Package: ballast (via r-universe)

November 1, 2026

Title Stress-Test a Pre-Specified Mixed Model for Repeated Measures

Version 0.7.4

Description Refits one pre-specified mixed model for repeated measures (MMRM) under families of data and analytic perturbations (covariance structure, degrees-of-freedom method and variance estimator, covariate set, analysis population, alternative analysis data sets, leave-one-out deletions, crossed specification grids, re-randomisation tests, delta-adjusted and reference-based multiple imputation, and alternative estimators) and judges every refit against the pre-specified estimate, confidence interval and p-value. Classifies the result as robust, sensitive or fragile, with a separate flag for marginal significance, and estimates by simulation the probability that a planned design gives a significant and robust result. Fitting uses 'mmrm' (Sabanes Bove et al., 2024, <https://openpharma.github.io/mmrm/>) with 'SAS' 'PROC MIXED' conventions for the least-squares-mean contrast, Kenward-Roger adjustment and the corrected Akaike criterion. Reference-based multiple imputation follows Carpenter, Roger and Kenward (2013) <doi:10.1080/10543406.2013.834911> and was checked against 'rbmi' (Gower-Page, Noci and Wolbers, 2022) <doi:10.21105/joss.04251>.

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports mmrm (>= 0.3.0), stats, utils, parallel

Suggests ggplot2, emmeans, nlme, MASS, flextable, officer, plotly, htmlwidgets, htmltools, jsonlite, testthat (>= 3.0.0), knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/Sullivanstatistics/ballast,
https://sullivanstatistics.github.io/ballast/

BugReports https://github.com/Sullivanstatistics/ballast/issues

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev

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

Date/Publication 2026-10-30 17:29:09 UTC

RemoteUrl https://github.com/sullivanstatistics/ballast

RemoteRef HEAD

RemoteSha 169b6a3ad6d522e7c4b47562ee6342b4c6e31322

Contents

default_inference . . . 3
mask_ids . . . 4
path_changes . . . 4
plot.stress_census . . . 6
plot.stress_design . . . 6
plot_forest . . . 7
plot_pcdf . . . 8
plot_pcurve . . . 9
plot_pstrip . . . 10
plot_swarm . . . 11
run_plan . . . 13
run_plans . . . 13
simulate_rare_trial . . . 14
simulate_trial . . . 16
spec_update . . . 18
standard_plan . . . 18
stress_calibrate . . . 20
stress_census . . . 21
stress_covariance . . . 22
stress_covariates . . . 23
stress_curve . . . 24
stress_data . . . 25
stress_design . . . 26
stress_direction . . . 28
stress_estimand . . . 29
stress_estimator . . . 31
stress_explain . . . 32
stress_fit . . . 33
stress_fragility . . . 34
stress_grid . . . 36
stress_html . . . 37

stress_inference . . . 38
stress_log . . . 38
stress_loo . . . 39
stress_path . . . 40
stress_permute . . . 41
stress_plan . . . 42
stress_population . . . 44
stress_ranktest . . . 44
stress_reference . . . 45
stress_report . . . 47
stress_sap . . . 48
stress_spec . . . 49
stress_spec_adam . . . 51
stress_spec_mmrm . . . 52
stress_tables . . . 53
stress_tipping . . . 54
stress_verdict . . . 55
stress_visits . . . 56
stress_weights . . . 57
summary.stress_census . . . 58
tipping_point . . . 58

Index **60**

`default_inference` *Default inference settings for* `stress_inference()`

Description

Default inference settings for `stress_inference()`

Usage

```
default_inference()
```

Value

A named list of `list(method = , vcov = )` pairs: Satterthwaite, Kenward-Roger, linear Kenward-Roger (the SAS `DDFM=KR` default), residual, sandwich with between-within degrees of freedom (SAS `EMPIRICAL`) and the jackknife sandwich.

Examples

```
names(default_inference())
```

mask_ids *Mask subject identifiers in a census*

Description

Replaces subject identifiers in the labels of the leave-one-subject-out and deletion-search rows with sequence numbers (`S001`, `S002`, ...) in the order of the spec's subject levels, so that a census, report or page can leave the organisation without participant identifiers. The mapping is returned as `attr(, "id_map")`.

Usage

```
mask_ids(census, spec)
```

Arguments

census	A `stress_census` (or `stress_rows`).
spec	The `stress_spec()` the census was built from.

Value

The same object with masked labels.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:40), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_loo(spec))
head(mask_ids(cen, spec)$label)
```

path_changes *Candidate analytic changes for the path search*

Description

Lists the analytic changes `stress_path()` may combine. Each change belongs to a family (covariance structure, inference, covariate set, LS-mean weighting, reference-based imputation, analysis data set), and a path changes each family at most once. Participant and site deletions are not analytic changes; they are measured separately by `stress_fragility()`.

Usage

```
path_changes(
  spec,
  covariance = c("us", "ar1h", "cs"),
  inference = default_inference()[c("Satterthwaite", "Kenward-Roger (linear)",
    "Sandwich, between-within")],
  covariates = TRUE,
  weights = TRUE,
  reference = c("J2R", "CR", "CIR"),
  delta_adjusted = numeric(),
  m = 10,
  data = list()
)
```

Arguments

spec	A stress_spec().
covariance	Alternative covariance structures.
inference	Named list of list(method = , vcov =) alternatives.
covariates	TRUE to drop each covariate and (with two or more) all of them; a named list of covariate sets to use instead; FALSE for none.
weights	TRUE to include the other LS-mean weighting.
reference	Reference-based imputation assumptions (any of "MAR", "J2R", "CR", "CIR"); character() for none.
delta_adjusted	Shifts for delta-adjusted imputation under MAR. Each value is one change that adds the shift to every imputed value of the active arm (for example minus half the residual SD, against the active arm). It belongs to the reference-based imputation family, so a path uses at most one missing-data assumption. numeric() for none.
m	Imputations per reference-based analysis.
data	Named list of alternative analysis data sets (for example, other intercurrent-event handlings).

Value

A list of changes, each a list with family, label, type ("spec" or "mi") and value.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + FEV1_BL)
vapply(path_changes(spec, reference = character()), `[[`, "", "label")
```

plot.stress_census *Plot a census: one row per family on the percent-of-base axis*

Description

Each counted analysis is a point at its estimate as a percent of the base estimate, filled by significance; the shaded band is the base confidence interval on the same axis and the vertical line is the base estimate. Deletions are drawn in a separate panel from alternative analyses. Diagnostic refits (`kind = "diagnostic"`), which are fitted but not counted, are drawn as triangles in the alternative-analyses panel.

Usage

```
## S3 method for class 'stress_census'
plot(x, ...)
```

Arguments

`x`	A `stress_census`.
`...`	Unused.

Value

A `ggplot` object.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:80), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_covariance(spec, c("cs", "ar1h")),
                     stress_loo(spec, by = "RACE"))
if (requireNamespace("ggplot2", quietly = TRUE)) plot(cen)
```

plot.stress_design *Plot power and robustness power against the sample size*

Description

Plot power and robustness power against the sample size

Usage

```
## S3 method for class 'stress_design'
plot(x, target = 0.8, ...)
```

Arguments

x	A stress_design object from stress_design().
target	Reference line for the robustness power.
...	Ignored.

Value

A ggplot: the probability of a significant result and of a significant and robust result against subjects per arm, one panel per effect size.

---

plot_forest *Forest plot of the filed analysis and the supplementary rows*

Description

Estimate and 95% interval for the filed analysis and the named rows of each census, one panel per endpoint-metric row, the filed row first. When several results objects are given (a trial alone and pooled with an earlier trial, say), the objects are drawn side by side within each row in different colours.

Usage

```
plot_forest(..., rows = NULL, xlab = "Treatment difference (95% interval)")
```

Arguments

...	One or more stress_census or stress_result objects (or lists of them, as in plot_swarm()). Several objects are compared side by side; name the arguments to label them.
rows	Character vector of census row labels to show. The default is every counted row with role = "supplementary".
xlab	Axis label.

Value

A ggplot object.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
d <- mmrm::fev_data
cen <- stress_census(spec, stress_covariance(spec, "cs"),
                  stress_data(spec, list("Black and White only" = d[d$RACE != "Asian", ],
                                         "Female only" = d[d$SEX == "Female", ])))
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plot_forest(cen)
```

```
  plot_forest("All" = cen, "Complete cases" = cen, rows = "Black and White only")
}
```

`plot_pcdf` *Empirical distribution of the p-values across a census*

Description

The empirical cumulative distribution of the p-values of every counted refit, against the uniform diagonal that a result with no true effect would produce (the p-value of each refit of a null result is uniform, so the curve of a null census hugs the diagonal; a census far above it is far from random). One curve per census when several are given, with the pooled curve in black; the share below alpha is annotated.

Usage

```
plot_pcdf(x, alpha = NULL, pooled = TRUE)
```

Arguments

`x`	A `stress_census`, a `stress_result`, the list from `run_plans()`, or a list of censuses or results (one curve each).
`alpha`	Significance level; defaults to the census's own.
`pooled`	Draw the pooled curve over all censuses.

Details

The refits are not independent, so the curve is a description, not a test.

Value

A `ggplot` object.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_covariance(spec, c("cs", "ar1h")), stress_loo(spec, by = "RACE"))
if (requireNamespace("ggplot2", quietly = TRUE)) plot_pcdf(cen)
```

plot_pcurve *Plot p-value curves over a continuous perturbation*

Description

Draws the p-value against the parameter for every family of rows that carries a `param` value (from `stress_tipping()` or `stress_curve()`): one line per family, the alpha reference line, the anchor point (the row of `kind = "alternative"`, usually the unshifted analysis) emphasised, and a dotted vertical line at each interpolated crossing with its value.

Usage

```
plot_pcurve(
  rows,
  alpha = NULL,
  xlab = "Perturbation",
  ylab = "p-value",
  log_p = FALSE,
  ncol = 1
)
```

Arguments

`rows`	A `stress_rows` data frame, a `stress_census` or a `stress_result`; or a named list of these, one panel each. Rows without a `param` are ignored.
`alpha`	Significance level; defaults to the rows' own `alpha`.
`xlab, ylab`	Axis labels.
`log_p`	Draw the p-value axis on a log scale.
`ncol`	Number of columns of panels when `rows` is a list.

Details

Given a named list (one element per endpoint-metric row, say), it draws one panel per element, with the crossings found within each panel.

Value

A `ggplot` object.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:80), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
tp <- stress_tipping(spec, deltas = c(0, -4, -8), m = 3)
if (requireNamespace("ggplot2", quietly = TRUE)) plot_pcurve(tp)

tp4 <- stress_tipping(spec, deltas = c(0, -4, -8), m = 3, shift_visits = "VIS4")
```

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plot_pcurve(list("Every visit" = tp, "Visit 4 only" = tp4))
}
```

`plot_pstrip` *Strip plot of p-values after each deletion*

Description

Each dot is one refit with one unit removed, on a p-value axis (linear by default, log on request), one strip per leave-one-out family; the diamond is the pre-specified analysis with nothing removed and the dashed line is alpha. Counts of deletions below alpha label each strip.

Usage

```
plot_pstrip(
  x,
  alpha = NULL,
  log_p = FALSE,
  xmax = NULL,
  label_family = TRUE,
  highlight = NULL,
  seed = 1L
)
```

Arguments

`x`	A `stress_census`, a `stress_result`, or a list of them (one strip each).
`alpha`	Significance level; defaults to the census's own.
`log_p`	Draw the p-value axis on a log scale (spreads out the small p-values; harder to read for most audiences).
`xmax`	Upper limit of the linear p-value axis; `NULL` fits the data.
`label_family`	Include the deletion family in each strip label; set `FALSE` when every census contributes one family (subjects only, say).
`highlight`	Optional labels of deletion rows (the removed unit, as in the census `label` column) to ring and label, for example the participant whose removal crosses alpha. Use `mask_ids()` first if identifiers must not appear.
`seed`	Seed of the vertical scatter within each strip, so the figure is the same every time it is drawn; the caller's random-number state is restored on exit.

Value

A `ggplot` object.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_loo(spec, by = "RACE"), stress_loo(spec, by = "SEX"))
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plot_pstrip(cen)
  plot_pstrip(cen, highlight = "Asian")
}
```

`plot_swarm` *Swarm of every re-analysis against the filed result*

Description

Two forms. **Single census** (`style = "bubbles"`, the default for one census): the bubble swarm of earlier versions, one row per census on a percent-of-filed axis. **Several rows** (`style = "dots"`, the default for a list): one panel per endpoint-metric row, the filed estimate and its 95% interval drawn as the axis of the panel with the filed value labelled, every counted alternative analysis as a dot placed by its estimate and coloured by the decision it varies (statistical: covariance, inference, covariate set, weighting, grid, estimator; data-handling: data set, population, intercurrent-event strategy, imputation), the counted deletions as a strip of small light dots beneath the axis, a vertical line at no difference, and optional labelled reference lines, one value for every panel or one per panel. Each panel labels its own filed estimate and interval ends to three significant digits. Diagnostic refits (`kind = "diagnostic"`, such as the complete-case refit of the standard plan) are drawn as hollow dots in the colour of their decision group; they are not counted and do not enter the verdict. Counted rows that return no estimate (rank tests) are not drawn; their number per row is in the `caption` attribute of the returned plot, with the counts of alternatives, deletions and diagnostic refits.

Usage

```
plot_swarm(
  x,
  style = c("auto", "bubbles", "dots"),
  reference = NULL,
  reference_label = NULL,
  scale = c("percent", "estimate"),
  digits = 2,
  r0 = 0.004,
  favours = NULL,
  xlab = NULL,
  xlim = NULL
)
```

Arguments

x	A stress_census, a stress_result (from run_plan()), the stress_results list of run_plans(), or a list of censuses or results (one row each). List names, plan titles or census labels name the rows.
style	"bubbles" or "dots"; "auto" chooses bubbles for one census and dots for a list.
reference	Optional x position of a reference line (a clinically meaningful difference, say). In the dots form either one value, drawn in every panel, or one value per row: a vector named by row, or unnamed in row order. NA leaves a panel without a line. Rows on different scales (points and percent change, say) need their own values.
reference_label	Text printed at the reference line: one string, or one per row in the same form as reference.
scale, digits, r0, favours, xlab, xlim	Settings of the bubble form (see the earlier documentation): axis scale, rounding for "the same value", bubble radius, side labels, axis label and limits.

Value

A ggplot object. In the dots form the attribute caption is a character vector, one entry per row, with the counts of alternatives drawn, deletions drawn and rows without an estimate.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
cen <- stress_census(spec, stress_covariance(spec, c("us", "cs", "ar1h")),
                     stress_covariates(spec), stress_loo(spec, by = "RACE"))
if (requireNamespace("ggplot2", quietly = TRUE)) {
  plot_swarm(cen)                                          # one census: bubbles
  spec2 <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                       covariates = ~ FEV1_BL, contrast_visit = "VIS3")
  cen2 <- stress_census(spec2, stress_covariance(spec2, c("us", "cs")),
                        stress_loo(spec2, by = "SEX"))
  g <- plot_swarm(list("FEV1 at visit 4" = cen, "FEV1 at visit 3" = cen2), reference = 1,
                  reference_label = "1 unit")
  attr(g, "caption")
 plot_swarm(list("FEV1 at visit 4" = cen, "FEV1 at visit 3" = cen2),    # one line per panel
             reference = c("FEV1 at visit 4" = 1, "FEV1 at visit 3" = 2),
             reference_label = c("1 unit", "2 units"))
}
```

run_plan *Run a pre-specified stress test*

Description

Runs every family in a `stress_plan()`, assembles the census and the verdict, and keeps the rows by family.

Usage

```
run_plan(plan, cores = 1L, seed = 1)
```

Arguments

plan	A stress_plan.
cores	Worker processes; the cluster is started once and shared by every family.
seed	Seed passed to the families that take one, unless the plan sets it.

Value

An object of class `stress_result`: `plan`, `rows` (a list of `stress_rows` by family), `census`, `verdict`, `base` and `elapsed`.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
plan <- stress_plan(spec, covariance = list(structures = c("cs", "ar1h")),
                    inference = NULL, covariates = NULL, weights = FALSE,
                    loo = list(by = "RACE"), permute = list(n = 9), reference = NULL,
                   estimator = list(estimators = "ancova"), fragility = list(max_k = 1))
res <- run_plan(plan)
res
```

run_plans *Run several plans (several endpoints or comparisons) and compare*

Description

Run several plans (several endpoints or comparisons) and compare

Usage

```
run_plans(plans, cores = 1L, seed = 1)
```

Arguments

plans	A named list of stress_plan objects.
cores, seed	Passed to run_plan().

Value

A named list of stress_result objects of class stress_results; summary() gives one line per plan and plot_swarm() accepts the censuses (lapply(results, function(r) r$census)).

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
mk <- function(s) stress_plan(s, covariance = list(structures = c("cs", "ar1h")), inference = NULL,
                          covariates = NULL, weights = FALSE, loo = NULL, permute = NULL,
                              reference = NULL, estimator = NULL, fragility = NULL)
res <- run_plans(list(VIS4 = mk(spec), VIS3 = mk(spec_update(spec, contrast_visit = "VIS3"))))
summary(res)
```

---

simulate_rare_trial *Simulate a small trial with rare-disease features*

Description

Draws a two-arm longitudinal trial with the features that make small trials in rare disease hard to read: few participants, heterogeneous progression (a mixture of fast and slow progressors with a random slope), an effect that varies with progression, sites, dropout, and optionally one planted defect. It is the data generator of the simulation studies in the methods paper and complements simulate_trial(), which it leaves unchanged.

Usage

```
simulate_rare_trial(
  n_per_arm = 25,
  delta = 0,
  n_visits = 4,
  sigma = 10,
  rho = 0.6,
  dropout = 0.03,
  baseline_beta = 0.5,
  population = c("homogeneous", "mixture", "heavy"),
  fast_frac = 0.35,
  fast_decline = 8,
  slope_sd = 4,
  fast_effect = 1.6,
  t_df = 3,
```

```
  n_sites = 8,
  site_sd = 2.5,
  defect = c("none", "outlier", "site", "mnar", "allocation"),
  n_outliers = 1,
  outlier_size = 10,
  mnar_strength = 1.5,
  mnar_dropout = 3 * dropout,
  fast_to_active = 0.25,
  seed = NULL
)
```

Arguments

`n_per_arm`	Participants per arm (exactly balanced).
`delta`	Average treatment effect at the last visit (higher is better).
`n_visits`, `sigma`, `rho`	Visits, residual SD and AR(1) correlation.
`dropout`	Missing-at-random dropout probability per visit.
`baseline_beta`	Slope of the response on the baseline.
`population`	`"homogeneous"`, `"mixture"` or `"heavy"`.
`fast_frac`, `fast_decline`, `slope_sd`, `fast_effect`	Mixture settings.
`t_df`	Degrees of freedom of the heavy-tailed errors.
`n_sites`, `site_sd`	Number of sites and SD of the site effect on progression.
`defect`	`"none"`, `"outlier"`, `"site"`, `"mnar"` or `"allocation"`.
`n_outliers`, `outlier_size`	Outlier settings (size in units of `sigma`).
`mnar_strength`, `mnar_dropout`	Informative-dropout settings.
`fast_to_active`	Allocation-defect setting: the weight with which a fast progressor is drawn into the active arm (slow progressors get `1 - fast_to_active`); 0.5 is fair allocation.
`seed`	Random seed; the caller's random-number state is restored. The draws use R's default generators whatever the caller's `RNGkind()`, so a seed gives the same trial in a plain session and on parallel workers.

Details

The response is a change score at visits 1 to `n_visits`. For participant *i* at visit *v*, with $t_v = v/V$,

$$y_{iv} = \beta(b_i - 50) + (d_i + s_i + u_{site(i)})t_v + T_i e_i t_v + \epsilon_{iv},$$

where b_i is the baseline, d_i the extra decline of a fast progressor, s_i a random slope, u a site effect, T_i the active-arm indicator, e_i the participant's treatment effect and ϵ AR(1) errors with standard deviation `sigma`.

Populations:

- "homogeneous": no progression heterogeneity; fast is recorded but is not prognostic, so models with and without it target the same effect.
- "mixture": a fraction fast_frac of fast progressors lose a further fast_decline points by the last visit; random slope with SD slope_sd; the effect is fast_effect times larger in fast progressors, scaled so that the average effect is delta.
- "heavy": as homogeneous with Student-t errors on t_df degrees of freedom, scaled to the same standard deviation.

Defects (each makes a significant result one that should not be trusted):

- "outlier": n_outliers active-arm participants shifted up by outlier_size standard deviations at every visit (a data error).
- "site": the treatment effect delta applies at site 1 only.
- "mnar": active-arm dropout also depends on the unobserved next value, so participants about to decline leave (informative dropout), with a baseline dropout rate of mnar_dropout per visit.
- "allocation": allocation is not independent of prognosis (for example a stratification error or selection at enrolment). Participants are drawn into the active arm with weight fast_to_active for fast progressors and 1 - fast_to_active for slow ones, so with fast_to_active < 0.5 fast progressors (who decline more in the "mixture" population) gather in the reference arm. An analysis that omits fast is then biased in favour of the active arm; one that adjusts for it is not.

Value

A data frame in long format with one row per observed visit: id, arm ("REF" or "ACT"), site, base, fast (0/1), visit ("V1", ...) and y.

See Also

simulate_trial() for the simpler generator.

Examples

```
d <- simulate_rare_trial(n_per_arm = 20, delta = 5, population = "mixture", seed = 1)
head(d)
spec <- stress_spec(d, "y", "id", "visit", "arm", "REF", covariates = ~ base + fast)
stress_fit(spec)[, c("estimate", "stderr", "p")]
```

---

simulate_trial *Simulate a two-arm longitudinal trial*

Description

Multivariate-normal responses at n_visits post-baseline visits with an AR(1) correlation, a baseline covariate, a treatment effect growing linearly to delta at the last visit, and dropout after each visit whose probability rises as the current value falls (missing at random).

Usage

```
simulate_trial(
  n_per_arm = 50,
  delta = 0,
  n_visits = 4,
  sigma = 10,
  rho = 0.6,
  dropout = 0.08,
  baseline_beta = 0.5,
  errors = c("normal", "t"),
  t_df = 3,
  hetero = 1,
  mnar = 0,
  interaction = 0,
  seed = NULL
)
```

Arguments

`n_per_arm`	Subjects per arm.
`delta`	Treatment effect at the last visit (active minus reference).
`n_visits`	Number of post-baseline visits.
`sigma`	Residual standard deviation at each visit.
`rho`	AR(1) correlation between adjacent visits.
`dropout`	Baseline per-visit dropout probability.
`baseline_beta`	Slope of the response on the baseline covariate.
`errors`	`"normal"` or `"t"` (Student t with `t_df` degrees of freedom, scaled to the same standard deviation: heavy tails).
`t_df`	Degrees of freedom for `errors = "t"`.
`hetero`	Multiplier of the residual standard deviation at the last visit relative to the first (1 = homoscedastic); the SD grows linearly.
`mnar`	Strength of missing-not-at-random dropout: dropout after a visit depends on the value the subject would have had at the NEXT visit (0 = missing at random, as the default).
`interaction`	Treatment effect modification by the baseline covariate: the effect at the last visit is `delta * (1 + interaction * (base - 50) / 10)`.
`seed`	Random seed.

Value

A long data frame with `id`, `arm` (`"REF"`, `"ACT"`), `base`, `visit` (`"V1"`, ...) and `y`, missing rows removed.

Examples

```
d <- simulate_trial(40, delta = 3, seed = 1); head(d)
```

spec_update *Modify a stress spec*

Description

Returns a copy of spec with the named settings replaced. Settings that change the data conventions (reference, visit, treatment, subject, response) cannot be changed here; build a new spec instead.

Usage

```
spec_update(spec, ...)
```

Arguments

spec	A stress_spec.
...	Settings to replace: covariance, method, vcov, covariates, contrast_visit, alpha, label, data, weights, active.

Value

The modified stress_spec.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
spec_update(spec, covariance = "ar1h", method = "Satterthwaite", vcov = "Asymptotic")
```

standard_plan *The standard stress test*

Description

A recommended, citable set of families with fixed settings, so that a statistical analysis plan can say "the standard stress test of ballast" and mean one thing. It contains, in order:

Usage

```
standard_plan(
  spec,
  site = NULL,
  strata = site,
  baseline = NULL,
  deltas = NULL,
  shift_visits = NULL,
```

```
  ice = NULL,
  m = 25,
  n_perm = 2000,
  max_k = 5
)
```

Arguments

spec	A stress_spec().
site	Optional site or centre column (deletions, random intercept, re-randomisation strata unless strata is given).
strata	Optional stratification column for the re-randomisation test.
baseline	Optional baseline column for cLDA.
deltas	Optional shifts for the tipping-point family.
shift_visits	Visits whose imputed values the tipping family shifts; NULL shifts the contrast visit only (the plan text states the visits; pass every visit level to shift all imputed values).
ice	Optional intercurrent-event table for stress_estimand().
m	Imputations for the multiple-imputation families.
n_perm	Permutations for the re-randomisation test.
max_k	Maximum removals in the deletion search.

Details

1. covariance structure: us, ar1h, ar1, cs, csh, toep, toeph;
2. inference: Satterthwaite, Kenward-Roger, linear Kenward-Roger, residual, sandwich with between-within df, jackknife sandwich;
3. covariate set: each covariate dropped, all dropped;
4. LS-mean weighting: equal and observed margins;
5. leave-one-out: every subject, and every level of site if given;
6. re-randomisation test: 2,000 permutations, within strata if given;
7. reference-based multiple imputation: MAR, J2R, CR, CIR, 25 imputations;
8. delta-adjusted multiple imputation over deltas if given;
9. estimator class: GEE independence, ANCOVA, robust ANCOVA, plus a random intercept for site and cLDA for baseline if given; the pre-specified MMRM on complete cases is also fitted, drawn and tabulated, as a diagnostic that is not counted (it answers a different question from the filed analysis, so it does not enter the verdict);
10. greedy deletion search for the loss of significance, up to 5 subjects;
11. intercurrent-event strategies if ice is given (treatment policy, hypothetical, while on treatment).

Value

A stress_plan.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
standard_plan(spec, site = "RACE", baseline = "FEV1_BL")
```

`stress_calibrate` *Pre-trial calibration of the stress test*

Description

Simulates trials from a planned design and runs a light stress test on each (covariance, inference, covariate, weighting and leave-one-out families, plus the greedy deletion search if `fragility = TRUE`), so that a sponsor can see, before unblinding, how often a trial of this design would be classed robust, sensitive or fragile at each power, and how often the marginal flag would be set.

Usage

```
stress_calibrate(
  n_per_arm = 50,
  power = c(0, 0.5, 0.8, 0.95),
  n_trials = 100,
  n_visits = 4,
  sigma = 10,
  rho = 0.6,
  dropout = 0.08,
  baseline_beta = 0.5,
  errors = "normal",
  t_df = 3,
  hetero = 1,
  mnar = 0,
  interaction = 0,
  fragility = FALSE,
  cores = 1L,
  seed = 1,
  rule = c("interval", "interval_or_direction")
)
```

Arguments

`n_per_arm`	Subjects per arm.
`power`	Target powers for the pre-specified analysis; the effect size giving each is derived from the design (0 gives the null).
`n_trials`	Trials per power.
`n_visits`	Number of post-baseline visits.
`sigma`	Residual standard deviation at each visit.

rho	AR(1) correlation between adjacent visits.
dropout	Baseline per-visit dropout probability.
baseline_beta	Slope of the response on the baseline covariate.
errors, t_df, hetero, mnar, interaction	Misspecification settings passed to simulate_trial().
fragility	Run the deletion search (slower).
cores	Worker processes (the trials are distributed, so each trial's families run on one core).
seed	Base seed.
rule	Classification rule for stress_verdict().

Value

A data frame of class stress_calibration with one row per power: the effect size, observed power, and the proportion of trials with each verdict, the marginal flag, and (if run) a one-subject fragility; attr(, "trials") holds the per-trial rows.

Examples

```
stress_calibrate(n_per_arm = 20, power = c(0, 0.8), n_trials = 2)
```

---

stress_census *Assemble a census of perturbations against the base analysis*

Description

Binds the rows returned by the family functions and scores each against the pre-specified (base) fit: percent of the base estimate, shift in base standard errors, whether the estimate lies inside the base confidence interval, keeps the base direction, is significant at alpha, and whether significance changed. Rows that reproduce the base (is_base), did not converge, are tipping-point searches, are diagnostic refits (kind = "diagnostic", such as the complete-case refit of the standard plan, which is fitted and shown but does not enter the verdict), or duplicate an earlier row are not counted.

Usage

```
stress_census(spec, ..., base = NULL)
```

Arguments

spec	A stress_spec(); the base fit is stress_fit(spec).
...	stress_rows data frames from the family functions.
base	Optional pre-computed base row, to avoid refitting.

Value

A data frame of class `stress_census` with the family rows plus `pct_of_base`, `shift_se`, `inside_ci`, `same_direction`, `significant`, `crossed`, `duplicate` (a later row numerically identical to an earlier counted row, so the same specification reached by two families), `counted`; the base row is `attr(, "base")`.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
cen <- stress_census(spec, stress_covariance(spec, c("us", "cs", "ar1h")),
                     stress_covariates(spec))
summary(cen)
stress_verdict(cen)
```

`stress_covariance` *Perturbation family: within-subject covariance structure*

Description

Refits the pre-specified model under each covariance structure. The row that reproduces the spec's own structure is flagged `is_base`.

Usage

```
stress_covariance(
  spec,
  structures = c("us", "ar1h", "ar1", "cs", "csh", "toep", "toeph"),
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`structures`	Covariance structure names (see `stress_spec()`).
`cores`	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame, one row per structure.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_covariance(spec, c("us", "cs", "ar1h"))
```

`stress_covariates` *Perturbation family: covariate set*

Description

Refits the model with each covariate dropped in turn, with no covariates, and with any user-supplied covariate sets.

Usage

```
stress_covariates(
  spec,
  drop_each = TRUE,
  drop_all = TRUE,
  sets = list(),
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`drop_each`	Drop each pre-specified covariate term in turn.
`drop_all`	Fit with treatment and visit only.
`sets`	Named list of additional covariate sets (formulas or character vectors of term labels); each is fitted as given.
`cores`	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_covariates(spec, sets = list("with weight" = ~ RACE + SEX + FEV1_BL + WEIGHT))
```

stress_curve *Perturbation family: a curve over a continuous data perturbation*

Description

Applies a user-supplied modification to the spec's data at each value of a continuous parameter, refits the pre-specified model and returns one row per value with the value in `param`. This generalises tipping-point searches that are not multiple-imputation based: assigning a contested participant a post-deviation trajectory, shifting one arm's observed values, moving a baseline definition, and so on. Rows are `kind = "search"` except the row at `anchor`, which is a counted alternative.

Usage

```
stress_curve(
  spec,
  values,
  modify,
  family = "perturbation curve",
  anchor = NULL,
  role = "sensitivity",
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`values`	Numeric vector of parameter values.
`modify`	A function `function(data, value)` returning a data frame in the spec's layout. It receives the spec's own data.
`family`	Family label for the rows.
`anchor`	The parameter value (if any) that represents an alternative analysis in its own right rather than a search point; `NULL` for none.
`role`	`"sensitivity"` or `"supplementary"`; see `stress_fit()`.
`cores`	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame, one row per value, with `param` set.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
# shift every observed value of one subject on the active arm
shift_pt3 <- function(d, v) { k <- d$USUBJID == "PT3"; d$FEV1[k] <- d$FEV1[k] + v; d }
cv <- stress_curve(spec, values = c(-20, -10, 0, 10, 20), modify = shift_pt3, anchor = 0)
tipping_point(cv)
```

`stress_data` *Perturbation family: alternative analysis data sets*

Description

The generic data-decision family. Each alternative is a data frame in the spec's layout (an intercurrent-event handling, a different baseline definition, a contested record reversed, a different visit window). The pre-specified model is refitted on each.

Usage

```
stress_data(
  spec,
  alternatives,
  family = "data handling",
  role = "supplementary",
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`alternatives`	Named list of data frames.
`family`	Family label for the rows.
`role`	`"supplementary"` (the default: an alternative data set usually targets a different estimand) or `"sensitivity"`.
`cores`	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame, one row per alternative.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
alt <- mmrm::fev_data
alt$FEV1[alt$USUBJID == "PT1"] <- NA   # one subject's records set missing
stress_data(spec, list("PT1 removed" = alt))
```

stress_design *Size a trial for a robust result*

Description

Simulates trials from a planned design over a grid of sample sizes and effect sizes, runs the light stress test on each (covariance, inference, covariate, weighting and leave-one-subject-out families) and reports, for every cell, the conventional power next to the robustness power: the probability that the pre-specified analysis is significant and classed robust. The deletion search and a delta-adjusted tipping grid are optional and give the expected fragility index and the shift the result can absorb. This is the design-stage use of the stress test: choose the sample size so that a true effect is expected to come out robust rather than only significant, and write the expected verdict rates into the statistical analysis plan before unblinding so that the observed verdict has a pre-registered comparison.

Usage

```
stress_design(
  n_per_arm,
  delta,
  n_trials = 100,
  n_visits = 4,
  sigma = 10,
  rho = 0.6,
  dropout = 0.08,
  baseline_beta = 0.5,
  covariance = "us",
  method = "Satterthwaite",
  vcov = "Asymptotic",
  alpha = 0.05,
  alternatives = c("us", "ar1h", "cs"),
  loo = TRUE,
  fragility = FALSE,
  max_k = 5,
  tipping = FALSE,
  shifts = seq(0, 2, by = 0.25),
  m = 5,
  errors = "normal",
  t_df = 3,
  hetero = 1,
  mnar = 0,
  interaction = 0,
  cores = 1L,
  seed = 1,
  rule = c("interval", "interval_or_direction")
)
```

```
## S3 method for class 'stress_design'
print(x, target = 0.8, ...)
```

Arguments

n_per_arm	Subjects per arm to evaluate (a vector).
delta	Treatment effects at the last visit to evaluate (a vector); include 0 to see the null.
n_trials	Trials per cell.
n_visits	Number of post-baseline visits.
sigma	Residual standard deviation at each visit.
rho	AR(1) correlation between adjacent visits.
dropout	Baseline per-visit dropout probability.
baseline_beta	Slope of the response on the baseline covariate.
covariance, method, vcov	The pre-specified analysis model, as in stress_spec().
alpha	Two-sided significance level.
alternatives	Covariance structures offered as alternatives; the pre-specified one is removed from the set.
loo	Run the leave-one-subject-out family.
fragility	Run the greedy deletion search for significance up to max_k removals (slow). Otherwise a one-subject loss of significance is read from the leave-one-out rows.
max_k	Removals allowed in the deletion search.
tipping	Run a delta-adjusted tipping grid (imputed active-arm values shifted down) and record the shift at which p crosses alpha.
shifts	Shifts for the tipping grid as multiples of delta (of sigma when delta is 0).
m	Imputations per shift.
errors	"normal" or "t" (Student t with t_df degrees of freedom, scaled to the same standard deviation: heavy tails).
t_df	Degrees of freedom for errors = "t".
hetero	Multiplier of the residual standard deviation at the last visit relative to the first (1 = homoscedastic); the SD grows linearly.
mnar	Strength of missing-not-at-random dropout: dropout after a visit depends on the value the subject would have had at the NEXT visit (0 = missing at random, as the default).
interaction	Treatment effect modification by the baseline covariate: the effect at the last visit is delta * (1 + interaction * (base - 50) / 10).
cores	Worker processes. Trials are distributed, so each trial's families run on one core.
seed	Base seed; every trial's seed is derived from it.
rule	Classification rule for stress_verdict().
x	A stress_design object.
target	Robustness power to reach; the print method names the smallest sample size per arm that reaches it for each non-zero effect.
...	Ignored.

Details

The light family set offers the alternative covariance structures, three inference methods, the model without the baseline covariate, observed-margin weighting and, if `loo = TRUE`, every single-subject deletion. Deletions dominate the run time at large sample sizes; `loo = FALSE` gives a fast first pass whose verdicts omit the deletion family.

Value

A data frame of class `stress_design` with one row per cell: `n_per_arm`, `delta`, `trials`, `power` (proportion significant), `robustness_power` (significant and robust), `robust_given_sig`, `sensitive_given_sig`, `fragile_given_sig`, `marginal_given_sig`, `k1_given_sig` (one deletion loses significance), `k_reached_given_sig` and `k_median` (deletion search, if run), `tipping_margin` (median shift that crosses `alpha` among significant trials, as a multiple of `delta` or `sigma`) and `tipping_absorbed` (share of significant trials with no crossing within the grid), and `max_shift_se` (median largest shift in pre-specified standard errors). `attr(, "trials")` holds the per-trial rows and `attr(, "design")` the settings. `plot.stress_design()` draws power and robustness power against the sample size.

See Also

`stress_calibrate()` for the verdict rates of one design at a set of planned powers; `stress_sap()` for the analysis-plan text.

Examples

```
des <- stress_design(n_per_arm = c(40, 80), delta = c(0, 5), n_trials = 4, loo = FALSE)
des
```

---

stress_direction *Direction stability of a result that misses significance*

Description

A small trial can carry a real effect and still miss significance for want of power. For such a result the stress test reports how stable what was seen is: how many pre-specified re-analyses reverse the direction of the estimate, and which; how many move it outside the filed confidence interval; and how many participants the greedy deletion search removes before the direction reverses.

Usage

```
stress_direction(
  spec,
  census = NULL,
  max_k = 10L,
  method = c("ranked", "greedy"),
  cores = 1L
)
```

Arguments

spec	A stress_spec().
census	Optional stress_census over the pre-specified re-analyses. If given, its counted rows supply the reversal and interval counts; without it only the deletion count is computed.
max_k	Most participants to remove in the deletion search (stress_fragility() with target = "direction"). The default is 10.
method	"ranked" (default) or "greedy", as in stress_fragility().
cores	Passed to stress_fragility().

Value

A list of class stress_direction: base (the pre-specified fit), z, n, significant, alpha; from the census n_counted, n_reversed, n_outside_ci and reversed_rows (NA and NULL without one); from the search k_rev (participants removed to reverse, NA if not reached within max_k), k_rev_reached, subjects, share_of_n and search, the rows of the search.

See Also

stress_verdict() for the summary label, stress_fragility() for the deletion searches.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_covariance(spec, c("cs", "ar1h")), stress_covariates(spec))
stress_direction(spec, cen, max_k = 6)
```

---

stress_estimand *Perturbation family: intercurrent-event strategies (ICH E9(R1))*

Description

Declares the intercurrent events that occurred and derives the analysis data set implied by each handling strategy, then refits the pre-specified model on each. Strategies per event type:

Usage

```
stress_estimand(
  spec,
  ice,
  strategies = c("treatment_policy", "hypothetical", "while_on_treatment", "composite"),
  composite_value = NULL,
  cores = 1L
)
```

Arguments

spec	A stress_spec().
ice	A data frame with one row per intercurrent event: the subject identifier (column named as the spec's subject), the first visit affected (column named as the spec's visit) and type (a label such as "discontinuation" or "rescue").
strategies	Named list, one element per event type, each a character vector of strategies to apply; or a single character vector applied to every type.
composite_value	A number or function used by the composite strategy.
cores	Number of worker processes; 1 fits sequentially.

Details

- "treatment_policy": use the data as observed, whatever happened.
- "hypothetical": set the responses at and after the event to missing; the model's missing-at-random assumption then supplies the value the subject would have had without the event.
- "while_on_treatment": drop the records at and after the event.
- "composite": replace the responses at and after the event with an unfavourable value: a number, or a function function(data, rows) returning one value per affected row (the worst observed change in the subject's arm at that visit, say).

With several event types the strategies are crossed, one row per combination, and the row that applies "treatment_policy" to every type reproduces the pre-specified data and is flagged is_base. Every other row targets a different estimand, so they are role = "supplementary"; a hypothetical strategy can be followed by stress_reference() on the derived data for its reference-based sensitivity analyses.

Value

A stress_rows data frame, one row per strategy combination; attr(, "data") holds the derived data sets by label.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:80), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
ice <- data.frame(USUBJID = c("PT2", "PT5", "PT9"), AVISIT = c("VIS3", "VIS2", "VIS4"),
                  type = c("discontinuation", "discontinuation", "rescue"))
strat <- list(discontinuation = c("treatment_policy", "hypothetical"),
              rescue = c("treatment_policy", "composite"))
stress_estimand(spec, ice, strategies = strat, composite_value = 20)
```

`stress_estimator` *Perturbation family: estimator class*

Description

Refits the question with estimators other than the pre-specified MMRM:

Usage

```
stress_estimator(
  spec,
  estimators = c("complete_cases", "gee_independence", "ancova", "robust_ancova",
    "random_intercept", "clda"),
  random = NULL,
  baseline = NULL,
  seed = 1,
  cores = 1L,
  diagnostic = NULL
)
```

Arguments

`spec`	A `stress_spec()`.
`estimators`	Which estimators to run.
`random`	Name of a grouping column for `"random_intercept"` (a site or centre variable that is also one of the spec's covariates).
`baseline`	Name of the baseline column for `"clda"`; the response must be change from that baseline.
`seed`	Seed for the robust regression's random initial estimate.
`cores`	Number of worker processes; `1` fits sequentially.
`diagnostic`	Names of estimators (from `estimators`) whose rows are diagnostic: fitted, drawn and tabulated with `kind = "diagnostic"`, but not counted by `stress_census()`, so they do not enter the verdict or its counts. `NULL` (the default) counts every estimator. The standard plan (`standard_plan()`) makes the complete-case refit diagnostic.

Details

- `"complete_cases"`: the pre-specified MMRM on subjects observed at every visit (no reliance on missing-at-random within the likelihood).
- `"gee_independence"`: working-independence weighted least squares with a visit-specific residual variance and a cluster-robust (sandwich) variance, between-within degrees of freedom; no within-subject covariance is modelled. Needs `nlme`.
- `"ancova"`: analysis of covariance at the contrast visit, observed cases, residual degrees of freedom. Covariate terms involving the visit variable are dropped.

- `"robust_ancova"`: MM-type robust regression at the contrast visit (`MASS::rlm`), normal-theory inference. Needs `MASS`.
- `"clda"`: constrained longitudinal data analysis: the raw outcome (`baseline` plus the response, which must be a change from baseline) is modelled at baseline and every post-baseline visit with one common baseline mean across arms, an unstructured covariance over all visits and the other covariates as fixed effects; the contrast is the difference in the arms' means at the contrast visit. Needs `baseline`.
- `"random_intercept"`: the pre-specified fixed effects with `random` as a random intercept instead of a fixed covariate, the spec's covariance structure within subject where `nlme` has an equivalent (`us`, `ar1h`, `ar1`, `cs`, `csh`); normal-theory inference. Needs `nlme` and `random`.

Every estimator reports the same least-squares-mean contrast at the contrast visit (class covariates at equal weights, continuous covariates at their means).

Value

A `stress_rows` data frame, one row per estimator.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_estimator(spec, c("complete_cases", "ancova"))
```

`stress_explain` *Explain why an alternative analysis moved*

Description

For one row of a census (an alternative that crossed alpha, say), attributes the difference between its estimate and the pre-specified estimate to subjects. Each subject's influence is the change in the contrast when that subject is removed, computed under the pre-specified settings and again under the alternative's settings; the difference of the two influences says how much more (or less) that subject drives the result under the alternative. Subjects are ranked by the absolute difference.

Usage

```
stress_explain(spec, row, data = NULL, top = 10, cores = 1L)
```

Arguments

`spec`	A `stress_spec()`.
`row`	One row of a `stress_census` or `stress_rows` data frame.
`data`	The alternative's data set, if it differs from the spec's.
`top`	Number of subjects to print.
`cores`	Number of worker processes; `1` fits sequentially.

Details

The alternative's model settings are read from the row (`covariance`, `method`, `vcov`, `covariates`). For an alternative that used different data, pass that data as `data`.

Value

A data frame of class `stress_explanation` with one row per subject: `subject`, `arm`, `n_visits`, `infl_base`, `infl_alt`, `difference`, sorted by absolute difference; the row explained is `attr(, "row")` and the two full-data estimates are `attr(, "estimates")`.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:30), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
cen <- stress_census(spec, stress_covariates(spec))
stress_explain(spec, cen[1, ], top = 5)
```

---

stress_fit *Fit one MMRM and return one row*

Description

Fits the model described by a `stress_spec()` (optionally on alternative data or with alternative settings) and returns the treatment contrast at the contrast visit as a single row in the package's common schema. This is the engine every family function calls.

Usage

```
stress_fit(
  spec,
  data = NULL,
  label = spec$label,
  family = "base",
  kind = "alternative",
  is_base = is.null(data),
  note = "",
  covariates = spec$covariates,
  covariance = spec$covariance,
  method = spec$method,
  vcov = spec$vcov,
  at = list(),
  role = "sensitivity",
  weights = spec$weights
)
```

Arguments

spec	A stress_spec.
data	Alternative data set in the same layout as spec$data; NULL fits the spec's own data.
label, family, kind	Labels for the output row. kind is one of "alternative" (a counted alternative analysis), "deletion" (a leave-one-out refit), "search" (a point on a tipping-point grid) or "diagnostic" (fitted and shown but not counted).
is_base	TRUE when the row reproduces the pre-specified analysis exactly; such rows are not counted in a census.
note	Free text carried into the row.
covariates, covariance, method, vcov	Overrides of the spec's settings.
at	Named list of continuous covariates fixed at a value instead of their mean when forming the least-squares means (SAS LSMEANS ... AT).
role	"sensitivity" for an analysis that keeps the estimand and changes an assumption or a modelling choice; "supplementary" for one that targets a different estimand (ICH E9(R1)).
weights	LS-mean weighting, "equal" or "observed"; see stress_spec().

Value

A one-row data frame of class stress_rows with columns family, label, kind, role, is_base, converged, note, estimate, stderr, df, lcl, ucl, p, n_subj, n_obs, aicc, param (the numeric value of a continuous perturbation, NA otherwise), covariance, method, vcov, covariates, alpha.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_fit(spec)
```

---

stress_fragility *Smallest set of subjects whose removal changes the conclusion*

Description

Two searches for the fewest subjects whose removal moves the result to the target. The number needed is a continuous-outcome analogue of the fragility index; both searches give an upper bound on the true minimum.

Usage

```
stress_fragility(
  spec,
  target = c("significance", "ci", "direction"),
  max_k = 5,
  candidates = NULL,
  cores = 1L,
  method = c("greedy", "ranked")
)
```

Arguments

spec	A stress_spec().
target	What has to change: "significance" (the p-value crosses alpha in either direction, depending on the base result), "ci" (the estimate leaves the base confidence interval) or "direction" (the estimate changes sign).
max_k	Maximum number of subjects to remove.
candidates	Optional character vector of subject identifiers to restrict the search to (one arm, say).
cores	Number of worker processes; 1 fits sequentially.
method	"greedy" or "ranked".

Details

- method = "greedy" (default): at each step every remaining subject is removed in turn from the current data, the pre-specified model is refitted, and the subject whose removal moves the result furthest towards the target is dropped for good. About k * n refits.
- method = "ranked": subjects are ranked once, by how far removing each alone moves the result towards the target (the leave-one-out refits), and the top 1, 2, ..., k are removed together, one refit per step. About refits, so it is practical for large trials. influences at once.

Value

A stress_rows data frame with one row per step (kind = "search", so the rows are not counted in a census, since the search crosses the target by construction; param = number removed, label = the subjects removed so far). attr(, "fragility") is a list with target, method, k (the step at which the target was reached, NA if never), subjects and reached.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
fr <- stress_fragility(spec, max_k = 1)
attr(fr, "fragility")
attr(stress_fragility(spec, "ci", max_k = 10, method = "ranked"), "fragility")$k
```

stress_grid *Perturbation family: crossed specification grid*

Description

Fits every combination of the supplied covariance structures, inference settings, covariate sets and data sets (a specification curve). The cell that reproduces the spec is flagged `is_base`.

Usage

```
stress_grid(
  spec,
  covariance = spec$covariance,
  inference = list(base = list(method = spec$method, vcov = spec$vcov)),
  covariates = list(base = spec$covariates),
  data = list(base = NULL),
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`covariance`	Character vector of covariance structures.
`inference`	Named list of `list(method = , vcov = )` pairs.
`covariates`	Named list of covariate sets (formulas or term labels).
`data`	Named list of data frames; a `NULL` element is the spec's data.
`cores`	Number of worker processes; `1` fits sequentially.

Value

A `stress_rows` data frame, one row per cell, labelled `"covariance | inference | covariates | data"`.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_grid(spec, covariance = c("us", "cs"),
            covariates = list(full = ~ RACE + SEX + FEV1_BL, baseline = ~ FEV1_BL))
```

stress_html *Interactive HTML report*

Description

Writes an HTML page (its JavaScript dependencies in a `lib` folder beside it) with the verdict, the summary table and two interactive figures built with `plotly`: the bubble swarm, where hovering a bubble lists the specifications that give that estimate, and the deletion strip, where hovering a dot names the subject removed and shows the refit's estimate and p-value. Needs `plotly` and `htmlwidgets`.

Usage

```
stress_html(result, file, digits = 2, mask_ids = TRUE)
```

Arguments

`result`	A `stress_result` from `run_plan()`.
`file`	Path of the HTML file to write. There is no default, so nothing is written unless a path is given.
`digits`	Rounding used to group estimates into bubbles.
`mask_ids`	Replace subject identifiers with sequence numbers (on by default: the page is the deliverable most likely to leave the organisation).

Value

The file path, invisibly.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
plan <- stress_plan(spec, covariance = list(structures = c("cs", "ar1h")), inference = NULL,
                covariates = NULL, weights = FALSE, loo = list(by = "RACE"), permute = NULL,
                    reference = NULL, estimator = NULL, fragility = NULL)
if (requireNamespace("plotly", quietly = TRUE) && requireNamespace("htmlwidgets", quietly = TRUE))
  stress_html(run_plan(plan), file.path(tempdir(), "stress.html"))
```

stress_inference *Perturbation family: degrees-of-freedom method and variance estimator*

Description

Perturbation family: degrees-of-freedom method and variance estimator

Usage

```
stress_inference(spec, settings = default_inference(), cores = 1L)
```

Arguments

spec	A stress_spec().
settings	Named list of list(method = , vcov =) pairs; see default_inference().
cores	Number of worker processes; 1 fits sequentially.

Value

A stress_rows data frame, one row per setting.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_inference(spec, default_inference()[c("Satterthwaite", "Kenward-Roger (linear)")])
```

stress_log *Write the run log for an audit trail*

Description

Records the plan text and its MD5 hash, the seed, the package and R versions, the platform, the elapsed time and the verdict, as JSON if jsonlite is installed and as plain text otherwise. The hash of the plan text lets a reviewer confirm that the stress test run is the one that was pre-specified.

Usage

```
stress_log(result, file = NULL, seed = 1)
```

Arguments

result	A stress_result.
file	Path to write; NULL returns the log without writing.
seed	The seed that was passed to run_plan().

Value

The log as a list, invisibly.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
plan <- stress_plan(spec, covariance = list(structures = c("cs", "ar1h")), inference = NULL,
                    covariates = NULL, weights = FALSE, loo = NULL, permute = NULL,
                    reference = NULL, estimator = NULL, fragility = NULL)
str(stress_log(run_plan(plan)))
```

---

stress_loo *Perturbation family: leave-one-out deletions*

Description

Refits the model with each subject removed, or with each level of a grouping variable (site, cohort, country) removed. Rows are `kind = "deletion"` and are summarised separately from alternative analyses.

Usage

```
stress_loo(spec, by = NULL, cores = 1L)
```

Arguments

spec	A `stress_spec()`.
by	Name of a grouping column; `NULL` deletes subjects one at a time.
cores	Number of worker processes; `1` fits sequentially.

Value

A `stress_rows` data frame, one row per deleted unit.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:40), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
stress_loo(spec, by = "RACE")
```

`stress_path` *Smallest combination of analytic changes that leaves the confidence interval*

Description

A greedy search for the fewest analytic changes, made together, that move the estimate outside the pre-specified 95% confidence interval. At each step every change from a family not yet used is tried on top of the changes already made, and the one that moves the estimate furthest in the search direction is kept. The search stops when the estimate leaves the interval or `max_j` changes have been made. It runs towards zero (the direction that weakens the result) and, with `direction = "both"`, also away from zero; `j` is the smaller count.

Usage

```
stress_path(
  spec,
  changes = path_changes(spec),
  max_j = 3,
  direction = c("both", "toward_zero", "away_from_zero"),
  seed = 1,
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`: the pre-specified analysis.
`changes`	Candidate changes from `path_changes()`.
`max_j`	Largest number of changes to combine.
`direction`	`"both"`, `"toward_zero"` or `"away_from_zero"`.
`seed`	Seed for the reference-based imputations.
`cores`	Worker processes (or a cluster) for the candidates of a step.

Details

Changes combine: a reference-based or delta-adjusted imputation chosen at one step is kept, and later changes are evaluated under it. A path that leaves the interval through either is flagged `assumption_dependent`, so that the reader can see the path ran through a missing-data assumption rather than through the observed data alone.

Because the search is greedy, `j` is an upper bound on the smallest combination; to first order the shifts of different families add, so the greedy path is the worst case when they do.

The search is exact but avoids repeated work: the imputation draws are made once per imputation model and reused by every change that leaves the model alone, a state reached twice (by both directions, or by two orders of the same changes) is fitted once, and the second direction is searched only as far as it could still give a shorter path than the first.

Value

A `stress_rows` data frame with one row per step of each direction searched (`family` is `"analytic path"`), with attribute `"path"`: a list with `j` (`NA` if not reached within `max_j`), `reached`, `direction`, `changes` (the labels of the path that reached the boundary first), `assumption_dependent` and the base estimate and interval.

See Also

`stress_fragility()` for the deletion searches.

Examples

```
d <- simulate_rare_trial(25, delta = 6, population = "mixture", seed = 2)
spec <- stress_spec(d, "y", "id", "visit", "arm", "REF", covariates = ~ base + fast,
                    method = "Satterthwaite", vcov = "Asymptotic")
p <- stress_path(spec, path_changes(spec, reference = "J2R", m = 5), max_j = 3)
attr(p, "path")$j
```

---

stress_permute *Perturbation family: re-randomisation (permutation) test*

Description

Permutes treatment labels across subjects, optionally within strata, refits the pre-specified model each time and returns the permutation p-value for the observed contrast. The returned row carries the observed estimate and the permutation p; the null distribution is attached as `attr(, "null")`.

Usage

```
stress_permute(
  spec,
  n = 1000,
  strata = NULL,
  statistic = c("estimate", "t"),
  seed = 1,
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`n`	Number of permutations.
`strata`	Name of a subject-level column to permute within; `NULL` permutes freely.
`statistic`	`"estimate"` (the contrast; the permutation refits skip the adjusted degrees of freedom, so this is the fast choice) or `"t"` (estimate over its standard error, refitted with the spec's inference).

seed	Random seed.
cores	Number of worker processes; 1 fits sequentially.

Value

A one-row stress_rows data frame with p equal to (1 + number of |null| >= |observed|) / (n + 1).

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
stress_permute(spec, n = 19)
```

---

stress_plan *Pre-specify a stress test*

Description

A plan names the families to run and their settings before unblinding, so the stress test can be written into a statistical analysis plan and run with one call (run_plan()). Each family argument is NULL to skip the family, an empty list() to run it with its defaults, or a named list of arguments for the family function. Printing the plan gives the text a statistical analysis plan can carry.

Usage

```
stress_plan(
  spec,
  covariance = list(),
  inference = list(),
  covariates = list(),
  weights = TRUE,
  population = NULL,
  data = NULL,
  loo = list(),
  grid = NULL,
  permute = list(n = 2000),
  tipping = NULL,
  reference = list(),
  estimator = list(),
  fragility = list(),
  curve = NULL,
  estimand = NULL,
  visits = NULL,
  title = "Pre-specified stress test",
  rule = c("interval", "interval_or_direction")
)
```

Arguments

spec	A stress_spec(): the pre-specified analysis.
covariance	Arguments for stress_covariance().
inference	Arguments for stress_inference().
covariates	Arguments for stress_covariates().
weights	TRUE to run stress_weights().
population	Arguments for stress_population() (needs rules).
data	Arguments for stress_data() (needs alternatives).
loo	Arguments for stress_loo(); a list of lists runs it several times (subjects and a site variable, say).
grid	Arguments for stress_grid().
permute	Arguments for stress_permute().
tipping	Arguments for stress_tipping() (needs deltas).
reference	Arguments for stress_reference().
estimator	Arguments for stress_estimator().
fragility	Arguments for stress_fragility().
curve	Arguments for stress_curve() (needs values and modify).
estimand	Arguments for stress_estimand() (needs ice).
visits	Arguments for stress_visits().
title	A title for the plan.
rule	Classification rule passed to stress_verdict(): "interval" (default) or "interval_or_direction".

Value

An object of class stress_plan.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
plan <- stress_plan(spec, covariance = list(structures = c("us", "cs", "ar1h")),
                    inference = list(), covariates = list(), loo = list(by = "RACE"),
                permute = NULL, reference = NULL, estimator = list(estimators = "ancova"),
                    fragility = list(max_k = 2))
plan
```

stress_population *Perturbation family: analysis population*

Description

Each rule is a function of the spec's data returning a logical vector of rows to keep (a per-protocol flag, a completer definition, a cohort).

Usage

```
stress_population(spec, rules, role = "supplementary", cores = 1L)
```

Arguments

spec	A stress_spec().
rules	Named list of functions function(data) logical.
role	"supplementary" (the default: a different population is a different estimand) or "sensitivity".
cores	Number of worker processes; 1 fits sequentially.

Value

A stress_rows data frame, one row per rule.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_population(spec, list("female" = function(d) d$SEX == "Female"))
```

stress_ranktest *Perturbation family: rank tests at the contrast visit*

Description

Model-free tests of the treatment difference in the response at the contrast visit: the Wilcoxon rank-sum test (normal approximation with the tie correction and continuity correction, as SAS PROC NPAR1WAY reports it) and, if strata is given, the van Elteren test (the Cochran-Mantel- Haenszel row-mean-score statistic on modified ridit scores within strata). The rows carry a p-value and the Hodges-Lehmann shift as the estimate, with no standard error or interval; they count towards significance only.

Usage

```
stress_ranktest(spec, strata = NULL, cores = 1L)
```

Arguments

spec	A stress_spec().
strata	Name of a subject-level stratification column for the van Elteren test; NULL runs the Wilcoxon test only.
cores	Number of worker processes; 1 fits sequentially.

Value

A stress_rows data frame with one or two rows.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_ranktest(spec, strata = "RACE")
```

---

stress_reference *Perturbation family: reference-based multiple imputation*

Description

Imputes the missing responses from a multivariate-normal imputation model (an MMRM on the observed data with treatment by visit, the subject-level covariates and an unstructured covariance) under the standard reference-based assumptions (Carpenter, Roger and Kenward, 2013), refits the pre-specified model on each completed data set and pools with Rubin's rules:

Usage

```
stress_reference(
  spec,
  methods = c("MAR", "J2R", "CR", "CIR"),
  m = 25,
  seed = 1,
  delta = 0,
  arm = NULL,
  impute_covariates = NULL,
  same_cov = TRUE,
  max_tries = 3L,
  cores = 1L
)
```

Arguments

spec	A stress_spec().
methods	Assumptions to run, from "MAR", "J2R", "CR", "CIR".
m	Number of imputations.

seed	Random seed (the same seed is used for every delta). The m imputations are drawn in turn from one stream started at seed, so different seeds give unrelated imputations.
delta	Shift added to the active arm's imputed values (response units); 0 for none.
arm	Treatment level receiving the shift; defaults to the active arm.
impute_covariates	Subject-level covariates used as predictors in the imputation model (with treatment and the visit columns); NULL uses every spec covariate that is constant within subject. Covariates that vary within subject cannot be used.
same_cov	If TRUE one covariance matrix is shared by both arms in the imputation model; if FALSE each arm has its own, and the reference arm's is used for the reference-based draws.
max_tries	Bootstrap refits attempted per imputation before giving up.
cores	Number of worker processes; 1 fits sequentially.

Details

- MAR: every subject's missing values follow their own arm.
- J2R (jump to reference): an active-arm subject's visits after the last observed one take the reference arm's means.
- CR (copy reference): an active-arm subject with any missing value follows the reference arm's means at every visit.
- CIR (copy increments in reference): after the last observed visit the subject keeps their own-arm mean at that visit and then changes as the reference arm does.

Visits missing before the last observed one are treated as missing at random under every assumption. Parameter uncertainty is carried by approximate Bayesian draws: for each imputation the imputation model is refitted on a bootstrap sample of subjects (stratified by arm) and its fixed effects and covariance are used to draw the missing values from the conditional normal distribution given the subject's observed values. An optional delta is added to every imputed value of the active arm.

Imputation i uses seed seed + i on whichever worker draws it, so the result is the same for any number of cores. Two seeds less than m apart therefore share imputations (seeds 1 and 2 share 24 of 25 when m = 25); to estimate Monte Carlo error by repeating the analysis, use seeds at least m apart, such as 1000, 2000, 3000. Within an imputation the methods share the bootstrap refit and the standard-normal draws (common random numbers), so they differ only through the means they impute from: a method gives the same rows whether it is requested alone or with others, and delta is exactly a shift of the imputed values.

Value

A stress_rows data frame, one counted row per method (role = "sensitivity"), with the number of imputations and the Monte Carlo standard error in note.

References

Carpenter JR, Roger JH, Kenward MG (2013). Analysis of longitudinal trials with protocol deviation: a framework for relevant, accessible assumptions, and inference via multiple imputation. Journal of Biopharmaceutical Statistics 23, 1352-1371.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:100), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
stress_reference(spec, methods = c("MAR", "J2R"), m = 3)
```

`stress_report` *Write a stress-test report*

Description

Writes a Markdown report with the plan, the base analysis, the per-family summary table, the verdict and the standard figures (census dot plot, bubble swarm, deletion p-value strip, p-value curves where the result has rows with a parameter) as PNG files beside it.

Usage

```
stress_report(
  result,
  dir,
  name = "stress_report",
  width = 9,
  height = 5,
  dpi = 150,
  mask_ids = FALSE
)
```

Arguments

`result`	A `stress_result` from `run_plan()`.
`dir`	Directory to write into (created if needed). There is no default, so nothing is written unless a location is given.
`name`	File stem for the report and figures.
`width, height, dpi`	Figure size in inches and resolution.
`mask_ids`	Replace subject identifiers with sequence numbers.

Value

The path of the Markdown file, invisibly.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
plan <- stress_plan(spec, covariance = list(structures = c("cs", "ar1h")), inference = NULL,
               covariates = NULL, weights = FALSE, loo = list(by = "RACE"), permute = NULL,
                    reference = NULL, estimator = NULL, fragility = NULL)
stress_report(run_plan(plan), dir = tempdir())
```

---

stress_sap *Statistical-analysis-plan text for a stress test*

Description

Returns the paragraphs a statistical analysis plan can carry to pre-specify a stress test: the purpose, the pre-specified analysis, the families and their settings (from the plan), the counting rule, the classification, what will be reported, and the plan's MD5 hash so the run after unblinding can be shown to match what was pre-specified. The same text is installed as a template at `system.file("templates", "sap_stress_test.md", package = "ballast")`.

Usage

```
stress_sap(plan, file = NULL, trial = "the study")
```

Arguments

`plan`	A `stress_plan()`.
`file`	Optional path to write the text to (Markdown).
`trial`	A short name for the trial, used in the text.

Value

A character vector of paragraphs, invisibly if `file` is given.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
cat(stress_sap(standard_plan(spec, site = "RACE"), trial = "Study 001"), sep = "\n\n")
```

`stress_spec` *Define the pre-specified MMRM to stress-test*

Description

A `stress_spec` holds the analysis data and every decision of the pre-specified mixed model for repeated measures: the response, subject, visit and treatment variables, the covariates, the visit at which the treatment contrast is taken, the within-subject covariance structure, the degrees-of-freedom method, the variance estimator and the significance level. Every family function in the package takes a spec and returns rows in the schema of `stress_fit()`.

Usage

```
stress_spec(
  data,
  response,
  subject,
  visit,
  treatment,
  reference = NULL,
  active = NULL,
  covariates = NULL,
  contrast_visit = NULL,
  covariance = "us",
  method = "Kenward-Roger",
  vcov = "Kenward-Roger-Linear",
  alpha = 0.05,
  weights = c("equal", "observed"),
  label = "pre-specified",
  sas_conventions = TRUE,
  time = NULL
)
```

Arguments

`data`	A data frame in long format, one row per subject and visit.
`response`	Name of the response column (change from baseline, say).
`subject`	Name of the subject identifier column.
`visit`	Name of the visit column; coerced to a factor whose level order is the visit order.
`treatment`	Name of the treatment column (two or more arms).
`reference`	The reference (control) level of `treatment`. Defaults to the first factor level.
`active`	The treatment level compared with the reference. Defaults to the first non-reference level. With three or more arms, change it with `spec_update()` to stress-test each comparison in turn.

covariates	Fixed-effect covariates besides treatment and visit, as a one-sided formula (~ base + site) or a character vector of term labels. Character and logical columns are treated as class variables.
contrast_visit	The visit level at which the treatment contrast is estimated. Defaults to the last visit level.
covariance	Covariance structure name as used by mmrm: "us", "ar1", "ar1h", "cs", "csh", "toep", "toeph", "ad", "adh", or "sp_exp" (spatial exponential, the SAS SP(POW) structure, on the continuous time variable).
method	Degrees-of-freedom method passed to mmrm::mmrm_control(): "Satterthwaite", "Kenward-Roger", "Residual" or "Between-Within".
vcov	Variance estimator passed to mmrm::mmrm_control(): "Asymptotic", "Empirical", "Empirical-Jackknife", "Empirical-Bias-Reduced", "Kenward-Roger" or "Kenward-Roger-Linear". The SAS DDFM=KR default corresponds to method = "Kenward-Roger" with vcov = "Kenward-Roger-Linear".
alpha	Two-sided significance level.
weights	How class covariates are weighted in the least-squares means: "equal" (every level equally, SAS LSMEANS default) or "observed" (the design is averaged over the covariate values of the analysis rows, as SAS OBSMARGINS and g-computation do).
label	Label for the base analysis in output rows.
sas_conventions	If TRUE, the between-within degrees of freedom for a sandwich-variance contrast are computed as SAS PROC MIXED does when EMPIRICAL is combined with a DDFM it cannot honour (observations minus subjects minus the rank of the within-subject design columns).
time	Name of a numeric time column for the "sp_exp" structure (days from randomisation, say; mmrm converges more readily with time rescaled to hundreds of days).

Details

The fixed effects are covariates + treatment + visit + treatment:visit; the treatment contrast at contrast_visit is the difference of least-squares means (active minus reference) with class covariates weighted equally over their levels and continuous covariates at their means, the convention of SAS LSMEANS.

Value

An object of class stress_spec.

Examples

```
spec <- stress_spec(mmrm::fev_data, response = "FEV1", subject = "USUBJID",
                    visit = "AVISIT", treatment = "ARMCD", reference = "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL, contrast_visit = "VIS4")
spec
```

`stress_spec_adam` *Build a stress spec from an ADaM basic data structure*

Description

Subsets a CDISC ADaM BDS data set (ADEFF, ADQS and the like) to one parameter, the analysis-flagged post-baseline records, orders the visits by their numeric code and builds the spec with the baseline value as a covariate by default.

Usage

```
stress_spec_adam(
  bds,
  paramcd,
  response = "CHG",
  subject = "USUBJID",
  visit = "AVISIT",
  visitn = "AVISITN",
  treatment = "TRTP",
  reference = NULL,
  active = NULL,
  covariates = ~BASE,
  contrast_visit = NULL,
  analysis_flag = "ANL01FL",
  drop_baseline = TRUE,
  filter = NULL,
  covariance = "us",
  method = "Kenward-Roger",
  vcov = "Kenward-Roger-Linear",
  alpha = 0.05,
  weights = "equal",
  label = paste(paramcd, response)
)
```

Arguments

`bds`	A BDS data frame.
`paramcd`	The `PARAMCD` value to analyse.
`response`	Response column, usually `"CHG"` or `"PCHG"`.
`subject`, `visit`, `visitn`, `treatment`	Column names (`USUBJID`, `AVISIT`, `AVISITN`, `TRTP` by default).
`reference`, `active`, `contrast_visit`, `covariance`, `method`, `vcov`, `alpha`, `weights`, `label`	As in `stress_spec()`.
`covariates`	Covariates; the default is the baseline value `BASE`.

analysis_flag	Name of a flag column kept where equal to "Y"; NULL keeps all records.
drop_baseline	Drop records flagged ABLFL == "Y" or with a non-positive visitn.
filter	Optional function of the data returning the rows to keep, applied after the other subsets (an analysis-phase restriction, say).

Value

A stress_spec.

Examples

```
bds <- transform(mmrm::fev_data, PARAMCD = "FEV1", CHG = FEV1 - FEV1_BL, BASE = FEV1_BL,
                 AVISITN = VISITN, TRTP = ARMCD, ANL01FL = "Y")
stress_spec_adam(bds, "FEV1", response = "CHG", visit = "AVISIT", reference = "PBO")
```

---

stress_spec_mmrm *Build a stress spec from a fitted mmrm model*

Description

Takes the data, response, subject, visit, covariance structure, degrees-of-freedom method and variance estimator from an mmrm fit. The treatment variable is the factor that interacts with the visit variable unless given.

Usage

```
stress_spec_mmrm(
  fit,
  treatment = NULL,
  reference = NULL,
  active = NULL,
  contrast_visit = NULL,
  alpha = 0.05,
  weights = "equal",
  label = "pre-specified",
  sas_conventions = TRUE
)
```

Arguments

fit	An object of class mmrm.
treatment	Name of the treatment column; NULL finds the factor in an interaction with the visit variable.
reference, active, contrast_visit, alpha, weights, label, sas_conventions	As in stress_spec().

Value

A `stress_spec`.

Examples

```
fit <- mmrm::mmrm(FEV1 ~ RACE + SEX + FEV1_BL + ARMCD * AVISIT + us(AVISIT | USUBJID),
                  data = mmrm::fev_data)
stress_spec_mmrm(fit, reference = "PBO")
```

`stress_tables` *Write regulatory-style tables and the census*

Description

Writes the per-family summary, the rows that crossed alpha or left the interval, the deletion-search path and the full census. CSV files are always written; with `flextable` and `officer` installed, a Word document with the tables in a standard clinical-report layout is written too, and an RTF version of each table.

Usage

```
stress_tables(
  result,
  dir,
  name = "stress",
  formats = c("csv", "docx", "rtf"),
  digits = 3,
  mask_ids = FALSE
)
```

Arguments

`result`	A `stress_result` from `run_plan()`.
`dir`	Directory to write into (created if needed). There is no default, so nothing is written unless a location is given.
`name`	File stem.
`formats`	Any of `"csv"`, `"docx"`, `"rtf"`.
`digits`	Decimal places in the formatted tables.
`mask_ids`	Replace subject identifiers with sequence numbers.

Value

The paths written, invisibly.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL,
                    covariance = "cs", method = "Satterthwaite", vcov = "Asymptotic")
plan <- stress_plan(spec, covariance = list(structures = c("cs", "ar1h")), inference = NULL,
               covariates = NULL, weights = FALSE, loo = list(by = "RACE"), permute = NULL,
                    reference = NULL, estimator = NULL, fragility = NULL)
stress_tables(run_plan(plan), dir = tempdir(), formats = "csv")
```

`stress_tipping` *Perturbation family: delta-adjusted multiple imputation (tipping point)*

Description

Imputes the missing responses by fully conditional specification (regression method, one column per visit, subject-level covariates and treatment as predictors), adds `delta` to every imputed value of the active arm, refits the pre-specified model on each completed data set and pools with Rubin's rules. `delta = 0` is the missing-at-random multiple-imputation analysis and is a counted alternative; other deltas are tipping-point searches (`kind = "search"`). Rows carry the delta in `param`.

Usage

```
stress_tipping(
  spec,
  deltas = seq(0, -10, by = -2),
  m = 25,
  seed = 1,
  arm = NULL,
  shift_visits = NULL,
  impute_covariates = NULL,
  nbiter = 10,
  shift_every = TRUE,
  cores = 1L
)
```

Arguments

`spec`	A `stress_spec()`.
`deltas`	Shifts added to the imputed values of the active arm, in response units.
`m`	Number of imputations.
`seed`	Random seed (the same seed is used for every delta). The `m` imputations are drawn in turn from one stream started at `seed`, so different seeds give unrelated imputations.
`arm`	Treatment level receiving the shift; defaults to the active arm.

`shift_visits` Visit levels whose imputed values are shifted; `NULL` shifts every visit.

`impute_covariates`
Subject-level covariates used as predictors in the imputation model (with treatment and the visit columns); `NULL` uses every spec covariate that is constant within subject. Covariates that vary within subject cannot be used.

`nbiter` Burn-in iterations of the FCS sampler.

`shift_every` If `TRUE` the shift is applied at every draw (SAS `MNAR ADJUST` under FCS); if `FALSE` only to the final imputation.

`cores` Number of worker processes; 1 fits sequentially.

Details

Covariates that vary within subject are excluded from the imputation model and are kept in the analysis model; subject-visit rows absent from the data are created for the imputed response with those covariates missing, so they drop out of the refit.

Value

A `stress_rows` data frame, one row per delta; `note` records the number of imputations and the Monte Carlo standard error of the pooled estimate.

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:80), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
stress_tipping(spec, deltas = c(0, -5), m = 3)
```

`stress_verdict` *Classify the census: robust, sensitive or fragile, and marginal*

Description

Applies the definitions: *fragile* if any counted row lies outside the base 95% confidence interval (the margin-of-error rule, the default from 0.7.0); otherwise *sensitive* if any counted row changes significance at `alpha`; otherwise *robust*. *Marginal* is a separate flag on the base p-value, true when it lies within `alpha * marginal_band`.

Usage

```
stress_verdict(
  census,
  marginal_band = c(0.5, 2),
  rule = c("interval", "interval_or_direction")
)
```

Arguments

census	A stress_census.
marginal_band	Multiplicative band around alpha for the marginal flag; the default flags a base p between alpha / 2 and 2 * alpha.
rule	"interval" (default): fragile means outside the base confidence interval. "interval_or_direction": outside the interval or a reversed direction (the rule of ballast 0.6 and earlier).

Details

For a significant base result the interval excludes zero, so a refit that reverses the direction always lies outside it and the two rules agree. They differ only for a non-significant base result: there a sign change of an estimate near zero is a small move inside the interval, and the default rule does not call it fragile. rule = "interval_or_direction" restores the 0.6 behaviour, in which any reversal is fragile.

Value

A list of class stress_verdict: stability over every counted row, stability_sensitivity over the rows that keep the estimand (role = "sensitivity"), marginal, rule, the counts behind them, and the rows that drove the classification.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
cen <- stress_census(spec, stress_covariance(spec, c("us", "cs")), stress_covariates(spec))
v <- stress_verdict(cen)
v$stability
v
```

---

stress_visits *Perturbation family: the contrast at every visit*

Description

Fits the pre-specified model once per visit and reports the treatment contrast at each; the contrast at the pre-specified visit is flagged is_base. A different visit is a different estimand, so the rows are role = "supplementary".

Usage

```
stress_visits(spec, visits = NULL, cores = 1L)
```

Arguments

spec	A stress_spec().
visits	Visit levels; NULL for all.
cores	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame, one row per visit.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_visits(spec)
```

`stress_weights` *Perturbation family: least-squares-mean weighting*

Description

Refits the pre-specified model and forms the treatment contrast with the other weighting of the class covariates: equal weights over levels (SAS `LSMEANS` default) versus the observed covariate distribution of the analysis rows (SAS `OBSMARGINS`, g-computation). The point estimate differs only when a class covariate interacts with treatment or visit; the two rows otherwise coincide.

Usage

```
stress_weights(spec, cores = 1L)
```

Arguments

`spec`	A `stress_spec()`.
`cores`	Number of worker processes; 1 fits sequentially.

Value

A `stress_rows` data frame with two rows, one flagged `is_base`.

Examples

```
spec <- stress_spec(mmrm::fev_data, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO",
                    covariates = ~ RACE + SEX + FEV1_BL)
stress_weights(spec)
```

summary.stress_census *Summarise a census by family*

Description

Summarise a census by family

Usage

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

Arguments

object	A stress_census.
...	Unused.

Value

A data frame with one row per family, kind and role: counted analyses, number significant, number that crossed alpha, number outside the base confidence interval, number reversing direction, the range of percent of base, the largest shift in base standard errors, and counted. Diagnostic refits (kind = "diagnostic") get their own line with counted = FALSE so a reader sees they were fitted; they are left out of the closing rows, which total the sensitivity analyses (same estimand) and all counted rows.

tipping_point *Find where a p-value curve crosses alpha*

Description

For rows that carry a param value (from stress_tipping() or stress_curve()), interpolates linearly between adjacent parameter values to find where the p-value crosses alpha, separately for each family.

Usage

```
tipping_point(rows, alpha = NULL)
```

Arguments

rows	A stress_rows data frame or a stress_census.
alpha	Significance level; defaults to the rows' own alpha.

Value

A data frame with one row per family: `family`, `n` points, `param_min`, `param_max`, `p_min`, `p_max`, `crossing` (the interpolated parameter value, `NA` when the curve does not cross) and `direction` (`"up"` when p rises through alpha with increasing parameter, `"down"` otherwise).

Examples

```
small <- mmrm::fev_data[mmrm::fev_data$USUBJID %in% paste0("PT", 1:60), ]
spec <- stress_spec(small, "FEV1", "USUBJID", "AVISIT", "ARMCD", "PBO", covariates = ~ FEV1_BL)
shift_pt3 <- function(d, v) { k <- d$USUBJID == "PT3"; d$FEV1[k] <- d$FEV1[k] + v; d }
tipping_point(stress_curve(spec, values = c(-20, 0, 20), modify = shift_pt3))
```

Index

default_inference, 3
default_inference(), *38*

mask_ids, 4
mask_ids(), *10*
mmrm::mmrm_control(), *50*

path_changes, 4
path_changes(), *40*
plot.stress_census, 6
plot.stress_design, 6
plot.stress_design(), *28*
plot_forest, 7
plot_pcdf, 8
plot_pcurve, 9
plot_pstrip, 10
plot_swarm, 11
plot_swarm(), *7*, *14*
print.stress_design *(*stress_design*)*, 26

run_plan, 13
run_plan(), *12*, *14*, *37, 38*, *42*, *47*, *53*
run_plans, 13
run_plans(), *8*, *12*

simulate_rare_trial, 14
simulate_trial, 16
simulate_trial(), *14*, *16*, *21*
spec_update, 18
spec_update(), *49*
standard_plan, 18
standard_plan(), *31*
stress_calibrate, 20
stress_calibrate(), *28*
stress_census, 21
stress_census(), *31*
stress_covariance, 22
stress_covariance(), *43*
stress_covariates, 23
stress_covariates(), *43*
stress_curve, 24
stress_curve(), *9*, *43*, *58*
stress_data, 25
stress_data(), *43*
stress_design, 26
stress_design(), *7*
stress_direction, 28
stress_estimand, 29
stress_estimand(), *19*, *43*
stress_estimator, 31
stress_estimator(), *43*
stress_explain, 32
stress_fit, 33
stress_fit(), *24*, *49*
stress_fragility, 34
stress_fragility(), *4*, *29*, *41*, *43*
stress_grid, 36
stress_grid(), *43*
stress_html, 37
stress_inference, 38
stress_inference(), *3*, *43*
stress_log, 38
stress_loo, 39
stress_loo(), *43*
stress_path, 40
stress_path(), *4*
stress_permute, 41
stress_permute(), *43*
stress_plan, 42
stress_plan(), *13*, *48*
stress_population, 44
stress_population(), *43*
stress_ranktest, 44
stress_reference, 45
stress_reference(), *30*, *43*
stress_report, 47
stress_sap, 48
stress_sap(), *28*
stress_spec, 49

stress_spec(), *4, 5*, *19*, *21–25*, *27*, *29–36*, *38–41*, *43–45*, *51, 52*, *54*, *56, 57*
stress_spec_adam, 51
stress_spec_mmrm, 52
stress_tables, 53
stress_tipping, 54
stress_tipping(), *9*, *43*, *58*
stress_verdict, 55
stress_verdict(), *21*, *27*, *29*, *43*
stress_visits, 56
stress_visits(), *43*
stress_weights, 57
stress_weights(), *43*
summary.stress_census, 58

tipping_point, 58