Package 'forestsearch'

May 8, 2026

Title Exploratory Subgroup Identification in Clinical Trials with Survival Endpoints

Version 0.1.0

Description Implements statistical methods for exploratory subgroup identification in clinical trials with survival endpoints. Provides tools for identifying patient subgroups with differential treatment effects using machine learning approaches including Generalized Random Forests (GRF), LASSO regularization, and exhaustive combinatorial search algorithms. Features bootstrap bias correction using infinitesimal jackknife methods to address selection bias in post-hoc analyses. Designed for clinical researchers conducting exploratory subgroup analyses in randomized controlled trials, particularly for multi-regional clinical trials (MRCT) requiring regional consistency evaluation. Supports both accelerated failure time (AFT) and Cox proportional hazards models with comprehensive diagnostic and visualization tools. Methods are described in León et al. (2024) <doi:10.1002/sim.10163>.

License MIT + file LICENSE

Depends R (>= 4.1.0)

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

Imports data.table, doFuture, dplyr, foreach, future, future.apply, future.callr, ggplot2, glmnet, grf, gt, patchwork, policytree, progressr, randomForest, rlang, stringr, survival, weightedsurv

Suggests DiagrammeR, doRNG, htmltools, tidyr, forestploter, cubature, svglite, knitr, rmarkdown, katex

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

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

NeedsCompilation no

Author Larry Leon [aut, cre]

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

Repository CRAN

Date/Publication 2026-03-23 17:20:14 UTC

Contents

`build_classification_table` *Build Classification Rate Table from Simulation Results*

Description

Constructs a publication-quality `gt` table summarizing subgroup identification and classification rates across one or more data generation scenarios and analysis methods. The layout mirrors Table 4 of Leon et al. (2024) with metrics grouped by model scenario (null / alt) and columns for each analysis method.

Usage

```
build_classification_table(
  scenario_results,
  analyses = NULL,
  digits = 2,
  title = "Subgroup Identification and Classification Rates",
  n_sims = NULL,
  bold_threshold = 0.05,
  font_size = 12
)
```

Arguments

`scenario_results`

Named list. Each element is itself a list with:

results `data.table` from `run_simulation_analysis`.

label Character scenario label, e.g., `"M1"`.

n_sample Integer sample size.

dgm DGM object (for true HRs and subgroup prevalence).

hypothesis Character: `"null"` or `"alt"`.

analyses	Character vector of analysis labels to include (e.g., c("FS", "FSlg", "GRF")). When NULL, all unique values of results$analysis across scenarios are used.
digits	Integer. Decimal places for proportions. Default: 2.
title	Character. Table title. Default: "Subgroup Identification and Classification Rates".
n_sims	Integer. Number of simulations (for subtitle). Default: NULL.
bold_threshold	Numeric. Type I error threshold above which the any(H) value is shown in bold. Set NULL to disable. Default: 0.05.
font_size	Numeric. Font size in pixels for table text. Default: 12. Increase to 14 or 16 for larger display.

Details

For each scenario the function computes:

- any(H): Proportion of simulations identifying any subgroup.
- sens(H): Mean sensitivity (only under alternative).
- sens(Hc): Mean specificity.
- ppv(H): Mean positive predictive value (only under alternative).
- ppv(Hc): Mean negative predictive value.
- avg|H|: Mean size of identified subgroup (when found).

Under the null hypothesis the rows are reduced to any(H), sens(Hc), ppv(Hc), and avg|H|.

Value

A gt table object.

See Also

format_oc_results, summarize_simulation_results

---

build_estimation_table

Build Estimation Properties Table from Simulation Results

Description

Constructs a publication-quality gt table summarizing estimation properties for hazard ratios in the identified subgroup and its complement. The layout mirrors Table 5 of Leon et al. (2024), showing average estimate, empirical SD, min, max, and relative bias for each estimator.

Usage

```
build_estimation_table(
  results,
  dgm,
  analysis_method = "FSlg",
  n_boots = NULL,
  digits = 2,
  title = "Estimation Properties",
  subtitle = NULL,
  font_size = 12,
  cde_H = NULL,
  cde_Hc = NULL
)
```

Arguments

results	data.table or data.frame. Simulation results from run_simulation_analysis, optionally enriched with bootstrap bias-corrected columns (hr.H.bc, hr.Hc.bc).
dgm	DGM object. Used for true parameter values (hr_H_true, hr_Hc_true, and AHR truth via get_dgm_hr).
analysis_method	Character. Which analysis method to tabulate (e.g., "FSlg"). Default: "FSlg".
n_boots	Integer or NULL. Number of bootstraps. When non-NULL, appended to the subtitle as "(B = n_boots bootstraps)". Default: NULL.
digits	Integer. Decimal places. Default: 2.
title	Character. Table title.
subtitle	Character or NULL. Optional user-supplied subtitle text. Auto-populated statistics (method, estimable count, proportion) are always shown. When non-NULL, the custom text is displayed first with stats appended in parentheses, e.g. "My title (FSlg: 18/20 (90%) estimable)".
font_size	Numeric. Font size in pixels for table text. Default: 12. Increase to 14 or 16 for larger display.
cde_H	Numeric or NULL. Controlled direct effect (theta-ddagger(H)) for the true harm subgroup. When non-NULL, an additional b-ddagger bias column is shown. When NULL (default), auto-detected from dgm$cde_H or dgm$hazard_ratios$CDE_harm.
cde_Hc	Numeric or NULL. Controlled direct effect for the complement. Auto-detected analogously.

Details

Uses the paper's notation conventions:

- theta-dagger: Marginal (causal) HR truth
- theta-ddagger: Controlled direct effect (CDE) truth
- theta-hat(H-hat): Plugin Cox estimate in identified subgroup

- theta-hat*(H-hat): Bootstrap bias-corrected estimate

Includes both Cox-based HR and AHR (Average Hazard Ratio from loghr_po) estimators when AHR columns are present in the results.

For each subgroup (H and Hc) the function reports:

- **Avg**: Mean of the estimates across estimable simulations.
- **SD**: Empirical standard deviation.
- **Min / Max**: Range.
- **b-dagger**: Relative bias (percent) vs marginal truth, `100 * (Avg - theta_dagger) / theta_dagger`.
- **b-ddagger** (conditional): Relative bias (percent) vs CDE truth, shown when CDE values are available.

When bootstrap-corrected columns (`hr.H.bc`, `hr.Hc.bc`) are present in `results`, an additional bias-corrected row (theta-hat*(H-hat)) is added per subgroup.

When AHR columns (`ahr.H.hat`, `ahr.Hc.hat`) are present, AHR estimation rows are appended using the DGM's true AHR values for relative bias calculation.

When CDE columns (`cde.H.hat`, `cde.Hc.hat`) are present and CDE truth values are available, CDE estimation rows (theta-ddagger(H-hat)) are appended. The b-dagger column for CDE rows reports bias relative to the CDE truth rather than the marginal HR.

Value

A `gt` table object, or `NULL` if no estimable realizations exist.

See Also

`build_classification_table`, `format_oc_results`, `get_dgm_hr`

---

calibrate_cens_adjust *Calibrate Censoring Adjustment to Match DGM Reference Distribution*

Description

Uses root-finding to select a value of `cens_adjust` for `simulate_from_dgm` such that a chosen censoring summary statistic in the simulated data matches the corresponding statistic from the DGM reference data (`dgm$df_super`).

Usage

```
calibrate_cens_adjust(
  dgm,
  target = c("rate", "km_median"),
  n = 1000,
  rand_ratio = 1,
  analysis_time = 48,
  max_entry = 24,
  seed = 42,
  interval = c(-3, 3),
  tol = 1e-04,
  n_eval = 2000,
  verbose = TRUE,
  ...
)
```

Arguments

dgm	An "aft_dgm_flex" object from generate_aft_dgm_flex.
target	Character. Calibration target: "rate" (default) or "km_median".
n	Integer. Sample size passed to simulate_from_dgm. Default 1000.
rand_ratio	Numeric. Randomisation ratio passed to simulate_from_dgm. Default 1.
analysis_time	Numeric. Calendar analysis time passed to simulate_from_dgm. Must be on the DGM time scale. Default 48.
max_entry	Numeric. Maximum staggered entry time passed to simulate_from_dgm. Default 24.
seed	Integer. Base random seed. Each evaluation of the objective function uses this seed for reproducibility. Default 42.
interval	Numeric vector of length 2. Search interval for cens_adjust on the log scale. Default c(-3, 3) (corresponding roughly to a 20-fold decrease/increase in censoring times).
tol	Numeric. Root-finding tolerance. Default 1e-4.
n_eval	Integer. Sample size used inside the objective function during root-finding. Smaller values are faster but noisier; increase for precision. Default 2000.
verbose	Logical. Print search progress and final result. Default TRUE.
...	Additional arguments passed to simulate_from_dgm (e.g. strata_rand, time_eos).

Details

Two calibration targets are supported:

"rate" Overall censoring rate (proportion censored). Finds cens_adjust such that mean(event_sim == 0) in simulated data equals mean(event == 0) in dgm$df_super.

"km_median" KM-based median censoring time, estimated by reversing the event indicator so censored observations become the "event" of interest. Finds cens_adjust such that the simulated KM median matches the reference KM median.

How the objective function works:

At each candidate `cens_adjust` value, the objective function:

1. Calls `simulate_from_dgm()` with `n = n_eval` and the candidate `cens_adjust`.
2. Calls `check_censoring_dgm()` with `verbose = FALSE` to extract the target metric.
3. Returns `sim_metric - ref_metric`.

`uniroot` finds the zero crossing, i.e. the `cens_adjust` at which simulated and reference metrics are equal.

Monotonicity:

The objective is monotone in `cens_adjust` for both targets:

- Larger `cens_adjust` → longer censoring times → lower censoring rate and higher KM median.
- Smaller `cens_adjust` → shorter censoring times → higher censoring rate and lower KM median.

If `uniroot` fails (the target lies outside the search interval), the boundary values are printed and a wider `interval` should be tried.

Stochastic noise:

Because the objective function involves simulation, there is Monte Carlo noise. Setting a fixed `seed` and a sufficiently large `n_eval` (>= 2000) reduces noise enough for reliable root-finding. The `tol` argument controls the root-finding tolerance on the `cens_adjust` scale (not the metric scale).

Value

A named list with elements:

`cens_adjust` Calibrated `cens_adjust` value.

`target` Calibration target used.

`ref_value` Reference metric value from `dgm$df_super`.

`sim_value` Achieved metric value in simulated data at the calibrated `cens_adjust`.

`residual` Absolute difference between `sim_value` and `ref_value`.

`iterations` Number of `uniroot` iterations.

`diagnostic` Output of `check_censoring_dgm` at the calibrated value (invisibly).

See Also

`simulate_from_dgm`, `check_censoring_dgm`, `generate_aft_dgm_flex`

Examples

```
library(survival)

# Build DGM on months scale
gbsg$time_months <- gbsg$rfstime / 30.4375
```

```
dgm <- generate_aft_dgm_flex(
  data            = gbsg,
  continuous_vars = c("age", "size", "nodes", "pgr", "er"),
  factor_vars     = c("meno", "grade"),
  outcome_var     = "time_months",
  event_var       = "status",
  treatment_var   = "hormon",
  subgroup_vars   = c("er", "meno"),
  subgroup_cuts   = list(er = 20, meno = 0)
)

# Calibrate so simulated censoring rate matches reference
cal_rate <- calibrate_cens_adjust(
  dgm           = dgm,
  target        = "rate",
  n             = 1000,
  analysis_time = 84,
  max_entry     = 24
)
cat("Calibrated cens_adjust (rate):", cal_rate$cens_adjust, "\n")

# Calibrate to KM median censoring time instead
cal_km <- calibrate_cens_adjust(
  dgm           = dgm,
  target        = "km_median",
  n             = 1000,
  analysis_time = 84,
  max_entry     = 24
)
cat("Calibrated cens_adjust (km_median):", cal_km$cens_adjust, "\n")

# Use calibrated value in simulation
sim <- simulate_from_dgm(
  dgm           = dgm,
  n             = 1000,
  analysis_time = 84,
  max_entry     = 24,
  cens_adjust   = cal_rate$cens_adjust,
  seed          = 123
)
mean(sim$event_sim)   # event rate
mean(sim$event_sim == 0)  # censoring rate — should match ref
```

`calibrate_k_inter` *Calibrate k_inter for Target Subgroup Hazard Ratio*

Description

Finds the interaction effect multiplier (k_inter) that achieves a target hazard ratio in the harm subgroup.

Usage

```
calibrate_k_inter(
  target_hr_harm,
  model = "alt",
  k_treat = 1,
  cens_type = "weibull",
  k_inter_range = c(-100, 100),
  tol = 1e-06,
  use_ahr = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

target_hr_harm	Numeric. Target hazard ratio for the harm subgroup
model	Character. Model type ("alt" only). Default: "alt"
k_treat	Numeric. Treatment effect multiplier. Default: 1
cens_type	Character. Censoring type. Default: "weibull"
k_inter_range	Numeric vector of length 2. Search range for k_inter. Default: c(-100, 100)
tol	Numeric. Tolerance for root finding. Default: 1e-6
use_ahr	Logical. If TRUE, calibrate to AHR instead of Cox-based HR. Default: FALSE
verbose	Logical. Print diagnostic information. Default: FALSE
...	Additional arguments passed to create_gbsg_dgm

Details

This function uses uniroot to find the k_inter value such that the empirical HR (or AHR) in the harm subgroup equals target_hr_harm.

Value

Numeric value of k_inter that achieves the target HR

Examples

```
# Find k_inter for HR = 1.5 in harm subgroup
k <- calibrate_k_inter(target_hr_harm = 1.5, verbose = TRUE)

# Verify
dgm <- setup_gbsg_dgm(model = "alt", k_inter = k, verbose = FALSE)
print(dgm)
```

```
# Calibrate to AHR instead
k_ahr <- calibrate_k_inter(target_hr_harm = 1.5, use_ahr = TRUE, verbose = TRUE)
dgm_ahr <- setup_gbsg_dgm(model = "alt", k_inter = k_ahr, verbose = FALSE)
print(dgm_ahr)
```

`check_censoring_dgm` *Diagnose Censoring Consistency Between DGM Source Data and Simulated Data*

Description

Compares the censoring distribution observed in the data used to build the DGM against the censoring generated by `simulate_from_dgm`. Reports censoring rates, time quantiles, KM-based median censoring times, and flags substantial discrepancies.

Usage

```
check_censoring_dgm(
  sim_data,
  dgm,
  treat_var = "treat_sim",
  rate_tol = 0.1,
  median_tol = 0.25,
  verbose = TRUE
)
```

Arguments

`sim_data`	A `data.frame` returned by `simulate_from_dgm`.
`dgm`	An `"aft_dgm_flex"` object from `generate_aft_dgm_flex`. The super population (`dgm$df_super`) provides reference censoring times and event indicators on the DGM time scale.
`treat_var`	Character. Name of the treatment column in `sim_data` used for arm-stratified comparisons. Default `"treat_sim"`.
`rate_tol`	Numeric. Absolute tolerance (proportion scale) for flagging a censoring-rate discrepancy. Default `0.10` (10 pp).
`median_tol`	Numeric. Relative tolerance for flagging a KM median censoring-time discrepancy. Default `0.25` (25 percent).
`verbose`	Logical. If `TRUE`, prints the full diagnostic table. Default `TRUE`.

Details

The reference censoring distribution is derived from `dgm$df_super`, sampled with replacement from the data passed to `generate_aft_dgm_flex()`. Columns `y` (observed time) and `event` (event indicator) in `df_super` reflect the original observed censoring process on the DGM time scale.

The KM median censoring time is estimated by reversing the event indicator (`1 - event`), treating events as censored and censored observations as the event of interest. This gives a non-parametric estimate of the censoring time distribution unconfounded by event occurrence.

Common causes of discrepancy: (1) time-scale mismatch (DGM built on days, `analysis_time` in months); check `exp(dgm$model_params$mu)` against your `analysis_time`. (2) Large `cens_adjust` shifting censoring substantially from the fitted model. (3) Short `analysis_time` or `time_eos` making administrative censoring dominate the censoring process.

Value

Invisibly returns a named list. Elements are: `rates` (data frame of censoring rates overall and by arm); `quantiles` (data frame of censoring-time quantiles among censored subjects); `km_medians` (data frame of KM-based median censoring times); and `flags` (character vector of triggered warnings, empty if none).

See Also

`simulate_from_dgm`, `generate_aft_dgm_flex`

Examples

```
dgm <- setup_gbsg_dgm(model = "null", verbose = FALSE)
sim_data <- simulate_from_dgm(dgm, n = 200)
check_censoring_dgm(sim_data, dgm = dgm)
```

---

compute_detection_probability

Compute Probability of Detecting True Subgroup

Description

Calculates the probability that a true subgroup with given hazard ratio will be detected using the ForestSearch consistency-based criteria.

Usage

```
compute_detection_probability(
  theta,
  n_sg,
  prop_cens = 0.3,
  hr_threshold = 1.25,
  hr_consistency = 1,
```

```
  method = c("cubature", "monte_carlo"),
  n_mc = 100000L,
  tol = 1e-04,
  verbose = FALSE
)
```

Arguments

theta	Numeric. True hazard ratio in the subgroup. Can be a vector for computing detection probability across multiple HR values.
n_sg	Integer. Subgroup sample size.
prop_cens	Numeric. Proportion censored (0-1). Default: 0.3
hr_threshold	Numeric. HR threshold for detection (e.g., 1.25). This is the threshold that the average HR across splits must exceed.
hr_consistency	Numeric. HR consistency threshold (e.g., 1.0). This is the threshold each individual split must exceed. Default: 1.0
method	Character. Integration method: "cubature" (recommended for accuracy) or "monte_carlo" (faster for exploration). Default: "cubature"
n_mc	Integer. Number of Monte Carlo samples if method = "monte_carlo". Default: 100000
tol	Numeric. Relative tolerance for cubature integration. Default: 1e-4
verbose	Logical. Print progress for vector inputs. Default: FALSE

Details

This function computes P(detect | theta) using the asymptotic normal approximation for the log hazard ratio estimator. The detection criterion is based on ForestSearch's split-sample consistency evaluation:

1. The subgroup HR estimate must exceed hr_threshold on average
2. Each split-half must individually exceed hr_consistency

The approximation assumes:

- Large sample sizes (CLT applies)
- Var(log(HR)) ~ 4/d per treatment arm
- Independence between split-halves (conditional on true effect)

Value

If theta is scalar, returns a single probability. If theta is a vector, returns a data.frame with columns: theta, probability.

Examples

```
# Single HR value
prob <- compute_detection_probability(
  theta = 1.5,
  n_sg = 60,
  prop_cens = 0.2,
  hr_threshold = 1.25
)

# Vector of HR values for power curve
hr_values <- seq(1.0, 2.5, by = 0.1)
results <- compute_detection_probability(
  theta = hr_values,
  n_sg = 60,
  prop_cens = 0.2,
  hr_threshold = 1.25,
  verbose = TRUE
)

# Plot detection probability curve
plot(results$theta, results$probability, type = "l",
     xlab = "True HR", ylab = "P(detect)")
```

`compute_dgm_cde` *Compute and Attach CDE Values to a DGM Object*

Description

Calculates Controlled Direct Effect (CDE) hazard ratios from the super-population potential outcomes (`theta_0`, `theta_1`) and attaches them to the DGM's `hazard_ratios` list. This enables automatic CDE detection by `build_estimation_table`.

Usage

```
compute_dgm_cde(dgm, harm_col = NULL)
```

Arguments

`dgm`	A DGM object (e.g., from `create_gbsg_dgm` or `generate_aft_dgm_flex`). Must contain `df_super_rand` with columns `theta_0` and `theta_1`.
`harm_col`	Character. Name of the subgroup indicator column in `dgm$df_super_rand`. If `NULL` (default), auto-detected from `flag.harm`, `flag_harm`, or `H`.

Details

The CDE for subgroup S is defined as:

`CDE(S) = mean(exp(theta_1[S])) / mean(exp(theta_0[S]))`

which is the ratio of average hazard contributions on the natural scale. This differs from the AHR (`exp(mean(loghr_po))`) due to Jensen's inequality. In the notation of Leon et al. (2024), CDE corresponds to theta-ddagger.

The function detects the subgroup indicator column automatically, checking for `flag.harm`, `flag_harm`, and `H` in the super-population data frame.

Value

The DGM object with CDE values added to `dgm$hazard_ratios` (`CDE`, `CDE_harm`, `CDE_no_harm`) and to top-level fields (`dgm$CDE`, `dgm$cde_H`, `dgm$cde_Hc`).

See Also

`build_estimation_table`, `get_dgm_hr`

Examples

```
dgm <- setup_gbsg_dgm(model = "alt", k_inter = 2.0, verbose = FALSE)
dgm <- compute_dgm_cde(dgm)
dgm$hazard_ratios$CDE_harm   # theta-ddagger(H)
dgm$hazard_ratios$CDE         # theta-ddagger overall
```

---

cox_ahr_cde_analysis *Comprehensive Wrapper for Cox Spline Analysis with AHR and CDE Plotting*

Description

This wrapper function combines Cox spline fitting with comprehensive visualization of Average Hazard Ratios (AHRs) and Controlled Direct Effects (CDEs) as described in the MRCT subgroups analysis documentation.

Usage

```
cox_ahr_cde_analysis(
  df,
  tte_name = "os_time",
  event_name = "os_event",
  treat_name = "treat",
  z_name = "biomarker",
  loghr_po_name = "loghr_po",
  theta1_name = "theta_1",
```

```
  theta0_name = "theta_0",
  spline_df = 3,
  alpha = 0.2,
  hr_threshold = 0.7,
  plot_style = c("combined", "separate", "grid"),
  plot_select = c("all", "profile_ahr", "ahr_only"),
  save_plots = FALSE,
  output_dir = tempdir(),
  verbose = TRUE
)
```

Arguments

df	Data frame containing survival data with potential outcomes.
tte_name	Character string specifying time-to-event variable name. Default: "os_time".
event_name	Character string specifying event indicator variable name. Default: "os_event".
treat_name	Character string specifying treatment variable name. Default: "treat".
z_name	Character string specifying continuous covariate/biomarker name. Default: "biomarker".
loghr_po_name	Character string specifying potential outcome log HR variable. Default: "loghr_po".
theta1_name	Optional: variable name for theta_1 (treated potential outcome). Default: "theta_1".
theta0_name	Optional: variable name for theta_0 (control potential outcome). Default: "theta_0".
spline_df	Integer degrees of freedom for spline fitting. Default: 3.
alpha	Numeric significance level for confidence intervals. Default: 0.20.
hr_threshold	Numeric hazard ratio threshold for subgroup identification, or NULL to suppress threshold-based elements. When NULL, plots omit the HR threshold line, optimal cutpoint line, and subgroup-dependent panels (Plots 5–6); subgroup statistics report only the overall population. Default: 0.7.
plot_style	Character: "combined", "separate", or "grid" for plot layout. Default: "combined".
plot_select	Character controlling which panels to display: "all" (default) shows the full grid layout; "profile_ahr" shows only the treatment effect profile (top-left) and the AHR curve for $z \geq$ threshold (top-middle) side by side, with the AHR panel's y-axis scaled to the data range; "ahr_only" shows only the AHR curve for $z \geq$ threshold as a single panel with self-contained y-axis.
save_plots	Logical whether to save plots to file. Default: FALSE.
output_dir	Character directory for saving plots. Default: tempdir().
verbose	Logical for diagnostic output. Default: TRUE.

Value

List of class "cox_ahr_cde" containing:

cox_fit Results from cox_cs_fit function.

ahr_results AHR calculations for different subgroup definitions.

cde_results CDE calculations if theta variables available.

optimal_cutpoint Optimal biomarker cutpoint, or NULL when hr_threshold is NULL.

subgroup_stats Statistics for recommended and questionable subgroups, or overall-only when hr_threshold is NULL.

data List with z_values, loghr_po, and subgroup assignments.

Examples

```
# Build a small synthetic dataset with required columns
set.seed(42)
n <- 200
df_ex <- data.frame(
  os_time   = rexp(n, rate = 0.01),
  os_event  = rbinom(n, 1, 0.6),
  treat     = rep(0:1, each = n / 2),
  biomarker = rnorm(n),
  loghr_po  = rnorm(n, mean = -0.3, sd = 0.5)
)

# With threshold - full subgroup analysis
results <- cox_ahr_cde_analysis(
  df = df_ex, z_name = "biomarker",
  hr_threshold = 1.25, plot_style = "grid",
  verbose = FALSE
)

# Without threshold - pure AHR curves
results <- cox_ahr_cde_analysis(
  df = df_ex, z_name = "biomarker",
  hr_threshold = NULL, plot_select = "ahr_only",
  verbose = FALSE
)
```

---

cox_cs_fit *Fit Cox Model with Cubic Spline for Treatment Effect Heterogeneity*

Description

Estimates treatment effects as a function of a continuous covariate using a Cox proportional hazards model with natural cubic splines. The function models treatment-by-covariate interactions to detect effect modification.

Usage

```
cox_cs_fit(
  df,
  tte_name = "os_time",
  event_name = "os_event",
```

```
  treat_name = "treat",
  strata_name = NULL,
  z_name = "bm",
  alpha = 0.2,
  spline_df = 3,
  z_max = Inf,
  z_by = 1,
  z_window = 0,
  z_quantile = 0.9,
  show_plot = TRUE,
  plot_params = NULL,
  truebeta_name = NULL,
  verbose = TRUE
)
```

Arguments

df	Data frame containing survival data
tte_name	Character string specifying time-to-event variable name. Default: "os_time"
event_name	Character string specifying event indicator variable name (1=event, 0=censored). Default: "os_event"
treat_name	Character string specifying treatment variable name (1=treated, 0=control). Default: "treat"
strata_name	Character string specifying stratification variable name. If NULL, no stratification is used. Default: NULL
z_name	Character string specifying continuous covariate name for effect modification. Default: "bm"
alpha	Numeric value for confidence level (two-sided). Default: 0.20 (80% confidence intervals)
spline_df	Integer specifying degrees of freedom for natural spline. Default: 3
z_max	Numeric maximum value for z in predictions. Values beyond this are truncated. Default: Inf (no truncation)
z_by	Numeric increment for z values in prediction grid. Default: 1
z_window	Numeric half-width for counting observations near each z value. Default: 0.0 (exact matches only)
z_quantile	Numeric quantile (0-1) for upper limit of z profile. Default: 0.90 (90th percentile)
show_plot	Logical indicating whether to display plot. Default: TRUE
plot_params	List of plotting parameters (see Details). Default: NULL
truebeta_name	Character string specifying variable containing true log(HR) values for validation/simulation. Default: NULL
verbose	Logical indicating whether to print diagnostic information. Default: TRUE

Details

Model Structure:

The function fits:

$$h(t|Z, A) = h_0(t) \exp(\beta_0 A + f(Z) + g(Z) \cdot A)$$

Where:

- A is treatment (0/1)
- Z is the continuous effect modifier
- f(Z) is modeled with natural splines (main effect)
- g(Z) is modeled with natural splines (interaction)
- The log hazard ratio is: $\beta(Z) = \beta_0 + g(Z)$

Plot Parameters:

The `plot_params` argument accepts a list with:

- `xlab`: x-axis label
- `main_title`: plot title
- `ylimit`: y-axis limits c(min, max)
- `y_pad_zero`: padding below zero line
- `y_delta`: extra space for count labels
- `cex_legend`: legend text size
- `cex_count`: count text size
- `show_cox_primary`: show standard Cox estimate line
- `show_null`: show null effect line (log(HR)=0)
- `show_target`: show target effect line (e.g., log(0.80))

Value

List containing:

z_profile Vector of z values where treatment effect is estimated

loghr_est Point estimates of log(HR) at each z value

loghr_lower Lower confidence bound

loghr_upper Upper confidence bound

se_loghr Standard errors of log(HR) estimates

counts_profile Number of observations near each z value

cox_primary Log(HR) from standard Cox model (no interaction)

model_fit The fitted coxph model object

spline_basis The natural spline basis object

Examples

```
# Simulate data
set.seed(123)
df <- data.frame(
  os_time = rexp(500, 0.01),
  os_event = rbinom(500, 1, 0.7),
  treat = rbinom(500, 1, 0.5),
  bm = rnorm(500, 50, 10)
)

# Fit model
result <- cox_cs_fit(df, z_name = "bm", alpha = 0.20)

# Custom plotting
result <- cox_cs_fit(
  df,
  z_name = "bm",
  plot_params = list(
    xlab = "Biomarker Level",
    main_title = "Treatment Effect by Biomarker",
    cex_legend = 1.2
  )
)
```

`cox_summary` *Cox model summary for subgroup (OPTIMIZED)*

Description

Called in analyze_subgroup() <– SG_tab_estimates

Usage

```
cox_summary(
  Y,
  E,
  Treat,
  Strata = NULL,
  use_strata = !is.null(Strata),
  return_format = c("formatted", "numeric")
)
```

Arguments

`Y`	Numeric vector of outcome.
`E`	Numeric vector of event indicators.

Treat	Numeric vector of treatment indicators.
Strata	Vector of strata (optional).
use_strata	Logical. Whether to use strata in the model (default: TRUE if Strata provided).
return_format	Character. "formatted" (default) or "numeric" for downstream use.

Details

Calculates hazard ratio and confidence interval for a subgroup using Cox regression. Optimized version with reduced overhead and better error handling.

Value

Character string with formatted HR and CI (or numeric vector if return_format="numeric").

Examples

```
library(survival)
cox_summary(
  Y     = gbsg$rfstime / 30.4375,
  E     = gbsg$status,
  Treat = gbsg$hormon
)
```

---

create_dgm_for_mrct *Create Data Generating Mechanism for MRCT Simulations*

Description

Wrapper function to create a data generating mechanism (DGM) for MRCT simulation scenarios using generate_aft_dgm_flex.

Usage

```
create_dgm_for_mrct(
  df_case,
  model_type = c("alt", "null"),
  log_hrs = NULL,
  confounder_var = NULL,
  confounder_effect = NULL,
  include_regA = TRUE,
  verbose = FALSE
)
```

Arguments

df_case	Data frame containing case study data
model_type	Character. Either "alt" (alternative hypothesis with heterogeneous treatment effects) or "null" (uniform treatment effect)
log_hrs	Numeric vector. Log hazard ratios for spline specification. If NULL, defaults are used based on model_type
confounder_var	Character. Name of a confounder variable to include with a forced prognostic effect. Default: NULL (no forced effect)
confounder_effect	Numeric. Log hazard ratio for confounder_var effect. Only used if confounder_var is specified
include_regA	Logical. Include regA as a factor in the model. Default: TRUE
verbose	Logical. Print detailed output. Default: FALSE

Details

Model Types:

alt Alternative hypothesis: Treatment effect varies by biomarker level (heterogeneous treatment effect). Default log_hrs create HR ranging from 2.0 (harm) to 0.5 (benefit) across biomarker range

null Null hypothesis: Uniform treatment effect regardless of biomarker level. Default log_hrs = log(0.7) uniformly

Confounder Effects:

By default, NO prognostic confounder effect is forced. The confounder_var and confounder_effect parameters allow optionally specifying ANY baseline covariate to have a fixed prognostic effect in the outcome model.

The regA variable (region indicator) is included as a factor by default but without a forced effect - its coefficient is estimated from data.

Value

An object of class "aft_dgm_flex" for use with `simulate_from_dgm` and `mrct_region_sims`

See Also

`generate_aft_dgm_flex` for underlying DGM creation `mrct_region_sims` for running simulations with the DGM

`create_forest_theme` *Create Forest Plot Theme with Size Controls*

Description

Creates a forestploter theme with parameters that control overall plot sizing and appearance. This is the primary way to control how large the forest plot renders.

Usage

```
create_forest_theme(
  base_size = 10,
  scale = 1,
  row_padding = NULL,
  ci_pch = 15,
  ci_lwd = NULL,
  ci_Theight = NULL,
  ci_col = "black",
  header_fontsize = NULL,
  body_fontsize = NULL,
  footnote_fontsize = NULL,
  footnote_col = "darkcyan",
  title_fontsize = NULL,
  cv_fontsize = NULL,
  cv_col = "gray30",
  refline_lwd = NULL,
  refline_lty = "dashed",
  refline_col = "gray30",
  vertline_lwd = NULL,
  vertline_lty = "dashed",
  vertline_col = "gray20",
  arrow_type = "closed",
  arrow_col = "black",
  summary_fill = "black",
  summary_col = "black"
)
```

Arguments

`base_size`	Numeric. Base font size in points. This is the primary scaling parameter - increasing it will proportionally scale all fonts, row padding, and line widths. Default: 10.
`scale`	Numeric. Additional scaling multiplier applied on top of base_size. Use for quick overall scaling. Default: 1.0.
`row_padding`	Numeric vector of length 2. Padding around row content in mm as c(vertical, horizontal). If NULL, auto-calculated from base_size. Default: NULL.

ci_pch	Integer. Point character for CI. 15=square, 16=circle, 18=diamond. Default: 15.
ci_lwd	Numeric. Line width for CI lines. If NULL, auto-calculated from base_size. Default: NULL.
ci_Theight	Numeric. Height of T-bar ends on CI. If NULL, auto-calculated from base_size. Default: NULL.
ci_col	Character. Color for CI lines and points. Default: "black".
header_fontsize	Numeric. Font size for column headers. If NULL, auto-calculated as base_size * scale + 1. Default: NULL.
body_fontsize	Numeric. Font size for body text. If NULL, auto-calculated as base_size * scale. Default: NULL.
footnote_fontsize	Numeric. Font size for footnotes. If NULL, auto-calculated as base_size * scale - 1. Default: NULL.
footnote_col	Character. Color for footnote text. Default: "darkcyan".
title_fontsize	Numeric. Font size for title. If NULL, auto-calculated as base_size * scale + 4. Default: NULL.
cv_fontsize	Numeric. Font size for CV annotation text. If NULL, auto-calculated as base_size * scale. Default: NULL.
cv_col	Character. Color for CV annotation text. Default: "gray30".
refline_lwd	Numeric. Reference line width. If NULL, auto-calculated. Default: NULL.
refline_lty	Character. Reference line type. Default: "dashed".
refline_col	Character. Reference line color. Default: "gray30".
vertline_lwd	Numeric. Vertical line width. If NULL, auto-calculated. Default: NULL.
vertline_lty	Character. Vertical line type. Default: "dashed".
vertline_col	Character. Vertical line color. Default: "gray20".
arrow_type	Character. Arrow type: "open" or "closed". Default: "closed".
arrow_col	Character. Arrow color. Default: "black".
summary_fill	Character. Fill color for summary diamonds. Default: "black".
summary_col	Character. Border color for summary diamonds. Default: "black".

Details

The base_size parameter is the primary way to control plot size. When you change base_size, the following are automatically scaled:

- All font sizes (body, header, footnote, CV, title)
- Row padding (vertical and horizontal)
- CI line width and T-bar height
- Reference and vertical line widths

The scaling formula uses base_size = 10 as the reference point:

- base_size = 10: Default sizing
- base_size = 12: 20% larger
- base_size = 14: 40% larger
- base_size = 16: 60% larger

You can override any individual parameter by specifying it explicitly.

The theme does NOT set row background colors - those are determined automatically by `plot_subgroup_results_forestpl` based on row types (ITT, reference, posthoc, etc.).

Value

A list of class "fs_forest_theme" containing all theme parameters.

See Also

`plot_subgroup_results_forestplot`, `render_forestplot`

Examples

```
# Simple: just increase base_size for larger plot
large_theme <- create_forest_theme(base_size = 14)
print(large_theme)

# Or use scale for quick adjustment
large_theme <- create_forest_theme(base_size = 10, scale = 1.4)

# Fine-tune specific elements
custom_theme <- create_forest_theme(
  base_size = 14,
  cv_fontsize = 12,
  ci_lwd = 2.5
)
```

---

create_summary_table *Create Enhanced Summary Table for Baseline Characteristics*

Description

Generates a formatted summary table comparing baseline characteristics between treatment arms. Supports continuous, categorical, and binary variables with p-values, standardized mean differences (SMD), and missing data summaries.

Usage

```
create_summary_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,
  table_title = "Baseline Characteristics by Treatment Arm",
  table_subtitle = NULL,
  source_note = NULL,
  font_size = 12,
  header_font_size = 14,
  footnote_font_size = 10,
  use_alternating_rows = TRUE,
  stripe_color = "#f9f9f9",
  indent_size = 20,
  highlight_pval = 0.05,
  highlight_smd = 0.2,
  highlight_color = "#fff3cd",
  compact_mode = FALSE,
  column_width_var = 200,
  column_width_stats = 120,
  show_column_borders = FALSE,
  custom_css = NULL
)
```

Arguments

data	Data frame containing the analysis data
treat_var	Character. Name of treatment variable (must have 2 levels)
vars_continuous	Character vector. Names of continuous variables
vars_categorical	Character vector. Names of categorical variables
vars_binary	Character vector. Names of binary (0/1) variables
var_labels	Named list. Custom labels for variables (optional)
digits	Integer. Number of decimal places for continuous variables
show_pvalue	Logical. Include p-values column
show_smd	Logical. Include SMD (effect size) column
show_missing	Logical. Include missing data rows
table_title	Character. Main title for the table

table_subtitle Character. Subtitle for the table (optional)

source_note Character. Source note at bottom (optional)

font_size Numeric. Base font size in pixels (default: 12)

header_font_size
Numeric. Header font size in pixels (default: 14)

footnote_font_size
Numeric. Footnote font size in pixels (default: 10)

use_alternating_rows
Logical. Apply zebra striping (default: TRUE)

stripe_color Character. Color for alternating rows (default: "#f9f9f9")

indent_size Numeric. Indentation for sub-levels in pixels (default: 20)

highlight_pval Numeric. Highlight p-values below this threshold (default: 0.05)

highlight_smd Numeric. Highlight SMD values above this threshold (default: 0.2)

highlight_color
Character. Color for highlighting (default: "#fff3cd")

compact_mode Logical. Reduce spacing for compact display (default: FALSE)

column_width_var
Numeric. Width for Variable column in pixels (default: 200)

column_width_stats
Numeric. Width for stat columns in pixels (default: 120)

show_column_borders
Logical. Show vertical column borders (default: FALSE)

custom_css Character. Additional custom CSS styling (optional)

Details

Binary variables specified via vars_binary display a single row showing the count and proportion for the "1" level. Categorical variables specified via vars_categorical that happen to be binary-coded (i.e., have exactly two levels: 0 and 1) are automatically detected and displayed in the same compact single-row format, showing only the "1" proportion.

Value

A gt table object (or data frame if gt not available)

---

cv_metrics_tables *Create Metrics Tables for Cross-Validation Results*

Description

Formats the find_summary and sens_summary outputs from forestsearch_tenfold or forestsearch_Kfold into publication-ready gt tables.

Usage

```
cv_metrics_tables(
  cv_result,
  sg_definition = NULL,
  title = "Cross-Validation Metrics",
  show_percentages = TRUE,
  digits = 1,
  include_raw = FALSE,
  table_style = c("combined", "separate", "minimal"),
  use_gt = TRUE
)
```

Arguments

cv_result	List. Result from forestsearch_tenfold() or forestsearch_Kfold(). Must contain find_summary and sens_summary elements.
sg_definition	Character vector. Subgroup factor definitions for labeling (optional). If NULL, extracted from cv_result$sg_analysis.
title	Character. Main title for combined table. Default: "Cross-Validation Metrics".
show_percentages	Logical. Display metrics as percentages (0-100) instead of proportions (0-1). Default: TRUE.
digits	Integer. Decimal places for formatting. Default: 1.
include_raw	Logical. Include raw matrices (sens_out, find_out) in the output for detailed analysis. Default: FALSE.
table_style	Character. One of "combined", "separate", or "minimal". • "combined": Single table with both agreement and finding metrics • "separate": Two separate gt tables • "minimal": Compact single-row summary Default: "combined".
use_gt	Logical. Return gt table(s) if TRUE, data.frame(s) if FALSE. Default: TRUE.

Value

Depending on table_style:

- "combined": A single gt table (or data.frame)
- "separate": A list with agreement_table and finding_table
- "minimal": A single-row gt table (or data.frame)

If include_raw = TRUE, also includes sens_out and find_out matrices in the returned list.

See Also

cv_summary_tables for formatting forestsearch_KfoldOut(outall=TRUE) results

cv_summary_tables *Create Summary Tables from forestsearch_KfoldOut Results*

Description

Formats the detailed output from `forestsearch_KfoldOut(outall=TRUE)` into publication-ready gt tables. This includes ITT estimates, original subgroup estimates, and K-fold subgroup estimates.

Usage

```
cv_summary_tables(
  kfold_out,
  title = "Cross-Validation Summary",
  subtitle = NULL,
  show_metrics = TRUE,
  digits = 3,
  font_size = 12,
  use_gt = TRUE
)
```

Arguments

kfold_out	List. Result from `forestsearch_KfoldOut(res, outall = TRUE)`. Must contain `itt_tab`, `SG_tab_original`, `SG_tab_Kfold`, and optionally `tab_all`.
title	Character. Main title for combined table. Default: "Cross-Validation Summary".
subtitle	Character. Subtitle for table. Default: NULL (auto-generated).
show_metrics	Logical. Include agreement and finding metrics in output. Default: TRUE.
digits	Integer. Decimal places for numeric formatting. Default: 3.
font_size	Integer. Font size in pixels. Default: 12.
use_gt	Logical. Return gt table if TRUE, data.frame if FALSE. Default: TRUE.

Value

If `use_gt = TRUE`, returns a list with gt table objects:

- `combined_table`: Combined ITT and subgroup estimates
- `itt_table`: ITT estimates only
- `original_table`: Original full-data subgroup estimates
- `kfold_table`: K-fold subgroup estimates
- `metrics_table`: Agreement and finding metrics (if `show_metrics = TRUE`)

If `use_gt = FALSE`, returns equivalent data.frames.

See Also

`cv_metrics_tables` for formatting `forestsearch_tenfold()` results

figure_note *Generate Figure Note for Quarto/RMarkdown*

Description

Formats the figure note from plot_sg_weighted_km() output for use in Quarto or RMarkdown documents.

Usage

```
figure_note(
  x,
  prefix = "*Note*: ",
  include_definition = TRUE,
  include_hr_explanation = TRUE,
  custom_text = NULL
)
```

Arguments

x	Output from plot_sg_weighted_km()
prefix	Character. Prefix for the note. Default uses italic Note.
include_definition	Logical. Include subgroup definition. Default: TRUE
include_hr_explanation	Logical. Include HR(bc) explanation. Default: TRUE
custom_text	Character. Additional custom text to append. Default: NULL

Value

Character string formatted as a figure note, or NULL if no content

forestsearch *ForestSearch: Exploratory Subgroup Identification*

Description

Identifies subgroups with differential treatment effects in clinical trials using a combination of Generalized Random Forests (GRF), LASSO variable selection, and exhaustive combinatorial search with split-sample validation.

Usage

```
forestsearch(
  df.analysis,
  outcome.name = "tte",
  event.name = "event",
  treat.name = "treat",
  id.name = "id",
  potentialOutcome.name = NULL,
  flag_harm.name = NULL,
  confounders.name = NULL,
  parallel_args = list(plan = "callr", workers = 6, show_message = TRUE),
  df.predict = NULL,
  df.test = NULL,
  is.RCT = TRUE,
  seedit = 8316951,
  est.scale = "hr",
  use_lasso = TRUE,
  use_grf = TRUE,
  grf_res = NULL,
  grf_cuts = NULL,
  max_n_confounders = 1000,
  grf_depth = 2,
  dmin.grf = 12,
  frac.tau = 0.6,
  return_selected_cuts_only = TRUE,
  conf_force = NULL,
  defaultcut_names = NULL,
  cut_type = "default",
  exclude_cuts = NULL,
  replace_med_grf = FALSE,
  cont.cutoff = 4,
  conf.cont_medians = NULL,
  conf.cont_medians_force = NULL,
  n.min = 60,
  hr.threshold = 1.25,
  hr.consistency = 1,
  sg_focus = "hr",
  fs.splits = 1000,
  m1.threshold = Inf,
  pconsistency.threshold = 0.9,
  stop_threshold = 0.95,
  showten_subgroups = FALSE,
  d0.min = 12,
  d1.min = 12,
  max.minutes = 3,
  minp = 0.025,
  details = FALSE,
  maxk = 2,
```

```
  by.risk = 12,
  plot.sg = FALSE,
  plot.grf = FALSE,
  max_subgroups_search = 10,
  vi.grf.min = -0.2,
  use_twostage = TRUE,
  twostage_args = list()
)
```

Arguments

df.analysis	Data frame. Analysis dataset with required columns.
outcome.name	Character. Name of time-to-event outcome variable. Default "tte".
event.name	Character. Name of event indicator (1=event, 0=censored). Default "event".
treat.name	Character. Name of treatment variable (1=treatment, 0=control). Default "treat".
id.name	Character. Name of subject ID variable. Default "id".
potentialOutcome.name	Character. Name of potential outcome variable (optional).
flag_harm.name	Character. Name of true harm flag for simulations (optional).
confounders.name	Character vector. Names of candidate subgroup-defining variables.
parallel_args	List. Parallel processing configuration: **plan** Character. One of "multisession", "multicore", "callr", "sequential" **workers** Integer. Number of parallel workers **show_message** Logical. Show parallel setup messages
df.predict	Data frame. Prediction dataset (optional).
df.test	Data frame. Test dataset (optional).
is.RCT	Logical. Is this a randomized controlled trial? Default TRUE.
seedit	Integer. Random seed. Default 8316951.
est.scale	Character. Estimation scale ("hr" or "rmst"). Default "hr".
use_lasso	Logical. Use LASSO for variable selection. Default TRUE.
use_grf	Logical. Use GRF for variable importance. Default TRUE.
grf_res	GRF results object (optional, for reuse).
grf_cuts	List. Custom GRF cut points (optional).
max_n_confounders	Integer. Maximum confounders to consider. Default 1000.
grf_depth	Integer. GRF tree depth. Default 2.
dmin.grf	Integer. Minimum events for GRF. Default 12.
frac.tau	Numeric. Fraction of tau for RMST. Default 0.6.
return_selected_cuts_only	Logical. If TRUE (default), GRF returns only cuts from the tree depth that identified the selected subgroup meeting dmin.grf. If FALSE returns all cuts from all fitted trees (depths 1 through grf_depth). See grf.subg.harm.survival for details.

`conf_force` Character vector. Variables to force include (optional).

`defaultcut_names`
Character vector. Default cut variable names (optional).

`cut_type` Character. Cut type ("default" or "custom"). Default "default".

`exclude_cuts` Character vector. Variables to exclude from cutting (optional).

`replace_med_grf`
Logical. Replace median with GRF cuts. Default FALSE.

`cont.cutoff` Integer. Cutoff for continuous vs categorical. Default 4.

`conf.cont_medians`
Named numeric vector. Median values for continuous variables (optional).

`conf.cont_medians_force`
Named numeric vector. Forced median values (optional).

`n.min` Integer. Minimum subgroup size. Default 60.

`hr.threshold` Numeric. Minimum HR for candidate subgroups. Default 1.25.

`hr.consistency` Numeric. Minimum HR for consistency validation. Default 1.0.

`sg_focus` Character. Subgroup selection focus. One of "hr", "hrMaxSG", "maxSG", "hrMinSG", "minSG". Default "hr".

`fs.splits` Integer. Number of splits for consistency evaluation (or maximum splits when `use_twostage = TRUE`). Default 1000.

`m1.threshold` Numeric. Maximum median survival threshold. Default Inf.

`pconsistency.threshold`
Numeric. Minimum consistency proportion. Default 0.90.

`stop_threshold` Numeric in `[0, 1]` or `NULL`. Early stopping threshold for consistency evaluation. When a candidate subgroup's consistency probability (`Pcons`) meets or exceeds this threshold, evaluation stops early — remaining candidates are skipped. Set to `NULL` to disable early stopping and force evaluation of all candidates up to `max_subgroups_search`. Default `0.95`.

Note: Values > 1.0 are not permitted. To disable early stopping, use `stop_threshold = NULL`, not a value above 1.

Automatically reset to `NULL` (with a warning) when `sg_focus` is `"hrMaxSG"` or `"hrMinSG"`, as these compound criteria require comparing HR *and* size across all candidates.

`showten_subgroups`
Logical. Show top 10 subgroups. Default FALSE.

`d0.min` Integer. Minimum control arm events. Default 12.

`d1.min` Integer. Minimum treatment arm events. Default 12.

`max.minutes` Numeric. Maximum search time in minutes. Default 3.

`minp` Numeric. Minimum prevalence threshold. Default 0.025.

`details` Logical. Print progress details. Default FALSE.

`maxk` Integer. Maximum number of factors per subgroup. Default 2.

`by.risk` Integer. Risk table interval. Default 12.

`plot.sg` Logical. Plot subgroup survival curves. Default FALSE.

`plot.grf` Logical. Plot GRF results. Default FALSE.

`max_subgroups_search`
Integer. Maximum subgroups to evaluate. Default 10.

`vi.grf.min` Numeric. Minimum GRF variable importance. Default -0.2.

`use_twostage` Logical. Use two-stage sequential consistency algorithm for improved performance. Default FALSE for backward compatibility. When TRUE, `fs.splits` becomes the maximum number of splits, and early stopping is enabled. See Details.

`twostage_args` List. Parameters for two-stage algorithm (only used when `use_twostage = TRUE`):

n.splits.screen Integer. Splits for Stage 1 screening. Default 30.

screen.threshold Numeric. Consistency threshold for Stage 1. Default is automatically calculated to provide ~2.5 SE margin.

batch.size Integer. Splits per batch in Stage 2. Default 20.

conf.level Numeric. Confidence level for early stopping. Default 0.95.

min.valid.screen Integer. Minimum valid Stage 1 splits. Default 10.

Details

Algorithm Overview:

1. **Variable Selection**: GRF identifies variables with treatment effect heterogeneity; LASSO selects most predictive
2. **Subgroup Discovery**: Exhaustive search over factor combinations up to `maxk`
3. **Consistency Validation**: Split-sample validation ensures reproducibility
4. **Selection**: Choose subgroup based on `sg_focus` criterion

Two-Stage Consistency Algorithm: When `use_twostage = TRUE`, the consistency evaluation uses an optimized algorithm that can provide 3-10x speedup:

- **Stage 1**: Quick screening with `n.splits.screen` splits eliminates clearly non-viable candidates
- **Stage 2**: Sequential batched evaluation with early stopping for candidates passing Stage 1

The two-stage algorithm is recommended for:

- Exploratory analyses with many candidate subgroups
- Large `fs.splits` values (>200)
- Iterative model development

For final regulatory submissions, `use_twostage = FALSE` may be preferred for exact reproducibility.

Value

A list of class "forestsearch" containing:

grp.consistency Consistency evaluation results including:

- out_sg: Selected subgroup based on sg_focus
- sg_focus: Focus criterion used
- df_flag: Treatment recommendations
- algorithm: "twostage" or "fixed"
- n_candidates_evaluated: Number evaluated
- n_passed: Number passing threshold

find.grps Subgroup search results

confounders.candidate Candidate confounders considered

confounders.evaluated Confounders after variable selection

df.est Analysis data with treatment recommendations

df.predict Prediction data with recommendations (if provided)

df.test Test data with recommendations (if provided)

minutes_all Total computation time

grf_res GRF results object

sg_focus Subgroup focus criterion used

sg.harm Selected subgroup definition

grf_cuts GRF cut points used

prop_maxk Proportion of max combinations searched

max_sg_est Maximum subgroup HR estimate

grf_plot GRF plot object (if plot.grf = TRUE)

args_call_all All arguments for reproducibility

References

- FDA Guidance for Industry: Enrichment Strategies for Clinical Trials
- Athey & Imbens (2016). Recursive partitioning for heterogeneous causal effects. PNAS.
- Wager & Athey (2018). Estimation and inference of heterogeneous treatment effects using random forests. JASA.

See Also

`subgroup.consistency` for consistency evaluation details `forestsearch_bootstrap_dofuture` for bootstrap inference `forestsearch_Kfold` for cross-validation

Package website: https://larry-leon.github.io/forestsearch/

Source code: https://github.com/larry-leon/forestsearch

`forestsearch_bootstrap_dofuture` *ForestSearch Bootstrap with doFuture Parallelization*

Description

Orchestrates bootstrap analysis for ForestSearch using doFuture parallelization. Implements bias correction methods to adjust for optimism in subgroup selection.

Usage

```
forestsearch_bootstrap_dofuture(
  fs.est,
  nb_boots,
  seed = 8316951L,
  details = FALSE,
  show_three = FALSE,
  parallel_args = list()
)
```

Arguments

`fs.est`	List. ForestSearch results object from `forestsearch`. Must contain `df.est` (data frame) and `args_call_all` (list of arguments).
`nb_boots`	Integer. Number of bootstrap samples (recommend 500-1000).
`seed`	Integer. Random seed for reproducibility of bootstrap sample generation. Default `8316951L`. The value is passed to both `bootstrap_ystar` (which constructs the $n \times B$ bootstrap index matrix) and `bootstrap_results` (which reruns the ForestSearch algorithm on each replicate); both calls must use the same seed to ensure bootstrap index alignment. Set to `NULL` for a non-reproducible run.
`details`	Logical. If `TRUE`, prints detailed progress information. Default: `FALSE`.
`show_three`	Logical. If `TRUE`, shows verbose output for first 3 bootstrap iterations for debugging. Default: `FALSE`.
`parallel_args`	List. Parallelization configuration with elements: • `plan`: Character. One of "multisession", "multicore", "callr", or "sequential" • `workers`: Integer. Number of parallel workers • `show_message`: Logical. Show parallel setup messages If empty list, inherits settings from original forestsearch call.

Value

List with the following components:

results Data.table with bias-corrected estimates for each bootstrap iteration

SG_CIs List of confidence intervals for H and Hc (raw and bias-corrected)

FSsg_tab Formatted table of subgroup estimates

Ystar_mat Matrix (nb_boots x n) of bootstrap sample indicators

H_estimates Detailed estimates for subgroup H

Hc_estimates Detailed estimates for subgroup Hc

summary (If create_summary=TRUE) Enhanced summary with tables and diagnostics

Bias Correction Methods

Two bias correction approaches are implemented:

1. **Method 1 (Simple Optimism)**:

$$H_{adj1} = H_{obs} - (H^*_* - H^*_{obs})$$

   where H^*_* is the new subgroup HR on bootstrap data and H^*_{obs} is the new subgroup HR on original data.

2. **Method 2 (Double Bootstrap)**:

$$H_{adj2} = 2 \times H_{obs} - (H_* + H^*_* - H^*_{obs})$$

   where H_* is the original subgroup HR on bootstrap data.

Variable Naming Convention

- `H`: Original subgroup (harm/questionable, treat.recommend == 0)
- `Hc`: Complement subgroup (recommend, treat.recommend == 1)
- `_obs`: Estimate from original data
- `_star`: Estimate from bootstrap data
- `_biasadj_1`: Bias correction method 1
- `_biasadj_2`: Bias correction method 2

Performance

Typical runtime: 1-5 seconds per bootstrap iteration. For 1000 bootstraps with 6 workers, expect 3-10 minutes total. Memory usage scales with dataset size and number of workers.

Requirements

- Original `fs.est` must have identified a valid subgroup
- Requires packages: `data.table`, `foreach`, `doFuture`, `survival`
- For plots: requires `ggplot2`

See Also

forestsearch for initial subgroup identification bootstrap_results for the core bootstrap worker function build_cox_formula for Cox formula construction fit_cox_models for Cox model fitting

---

forestsearch_Kfold *ForestSearch K-Fold Cross-Validation*

Description

This function assesses the stability and reproducibility of ForestSearch subgroup identification through cross-validation. For each fold:

1. Train ForestSearch on (K-1) folds
2. Apply the identified subgroup to the held-out fold
3. Compare predictions to the original full-data analysis

Usage

```
forestsearch_Kfold(
  fs.est,
  Kfolds = nrow(fs.est$df.est),
  seedit = 8316951L,
  parallel_args = list(plan = "multisession", workers = 6, show_message = TRUE),
  sg0.name = "Not recommend",
  sg1.name = "Recommend",
  details = FALSE
)
```

Arguments

fs.est	List. ForestSearch results object from forestsearch. Must contain df.est (data frame) and args_call_all (list of arguments).
Kfolds	Integer. Number of folds (default: nrow(fs.est$df.est) for LOO).
seedit	Integer. Random seed for fold assignment (default: 8316951).
parallel_args	List. Parallelization configuration with elements: • plan: Character. One of "multisession", "multicore", "sequential" • workers: Integer. Number of parallel workers • show_message: Logical. Show parallel setup messages
sg0.name	Character. Label for subgroup 0 (default: "Not recommend").
sg1.name	Character. Label for subgroup 1 (default: "Recommend").
details	Logical. Print progress details (default: FALSE).

Details

Performs K-fold cross-validation for ForestSearch, evaluating subgroup identification and agreement between training and test sets.

Value

List with components:

resCV Data frame with CV predictions for each observation

cv_args Arguments used for CV ForestSearch calls

timing_minutes Execution time in minutes

prop_SG_found Percentage of folds where a subgroup was found

sg_analysis Original subgroup definition from full-data analysis

sg0.name, sg1.name Subgroup labels

Kfolds Number of folds used

sens_summary Named vector of sensitivity metrics (sens_H, sens_Hc, ppv_H, ppv_Hc)

find_summary Named vector of subgroup-finding metrics (Any, Exact, etc.)

Cross-Validation Types

- **Leave-One-Out (LOO)**: When `Kfolds = nrow(df)`, each observation is held out once. Most thorough but computationally intensive.
- **K-Fold**: When `Kfolds < nrow(df)`, data is split into K roughly equal folds. Good balance of bias-variance tradeoff.

Output Metrics

The returned `resCV` data frame contains:

- `treat.recommend`: Prediction from CV model
- `treat.recommend.original`: Prediction from full-data model
- `cvindex`: Fold assignment
- `sg1`, `sg2`: Subgroup definitions found in each fold

See Also

`forestsearch` for initial subgroup identification `forestsearch_KfoldOut` for summarizing CV results `forestsearch_tenfold` for repeated K-fold simulations

forestsearch_KfoldOut *ForestSearch K-Fold Cross-Validation Output Summary*

Description

Summarizes cross-validation results for ForestSearch, including subgroup agreement and performance metrics.

Usage

```
forestsearch_KfoldOut(res, details = FALSE, outall = FALSE)
```

Arguments

res	List. Result object from ForestSearch cross-validation, must contain elements: cv_args, sg_analysis, sg0.name, sg1.name, Kfolds, resCV.
details	Logical. Print details during execution (default: FALSE).
outall	Logical. If TRUE, returns all summary tables; if FALSE, returns only metrics (default: FALSE).

Value

If outall=FALSE, a list with sens_metrics_original and find_metrics. If outall=TRUE, a list with summary tables and metrics.

forestsearch_tenfold *ForestSearch Repeated K-Fold Cross-Validation*

Description

This function performs multiple independent K-fold cross-validations to assess the variability in subgroup identification. Each simulation:

1. Randomly shuffles the data
2. Performs K-fold CV
3. Records sensitivity and agreement metrics

Results are summarized across all simulations.

Usage

```
forestsearch_tenfold(
  fs.est,
  sims,
  Kfolds = 10,
  details = TRUE,
  seed = 8316951L,
  parallel_args = list(plan = "multisession", workers = 6, show_message = TRUE)
)
```

Arguments

fs.est	List. ForestSearch results object from forestsearch.
sims	Integer. Number of simulation repetitions.
Kfolds	Integer. Number of folds per simulation (default: 10).
details	Logical. Print progress details (default: TRUE).
seed	Integer. Base random seed for fold shuffling. Default 8316951L. Each simulation uses seed + 1000 * ksim for reproducibility.
parallel_args	List. Parallelization configuration.

Details

Runs repeated K-fold cross-validation simulations for ForestSearch and summarizes subgroup identification stability across repetitions.

Value

List with components:

sens_summary Named vector of median sensitivity metrics across simulations

find_summary Named vector of median subgroup-finding metrics

sens_out Matrix of sensitivity metrics (sims x metrics)

find_out Matrix of finding metrics (sims x metrics)

timing_minutes Total execution time

sims Number of simulations run

Kfolds Number of folds per simulation

Parallelization Strategy

Unlike the single K-fold function which parallelizes across folds, this function parallelizes across simulations for better efficiency when running many repetitions. Each simulation runs its K-fold CV sequentially.

See Also

forestsearch_Kfold for single K-fold CV forestsearch_KfoldOut for summarizing CV results

`format_oc_results` *Format Operating Characteristics Results as GT Table*

Description

Creates a formatted gt table from simulation operating characteristics results.

Usage

```
format_oc_results(
  results,
  analyses = NULL,
  metrics = "all",
  digits = 3,
  digits_hr = 3,
  title = "Operating Characteristics Summary",
  subtitle = NULL,
  use_gt = TRUE
)
```

Arguments

`results`	data.table or data.frame. Simulation results from `run_simulation_analysis` or combined results from multiple simulations.
`analyses`	Character vector. Analysis methods to include. Default: NULL (all analyses in results)
`metrics`	Character vector. Metrics to display. Options include: "detection", "classification", "hr_estimates", "ahr_estimates", "cde_estimates", "subgroup_size", "all". Default: "all"
`digits`	Integer. Decimal places for proportions. Default: 3
`digits_hr`	Integer. Decimal places for hazard ratios. Default: 3
`title`	Character. Table title. Default: "Operating Characteristics Summary"
`subtitle`	Character. Table subtitle. Default: NULL
`use_gt`	Logical. Return gt table if TRUE, data.frame if FALSE. Default: TRUE

Details

The function summarizes simulation results across multiple metrics:

- **Found**: Proportion of simulations finding a subgroup (any.H)
- **Classification**: Sen, spec, PPV, NPV
- **HR Estimates**: Mean Cox hazard ratios in true (H) and identified (H-hat) subgroups and their complements
- **AHR Estimates**: Mean average hazard ratios (from loghr_po) in true and identified subgroups

- **CDE Estimates**: Controlled direct effects (from theta_0/theta_1) in true and identified subgroups
- **Subgroup Size**: Average, min, max sizes

Column notation aligns with `build_estimation_table` and Leon et al. (2024): `H` = true (oracle) subgroup, `H-hat` = identified subgroup. The asterisk (`*`) is reserved for bootstrap bias-corrected estimates and is not used in this summary table.

Value

A gt table object (if use_gt = TRUE and gt package available) or data.frame

---

`generate_aft_dgm_flex` *Generate Synthetic Survival Data using AFT Model with Flexible Subgroups*

Description

Creates a data generating mechanism (DGM) for survival data using an Accelerated Failure Time (AFT) model with Weibull distribution. Supports flexible subgroup definitions and treatment-subgroup interactions.

Usage

```
generate_aft_dgm_flex(
  data,
  continuous_vars,
  factor_vars,
  continuous_vars_cens = NULL,
  factor_vars_cens = NULL,
  set_beta_spec = list(set_var = NULL, beta_var = NULL),
  outcome_var,
  event_var,
  treatment_var = NULL,
  subgroup_vars = NULL,
  subgroup_cuts = NULL,
  draw_treatment = FALSE,
  model = "alt",
  k_treat = 1,
  k_inter = 1,
  n_super = 5000,
  select_censoring = TRUE,
  cens_type = "weibull",
  cens_params = list(),
  seed = 8316951,
  verbose = TRUE,
  standardize = FALSE,
  spline_spec = NULL
)
```

Arguments

data	A data.frame containing the input dataset to base the simulation on
continuous_vars	Character vector of continuous variable names to be standardized and included as covariates
factor_vars	Character vector of factor/categorical variable names to be converted to dummy variables (largest value as reference)
continuous_vars_cens	Character vector of continuous variable names to be used for censoring model. If NULL, uses same as continuous_vars. Default NULL
factor_vars_cens	Character vector of factor variable names to be used for censoring model. If NULL, uses same as factor_vars. Default NULL
set_beta_spec	List with elements 'set_var' and 'beta_var' for manually setting specific beta coefficients. Default list(set_var = NULL, beta_var = NULL)
outcome_var	Character string specifying the name of the outcome/time variable
event_var	Character string specifying the name of the event/status variable (1 = event, 0 = censored)
treatment_var	Character string specifying the name of the treatment variable. If NULL, treatment will be randomly simulated with 50/50 allocation
subgroup_vars	Character vector of variable names defining the subgroup. Default is NULL (no subgroups)
subgroup_cuts	Named list of cutpoint specifications for subgroup variables. See Details section for flexible specification options
draw_treatment	Logical indicating whether to redraw treatment assignment in simulation. Default is FALSE (use original assignments)
model	Character string: "alt" for alternative model with subgroup effects, "null" for null model without subgroup effects. Default is "alt"
k_treat	Numeric treatment effect modifier. Values >1 increase treatment effect, <1 decrease it. Default is 1 (no modification)
k_inter	Numeric interaction effect modifier for treatment-subgroup interaction. Default is 1 (no modification)
n_super	Integer specifying size of super population to generate. Default is 5000
select_censoring	Logical. If `TRUE` (default), fits the censoring distribution to the observed censoring times in `data` using `survreg` with AIC-based selection among Weibull and log-normal models (with and without covariates). If `FALSE`, no model is fitted; the censoring distribution is specified entirely by `cens_params`. Default `TRUE`.
cens_type	Character string specifying censoring distribution type: `"weibull"` or `"uniform"`. Controls which parametric family is considered when `select_censoring = TRUE`, and determines the required structure of `cens_params` when `select_censoring = FALSE`. Default `"weibull"`.

`cens_params` Named list of censoring distribution parameters. Interpretation depends on `select_censoring` and `cens_type`:

`select_censoring = TRUE` Ignored; all parameters are estimated from data.

`select_censoring = FALSE, cens_type = "uniform"` Must supply `min` and `max`. If either is absent, defaults to `0.5 * min(y)` and `1.5 * max(y)` with a message.

`select_censoring = FALSE, cens_type = "weibull"` Must supply `mu` (log-scale location) and `tau` (scale). Optionally supply `type` (`"weibull"` or `"lognormal"`); defaults to `"weibull"`. Censoring is treated as intercept-only (no covariate or treatment dependence): `lin_pred_cens_0 = lin_pred_cens_1 = mu`.

Default `list()`.

`seed` Integer random seed for reproducibility. Default is 8316951

`verbose` Logical indicating whether to print diagnostic information during execution. Default is TRUE

`standardize` Logical indicating whether to standardize continuous variables. Default is FALSE

`spline_spec` List specifying spline configuration for treatment effect. Must include 'var' (variable name), 'knot', 'zeta', and 'log_hrs' (vector of length 3). Default NULL (no spline)

Details

Subgroup Cutpoint Specifications:

The `subgroup_cuts` parameter accepts multiple flexible specifications:

Fixed Value:

```
subgroup_cuts = list(er = 20)  # er <= 20
```

Quantile-based:

```
subgroup_cuts = list(
  er = list(type = "quantile", value = 0.25)  # er <= 25th percentile
)
```

Function-based:

```
subgroup_cuts = list(
  er = list(type = "function", fun = median)  # er <= median
)
```

Range:

```
subgroup_cuts = list(
  age = list(type = "range", min = 40, max = 60)  # 40 <= age <= 60
)
```

Greater than:

```
subgroup_cuts = list(
  nodes = list(type = "greater", quantile = 0.75)  # nodes > 75th percentile
)
```

Multiple values (for categorical):

```
subgroup_cuts = list(
  grade = list(type = "multiple", values = c(2, 3))  # grade in (2, 3)
)
```

Custom function:

```
subgroup_cuts = list(
  er = list(
    type = "custom",
    fun = function(x) x <= quantile(x, 0.3) | x >= quantile(x, 0.9)
  )
)
```

Model Structure:

The AFT model with Weibull distribution is specified as:

$$\log(T) = \mu + \gamma' X + \sigma\epsilon$$

Where:

- T is the survival time
- μ is the intercept
- γ contains the covariate effects
- X includes treatment, covariates, and treatment x subgroup interaction
- σ is the scale parameter
- ϵ follows an extreme value distribution

Interaction Term:

The model creates a SINGLE interaction term representing the treatment effect modification when ALL subgroup conditions are simultaneously satisfied. This is not multiple separate interactions but one combined indicator.

Value

A named list of class `aft_dgm` containing:

`data`	Simulated trial data frame with outcome, event, and treatment columns.
`model_params`	Model parameters used for data generation (coefficients, dispersion, spline info if applicable).
`subgroup_info`	Subgroup definition and membership indicators, if a heterogeneous treatment effect was specified.
`censoring_info`	Censoring model parameters and observed censoring rate.
`call_args`	Arguments used in the call, for reproducibility.

Author(s)

Your Name

References

Leon, L.F., et al. (2024). Statistics in Medicine.

Kalbfleisch, J.D. and Prentice, R.L. (2002). The Statistical Analysis of Failure Time Data (2nd ed.). Wiley.

Examples

```
df <- survival::gbsg
dgm <- generate_aft_dgm_flex(
  data            = df,
  outcome_var     = "rfstime",
  event_var       = "status",
  treatment_var   = "hormon",
  continuous_vars = c("age", "size", "nodes", "pgr", "er"),
  factor_vars     = "meno",
  model           = "null",
  verbose         = FALSE
)
str(dgm)
```

---

generate_detection_curve

Generate Detection Probability Curve

Description

Computes detection probability across a range of hazard ratios to create a power-like curve for subgroup detection.

Usage

```
generate_detection_curve(
  theta_range = c(0.5, 3),
  n_points = 50L,
  n_sg,
  prop_cens = 0.3,
  hr_threshold = 1.25,
  hr_consistency = 1,
  include_reference = TRUE,
  method = "cubature",
  verbose = TRUE
)
```

Arguments

theta_range	Numeric vector of length 2. Range of HR values to evaluate. Default: c(0.5, 3.0)
n_points	Integer. Number of points to evaluate. Default: 50
n_sg	Integer. Subgroup sample size.
prop_cens	Numeric. Proportion censored (0-1). Default: 0.3
hr_threshold	Numeric. HR threshold for detection. Default: 1.25
hr_consistency	Numeric. HR consistency threshold. Default: 1.0
include_reference	Logical. Include reference HR values (0.5, 0.75, 1.0). Default: TRUE
method	Character. Integration method. Default: "cubature"
verbose	Logical. Print progress. Default: TRUE

Value

A data.frame with columns:

theta	Hazard ratio values
probability	Detection probability
n_sg	Subgroup size (repeated)
prop_cens	Censoring proportion (repeated)
hr_threshold	Detection threshold (repeated)

---

gg_forest *ggplot2 / patchwork forest plot*

Description

Creates a publication-quality forest plot using ggplot2 for the CI panel and patchwork to assemble label and annotation columns alongside it. Unlike forestploter, `fig.height` maps directly to row density — `row_height = fig.height / n_rows` with no hidden scaling.

Usage

```
gg_forest(
  subgroups,
  est,
  lo,
  hi,
  cat_vec = NULL,
  cat_colours = NULL,
  annot = NULL,
  ref_line = 1,
```

```
  vert_lines = NULL,
  ref_col = "firebrick",
  ref_lty = "dashed",
  vert_col = "grey50",
  vert_lty = "dotted",
  xlim = NULL,
  ticks_at = NULL,
  tick_labels = NULL,
  xlog = TRUE,
  xlab = "Hazard Ratio",
  title = NULL,
  subtitle = NULL,
  footnote = NULL,
  point_size = 2.5,
  line_size = 0.8,
  point_shape = 21,
  base_size = 11,
  widths = NULL,
  row_expand = 0.6
)
```

Arguments

subgroups	Character vector of subgroup names (displayed top to bottom).
est	Numeric vector of point estimates (median HR or similar).
lo	Numeric vector of lower bounds (e.g. 1st percentile ECI).
hi	Numeric vector of upper bounds (e.g. 99th percentile ECI).
cat_vec	Optional character vector of category labels (one per row). Used to colour CI lines and label text.
cat_colours	Optional named character vector mapping category labels to colours. Defaults to grey for all rows.
annot	Optional named list of character vectors, one per annotation column. Names become column headers. Each vector must match `length(subgroups)`.
ref_line	Numeric. X position of the primary reference line (default 1). Drawn as a dashed red line.
vert_lines	Numeric vector. X positions of secondary vertical lines (default NULL). Drawn as dotted grey lines.
ref_col	Colour of the primary reference line (default "firebrick").
ref_lty	Line type of the primary reference line (default "dashed").
vert_col	Colour of secondary vertical lines (default "grey50").
vert_lty	Line type of secondary vertical lines (default "dotted").
xlim	Numeric vector length 2. X-axis limits for the CI panel.
ticks_at	Numeric vector. X-axis tick positions.
tick_labels	Character vector. Custom tick labels (default: as.character(ticks_at)).

xlog	Logical. If TRUE (default), x-axis on log scale.
xlab	Character. X-axis label (default "Hazard Ratio").
title	Character. Overall plot title (default NULL).
subtitle	Character. Plot subtitle (default NULL).
footnote	Character. Footnote appended below the CI panel (default NULL).
point_size	Numeric. Size of point estimate symbol (default 2.5).
line_size	Numeric. Line width of CI segments (default 0.8).
point_shape	Integer. pch for point estimates (default 21, filled circle).
base_size	Numeric. ggplot2 base font size in pt (default 11). Controls all text — increase to make the plot larger; no other knob needed.
widths	Numeric vector. Relative patchwork column widths: c(label, ci, annot_1, annot_2, ...). Default: c(3.5, 5, rep(1, n_annot)).
row_expand	Numeric. Extra space above and below row range on y-axis, in row units (default 0.6).

Value

A patchwork object. Render with `print()` or `plot()`. Control dimensions entirely via knitr chunk options `fig.width` / `fig.height`: row height = fig.height / n_rows.

`grf.subg.harm.survival`

GRF Subgroup Identification for Survival Data

Description

Identifies subgroups with differential treatment effect using generalized random forests (GRF) and policy trees. This function uses causal survival forests to identify heterogeneous treatment effects and policy trees to create interpretable subgroup definitions.

Usage

```
grf.subg.harm.survival(
  data,
  confounders.name,
  outcome.name,
  event.name,
  id.name,
  treat.name,
  frac.tau = 1,
  n.min = 60,
  dmin.grf = 0,
  RCT = TRUE,
  details = FALSE,
```

```
  sg.criterion = "mDiff",
  maxdepth = 2,
  seedit = 8316951,
  return_selected_cuts_only = FALSE
)
```

Arguments

`data`	Data frame containing the analysis data.
`confounders.name`	Character vector of confounder variable names.
`outcome.name`	Character. Name of outcome variable (e.g., time-to-event).
`event.name`	Character. Name of event indicator variable (0/1).
`id.name`	Character. Name of ID variable.
`treat.name`	Character. Name of treatment group variable (0/1).
`frac.tau`	Numeric. Fraction of tau for GRF horizon (default: 1.0).
`n.min`	Integer. Minimum subgroup size (default: 60).
`dmin.grf`	Numeric. Minimum difference in subgroup mean (default: 0.0).
`RCT`	Logical. Is the data from a randomized controlled trial? (default: TRUE)
`details`	Logical. Print details during execution (default: FALSE).
`sg.criterion`	Character. Subgroup selection criterion ("mDiff" or "Nsg").
`maxdepth`	Integer. Maximum tree depth (1, 2, or 3; default: 2).
`seedit`	Integer. Random seed (default: 8316951).
`return_selected_cuts_only`	Logical. If TRUE, returns only cuts from the tree depth that identified the selected subgroup meeting `dmin.grf`. If FALSE (default), returns all cuts from all fitted trees (depths 1 through `maxdepth`).

Details

The `return_selected_cuts_only` parameter controls which cuts are returned:

FALSE (default) Returns all cuts from all fitted trees (depths 1 to `maxdepth`). This provides the full set of candidate splits for downstream exploration and is the original behavior for backward compatibility.

TRUE Returns only cuts from the tree at the depth that identified the "winning" subgroup meeting the `dmin.grf` criterion. This is useful when you want a focused set of cuts associated with the selected subgroup, reducing noise from non-selected trees.

When `return_selected_cuts_only = TRUE` and no subgroup meets the criteria, `tree.cuts` will be empty (character(0)).

Value

A list with GRF results, including:

`data`	Original data with added treatment recommendation flags
`grf.gsub`	Selected subgroup information
`sg.harm.id`	Expression defining the identified subgroup
`tree.cuts`	Cut expressions - either all cuts from all trees (if `return_selected_cuts_only = FALSE`) or only cuts from the selected tree depth (if `return_selected_cuts_only = TRUE`)
`tree.names`	Unique variable names used in cuts
`tree`	Selected policy tree object
`tau.rmst`	Time horizon used for RMST
`harm.any`	All subgroups with positive treatment effect difference
`selected_depth`	Depth of the tree that identified the subgroup (when found)
`return_selected_cuts_only`	Logical indicating which cut extraction mode was used

Additional tree-specific cuts and objects (tree1, tree2, tree3) based on maxdepth

---

`interpret_estimation_table` *Generate Narrative Interpretation of Estimation Properties*

Description

Produces a templated text summary of the estimation properties table, automatically populating numerical results from the simulation output. Useful for reproducible vignettes where interpretation paragraphs should update when simulations are re-run.

Usage

```
interpret_estimation_table(
  results,
  dgm,
  analysis_method = "FSlg",
  n_sims = NULL,
  n_boots = 300,
  digits = 2,
  scenario = NULL,
  cat = TRUE
)
```

Arguments

results	Data frame of simulation results (same as for build_estimation_table).
dgm	DGM object with true parameter values.
analysis_method	Character. Which analysis method to summarise. Default: "FSlg".
n_sims	Integer. Total number of simulations (for detection rate). If NULL (default), derived from nrow(results) after filtering to the analysis method.
n_boots	Integer. Number of bootstraps (for narrative). Default: 300.
digits	Integer. Decimal places for reported values. Default: 2.
scenario	Character. One of "null" or "alt" (default). Controls the interpretive framing: • "null": emphasises false-positive rate and selection bias • "alt": emphasises power, bias relative to true effect If NULL, inferred from the DGM (hr_H_true == hr_Hc_true implies null).
cat	Logical. If TRUE (default), prints the paragraph via cat(). If FALSE, returns it invisibly as a character string (useful for programmatic insertion into Rmd via results = "asis").

Value

Invisibly returns the interpretation as a character string.

See Also

build_estimation_table, format_oc_results, get_dgm_hr

---

mrct_region_sims *MRCT Regional Subgroup Simulation*

Description

Simulates multi-regional clinical trials and evaluates ForestSearch subgroup identification. Splits data by region into training and testing populations, identifies subgroups using ForestSearch on training data, and evaluates performance on the testing region.

Usage

```
mrct_region_sims(
  dgm,
  n_sims,
  n_sample = NULL,
  region_var = "z_regA",
  sg_focus = "minSG",
  maxk = 1,
  hr.threshold = 0.9,
```

```
  hr.consistency = 0.8,
  pconsistency.threshold = 0.9,
  confounders.name = NULL,
  conf_force = NULL,
  fs_args = list(),
  sim_args = list(rand_ratio = 1, draw_treatment = TRUE),
  analysis_time = 60,
  cens_adjust = 0,
 parallel_args = list(plan = "multisession", workers = NULL, show_message = TRUE),
  details = FALSE,
  verbose_n_sims = 2L,
  seed = NULL
)
```

Arguments

dgm	Data generating mechanism object from generate_aft_dgm_flex
n_sims	Integer. Number of simulations to run
n_sample	Integer. Sample size per simulation. If NULL (default), uses the entire superpopulation from dgm
region_var	Character. Name of the region indicator variable used to split data into training (region_var == 0) and testing (region_var == 1) populations. Default: "z_regA"
sg_focus	Character. Subgroup selection criterion passed to forestsearch: "minSG", "hr", or "maxSG". Default: "minSG"
maxk	Integer. Maximum number of factors in subgroup combinations (1 or 2). Default: 1
hr.threshold	Numeric. Hazard ratio threshold for subgroup identification. Default: 0.90
hr.consistency	Numeric. Consistency threshold for hazard ratio. Default: 0.80
pconsistency.threshold	Numeric. Probability threshold for consistency. Default: 0.90
confounders.name	Character vector. Confounder variable names for ForestSearch. If NULL, automatically extracted from dgm
conf_force	Character vector. Forced cuts to consider in ForestSearch. Default: c("z_age <= 65", "z_bm <= 0", "z_bm <= 1", "z_bm <= 2", "z_bm <= 5")
fs_args	Named list. Additional arguments passed directly to forestsearch inside each simulation replicate. Use this to control parameters not exposed by mrct_region_sims (e.g., use_grf, use_lasso, cut_type, d0.min, d1.min, n.min, max_subgroups_search, use_twostage, twostage_args). Parameters already in the mrct_region_sims signature (hr.threshold, hr.consistency, pconsistency.threshold, sg_focus, maxk, confounders.name, conf_force) take precedence over values in fs_args. Default: list() (uses forestsearch defaults)
sim_args	Named list. Additional arguments passed to simulate_from_dgm inside each replicate (e.g., rand_ratio, draw_treatment). Parameters already in the mrct_region_sims signature (analysis_time, cens_adjust) take precedence. Default: list(rand_ratio = 1, draw_treatment = TRUE)

analysis_time	Numeric. Time of analysis for administrative censoring. Default: 60
cens_adjust	Numeric. Adjustment factor for censoring rate on log scale. Default: 0
parallel_args	List. Parallel processing configuration with components: • plan: "multisession", "multicore", "callr", or "sequential" • workers: Number of workers (NULL for auto-detect) • show_message: Logical for progress messages
details	Logical. Print detailed progress information. Default: FALSE
verbose_n_sims	Integer. When details = TRUE, print full ForestSearch diagnostics (including internal output) for only the first verbose_n_sims simulation replicates. Set to 0 to suppress per-sim output, or Inf to print all. Default: 2
seed	Integer. Base random seed for reproducibility. Default: NULL

Details

Simulation Process:

For each simulation:

1. Sample from super-population using simulate_from_dgm
2. Split by region_var into training and testing populations
3. Estimate HRs in ITT, training, and testing populations
4. Run forestsearch on training population
5. Apply identified subgroup to testing population
6. Calculate subgroup-specific estimates

Region Variable:

The region_var parameter is used ONLY for splitting data into training/testing populations. It does not imply any prognostic effect. To include prognostic confounder effects, specify them when creating the DGM using create_dgm_for_mrct or generate_aft_dgm_flex.

Value

A data.table with simulation results containing:

sim Simulation index

n_itt ITT sample size

hr_itt ITT hazard ratio (stratified if strat variable present)

hr_ittX ITT hazard ratio stratified by region

n_train Training (non-region A) sample size

hr_train Training population hazard ratio

n_test Testing (region A) sample size

hr_test Testing population hazard ratio

any_found Indicator: 1 if subgroup identified, 0 otherwise

sg_found Character description of identified subgroup

n_sg Subgroup sample size

hr_sg Subgroup hazard ratio in testing population

POhr_sg Potential outcome hazard ratio in subgroup (testing)

prev_sg Subgroup prevalence (proportion of testing population)

n_sg_train Subgroup sample size in training population

hr_sg_train Subgroup hazard ratio in training population

POhr_sg_train Potential outcome hazard ratio in subgroup (training)

hr_sg_null Subgroup HR when found, NA otherwise

See Also

forestsearch for subgroup identification algorithm generate_aft_dgm_flex for DGM creation simulate_from_dgm for data simulation create_dgm_for_mrct for MRCT-specific DGM wrapper summaryout_mrct for summarizing simulation results

---

plot.forestsearch *Plot ForestSearch Results*

Description

Dispatches to plot_sg_results for Kaplan-Meier curves, hazard-ratio forest plots, or combined panels.

Usage

```
## S3 method for class 'forestsearch'
plot(
  x,
  type = c("combined", "km", "forest", "summary"),
  outcome.name = "Y",
  event.name = "Event",
  treat.name = "Treat",
  ...
)
```

Arguments

x	A forestsearch object returned by forestsearch.
type	Character. Type of plot:
	"combined" KM curves + forest plot (default)
	"km" Kaplan-Meier survival curves only
	"forest" Hazard-ratio forest plot only
	"summary" Summary statistics panel
outcome.name	Character. Name of time-to-event column. Default: "Y".
event.name	Character. Name of event indicator column. Default: "Event".

treat.name	Character. Name of treatment column. Default: "Treat".
...	Additional arguments passed to plot_sg_results, such as by.risk, conf.level, est.scale, sg0_name, sg1_name, treat_labels, colors, title, show_events, show_ci, show_logrank, show_hr.

Value

Invisibly returns the plot result from plot_sg_results.

See Also

plot_sg_results for full control over appearance, plot_sg_weighted_km for weighted KM curves, plot_subgroup_results_forestplot for publication-ready forest plots.

---

plot_detection_curve *Plot Detection Probability Curve*

Description

Creates a visualization of the detection probability curve.

Usage

```
plot_detection_curve(
  curve_data,
  add_reference_lines = TRUE,
  add_threshold_line = TRUE,
  title = NULL,
  ...
)
```

Arguments

curve_data	A data.frame from generate_detection_curve or with columns: theta, probability.
add_reference_lines	Logical. Add horizontal reference lines at 0.05, 0.10, 0.80. Default: TRUE
add_threshold_line	Logical. Add vertical line at hr_threshold. Default: TRUE
title	Character. Plot title. Default: auto-generated
...	Additional arguments passed to plot()

Value

Invisibly returns the input data.

`plot_km_band_forestsearch` *Plot Kaplan-Meier Survival Difference Bands for ForestSearch Subgroups*

Description

Creates Kaplan-Meier survival difference band plots comparing the identified ForestSearch subgroup (sg.harm) and its complement against the ITT population. This function wraps `plotKM.band_subgroups()` from the weightedsurv package, automatically extracting subgroup definitions from ForestSearch results.

Usage

```
plot_km_band_forestsearch(
  df,
  fs.est = NULL,
  sg_cols = NULL,
  sg_labels = NULL,
  sg_colors = NULL,
  itt_color = "azure3",
  outcome.name = "tte",
  event.name = "event",
  treat.name = "treat",
  xlabel = "Time",
  ylabel = "Survival differences",
  yseq_length = 5,
  draws_band = 1000,
  tau_add = NULL,
  by_risk = 6,
  risk_cex = 0.75,
  risk_delta = 0.035,
  risk_pad = 0.015,
  ymax_pad = 0.11,
  show_legend = TRUE,
  legend_pos = "topleft",
  legend_cex = 0.75,
  ref_subgroups = NULL,
  verbose = FALSE
)
```

Arguments

`df`	Data frame. The analysis dataset containing all required variables including subgroup indicator columns.
`fs.est`	A forestsearch object containing the identified subgroup, or `NULL` if using predefined subgroup indicators.

sg_cols	Character vector. Names of columns in df containing subgroup indicators (0/1). These columns must already exist in df. If NULL and fs.est is provided, columns will be created automatically. Default: NULL
sg_labels	Character vector. Subsetting expressions for each subgroup, corresponding to sg_cols. These are passed to plotKM.band_subgroups() which evaluates them as R expressions (e.g., "age < 65", "er <= 0", "Qrecommend == 1"). Must be same length as sg_cols. Default: NULL (auto-generated as "colname == 1").
sg_colors	Character vector. Colors for each subgroup curve, corresponding to sg_cols. Must be same length as sg_cols. Default: NULL (uses default color palette).
itt_color	Character. Color for ITT population band. Default: "azure3".
outcome.name	Character. Name of time-to-event column. Default: "tte".
event.name	Character. Name of event indicator column. Default: "event".
treat.name	Character. Name of treatment column. Default: "treat".
xlabel	Character. X-axis label. Default: "Time".
ylabel	Character. Y-axis label. Default: "Survival differences".
yseq_length	Integer. Number of y-axis tick marks. Default: 5.
draws_band	Integer. Number of bootstrap draws for confidence band. Default: 1000.
tau_add	Numeric. Time horizon for the plot. If NULL, auto-calculated from data. Default: NULL.
by_risk	Numeric. Interval for risk table. Default: 6.
risk_cex	Numeric. Character expansion for risk table text. Default: 0.75.
risk_delta	Numeric. Vertical spacing for risk table. Default: 0.035.
risk_pad	Numeric. Padding for risk table. Default: 0.015.
ymax_pad	Numeric. Y-axis maximum padding. Default: 0.11.
show_legend	Logical. Whether to display the legend. Default: TRUE.
legend_pos	Character. Legend position (e.g., "topleft", "bottomright"). Default: "topleft".
legend_cex	Numeric. Character expansion for legend text. Default: 0.75.
ref_subgroups	Named list. Optional additional reference subgroups to include. Each element should be a list with: **subset_expr** Character. R expression to define subgroup (e.g., "age < 65") **label** Character. Display label (optional, defaults to subset_expr) **color** Character. Color for the curve The function automatically creates indicator columns from the expressions. Default: NULL.
verbose	Logical. Print diagnostic messages. Default: FALSE.

Details

This function simplifies the workflow of creating KM survival difference band plots for ForestSearch-identified subgroups. It can work in two modes:

Mode 1: With ForestSearch result (fs.est provided)

1. Extracts the subgroup definition from the ForestSearch result
2. Creates binary indicator columns (Qrecommend, Brecommend) in `df`
3. Generates appropriate labels from the subgroup definition
4. Calls `plotKM.band_subgroups()` with configured parameters

Mode 2: With pre-defined columns (`sg_cols` provided)

1. Uses existing indicator columns in `df`
2. Requires `sg_labels` and `sg_colors` to match `sg_cols`

The sg.harm subgroup (Qrecommend) represents patients with questionable treatment benefit (where `treat.recommend == 0` in ForestSearch output). The complement (Brecommend) represents patients recommended for treatment.

Value

Invisibly returns a list containing:

df The modified data frame with subgroup indicators

sg_cols Character vector of subgroup column names used

sg_labels Character vector of subgroup labels used

sg_colors Character vector of colors used

sg_harm_definition The subgroup definition extracted from fs.est

ref_subgroups The reference subgroups list (if provided)

Subgroup Extraction

When `fs.est` is provided, the subgroup definition is extracted from:

- `fs.est$grp.consistency$out_sg$sg.harm_label` - Human-readable labels
- `fs.est$sg.harm` - Technical factor names (fallback)
- `fs.est$df.est$treat.recommend` - Subgroup membership indicator

Note

This function requires the **weightedsurv** package, which can be installed from GitHub: `devtools::install_github("larr`

See Also

`forestsearch` for running the subgroup analysis `plot_sg_weighted_km` for weighted KM plots `plot_sg_results` for comprehensive subgroup visualization

plot_sg_weighted_km *Plot Weighted Kaplan-Meier Curves for ForestSearch Subgroups*

Description

Creates weighted Kaplan-Meier survival curves for the identified subgroups (H and Hc) using the weightedsurv package, matching the pattern used in `sg_consistency_out()`.

Usage

```
plot_sg_weighted_km(
  fs.est,
  fs_bc = NULL,
  outcome.name = "Y",
  event.name = "Event",
  treat.name = "Treat",
  by.risk = NULL,
  sg0_name = NULL,
  sg1_name = NULL,
  conf.int = TRUE,
  show.logrank = TRUE,
  show.cox = TRUE,
  show.cox.bc = TRUE,
  put.legend.lr = "topleft",
  ymax = 1.05,
  xmed.fraction = 0.65,
  hr_bc_position = "bottomright",
  hr_bc_cex = 0.725,
  title = NULL,
  verbose = FALSE
)
```

Arguments

`fs.est`	A forestsearch object containing `df.est` with `treat.recommend` column, or a data frame directly.
`fs_bc`	Optional. Bootstrap results from `forestsearch_bootstrap_dofuture()` containing bias-corrected HR estimates. If provided, bias-corrected HRs will be annotated on the plots.
`outcome.name`	Character. Name of time-to-event column. Default: `"Y"`
`event.name`	Character. Name of event indicator column. Default: `"Event"`
`treat.name`	Character. Name of treatment column. Default: `"Treat"`
`by.risk`	Numeric. Risk interval for plotting. Default: `NULL` (auto-calculated as max(outcome)/12)
`sg0_name`	Character. Label for H subgroup (treat.recommend == 0). Default: `NULL` (auto-extracted from forestsearch object as "H: {definition}" or "Questionable (H)" if not available)

sg1_name	Character. Label for Hc subgroup (treat.recommend == 1). Default: NULL (auto-generated as "Hc: NOT {definition}" or "Recommend (Hc)" if not available)
conf.int	Logical. Show confidence intervals. Default: TRUE
show.logrank	Logical. Show log-rank test. Default: TRUE
show.cox	Logical. Show unadjusted Cox HR from weightedsurv. Default: TRUE
show.cox.bc	Logical. Show bootstrap bias-corrected HR annotation (requires fs_bc). Default: TRUE
put.legend.lr	Character. Legend position. Default: "topleft"
ymax	Numeric. Max y-axis value. Default: 1.05
xmed.fraction	Numeric. Fraction for median lines. Default: 0.65
hr_bc_position	Character. Position for bias-corrected HR annotation. One of "bottomright", "bottomleft", "topright", "topleft". Default: "bottomright"
hr_bc_cex	Numeric. Character expansion factor for bias-corrected HR annotation text. Default: 0.725 (matches weightedsurv cox.cex default)
title	Character. Overall plot title. Default: NULL
verbose	Logical. Print diagnostic messages. Default: FALSE

Details

This function uses the exact same calling pattern as `plot_subgroup()` in the ForestSearch package. Column names are mapped internally to the standard names (Y, Event, Treat) expected by weightedsurv.

Subgroup definitions are automatically extracted from the forestsearch object if available:

- `fs$grp.consistency$out_sg$sg.harm_label` - Human-readable labels
- `fs$sg.harm` - Technical factor names (fallback)

HR display options controlled by `show.cox` and `show.cox.bc`:

- Both TRUE (default): Shows unadjusted HR from weightedsurv AND bias-corrected HR annotation
- `show.cox` = TRUE, `show.cox.bc` = FALSE: Shows only unadjusted HR
- `show.cox` = FALSE, `show.cox.bc` = TRUE: Shows only bias-corrected HR
- Both FALSE: Shows neither HR estimate

Value

Invisibly returns a list with subgroup data frames and counting data

plot_spline_treatment_effect
Plot Spline Treatment Effect Function

Description

Plot Spline Treatment Effect Function

Usage

```
plot_spline_treatment_effect(dgm_result, add_points = TRUE)
```

Arguments

dgm_result	Result object from generate_aft_dgm_flex with spline
add_points	Logical; add observed data points. Default TRUE

Value

No return value, called for side effects (produces a plot).

plot_subgroup_results_forestplot
Plot Subgroup Results Forest Plot

Description

Generates a comprehensive forest plot showing:

- ITT (Intent-to-Treat) population estimate
- Reference subgroups (e.g., by biomarker levels)
- Post-hoc identified subgroups with bias-corrected estimates
- Cross-validation agreement metrics as annotations

Usage

```
plot_subgroup_results_forestplot(
  fs_results,
  df_analysis,
  subgroup_list = NULL,
  outcome.name,
  event.name,
  treat.name,
  E.name = "Experimental",
```

```
  C.name = "Control",
  est.scale = "hr",
  xlog = TRUE,
  title_text = NULL,
  arrow_text = c("Favors Experimental", "Favors Control"),
 footnote_text = c("Eg 80% of training found SG: 70% of B (+) also B in CV testing"),
  xlim = c(0.25, 1.5),
  ticks_at = c(0.25, 0.7, 1, 1.5),
  show_cv_metrics = TRUE,
  cv_source = c("auto", "kfold", "oob", "both"),
  posthoc_colors = c("powderblue", "beige"),
  reference_colors = c("yellow", "powderblue"),
  ci_column_spaces = 20,
  conf.level = 0.95,
  theme = NULL
)
```

Arguments

fs_results	List. A list containing ForestSearch analysis results with elements: • fs.est: ForestSearch estimation object from forestsearch • fs_bc: Bootstrap bias-corrected results from forestsearch_bootstrap_dofuture • fs_kfold: K-fold cross-validation results from forestsearch_Kfold or forestsearch_tenfold (optional) • fs_OOB: Out-of-bag cross-validation results (optional, alternative to fs_kfold)
df_analysis	Data frame. The analysis dataset with outcome, event, and treatment variables.
subgroup_list	List. Named list of subgroup definitions to include in the plot. Each element should be a list with: • subset_expr: Character string for subsetting (e.g., "BM> 1") • name: Display name for the subgroup • type: Either "reference" for pre-specified or "posthoc" for identified
outcome.name	Character. Name of the survival time variable.
event.name	Character. Name of the event indicator variable.
treat.name	Character. Name of the treatment variable.
E.name	Character. Label for experimental arm (default: "Experimental").
C.name	Character. Label for control arm (default: "Control").
est.scale	Character. Estimate scale: "hr" or "1/hr" (default: "hr").
xlog	Logical. If TRUE (default), the x-axis is plotted on a logarithmic scale. This is standard for hazard ratio forest plots where equal distances represent equal relative effects.
title_text	Character. Plot title (default: NULL).
arrow_text	Character vector of length 2. Arrow labels for forest plot (default: c("Favors Experimental", "Favors Control")).

`footnote_text` Character vector. Footnote text for the plot explaining CV metrics (default provides CV interpretation guidance; set to NULL to omit).

`xlim` Numeric vector of length 2. X-axis limits (default: c(0.25, 1.5)).

`ticks_at` Numeric vector. X-axis tick positions (default: c(0.25, 0.70, 1.0, 1.5)).

`show_cv_metrics` Logical. Whether to show cross-validation metrics (default: TRUE if fs_kfold or fs_OOB available).

`cv_source` Character. Source for CV metrics: "auto" (default, uses both if available, otherwise whichever is present), "kfold" (use fs_kfold only), "oob" (use fs_OOB only), or "both" (explicitly use both fs_kfold and fs_OOB, with K-fold first then OOB).

`posthoc_colors` Character vector. Colors for post-hoc subgroup rows (default: c("powderblue", "beige")).

`reference_colors` Character vector. Colors for reference subgroup rows (default: c("yellow", "powderblue")).

`ci_column_spaces` Integer. Number of spaces for the CI plot column width. More spaces = wider CI column (default: 20).

`conf.level` Numeric. Confidence level for intervals (default: 0.95 for 95% CI). Used to calculate the z-multiplier as qnorm(1 - (1 - conf.level)/2).

`theme` An fs_forest_theme object from `create_forest_theme()`. Use this to control plot sizing (fonts, row height, CI appearance, CV annotation font size). Default: NULL (uses default theme).

Details

Creates a publication-ready forest plot displaying identified subgroups with hazard ratios, bias-corrected estimates, and cross-validation metrics. This wrapper integrates ForestSearch results with the forestploter package.

ForestSearch Labeling Convention:

ForestSearch identifies subgroups based on hazard ratio thresholds:

- `sg.harm`: Contains the definition of the "harm" or "questionable" subgroup (H)
- `treat.recommend == 0`: Patient is IN the harm subgroup (H)
- `treat.recommend == 1`: Patient is in the COMPLEMENT subgroup (Hc, typically benefit)

For `est.scale = "hr"` (searching for harm):

- H (treat.recommend=0): Subgroup defined by sg.harm with elevated HR (harm/questionable)
- Hc (treat.recommend=1): Complement of sg.harm (potential benefit)

For `est.scale = "1/hr"` (searching for benefit):

- Roles are reversed: H becomes the benefit group

Value

A list containing:

plot The forestploter grob object (can be rendered with plot())

data The data frame used for the forest plot

row_types Character vector of row types for styling reference

cv_metrics Cross-validation metrics text (if available)

See Also

forestsearch for running the subgroup analysis forestsearch_bootstrap_dofuture for bootstrap bias correction forestsearch_Kfold for cross-validation create_forest_theme for customizing plot appearance render_forestplot for rendering the plot

---

print.cox_ahr_cde *Print method for cox_ahr_cde objects*

Description

Print method for cox_ahr_cde objects

Usage

```
## S3 method for class 'cox_ahr_cde'
print(x, ...)
```

Arguments

x	A cox_ahr_cde object from cox_ahr_cde_analysis.
...	Additional arguments (not used).

Value

Invisibly returns the input object.

print.forestsearch *Print Method for forestsearch Objects*

Description

Displays a concise summary of ForestSearch results including the identified subgroup definition, consistency metrics, algorithm details, and computation time.

Usage

```
## S3 method for class 'forestsearch'
print(x, ...)
```

Arguments

x	A forestsearch object returned by forestsearch.
...	Additional arguments (currently unused).

Value

Invisibly returns x.

See Also

summary.forestsearch for detailed output, plot.forestsearch for visualization.

print.fs_kfold *Print Method for K-Fold CV Results*

Description

Print Method for K-Fold CV Results

Usage

```
## S3 method for class 'fs_kfold'
print(x, ...)
```

Arguments

x	An fs_kfold object
...	Additional arguments (ignored)

Value

Invisibly returns x.

print.fs_tenfold *Print Method for Repeated K-Fold CV Results*

Description

Print Method for Repeated K-Fold CV Results

Usage

```
## S3 method for class 'fs_tenfold'
print(x, ...)
```

Arguments

x	An fs_tenfold object
...	Additional arguments (ignored)

Value

Invisibly returns x.

print.gbsg_dgm *Print Method for gbsg_dgm Objects*

Description

Print Method for gbsg_dgm Objects

Usage

```
## S3 method for class 'gbsg_dgm'
print(x, ...)
```

Arguments

x	A gbsg_dgm object
...	Additional arguments (unused)

Value

Invisibly returns x.

Examples

```
dgm <- setup_gbsg_dgm(model = "alt", verbose = FALSE)
print(dgm)
```

render_forestplot *Render ForestSearch Forest Plot*

Description

Renders a forest plot from `plot_subgroup_results_forestplot()`.

Usage

```
render_forestplot(x, newpage = TRUE)
```

Arguments

x	An fs_forestplot object from `plot_subgroup_results_forestplot()`.
newpage	Logical. Call grid.newpage() before drawing. Default: TRUE.

Details

To control plot sizing, create a custom theme using `create_forest_theme()` and pass it to `plot_subgroup_results_fores`

```
my_theme <- create_forest_theme(base_size = 14, row_padding = c(6, 4))
result <- plot_subgroup_results_forestplot(..., theme = my_theme)
render_forestplot(result)
```

Value

Invisibly returns the grob object.

render_reference_table *Render Reference Simulation Table as gt*

Description

Converts a data frame of pre-computed reference simulation results (e.g., digitized from a published LaTeX table) into a styled `gt` table. This is useful for displaying published benchmark results alongside new simulation output within vignettes or reports.

Usage

```
render_reference_table(
  ref_df,
  title = "Reference Simulation Results",
  subtitle = NULL,
  bold_threshold = 0.05
)
```

Arguments

ref_df	data.frame. Must contain a Metric column and one column per analysis method, with a Scenario column for row grouping.
title	Character. Table title.
subtitle	Character. Table subtitle. Default: NULL.
bold_threshold	Numeric. Values in any(H) rows exceeding this threshold are shown in bold. Set NULL to disable. Default: 0.05.

Value

A gt table object.

Examples

```
ref <- data.frame(
  Scenario = "M1 Null: N=700",
  Metric   = "any(H)",
  FS       = 0.02,
  FSlg     = 0.03,
  GRF      = 0.25
)
render_reference_table(ref, title = "Reference Results")
```

---

run_simulation_analysis

Run One Simulation Replicate

Description

General replacement for the legacy run_simulation_analysis() that was coupled to simulate_from_gbsg_dgm() and GBSG-specific column names. This version calls simulate_from_dgm and accepts explicit column-name parameters, making it applicable to any DGM built with generate_aft_dgm_flex.

Usage

```
run_simulation_analysis(
  sim_id,
  dgm,
  n_sample,
  analysis_time = Inf,
  cens_adjust = 0,
  max_follow = NULL,
  muC_adj = NULL,
  confounders_base = c("v1", "v2", "v3", "v4", "v5", "v6", "v7"),
  n_add_noise = 0L,
```

```
  outcome_name = "y_sim",
  event_name = "event_sim",
  treat_name = "treat_sim",
  harm_col = "flag_harm",
  run_fs = TRUE,
  run_fs_grf = TRUE,
  run_grf = TRUE,
  fs_params = list(),
  grf_params = list(),
  cox_formula = NULL,
  cox_formula_adj = NULL,
  n_sims_total = NULL,
  seed_base = 8316951L,
  verbose = FALSE,
  verbose_n = NULL,
  debug = FALSE
)
```

Arguments

sim_id	Integer. Simulation replicate index (used as seed offset).
dgm	An "aft_dgm_flex" object from generate_aft_dgm_flex or setup_gbsg_dgm.
n_sample	Integer. Per-replicate sample size.
analysis_time	Numeric. Calendar time of analysis on the DGM time scale. Use Inf (default) for no administrative censoring — equivalent to the legacy max_follow = Inf.
cens_adjust	Numeric. Log-scale shift to censoring times passed to simulate_from_dgm(cens_adjust = ...). Replaces legacy muC_adj. Default 0.
max_follow	**Deprecated.** Use analysis_time instead. If supplied, its value is forwarded to analysis_time with a warning. Retained for backward compatibility with legacy scripts.
muC_adj	**Deprecated.** Use cens_adjust instead. If supplied, its value is forwarded to cens_adjust with a warning. Retained for backward compatibility with legacy scripts.
confounders_base	Character vector of base confounder names.
n_add_noise	Integer. Number of independent N(0,1) noise variables to append. Default 0L.
outcome_name	Name of the observed time column in simulated data. Default "y_sim".
event_name	Name of the event indicator column. Default "event_sim".
treat_name	Name of the treatment column. Default "treat_sim".
harm_col	Name of the true-subgroup indicator column. Default "flag_harm".
run_fs	Logical. Run ForestSearch (LASSO). Default TRUE.
run_fs_grf	Logical. Run ForestSearch (LASSO + GRF). Default TRUE.
run_grf	Logical. Run standalone GRF. Default TRUE.
fs_params	Named list of ForestSearch parameter overrides.

grf_params	Named list of GRF parameter overrides.
cox_formula	Optional Cox formula for unadjusted ITT.
cox_formula_adj	Optional adjusted Cox formula.
n_sims_total	Integer. Total simulations (for progress messages).
seed_base	Integer. Base seed; replicate seed = seed_base + sim_id. Default 8316951L.
verbose	Logical. Print progress messages. Default FALSE.
verbose_n	Integer. If set, only print for sim_id <= verbose_n. Default NULL.
debug	Logical. Print detailed debug output. Default FALSE.

Value

A data.table with one row per analysis method, containing subgroup size, HR, AHR, CDE, and classification metrics.

See Also

simulate_from_dgm, generate_aft_dgm_flex, setup_gbsg_dgm

---

select_best_subgroup *Select best subgroup based on criterion*

Description

Identifies the optimal subgroup according to the specified criterion

Usage

```
select_best_subgroup(values, sg.criterion, dmin.grf, n.max)
```

Arguments

values	Data frame. Node metrics from policy trees
sg.criterion	Character. "mDiff" for maximum difference, "Nsg" for largest size
dmin.grf	Numeric. Minimum difference threshold
n.max	Integer. Maximum allowed subgroup size (total sample size)

Value

Data frame row with best subgroup or NULL if none found

Examples

```
vals <- data.frame(diff = c(8.5, 6.2, 3.1), Nsg = c(120, 95, 80))
select_best_subgroup(values = vals, sg.criterion = "mDiff",
                     dmin.grf = 6, n.max = 500)
```

setup_gbsg_dgm *Set Up a GBSG-Based AFT Data Generating Mechanism*

Description

Creates a GBSG-based data generating mechanism that is fully compatible with simulate_from_dgm. This is the replacement for create_gbsg_dgm(): it accepts exactly the same arguments and produces the same numeric output, but returns an object of class "aft_dgm_flex" instead of "gbsg_dgm".

Usage

```
setup_gbsg_dgm(
  model = c("alt", "null"),
  k_treat = 1,
  k_inter = 1,
  k_z3 = 1,
  z1_quantile = 0.25,
  n_super = 5000L,
  cens_type = c("weibull", "uniform"),
  use_rand_params = FALSE,
  seed = 8316951L,
  verbose = FALSE
)
```

Arguments

model	Character. Either "alt" for alternative hypothesis with heterogeneous treatment effects, or "null" for uniform treatment effect. Default: "alt"
k_treat	Numeric. Treatment effect multiplier applied to the treatment coefficient from the fitted AFT model. Values > 1 strengthen the treatment effect. Default: 1
k_inter	Numeric. Interaction effect multiplier for the treatment-subgroup interaction (z1 * z3). Only used when model = "alt". Higher values create more heterogeneity between HR(H) and HR(Hc). Default: 1
k_z3	Numeric. Effect multiplier for the z3 (menopausal status) coefficient. Default: 1
z1_quantile	Numeric. Quantile threshold for z1 (estrogen receptor). Observations with ER <= quantile are coded as z1 = 1. Default: 0.25
n_super	Integer. Size of super-population for empirical HR estimation. Default: 5000
cens_type	Character. Censoring distribution type: "weibull" or "uniform". Default: "weibull"
use_rand_params	Logical. If TRUE, modifies confounder coefficients using estimates from randomized subset (meno == 0). Default: FALSE
seed	Integer. Random seed for super-population generation. Default: 8316951
verbose	Logical. Print diagnostic information. Default: FALSE

Details

Internally the function calls `create_gbsg_dgm()` and then:

1. Adds a `df_super` field with column names aligned to `simulate_from_dgm()` conventions (`lin_pred_1`, `lin_pred_0`, `lin_pred_cens_1`, `lin_pred_cens_0`, `flag_harm`).
2. Adds a `model_params$tau` field (= `model_params$sigma`) and a `model_params$censoring` sub-list.
3. Sets class to `c("aft_dgm_flex", "gbsg_dgm", "list")`.

The original `df_super_rand` field is kept so that `compute_dgm_cde()` and `print.gbsg_dgm` continue to work.

Value

An object of class `c("aft_dgm_flex", "gbsg_dgm", "list")` with all fields from `create_gbsg_dgm()` plus:

`df_super` Super-population data frame with `simulate_from_dgm()`-compatible column names.

`model_params$tau` Copy of `model_params$sigma`.

`model_params$censoring` Sub-list with `type`, `mu`, `tau` for the censoring model.

See Also

`create_gbsg_dgm`, `simulate_from_dgm`, `compute_dgm_cde`

Examples

```
dgm <- setup_gbsg_dgm(model = "alt", k_inter = 2, verbose = FALSE)
dgm <- compute_dgm_cde(dgm)
print(dgm)
sim <- simulate_from_dgm(dgm, n = 400, seed = 1)
```

---

sg_tables *Enhanced Subgroup Summary Tables (gt output)*

Description

Returns formatted summary tables for subgroups using the gt package, with search metadata and customizable decimal precision. Produces two tables: a treatment effect estimates table and an identified subgroups table, each with fully customizable titles and subtitles.

Usage

```
sg_tables(
  fs,
  which_df = "est",
  est_title = "Treatment Effect Estimates",
  est_caption = "Training data estimates",
  sg_title = "Identified Subgroups",
  sg_subtitle = NULL,
  potentialOutcome.name = NULL,
  hr_1a = NA,
  hr_0a = NA,
  ndecimals = 3,
  include_search_info = TRUE,
  font_size = 12
)
```

Arguments

`fs`	ForestSearch results object.
`which_df`	Character. Which data frame to use ("est" or "testing").
`est_title`	Character or NULL. Main title for the estimates table (default: "Treatment Effect Estimates"). Rendered as bold markdown. Set to NULL to suppress the title and display only `est_caption`.
`est_caption`	Character. Subtitle for the estimates table (default: "Training data estimates").
`sg_title`	Character or NULL. Main title for the identified subgroups table (default: "Identified Subgroups"). Rendered as bold markdown. Set to NULL to suppress the title and display only `sg_subtitle`.
`sg_subtitle`	Character or NULL. Subtitle for the identified subgroups table. When NULL (default), an informative subtitle is auto-generated from `maxk` (e.g., "Two-factor subgroups (maxk=2)").
`potentialOutcome.name`	Character. Name of potential outcome variable (optional).
`hr_1a`	Character. Adjusted HR for subgroup 1 (optional).
`hr_0a`	Character. Adjusted HR for subgroup 0 (optional).
`ndecimals`	Integer. Number of decimals for formatted numbers (default: 3).
`include_search_info`	Logical. Include search metadata table (default: TRUE).
`font_size`	Numeric. Font size in pixels for table text (default: 12).

Value

List with gt tables for estimates, subgroups, and optionally search info.

simulate_from_dgm *Simulate Survival Data from AFT Data Generating Mechanism*

Description

Generates simulated survival data from a previously created AFT data generating mechanism (DGM). Samples from the super population and generates survival times with specified censoring.

Usage

```
simulate_from_dgm(
  dgm,
  n = NULL,
  rand_ratio = 1,
  entry_var = NULL,
  max_entry = 24,
  analysis_time = 48,
  cens_adjust = 0,
  draw_treatment = TRUE,
  seed = NULL,
  strata_rand = NULL,
  hrz_crit = NULL,
  keep_rand = FALSE,
  time_eos = NULL
)
```

Arguments

dgm	An object of class "aft_dgm_flex" created by generate_aft_dgm_flex.
n	Integer specifying the sample size. If NULL (default), uses the entire super population without sampling.
rand_ratio	Numeric randomisation ratio (treatment:control). Default 1 (1:1 allocation).
entry_var	Character string naming an entry-time variable in the super population. If NULL, entry times are drawn as Uniform(0, max_entry). Default NULL.
max_entry	Numeric maximum entry time for staggered entry simulation. Only used when entry_var = NULL. Default 24.
analysis_time	Numeric calendar time of analysis. Follow-up is analysis_time - entry_time. Must be on the same time scale as the DGM (i.e. the same units as outcome_var passed to generate_aft_dgm_flex). Default 48.
cens_adjust	Numeric log-scale adjustment to censoring distribution. Positive values increase censoring times; negative values decrease them. Default 0 (no adjustment).
draw_treatment	Logical. If TRUE (default), reassigns treatment according to rand_ratio. If FALSE, retains original treatment assignments from the super population.
seed	Integer random seed. Default NULL.

strata_rand	Character string naming a column in the sampled data for within-stratum balanced treatment allocation. If NULL, marginal allocation is used. Default NULL.
hrz_crit	Numeric log-HR threshold. If supplied, a column hrz_flag is added marking subjects with lin_pred_1 - lin_pred_0 >= hrz_crit. Default NULL.
keep_rand	Logical. If TRUE, appends a rand_order column preserving the randomisation sequence. Default FALSE.
time_eos	Numeric secondary administrative censoring cutoff (end-of-study time on the DGM scale). Applied after follow_up censoring. Default NULL.

Details

Time-scale consistency:

All time parameters (analysis_time, max_entry, time_eos) must be expressed in the same units as outcome_var supplied to generate_aft_dgm_flex(). A common error is building the DGM on days (e.g. rfstime) and then passing analysis_time in months, which causes follow-up windows far shorter than the DGM event-time scale and produces universal administrative censoring (event_sim = 0 for all subjects).

Verify with: exp(dgm$model_params$mu) — the implied median event time should be plausible given your analysis_time.

n = NULL path:

When n = NULL the entire super population is used as-is, with no staggered entry and no administrative censoring (follow_up = Inf). Treatment assignments and linear predictors already stored in dgm$df_super are retained unchanged.

Censoring adjustment:

cens_adjust shifts the log-scale location parameter of the censoring distribution:

- cens_adjust = log(2) doubles expected censoring times.
- cens_adjust = log(0.5) halves expected censoring times.

Value

A data.frame with columns:

id Subject identifier.

treat Original treatment from super population.

treat_sim Simulated treatment assignment.

flag_harm Subgroup indicator (1 = all subgroup conditions met).

z_* Covariate values.

lin_pred_1, lin_pred_0 Counterfactual log-time linear predictors.

y_sim Observed survival time (min(T, C)).

event_sim Event indicator (1 = event, 0 = censored).

t_true Latent true survival time (pre-censoring).

c_time Effective censoring time (post admin-censoring).

hrz_flag (Optional) Individual harm-zone indicator.

rand_order (Optional) Randomisation sequence index.

See Also

generate_aft_dgm_flex, check_censoring_dgm

Examples

```
dgm <- setup_gbsg_dgm(model = "null", verbose = FALSE)
sim_data <- simulate_from_dgm(dgm, n = 200, seed = 42)
dim(sim_data)
head(sim_data[, c("y_sim", "event_sim", "treat_sim")])
```

---

subgroup.consistency *Evaluate Subgroup Consistency*

Description

Evaluates candidate subgroups using split-sample consistency validation. For each candidate, repeatedly splits the data and checks whether the treatment effect direction is consistent across splits.

Usage

```
subgroup.consistency(
  df,
  hr.subgroups,
  hr.threshold = 1,
  hr.consistency = 1,
  pconsistency.threshold = 0.9,
  m1.threshold = Inf,
  n.splits = 100,
  details = FALSE,
  by.risk = 12,
  plot.sg = FALSE,
  maxk = 7,
  Lsg,
  confs_labels,
  sg_focus = "hr",
  stop_Kgroups = 10,
  stop_threshold = NULL,
  showten_subgroups = FALSE,
  pconsistency.digits = 2,
  seed = 8316951,
  checking = FALSE,
  use_twostage = FALSE,
  twostage_args = list(),
  parallel_args = list()
)
```

Arguments

df	Data frame containing the analysis dataset. Must include columns for outcome (Y), event indicator (Event), and treatment (Treat).
hr.subgroups	Data.table of candidate subgroups from subgroup search, containing columns: HR, n, E, K, d0, d1, m0, m1, grp, and factor indicators.
hr.threshold	Numeric. Minimum hazard ratio threshold for candidates. Default: 1.0
hr.consistency	Numeric. Minimum HR required in each split for consistency. Default: 1.0
pconsistency.threshold	Numeric. Minimum proportion of splits that must be consistent. Default: 0.9
m1.threshold	Numeric. Maximum m1 threshold for filtering. Default: Inf
n.splits	Integer. Number of splits for consistency evaluation. Default: 100
details	Logical. Print progress details. Default: FALSE
by.risk	Numeric. Risk interval for KM plots. Default: 12
plot.sg	Logical. Generate subgroup plots. Default: FALSE
maxk	Integer. Maximum number of factors in subgroup. Default: 7
Lsg	List of subgroup parameters.
confs_labels	Character vector mapping factor names to labels.
sg_focus	Character. Subgroup selection criterion: "hr", "maxSG", or "minSG". Default: "hr"
stop_Kgroups	Integer. Maximum number of candidates to evaluate. Default: 10
stop_threshold	Numeric in [0, 1] or NULL. When a candidate subgroup's consistency probability (Pcons) meets or exceeds this threshold, evaluation stops early — remaining candidates are skipped. Set to NULL to disable early stopping and evaluate all candidates up to stop_Kgroups. Default: NULL. **Note:** Values > 1.0 are not permitted. To disable early stopping, use stop_threshold = NULL, not a value above 1. **Interaction with** sg_focus**:** "hr", "maxSG", "minSG" Early stopping is valid because candidates are sorted by a single criterion. The first candidate passing the threshold is optimal under that criterion. "hrMaxSG", "hrMinSG" Should generally be NULL, because these compound criteria require comparing HR *and* size across all candidates. forestsearch() automatically resets to NULL with a warning for these. For parallel execution, early stopping is checked after each batch completes, so some additional candidates beyond the first meeting the threshold may be evaluated. Use a smaller batch_size in parallel_args for finer-grained early stopping.
showten_subgroups	Logical. If TRUE, prints up to 10 candidate subgroups after sorting by sg_focus, showing their rank, HR, sample size, events, and factor definitions. Useful for reviewing which candidates will be evaluated for consistency. Default: FALSE

`pconsistency.digits`
: Integer. Decimal places for consistency proportion. Default: 2

`seed`
: Integer. Random seed for reproducible consistency splits. Default: 8316951. Set to NULL for non-reproducible random splits. The seed is used both for sequential execution (via set.seed()) and parallel execution (via future.seed).

`checking`
: Logical. Enable additional validation checks. Default: FALSE

`use_twostage`
: Logical. Use two-stage adaptive algorithm. Default: FALSE

`twostage_args`
: List. Parameters for two-stage algorithm:

  n.splits.screen Splits for Stage 1 screening. Default: 30

  screen.threshold Consistency threshold for Stage 1. Default: auto

  batch.size Splits per batch in Stage 2. Default: 20

  conf.level Confidence level for early stopping. Default: 0.95

  min.valid.screen Minimum valid Stage 1 splits. Default: 10

`parallel_args`
: List. Parallel processing configuration:

  plan Future plan: "multisession", "multicore", or "sequential"

  workers Number of parallel workers

  batch_size Number of candidates to evaluate per batch. Smaller values provide finer-grained early stopping but may increase overhead. Default: When stop_threshold is set and sg_focus is "hr" or "minSG", defaults to 1 (stop immediately when first candidate passes). For other sg_focus values with stop_threshold, defaults to min(workers, n_candidates/4). When stop_threshold is NULL, defaults to workers*2 for efficiency.

  show_message Print parallel config messages

Value

A list containing:

out_sg Selected subgroup results

sg_focus Selection criterion used

df_flag Data frame with treatment recommendations

sg.harm Subgroup definition labels

sg.harm.id Subgroup membership indicator

algorithm "twostage" or "fixed"

n_candidates_evaluated Number of candidates actually evaluated

n_candidates_total Total candidates available

n_passed Number meeting consistency threshold

early_stop_triggered Logical indicating if early stop occurred

early_stop_candidate Index of candidate triggering early stop

stop_threshold Threshold used for early stopping

seed Random seed used for reproducibility (NULL if not set)

`summarize_bootstrap_events` *Summarize Bootstrap Event Counts*

Description

Provides summary statistics for event counts across bootstrap iterations, helping assess the reliability of HR estimates when events are sparse.

Usage

```
summarize_bootstrap_events(boot_results, threshold = 5)
```

Arguments

`boot_results`	List. Output from forestsearch_bootstrap_dofuture()
`threshold`	Integer. Minimum event threshold for flagging low counts (default: 5)

Details

This function summarizes event counts in four scenarios:

1. ORIGINAL subgroup H evaluated on BOOTSTRAP samples
2. ORIGINAL subgroup Hc evaluated on BOOTSTRAP samples
3. NEW subgroup H* (found in bootstrap) evaluated on ORIGINAL data
4. NEW subgroup Hc* (found in bootstrap) evaluated on ORIGINAL data

Low event counts (below threshold) can lead to unstable HR estimates. This summary helps identify potential issues with sparse events.

Value

Invisibly returns a list with summary statistics:

threshold The event threshold used

nb_boots Total number of bootstrap iterations

n_successful Number of iterations that found a new subgroup

original_H List with low event counts for original H on bootstrap samples

original_Hc List with low event counts for original Hc on bootstrap samples

new_Hstar List with low event counts for new H* on original data

new_Hcstar List with low event counts for new Hc* on original data

summarize_bootstrap_results

Enhanced Bootstrap Results Summary

Description

Creates comprehensive output including formatted table with subgroup footnote, diagnostic plots, bootstrap quality metrics, and detailed timing analysis.

Usage

```
summarize_bootstrap_results(
  sgharm,
  boot_results,
  create_plots = FALSE,
  est.scale = "hr"
)
```

Arguments

sgharm	The selected subgroup object from forestsearch results. Can be: • Character vector of factor definitions (e.g., c("{age>=50}", "{nodes>=3}")) • List with sgharm element containing factor definitions • List with sg.harm_label element (human-readable labels)
boot_results	List. Output from forestsearch_bootstrap_dofuture()
create_plots	Logical. Generate diagnostic plots (default: FALSE)
est.scale	Character. "hr" or "1/hr" for effect scale

Details

The table output includes a footnote displaying the identified subgroup definition, analogous to the tab_estimates table from sg_tables. This is achieved by extracting the subgroup definition from sgharm and passing it to format_bootstrap_table.

Value

List with components:

table gt table with treatment effects and subgroup footnote

diagnostics List of bootstrap quality metrics

diagnostics_table_gt gt table of diagnostics

plots List of ggplot2 diagnostic plots (if create_plots=TRUE)

timing List of timing analysis (if timing data available)

subgroup_summary List from summarize_bootstrap_subgroups()

See Also

format_bootstrap_table for table creation sg_tables for analogous main analysis tables summarize_bootstrap_subgro for subgroup stability analysis

---

summarize_simulation_results *Summarize Simulation Results*

Description

Creates a summary table of operating characteristics across all simulations. Includes both HR and AHR metrics.

Usage

```
summarize_simulation_results(
  results,
  analyses = NULL,
  digits = 2,
  digits_hr = 3
)
```

Arguments

results	data.table with simulation results from run_simulation_analysis
analyses	Character vector. Analysis methods to include. Default: all
digits	Integer. Decimal places for proportions. Default: 2
digits_hr	Integer. Decimal places for hazard ratios. Default: 3

Value

Data frame with summary statistics

---

summary.cox_ahr_cde *Summary method for cox_ahr_cde objects*

Description

Summary method for cox_ahr_cde objects

Usage

```
## S3 method for class 'cox_ahr_cde'
summary(object, ...)
```

Arguments

object	A cox_ahr_cde object from cox_ahr_cde_analysis.
...	Additional arguments (not used).

Value

Invisibly returns the input object.

---

summary.forestsearch *Summary Method for forestsearch Objects*

Description

Provides a detailed summary of a ForestSearch analysis including input parameters, variable selection results, consistency evaluation, and the selected subgroup with key metrics.

Usage

```
## S3 method for class 'forestsearch'
summary(object, ...)
```

Arguments

object	A forestsearch object returned by forestsearch.
...	Additional arguments (currently unused).

Value

Invisibly returns object.

---

validate_k_inter_effect *Validate k_inter Effect on HR Heterogeneity*

Description

Test function to verify that k_inter properly modulates the difference between HR(H) and HR(Hc), and that AHR metrics align with Cox-based HRs.

Usage

```
validate_k_inter_effect(
  k_inter_values = c(-2, -1, 0, 1, 2, 3),
  verbose = TRUE,
  ...
)
```

Arguments

k_inter_values	Numeric vector of k_inter values to test. Default: c(-2, -1, 0, 1, 2, 3)
verbose	Logical. Print results. Default: TRUE
...	Additional arguments passed to create_gbsg_dgm

Value

Data frame with k_inter, hr_H, hr_Hc, AHR_H, AHR_Hc, and ratio columns

Examples

```
# Test k_inter effect
results <- validate_k_inter_effect()

# k_inter = 0 should give hr_H approximately equals hr_Hc (ratio approximately 1)
```

Index