or a `df_counting` result, which holds the same elements together with the subject-level data (`time`, `delta`, `z`, `w_hat`).
`scheme`	Weighting scheme.
`scheme_params`	List of scheme parameters.
`return_cumulative`	Logical; whether to return cumulative statistics.

Details

With no subject weights, with all subject weights equal, and for a bare table of counts (a data frame without the subject-level data) the variance of the score is the formula used before, computed from the counts; these results are unchanged. When `df_weights` is a `df_counting` result whose subject weights are not all equal (and inside `df_counting()` itself, unstratified and per stratum), `sig2.score` is the sum of squared per-subject score contributions $\sum_i \psi_i(0)^2$ described below, and `z.score` follows; `score` is unchanged. With `return_cumulative = TRUE` the variance at grid time t is the sum of squared contributions to the score up to t: a subject with $U_i \le t$ contributes $\psi_i(0)$, a subject still at risk contributes $-a_i\{Z_i c_K(t) - c_{Kz}(t)\}$, with $c_K(t) = \sum_{s \le t} K(s) d\hat{\Lambda}(s)$ and $c_{Kz}(t) = \sum_{s \le t} K(s)\bar{z}(s) d\hat{\Lambda}(s)$; its last value is the non-cumulative variance. A table of counts taken from a weighted analysis does not carry the subject-level data, so it gets the earlier formula: pass the `df_counting()` result itself to get the variance described below. With a `df_counting()` result made with `weight_variance = "estimated"`, `sig2.score` and `z.score`, cumulative or not, carry the estimation of the weights (section "Estimated weights"); on a bare table of counts nothing changes.

Value

List with `z.score`, `time` (the grid), `score` and `sig2.score` (the variance of the score); with `return_cumulative = TRUE` each is a vector along the grid.

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i\Big(D_i K(U_i)\{Z_i - \bar{z}(U_i,\beta)\} - e^{\beta Z_i}\sum_{s\le U_i} K(s)\{Z_i - \bar{z}(s,\beta)\}d\hat{\Lambda}(s,\beta)\Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s,\beta) = \sum_j a_j 1(U_j \ge s)e^{\beta Z_j}$, $\bar{z}(s,\beta) = \sum_j a_j 1(U_j \ge s)Z_j e^{\beta Z_j}/S_0(s,\beta)$ and $d\hat{\Lambda}(s,\beta) = \sum_j a_j D_j 1(U_j = s)/S_0(s,\beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2/i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the

score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i(A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t) \, dw_k, \qquad \Phi_i(t) = 1(i \in g) \, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{ \hat{S}_1(t) \Phi_i^1(t) - \hat{S}_0(t) \Phi_i^0(t) \}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t) \sum_{i=1}^{n} G_i \Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j} \, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i \psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i \{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j \{c_j(t)/a_j\} dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j) dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)
- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.
- With `"estimated"` the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (`df_counting()$ess`) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

`wlr_dhat_estimates` *Weighted Log-Rank and Difference Estimate at a Specified Time*

Description

Computes the weighted log-rank statistic, its variance, the difference in survival at a specified time (`tzero`), the variance of the difference, their covariance, and correlation, using flexible time-dependent weights. The weighting scheme is selected via the `scheme` argument and is calculated using `wt.rg.S`.

Usage

```
wlr_dhat_estimates(
  dfcounting,
  scheme = "fh",
  scheme_params = list(rho = 0, gamma = 0),
  tzero = NULL
)
```

Arguments

`dfcounting`	List output from `df_counting` containing risk sets, event counts, and survival estimates.
`scheme`	Character string specifying weighting scheme. One of: "fh" (Fleming-Harrington), "schemper", "XO", "MB", "custom_time", or "custom_code".

scheme_params	Named list with numeric weighting parameters rho and gamma (used for "fh" and "custom_code" schemes).
tzero	Time point at which to evaluate the difference in survival (default: NULL, no difference computed). The curves and the pooled survival are read at the last grid point <= tzero; before the first grid point the difference is 0 with variance 0, and at or after the last grid point the values at the last grid point are used.

Details

The weighting scheme is selected via the scheme argument and calculated using wt.rg.S. Supports standard Fleming-Harrington, Schemper, XO (Xu & O'Quigley), MB (Maggir-Burman), custom time-based, and custom code weights.

With no subject weights or all subject weights equal the three variances are the formulas used before, and these results are unchanged. When dfcounting is a df_counting result whose subject weights are not all equal, the log-rank statistic, the survival difference and their covariance rest on the same per-subject contributions: sig2.score is $\sum_i \psi_i(0)^2$ (below; the same as df_counting()$sig2_lr), sig2_dhat is $\sum_i d_i^2$ and cov_wlr_dhat is $\sum_i\{-\psi_i(0)\}d_i$, where d_i is subject i's contribution to the survival difference at the grid point read for tzero: $d_i = -\hat{S}_1 a_i \psi_i^{KM}$ in the treatment arm and $d_i = \hat{S}_0 a_i \psi_i^{KM}$ in the control arm, with ψ_i^{KM} the Kaplan-Meier influence contribution of that arm (see KM_diff, section "Variance of the Kaplan-Meier estimate"). sig2_dhat then equals KM_diff(draws = 0)$sig2_dhat at that grid point. score and dhat are unchanged.

With a df_counting() result made with weight_variance = "estimated", sig2.score, z.score, sig2_dhat, cov_wlr_dhat and cor_wlr_dhat carry the estimation of the weights (section "Estimated weights"); sig2_dhat then equals KM_diff(weight_variance = "estimated")$sig2_dhat at that grid point.

Value

A list with elements:

lr	Weighted log-rank test statistic.
sig2_lr	Variance of the log-rank statistic.
dhat	Difference in survival at tzero.
cov_wlr_dhat	Covariance between log-rank and difference at tzero.
sig2_dhat	Variance of the difference at tzero.
cor_wlr_dhat	Correlation between log-rank and difference at tzero.

Variance of the log-rank and Cox score with subject weights

With no subject weights, or with all subject weights equal, the weighted log-rank test and the weighted Cox estimate are computed as before, and their results are unchanged. When the subject weights are not all equal, the variance of the weighted score is the sum of squared per-subject score contributions,

$$\widehat{Var}\{U(\beta)\} = \sum_i \psi_i(\beta)^2,$$

$$\psi_i(\beta) = a_i\Big(D_i K(U_i)\{Z_i - \bar{z}(U_i,\beta)\} - e^{\beta Z_i}\sum_{s\le U_i} K(s)\{Z_i - \bar{z}(s,\beta)\}d\hat{\Lambda}(s,\beta)\Big),$$

where U_i, D_i, Z_i and a_i are subject i's time, event indicator, arm (0/1) and subject weight, $K(s)$ is the time weight of the scheme, $S_0(s,\beta) = \sum_j a_j 1(U_j \geq s) e^{\beta Z_j}$, $\bar{z}(s,\beta) = \sum_j a_j 1(U_j \geq s) Z_j e^{\beta Z_j}/S_0(s,\beta)$ and $d\hat{\Lambda}(s,\beta) = \sum_j a_j D_j 1(U_j = s)/S_0(s,\beta)$. Sums over s run over distinct times (events at a tied time are added up), and a term whose S_0 is 0 is 0. The score itself, $-\sum_i \psi_i(\beta)$, is the one computed before and does not change, and neither does any point estimate. The log-rank z-score is the score at $\beta = 0$ over $\sqrt{\sum_i \psi_i(0)^2}$; the variance of the log hazard ratio estimate is $\sum_i \psi_i(\hat{\beta})^2 / i(\hat{\beta})^2$ with $i(\hat{\beta})$ the information (unchanged); in a stratified analysis the contributions are computed within each stratum and the score and its variance are sums over strata. For Fleming-Harrington (0, 0) weights this is the robust variance of `survival::coxph(weights = w, robust = TRUE, ties = "breslow")`: the same robust score test and the same robust standard error. Results no longer depend on how the weights are scaled (weights `w`, `10 * w` and `w / sum(w)` give the same z-score and standard error). By default the weights are treated as fixed and known. For weights estimated by a logistic propensity model, `weight_variance = "estimated"` carries their estimation into the test and the intervals (section "Estimated weights"). When the weights are extreme (low effective sample size) the intervals can be too short under either option.

Estimated weights

Propensity-score weights are estimated from the data. `weight_variance` says how standard errors, intervals, bands and tests treat them.

`"known"` **(default)** The weights are fixed, known numbers: the formulas of the two sections above. `weight_model` is ignored.

`"estimated"` The weights were estimated by a logistic regression for the treatment, given as `weight_model` (the object returned by `ps_weights`), and that estimation is carried into the inference.

Estimates never change: survival curves, their difference, RMST, the log-rank score, hazard ratios, counts and risk tables are identical under the two options. Only standard errors, intervals, bands and tests differ.

Notation. There are n subjects, the rows analysed. Subject i has arm A_i, row x_i of the model matrix of the propensity model (p columns), fitted probability e_i and weight w_i (written a_i in the score formulas above). The influence of subject i on the fitted coefficients and the derivative of its weight in the coefficients are

$$IF_i = \Big\{ \sum_j x_j x_j' e_j(1-e_j) \Big\}^{-1} x_i (A_i - e_i), \qquad dw_i = -\frac{1-e_i}{e_i} x_i \, (A_i = 1), \quad dw_i = \frac{e_i}{1-e_i} x_i \, (A_i = 0)$$

(for stabilised weights w_i and dw_i are multiplied by the arm proportion).

Kaplan-Meier estimate of a group g (an arm; an arm within a stratum; all subjects). With $\psi_i(t)$ the influence value of the section "Variance of the Kaplan-Meier estimate", computed from the group's own weighted counts,

$$D_g(t) = \sum_{k \in g} \psi_k(t)\, dw_k, \qquad \Phi_i(t) = 1(i \in g)\, w_i \psi_i(t) + IF_i' D_g(t) \quad (i = 1, \ldots, n),$$

$$\widehat{Var}[\hat{S}_g(t)] = \hat{S}_g(t)^2 \sum_{i=1}^{n} \Phi_i(t)^2.$$

Every subject contributes, also those outside the group, through the model. The variance of the difference of the two arms is

$$\widehat{Var}[\hat{S}_1(t) - \hat{S}_0(t)] = \sum_{i=1}^{n} \{\hat{S}_1(t)\Phi_i^1(t) - \hat{S}_0(t)\Phi_i^0(t)\}^2,$$

which is **not** the sum of the two arm variances: the arms share the model's coefficients. A resampled curve of arm a is $-\hat{S}_a(t)\sum_{i=1}^{n} G_i\Phi_i^a(t)$ with the standard-normal multipliers G_i that `KM_diff` generates in any case; the resampled difference curves, the simultaneous band and the RMST variance and band of `cumulative_rmst_bands` follow from them unchanged.

Score of the log-rank test and of the weighted Cox estimate. With $\psi_i(\beta)$ the contribution of the section "Variance of the log-rank and Cox score with subject weights", computed within each stratum,

$$B(\beta) = \sum_{j=1}^{n} \frac{\psi_j(\beta)}{a_j}\, dw_j, \qquad \psi_i^{ew}(\beta) = \psi_i(\beta) + IF_i' B(\beta), \qquad \widehat{Var}\{U(\beta)\} = \sum_{i=1}^{n} \psi_i^{ew}(\beta)^2.$$

The log-rank z-score, the variance of the log hazard ratio estimate (the sum at $\hat{\beta}$ over the squared information) and a resampled score ($-\sum_i G_i\psi_i^{ew}(\beta)$) are formed as before with ψ^{ew} in place of ψ. In a stratified analysis the variance is this single sum over all subjects, not a sum of per-stratum variances. The cumulative variance of `wlr_cumulative` is $\sum_i\{c_i(t) + IF_i' B(t)\}^2$ with $c_i(t)$ the contribution of subject i to the score up to t and $B(t) = \sum_j\{c_j(t)/a_j\}dw_j$; in `wlr_dhat_estimates` the contribution to the survival difference is $d_i + IF_i'(\hat{S}_0 D_0 - \hat{S}_1 D_1)$ at the grid point read for `tzero`, and the covariance uses both adjusted contributions. The outputs of `df_counting()` that come from the `survival` package (`logrank_results`, `cox_results$score`, `$se`, `$hr_ci`, `$pval`) carry the same adjustment, applied to `coxph()`'s weighted score and dfbeta residuals r: $r_i + IF_i' \sum_j (r_j/w_j)dw_j$.

How to use it. Fit the propensity model on exactly the rows to be analysed with `ps_weights`, put its weights in a column of `df`, and pass `weight_variance = "estimated"` and `weight_model` to `df_counting()`, `KM_diff`, `cumulative_rmst_bands` or `plotKM.band_subgroups`. `cox_rhogamma`, `wlr_cumulative` and `wlr_dhat_estimates` take the option from the `df_counting()` result they are given (its elements `weight_variance` and `ew_parts`). The rows of `df`, the treatment column and the weight column must be those of `weight_model`; anything else is an error.

Scope and limits of this version.

- A logistic `glm` for a binary treatment, fitted on exactly the rows analysed. For weights from any other model use `"known"`.
- A subgroup analysis that reuses weights from a model fitted on the full sample must use `"known"`: the model's rows are then not the rows analysed. (The subgroup curves of `plotKM.band_subgroups()` are point estimates and are not affected.)
- `draws > 0` (the resampling-based pointwise variance) is not available with `"estimated"`; `draws.band` and the resampling of `cox_rhogamma()` are.
- The time weights of the scheme (Fleming-Harrington and the others) depend on the weighted pooled curve; that dependence on the estimated weights is ignored.
- With stabilised weights the arm proportion is treated as a fixed number. The Kaplan-Meier quantities do not depend on that: they are unchanged when an arm's weights are multiplied by a constant.

- With "estimated" the influence-based formulas always run; whether the weights are all equal is not tested.
- Carrying the estimation through does not repair extreme weights: when the effective sample size (df_counting()$ess) is a small fraction of an arm, standard errors, intervals and tests can be too optimistic under either option.

The device of carrying the estimation of a nuisance quantity into the inference by perturbing its estimate is that of Leon, Cai and Wei (2009); it is used here in linearised form.

---

wt.rg.S *Compute Time-Dependent Weights for Survival Analysis*

Description

Calculates time-dependent weights for survival analysis according to various schemes (Fleming-Harrington, Schemper, XO, MB, custom).

Usage

```
wt.rg.S(
  S,
  scheme = c("fh", "schemper", "XO", "MB", "custom_time", "fh_exp1", "fh_exp2"),
  rho = NULL,
  gamma = NULL,
  Scensor = NULL,
  Ybar = NULL,
  tpoints = NULL,
  t.tau = NULL,
  w0.tau = 0,
  w1.tau = 1,
  mb_tstar = NULL,
  details = FALSE
)
```

Arguments

S	Numeric vector of survival probabilities at each time point.
scheme	Character; weighting scheme. One of: 'fh', 'schemper', 'XO', 'MB', 'custom_time', 'fh_exp1', 'fh_exp2'.
rho	Numeric; rho parameter for Fleming-Harrington weights. Required if scheme='fh'.
gamma	Numeric; gamma parameter for Fleming-Harrington weights. Required if scheme='fh'.
Scensor	Numeric vector; censoring KM curve for Schemper weights. Must have same length as S.
Ybar	Numeric vector; risk set sizes for XO weights. Must have same length as S.
tpoints	Numeric vector; time points corresponding to S. Required for MB and custom_time schemes.

t.tau	Numeric; cutoff time for custom_time weights.
w0.tau	Numeric; weight before t.tau for custom_time weights.
w1.tau	Numeric; weight after t.tau for custom_time weights.
mb_tstar	Numeric; cutoff time for Magirr-Burman weights.
details	Logical; if TRUE, returns list with weights and diagnostic info. Default: FALSE.

Details

This function implements several weighting schemes for weighted log-rank tests: See vignette for details This function implements several weighting schemes for weighted log-rank tests:

Fleming-Harrington (fh) w(t) = S(t-)^rho * (1-S(t-))^gamma. See vignette for common choices.

Schemper w(t) = S(t-)/G(t-) where G is the censoring distribution.

Xu-O'Quigley (XO) w(t) = S(t-)/Y(t) where Y is risk set size.

Magirr-Burman (MB) w(t) = 1/max(S(t-), S(t*)), with S(t*) the survival at the last time point <= t* (carried forward when t* is at or beyond the last point); 0 where both are 0.

Custom time Step function with weight w_0 before t* and w_1 after t*.

Value

If details=FALSE (default): Numeric vector of weights with length equal to S.

If details=TRUE: List containing:

weights Numeric vector of calculated weights

S Input survival probabilities

S_left Left-continuous survival probabilities

scheme Scheme name

Note

All weights are calculated using left-continuous survival probabilities S(t-) to ensure consistency with counting process notation.

References

Fleming, T. R. and Harrington, D. P. (1991). Counting Processes and Survival Analysis. Wiley.

Magirr, D. and Burman, C. F. (2019). Modestly weighted logrank tests. Statistics in Medicine, 38(20), 3782-3790.

Schemper, M., Wakounig, S., and Heinze, G. (2009). The estimation of average hazard ratios by weighted Cox regression. Statistics in Medicine, 28(19), 2473-2489.

See Also

Other survival_analysis: KM_diff(), cox_rhogamma(), df_counting(), ps_weights()

Other weighted_tests: cox_rhogamma(), df_counting()

Examples

```
# Generate example survival curve
time <- seq(0, 24, by = 0.5)
surv <- exp(-0.05 * time)  # Exponential survival

# Fleming-Harrington (0,1) weights
w_fh01 <- wt.rg.S(surv, scheme = "fh", rho = 0, gamma = 1, tpoints = time)

# Magirr-Burman weights
w_mb <- wt.rg.S(surv, scheme = "MB", mb_tstar = 12, tpoints = time)

# Standard log-rank (equal weights)
w_lr <- wt.rg.S(surv, scheme = "fh", rho = 0, gamma = 0)

# Plotting examples
plot(time, w_fh01, type = "l", main = "FH(0,1) Weights")
plot(time, w_mb, type = "l", main = "MB(12) Weights")

# Compare multiple schemes
plot(time, w_lr, type = "l", ylim = c(0, 2))
lines(time, w_fh01, col = "blue")
legend("topleft", c("Log-rank", "FH(0,1)"), col = 1:2, lty = 1)
```

Index

∗ **diagnostic_functions**
check_results, 7
∗ **plotting_functions**
KM_diff, 37
∗ **survival_analysis**
cox_rhogamma, 10
df_counting, 24
KM_diff, 37
ps_weights, 59
wt.rg.S, 72
∗ **weighted_tests**
cox_rhogamma, 10
df_counting, 24
wt.rg.S, 72

add_legends, 3
add_median_annotation, 4
add_risk_table, 5

calculate_risk_event_counts, 6
check_km_curve, 7
check_results, 7, *27*
ci_cox, 9
count_weighted, 10
cox_rhogamma, 10, *14*, *23*, *27*, *30, 31*, *41*, *58*, *67*, *71*
cox_rhogamma(), *31*, *41*, *61*, *73*
cox_rhogamma_resample, *11*, 15
cox_score_rhogamma, *11*, *15*, 17
coxph, *31*
create_baseline_table, 18
cumulative_rmst_bands, *13, 14*, 20, *23*, *29, 30*, *38*, *40, 41*, *58*, *67*, *71*
cumulative_rmst_bands(), *59*

df_counting, *7, 8*, *11*, *15, 16*, *18*, 24, *28*, *34, 35*, *39*, *41*, *65*, *69*
df_counting(), *15*, *41*, *59*, *61*, *73*

extract_and_calc_weights, 32
extract_group_data, 32

find_cox_root, 33
format_pval, 33

get_censoring_and_events, 34
get_dfcounting, 34
get_event_risk_matrices, 35
get_riskpoints, 35
get_validated_weights, 36
get_weights, 36

KM_diff, *13, 14*, *21*, *23*, *29–31*, 37, *40, 41*, *43*, *55*, *57, 58*, *62*, *67*, *69*, *71*
KM_diff(), *15*, *31*, *59*, *61*, *73*
KM_estimates, *6*, 42
KM_plot_2sample_weighted_counting, 43
km_quantile, 46
km_quantile_table, 47
kmq_calculations, *46*, 48

N_rhogamma, 48

plot_km, 49
plot_km_confint_polygon, 50
plot_km_curves_counting, 50
plot_weight_schemes, 52
plot_weighted_km, 53
plotKM.band_subgroups, *14*, *23*, *30*, *41*, 54, *58*, *67*, *71*
plotKM.band_subgroups(), *59*
print.ps_weights *(*ps_weights*)*, 59
ps_weights, *11*, *13, 14*, *21–23*, *26*, *29, 30*, *38*, *40, 41*, *56–58*, 59, *66, 67*, *70, 71*
ps_weights(), *15*, *31*, *41*, *73*

resampling_survival, *7*, *21*, *28*, *38, 39*, 61
risk_weighted, 62

safe_run, 63
score_calculation, *18*, 63

survdiff, *31*
survfit, *31*

validate_input, 64
validate_scheme_params, 64

wlr_cumulative, *14*, *23*, *27*, *30*, *40*, *41*, *58*, 65, *67*, *71*
wlr_dhat_estimates, *14*, *23*, *27*, *30*, *40*, *41*, *58*, *67*, 68, *71*
wt.rg.S, *11*, *15*, 72
wt.rg.S(), *15*, *31*, *41*, *61*