Package: grassr (via r-universe)

November 14, 2026

Type Package

Title Context-Conditioned Reporting for Binary Rater Reliability

Version 0.8.0

Description The name abbreviates ``Guide for Rater Agreement under Structural Skew''. Generates a Report Card for rater reliability on binary outcomes from an N x k subject-by-rater rating matrix, on both the inter-rater and intra-rater axes. Each panel coefficient is positioned on a data-generating-process-calibrated reference surface conditioned on the study's rater count, sample size, and prevalence, yielding a pooled percentile (the coefficient's position within the design's achievable agreement range) together with a consistency band on panel quality: the quality levels whose sampling distributions are consistent with the observed value at that design. The panel coefficients are the prevalence-adjusted bias-adjusted kappa (PABAK) of Byrt, Bishop, and Carlin (1993) <doi:10.1016/0895-4356(93)90018-V>, the first-order agreement coefficient (AC1) of Gwet (2008) <doi:10.1348/000711006X126600>, the multi-rater kappa of Fleiss (1971) <doi:10.1037/h0031619>, and the observed intraclass correlation. A cross-coefficient discordance diagnostic (delta-hat) reports the spread of the coefficients' implied panel qualities and flags panels for which no single coefficient is a stable summary by the spread's percentile on a matched null distribution; for such divergent panels the report routes to a pairwise PABAK matrix and per-rater sensitivity and specificity recovered from the latent-class model of Dawid and Skene (1979) <doi:10.2307/2346806>, with the two-rater bounds of Hui and Walter (1980) <doi:10.2307/2530508>.

License MIT + file LICENSE

Encoding UTF-8

LazyData false

Depends R (>= 4.0)

Imports stats

Suggests boot, ggplot2, broom, future, future.apply, irr, irrCAC, patchwork (>= 1.2.0), progressr, knitr, lme4 (>= 1.1-30), rmarkdown, testthat (>= 3.0.0), viridisLite

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

URL https://defense031.github.io/grassr/, https://github.com/defense031/grassr

BugReports https://github.com/defense031/grassr/issues

Repository https://defense031.r-universe.dev

Date/Publication 2026-11-14 03:09:09 UTC

RemoteUrl https://github.com/defense031/grassr

RemoteRef HEAD

RemoteSha 331e074da150c07ff31141566242cb4213612d87

Contents

as.data.frame.grass_card . . . 3
as.data.frame.grass_surface_position . . . 3
check_asymmetry . . . 4
format.grass_card . . . 6
grass_compute . . . 7
grass_plot . . . 8
grass_power . . . 9
grass_reference . . . 11
grass_reference_table . . . 12
grass_report . . . 12
grass_roadmap . . . 15
latent_class_fit . . . 17
obs_krippendorff_alpha . . . 19
pairwise_agreement . . . 19
plot.grass_card . . . 21
plot.grass_power . . . 22
plot.grass_result . . . 22
plot_surface . . . 23
position_on_surface . . . 25
print.grass_card . . . 29
reset_grass_warnings . . . 30
scale_grass_metric . . . 30
summary.grass_card . . . 31
theme_grass . . . 31

Index **32**

as.data.frame.grass_card

Coerce a grass_card to a tidy data.frame (panel; per_rater appended on divergent)

Description

Returns the panel of observed coefficients with surface-position metadata as a data.frame. Each row carries the v0.7.1 panel columns (`surface_percentile`, `band_lo` / `band_hi` / `band_open_low` / `band_open_high`, `q_hat`, `se_q_hat`, `clamped`, `reference_used`, `in_delta_hat`), an `is_primary` flag, and the panel-level delta diagnostics recycled across rows (`delta_hat` implied-quality spread in pp, `delta_percentile`, `delta_flag`, and the matched-null cell `matched_null_k` / `matched_null_N` / `matched_null_q`). Percentiles are rounded to one decimal: the pooled percentile is computed on the thirteen calibrated quality levels and moves in steps between them at large `N`, so finer digits are not resolution. When the cross-coefficient flag is `divergent` and the per-rater latent-class table is populated, returns a list `c(panel = ..., per_rater = ...)` so tidy consumers can pivot the per-rater rows separately.

Usage

```
## S3 method for class 'grass_card'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	A `grass_card` object.
`row.names, optional`	Standard `as.data.frame` arguments.
`...`	Unused.

Value

A `data.frame` (the panel) when `x$per_rater` is `NULL` or empty. When the divergent flag fires, returns a list with `panel` and `per_rater` data frames.

as.data.frame.grass_surface_position

Coerce a grass_surface_position to a one-row data.frame

Description

Coerce a grass_surface_position to a one-row data.frame

Usage

```
## S3 method for class 'grass_surface_position'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	A grass_surface_position object.
row.names, optional, ...	Standard arguments; ignored except row.names.

Value

A one-row data.frame summarising the position: pooled percentile, consistency-band endpoints (band_lo, band_hi, band_open_low, band_open_high), implied quality, and method.

---

check_asymmetry *Cross-coefficient panel asymmetry diagnostic*

Description

check_asymmetry() takes an N x k binary ratings matrix and returns a scalar delta_hat (in quality percentage points, "pp"): the max-min spread of the *implied panel qualities* across the three agreement coefficients (PABAK, mean AC1, Fleiss kappa). Each coefficient inverts to its own q_hat on the shared (q, pi_+) reference; if the calibration DGP held exactly, all three would imply the same quality, so the spread measures cross-coefficient model discordance in interpretable units of quality. ICC never enters delta_hat, since its reference is distribution-sensitive in ways the agreement family is not.

Usage

```
check_asymmetry(ratings, axis = c("inter", "intra"), fit_icc = TRUE, ...)
```

Arguments

ratings	User input: an N x k binary matrix, an N x k data.frame whose columns are 0/1 / logical / 2-level factor, or a list of two equal-length 0/1 vectors (k = 2 paired form); the Quick start section of vignette("grassr") lists the accepted shapes.
axis	"inter" (default) or "intra". Selects the surface family.
fit_icc	If FALSE, skip the lme4::glmer fit behind icc and drop ICC from the panel. icc never enters delta_hat, and the fit draws no random numbers, so a caller that reports only delta_hat and the implied qualities gets identical results at a fraction of the cost. The Monte Carlo null loop sets this; interactive users should not.
...	Forwarded to position_on_surface() (e.g. reference_type).

Details

`delta_hat` is a *split-bias* detector. It fires when raters tilt in different directions across (Se, Sp) – e.g., one rater high-Se / low-Sp, another high-Sp / low-Se – because the three coefficients respond to heterogeneous per-rater behavior differently and end up implying different panel qualities. The framework's other failure mode, *shared/uniform bias* (every rater tilts the same direction, e.g., a panel trained on one protocol all favoring specificity over sensitivity), produces uniform degradation across coefficients: small `delta_hat`, low implied quality. Shared bias is detected by the panel's implied quality (and its consistency band), not by `delta_hat`. A divergent flag therefore identifies a specific kind of disagreement – cross-coefficient inversion from heterogeneous rater behavior – and routes the user to the per-rater pairwise PABAK matrix and pooled-reference (Se_tilde, Sp_tilde) diagnostic.

Each coefficient is positioned on its reference surface via `position_on_surface()`, which reports its implied `q_hat`. The panel diagnostic is `delta_hat = max(q_hat) - min(q_hat)` (in pp of quality), computed over agreement-family coefficients whose observed value falls within the achievable range of their reference surface (see *Surface-envelope clamp* below).

Value

An S3 object of class `grass_asymmetry_panel` with fields:

- `delta_hat`: scalar implied-quality spread, in pp of quality
- `delta_percentile`: `delta_hat`'s percentile on the matched (k, N, q_hat) null ECDF (`NA` if the null is uncalibrated at the design)
- `flag`: one of `"aligned"`, `"caution"`, `"divergent"`
- `matched_null`: list describing the matched null cell (`k`, `N`, `q`, `prev` — the bridged true-prevalence estimate the lookup conditioned on, `prev_apparent` — the panel's raw positive rate, `prev_bridged`, `q_hat_panel`, `n_draws`, `snapped`, `interpolated`, `unstable_tail`), or `NULL` if uncalibrated
- `thresholds`: named numeric vector of the implied (caution, divergent) pp cuts (95th/99th of the matched null)
- `thresholds_source`: one of `"matched_null_ecdf"`, `"not_applicable_k2"`, `"not_calibrated"`
- `panel`: data.frame with `coefficient`, `observed`, `implied_q`, `percentile_pp` (pooled percentile), `clamped`, `in_delta_hat`
- `notes`: character vector of unique caveats from the underlying surface positioning calls (e.g. nearest-neighbor gaps, ICC unavailability, matched-null provenance)

Surface-envelope clamp (v0.2.1+)

If an observed coefficient value falls outside the achievable range of its reference surface at the study's design (`pi_hat, k, N`), the inversion to `q_hat` clamps to the boundary. Including such clamped implied qualities in the max-min `delta_hat` would inflate the panel spread purely because of the clamp, not because the panel disagrees on quality. Since v0.2.1 the function therefore *excludes* clamped coefficients from `delta_hat` whenever at least two unclamped agreement-family coefficients remain. The affected coefficients are still shown in the returned `panel` data.frame with `clamped = TRUE`, and a note in `$notes` names which coefficients were excluded. This matters most often for ICC (which never enters `delta_hat` anyway) and at designs beyond the bundled reference range, where a coefficient clamps to the achievable boundary. If fewer than two unclamped

agreement-family coefficients remain, `delta_hat` falls back to the raw spread including clamped values and the note records the fallback.

Flag from the matched null (v0.7.0/0.7.1)

The per-(k, N) size-alpha threshold table is retired. The flag is `delta_hat`'s percentile on the null distribution of `delta_hat` at the matched (k, N, q_hat) cell of the bundled `delta_null_ecdf`, with the cut convention >= 95th caution, >= 99th divergent. The panel's `q_hat` is resolved first (median of the agreement-family implied qualities), then the matched null cell is looked up; the reported `thresholds` carry the implied pp cuts (95th/99th of that null) as context, and `thresholds_source` records how the flag was resolved. The three flags are:

- `aligned` (below the 95th percentile of the matched null): the panel agrees on the implied quality. Any single coefficient is a stable summary; the primary coefficient (Table 2) carries the headline.
- `caution` (>= 95th, < 99th): the panel is mildly inconsistent. Report the primary coefficient with a caution flag and the `delta_hat` value.
- `divergent` (>= 99th): no single coefficient is a stable summary. Use `latent_class_fit()` to recover per-rater `(Se, Sp)` and report those instead.

The new `check_asymmetry(ratings, ...)` signature replaces the v0.1.x `check_asymmetry(se, sp, ...)` per-rater signature. For Calling `check_asymmetry()` with the pre-0.2.0 `se = ...` and `sp = ...` arguments is an error; per-rater sensitivity and specificity come from `latent_class_fit()`.

See Also

`latent_class_fit()` for the divergent-branch recovery of per-rater `(Se, Sp)`; `position_on_surface()` for the underlying surface positioning.

Examples

```
set.seed(1)
# Build a 5x200 symmetric panel -- should print as 'aligned'.
Y <- matrix(rbinom(5 * 200, 1, 0.30), nrow = 200, ncol = 5)
check_asymmetry(Y)
```

`format.grass_card` *Format a grass_card as a character vector*

Description

Returns the four-field Report Card rendered as a character vector, one line per element. Used by `print.grass_card()` and by downstream embedders (Markdown, knitr) that want the lines as data.

Usage

```
## S3 method for class 'grass_card'
format(x, digits = 2, ...)
```

Arguments

x	A grass_card object (returned by grass_report()).
digits	Numeric. Decimals for observed coefficient values. Default 2.
...	Ignored.

Value

A character vector (one element per line).

---

grass_compute *Compute the full inter-rater agreement panel*

Description

Computes Cohen's kappa, PABAK, Gwet's AC1, positive and negative agreement, Byrt prevalence and bias indices, and two confidence intervals for kappa (Wald, and a logit-transformed Wilson-type interval).

Usage

```
grass_compute(
  data,
  format = c("wide", "matrix", "long", "paired"),
  positive = NULL,
  ...
)
```

Arguments

data	Input data. Form depends on format: • "matrix" – a 2x2 integer count matrix with R1 rows, R2 columns. • "wide" – a data.frame with exactly two rater columns (or pass rater_cols = c("r1", "r2") if the data contains extra columns). • "long" – a data.frame with subject, rater, and rating columns (names configurable via subject =, rater =, rating =). Must contain exactly two distinct raters. • "paired" – a list of two equal-length rating vectors, a 2-column matrix, or a 2-column data.frame.
format	One of "matrix", "wide", "long", "paired".
positive	Optional character string naming the level to treat as the positive (=1) class. Defaults to "yes" if present among levels, then "1" / "true" / "positive" / "case", then first-encountered.
...	Format-specific arguments: rater_cols for "wide"; subject, rater, rating for "long".

Value

A `grass_metrics` S3 object.

Deprecated

`grass_compute()` is deprecated in grass 0.2.0. The new headline API is `grass_report(ratings = Y)` returning a `grass_card` object. The full coefficient panel is available via `summary()` or `as.data.frame()`. See `vignette("grassr")` and `grass_report()`.

Examples

```
# 2x2 counts (Cohen 1960 Table 1)
tab <- matrix(c(88, 10, 14, 88), nrow = 2,
              dimnames = list(R1 = c("0", "1"), R2 = c("0", "1")))
grass_compute(tab, format = "matrix")

# Paired vectors
r1 <- c(1, 1, 0, 0, 1, 0, 1)
r2 <- c(1, 0, 0, 0, 1, 1, 1)
grass_compute(list(r1, r2), format = "paired")
```

`grass_plot` *Plot a grass object*

Description

Dispatches to `plot.grass_result` or `plot.grass_reference`.

Usage

```
grass_plot(x, ...)
```

Arguments

`x`	A grass object.
`...`	Passed through to the class-specific plot method.

Value

A ggplot object.

grass_power *Plan a rating study: what a design can show*

Description

grass_power() plans a study's size. Assume a prevalence and a rough rater quality, and it says what a design of a given size can show about the panel, and how many more subjects or raters it would take to show more. The result describes the design, how much a study of that size can learn about its raters.

Usage

```
grass_power(
  metric,
  q = NULL,
  q0 = NULL,
  target = NULL,
  pi_hat = NULL,
  prevalence = NULL,
  k = NULL,
  N = NULL,
  power = NULL
)
```

Arguments

metric	One of "pabak", "fleiss_kappa", "mean_ac1", "icc".
q	Panel quality, the probability of a correct call on the Se = Sp diagonal, in [0.55, 0.99].
q0	The lower edge of the quality resolution the study is planned for, in [0.55, 0.99]: the study is sized to show that a panel of quality q is above q0. Give q0 or target, not both.
target	A fixed coefficient value to reach. Give q0 or target, not both.
pi_hat	Observed positive rate, the share of all N x k ratings that are positive, in [0.05, 0.95]. This is what the card measures and what the surfaces are indexed by. Give pi_hat or prevalence, not both.
prevalence	True positive rate of the finding, in [0.01, 0.99]. A planner usually has this rather than pi_hat. Under the symmetric reference model a panel of quality q turns it into pi_hat = prevalence * q + (1 - prevalence) * (1 - q), and the function does that conversion, at each candidate q when q is the one being solved. The result carries both rates. A pi_hat that no panel of quality q can produce at any prevalence is refused.
k	Number of raters. Snaps to the nearest calibrated rater count (2, 3, 5, 8, 15, 25), as the surfaces do everywhere.
N	Number of subjects in [15, 1000].
power	Probability, in (0, 1).

Details

A study sized to show that a panel of quality q is above q0 returns a 95% consistency band on quality whose lower edge lies above q0. q0 sets the resolution of the plan. A panel weaker than assumed returns a band that is lower and about as wide, so the study reports the quality it finds at the precision it was planned for. The convention follows stats::power.t.test(). Fix four of q, the positive rate (prevalence or pi_hat), k, N, power, leave one NULL, and the function solves for it.

Whether more subjects or more raters raises power depends on prevalence. At balanced prevalence a few more raters do the work of subjects. At a rare or very common finding they do not, because a small sample holds only a handful of the minority class, and the answer is more subjects.

The function also accepts target, a fixed coefficient value, for a threshold imposed from outside (a journal's or regulator's band). A fixed coefficient value means something different at each design. When the value a panel of quality q produces at the design is below target, power falls with N. Give q0 or target, not both.

Value

An object of class grass_power: the design quantities with the solved one filled in, solved naming it, mode ("quality" or "value"), feasible, reason (when not feasible), expected (the median coefficient a panel of quality q produces at the design, in "value" mode), curve (power across the solved variable's range, the data plot() draws), and notes from the surface lookup.

How it is computed

Every quantity is a direct read of the quality sweep that position_on_surface() returns, p(q) = P(coefficient <= c | q, design). Nothing is simulated at call time. For q0, by test inversion the lower end of the band is above q0 exactly when the observed coefficient exceeds the 97.5th percentile of the q0 distribution, c0, so power = 1 - p_q(c0). For target, power = 1 - p_q(target).

Solving for N or k

The smallest value on the calibrated surface at which power is reached. Below about 25 subjects the curve can step or dip between adjacent sample sizes, because the coefficient takes few distinct values there. When none reaches it the result carries NA, feasible = FALSE, and the reason, including the best power any design on the surface reaches.

Solving for the positive rate

Leave both pi_hat and prevalence NULL. The result is the range of observed positive rates over which power is reached, solution holds its two endpoints, and prevalence holds the same range converted at q.

Examples

```
# Raters assumed near quality 0.90, prevalence 10%, three raters: how
# many subjects to show the panel is above quality 0.80, 80% power?
pw <- grass_power("fleiss_kappa", q = 0.90, q0 = 0.80, prevalence = 0.10,
                  k = 3, power = 0.80)
pw
```

```
if (requireNamespace("ggplot2", quietly = TRUE)) plot(pw)
# Leave `prevalence` out to get the range of positive rates over which
# a fixed design keeps that power; give `target =` instead of `q0` to
# size against a fixed coefficient value. Both are shown in the vignette.
```

`grass_reference` *Look up prevalence-conditioned reference values for agreement metrics*

Description

Returns the prevalence-conditioned reference values for Cohen's kappa, PABAK, and Gwet's AC1 at a given prevalence and a chosen rater-quality band. The reference is the analytical Youden-J-optimal metric value on the Se = Sp diagonal at the chosen `reference_level`, under conditional independence of the two raters given the true class.

Usage

```
grass_reference(prevalence, reference_level = 0.85, quality = NULL)
```

Arguments

`prevalence`	Numeric in `[0, 1]`. Values outside `[0.01, 0.99]` are clamped with a warning.
`reference_level`	One of `0.70`, `0.80`, `0.85`, `0.90`. Default `0.85`.
`quality`	Deprecated alias. `"high"` -> `reference_level` = `0.85`, `"medium"` -> `0.70`. Emits a one-time soft deprecation warning.

Details

In the paper these are called the *prevalence-conditioned thresholds*. The package uses the name `reference` because its API reports the values alongside prevalence index and bias index rather than applying them as cutoffs.

Value

A `grass_reference` S3 object containing a 3-row data.frame (one row per metric) with columns `metric`, `reference`, `J`, `reference_level`, `quality`, `prevalence`. The `quality` column retains the legacy `"high"`/`"medium"` label for 0.85 and 0.70 bands and is `NA_character_` for the new 0.80 / 0.90 bands.

Deprecated

`grass_reference()` is deprecated in grass 0.2.0. The percentile machinery in `position_on_surface()` and the new `grass_report()` supersede the fixed reference-curve lookup. See `vignette("grassr")` for the v0.2.0 surface-position workflow.

Examples

```
grass_reference(0.5)
grass_reference(0.1, reference_level = 0.70)
```

`grass_reference_table` *Return the full internal reference table*

Description

For inspection, custom interpolation, or reproducibility. In the paper this is the prevalence-conditioned threshold table.

Usage

```
grass_reference_table(reference_level = NULL)
```

Arguments

`reference_level`
: Optional numeric band (`0.70`, `0.80`, `0.85`, or `0.90`). If supplied, only that band is returned. Default `NULL` returns all four bands in long form.

Value

A long-form data.frame with columns `prevalence`, `reference_level`, `metric`, `reference`, and `J`.

Examples

```
head(grass_reference_table())
head(grass_reference_table(reference_level = 0.85))
```

`grass_report` *Generate a GRASS Report Card from a rating matrix*

Description

`grass_report()` is the package's headline entry point. It takes an `N x k` binary rating matrix and returns a four-field Report Card: the sample summary (`k, N, pi_hat`), the primary coefficient and its surface position, the cross-coefficient asymmetry diagnostic `delta_hat` and flag, and (when `flag == "divergent"`) the per-rater latent-class fit. The full panel of coefficients, pooled percentiles, consistency bands on quality, and reference-surface artifacts ride along on the same object for `summary()`, `as.data.frame()`, and `plot()` access.

Usage

```
grass_report(
  ratings,
  axis = c("inter", "intra"),
  metric = "auto",
  bootstrap_B = 1000L,
  bootstrap_delta_B = 0L,
  verbose = FALSE,
  ...
)
```

Arguments

ratings	User input: an N x k binary matrix, an N x k data.frame whose columns are 0/1, logical, or two-level factor or character (positive level named yes, TRUE, positive, present, or case), or a list of two equal-length 0/1 vectors (k = 2 paired form); the Quick start section of vignette("grassr") lists the accepted shapes. Rows must be independent subjects, one row each: the calibration assumes every row is a new subject, and stacking repeated measurements of one subject as rows overstates the effective sample size (analyze one card per occasion instead; vignette("grassr"), section "Quick start").
axis	One of "inter" (default) or "intra". Both axes use the same reference panels; on the intra axis the columns are one rater's occasions and ICC is marked primary.
metric	One of "auto" (default; calls pick_primary_coefficient() by prevalence), "pabak", "ac1", "fleiss_kappa", "icc". Selects which coefficient is the headline in the printed Report Card; the full panel is always populated. (Krippendorff's alpha left the Report Card panel at v0.6.0; it coincides with Fleiss' kappa in the binary fully-crossed case. Use obs_krippendorff_alpha() or position_on_surface() to compute it manually.)
bootstrap_B	Integer; bootstrap replicates for the divergent-branch latent-class CIs. Default 1000L. Set lower for fast tests.
bootstrap_delta_B	Integer; subject-resampling replicates for the optional bootstrap distribution of delta_hat. Default 0L (off); values below 50L are treated as off.
verbose	Logical; emit progress messages on long calls. Default FALSE.
...	Passed to position_on_surface(). There is no seed argument: the divergent-branch bootstrap is seeded internally, so repeated calls on the same ratings reproduce.

Details

The body, in order:

1. Normalize ratings to a canonical N x k integer matrix Y and derive pi_hat = mean(Y), k = ncol(Y), N = nrow(Y). Validate k >= 2; warn at N < 10; note at N < 30.

2. Compute the panel of observed coefficients (compute_panel(), internal): at k = 2, PABAK / AC1; at k >= 3, PABAK / AC1 / Fleiss kappa / ICC. Cohen's kappa is not on the card at any k; it has no calibrated reference.
3. For each panel coefficient, position the observed value on its DGP-calibrated reference surface via position_on_surface().
4. Pick the primary coefficient by prevalence (metric = "auto") or accept the user's override.
5. Compute the cross-coefficient implied-quality spread delta_hat (in pp of quality) via check_asymmetry() and flag aligned / caution / divergent by delta_hat's percentile on the matched (k, N, q_hat) null.
6. If flag == "divergent": run a latent_class_fit() (Dawid-Skene EM at k >= 3; Hui-Walter bounds at k = 2) and attach the per-rater (Se_j, Sp_j) table.
7. Assemble the grass_card S3 object.

Value

An object of class c("grass_card", "list") with fields sample, coefficient, delta, panel, per_rater, surface, call, grass_version, timestamp, inputs, notes.

coefficient carries the primary coefficient's observed_value, its surface_percentile (the pooled percentile – position within the design's achievable agreement range – on the 0-100 scale), the implied panel quality q_hat, and the 95% test-inversion consistency_band on quality (suppressed at the divergent flag).

panel is a data.frame with one row per coefficient carrying its observed_value, surface_percentile, consistency-band columns (band_lo, band_hi, band_open_low, band_open_high), q_hat, se_q_hat, clamped, reference_used, and in_delta_hat.

delta carries delta_hat (implied-quality spread, in pp of quality), delta_percentile (its percentile on the matched null), flag, matched_null, thresholds, and thresholds_source.

Percentile units. card$coefficient$surface_percentile and card$panel$surface_percentile are reported on the 0-100 scale (e.g., 46.3 means the 46th percentile). The underlying position_on_surface() returns percentile on the 0-1 fraction scale; grass_report() multiplies by 100 to match the paper's prose convention. The print and format methods use ordinal notation ("46th percentile").

Examples

```
# k >= 3 computes ICC via a glmer fit when lme4 (Suggests) is
# available; without lme4 the card degrades gracefully to the
# agreement family.
set.seed(1)
Y <- matrix(rbinom(1000, 1, 0.3), nrow = 200, ncol = 5)
card <- grass_report(ratings = Y)
card                       # print
summary(card)              # full panel + per-rater
as.data.frame(card)        # tidy long-format
```

grass_roadmap *The grass ecosystem roadmap*

Description

grass is the binary categorical-agreement submodule of the MEADOW framework. Its single headline entry point – grass_report() – takes an N x k binary rating matrix and returns a grass_card: a four-field Report Card carrying the sample summary, the primary coefficient with its pooled percentile and 95% consistency band on panel quality, and the cross-coefficient asymmetry diagnostic delta_hat with its matched-null flag. The full panel of coefficients, percentiles, bands, and reference-surface artifacts ride along on the same object for summary(), as.data.frame(), and plot() access.

Framework foundation

A core idea animates the framework: **fixed interpretation bands (Landis-Koch 1977 and its descendants) are mathematically invalid across prevalences, sample sizes, and rater counts.** The Target-2 principle of MEADOW is *context-conditioned reporting*: the percentile a coefficient lands at and the quality band reported with it are both conditioned on the actual study (k, N, pi_hat), computed against a calibrated reference surface rather than read off a fixed table. grass ships the binary submodule; FIELD is planned to ship alongside Paper 3 with the same Target-2 contract for variance-component reliability on continuous outcomes.

MEADOW submodules

MEADOW is the umbrella; each submodule covers one scale type. The user-facing API of each submodule is the same – a single rating matrix in, a Report Card out – so user code that works on a binary panel today will work on a continuous panel once FIELD ships.

Submodule	Scope
GRASS	Binary categorical agreement (Cohen's kappa, PABAK, AC1, Fleiss kappa, observed ICC for binary). Surface p
FIELD	Continuous variance-component reliability (Shrout-Fleiss ICC family, Lin's CCC, Bland-Altman bounds, gener

Earlier drafts of the roadmap referenced TURF and a separate MEADOW submodule for nominal multi-rater agreement. Those are retired: GRASS now covers the binary multi-rater case (Fleiss kappa, Krippendorff alpha, observed ICC) directly, and the framework taxonomy collapses to MEADOW = GRASS + FIELD.

Stable API

The Target-2 contract is the same across submodules:

- grass_report() – primary entry point; rating matrix in, grass_card out. The grass_card carries the four-field summary (sample, primary coefficient with pooled percentile and consistency band, delta_hat with its matched-null flag), the full panel of coefficients, and the reference-surface artifacts.

- `position_on_surface()` – granular access to a single coefficient's pooled percentile, consistency band, and sweep profile.
- `check_asymmetry()` – granular access to the cross-coefficient `delta_hat` and three-tier stability flag.
- `latent_class_fit()` – per-rater Se/Sp via Dawid-Skene EM (k >= 3) or Hui-Walter bounds (k = 2), populated automatically in the divergent branch of `grass_report()`.
- `summary()`, `as.data.frame()`, `plot()` – layered access to the full underlying panel and the surface-position visualization.

Target-2 vocabulary

These terms, used throughout the package documentation and printed Report Card, come from the merged GRASS binary-rater-reliability paper (Sec.Sec.3-4):

- **context-conditioned reporting convention** – the principle that the percentile and band reported for a coefficient must condition on `(k, N, pi_hat)`, not on a fixed table.
- **surface-position percentile** – the empirical percentile of the observed coefficient against the calibrated reference surface at the study's `(k, N, pi_hat)`.
- **consistency band on quality** – the 95% test-inversion band on panel quality `q_hat` (the operating-quality projection onto the `Se = Sp` diagonal): the quality levels whose sampling distributions are consistent with the observed coefficient at this design. The stipulated four-band `Poor` / `Moderate` / `Strong` / `Excellent` partition is retired (0.7.1).
- **delta-hat (`delta_hat`) stability flag** – the cross-coefficient implied-quality spread (in pp of quality) with three flags `aligned` / `caution` / `divergent` set by `delta_hat`'s percentile on the matched (k, N, q_hat) null (>= 95th caution, >= 99th divergent). When `divergent`, the band is suppressed and per-rater Se/Sp from a latent-class fit are reported instead.

What is not yet implemented

Ordinal agreement, nominal multi-category agreement, and the FIELD continuous submodule are outside the current calibration. They are roadmap items, not shipped API: nothing in the package accepts them yet, and there are no placeholder constructors.

Paper

The foundational paper for MEADOW and the GRASS submodule is in review (Semmel 202X, *Context-Conditioned Reporting for Binary Rater Reliability*). The FIELD paper will follow, citing the GRASS paper as the methodological precedent. Each paper accompanies a minor or major release of this package rather than a new package.

`latent_class_fit` *Latent-class fit: per-rater Sensitivity and Specificity from a binary rating matrix*

Description

`latent_class_fit()` is the divergent-branch fallback for the GRASS Reporting Card. When the cross-coefficient panel disagrees, the framework abandons the single q_hat summary and reports per-rater (Se_j, Sp_j) instead. This function fits those per-rater accuracy parameters from an N x k binary rating matrix.

Usage

```
latent_class_fit(
  ratings,
  B = 1000L,
  method = NULL,
  max_iter = 1000L,
  tol = 1e-06,
  seed = NULL,
  ...
)
```

Arguments

`ratings`	An N x k binary rating matrix. Rows are subjects, columns are raters, values in {0, 1}. Data.frame and list-of-rater- columns inputs are coerced. Requires N >= 10, k >= 2, no NA, no all-constant columns.
`B`	Integer >= 0. Number of nonparametric bootstrap replicates. `B = 0` skips the bootstrap and returns the EM point estimate (or bounds at k = 2) without CIs.
`method`	One of `"dawid_skene_em"`, `"hui_walter"`, or `NULL` (default). When `NULL`, dispatches to Hui-Walter at k = 2 and to Dawid-Skene EM at k >= 3. Supplying `"dawid_skene_em"` at k = 2 errors – the EM is unidentified there.
`max_iter`	Integer. EM iteration cap. Default 1000.
`tol`	Numeric. EM log-likelihood tolerance. Default 1e-6.
`seed`	Optional integer. Sets the bootstrap RNG; the EM itself is deterministic.
`...`	Reserved for future extensions; currently ignored.

Details

Two regimes, dispatched by `k`:

- **k >= 3: Dawid-Skene 1979 expectation-maximization.** Latent binary class C_i in {0, 1}; conditional independence of raters given C; parameters (pi, Se_j, Sp_j) for j = 1..k. Initialized from majority-vote consensus. Iterated to log-likelihood tolerance `tol` or `max_iter`. Hand-rolled in base R, no new package dependency.

- **k == 2: Hui-Walter 1980 inequality bounds.** Per-rater Se and Sp are NOT point-identified from a single 2x2 table without external information (e.g., a known-prevalence subgroup or a reference standard). We return the inequality bounds the observed marginals support: with rater j positive rate `P_j`, `Se_j` is bounded below by `P_j`, `Sp_j` is bounded below by `1 - P_j`, both above by 1. The returned rows have `bound_only = TRUE` and `se_hat = sp_hat = NA`. This is a documented limitation of the design, not a bug.

Bootstrap CI (when `B > 0`): nonparametric, subjects-with-replacement. At k >= 3 the EM is refit on each bootstrap sample; at k = 2 the bounds are recomputed and the bootstrap distribution is on the bound midpoints. Reports the empirical 2.5 / 97.5 percentiles on the per-rater Se_j and Sp_j (or their bound midpoints at k = 2).

Value

An S3 object of class `c("grass_latent_class", "list")`:

- `per_rater`: data.frame with `rater`, `se_hat`, `sp_hat` (NA at k = 2), `se_lower`, `se_upper`, `sp_lower`, `sp_upper`, `bound_only` (TRUE at k = 2).
- `method`: `"dawid_skene_em"` or `"hui_walter"`.
- `converged`: logical (NA at k = 2).
- `iterations`: integer (NA at k = 2).
- `B`: integer; bootstrap replicates run.
- `prevalence_hat`: numeric estimated prevalence (NA at k = 2).
- `log_likelihood`: numeric (NA at k = 2).

References

Dawid, A. P. and Skene, A. M. (1979). Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1), 20–28.

Hui, S. L. and Walter, S. D. (1980). Estimating the error rates of diagnostic tests. *Biometrics*, 36(1), 167–171.

Examples

```
set.seed(1)
N <- 500; k <- 5
Se <- 0.90; Sp <- 0.85; pi <- 0.30
C <- rbinom(N, 1, pi)
Y <- matrix(0L, N, k)
for (j in seq_len(k))
  Y[, j] <- rbinom(N, 1, ifelse(C == 1, Se, 1 - Sp))
fit <- latent_class_fit(Y, B = 200, seed = 1)
print(fit)
```

obs_krippendorff_alpha *Krippendorff's alpha (nominal, binary, fully crossed)*

Description

Fully-crossed design (`m_i = k` for all i). Hayes-Krippendorff coincidence- matrix reduction: `o_{01} = 2 * sum_i r_i * (k - r_i)`, `n_0 = N*k - sum(r_i)`, `n_1 = sum(r_i)`, `n_total = N*k`, `alpha = 1 - (n_total - 1) * o_{01} / (2 * n_0 * n_1)`.

Usage

```
obs_krippendorff_alpha(Y)
```

Arguments

`Y`	N x k integer matrix in {0L, 1L}.

Details

Returns `NA_real_` (with a `note` attribute) if the design is degenerate (no positives or no negatives in `Y`).

As of v0.6.0, alpha is not part of the Report Card panel or the `delta_hat` coefficient set: in the binary fully-crossed case it coincides with Fleiss' kappa asymptotically (median absolute difference 0.00024 across the 10,140-cell calibration grid). This function is exported for users who need the value for cross-study comparison.

Value

Single numeric.

pairwise_agreement *Pairwise reliability for a divergent panel*

Description

Computes pairwise PABAK between every pair of raters in an $N \times k$ binary rating matrix and places each entry on the $k = 2$ reference surface at the pair's observed marginal. Also returns per-rater pooled-reference sensitivity and specificity — each rater's call rate against the panel-majority of the other $k - 1$ raters — which uses the larger panel's information about rater behavior rather than discarding it to a strictly pairwise comparison.

Usage

```
pairwise_agreement(ratings, axis = c("inter", "intra"))
```

Arguments

`ratings`	An $N \times k$ binary rating matrix (rows = subjects, columns = raters), or a data.frame whose columns are 0/1 / logical / two-level factor. Same input conventions as `grass_report`.
`axis`	Character; `"inter"` (default) or `"intra"`. The intra-axis path treats the columns of `ratings` as per-rater viewing pairs; see paper Sec.3.3 intra-rater section.

Details

This function is the recommended primary deliverable when `grass_report()` flags the panel as *divergent*; the panel-aggregate coefficients no longer summarize the panel adequately, but the pairwise matrix exposes the panel's structure directly (uniform inconsistency, sub-group clustering, or single- rater outliers).

Value

An object of class `c("grass_pairwise", "list")` with fields:

- `pabak_matrix` — $k \times k$ symmetric numeric matrix; $[i, j]$ is PABAK_{ij}, the diagonal is 1.
- `percentile_matrix` — $k \times k$ symmetric numeric (0–100); $[i, j]$ is the surface percentile of PABAK_{ij} on the $k = 2$ reference at the pair's observed marginal. Diagonal is `NA`.
- `marginal_matrix` — $k \times k$ symmetric numeric; $[i, j]$ is $\hat{\pi}_{+,ij}$. Diagonal is `NA`.
- `band_lo_matrix` — $k \times k$ numeric matrix; lower endpoint of each pair's 95\ (v0.7.1 sweep convention). Diagonal is `NA`.
- `band_hi_matrix` — $k \times k$ numeric matrix; upper endpoint of each pair's 95\ Diagonal is `NA`.
- `pooled_per_rater` — data frame with k rows, one per rater, columns `rater`, `se_tilde`, `sp_tilde`, `n_pool_pos`, `n_pool_neg`, `n_pool_excluded` (subjects with tied panel majority).
- `sample` — list with `k`, `N`, `pi_hat`, `tau2_hat`, `axis`.
- `notes` — character vector with caveats, if any (e.g., undefined pooled-reference at $k = 2$).
- `call` — the matched call.

Examples

```
set.seed(6)
Se <- c(0.95, 0.75, 0.95, 0.75, 0.95)
Sp <- c(0.75, 0.95, 0.75, 0.95, 0.75)
truth <- rbinom(200, 1, 0.5)
Y <- sapply(seq_along(Se),
            function(j) ifelse(truth == 1,
                               rbinom(200, 1, Se[j]),
                               rbinom(200, 1, 1 - Sp[j])))
pw <- pairwise_agreement(Y)
pw
```

plot.grass_card *Plot a* grass_card *Report Card object*

Description

Six views of a Report Card returned by grass_report(). The default "surface" view places the study on its DGP-calibrated reference surface; the other views are for cross-coefficient comparison, the asymmetry diagnostic, observed-coefficient confidence intervals, per-rater Se/Sp from the latent-class fit (only valid when card$delta$flag == "divergent"), and a patchwork composite of the most useful at-a-glance panels.

Usage

```
## S3 method for class 'grass_card'
plot(
  x,
 type = c("surface", "panel", "thermometer", "intervals", "per_rater", "pairwise",
    "diagnostic"),
  ...
)
```

Arguments

x	A grass_card object.
type	One of:

- "surface" (default) – 2D heatmap of the expected primary metric over (M_1, q) at the study's (k, N), with the observation pinned and dotted band contours at q in {0.5, 0.625, 0.75, 0.875, 1.0}.
- "panel" – forest plot of all panel coefficients on the percentile axis (0-100), so the cross-coefficient spread is visible.
- "thermometer" – colored gauge for delta_hat with the aligned/caution/divergent thresholds shown.
- "intervals" – forest plot of observed coefficients with 95% Wilson-logit CIs.
- "per_rater" – forest plot of per-rater Se-hat and Sp-hat (latent-class bootstrap CIs). Errors when card$per_rater is NULL.
- "pairwise" – k x k tile heatmap of pairwise surface percentiles (cell label = percentile; parenthetical label = PABAK_ij). Only available when card$delta$flag == "divergent" (auto-populated by grass_report() via pairwise_agreement()).
- "diagnostic" – patchwork composite of "panel" + "thermometer" + "pairwise" (under divergent) / "per_rater" / "intervals".

...	Currently unused.

Value

A ggplot object (or a patchwork composite for type = "diagnostic").

---

plot.grass_power *Plot a power curve from* grass_power()

Description

Draws power across the solved variable's range, with the requested power as a reference line and the solution marked when one exists. When power itself was solved, the curve runs over N.

Usage

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

Arguments

x	A grass_power object.
...	Ignored.

Value

A ggplot object.

---

plot.grass_result *Plot a grass report*

Description

The default "landing" plot overlays observed kappa, PABAK, and AC1 on the GRASS prevalence-conditioned reference curves. The "regime" plot places the study on the PI^2 vs BI^2 plane.

Usage

```
## S3 method for class 'grass_result'
plot(
  x,
  type = c("landing", "regime"),
  labels = c("auto", "inline", "legend"),
  show_medium = FALSE,
  title = NULL,
  subtitle = NULL,
  ...
)
```

Arguments

x	A grass_result object.
type	Either "landing" (default) or "regime".
labels	Where to identify the three curves: "auto" (default – inline at the right edge unless curves are too close together), "inline", or "legend".
show_medium	If TRUE, also plot the medium-quality reference curves as a faint dotted band. Default FALSE.
title, subtitle	Optional strings to override the default title and subtitle. Pass NULL to omit either.
...	Unused.

Details

Reference curves are the Youden-J-optimal metric values from the GRASS simulation across prevalence, for competent raters (Se, Sp at or above 0.85 for "high" quality; 0.70 for "medium").

Value

A ggplot object.

---

plot_surface *Standalone reference-surface plot for prospective study design*

Description

Renders the same context-conditioned reference surface that plot.grass_card(type = "surface") shows, but driven by scalar inputs rather than a fitted grass_report() result. The intended use case is prospective study design: a practitioner planning a multi-rater study at design (pi_hat, k, N) can ask "what does my reference surface look like for this metric, before I collect data?" – and plot_surface() answers without requiring a rating matrix.

Usage

```
plot_surface(
  metric,
  pi_hat = NULL,
  observed = NULL,
  k = NULL,
  N = NULL,
  axis = c("inter", "intra"),
  bands = NULL,
  ...
)
```

Arguments

metric	Character scalar. One of "pabak", "mean_pabak", "fleiss_kappa", "kappa", "mean_ac1", "ac1", "krippendorff_a". ICC is not supported (see Details).
pi_hat	Optional numeric scalar in (0, 1). The design's marginal positive rate. If supplied alone, draws a vertical reference line on the surface; if supplied together with observed, anchors the pin point.
observed	Optional numeric scalar. An observed (or hypothetical) coefficient value. When supplied with pi_hat, the function inverts to hat q and pins a marker at (pi_hat, hat q).
k	Optional integer. Rater count. Used for the subtitle only; non-ICC closed-form surfaces are k-invariant.
N	Optional integer. Sample size. Used for the subtitle only; non-ICC closed-form surfaces are N-invariant in the large-N limit the surface represents.
axis	One of "inter" (default) or "intra". Used for the subtitle only.
bands	Optional numeric length-5 increasing vector of q values at which to draw dotted reference gridlines. Default NULL draws none. Cosmetic only; nothing is labeled or classified by these lines.
...	Reserved for future extension.

Details

The plot is a heatmap of the closed-form expectation E[metric](M_1, q) over the surface, where M_1 is the marginal positive rate and q in [0.5, 1] is the diagonal rater operating quality. When pi_hat is supplied alone, a dashed vertical line marks the design's marginal. When both pi_hat and observed are supplied, a filled marker pins the implied (M_1, hat q) point on the surface; hat q is recovered by closed-form inversion of observed against the reference curve at pi_hat.

This function uses the closed-form non-ICC surfaces only (PABAK, Fleiss kappa, mean-pairwise AC1, Krippendorff alpha). ICC surfaces depend on the full subject-prevalence distribution F and the GLMM-gap-corrected fitted reference; those are accessed via position_on_surface() with metric = "icc" and (if available) a fitted card via plot.grass_card(type = "surface").

Value

A ggplot object.

See Also

plot.grass_card() for the card-driven surface view that adds card-specific annotations (consistency band, primary-coefficient pin); position_on_surface() for the underlying inversion machinery.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  # Bare surface -- what does PABAK's reference look like?
  plot_surface("pabak")

  # Mark a design context (no observed value yet -- pre-data).
```

```
  plot_surface("fleiss_kappa", pi_hat = 0.30, k = 5, N = 200)

  # Pin a hypothetical observation on the surface.
  plot_surface("mean_ac1", pi_hat = 0.30, observed = 0.62,
               k = 5, N = 200)
}
```

`position_on_surface` *Position an observed agreement coefficient on its DGP-calibrated surface*

Description

`position_on_surface()` is the Target-2 reporting primitive for the merged GRASS binary-rater-reliability paper. Given an observed coefficient value and the study design (`pi_hat, k, N`), it inverts the coefficient to an implied panel quality `q_hat` (rater operating quality on the Se = Sp diagonal under the clustered latent-class DGP) and evaluates the observed value against EVERY calibrated quality level at the matched design – a sweep – from which it derives three read-outs: the pooled percentile, the consistency band on quality, and the p(q) sweep profile.

Usage

```
position_on_surface(
  obs_value = NULL,
  metric,
  pi_hat = NULL,
  k = NULL,
  N = NULL,
  method = c("empirical", "delta"),
  surface_data = NULL,
  ratings = NULL,
  reference_type = c("fitted", "oracle"),
  ...
)
```

Arguments

`obs_value`	Numeric scalar. The observed agreement coefficient. Optional when `ratings` is supplied (auto-derived via `compute_observed(metric, Y)`).
`metric`	Character scalar. One of `"pabak"`, `"fleiss_kappa"`, `"mean_ac1"`, `"krippendorff_a"`, `"icc"`.
`pi_hat`	Numeric scalar in `(0, 1)`. The panel-identified marginal positive rate. Optional when `ratings` is supplied (auto-derived via `mean(Y)`); otherwise supply `mean(Y)` from the rating matrix.
`k`	Integer >= 2. Number of raters. Optional when `ratings` is supplied (auto-derived as `ncol(Y)`).

`N` Integer >= 1. Number of subjects. Optional when `ratings` is supplied (auto-derived as `nrow(Y)`).

`method` One of `"empirical"` (default; uses the bundled sim-derived empirical q_hat sampling distribution) or `"delta"` (closed-form normal approximation from delta-method SE).

`surface_data` Optional. A list with one or more of the following components, used when `method = "empirical"`:

- `per_rep`: a vector of per-rep metric values at the caller's own `(q, pi_hat, k, N)` cell – an empirical sampling distribution at that design. Honored for reproducibility audits; yields a plain cohort percentile with no sweep or consistency band.
- `q_grid_per_rep`, `q_grid`: legacy per-q-grid empirical inputs. Retained for backward compatibility but no longer consumed internally (the sweep convention consults the bundled per-cell quantile surface); supplying them draws a note. When `surface_data` is `NULL` and `method = "empirical"`, the function uses the bundled empirical q_hat surface, falling back to the delta-method sweep when that surface is unavailable at the design.

`ratings` Optional. From v0.2.0 this is the **primary input for all metrics** (not just ICC): supplying an `N x k` rating matrix auto-derives `obs_value`, `pi_hat`, `k`, and `N`. Accepts an `N x k` integer matrix of 0/1 values (rows = subjects, cols = raters), a data.frame with `k` rater columns, or a length-2 list of equal-length 0/1 vectors (`k = 2`). For `metric = "icc"`, supplying `ratings` (also accepted as a long data.frame with columns `subject` and `rating`) additionally enables a `glmer` fit for `(mu, tau2)` that pins down the correct `F_key` for ICC inversion; without `ratings`, ICC falls back to a nearest-M1 `F_key` lookup with a prominent caveat note (tau2 is unidentified from `pi_hat` alone). The `glmer` path requires `lme4` (Suggests).

`reference_type` For `metric = "icc"` only. One of `"fitted"` (default; GLMM-gap-corrected reference matching what practitioners compute via `glmer`) or `"oracle"` (closed-form `sigma^2_subject / (sigma^2_subject + pi^2/3)` with `sigma^2_subject` known from F). Use `"oracle"` only if `obs_value` was computed via oracle variance decomposition (non-standard for applied work). For N beyond the fitted-reference sim range (currently N > 200), the function auto-falls-back to oracle with an explanatory note.

`...` Reserved for future extension.

Details

The function implements the v0.7.1 sweep convention (ratified 2026-07-05): the practitioner cites the observed coefficient, its pooled percentile (position within the design's achievable range), and the consistency band on panel quality. `q_hat` is promoted to the card via the consistency band; it also carries the surface parameterization and delta-method SE. The stipulated four-band adjective (Poor/Moderate/Strong/Excellent) and the modal-band confidence qualifier (decisive/moderate/weak) are retired.

Value

A list of class `grass_surface_position` with fields:

- observed_value – echo of obs_value
- metric – echo of metric
- design – list(pi_hat, k, N)
- q_hat – implied panel quality (coefficient inverted on the reference curve); the point estimate the consistency band surrounds
- se_q_hat – delta-method SE of q_hat
- percentile – pooled percentile in [0, 1] (the print method and the card show it on the 0-100 scale): the observed coefficient's position within the design's full achievable range (trapezoid-weighted mixture over every calibrated quality level). Monotone in obs_value by construction.
- percentile_basis – provenance string for percentile
- band – 95% test-inversion consistency band on quality (printed as "consistency band"): list(lo, hi, level, open_low, open_high, note). The quality levels whose sampling distributions are consistent with the observed value at this design.
- sweep – data.frame(q, p): the full profile p(q) = P(coefficient <= obs_value | quality q, this design)
- sampling_method – which method was used
- reference_used – which reference produced the curve
- notes – character vector of caveats (e.g. nearest-neighbor gaps)

The three read-outs of the sweep

At the matched design (F/pi_hat, k, N) the observed coefficient is evaluated against every calibrated quality level q with no cell selection and no q snapping. Write p(q) = P(coefficient <= obs_value | panel of qua

- percentile – the pooled percentile: a trapezoid-weighted average of p(q) over the calibrated quality axis (trapezoid because the grid is non-uniform). It reads as the observed coefficient's position within the design's full achievable agreement range and is monotone in obs_value by construction. Returned in [0, 1]; callers such as grass_report() print it on the [0, 100] scale. This replaces the retired nearest-q_hat-cell percentile, whose cohort selection by a statistic derived from the coefficient made it a non-monotone sawtooth (panel review 2026-07-05).
- band – the 95% test-inversion consistency band on quality: the quality levels q with 0.025 <= p(q) <= 0.975, endpoints interpolated where p(q) crosses 0.975 (lower) and 0.025 (upper). Open-ended at the grid boundary is reported with a boundary flag.
- sweep – the full data.frame(q, p) profile, the object the sweep-ridgeline graphic renders.

Sweep construction

Two methods are implemented.

Empirical method (method = "empirical", default): at the matched (F_key, k, N) cell of the bundled empirical_q_hat_surface, ranks q_hat within each calibrated quality cell's q_hat_rep distribution (monotone-equivalent to ranking obs_value within the cell's coefficient distribution). The full per-rep data is not bundled (~300 MB); the package ships a precomputed multi-point empirical-quantile summary per cell. The whole quality axis is consulted – there is deliberately no

q selection. When the design (pi_hat, k, N) falls outside the simulated grid, nearest-neighbour clamping is applied and flagged in notes.

Delta method (method = "delta"): the summary-stats-only fallback (also used for ICC, whose reference curve carries the F-shape conditioning). At each swept q it approximates the sampling distribution as Normal(E[metric](q), sd_metric(q)) and evaluates p(q) = pnorm(obs_value; mean, sd) on the calibrated q axis. A caller-supplied surface_data$per_rep single-cohort vector is still honored for reproducibility audits, yielding a plain cohort percentile with no sweep or band.

Internal reference-surface arithmetic

Under the clustered latent-class DGP with symmetric raters (Se = Sp = q), the large-N closed forms for PABAK, Fleiss kappa, AC1, and Krippendorff's alpha depend on q and the marginal positive rate pi_+ only. This function uses pi_hat as a plug-in for pi_+ and inverts the observed value on a 501-point q-grid on [0.5, 1] (matching paper2/code/12_q_inversion.R resolution). ICC requires the full subject-prevalence distribution F; in the absence of surface_data containing an ICC lookup, ICC requests fall through to a warning-noted delta-method approximation using the caller-supplied q_hat_override / se_q_hat_override if present, or stop with a clear message.

Ratings-primary path

From v0.2.0, the preferred entry point is to hand the rating matrix directly: position_on_surface(ratings = Y, metric = "pabak"). When ratings is supplied, the function auto-derives obs_value (via compute_observed(metric, Y)), pi_hat (mean(Y)), k (ncol(Y)), and N (nrow(Y)); any of those four arguments still supplied by the caller wins. This collapses the audit-style scalar-input path used in v0.1.x to a single matrix argument while keeping the scalar path callable for reproducibility checks. ratings accepts an N x k integer matrix in {0, 1}, a data.frame with k rater columns, or a length-2 list of equal-length 0/1 vectors (k = 2). Round-trip equality with the scalar path is a tested invariant.

See Also

check_asymmetry() for the companion Column A tier (rater asymmetry model-safety).

Examples

```
# Ratings-primary path: just hand it the matrix.
set.seed(1)
Y <- matrix(rbinom(1000, 1, 0.3), nrow = 200, ncol = 5)
position_on_surface(ratings = Y, metric = "pabak")

# Equivalent scalar-input path (audit):
position_on_surface(
  obs_value = 2 * mean(Y[, 1] == Y[, 2]) - 1,  # PABAK on first pair
  metric = "pabak", pi_hat = mean(Y), k = ncol(Y), N = nrow(Y)
)

# Scalar path -- the three read-outs of the sweep convention.
r <- position_on_surface(
  obs_value = 0.62,
  metric    = "pabak",
```

```
  pi_hat    = 0.42,
  k         = 5,
  N         = 50
)
r$percentile   # pooled percentile of the achievable range
r$band         # consistency band on panel quality
head(r$sweep)  # the full p(q) profile

# Fleiss kappa at imbalanced prevalence.
position_on_surface(
  obs_value = 0.18,
  metric    = "fleiss_kappa",
  pi_hat    = 0.08,
  k         = 3,
  N         = 200
)
```

`print.grass_card` *Print a GRASS Report Card*

Description

Prints the card assembled by `format.grass_card()`: the sample line, one line per coefficient with its percentile and consistency band, the plain-language read, the `delta_hat` spread and flag, and, on a divergent card, the pairwise and per-rater tables.

Usage

```
## S3 method for class 'grass_card'
print(x, digits = 2, ...)
```

Arguments

`x`	A `grass_card` object from `grass_report()`.
`digits`	Decimals for coefficient values. Default 2.
`...`	Ignored.

Value

`x`, invisibly.

reset_grass_warnings *Reset grass's once-per-session message cache*

Description

grass emits coercion messages and keyword-fallthrough warnings at most once per R session, keyed by the column name. When looping grass_report() over many studies with consistently named columns, only the first study in the loop sees the message. Call reset_grass_warnings() to clear the cache so the next call re-evaluates.

Usage

```
reset_grass_warnings()
```

Value

invisible(NULL).

Examples

```
reset_grass_warnings()
```

scale_grass_metric *Color scale for grass metrics*

Description

Maps kappa, PABAK, and AC1 to the paper palette.

Usage

```
scale_color_grass_metric(...)

scale_fill_grass_metric(...)
```

Arguments

...	Passed through to ggplot2::scale_color_manual / ggplot2::scale_fill_manual.

Value

A ggplot2 scale.

summary.grass_card *Summarize a GRASS Report Card*

Description

The full detail behind the card: the primary coefficient with its estimate `q_hat`, the `delta_hat` diagnostic and its matched-null percentile, the whole panel table, the per-rater latent-class table on a divergent card, and every note the card carries, including the provenance notes the card print leaves out.

Usage

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

## S3 method for class 'summary.grass_card'
print(x, digits = 3, ...)
```

Arguments

`object`	A `grass_card` object from `grass_report()`.
`...`	Ignored.
`x`	A `summary.grass_card` object.
`digits`	Decimals for printed values. Default 3.

Value

For `summary()`, a `summary.grass_card` list; its print method returns it invisibly.

theme_grass *ggplot2 theme for grass plots*

Description

Minimal publication-oriented theme. No chrome beyond what the data require.

Usage

```
theme_grass(base_size = 12)
```

Arguments

`base_size`	Base font size passed to `ggplot2::theme_minimal()`.

Value

A ggplot2 theme object.

Index

as.data.frame.grass_card, 3
as.data.frame.grass_surface_position, 3

check_asymmetry, 4
check_asymmetry(), *14*, *16*, *28*

format.grass_card, 6
format.grass_card(), *29*

grass-roadmap *(*grass_roadmap*)*, 15
grass_compute, 7
grass_plot, 8
grass_power, 9
grass_reference, 11
grass_reference_table, 12
grass_report, 12, *20*
grass_report(), *7*, *8*, *11*, *15*, *16*, *21*, *23*, *29*, *31*
grass_roadmap, 15

latent_class_fit, 17
latent_class_fit(), *6*, *14*, *16*

obs_krippendorff_alpha, 19
obs_krippendorff_alpha(), *13*

pairwise_agreement, 19
pairwise_agreement(), *21*
plot.grass_card, 21
plot.grass_card(), *24*
plot.grass_power, 22
plot.grass_result, 22
plot_surface, 23
position_on_surface, 25
position_on_surface(), *4–6*, *10*, *11*, *13*, *14*, *16*, *24*
print.grass_card, 29
print.summary.grass_card *(*summary.grass_card*)*, 31

reset_grass_warnings, 30

scale_color_grass_metric *(*scale_grass_metric*)*, 30
scale_fill_grass_metric *(*scale_grass_metric*)*, 30
scale_grass_metric, 30
stats::power.t.test(), *10*
summary.grass_card, 31

theme_grass, 31