Package: crossStatR (via r-universe)

November 16, 2026

Type Package

Title Scientific Statistics for Biological, Electrochemical, Optical and Physics Experiments

Version 0.99.2

Date 2026-11-16

Author Rohan R [aut, cre, cph]

Maintainer Rohan R <phonics-tiffs1i@icloud.com>

Description A unified scientific statistics framework for biological experiments, electrochemical measurements, optical and spectroscopic measurements, and experimental physics data. Combines established methods (descriptive statistics, hypothesis testing, regression, ANOVA, power analysis, mixed-effects variance decomposition, calibration, peak fitting) with diagnostic methods for experimental analysis (region-level signal difference testing, peak-evidence assessment, calibration-transfer risk, kinetic discrimination, experimental fragility, drift uncertainty attribution, analytical detection reliability, and stage-wise uncertainty-origin attribution). Every advanced method is documented with its mathematical formulation, assumptions, comparison with established alternatives, and known limitations. The package is intentionally conservative: it does not silently remove observations, does not select hypothesis tests solely on the basis of a normality-test p-value, and distinguishes technical from biological replicates in its variance models.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.6.0)

Imports stats, utils, MASS, Matrix, lme4 (>= 1.1-28), ggplot2 (>= 3.4.0)

Suggests testthat (>= 3.0.0), roxygen2, knitr, rmarkdown, pkgdown, covr, nortest

URL https://github.com/rotsl/crossStatR

BugReports https://github.com/rotsl/crossStatR

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

Config/pak/sysreqs cmake make

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

Date/Publication 2026-11-16 13:27:45 UTC

RemoteUrl https://github.com/rotsl/crossStatR

RemoteRef HEAD

RemoteSha dbe682764e2dc32fd3dd53daf1fe71abc74d0cb1

Contents

adaptive_baseline . . . 3
adaptive_smoothing . . . 5
batch_effect_quantify . . . 8
calibration_compare . . . 11
calibration_transfer_risk . . . 13
curve_similarity . . . 15
detection_reliability . . . 19
drift_correct_stat . . . 21
drift_uncertainty . . . 24
experiment_fragility . . . 26
kinetic_discrimination . . . 28
kinetic_model_compare . . . 30
noise_characterize . . . 33
parameter_uncertainty . . . 35
peak_evidence . . . 37
replicate_decompose . . . 40
robust_lod . . . 42
sci_anova . . . 45
sci_bootstrap . . . 46
sci_compare . . . 49
sci_correlation . . . 51
sci_describe . . . 54
sci_effect_size . . . 56
sci_normality . . . 59
sci_outliers . . . 61
sci_power . . . 63
sci_regression . . . 66
signal_difference_regions . . . 68
spectral_change_test . . . 70

summary.crossstatr_result 73
tidy.crossstatr_result 74
timepoint_optimize 74
uncertainty_origin 77
weighted_peak_fit 80

Index **83**

`adaptive_baseline` *Adaptive baseline estimation*

Description

Estimates a slowly varying baseline under a signal that contains sharp peaks. Two methods are provided. Asymmetric least squares (Eilers & Boelens, 2005) iteratively reweights a penalised least squares fit so the baseline tracks the lower envelope of the signal; the lowess baseline smooths the signal and then clips the result to lie below the data.

Usage

```
adaptive_baseline(
  x,
  y,
  method = c("asls", "lowess"),
  lambda = 1e+07,
  p = 0.01,
  span = 0.5,
  max_iter = 20L,
  tol = 1e-06,
  conf_level = 0.95
)
```

Arguments

`x`	numeric vector of abscissae (wavelength, time, m/z, etc.).
`y`	numeric vector of signal intensities, same length as `x`.
`method`	character; one of `"asls"` (asymmetric least squares, default) or `"lowess"`.
`lambda`	numeric; smoothness penalty for `"asls"`. Larger values give smoother baselines. Typical values range from 1e5 to 1e9.
`p`	numeric in (0, 1); asymmetry parameter for `"asls"`. Smaller values push the baseline below the signal. The default 0.01 produces baselines that follow the lower envelope.
`span`	numeric in (0, 1]; span for `"lowess"`.
`max_iter`	integer; maximum number of reweighting iterations.
`tol`	numeric; convergence tolerance on the change in the baseline between iterations.
`conf_level`	currently unused; reserved for future uncertainty quantification on the baseline. Validated for being a probability.

Details

The documentation recommends asymmetric least squares for chromatography, Raman, and XPS data where the peak structure is sharp relative to the baseline. The lowess option is intended as a sanity check; it does not try to be asymmetric and will be biased upward when the signal sits well above the true baseline.

Value

A `crossstatr_baseline` object with elements:

`x`	input abscissae.
`y`	input signal.
`baseline`	fitted baseline.
`corrected`	`y - baseline`.
`weights`	final asymmetric weights (uniform for `"lowess"`).
`converged`	logical; whether reweighting converged within `max_iter`. For `"lowess"` this is always `TRUE`.
`n_iter`	number of iterations performed (1 for `"lowess"`).
`method`	baseline method used.
`call`	the matched call.

Methodology

Asymmetric least squares (Eilers & Boelens, 2005) minimises the penalised least squares objective

$$Q(\mathbf{z}) = (\mathbf{y} - \mathbf{z})^\top W (\mathbf{y} - \mathbf{z}) + \lambda \, \mathbf{z}^\top D_2^\top D_2 \, \mathbf{z},$$

where D_2 is the $(n-2)\times n$ second-order difference operator (so $D_2\mathbf{z}$ measures the discrete curvature of $\mathbf{z}$) and $W = \mathrm{diag}(w_i)$ holds the asymmetric weights $w_i = p$ when $y_i > z_i$ and $w_i = 1 - p$ otherwise. The closed-form update at each iteration is

$$\mathbf{z} = (W + \lambda D_2^\top D_2)^{-1} W \, \mathbf{y},$$

solved via `Matrix::solve`. The package iterates until $\max_i |z_i^{\mathrm{new}} - z_i^{\mathrm{old}}| < \mathrm{tol}$ or `max_iter` is reached.

The lowess baseline is `stats::lowess(x, y, f = span)`, aligned back to the input order, then clipped to the lower envelope by taking $\tilde{z}_i = \min(\hat{z}_i, y_i)$. This is not truly asymmetric but enforces the baseline-lies-below-signal property.

Limitations

- ALS with large `lambda` produces highly correlated baselines; small changes in the signal can shift the whole baseline. Inspect `corrected` for systematic residual structure before downstream use.
- All methods assume peaks are positive (the signal sits above the baseline). For negative peaks, multiply the signal by -1 before fitting or set `p > 0.5`.

- Convergence of the asymmetric reweighting is not guaranteed for pathological signals. Always check `converged` and `n_iter`.
- The lowess baseline is a poor default for signals with wide peaks; it will be biased upward and clip the corrected signal at zero.

References

- Eilers, P. H. C. and Boelens, H. F. M. (2005). Baseline Correction with Asymmetric Least Squares Smoothing. Unpublished manuscript. https://www.researchgate.net/publication/228961729_Baseline_Correction_with_Asymmetric_Least_Squares_Smoothing.
- Eilers, P. H. C. (2003). A Perfect Smoother. Analytical Chemistry, 75, 3631-3636. doi:10.1021/ac034173t.
- Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. Journal of the American Statistical Association, 74, 829-836. doi:10.1080/01621459.1979.10481038.

Examples

```
set.seed(1)
x <- seq(0, 10, length.out = 200)
peak1 <- 5 * exp(-(x - 3)^2 / 0.1)
peak2 <- 3 * exp(-(x - 7)^2 / 0.05)
baseline_true <- 0.5 + 0.2 * x + 0.03 * x^2
y <- baseline_true + peak1 + peak2 + rnorm(length(x), sd = 0.1)
fit <- adaptive_baseline(x, y, method = "asls", lambda = 1e6, p = 0.01)
print(fit)

fit_l <- adaptive_baseline(x, y, method = "lowess", span = 0.3)
print(fit_l)
```

---

adaptive_smoothing *Adaptive smoothing for noisy signals*

Description

Smooths a noisy signal with a choice of three backends: Savitzky-Golay polynomial smoothing, smoothing spline, and lowess. For Savitzky-Golay, the window size is chosen automatically by minimising leave-one-out PRESS over a candidate grid, subject to a peak-preservation constraint that vetoes windows which reduce the global maximum by more than 5%.

Usage

```
adaptive_smoothing(
  x,
  y,
  method = c("sgolay", "spline", "lowess"),
  window = NULL,
```

```
  poly_order = 3L,
  span = NULL,
  span = 0.5,
  preserve_peaks = TRUE
)
```

Arguments

x	numeric vector of abscissae.
y	numeric vector of signal intensities, same length as x.
method	character; one of "sgolay" (default), "spline", or "lowess".
window	integer; window size for "sgolay". Must be odd. If NULL, chosen adaptively from the candidate set c(5, 7, 9, 11, 15, 21).
poly_order	integer; polynomial order for "sgolay". Must be less than window.
spar	numeric; smoothing parameter for "spline", passed to stats::smooth.spline. If NULL, the spline chooses its own.
span	numeric in (0, 1]; span for "lowess".
preserve_peaks	logical; if TRUE (default), the adaptive search for sgolay vetoes windows that reduce the global maximum of y by more than 5%.

Details

Smoothing is a bias-variance tradeoff. Aggressive smoothing reduces variance (noise) at the cost of bias: peak heights shrink, peak positions can shift, and narrow peaks may disappear entirely. The package documents this carefully because the bias propagates into downstream inference. A smoothed peak height is not a peak height, it is a biased estimate of one; the same applies to peak widths, peak areas, and any test statistic computed on smoothed data. Whenever possible, fit a model directly to the raw noisy data and let the likelihood handle the noise.

Value

A crossstatr_smoothing object with elements:

x, y	inputs.
smoothed	smoothed signal.
method	smoothing method used.
window	window size, if method = "sgolay"; NA_integer_ otherwise.
spar	spline spar, if method = "spline"; NA_real_ otherwise.
span	lowess span, if method = "lowess"; NA_real_ otherwise.
press	cross-validated PRESS of the chosen smoothing parameter.
peak_height_loss	fraction of the global maximum lost to smoothing.
call	the matched call.

Methodology

Savitzky-Golay smoothing (Savitzky & Golay, 1964; Steinier, Termonia & Deltour, 1972) fits a polynomial of order m in a sliding window of size $2h + 1$ and replaces the central point with the value of the polynomial at the centre. Equivalently, the smoothed value at point i is

$$\hat{y}_i = \sum_{j=-h}^{h} c_j \, y_{i+j},$$

where the coefficient vector c is the first row of $(A^\top A)^{-1} A^\top$ and A is the Vandermonde design matrix of the polynomial in the window. The package implements this directly by solving for the polynomial at each window and reading off the central fitted value, with reflection padding at the boundaries.

Smoothing spline: `stats::smooth.spline(x, y, spar = spar)` minimises the penalised residual sum of squares

$$\sum_i (y_i - f(x_i))^2 + \lambda \int [f''(t)]^2 \, dt,$$

with `spar` controlling the tradeoff.

Lowess (Cleveland, 1979) performs locally weighted polynomial regression with tricube weights, controlled by `span`.

The cross-validated PRESS for a smoothing parameter θ is

$$\mathrm{PRESS}(\theta) = \sum_i (y_i - \hat{y}_\theta^{(-i)}(x_i))^2,$$

where $\hat{y}_\theta^{(-i)}$ is the smoothed signal computed without point i. For linear smoothers this can be computed exactly via the hat matrix S_θ:

$$\hat{y}_\theta^{(-i)}(x_i) = \frac{\hat{y}_i - S_{ii} y_i}{1 - S_{ii}},$$

where S_{ii} is estimated by a finite-difference probe of the smoother (perturb each y_i by a small ϵ and read off the resulting change in $\hat{y}_i$). The peak-preservation constraint vetoes any candidate whose smoothed global maximum is less than 95% of the raw global maximum.

Limitations

- Smoothing biases downstream inference. A smoothed peak height is a downward-biased estimate of the true peak height; the bias is largest for narrow peaks and aggressive smoothing. The same applies to peak widths (upward-biased) and peak areas (downward-biased for narrow peaks). Fit models on the raw data when the analyst can.
- The adaptive PRESS criterion tends to pick small windows because it rewards predictive accuracy, which is dominated by the noise level. The peak-preservation constraint partially mitigates this, but the analyst should confirm the choice is reasonable for the application.
- Savitzky-Golay with `poly_order` close to `window - 1` will overfit. The documentation recommends `poly_order <= 4` and `window >= 5`.
- For non-uniformly sampled `x`, `sgolay` uses the index position (not the `x` value) as the abscissa, which introduces a small bias if `x` is very unevenly spaced. Use `method = "spline"` in that case.

References

- Savitzky, A. and Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. Analytical Chemistry, 36, 1627-1639. doi:10.1021/ac60214a047.
- Steinier, J., Termonia, Y. and Deltour, J. (1972). Smoothing and differentiation of data by simplified least square procedure. Analytical Chemistry, 44, 1906-1909. doi:10.1021/ac60319a045.
- Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. Journal of the American Statistical Association, 74, 829-836. doi:10.1080/01621459.1979.10481038.

Examples

```
set.seed(1)
x <- seq(0, 10, length.out = 200)
y <- 5 * exp(-(x - 5)^2 / 0.1) + rnorm(length(x), sd = 0.5)
fit <- adaptive_smoothing(x, y, method = "sgolay", poly_order = 3L)
print(fit)

fit2 <- adaptive_smoothing(x, y, method = "spline")
print(fit2)

fit3 <- adaptive_smoothing(x, y, method = "lowess", span = 0.2)
print(fit3)
```

---

batch_effect_quantify *Quantify batch effects*

Description

Decomposes the variance in an experimental outcome into treatment, batch, and residual components, and reports the proportion of variance attributable to batch. Two model parameterisations are offered: a linear mixed model with a random intercept on batch (`type = "lmm"`, recommended when batches are a random sample from a larger population of batches), and a fixed-effect model with batch as a categorical covariate (`type = "fixed"`, recommended when the observed batches are the only ones of interest).

Usage

```
batch_effect_quantify(
  formula,
  data,
  batch,
  type = c("lmm", "fixed"),
  conf_level = 0.95
)
```

Arguments

formula	a formula of the form outcome ~ treatment. Only one predictor is allowed.
data	data frame containing the variables in formula and the batch column.
batch	either a character string naming a column in data, or a vector (factor or coercible) of batch identifiers of length nrow(data).
type	"lmm" (default) for a random-intercept LMM via lme4::lmer(), or "fixed" for a fixed-effect linear model via stats::lm().
conf_level	confidence level; stored on the result for downstream reporting.

Details

Confounding between treatment and batch is checked by a chi-square test of association. If treatment is perfectly confounded with batch (each treatment level occurring in only one batch, and vice versa), the function stops with an error because the treatment effect is not estimable.

Value

A crossstatr_batch object with elements:

batch_variance	estimated batch (random-intercept) variance. For type = "fixed" this is the one-way random-effects variance estimate derived from the batch sum of squares.
batch_pct_variance	batch variance as a percentage of total (batch + residual) variance.
icc	intraclass correlation coefficient $\sigma^2_{\text{batch}}/(\sigma^2_{\text{batch}} + \sigma^2_{\text{resid}})$.
confound_p	chi-square p-value for the treatment-by-batch association test. A small p indicates partial confounding.
confound_test	list with statistic, p_value, df from a chi-square test of association between treatment and batch.
model	model label.
treatment_effect_change	change in the treatment coefficient when batch is added to the model.
fit, base_fit	the fitted full and treatment-only models.
call	the matched call.

Methodology

For type = "lmm" The package fits the random-intercept model

$$y_{ij} = \mu + \alpha\, x_{ij} + b_j + \epsilon_{ij}, \qquad b_j \sim N(0, \sigma_b^2), \quad \epsilon_{ij} \sim N(0, \sigma^2),$$

where j indexes batch. The intraclass correlation coefficient is

$$\text{ICC} = \frac{\sigma_b^2}{\sigma_b^2 + \sigma^2}.$$

For type = "fixed" The package fits the linear model

$$y_{ij} = \mu + \alpha\, x_{ij} + \beta_j + \epsilon_{ij},$$

compare it to the treatment-only model via `stats::anova()`, and derive an approximate batch variance from the batch sum of squares using the one-way random-effects decomposition (Searle, Casella and McCulloch, 1992, sec. 3.7):

$$\hat{\sigma}_b^2 \approx \frac{\mathrm{SS}_{\mathrm{batch}}/\mathrm{df}_{\mathrm{batch}} - \hat{\sigma}^2}{n_0},$$

where n_0 is the effective number of observations per batch $(N - \sum_j n_j^2/N)/(J-1)$.

Treatment-effect change: $\Delta\alpha = \alpha_{\mathrm{full}} - \alpha_{\mathrm{base}}$. A large absolute value signals that batch was confounded with treatment.

Confounding check: a chi-square test of independence between treatment and batch. Perfect confounding raises an error.

Limitations

- The LMM assumes batches are a random sample from a larger population of batches. If only a small number of batches is observed, the batch-variance estimate is unstable; consider the fixed-effect model in that case.
- The fixed-effect model does not generalise to unobserved batches. The reported batch variance is a method-of-moments estimate and should be interpreted as a description of the observed batches only.
- ICC for the fixed-effect path is approximate and can be negative when between-batch variation is small; the package truncates at zero.
- The function does not check whether the treatment effect itself is well-estimated. A small `treatment_effect_change` is consistent with no confounding, but it does not rule out subtle confounding that the chi-square test misses.

References

- Bates, D., Mächler, M., Bolker, B. and Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. Journal of Statistical Software, 67, 1-48. doi:10.18637/jss.v067.i01.
- Searle, S. R., Casella, G. and McCulloch, C. E. (1992). Variance Components. Wiley. doi:10.1002/9780470316856.
- Pinheiro, J. C. and Bates, D. M. (2000). Mixed-Effects Models in S and S-PLUS. Springer New York. doi:10.1007/9781441903181.

Examples

```
set.seed(1)
# 3 treatments x 4 batches x 5 obs per (treatment, batch) cell,
# with batch-level shifts of 0, 1, -1, 0.5 respectively
df <- data.frame(
  y = rnorm(60, mean = rep(c(0, 1, 2), each = 20) +
                       rep(rep(c(0, 1, -1, 0.5), each = 5), times = 3)),
  treat = factor(rep(c("A","B","C"), each = 20)),
  batch = factor(rep(1:4, each = 5, times = 3))
)
r_lmm <- batch_effect_quantify(y ~ treat, data = df, batch = "batch",
```

```
                                  type = "lmm")
print(r_lmm)

r_fixed <- batch_effect_quantify(y ~ treat, data = df, batch = "batch",
                                   type = "fixed")
print(r_fixed)
```

`calibration_compare` *Compare two calibration curves via ANCOVA*

Description

Compares two linear calibration models (typically linear regressions of response on concentration) to test whether they share the same slope and the same intercept. The comparison uses a single ANCOVA fit on the stacked data with an interaction between concentration and a model indicator, then tests the interaction (slopes equal?) and the main effect of the indicator (intercepts equal given equal slopes).

Usage

```
calibration_compare(model1, model2, conf_level = 0.95, method = c("ancova"))
```

Arguments

`model1`, `model2`	`lm` objects (from `stats::lm(y ~ x)`) or data frames with columns `x` and `y`.
`conf_level`	confidence level for the slope-difference and intercept-difference CIs.
`method`	character; currently only `"ancova"` is implemented.

Details

Both models can be supplied either as `lm` objects (from `stats::lm(y ~ x)`) or as data frames with columns `x` and `y`. If a data frame is supplied, the linear model is fit internally by OLS.

Value

A `crossstatr_calibration_compare` object with elements:

`summary`	data frame with one row per comparison (slope difference, intercept difference), each with estimate, SE, CI, and p-value.
`ancova_fit`	the combined `lm` object from the ANCOVA fit.
`slope_equal`	logical; TRUE if slopes are not significantly different at the chosen `conf_level`.
`slope_equal_p`	p-value for the slope-equality test.
`intercept_equal`	logical; TRUE if intercepts are not significantly different.
`intercept_equal_p`	p-value for the intercept-equality test.

`slope1`, `slope2`, `intercept1`, `intercept2`
: the fitted slopes and intercepts of each calibration.

`residual_se1`, `residual_se2`
: residual SE of each model.

`n1`, `n2`
: number of observations in each calibration.

`f_statistic`, `f_p_value`
: F statistic and p-value for the overall model-comparison test.

`method`
: backend used.

`call`
: the matched call.

Methodology

Let the two calibrations be

$$y_{1i} = \alpha_1 + \beta_1 x_{1i} + \epsilon_{1i}, \qquad y_{2j} = \alpha_2 + \beta_2 x_{2j} + \epsilon_{2j},$$

with $\epsilon \sim N(0, \sigma^2)$ assumed common. Define an indicator $g \in \{0, 1\}$ for the second calibration. The combined ANCOVA model is

$$y = \alpha + \beta\, x + \delta\, g + \gamma\, (x \cdot g) + \epsilon,$$

where $\delta = \alpha_2 - \alpha_1$ and $\gamma = \beta_2 - \beta_1$. The slope-difference test is $H_0 : \gamma = 0$, and the intercept-difference test (given equal slopes) is $H_0 : \delta = 0$ in the reduced model without interaction.

The CIs on the slope and intercept differences are read directly off the ANCOVA fit coefficients. The overall F statistic tests the joint null $H_0 : \delta = \gamma = 0$ via

$$F = \frac{(\mathrm{RSS}_{\mathrm{reduced}} - \mathrm{RSS}_{\mathrm{full}})/2}{\mathrm{RSS}_{\mathrm{full}}/(n_1 + n_2 - 4)},$$

with 2 and $n_1 + n_2 - 4$ degrees of freedom under the null.

Limitations

- The ANCOVA assumes equal residual variance across the two calibrations. If the residual SDs differ by more than a factor of 2, the F test is anti-conservative. Inspect `residual_se1` and `residual_se2`.
- The test assumes linear calibrations. If either calibration is visibly nonlinear (e.g. a quadratic term is significant), the slope-comparison test is not meaningful.
- The CIs on slope and intercept differences assume the two calibrations are independent. If they share samples, instruments, or operators, this is wrong; the CIs will be too narrow.
- The combined ANCOVA fit uses the same residual variance for both calibrations. If the data clearly show heteroscedasticity between groups, use `nlme::gls` with a `varIdent` variance structure instead.

References

- Draper, N. R. and Smith, H. (1998). Applied Regression Analysis. 3rd ed. Wiley. doi:10.1002/9781118625590.

Examples

```
set.seed(1)
x1 <- seq(0, 10, length.out = 15)
x2 <- seq(0, 10, length.out = 12)
y1 <- 0.5 + 2.0 * x1 + rnorm(length(x1), sd = 0.3)
y2 <- 0.7 + 1.9 * x2 + rnorm(length(x2), sd = 0.3)
m1 <- lm(y1 ~ x1)
m2 <- lm(y2 ~ x2)
cmp <- calibration_compare(m1, m2)
print(cmp)

# Different data frame inputs
df1 <- data.frame(x = x1, y = y1)
df2 <- data.frame(x = x2, y = y2)
cmp2 <- calibration_compare(df1, df2)
print(cmp2)
```

---

calibration_transfer_risk

Statistical framework for calibration transfer risk

Description

Quantifies the risk of transferring a calibration model from a source instrument to a target instrument without full recalibration. The function reports the transfer prediction error, an out-of-distribution flag for unknown samples, and an uncertainty interval that accounts for the additional uncertainty introduced by the transfer.

Usage

```
calibration_transfer_risk(
  source_cal,
  source_data,
  target_data,
  transfer_samples = 6L,
  transfer_model = c("offset", "slope_offset", "none"),
  conf_level = 0.95,
  n_boot = 500L,
  seed = NULL
)
```

Arguments

`source_cal`	an `lm` object from `lm(y ~ x)` on the source instrument. Required.
`source_data, target_data`	data frames each with columns `x` (reference concentration) and `y` (instrument response). Required.

`transfer_samples` integer; number of samples shared between source and target used to estimate the transfer correction (default 6). If fewer common samples are present in `target_data`, this is reduced.

`transfer_model` `"offset"` (constant shift, default), `"slope_offset"` (linear correction), or `"none"` (no transfer correction; baseline for comparison).

`conf_level` confidence level for transfer prediction intervals.

`n_boot` number of bootstrap replicates for transfer uncertainty (default 500).

`seed` optional integer seed.

Value

A `crossstatr_cal_transfer_risk` object.

Methodology

Let $f_s(x) = a_s + b_s x$ be the source calibration. For a target instrument the method models

$$y_t = f_t(x) + \epsilon = g(f_s(x)) + \delta + \epsilon,$$

where $g(\cdot)$ is a transformation (identity for offset-only, affine for slope-offset) and δ captures systematic bias.

The transfer-corrected prediction for a new target response y_t^* at unknown concentration x^* is obtained by inverting the source calibration:

$$\hat{x}^* = f_s^{-1}(g^{-1}(y_t^* - \hat{\delta})),$$

where $\hat{\delta}$ (and the slope correction, if any) are estimated from the shared transfer samples via OLS.

Transfer prediction error (TPE) is the standard deviation of $\hat{x}^* - x_{\text{true}}$ over the target evaluation set.

Out-of-distribution detection: for a new target sample with response y_t^*, the package computes the standardized transfer residual

$$r^* = (y_t^* - \hat{\delta} - g(\hat{x}^*))/\hat{\sigma}_t.$$

If $|r^*| > z_{1-\alpha/2}$, the sample is flagged as out-of-distribution (matrix effect, sensor drift, or different analyte).

Transfer uncertainty combines the source calibration uncertainty, the transfer-correction estimator uncertainty, and the target measurement noise via a parametric bootstrap: simulate source calibration parameters from their MVN distribution, simulate transfer-correction parameters from their MVN, and recompute the inverse prediction; the resulting distribution of $\hat{x}^*$ captures the joint uncertainty.

Reliability summary: a transfer is deemed acceptable if TPE is below a user-supplied tolerance (default: 2x the source calibration standard error) and the OOD rate on the evaluation set is below 5%.

Limitations

- Assumes the source calibration is correctly specified. If the source calibration itself is biased, the transfer correction cannot fix it.
- The OOD detection is univariate; multivariate OOD (e.g. on full spectra) requires a separate approach.
- The bootstrap is parametric; nonparametric bootstrap is an alternative when transfer samples are plentiful (n > 30).
- For drift between source and target acquisition times, use `drift_uncertainty()` jointly with this function.

References

- Wang, Y., Veltkamp, D. J. and Kowalski, B. R. (1991). Multivariate instrument standardization. Analytical Chemistry, 63, 2750-2756. doi:10.1021/ac00023a016.
- Bouveresse, E. and Massart, D. L. (1996). Standardisation of near- infrared spectrometric instruments: A review. Vibrational Spectroscopy, 11, 3-15. doi:10.1016/09242031(95)00055-0.
- Feudale, R. N., Woody, N. A., Tan, H., Myles, A. J., Brown, S. D. and Ferré, J. (2002). Transfer of multivariate calibration models: a review. Chemometrics and Intelligent Laboratory Systems, 64, 181-192. doi:10.1016/S01697439(02)000850.

Examples

```
set.seed(1)
# Source calibration data
x <- seq(0.1, 10, length.out = 30)
y_s <- 2 + 3 * x + rnorm(length(x), sd = 0.2)
source_cal <- lm(y_s ~ x)

# Target: same samples with offset and slope change
y_t <- 2.5 + 2.7 * x + rnorm(length(x), sd = 0.25)
source_data <- data.frame(x = x, y = y_s)
target_data <- data.frame(x = x, y = y_t)

res <- calibration_transfer_risk(source_cal, source_data, target_data)
print(res)
```

---

curve_similarity *Curve similarity between two curves on possibly different x grids*

Description

Quantifies the similarity of two curves (x, y) sampled on possibly different x grids. When `align = TRUE`, both curves are linearly interpolated onto a common grid (the overlap of the two x ranges) before a distance or correlation metric is computed. If `align = FALSE`, the curves must already share the same x coordinates.

Usage

```
curve_similarity(
  x1,
  y1,
  x2,
  y2,
  method = c("l2", "l1", "max", "dtw", "correlation", "frobenius"),
  align = TRUE,
  conf_level = 0.95,
  n_boot = 500L,
  seed = NULL,
  equivalence_threshold = NULL
)
```

Arguments

x1, y1	numeric vectors giving the first curve. x1 should be suitable for stats::approx (typically increasing).
x2, y2	numeric vectors giving the second curve.
method	character; one of "l2", "l1", "max", "dtw", "correlation", "frobenius".
align	logical; if TRUE (default), interpolate both curves onto a common grid. If FALSE, x1 and x2 must be identical.
conf_level	confidence level for the bootstrap or Fisher-z interval.
n_boot	integer; number of bootstrap replicates for the distance CI.
seed	optional integer seed for the bootstrap.
equivalence_threshold	optional non-negative scalar; if supplied, the function reports whether distance < equivalence_threshold.

Details

Several distance definitions are offered: the root-mean-square (L2), the mean absolute (L1), the maximum absolute (max), the dynamic time warping (DTW) cost, the Pearson correlation, and the Frobenius norm. For the correlation method the package returns a Fisher-z confidence interval; for the distance-based methods the package returns a percentile bootstrap interval obtained by resampling the paired residuals.

Equivalence testing is offered as a one-sided statement. If the user supplies an equivalence_threshold (delta) the function reports whether the observed distance falls below it. The package deliberately do not claim equivalence when the observed distance exceeds delta. A small observed distance supports equivalence; a large observed distance is silent on equivalence until a formal two-one-sided-test (TOST) or interval-based equivalence procedure is performed. Absence of evidence is not evidence of absence.

Value

A crossstatr_curve_similarity object with elements:

`distance`	named numeric scalar: the chosen distance / correlation.
`ci_low`, `ci_high`	numeric bounds of the confidence interval.
`method`	method label.
`n_points`	number of grid points used.
`equivalence_threshold`	the threshold if supplied, else `NULL`.
`equivalent`	logical; `TRUE` if `distance` < `equivalence_threshold`, `NA` if no threshold was supplied.
`conf_level`	confidence level used.
`call`	the matched call.

Methodology

L2 distance (root-mean-square):

$$d_{L2} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{1i} - y_{2i})^2}.$$

L1 distance (mean absolute):

$$d_{L1} = \frac{1}{n}\sum_{i=1}^{n}|y_{1i} - y_{2i}|.$$

Maximum absolute deviation:

$$d_{\max} = \max_i |y_{1i} - y_{2i}|.$$

Frobenius norm (unnormalised):

$$d_F = \sqrt{\sum_{i=1}^{n}(y_{1i} - y_{2i})^2}.$$

DTW cost (Sakoe and Chiba, 1978) uses the recursion

$$D[i,j] = |y_{1i} - y_{2j}| + \min\{D[i-1,j], D[i,j-1], D[i-1,j-1]\},$$

with `D[1, 1] = |y_{11} - y_{21}|`. The reported cost is `D[n, m]`. No window constraint is imposed, so the algorithm is O(nm) in both time and memory.

Pearson correlation: $r = \mathrm{cor}(y_1, y_2)$ with Fisher-z confidence interval on $z = \mathrm{atanh}(r)$ of half-width $t_{1-\alpha/2,\,n-3}/\sqrt{n-3}$ (Fisher, 1921).

Bootstrap CI: the paired residuals `e_i = y_{1i} - y_{2i}` are resampled with replacement `n_boot` times and the distance is recomputed on each bootstrap sample. The percentile interval at level `conf_level` is reported. DTW is not bootstrapped because time-ordering is integral to the metric.

Limitations

- Linear interpolation onto a common grid assumes the true curve is piecewise linear between sampled x points. Coarse sampling reduces the fidelity of the interpolated values.
- The DTW implementation is O(nm) in time and memory; with large n it can be slow and memory-hungry.
- Bootstrap CIs are percentile-based and require a reasonable sample size. For `n < 10` they can be unstable.
- Failure to detect a difference is NOT proof of equivalence. The equivalence flag is one-directional: a small observed distance supports equivalence; a large observed distance is silent on equivalence until a formal TOST or interval-inclusion procedure is applied. See Wellek (2010) for equivalence testing methodology.

References

- Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing, 26, 43-49. doi:10.1109/TASSP.1978.1163055.
- Fisher, R. A. (1921). On the probable error of a coefficient of correlation deduced from a small sample. Metron, 1, 3-32. https://digital.library.adelaide.edu.au/items/002ad8fb-c23c-407b-8a89-f036d8da6030.
- Efron, B. and Tibshirani, R. J. (1993). An Introduction to the Bootstrap. Chapman & Hall. doi:10.1007/9781489945419.
- Wellek, S. (2010). Testing Statistical Hypotheses of Equivalence and Noninferiority. 2nd ed. Chapman & Hall/CRC. doi:10.1201/EBK1439808184.

Examples

```
set.seed(1)
t1 <- seq(0, 5, length.out = 50)
t2 <- seq(0, 5, length.out = 80)
y1 <- sin(t1) + rnorm(t1, sd = 0.05)
y2 <- sin(t2 + 0.05) + rnorm(t2, sd = 0.05)
result <- curve_similarity(t1, y1, t2, y2, method = "l2")
print(result)

# DTW handles unequal sampling and small shifts
curve_similarity(t1, y1, t2, y2, method = "dtw")

# Correlation with Fisher-z CI
curve_similarity(t1, y1, t2, y2, method = "correlation")

# Equivalence statement against a tolerance
curve_similarity(t1, y1, t2, y2, method = "l2",
                 equivalence_threshold = 0.2)
```

`detection_reliability` *Reliability analysis for analytical detection*

Description

Produces concentration-dependent detection reliability curves: probability of detection (PD), false-positive probability (FPR), and false-negative probability (FNR), with bootstrap confidence bands. This is the proper frequentist companion to a single-point LOD: it shows how detection performance varies across concentrations, rather than reducing the whole curve to one number.

Usage

```
detection_reliability(
  calibration,
  blank_data = NULL,
  conc_grid = NULL,
  decision_threshold = NULL,
  conf_level = 0.95,
  n_boot = 300L,
  seed = NULL,
  extra_sigma2 = 0
)
```

Arguments

`calibration`	an `lm` object from `lm(y ~ x)`, OR a list with elements `x`, `y`, and `coef`.
`blank_data`	optional numeric vector of blank measurements. If supplied, FPR is estimated empirically; otherwise, it is computed from the calibration residual SD assuming Gaussian noise.
`conc_grid`	numeric vector of concentrations at which to evaluate the reliability curve. If NULL, defaults to 50 points spanning 0 to the upper calibration concentration.
`decision_threshold`	the response threshold above which a sample is called "detected". If NULL, defaults to the upper 95% prediction bound of the blank (the IUPAC decision limit).
`conf_level`	confidence level for the reliability curves.
`n_boot`	number of bootstrap replicates for uncertainty bands (default 300).
`seed`	optional integer seed.
`extra_sigma2`	numeric scalar; additional variance to add to the calibration noise variance, e.g. from matrix effects, batch effects, or drift (see `batch_effect_quantify()`, `drift_uncertainty()`). Defaults to 0.

Value

A `crossstatr_detection_reliability` object.

Methodology

Under the calibration model $y = a + bx + \epsilon$ with $\epsilon \sim N(0, \sigma^2)$, the distribution of a measured response at concentration x_0 is

$$y(x_0) \sim N(a + bx_0, \sigma^2).$$

Given a decision threshold L_C (the critical level), the false-positive probability is

$$\mathrm{FPR} = P(y > L_C \mid x = 0) = 1 - \Phi\left(\frac{L_C - a}{\sigma}\right),$$

and the false-negative probability at concentration x_0 is

$$\mathrm{FNR}(x_0) = P(y \le L_C \mid x = x_0) = \Phi\left(\frac{L_C - a - bx_0}{\sigma}\right).$$

The detection probability is

$$\mathrm{PD}(x_0) = 1 - \mathrm{FNR}(x_0).$$

The classical LOD is the concentration where PD = 0.95 (equivalently, FNR = 0.05), which under Gaussian assumptions is

$$x_{\mathrm{LOD}} = (L_C + z_{0.95}\,\sigma - a)/b.$$

Matrix effects, batch effects, and drift can be incorporated by inflating σ additively:

$$\sigma^2_{\mathrm{eff}} = \sigma^2_{\mathrm{noise}} + \sigma^2_{\mathrm{matrix}} + \sigma^2_{\mathrm{batch}} + \sigma^2_{\mathrm{drift}},$$

provided those variance components have been estimated separately (e.g. via `batch_effect_quantify()`, `drift_uncertainty()`). Pass these as additional variance contributions via `extra_sigma2`.

Bootstrap CIs on PD/FNR are obtained by sampling calibration coefficients from their MVN distribution and recomputing the curves.

Interpretation

A single LOD summarises one decision threshold. The probability curve describes detection across concentrations under the fitted noise model; Currie (1999) supplies the analytical detection framework.

Limitations

- The Gaussian noise assumption must hold. Heavy-tailed noise (e.g. from sporadic contamination) will inflate FPR beyond the nominal rate.
- The default decision threshold is computed from the calibration data; if the user supplies a different one, it must be reported alongside the curves.
- Reliability curves assume the calibration is valid across the entire `conc_grid` range. Extrapolating below the lowest calibration concentration is risky.
- The bootstrap CI assumes the calibration model is correctly specified.

References

- Currie, L. A. (1999). Detection and quantification limits: origins and historical overview. Analytica Chimica Acta, 391, 127-134. doi:10.1016/S00032670(99)001051.
- Currie, L. A. (1995). Nomenclature in evaluation of analytical methods including detection and quantification capabilities (IUPAC Recommendations 1995). Pure and Applied Chemistry, 67, 1699-1723. doi:10.1351/pac199567101699.
- Hubaux, A. and Vos, G. (1970). Decision and detection limits for calibration curves. Analytical Chemistry, 42, 849-855. doi:10.1021/ac60290a013.

Examples

```
set.seed(1)
x <- seq(0.1, 10, length.out = 25)
y <- 0.5 + 0.9 * x + rnorm(length(x), sd = 0.3)
cal <- lm(y ~ x)
res <- detection_reliability(cal, blank_data = rnorm(20, 0.5, 0.3))
print(res)
```

---

drift_correct_stat *Drift correction with uncertainty quantification*

Description

Corrects a signal for instrumental drift by fitting a smooth drift function of time to reference standards and subtracting the fitted drift from the signal, anchored at a reference time. The drift model is one of: linear, polynomial, or first-order exponential approach to asymptote.

Usage

```
drift_correct_stat(
  y,
  time,
  reference = NULL,
  drift_model = c("linear", "polynomial", "exponential"),
  degree = 2L,
  conf_level = 0.95
)
```

Arguments

y	numeric vector of signal intensities.
time	numeric vector of measurement times, same length as y.
reference	optional data frame with columns time and y containing reference-standard measurements taken at known times. If NULL, the drift model is fit to a running median of y (see Description).

drift_model	character; one of "linear", "polynomial", or "exponential".
degree	integer; polynomial degree for drift_model = "polynomial". Default 2.
conf_level	confidence level for the correction uncertainty band.

Details

When reference = NULL, the drift model is fit to a running median of the signal itself. This is a less rigorous fallback: the running median conflates drift with genuine slow signal variation, so the correction will be biased if the signal is not approximately constant in the absence of drift. The documentation recommends supplying real reference-standard measurements whenever the experimental design allows it.

The correction uncertainty is propagated from the drift model via the delta method. For the linear and polynomial models this is the standard formula for the standard error of a fitted value, and for the exponential model it uses the Jacobian of the prediction function evaluated at the parameter estimates. The resulting SE is on the corrected signal at each time point.

Value

A crossstatr_drift object with elements:

time, y_original, y_corrected	vectors after alignment to time.
drift_fit	fitted drift values at each time point.
drift_se	delta-method standard error of the drift fit at each time point.
correction_ci	n x 2 matrix giving the CI on the corrected signal.
model_summary	list of model-specific summary statistics (coefficients, R^2 where applicable, residual SE).
call	the matched call.

Methodology

The drift model is fit to the pairs (t_j, r_j) where r_j is either the reference-standard reading at time t_j (when reference is supplied) or the running median of y. For the linear model the drift is

$$\hat{d}(t) = \hat{\alpha} + \hat{\beta}\, t,$$

fit by OLS. For the polynomial model,

$$\hat{d}(t) = \sum_{k=0}^{m} \hat{\beta}_k\, t^k,$$

with m = degree. For the exponential model,

$$\hat{d}(t) = \hat{a}\,(1 - \exp(-\hat{b}\, t)),$$

fit by stats::nls.

The corrected signal is

$$\hat{y}_i^{\mathrm{corr}} = y_i - \hat{d}(t_i) + \hat{d}(t_{\mathrm{ref}}),$$

where t_{ref} is the median of the measurement times. This anchors the corrected signal at t_{ref}: the mean of the corrected signal equals the mean of the original signal at that time.

The delta-method variance of the drift fit at time t is

$$\mathrm{Var}(\hat{d}(t)) = J(t)^\top \, V_\theta \, J(t),$$

where V_θ is the parameter covariance matrix and $J(t)$ is the Jacobian of $\hat{d}(t)$ with respect to the parameters. The SE on the correction $\hat{d}(t_i) - \hat{d}(t_{\mathrm{ref}})$ additionally accounts for the covariance between the two fitted values:

$$\mathrm{Var}(\hat{d}(t_i) - \hat{d}(t_{\mathrm{ref}})) = \mathrm{Var}(\hat{d}(t_i)) + \mathrm{Var}(\hat{d}(t_{\mathrm{ref}})) - 2\,\mathrm{Cov}(\hat{d}(t_i), \hat{d}(t_{\mathrm{ref}})).$$

Limitations

- The drift model assumes the drift is a smooth function of time. If the instrument suffers step changes (e.g. a calibration is re-applied mid-run), no smooth model will fit well. Inspect the residuals before trusting the correction.
- The exponential model assumes a monotonic approach to an asymptote. It will not fit a drift that reverses direction.
- When `reference = NULL`, the drift model is fit to a running median of the signal, which conflates drift with genuine signal variation. This is only defensible when the signal is expected to be approximately constant in the absence of drift.
- The delta-method SE is a local linearisation and may under-cover for highly nonlinear drift models. For critical applications, use a parametric bootstrap.

References

- Bates, D. M. and Watts, D. G. (1988). Nonlinear Regression Analysis and Its Applications. Wiley. doi:10.1002/9780470316757.
- Oehlert, G. W. (1992). A Note on the Delta Method. The American Statistician, 46, 27-29. doi:10.1080/00031305.1992.10475842.

Examples

```
set.seed(1)
time <- seq(0, 10, length.out = 100)
drift <- 0.5 + 0.1 * time
signal <- 5 * exp(-(time - 5)^2 / 2)
y <- signal + drift + rnorm(length(time), sd = 0.1)
# Reference standards at known times
ref_df <- data.frame(
  time = seq(0.5, 9.5, by = 1),
  y = 0.5 + 0.1 * seq(0.5, 9.5, by = 1) + rnorm(10, sd = 0.05))
fit <- drift_correct_stat(y, time, reference = ref_df,
                          drift_model = "linear")
print(fit)

# Fallback to running median when no reference is supplied
fit2 <- drift_correct_stat(y, time, drift_model = "exponential")
print(fit2)
```

`drift_uncertainty` *Statistical attribution of uncertainty to instrument drift*

Description

Decomposes the total measurement variance into a component attributable to instrument drift and a residual noise component. Drift is modelled as a smooth function of time, fit jointly with calibration parameters when reference standards are available.

Usage

```
drift_uncertainty(
  y,
  time,
  conc = NULL,
  drift_model = c("linear", "polynomial", "exponential"),
  degree = 2L,
  n_boot = 300L,
  conf_level = 0.95,
  seed = NULL
)
```

Arguments

`y`	numeric vector of measurements (responses, not concentrations).
`time`	numeric vector of acquisition times (e.g. seconds or hours).
`conc`	optional numeric vector of known concentrations (for calibration-based drift estimation). If supplied, drift is fit on the calibration residuals; otherwise drift is fit on `y` directly (less rigorous).
`drift_model`	`"linear"`, `"polynomial"`, or `"exponential"`.
`degree`	polynomial degree when `drift_model = "polynomial"`.
`n_boot`	number of bootstrap replicates for drift-parameter uncertainty (default 300).
`conf_level`	confidence level for CIs.
`seed`	optional integer seed.

Value

A `crossstatr_drift_uncertainty` object.

Methodology

Model 1 (no calibration, when `conc` is NULL):

$$y_i = \mu + d(t_i) + \epsilon_i, \qquad \epsilon_i \sim N(0, \sigma^2),$$

with $d(t)$ a smooth drift function. The drift variance is

$$\sigma^2_{\text{drift}} = \text{Var}[\hat{d}(t_i)],$$

and the noise variance is

$$\sigma^2_\epsilon = \frac{1}{n-p}\sum_i (y_i - \hat{\mu} - \hat{d}(t_i))^2.$$

The **fraction of variance attributed to drift** is

$$f_{\text{drift}} = \sigma^2_{\text{drift}} / (\sigma^2_{\text{drift}} + \sigma^2_\epsilon).$$

Model 2 (with calibration): jointly fit

$$y_i = a + b\,x_i + d(t_i) + \epsilon_i,$$

with `x = conc`. Drift is then the part of the residual not explained by the calibration.

Drift-parameter uncertainty is obtained by parametric bootstrap: simulate drift parameters from their MVN distribution (covariance = inverse of the Fisher information), refit, and report the bootstrap distribution of the drift variance. The CI on the drift fraction is the bootstrap percentile interval.

Temporal correlation in the residuals is assessed via the lag-1 autocorrelation of the post-drift residual; if this exceeds `2/sqrt(n)`, the independence assumption is questionable and the reported drift fraction may be biased (typically downward) because correlated noise is partially absorbed into the drift estimate.

Limitations

- Assumes drift is a smooth function of time. Sudden jumps (e.g. sensor replacement mid-run) are not modelled.
- If drift and concentration are confounded (e.g. low-conc samples all measured early, high-conc late), the drift estimate is biased. Check the design: if `cor(time, conc)` exceeds 0.5 in absolute value, treat results with caution.
- Temporal correlation in the residuals will inflate apparent drift. Always inspect the residual ACF.
- The bootstrap CI assumes the fitted drift model is correctly specified; under model misspecification the CI may be too narrow.

References

- Seber, G. A. F. and Wild, C. J. (1989). Nonlinear Regression. Wiley. doi:10.1002/0471725315.
- Davison, A. C. and Hinkley, D. V. (1997). Bootstrap Methods and their Application. Cambridge University Press. doi:10.1017/CBO9780511802843.

Examples

```
set.seed(1)
t <- seq(0, 10, length.out = 60)
drift_true <- 0.5 * (1 - exp(-0.3 * t))
y <- 2 + 1.5 * t + drift_true + rnorm(length(t), sd = 0.2)
res <- drift_uncertainty(y, time = t, drift_model = "exponential")
print(res)
```

`experiment_fragility` *Quantitative experimental fragility assessment*

Description

Quantifies how sensitive a reported result is to defensible analysis decisions: which replicates to include, how to handle outliers, which baseline correction to use, which model to fit, and how to propagate measurement uncertainty. This is a sensitivity analysis, not a confirmatory test; it does not change the original result.

Usage

```
experiment_fragility(
  analysis_fn,
  data,
  replicates = NULL,
  outlier_methods = c("none", "iqr", "mad"),
  baseline_methods = c("none"),
  model_alternatives = NULL,
  measurement_uncertainties = NULL,
  decision_threshold,
  n_boot = 200L,
  seed = NULL
)
```

Arguments

`analysis_fn`	function that takes a list of options and returns a numeric scalar (or a small named vector) summary statistic of interest. Required.
`data`	data frame passed to `analysis_fn` as the `data` argument.
`replicates`	optional character; name of the column identifying replicates. If supplied, leave-one-replicate-out is computed.
`outlier_methods`	character vector; subset of `c("none", "iqr", "mad")`. Defaults to all three.
`baseline_methods`	character vector; subset of `c("none", "asls", "lowess")`. Only used if `analysis_fn` accepts a `baseline` argument.
`model_alternatives`	optional list of named alternative model specifications. Only used if `analysis_fn` accepts a `model` argument.
`measurement_uncertainties`	optional numeric vector of SDs, used to compute a fragility index: the smallest noise level at which the result loses significance (i.e. crosses a threshold).
`decision_threshold`	numeric scalar; the value that defines a "decision" (e.g. p = 0.05, or zero for effect sizes). Required for the fragility index.

n_boot	number of bootstrap perturbations for the fragility index search (default 200).
seed	optional integer seed.

Value

A crossstatr_fragility object.

Methodology

The function systematically perturbs analysis decisions and measures the change in the result:

- **Leave-one-replicate-out**: recompute the statistic excluding each replicate in turn. The fragility index is the number of replicates whose exclusion flips the decision.
- **Outlier handling**: recompute under each outlier rule (none, IQR, MAD). The fragility index is 1 if any rule flips the decision.
- **Baseline correction**: recompute under each baseline method. The fragility index is 1 if any baseline flips the decision.
- **Model selection**: recompute under each alternative model.
- **Measurement uncertainty**: progressively inject Gaussian noise at increasing SD, recompute, and find the smallest SD at which the decision flips. This is the **noise fragility threshold**.

The overall fragility score is

$$F = \sum_k w_k F_k,$$

where $F_k \in [0, 1]$ is the fraction of perturbations of type k that flip the decision, and w_k are user-supplied weights (uniform by default).

The function reports the original result, the range across all perturbations, and the most influential decision (the one that produces the largest swing).

Comparison with established alternatives

- **Fragility index** (Walsh et al., 2014) in clinical trials: counts the number of events that would need to change to flip statistical significance. This function generalizes that idea to continuous outcomes and to a wider range of analysis decisions.
- **Multiverse analysis** (Steegen et al., 2016): a general framework for exploring analysis-decision sensitivity. This function is a focused, quantitative implementation for experimental science.

Limitations

- This is exploratory sensitivity analysis, not confirmatory inference. The reported fragility score does not have a known reference distribution; do not treat it as a p-value.
- The fragility index counts "decision flips" with respect to a user-supplied threshold. The threshold itself is a modeling decision and should be reported alongside the score.
- The measurement-uncertainty injection assumes additive Gaussian noise; multiplicative or correlated noise is not modelled.
- For continuous statistics (e.g. effect sizes), use the range and the most-influential-decision summary; the binary fragility index is less informative.

References

- Walsh, M., Srinathan, S. K., McAuley, D. F. et al. (2014). The statistical significance of randomized controlled trial results is frequently fragile: a case for a Fragility Index. Journal of Clinical Epidemiology, 67, 622-628. doi:10.1016/j.jclinepi.2013.10.019.
- Steegen, S., Tuerlinckx, F., Gelman, A. and Vanpaemel, W. (2016). Increasing Transparency Through a Multiverse Analysis. Perspectives on Psychological Science, 11, 702-712. doi:10.1177/1745691616658637.

Examples

```
set.seed(1)
df <- data.frame(
  y = c(rnorm(15, mean = 1), rnorm(15, mean = 1.5)),
  grp = factor(rep(c("A", "B"), each = 15)),
  rep = factor(rep(1:15, times = 2)))
# Analysis function: returns mean difference
af <- function(opts) {
  d <- opts$data
  if (!is.null(opts$rep_lo)) d <- d[d$rep != opts$rep_lo, ]
  diff(tapply(d$y, d$grp, mean))
}
res <- experiment_fragility(
  analysis_fn = af,
  data = df,
  replicates = "rep",
  decision_threshold = 0)
print(res)
```

`kinetic_discrimination` *Discriminability of competing kinetic models*

Description

Given two or more candidate kinetic models and the proposed experimental design (sampling times, measurement noise), this function quantifies the probability that the data will allow discrimination between the models. It is a pre-experimental planning tool: run it before collecting data to choose sampling times that maximize discriminability.

Usage

```
kinetic_discrimination(
  models,
  starts,
  true_model = 1L,
  true_params,
  times,
```

```
  sigma,
  n_sim = 500L,
  conf_level = 0.95,
  criterion = c("aic", "bic"),
  alternative_times = NULL,
  seed = NULL
)
```

Arguments

`models`	list of named functions. Each function takes `c(params, t)` and returns predicted concentration.
`starts`	list of starting parameter vectors for the models.
`true_model`	integer or character; which model is assumed to be the data-generating mechanism. Defaults to 1.
`true_params`	numeric vector of true parameters for the generating model.
`times`	numeric vector of candidate sampling times.
`sigma`	measurement noise SD.
`n_sim`	number of Monte Carlo simulations (default 500).
`conf_level`	confidence level for discrimination probability.
`criterion`	`"aic"` (default) or `"bic"` for model selection.
`alternative_times`	optional list of alternative sampling schedules to compare.
`seed`	optional integer seed.

Value

A `crossstatr_kinetic_discrimination` object.

Methodology

For each simulation $b = 1, \ldots, B$:

1. Simulate data $y_i^{(b)} = m^*(t_i; \theta^*) + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$ and m^* is the generating model with true parameters θ^*.
2. Fit each candidate model m_k via nonlinear least squares to obtain $\hat{\theta}_k^{(b)}$ and the model's AIC/BIC.
3. Select the model with the lowest AIC (or BIC).

The **discrimination probability** for model k is

$$P_k = \frac{1}{B} \sum_b \mathbf{1}\{\hat{k}^{(b)} = k\}.$$

The **power to distinguish model k from model j** is computed under the alternative that k is the true model:

$$\pi_{k \to j} = P(\mathrm{AIC}_k < \mathrm{AIC}_j \mid m_k \text{ true}).$$

Parameter identifiability for each model is assessed via the condition number κ of the sensitivity matrix $G_{ij} = \partial m_i / \partial \theta_j$. $\kappa > 10^6$ indicates practical non-identifiability.

Model separation (the geometric mean distance between model trajectories at the proposed sampling times, in units of sigma) is reported as a quick summary of design quality.

Limitations

- The simulated measurement noise is assumed homoscedastic Gaussian. Real experiments with heteroscedastic or correlated noise will give different discrimination rates.
- If models are very similar in functional form, very large `n_sim` may be needed for a precise discrimination probability estimate. The MCSE of P_k is $\sqrt{P_k(1-P_k)/B}$.
- Models that are mathematically nested can be discriminated via likelihood-ratio tests, which are more powerful than AIC for this purpose. Use `kinetic_model_compare()` for nested comparisons.
- This function assumes `nls` will converge; in practice, non-convergence of one model can bias the discrimination probability.

References

- Asprey, S. P. and Macchietto, S. (2000). Statistical tools for optimal dynamic model building. Computers & Chemical Engineering, 24, 1261-1267. doi:10.1016/S00981354(00)003288.

Examples

```
set.seed(1)
m1 <- function(p, t) p[1] * exp(-p[2] * t)
m2 <- function(p, t) p[1] * (1 + p[3] * t) * exp(-p[2] * t)
res <- kinetic_discrimination(
  models = list(simple = m1, extended = m2),
  starts = list(c(A = 1, k = 0.1), c(A = 1, k = 0.1, alpha = 0.05)),
  true_model = 1L, true_params = c(1, 0.1),
  times = seq(0, 50, length.out = 12), sigma = 0.05, n_sim = 50)
print(res)
```

`kinetic_model_compare` *Compare candidate kinetic models*

Description

Fits a set of candidate kinetic models to time-concentration data and reports side-by-side AIC, BIC, residual sum of squares, parameter estimates with confidence intervals, and identifiability diagnostics. A best model is selected by AIC, and AIC deltas are reported so the analyst can see whether one model is clearly superior or whether several models are essentially indistinguishable.

Usage

```
kinetic_model_compare(
  time,
  conc,
  models = NULL,
  start = NULL,
  conf_level = 0.95
)
```

Arguments

time	numeric vector of sampling times.
conc	numeric vector of measured concentrations (same length as time).
models	named list of functions, each taking (params, t) and returning predicted concentrations. If NULL, default models are used: first-order decay, second-order decay, biexponential decay, and Michaelis-Menten saturation.
start	named list of starting values, one named numeric vector per model. If NULL, data-driven defaults are computed from time and conc.
conf_level	confidence level for parameter intervals.

Value

A crossstatr_kinetic_compare object with elements:

comparison	data frame with one row per model: model, AIC, BIC, RSS, n_params, condition_number, identifiable, delta_AIC.
best_model	name of the model with smallest AIC.
parameter_estimates	named list of data frames (one per model) with columns parameter, estimate, se, ci_low, ci_high.
models	the model functions used.
call	the matched call.

Methodology

Each model is fit by minimising the residual sum of squares

$$\mathrm{RSS}(\theta) = \sum_{i=1}^{n} (y_i - f(\theta, t_i))^2$$

with stats::nlminb (box-constrained to non-negative parameters).

AIC and BIC use the Gaussian likelihood form

$$\mathrm{AIC} = n \log(\mathrm{RSS}/n) + 2k, \qquad \mathrm{BIC} = n \log(\mathrm{RSS}/n) + k \log(n),$$

where k is the number of parameters. AIC delta is $\Delta_i = \mathrm{AIC}_i - \min_j \mathrm{AIC}_j$.

Standard errors for parameters are computed via linearisation. With sensitivity matrix $J_{ij} = \partial f_i / \partial \theta_j$ (computed by central finite differences), the parameter covariance is $\mathrm{Cov}(\hat{\theta}) \approx \hat{\sigma}^2 (J'J)^{-1}$, with $\hat{\sigma}^2 = \mathrm{RSS}/(n-k)$. Confidence intervals use a Student-t critical value with `n - k` df.

Identifiability is diagnosed by the condition number $\kappa(J) = \sigma_{\max}(J)/\sigma_{\min}(J)$. When $\kappa > 10^8$ the parameters are flagged as poorly identifiable, following the rule of thumb discussed in Seber and Wild (2003).

Limitations

- The default optimiser is `nlminb` with non-negativity bounds. Models that need signed parameters will require a user-supplied wrapper.
- Standard errors derived from `J'J` are asymptotic and can be unreliable for small samples or near-flat likelihoods. Profile-likelihood intervals (`stats::confint` on an `nls` object) are more accurate but require refitting with `nls`; this function does not do that.
- Likelihood-ratio tests for nested models are not computed automatically because detecting nesting from a black-box function is not reliable. The user should inspect AIC deltas and, where nesting is known, perform an LRT manually using the RSS values.
- Identical or near-identical fits across models are common when the data are sparse; a small AIC delta (< 2) should be interpreted as evidence of model non-identifiability, not as evidence for the simpler model.

References

- Seber, G. A. F. and Wild, C. J. (1989). Nonlinear Regression. Wiley. doi:10.1002/0471725315.
- Akaike, H. (1974). A new look at the statistical model identification. IEEE Transactions on Automatic Control, 19, 716-723. doi:10.1109/TAC.1974.1100705.
- Bates, D. M. and Watts, D. G. (1988). Nonlinear Regression Analysis and Its Applications. Wiley. doi:10.1002/9780470316757.

Examples

```
set.seed(1)
t <- seq(0, 10, length.out = 40)
conc <- 5 * exp(-0.3 * t) + rnorm(t, sd = 0.1)
result <- kinetic_model_compare(t, conc)
print(result)

# Parameter estimates for the best model
result$parameter_estimates[[result$best_model]]
```

noise_characterize *Characterize measurement noise*

Description

Decomposes a sampled signal into a smooth trend and a residual noise component, then summarises the noise: overall standard deviation, lag-1 through lag-10 autocorrelation, heteroscedasticity (correlation of absolute residuals with the fitted trend), a power spectral density estimate via FFT with a dominant-frequency pick, and signal-dependent noise (residual SD within bins of the fitted signal level).

Usage

```
noise_characterize(
  x,
  y,
  method = c("residual", "difference"),
  degree = 3L,
  conf_level = 0.95
)
```

Arguments

x	numeric vector of abscissae (time, wavelength, etc.).
y	numeric vector of measurements (same length as x).
method	"residual" (default) or "difference".
degree	integer; polynomial degree for method = "residual".
conf_level	confidence level used for the heteroscedasticity test and the dominant-frequency CI label.

Details

Two trend-removal strategies are available. method = "residual" fits a polynomial of degree and takes the residuals. method = "difference" uses successive differences y[i+1] - y[i] as a noise proxy, which is appropriate when the signal is smooth relative to the sampling rate so that differences are dominated by noise rather than signal curvature.

Value

A crossstatr_noise object with elements:

noise_sd	overall standard deviation of the residual series.
acf	named numeric vector of autocorrelations at lags 1 to 10.
heteroscedasticity_r	Pearson correlation of |residual| with the fitted signal.

`heteroscedasticity_p`
: two-sided p-value of the correlation test.

`dominant_freq`
: frequency (in cycles per unit of `x`) with the largest PSD component (excluding DC).

`psd`
: data frame of `freq` and `power` (positive frequencies only).

`signal_dependent`
: data frame of `bin`, `mean_signal`, `sd_noise` summarising residual SD within bins of the fitted trend.

`method`
: method label.

`call`
: the matched call.

Methodology

For `method = "residual"` The package fits $y_i = \beta_0 + \sum_{k=1}^{d} \beta_k x_i^k + \epsilon_i$ by ordinary least squares. The residual series $e_i = y_i - \hat{y}_i$ is taken as the noise realisation.

For `method = "difference"` The package uses $e_i = y_{i+1} - y_i$. If the underlying signal is locally linear and the noise has variance σ_ϵ^2, then $\mathrm{Var}(e_i) \approx 2\sigma_\epsilon^2$, so the original noise SD is approximately $\mathrm{sd}(e)/\sqrt{2}$.

Autocorrelation at lag k:

$$\rho_k = \frac{\mathrm{Cov}(e_i, e_{i+k})}{\mathrm{Var}(e_i)}.$$

Heteroscedasticity is tested by the Pearson correlation between $|e_i|$ and $\hat{y}_i$. This is an exploratory correlation diagnostic, not the White or studentized Breusch-Pagan heteroscedasticity test.

The power spectral density is computed via the periodogram

$$P(\omega) = \frac{1}{n} \left| \sum_{i=1}^{n} e_i e^{-i 2\pi \omega x_i / T} \right|^2,$$

with `T` the span of `x`. The dominant non-DC frequency is reported.

For signal-dependent noise the fitted values are binned into approximately 10 equal-probability bins, and the residual SD is computed within each bin.

Limitations

- The polynomial trend is global; for sharply varying signals it can over-fit in some regions and under-fit in others. A smoothing spline or local polynomial regression is preferable for non-smooth trends; this function does not yet offer that option.
- The difference method assumes the signal is approximately linear between consecutive samples. If the sampling is sparse relative to signal curvature, the differences will be dominated by signal, not noise, and `noise_sd` will be inflated.
- The periodogram has high variance as a spectral estimator; the dominant-frequency pick is noisy for short series. Use Welch's method or a multitaper estimator for serious spectral analysis.
- Heteroscedasticity as tested here only flags a monotone relation between residual magnitude and signal level; it does not detect other patterns (e.g. variance depending on time).

References

- Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (2008). Time Series Analysis: Forecasting and Control. 4th ed. Wiley. doi:10.1002/9781118619193.

Examples

```
set.seed(1)
x <- seq(0, 10, length.out = 200)
y <- sin(x) + rnorm(x, sd = 0.1 + 0.05 * abs(sin(x)))
result <- noise_characterize(x, y, method = "residual", degree = 5L)
print(result)
head(result$psd)
head(result$signal_dependent)

# Difference method
noise_characterize(x, y, method = "difference")
```

---

parameter_uncertainty *Propagate parameter uncertainty to derived quantities*

Description

Given a parameter estimate and its variance-covariance matrix, propagate uncertainty through a function `func(params) -> g(params)` to obtain a point estimate, standard error, and confidence interval for g. Three methods are offered: the delta method (linearisation), Monte Carlo simulation from a multivariate normal, and a nonparametric bootstrap that resamples the original data and refits.

Usage

```
parameter_uncertainty(
  params,
  cov,
  func,
  method = c("delta", "monte_carlo", "bootstrap"),
  n_sim = 1000L,
  conf_level = 0.95,
  seed = NULL,
  original_data = NULL
)
```

Arguments

`params`	numeric vector of parameter estimates.
`cov`	variance-covariance matrix of `params` (square, same length).

`func`	function taking `params` and returning a numeric scalar or vector. The function defines the derived quantity g.
`method`	`"delta"`, `"monte_carlo"`, or `"bootstrap"`.
`n_sim`	integer; number of Monte Carlo or bootstrap replicates.
`conf_level`	confidence level for the interval.
`seed`	optional integer seed.
`original_data`	required for `method = "bootstrap"`: a list with elements `data` (the original data frame) and `refit` (a function taking a data subset and returning a parameter vector of the same length as `params`). If `NULL` and `method = "bootstrap"`, the function falls back to the delta method with a warning.

Details

The delta method is fast and asymptotically exact, but only accurate when `func` is well-approximated by a linear function in the neighbourhood of the parameter estimate. Monte Carlo captures nonlinearity but assumes the parameter estimate is approximately Gaussian (the same assumption that motivates the t-based CI of the original fit). Bootstrap makes the fewest distributional assumptions but requires the original data and a refit function.

Value

A `crossstatr_param_uncert` object with elements:

`estimate`	numeric vector (or scalar) of point estimates of g.
`se`	standard error(s) of g.
`ci_low`, `ci_high`	confidence interval bounds.
`method`	method label.
`jacobian`	the Jacobian $J = \partial g / \partial \theta$ (only for `"delta"`); `NULL` otherwise.
`sim_samples`	matrix of simulated g values (rows = replicates, columns = output components); `NULL` for delta.
`call`	the matched call.

Methodology

Delta method (Oehlert, 1992). For a differentiable function $g : \mathbb{R}^p \to \mathbb{R}^q$, the first-order Taylor expansion around $\hat{\theta}$ gives

$$g(\hat{\theta}) \approx g(\theta) + J(\theta)'(\hat{\theta} - \theta),$$

where $J_{ij} = \partial g_i / \partial \theta_j$ is computed numerically by central finite differences. The variance of $g(\hat{\theta})$ is approximated by

$$\mathrm{Cov}(g(\hat{\theta})) \approx J \, \mathrm{Cov}(\hat{\theta}) \, J'.$$

Confidence intervals use a Student-t critical value with `length(params)` df, matching the common choice in nonlinear regression software.

Monte Carlo: draw $\theta^{(b)} \sim \mathrm{MVN}(\hat{\theta}, \hat{\Sigma})$ for `b = 1, ..., n_sim`, compute $g^{(b)} = g(\theta^{(b)})$, and report the sample mean, sample SD, and the empirical quantiles of the simulated distribution.

Bootstrap: for each replicate `b`, draw `n` rows of `original_data` with replacement, refit the model via `refit(data_b)`, and compute $g^{(b)} = g(\hat{\theta}^{(b)})$. Percentile intervals are reported. Refit failures are dropped.

Limitations

- The delta method can be severely biased for highly nonlinear `func` or near parameter boundaries. Always compare against Monte Carlo if in doubt.
- Monte Carlo assumes the parameter estimate is approximately Gaussian. For small samples or constrained parameters this is optimistic.
- The bootstrap requires the original data and a working refit function. If `refit` fails on a substantial fraction of resamples, the resulting interval is biased. Refit-failure rate is not currently reported.
- The covariance matrix `cov` is taken as-is; it is not symmetrised or projected to the nearest positive-definite matrix.

References

- Oehlert, G. W. (1992). A Note on the Delta Method. The American Statistician, 46, 27-29. doi:10.1080/00031305.1992.10475842.
- Efron, B. and Tibshirani, R. J. (1993). An Introduction to the Bootstrap. Chapman & Hall. doi:10.1007/9781489945419.

Examples

```
set.seed(1)
# Fit a simple linear model y = a + b*x and propagate to g(a,b) = -a/b
x <- 1:20
y <- 2 + 0.5 * x + rnorm(x, sd = 1)
fit <- lm(y ~ x)
params <- coef(fit)
covmat <- vcov(fit)
g <- function(p) -p[1] / p[2]  # x-intercept

# Delta method
r_delta <- parameter_uncertainty(params, covmat, g, method = "delta")
print(r_delta)

# Monte Carlo
r_mc <- parameter_uncertainty(params, covmat, g, method = "monte_carlo",
                              n_sim = 1000, seed = 1)
print(r_mc)
```

---

peak_evidence *Statistical evidence assessment for signal peaks*

Description

Quantifies the evidence that a candidate feature in a measured signal is a true peak rather than a noise fluctuation. The function reports an exploratory evidence summary (likelihood ratio and an exploratory resampling tail fraction); it does not report Bayesian posterior probabilities because doing so would require specifying a prior, which the package does not presume.

Usage

```
peak_evidence(
  x,
  y,
  peak_index = NULL,
  baseline = NULL,
  noise_sd = NULL,
  conf_level = 0.95,
  method = c("lr", "snr"),
  n_perm = 999L,
  seed = NULL
)
```

Arguments

x	numeric vector of x-values (e.g. wavelength, m/z, voltage).
y	numeric vector of signal intensities.
peak_index	integer vector of candidate peak positions (indices into x and y). If NULL, peaks are auto-detected via local maxima with prominence above a noise threshold.
baseline	numeric vector of baseline estimates (same length as y); if NULL, a simple running-minimum baseline is used.
noise_sd	estimate of noise SD. If NULL, estimated from the MAD of the baseline-subtracted signal in flat regions.
conf_level	confidence level for the peak amplitude CI.
method	"lr" (default, likelihood ratio) or "snr" (signal-to- noise threshold with Bonferroni correction).
n_perm	number of permutations for the empirical null distribution of the peak height (default 999). Set to 0 to use the Gaussian null approximation.
seed	optional integer seed.

Value

A crossstatr_peak_evidence object with per-peak evidence table.

Methodology

Candidate amplitude is the baseline-subtracted value

$$A = y_{i_0} - b_{i_0}.$$

The reported Gaussian likelihood-style score is

$$\Lambda = \exp(A^2/(2\sigma^2)).$$

This alternative-to-null ratio is not the null-to-alternative ratio used in the conventional Wilks statistic. No chi-square calibration or exact false-positive guarantee is established for this score.

For resampling, a smoothing spline is retained, its residuals are shuffled, and a maximum local contrast across the supplied candidate indices is compared with the observed amplitude. The reported tail fraction uses a plus-one correction. The candidate set is not re-detected within each resample, and the observed and resampled amplitude definitions differ. Local simulations found poor null calibration and low detection rates; the result is exploratory.

The Gaussian SNR threshold uses $z_{1-\alpha/(2m)}$ for m candidates. With `n_perm = 0`, the implemented tail formula divides by m; it is not a valid Bonferroni-adjusted p-value. Peak-location uncertainty is a threshold-crossing half-width, not a calibrated confidence interval. Wilks (1938) and Currie (1968) provide background on likelihood inference and analytical detection; neither validates this implementation.

Limitations

- The likelihood-ratio form assumes a known noise SD; using an estimated SD inflates the apparent evidence in small samples. Neither the Gaussian nor permutation output has established calibration for small samples.
- Overlapping peaks are not modelled jointly; if two candidate peaks are within one FWHM of each other, the LR for each is optimistic. Use `weighted_peak_fit()` for proper deconvolution.
- Auto-detected peaks at the signal boundaries are unreliable because the baseline estimate is poorly constrained there.
- This is a frequentist evidence measure, not a posterior probability. If the analyst require Bayesian posterior probabilities, specify the prior explicitly and use a different tool.

References

- Wilks, S. S. (1938). The Large-Sample Distribution of the Likelihood Ratio for Testing Composite Hypotheses. The Annals of Mathematical Statistics, 9, 60-62. doi:10.1214/aoms/1177732360.
- Currie, L. A. (1968). Limits for qualitative detection and quantitative determination. Application to radiochemistry. Analytical Chemistry, 40, 586-593. doi:10.1021/ac60259a007.

Examples

```
set.seed(1)
x <- seq(400, 800, by = 0.5)
y <- dnorm(x, mean = 550, sd = 8) * 5 + dnorm(x, mean = 700, sd = 5) * 3 +
     0.1 + rnorm(length(x), sd = 0.05)
res <- peak_evidence(x, y)
print(res)
```

`replicate_decompose` *Variance component decomposition for nested and crossed replicates*

Description

Decomposes the total variance of a measurement into contributions from each level of replication: biological, technical, and residual. The decomposition is done by a linear mixed model fit by `lme4::lmer` with REML. The formula should contain random-effects terms such as `(1|bio) + (1|bio:tech)` to encode the nesting or crossing structure.

Usage

```
replicate_decompose(formula, data, type = "lmm", conf_level = 0.95)
```

Arguments

`formula`	a model formula containing random-effects terms, e.g. `y ~ treatment + (1|bio) + (1|bio:tech)`.
`data`	data frame containing the variables in `formula`.
`type`	character; currently only `"lmm"` is supported.
`conf_level`	confidence level for the variance component CIs.

Details

Variance components are read from `lme4::VarCorr`, and Wald-type confidence intervals are constructed by treating the standard deviations (square root of the diagonal of each `VarCorr` block) as approximately normal. This is the standard large-sample approximation; for designs with few levels per random effect, profile-likelihood CIs from `confint(model, method = "profile")` are more accurate and can be obtained directly from the stored `model` element.

Value

A `crossstatr_replicate` object with elements:

`variance_components`	data frame with columns `component`, `variance`, `sd`, `proportion`, `ci_low`, `ci_high`. The `Residual` row is always last.
`model`	the underlying `lmerMod` fit.
`method`	backend used (`"LMM (REML)"`).
`call`	the matched call.

Methodology

For the linear mixed model

$$y_{ij} = X_{ij}\beta + \sum_l b^{(l)}_{g_l(i,j)} + \epsilon_{ij},$$

where $b^{(l)} \sim N(0, \sigma_l^2)$ are random effects with grouping factor g_l and $\epsilon \sim N(0, \sigma_r^2)$ is the residual, the variance components are σ_l^2 for each random effect and σ_r^2 for the residual. The total variance is $\sigma^2_{\text{total}} = \sum_l \sigma_l^2 + \sigma_r^2$, and the proportion attributable to level l is $p_l = \sigma_l^2/\sigma^2_{\text{total}}$. The intra-class correlation for a single level is $\text{ICC}_l = \sigma_l^2/\sigma^2_{\text{total}}$.

The Wald CIs treat the standard deviation estimates $\hat{\sigma}_l = \sqrt{\text{diag}(\text{VarCorr}_l)}$ as approximately normal. The asymptotic variance of $\hat{\sigma}_l$ is approximated via the delta method on the variance estimate:

$$\text{SE}(\hat{\sigma}_l) \approx \frac{\hat{\sigma}_l}{\sqrt{2\,\nu_l}},$$

where ν_l is the degrees of freedom associated with component l (number of groups minus one for random effects, residual df for the residual). The CI on the SD is $\hat{\sigma}_l \pm z_{1-\alpha/2}\,\text{SE}(\hat{\sigma}_l)$, clipped to be non-negative, and the CI on the variance is the square of the SD CI bounds.

Limitations

- The LMM assumes normally distributed random effects and residuals. For count or proportion outcomes, use `lme4::glmer` with an appropriate family directly.
- Wald CIs on variance components are anti-conservative when the true variance is near zero (the boundary of the parameter space). For designs where this matters, use `confint(model, method = "profile")` directly on the stored `model`.
- The degrees-of-freedom approximation $\nu_l = n_l - 1$ is rough for unbalanced or crossed designs. Treat the CIs as approximate.
- The proportion of variance attributable to each level is meaningful only when the model is correctly specified. Including or excluding a fixed effect changes the variance decomposition.

References

- Bates, D., Mächler, M., Bolker, B. and Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. Journal of Statistical Software, 67, 1-48. doi:10.18637/jss.v067.i01.
- Searle, S. R., Casella, G. and McCulloch, C. E. (1992). Variance Components. Wiley. doi:10.1002/9780470316856.
- Pinheiro, J. C. and Bates, D. M. (2000). Mixed-Effects Models in S and S-PLUS. Springer New York. doi:10.1007/9781441903181.

Examples

```
set.seed(1)
# 4 bio samples, 3 tech replicates within each bio, 5 measurements per tech
df <- data.frame(
  y = rnorm(60, mean = rep(rnorm(12, sd = 1.5), each = 5), sd = 0.5),
  bio = factor(rep(1:4, each = 15)),
  tech = factor(rep(rep(1:3, each = 5), 4)))
```

```
fit <- replicate_decompose(y ~ 1 + (1|bio) + (1|bio:tech), data = df)
print(fit)
```

`robust_lod` *Limits of detection and quantification*

Description

Computes the limit of detection (LOD) and limit of quantification (LOQ) for a calibration curve, using one of three classical methods: the Currie (1968) critical-value approach, the simple blank-standard-deviation approach, or the Hubaux-Vos (1970) prediction-interval approach. For heteroscedastic calibration data the residual SD can be modelled as a function of concentration via lowess, in which case the LOD is computed at the local residual SD evaluated at the LOD itself.

Usage

```
robust_lod(
  calibration,
  response = NULL,
  conc = NULL,
  conf_level = 0.95,
  false_positive_rate = 0.05,
  false_negative_rate = 0.05,
  method = c("currie", "blank_sd", "hubaux_vos")
)
```

Arguments

`calibration`	an `lm` object (from `stats::lm(y ~ x)`) or a list with components `x` (concentration), `y` (response), and optionally `coef` (length-2: intercept, slope). If `coef` is missing it is fit by OLS.
`response`	optional; a numeric vector of blank-sample responses. Used to compute the blank SD for the Currie and blank-SD methods. If NULL, the calibration residual SD is used instead.
`conc`	optional; a numeric vector of concentrations corresponding to `response`. If supplied with `response`, used for heteroscedastic noise modelling.
`conf_level`	confidence level for the LOD CI.
`false_positive_rate`	target false-positive rate (alpha). Default 0.05.
`false_negative_rate`	target false-negative rate (beta). Default 0.05. The classical Currie 3*sigma LOD corresponds to alpha = beta ~ 0.00135 (one-sided z = 3). The default 0.05 gives a more conservative LOD.
`method`	character; `"currie"` (default), `"blank_sd"`, or `"hubaux_vos"`.

Details

The calibration can be supplied either as an `lm` object fit by `stats::lm(y ~ x)`, or as a list with components `x`, `y`, and (optionally) `coef`. If `response` and `conc` are supplied, they override the calibration curve as the source of the blank SD.

Value

A `crossstatr_lod` object with elements:

`lod`	limit of detection.
`loq`	limit of quantification.
`lod_ci`	Monte Carlo CI on the LOD (length-2).
`method`	method used.
`blank_sd`	blank SD used in the calculation.
`slope`, `intercept`	calibration slope and intercept.
`residual_se`	residual standard error of the calibration fit.
`df_resid`	residual degrees of freedom.
`false_positive_rate`, `false_negative_rate`	as input.
`call`	the matched call.

Methodology

The calibration is a linear model $y = \alpha + \beta x + \epsilon$ with $\epsilon \sim N(0, \sigma^2)$. The LOD is the lowest concentration that can be reliably distinguished from blank.

Currie (1968) critical value: the LOD is the concentration at which the probability of a false positive (signal exceeds blank threshold when true concentration is zero) and the probability of a false negative (signal below threshold when true concentration is LOD) are both controlled. The classical 3.3*sigma rule corresponds to $z_\alpha + z_\beta \approx 3.3$ (with $\alpha = \beta \approx 0.05$). The general form is

$$\mathrm{LOD} = \frac{(z_{1-\alpha} + z_{1-\beta})\,\sigma_b}{\beta},$$

where σ_b is the blank SD (or the calibration residual SD when no blanks are available) and β is the slope. The LOQ is conventionally

$$\mathrm{LOQ} = \frac{10\,\sigma_b}{\beta},$$

corresponding to a relative SD of 10% at the LOQ (Currie, 1968; IUPAC, 1995). The IUPAC simple form replaces $z_{1-\alpha} + z_{1-\beta}$ with the constant 3.

Blank-SD method: same form as Currie but using the empirical SD of the blank sample supplied via `response`, with no normal-theory calibration-based SE. The classical 3*sigma rule is used.

Hubaux-Vos (1970): the LOD is the concentration at which the lower one-sided prediction bound on the response equals the upper one-sided prediction bound at zero concentration. The closed-form fixed-point equation is

$$\mathrm{LOD} = \frac{(t_{1-\alpha,\,\nu} + t_{1-\beta,\,\nu})\,\sigma}{\beta}\sqrt{1 + \frac{1}{n} + \frac{\mathrm{LOD}^2}{S_{xx}}},$$

where σ is the calibration residual SE, ν is the residual df, n is the number of calibration points, and S_{xx} is the corrected sum of squares of the calibration concentrations. The equation is solved iteratively to convergence.

Heteroscedastic noise: if the residual SD varies appreciably with concentration (the ratio of high- to low-conc residual SD exceeds 2, assessed via lowess smoothing of $|r_i|$ on x_i), the LOD is recomputed using the local $\sigma(x)$ evaluated at the LOD estimate. This is iterated to convergence.

Monte Carlo CI on the LOD: simulate from the asymptotic distribution of $(\hat{\alpha}, \hat{\beta}, \hat{\sigma})$, recompute the LOD for each simulation, and take the percentile interval.

Limitations

- All three methods assume the calibration is linear in the relevant concentration range. If the calibration is nonlinear near zero, the LOD is unreliable.
- The Hubaux-Vos method assumes homoscedastic noise unless the heteroscedastic correction is invoked; even then the lowess fit is a smoother, not a model, and the resulting LOD is approximate.
- The Monte Carlo CI assumes the asymptotic normal distribution of the calibration parameters. For small calibration samples, this underestimates the true uncertainty.
- The 3*sigma convention is a rule of thumb, not a guarantee of any specific false-positive rate. Setting `false_positive_rate` and `false_negative_rate` explicitly is more honest.

References

- Currie, L. A. (1968). Limits for qualitative detection and quantitative determination. Application to radiochemistry. Analytical Chemistry, 40, 586-593. doi:10.1021/ac60259a007.
- Hubaux, A. and Vos, G. (1970). Decision and detection limits for calibration curves. Analytical Chemistry, 42, 849-855. doi:10.1021/ac60290a013.
- Currie, L. A. (1995). Nomenclature in evaluation of analytical methods including detection and quantification capabilities (IUPAC Recommendations 1995). Pure and Applied Chemistry, 67, 1699-1723. doi:10.1351/pac199567101699.

Examples

```
set.seed(1)
conc <- seq(0, 10, length.out = 20)
resp <- 0.5 + 2 * conc + rnorm(length(conc), sd = 0.3)
cal <- lm(resp ~ conc)
lod <- robust_lod(cal, method = "currie")
print(lod)

# With explicit blank measurements
blank_responses <- rnorm(10, mean = 0.5, sd = 0.3)
lod2 <- robust_lod(cal, response = blank_responses, method = "currie")
print(lod2)

# Hubaux-Vos prediction-interval approach
lod3 <- robust_lod(cal, method = "hubaux_vos")
print(lod3)
```

`sci_anova` *ANOVA and related analyses*

Description

Wraps and extends `stats::aov()` / `stats::lm()` to provide effect-size reporting, multiple-comparison adjustments, and Welch-type alternatives for heteroscedastic designs.

Usage

```
sci_anova(
  formula,
  data,
  welch = FALSE,
  posthoc = c("tukey", "bonferroni", "holm", "none"),
  conf_level = 0.95,
  ...
)
```

Arguments

`formula`	a model formula.
`data`	data frame.
`welch`	logical; use Welch's ANOVA (`oneway.test`) for one-way designs with unequal variances? Default FALSE.
`posthoc`	character; `"tukey"` (default), `"bonferroni"`, `"holm"`, or `"none"`.
`conf_level`	confidence level for CIs.
`...`	additional arguments passed to `aov` / `lm`.

Value

A `crossstatr_anova` object with `aov`, `anova_table`, `effect_sizes`, `posthoc` (if requested), and `diagnostics`.

Methodology

One-way ANOVA partitions the total sum of squares:

$$SS_{\text{total}} = SS_{\text{between}} + SS_{\text{within}},$$

with F-statistic

$$F = \frac{SS_{\text{between}}/(k-1)}{SS_{\text{within}}/(N-k)}.$$

Welch's ANOVA (Welch, 1951) replaces the within-group pooled variance with group-specific variances, and uses an Satterthwaite-adjusted denominator df.

Effect sizes:

- Eta-squared $\eta^2 = SS_{\text{between}} / SS_{\text{total}}$
- Partial eta-squared $\eta_p^2 = SS_{\text{effect}} / (SS_{\text{effect}} + SS_{\text{error}})$
- Omega-squared (see Carroll and Nordholm, 1975): $\omega^2 = \frac{SS_{\text{between}} - (k-1)MS_{\text{within}}}{SS_{\text{total}} + MS_{\text{within}}}$

Tukey HSD uses the studentized range distribution.

References

- Welch, B. L. (1951). On the comparison of several mean values: An alternative approach. Biometrika, 38, 330-336. doi:10.1093/biomet/38.34.330.
- Kelley, T. L. (1935). An Unbiased Correlation Ratio Measure. Proceedings of the National Academy of Sciences, 21, 554-559. doi:10.1073/pnas.21.9.554.
- Carroll, R. M. and Nordholm, L. A. (1975). Sampling characteristics of Kelley's epsilon-squared and Hays' omega-squared. Educational and Psychological Measurement, 35, 541-554. doi:10.1177/001316447503500304.
- Hsu, J. C. (1996). Multiple Comparisons: Theory and Methods. Chapman & Hall. doi:10.1201/b15074.

Examples

```
df <- data.frame(y = rnorm(60, mean = rep(c(5, 6, 8), each = 20)),
                 g = factor(rep(c("A","B","C"), each = 20)))
sci_anova(y ~ g, data = df)
```

`sci_bootstrap` *Bootstrap resampling with multiple CI types*

Description

Nonparametric, parametric, cluster, and stratified bootstrap. CIs are returned as percentile, normal-approximation, or BCa intervals. The resampling is implemented directly so that the user can pass any statistic function without depending on the `boot` package's calling conventions, although the package follows Efron & Tibshirani (1993) and Efron (1987) for the BCa formula.

Usage

```
sci_bootstrap(
  data,
  statistic,
  n_replicates = 1000L,
  type = c("nonparametric", "parametric", "cluster", "stratified"),
  cluster = NULL,
  strata = NULL,
  ci_type = c("perc", "bca", "normal"),
  conf_level = 0.95,
```

```
  seed = NULL,
  parallel = FALSE,
  parametric_sampler = NULL
)
```

Arguments

data	a numeric vector or data.frame. Passed to `statistic` after resampling.
statistic	a function that takes a resampled dataset of the same shape as `data` and returns a numeric scalar. Functions returning longer vectors are not supported; wrap them to extract a single statistic.
n_replicates	integer; number of bootstrap replicates (default 1000).
type	`"nonparametric"` (default), `"parametric"`, `"cluster"`, or `"stratified"`.
cluster	optional cluster IDs for `type = "cluster"`. Must be the same length as `data` (or `nrow(data)` if a data.frame).
strata	optional strata for `type = "stratified"`. Same length requirement as `cluster`.
ci_type	`"perc"` (default), `"bca"`, or `"normal"`.
conf_level	confidence level for the CI (default 0.95).
seed	optional integer seed for reproducibility.
parallel	logical; accepted but ignored. Included only so that the signature mirrors `boot::boot`.
parametric_sampler	optional function of a single argument `n` returning a synthetic dataset (vector or data.frame of `n` rows) for `type = "parametric"`. Required when `type = "parametric"`.

Details

The `parallel` argument exists for API symmetry with `boot::boot` but is a no-op here. The documentation does not recommend using it in this function; if the analyst needs parallelism wrap the call in `parallel::mclapply` over `n_replicates` batches yourself.

Value

A `crossstatr_bootstrap` object with elements:

statistic	the statistic computed on the original data.
replicates	numeric vector of bootstrap replicates.
ci	named numeric length-2 confidence interval.
n_replicates	number of replicates actually used.
type	resampling type.
ci_type	CI type.
call	the matched call.

Methodology

Nonparametric bootstrap: draw a resample of size `n` (with replacement) from the empirical distribution, recompute the statistic, repeat.

Parametric bootstrap: simulate a new dataset from a model fitted to the original data, supplied via `parametric_sampler`. The sampler is called `n_replicates` times.

Cluster bootstrap: resample whole clusters (with replacement) and stack the observations within the sampled clusters back into a dataset of size `n` (Davison and Hinkley, 1997).

Stratified bootstrap: resample (with replacement) independently within each stratum, preserving the per-stratum sample sizes.

Percentile CI: empirical $\alpha/2$ and $1-\alpha/2$ quantiles of the bootstrap distribution.

Normal CI: $\hat{\theta} \pm z_{1-\alpha/2}\,\hat{\sigma}_{boot}$, where $\hat{\sigma}_{boot}$ is the standard deviation of the replicates.

BCa (Efron, 1987): the interval is the percentile interval with adjusted quantiles. Let $\hat{\theta}$ be the original estimate and $\hat{\theta}^*_b$ the bootstrap replicates. The bias-correction is

$$z_0 = \Phi^{-1}\Big(\#\{\hat{\theta}^*_b < \hat{\theta}\}/B\Big).$$

The acceleration is obtained by jackknife:

$$a = \frac{\sum_i (\bar{\theta}_{(\cdot)} - \theta_{(i)})^3}{6\left[\sum_i (\theta_{(i)} - \bar{\theta}_{(\cdot)})^2\right]^{3/2}},$$

where $\theta_{(i)}$ is the leave-one-out statistic. The adjusted quantile levels are

$$\alpha_1 = \Phi\left(z_0 + \frac{z_0 + z_{\alpha/2}}{1 - a(z_0 + z_{\alpha/2})}\right), \qquad \alpha_2 = \Phi\left(z_0 + \frac{z_0 + z_{1-\alpha/2}}{1 - a(z_0 + z_{1-\alpha/2})}\right).$$

Limitations

- BCa is implemented for `type = "nonparametric"` only. For other types, the function falls back to the percentile interval and emits a warning. This is because the jackknife acceleration requires leave-one-out resampling of the original data, which is not well defined for the parametric or cluster cases without further specification.
- The statistic must return a finite numeric scalar. Statistics returning vectors, lists, or non-numeric values will error.
- For very small `n` the bootstrap distribution can be discrete and the percentile interval has poor coverage; consider `ci_type = "bca"` and inspecting the replicates directly.
- The `parallel` argument is a no-op. The package does not run any background processes.

References

- Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. The Annals of Statistics, 7, 1-26. doi:10.1214/aos/1176344552.
- Efron, B. (1987). Better Bootstrap Confidence Intervals. Journal of the American Statistical Association, 82, 171-185. doi:10.1080/01621459.1987.10478410.
- Efron, B. and Tibshirani, R. J. (1993). An Introduction to the Bootstrap. Chapman & Hall. doi:10.1007/9781489945419.
- Davison, A. C. and Hinkley, D. V. (1997). Bootstrap Methods and their Application. Cambridge University Press. doi:10.1017/CBO9780511802843.

Examples

```
set.seed(1)
x <- rnorm(30, mean = 5)
mean_stat <- function(d) mean(d)
result <- sci_bootstrap(x, mean_stat, n_replicates = 500, seed = 42)
print(result)

# data.frame input
df <- data.frame(y = rnorm(40), g = rep(c("A","B"), each = 20))
stat <- function(d) diff(tapply(d$y, d$g, mean))
sci_bootstrap(df, stat, type = "stratified", strata = df$g,
              ci_type = "bca", n_replicates = 300, seed = 7)
```

---

sci_compare *Group comparisons for experimental data*

Description

Provides the standard battery of two-group comparisons (Student's t, Welch's t, paired t, Wilcoxon rank-sum, Wilcoxon signed-rank) plus a permutation-based alternative for unequal-variance or non-Gaussian settings. Effect sizes (Cohen's d / Hedges' g, rank-biserial for the Wilcoxon tests) and confidence intervals are returned alongside the test statistic.

Usage

```
sci_compare(
  x,
  y = NULL,
  paired = FALSE,
  test = c("auto", "student", "welch", "perm", "wilcoxon"),
  alternative = c("two.sided", "less", "greater"),
  conf_level = 0.95,
  n_perm = 9999L,
  seed = NULL,
  data = NULL,
  ...
)
```

Arguments

x	numeric vector (first sample) or a formula referencing a two-column data.frame.
y	numeric vector (second sample) for two-sample tests; NULL for paired tests where y is implicit, or for formula interface.
paired	logical; use the paired test?

test	character; one of "auto" (Welch if n>=30 per group or balanced, otherwise Student or Wilcoxon — see Details), "student", "welch", "perm" (permutation), "wilcoxon".
alternative	"two.sided", "less", "greater".
conf_level	confidence level for the difference in means and for the effect size.
n_perm	number of permutations for test = "perm" (default 9999).
seed	optional integer seed for permutation reproducibility.
data	optional data frame when x is a formula.
...	additional arguments passed to underlying tests.

Details

Test selection is left to the analyst. This function does not pre-test for normality and pick a test on that basis; doing so inflates Type I error (Rochon et al., 2012).

Value

A crossstatr_compare object.

Methodology

Student's t-statistic (equal variance):

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{1/n_1 + 1/n_2}}, \qquad s_p^2 = \frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1 + n_2 - 2},$$

with $df = n_1 + n_2 - 2$.

Welch's t-statistic:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}}, \qquad df \approx \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{(s_1^2/n_1)^2/(n_1-1) + (s_2^2/n_2)^2/(n_2-1)}.$$

Paired t-statistic: applied to differences $d_i = x_i - y_i$, reducing to a one-sample t-test on d.

Permutation test: under the null hypothesis of exchangeability, group labels are randomly permuted n_perm times; the two-sided p-value is

$$p = \frac{1 + \sum_{b=1}^{B} \mathbf{1}\{|T_b| \geq |T_{\text{obs}}|\}}{B + 1}.$$

Wilcoxon rank-sum (Mann-Whitney U) and Wilcoxon signed-rank are computed via stats::wilcox.test with continuity correction.

Effect size Cohen's d / Hedges' g is computed as in sci_effect_size().

Limitations

- The permutation test is exact only when the exchangeability assumption holds (typically true for random assignment, not for observational data).
- Effect size CIs use the large-sample normal approximation and may be inaccurate for small samples or extreme effect sizes.
- Paired tests require length(x) == length(y).

References

- Welch, B. L. (1947). The generalization of Student's problem when several different population variances are involved. Biometrika, 34, 28-35. doi:10.1093/biomet/34.12.28.
- Rochon, J., Gondan, M. and Kieser, M. (2012). To test or not to test: Preliminary assessment of normality when comparing two independent samples. BMC Medical Research Methodology, 12, 81. doi:10.1186/147122881281.
- Davison, A. C. and Hinkley, D. V. (1997). Bootstrap Methods and their Application. Cambridge University Press. doi:10.1017/CBO9780511802843.

Examples

```
set.seed(1)
x <- rnorm(30, mean = 10)
y <- rnorm(30, mean = 11)
sci_compare(x, y, test = "welch")
```

`sci_correlation` *Correlation analysis with confidence intervals*

Description

Computes Pearson, Spearman, or Kendall correlations between two numeric vectors with confidence intervals derived from Fisher's variance-stabilising z-transform. When a grouping variable is supplied, the correlation is computed within each group and the resulting p-values are adjusted for multiplicity across groups. This is the natural use case when the analyst has repeated within-subject or within-condition measures and want to know whether the association generalises across strata, rather than trusting a single pooled estimate that may be driven by one subgroup.

Usage

```
sci_correlation(
  x,
  y,
  method = c("pearson", "spearman", "kendall"),
  alternative = c("two.sided", "less", "greater"),
  conf_level = 0.95,
  group = NULL,
  p_adjust = "holm"
)
```

Arguments

`x`	numeric vector.
`y`	numeric vector of the same length as `x`.
`method`	character; `"pearson"` (default), `"spearman"`, or `"kendall"`.

alternative	"two.sided" (default), "less", or "greater".
conf_level	confidence level for the interval on the correlation (default 0.95).
group	optional grouping variable (factor or coercible) of the same length as x. If supplied, per-group correlations are computed and the p-values in groupwise are adjusted using p_adjust.
p_adjust	character; p-value adjustment method for the groupwise p-values (default "holm"). Any method accepted by stats::p.adjust() is allowed.

Details

The package deliberately do not pre-test for normality and switch methods on that basis. The choice between Pearson (linear, sensitive to outliers, exact CI under bivariate normality) and Spearman/Kendall (rank-based, robust to monotone non-linearities and outliers) belongs to the analyst and should be motivated by the measurement scale and the underlying science.

Value

A crossstatr_correlation object with elements:

method	correlation method used.
correlation	numeric scalar point estimate (over all data).
p_value	p-value of the test of H0: \rho = 0.
ci	length-2 numeric confidence interval.
n	number of complete (finite) observations used.
alternative	alternative hypothesis.
conf_level	confidence level used.
groupwise	data frame of per-group estimates and adjusted p-values, or NULL if group was not supplied.
call	the matched call.

Methodology

For Pearson's r The package uses Fisher's variance-stabilising transform

$$z = \operatorname{atanh}(r) = \tfrac{1}{2} \ln \frac{1+r}{1-r},$$

which is approximately Gaussian with variance $1/(n-3)$ under the null of no correlation (Fisher, 1915; 1921). The confidence interval on z is

$$z \pm t_{1-\alpha/2,\, n-3} \, \frac{1}{\sqrt{n-3}},$$

which is then back-transformed to the correlation scale via $r = \tanh(z)$. The package uses a Student-t critical value rather than the Gaussian one; this gives a slightly wider interval for small n and is what stats::cor.test uses internally for Pearson intervals.

For Spearman's ρ, the same Fisher-z approximation is applied to the point estimate. This works well for moderate `n` (Bonett & Wright, 2000), although for very small samples or extreme correlations the coverage can be slightly off.

For Kendall's τ, `stats::cor.test` is called with `exact = FALSE` to avoid the combinatorial cost of the exact variance computation when `n` is large. The asymptotic variance used is the one for Kendall rank association. Confidence intervals for Kendall's tau are also constructed via the Fisher-z approximation on `atanh(tau)`.

When a `group` is supplied, p-values are adjusted across groups using `stats::p.adjust` with the requested method.

Limitations

- The Fisher-z interval for Spearman and Kendall is approximate and can be inaccurate for very small samples or when the correlation is close to ± 1. For `n < 10` consider exact permutation or bootstrap intervals instead.
- Kendall's tau with ties uses the normal approximation built into `cor.test`; the exact variance is not computed.
- Groupwise correlations discard groups with fewer than 3 finite observations. The reported adjusted p-values correspond only to the groups that were actually computed; do not interpret them as a global omnibus test.
- Pooling across groups via a single correlation ignores possible between-group heterogeneity. If the analyst supplies a `group`, read the `groupwise` table; the top-level `correlation` is the pooled estimate and may be misleading if the groups differ in location or scale.

References

- Fisher, R. A. (1915). Frequency Distribution of the Values of the Correlation Coefficient in Samples from an Indefinitely Large Population. Biometrika, 10, 507-521. doi:10.2307/2331838.
- Fisher, R. A. (1921). On the probable error of a coefficient of correlation deduced from a small sample. Metron, 1, 3-32. https://digital.library.adelaide.edu.au/items/002ad8fb-c23c-407b-8a89-f036d8da6030.
- Bonett, D. G. and Wright, T. A. (2000). Sample Size Requirements for Estimating Pearson, Kendall and Spearman Correlations. Psychometrika, 65, 23-28. doi:10.1007/BF02294183.

Examples

```
set.seed(1)
x <- rnorm(40)
y <- 0.6 * x + rnorm(40, sd = 0.8)
result <- sci_correlation(x, y, method = "pearson")
print(result)

# Groupwise
g <- rep(c("A", "B"), each = 20)
x2 <- c(rnorm(20), rnorm(20) + 1)
y2 <- c(rnorm(20), 0.8 * rnorm(20) + 1)
sci_correlation(x2, y2, group = g, method = "spearman")
```

`sci_describe` *Descriptive statistics for scientific experiments*

Description

Computes a comprehensive set of descriptive statistics tailored to experimental data, with explicit support for technical and biological replicates. Technical replicates are *not* treated as independent biological observations: when a `bio_rep` grouping is supplied, technical replicates are summarized to a per-biological-unit value before the final statistics are computed.

Usage

```
sci_describe(
  x,
  group = NULL,
  bio_rep = NULL,
  tech_rep = NULL,
  units = "",
  conf_level = 0.95,
  na.rm = TRUE,
  quantile_probs = c(0, 0.25, 0.5, 0.75, 1)
)
```

Arguments

`x`	numeric vector of observations.
`group`	optional factor or coercible vector giving experimental groups (e.g. treatment levels). If supplied, statistics are computed per group.
`bio_rep`	optional vector identifying biological replicates. When supplied together with `tech_rep`, the function first averages technical replicates within each `bio_rep` x `group` cell, then summarises biological replicates. This prevents pseudo-replication.
`tech_rep`	optional vector identifying technical replicates.
`units`	character scalar; measurement units for `x`. Stored on the result for downstream labelling; not used in computation.
`conf_level`	confidence level for the mean and median intervals (default 0.95).
`na.rm`	logical; remove non-finite observations before computation?
`quantile_probs`	numeric vector of probabilities for quantile reporting (default `c(0, 0.25, 0.5, 0.75, 1)`).

Value

A `crossstatr_describe` object with elements:

`summary`	data frame of per-group (or overall) statistics.

`n_obs`, `n_bio_rep`, `n_tech_rep`
: observation and replicate counts.

`units`
: passed-through measurement units.

`method`
: method label.

`call`
: the matched call.

Methodology

The mean and variance use the standard unbiased estimators with Bessel's correction:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2.$$

The standard error is $s/\sqrt{n}$. The confidence interval for the mean uses a Student-t critical value with `n - 1` df:

$$\bar{x} \pm t_{1-\alpha/2,n-1}\, s/\sqrt{n}.$$

The median interval uses order-statistic endpoints with a normal approximation to binomial bounds:

$$\left[x_{(k)}, x_{(n-k+1)}\right], \qquad k = \frac{n-1}{2} - z_{1-\alpha/2}\,\frac{\sqrt{n}}{2}.$$

The coefficient of variation is $s/\bar{x}$ (reported as a proportion, not a percentage). The MAD uses the standard consistency constant 1.4826 so that it estimates the same scale parameter as `sd` for Gaussian data.

When `bio_rep` is supplied, per-biological-replicate means are computed first by averaging technical replicates; this is the recommended workflow for experimental data and matches the analysis of Vaux et al. (2012, *EMBO reports*) on avoiding pseudo-replication. The reported `n` then equals the number of biological replicates, not the total number of measurements.

Limitations

- Confidence intervals assume approximate independence of the biological replicates. If the biological units are themselves clustered (e.g. mice within a cage), use `replicate_decompose()` instead.
- The median CI requires at least 5 observations; with fewer, it is reported as `NA`.
- The CV is meaningless for signed data near zero; the function still returns it but emits a warning when the mean changes sign within a group.

References

- Vaux, D. L., Fidler, F. and Cumming, G. (2012). Replicates and repeats—what is the difference and is it significant? The EMBO Reports, 13, 291-296. doi:10.1038/embor.2012.36.
- Hettmansperger, T. P. and McKean, J. W. (2010). Robust Nonparametric Statistical Methods. 2nd ed. CRC Press. doi:10.1201/b10451.

Examples

```
set.seed(1)
x <- c(rnorm(20, mean = 10), rnorm(20, mean = 12))
g <- rep(c("A", "B"), each = 20)
sci_describe(x, group = g)

# With biological/technical replicates
df <- data.frame(
  y = rnorm(60, mean = rep(c(8, 10, 12), each = 20)),
  grp = rep(c("A","B","C"), each = 20),
  bio = rep(1:6, each = 10),
  tech = rep(1:10, times = 6))
sci_describe(df$y, group = df$grp, bio_rep = df$bio, tech_rep = df$tech)
```

---

sci_effect_size *Standardized effect sizes with confidence intervals*

Description

Computes Cohen's d, Hedges' g, paired-d (Cohen's d_z), Pearson r as an effect size, and partial eta-squared, each with a confidence interval and an interpretation against Cohen's (1988) conventions. The function is a single entry point so that downstream code can request an effect size without having to remember which helper computes which.

Usage

```
sci_effect_size(
  x,
  y = NULL,
  paired = FALSE,
  type = c("auto", "cohens_d", "hedges_g", "cohens_d_z", "paired_d", "r", "eta2_partial"),
  conf_level = 0.95,
  n1 = NULL,
  n2 = NULL,
  ss_effect = NULL,
  ss_error = NULL
)
```

Arguments

x	numeric vector (first sample).
y	numeric vector (second sample), or NULL for one-sample d.
paired	logical; compute a paired effect size? Required for type = "cohens_d_z" and type = "paired_d".

type	character; "auto" (default), "cohens_d", "hedges_g", "cohens_d_z", "paired_d", "r", or "eta2_partial". "auto" dispatches as follows: if ss_effect is supplied, partial eta-squared; if y is NULL, one-sample Cohen's d; if paired is TRUE, paired d (d_z); otherwise Cohen's d for two independent samples.
conf_level	confidence level (default 0.95).
n1, n2	optional explicit sample sizes used to compute the SE. If NULL, the lengths of x and y are used.
ss_effect, ss_error	sums of squares for the effect and the error term. Required when type = "eta2_partial".

Details

The documentation does not recommend Cohen's benchmarks as anything more than a rough sanity check. They were derived for the behavioral sciences and may be inappropriate in domains where the meaningful size of an effect depends on measurement scale or cost (e.g. electrochemical measurements, where a "small" shift in half-wave potential can be diagnostic). The interpretation field is included as a convenience, not as a substitute for domain judgement.

Value

A crossstatr_effect_size object with elements:

type	effect size label.
estimate	numeric point estimate.
se	standard error (delta method for d-family; Fisher-z for r).
ci_low, ci_high	CI endpoints.
interpretation	text label: negligible / small / medium / large.
method	short method label.
call	the matched call.

Methodology

Cohen's d (independent samples, pooled SD):

$$d = \frac{\bar{x}_1 - \bar{x}_2}{s_p}, \qquad s_p^2 = \frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}.$$

Hedges' g applies the small-sample bias correction

$$J = 1 - \frac{3}{4(n_1+n_2-2)-1}, \qquad g = J \cdot d.$$

Paired d (d_z) is the mean of the paired differences divided by the SD of the differences:

$$d_z = \frac{\bar{d}}{s_d}, \qquad d_i = x_i - y_i.$$

This is larger in magnitude than the corresponding independent-samples d when the paired correlation is positive; do not confuse the two (Cohen, 1988; Morris & DeShon, 2002).

The standard error for the d-family uses the delta-method formula of Hedges & Olkin (1985, eq. 8.21):

$$\mathrm{SE}(d)^2 \approx \frac{n_1 + n_2}{n_1 n_2} + \frac{d^2}{2(n_1 + n_2)}$$

for independent samples (with a small-sample correction). The CI is then $d \pm z_{1-\alpha/2}\,\mathrm{SE}(d)$.

Pearson r as effect size: $r = \mathrm{cor}(x, y)$, with the CI from the Fisher-z transform, as in `sci_correlation()`.

Partial eta-squared:

$$\eta_p^2 = \frac{SS_{\text{effect}}}{SS_{\text{effect}} + SS_{\text{error}}}.$$

A CI based on the noncentral F distribution requires the F statistic and its degrees of freedom. The package does not estimate that here; the package reports a point estimate and a caveat in `interpretation`.

Cohen's (1988) conventions:

- d: small 0.2, medium 0.5, large 0.8
- r: small 0.1, medium 0.3, large 0.5
- eta²: small 0.01, medium 0.06, large 0.14

Limitations

- The d-family SE and CI rely on a large-sample normal approximation and can have poor coverage for `n < 20` per group or for very large effects. For small samples, use bootstrap (e.g. `sci_bootstrap()`).
- Partial eta-squared is returned without a CI; do not interpret the point estimate as if it had one.
- Hedges' g and Cohen's d are computed under an equal-variance assumption. For unequal variances consider Glass's Δ (not implemented here) or a Welch-style standardizer.
- The auto dispatch is intentionally conservative; if the analyst wants a specific effect size, name it explicitly via `type`.

References

- Cohen, J. (1988). Statistical Power Analysis for the Behavioral Sciences. 2nd ed. Lawrence Erlbaum Associates. doi:10.4324/9780203771587.
- Hedges, L. V. and Olkin, I. (1985). Statistical Methods for Meta- Analysis. Academic Press. doi:10.1016/C20090033960.
- Morris, S. B. and DeShon, R. P. (2002). Combining effect size estimates in meta-analysis with repeated measures and independent- groups designs. Psychological Methods, 7, 105-125. doi:10.1037/1082989X.7.1.105.

Examples

```
set.seed(1)
x <- rnorm(30, mean = 1)
y <- rnorm(30, mean = 0)
sci_effect_size(x, y, type = "cohens_d")
```

```
# Paired
sci_effect_size(x, y, paired = TRUE, type = "paired_d")

# r
sci_effect_size(x, y, type = "r")

# Partial eta-squared
sci_effect_size(ss_effect = 12.5, ss_error = 80, type = "eta2_partial")
```

`sci_normality` *Normality assessment for experimental data*

Description

Provides a multi-pronged assessment of distributional assumptions. **A non-significant normality test is not evidence of normality**: with small samples the test has very little power to detect deviations, and with large samples it will reject for practically irrelevant deviations. Both regimes are reported explicitly.

Usage

```
sci_normality(
  x,
  group = NULL,
  test = c("shapiro", "ad", "auto"),
  alpha = 0.05,
  min_n = 3L,
  plot = FALSE
)
```

Arguments

`x`	numeric vector.
`group`	optional grouping vector; assessments are done per group.
`test`	character; one of `"shapiro"` (Shapiro-Wilk, default for n <= 5000), `"ad"` (Anderson-Darling, requires the `nortest` package at runtime via `requireNamespace`), or `"auto"` (picks Shapiro-Wilk for n <= 5000, otherwise suggests Anderson-Darling).
`alpha`	significance threshold for flagging non-normality (default 0.05). Used only for the qualitative verdict.
`min_n`	minimum sample size below which normality testing is not attempted (default 3 for Shapiro-Wilk).
`plot`	logical; produce a Q-Q plot when `ggplot2` is available?

Value

A `crossstatr_normality` object with per-group test results, sample sizes, distributional moments, skewness and kurtosis, and an explicit qualitative verdict that distinguishes "not rejected" from "evidence of normality".

Methodology

The Shapiro-Wilk statistic is

$$W = \frac{\left(\sum_i a_i x_{(i)}\right)^2}{\sum_i (x_i - \bar{x})^2},$$

where `a_i` are tabulated coefficients that depend only on `n` (Royston, 1992). It is the most powerful general-purpose test for normality for `n` up to a few thousand.

Anderson-Darling is more sensitive in the tails. Its statistic is

$$A^2 = -n - \frac{1}{n}\sum_{i=1}^{n}(2i-1)\left[\ln z_{(i)} + \ln(1 - z_{(n+1-i)})\right],$$

with $z_{(i)} = \Phi((x_{(i)} - \bar{x})/s)$.

Skewness uses the bias-corrected Fisher-Pearson sample estimator (Doane & Seward, 2011):

$$g_1 = \frac{\sqrt{n(n-1)}}{n-2}\, g_1^*,$$

with g_1^* the biased sample skewness.

Limitations

- Normality testing is most useful as a *diagnostic*, not a decision rule. Pre-test selection of `t` vs. Wilcoxon on the basis of a Shapiro-Wilk p-value inflates Type I error of the downstream test (Rochon et al., 2012). Prefer robust methods or specify the test a priori.
- For very large `n` (>5000), Shapiro-Wilk will reject almost any real dataset; use the Anderson-Darling test only as a sanity check and rely on residual diagnostics instead.

References

- Royston, P. (1992). Approximating the Shapiro-Wilk W-test for non- normality. Statistics and Computing, 2, 117-119. doi:10.1007/BF01891203.
- Stephens, M. A. (1974). EDF Statistics for Goodness of Fit and Some Comparisons. Journal of the American Statistical Association, 69, 730-737. doi:10.1080/01621459.1974.10480196.
- Doane, D. P. and Seward, L. E. (2011). Measuring Skewness: A Forgotten Statistic? Journal of Statistics Education, 19. doi:10.1080/10691898.2011.11889611.
- Rochon, J., Gondan, M. and Kieser, M. (2012). To test or not to test: Preliminary assessment of normality when comparing two independent samples. BMC Medical Research Methodology, 12, 81. doi:10.1186/147122881281.

Examples

```
set.seed(1)
x <- rnorm(50)
sci_normality(x)
```

`sci_outliers` *Outlier diagnostics for experimental data*

Description

Flags observations that are statistically extreme relative to a distributional model. **Flagging does not remove observations.** The function returns a per-observation diagnostic table; removal is left to the analyst, who should record any decision with justification.

Usage

```
sci_outliers(
  x,
  group = NULL,
  method = c("iqr", "mad", "zscore", "robust_z"),
  iqr_k = 1.5,
  z_threshold = 3,
  robust_z_threshold = 3.5
)
```

Arguments

`x`	numeric vector.
`group`	optional grouping vector.
`method`	character vector; which diagnostics to compute. Subset of `c("iqr", "mad", "zscore", "robust_z")`.
`iqr_k`	IQR multiplier (default 1.5; use 3.0 for "extreme" only).
`z_threshold`	absolute z-score threshold (default 3.0).
`robust_z_threshold`	absolute MAD-based z-score threshold (default 3.5, Iglewicz & Hoaglin, 1993).

Value

A `crossstatr_outliers` object with elements:

`flags`	data frame with one row per observation.
`counts`	per-method flag counts per group.
`method`	method label.

Methodology

IQR rule (Tukey, 1977):

$$[Q_1 - k(Q_3 - Q_1),\ Q_3 + k(Q_3 - Q_1)]$$

MAD rule (Hampel, 1974):

$$[m - k\,\mathrm{MAD},\ m + k\,\mathrm{MAD}],$$

with `MAD` rescaled by 1.4826 to match the Gaussian scale. The package uses `k = 3.5` by default.

Standardized z-score:

$$z_i = (x_i - \bar{x})/s,$$

flagged when $|z_i| > z_{\text{threshold}}$.

Robust z-score (Iglewicz & Hoaglin, 1993):

$$z_i^R = 0.6745\,(x_i - m)/\mathrm{MAD},$$

flagged when $|z_i^R| > 3.5$.

Limitations

- For very small samples (n < 5) the IQR/MAD rules have low power and the z-score rule is unstable. Results are returned with a `n < 5` warning.
- Groupwise diagnostics assume the group itself is homogeneous; if within-group structure exists (e.g. time trends), use model-based influence diagnostics from `sci_regression()` instead.
- Multiple-test inflation is *not* corrected here because the diagnostics are exploratory.

References

- Tukey, J. W. (1977). Exploratory Data Analysis. Addison-Wesley. ISBN 9780201076165. https://www.pearson.com/en-us/subject-catalog/p/exploratory-data-analysis-classic-version-/P200000006400.
- Hampel, F. R. (1974). The Influence Curve and its Role in Robust Estimation. Journal of the American Statistical Association, 69, 383-393. doi:10.1080/01621459.1974.10482962.
- Iglewicz, B. and Hoaglin, D. C. (1993). How to Detect and Handle Outliers. ASQC Quality Press. https://asq.org/quality-press/display-item?item=E0801.

Examples

```
set.seed(1)
x <- c(rnorm(40), 8, 12)
sci_outliers(x)
```

`sci_power` *Power and sample size for common designs*

Description

Analytic power (or sample size) for the workhorse inferential designs used in experimental work: two-sample t, paired t, correlation, one-way ANOVA, and multiple regression (overall F test). When `n_sim > 0`, the analytic power is cross-checked (or replaced) by a simulation from the assumed generating model. The documentation recommends reporting the simulation whenever the effect size is small or the design has unequal group sizes, where the analytic formulae are known to be slightly anti-conservative.

Usage

```
sci_power(
 design = c("two_sample_t", "paired_t", "correlation", "anova_oneway", "regression"),
  effect_size,
  n = NULL,
  n1 = NULL,
  n2 = NULL,
  alpha = 0.05,
  power = NULL,
  alternative = c("two.sided", "one.sided"),
  n_sim = 0L,
  seed = NULL,
  k = 3L,
  predictors = 1L
)
```

Arguments

`design`	character; one of `"two_sample_t"`, `"paired_t"`, `"correlation"`, `"anova_oneway"`, `"regression"`.
`effect_size`	numeric; Cohen's d for t-tests, the Pearson r for correlation, Cohen's f for one-way ANOVA, or Cohen's f^2 for regression.
`n`	integer; sample size. For `two_sample_t` this is the per-group size when `n1`/`n2` are not given. For `anova_oneway` it is the per-group size. For `paired_t`, `correlation`, and `regression` it is the total sample size. If `NULL` and `power` is given, the function solves for `n`.
`n1, n2`	optional per-group sizes for `two_sample_t` (unequal-n case). If supplied, they override `n`.
`alpha`	type-I error rate (default 0.05).
`power`	target power (for sample-size determination). If `NULL`, power is computed from `n`.
`alternative`	`"two.sided"` (default) or `"one.sided"`. F-tests in ANOVA/regression are inherently two-sided; this argument is silently ignored for those designs.

n_sim	integer; if greater than 0, run n_sim simulations and return the empirical rejection rate in sim_results.
seed	optional integer seed for reproducibility of the simulation.
k	integer; number of groups for anova_oneway (default 3).
predictors	integer; number of predictors for regression (default 1).

Value

A crossstatr_power object with elements:

design	design label.
effect_size	effect size supplied.
n	the sample size (per-group for two_sample_t/anova_oneway, total otherwise). For two-sample t with unequal n, this is n1+n2.
power	computed or target power.
alpha	type-I error used.
method	short method label.
sim_results	NULL or a length-n_sim logical vector of rejection indicators.
n_sim	number of simulations run (0 if none).
call	the matched call.

Methodology

Two-sample t (equal variance, equal n):

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{2/n}}, \qquad \delta = d\sqrt{n/2}.$$

Power is $1 - F_{t_{2n-2,\delta}}(t_{1-\alpha/2,\,2n-2}) + F_{t_{2n-2,\delta}}(-t_{1-\alpha/2,\,2n-2})$ for two-sided alternatives. The package delegates to stats::power.t.test when n is equal across groups and compute directly otherwise.

Paired t: same machinery with one sample of differences of size n and noncentrality $\delta = d_z\sqrt{n}$.

Correlation (Fisher z): under the alternative $\rho \neq 0$,

$$z_r = \operatorname{atanh}(r), \qquad \lambda = z_r\sqrt{n-3},$$

and the two-sided power is

$$1 - \Phi(z_{1-\alpha/2} - \lambda) + \Phi(-z_{1-\alpha/2} - \lambda).$$

For one-sided tests the package uses $z_{1-\alpha}$ and the appropriate tail.

One-way ANOVA (k groups, n per group, total N = n*k): Cohen's $f^2 = \sigma_m^2/\sigma^2$ where σ_m^2 is the variance of the group means. The noncentrality parameter is

$$\lambda = Nf^2 = nkf^2,$$

with $df_1 = k - 1$, $df_2 = N - k$, and

$$\text{power} = 1 - F_{df_1, df_2, \lambda}(F_{1-\alpha, df_1, df_2}).$$

Regression (overall F test): with `p` predictors and total `n`, $f^2 = R^2/(1 - R^2)$, noncentrality $\lambda = nf^2$, $df_1 = p$, $df_2 = n - p - 1$, and the power is the upper tail of the noncentral F at the critical value.

Simulation: for `n_sim > 0`, the assumed model is simulated from a Gaussian with the supplied effect size (Cohen's d for t-tests, Pearson r for correlation, Cohen's f for ANOVA, Cohen's f² for regression), the test is run, and the proportion of rejections is reported as the empirical power. A finite seed makes the simulation reproducible.

Limitations

- Analytic formulae assume Gaussian errors, equal variances (for two-sample t and ANOVA), and the correctness of the supplied effect size. None of these are guaranteed in practice; the documentation recommends a sensitivity analysis with `n_sim > 0`.
- The correlation test formula uses the Fisher-z approximation, which is mildly biased for small `n` (Bonett & Wright, 2000). For `n < 25`, consider exact permutation power.
- Sample-size determination uses `uniroot` with a bracket of `[4, 1e6]`; if the target power cannot be reached within that range (e.g. effect size exactly zero), the function errors.
- Simulation for `anova_oneway` distributes group means evenly; if the real pattern of means is uneven, the empirical power will differ.

References

- Cohen, J. (1988). Statistical Power Analysis for the Behavioral Sciences. 2nd ed. Lawrence Erlbaum Associates. doi:10.4324/9780203771587.
- Fisher, R. A. (1921). On the probable error of a coefficient of correlation deduced from a small sample. Metron, 1, 3-32. https://digital.library.adelaide.edu.au/items/002ad8fb-c23c-407b-8a89-f036d8da6030.
- Bonett, D. G. and Wright, T. A. (2000). Sample Size Requirements for Estimating Pearson, Kendall and Spearman Correlations. Psychometrika, 65, 23-28. doi:10.1007/BF02294183.

Examples

```
# Power for two-sample t with d = 0.5, n = 30 per group
sci_power("two_sample_t", effect_size = 0.5, n = 30)

# Sample size for correlation r = 0.3, target power = 0.8
sci_power("correlation", effect_size = 0.3, power = 0.8)

# ANOVA power with Cohen's f = 0.25, 3 groups, 20 per group
sci_power("anova_oneway", effect_size = 0.25, n = 20, k = 3)

# Cross-check with simulation

sci_power("correlation", effect_size = 0.3, n = 100, n_sim = 50, seed = 1)
```

sci_regression *Regression for scientific experiments*

Description

Provides OLS, weighted least squares (WLS) with user-supplied measurement uncertainty, generalized least squares via `nlme::gls` (if available) for correlated residuals, nonlinear least squares via `stats::nls()`, and robust regression via `MASS::rlm()`.

Usage

```
sci_regression(
  formula,
  data,
  weights = NULL,
  weight_type = c("precision", "analytic"),
  robust = FALSE,
  nonlinear = FALSE,
  start = NULL,
  conf_level = 0.95,
  predict_at = NULL,
  ...
)
```

Arguments

`formula`	a model formula.
`data`	data frame.
`weights`	optional numeric weights. Interpretation depends on `weight_type`: `"precision"` treats weights as `1/sigma^2` (measurement uncertainties in y), `"analytic"` treats them as analytic GLS weights (default if non-NULL).
`weight_type`	character; one of `"precision"` or `"analytic"`.
`robust`	logical; use `MASS::rlm()` (Huber, default) instead of OLS?
`nonlinear`	logical; if TRUE, treat `formula` as nonlinear and use `nls()`; in that case `start` is required.
`start`	starting values for `nls()`.
`conf_level`	confidence level for CIs.
`predict_at`	optional data frame for which to produce predictions with CIs.
`...`	additional arguments to underlying fitting function.

Value

A `crossstatr_regression` object with the fitted model, coefficients with CIs, residual diagnostics, and (optionally) predictions.

Methodology

Ordinary least squares minimizes

$$S(\beta) = \sum_i (y_i - x_i^\top \beta)^2,$$

with closed-form solution $\hat{\beta} = (X^\top X)^{-1} X^\top y$.

Weighted least squares with precision weights $w_i = 1/\sigma_i^2$:

$$\hat{\beta} = (X^\top W X)^{-1} X^\top W y,$$

where $W = \mathrm{diag}(w_i)$. This is the BLUE under the heteroscedastic model $y_i \sim N(x_i^\top \beta, \sigma_i^2)$.

The coefficient covariance is

$$\widehat{\mathrm{Cov}}(\hat{\beta}) = \hat{\sigma}^2 (X^\top W X)^{-1},$$

with $\hat{\sigma}^2 = \frac{1}{n-p} \sum_i w_i (y_i - x_i^\top \hat{\beta})^2$.

Robust regression via `MASS::rlm` uses an M-estimator with Huber psi-function by default. The asymptotic covariance is the standard sandwich estimator; `MASS::rlm` returns a model-based covariance.

Prediction interval for a new observation at x_0:

$$\hat{y}_0 \pm t_{1-\alpha/2,\, n-p}\, \hat{\sigma} \sqrt{1 + x_0^\top (X^\top W X)^{-1} x_0}.$$

Limitations

- OLS standard errors are wrong under heteroscedasticity unless weights are supplied. Always supply `weights` when measurement uncertainties are known.
- For correlated residuals (e.g. time series), use `nlme::gls` directly with an appropriate `corStruct`; this function does not fit a correlation structure automatically.
- Nonlinear fits require good starting values; `nls()` may fail to converge if `start` is poor. Check the convergence message.

References

- Huber, P. J. (1981). Robust Statistics. Wiley. doi:10.1002/0471725250.
- Bates, D. M. and Watts, D. G. (1988). Nonlinear Regression Analysis and Its Applications. Wiley. doi:10.1002/9780470316757.
- Draper, N. R. and Smith, H. (1998). Applied Regression Analysis. 3rd ed. Wiley. doi:10.1002/9781118625590.

Examples

```
df <- data.frame(x = 1:20, y = 2 * (1:20) + rnorm(20, sd = 2))
fit <- sci_regression(y ~ x, data = df)
print(fit)

# Weighted regression with measurement uncertainties
df$sigma <- runif(20, 0.5, 2)
```

```
fit2 <- sci_regression(y ~ x, data = df, weights = 1 / df$sigma^2,
                       weight_type = "precision")
```

`signal_difference_regions` *Statistical identification of regions where experimental curves differ*

Description

Constructs simultaneous confidence bands for the difference between two (or two groups of) experimental curves, and identifies contiguous regions where the bands exclude zero. The method handles correlated measurements, unequal sampling grids, and replicate variability. It is genuinely useful for spectroscopic and electrochemical datasets where naive pointwise testing inflates the family-wise error rate.

Usage

```
signal_difference_regions(
  x,
  y1,
  y2,
  x1 = NULL,
  x2 = NULL,
  x_grid = NULL,
  conf_level = 0.95,
  method = c("max-t", "boot"),
  n_perm = 0L,
  seed = NULL,
  min_region_width = 0
)
```

Arguments

`x`	numeric vector of x-values.
`y1`	numeric matrix or data frame of curves in group 1. Each row is a replicate curve observed at `x`. If curves are observed on different grids, supply `x1`, `x2` instead.
`y2`	numeric matrix or data frame of curves in group 2.
`x1`, `x2`	optional x-vectors for each group; if supplied, `x` is ignored and both groups are interpolated onto `x_grid` (or a union of the two grids).
`x_grid`	optional target grid; if NULL, the union of `x1` and `x2` (or `x`) is used, restricted to the common range.
`conf_level`	nominal coverage of the simultaneous band (default 0.95).
`method`	`"max-t"` (default) for the max-t simultaneous band (max-statistic resampling; see Westfall and Young, 1993, when `n_perm > 0`). The accepted `"boot"` value currently follows the same computation; a separate bootstrap band is not implemented.

`n_perm` number of permutations for the Westfall-Young adjustment. 0 disables permutation adjustment (faster, more conservative).

`seed` optional integer seed.

`min_region_width` minimum width (in x units) for a region to be reported. Shorter regions are merged into neighbours.

Value

A `crossstatr_sdr` object with:

`x_grid` common grid.

`diff` pointwise mean difference.

`se` pointwise standard error.

`band_low`, `band_high` lower/upper simultaneous band.

`regions` data frame of contiguous significant regions (start, end, max_abs_diff, sign).

`n_perm` number of permutations used.

`method` method label.

Methodology

Let $\hat{d}(x) = \bar{y}_1(x) - \bar{y}_2(x)$ be the pointwise mean difference at grid point x, and let $s(x) = \sqrt{s_1^2(x)/n_1 + s_2^2(x)/n_2}$ be its standard error. The pointwise t-statistic is

$$T(x) = \hat{d}(x)/s(x).$$

A naive pointwise 95% interval $\hat{d}(x) \pm t_{0.975}\, s(x)$ has family-wise coverage far below 95% when the family is the whole grid. The max-t band instead uses the critical value

$$c_\alpha = Q_{1-\alpha}\left[\max_x |T(x)|\right],$$

where the quantile is taken over the distribution of the maximum absolute t-statistic under the null. The simultaneous band is then $\hat{d}(x) \pm c_\alpha\, s(x)$.

The distribution of the max-t statistic is estimated by a Westfall-Young permutation scheme: under the null hypothesis of no group difference, group labels are exchanged between replicate curves; the max-t statistic is recomputed for each of `n_perm` permutations; the empirical $1 - \alpha$ quantile of these permuted maxima is c_α.

When `n_perm = 0`, the Bonferroni-adjusted critical value is used: $c_\alpha = t_{1-\alpha/(2m),\nu}$ with m the number of grid points. This is conservative but does not require permutation.

Contiguous significant regions are extracted as runs of grid points where the band excludes zero. Region boundaries are reported with explicit x-values; the width uncertainty of each boundary is one grid step.

Limitations

- Assumes replicate curves are independent. For paired designs, permute within pairs (not implemented; use paired-test infrastructure instead).
- The permutation exchangeability assumption is violated when groups have different noise structures. In that case, use `n_perm = 0` (Bonferroni) instead.
- Region boundaries are limited by the grid resolution; sub-grid localization requires interpolation and is not done here.
- For very large grids (>1000 points) the Bonferroni adjustment becomes extremely conservative; permutation is recommended.

References

- Westfall, P. H. and Young, S. S. (1993). Resampling-Based Multiple Testing: Examples and Methods for p-Value Adjustment. Wiley. https://www.wiley.com/en-gb/Resampling-Based+Multiple+Testing%3A+Examples+and+Methods+for+p-Value+Adjustment-p-9780471557616.

Examples

```
set.seed(1)
x <- seq(400, 800, length.out = 80)
# Two groups of curves with a true difference in 500-560 region
make_curve <- function(mu, region) {
  base <- sin((x - 400) * pi / 200) * 0.2 + mu
  if (region) base <- base + dnorm(x, mean = 530, sd = 18) * 0.8
  base + rnorm(length(x), sd = 0.15)
}
y1 <- matrix(replicate(15, make_curve(1, FALSE)), nrow = 15, byrow = TRUE)
y2 <- matrix(replicate(15, make_curve(1, TRUE)),  nrow = 15, byrow = TRUE)
res <- signal_difference_regions(x, y1, y2, n_perm = 200)
print(res)
```

---

spectral_change_test *Test for differences across complete spectra*

Description

Tests whether complete spectra (rows of a matrix) differ across groups. Three complementary methods are offered: a Hotelling T-squared test for two groups (multivariate analogue of the two-sample t-test), a PERMANOVA F-test based on a Euclidean distance matrix with permutation-derived p-values, and a battery of pointwise t-tests with multiplicity adjustment.

Usage

```
spectral_change_test(
  spectra,
  group,
  wavelengths = NULL,
  method = c("t2", "permanova", "pointwise_t"),
  conf_level = 0.95,
  n_perm = 999L,
  seed = NULL,
  adjust = "fdr"
)
```

Arguments

spectra	numeric matrix or data frame; rows are observations, columns are wavelengths (or m/z channels, time bins, etc.).
group	factor or coercible vector of group memberships, length equal to nrow(spectra). Must have at least 2 levels.
wavelengths	optional numeric vector of wavelength values, length equal to ncol(spectra). If NULL, column indices are used.
method	"t2", "permanova", or "pointwise_t".
conf_level	confidence level (used only for some auxiliary intervals).
n_perm	integer; number of permutations for PERMANOVA.
seed	optional integer seed for permutation reproducibility.
adjust	p-value adjustment method for the pointwise tests; any method accepted by stats::p.adjust().

Details

The Hotelling test is the right tool when the dimensionality is small relative to the sample size and the analyst wantss an omnibus statement. PERMANOVA is preferred when the dimensionality exceeds the sample size or when the test must generalise to more than two groups. Pointwise tests answer a different question (where do the spectra differ?) and should be treated as exploratory follow-up rather than as a primary omnibus test.

Value

A crossstatr_spectral_change object with elements:

test_statistic	numeric scalar; the F-statistic, T-squared, or max abs t-statistic (depending on method).
p_value	numeric scalar p-value.
method	method label.
n_perm	number of permutations used (NA for non-permutation methods).
pointwise	for pointwise_t, a data frame of wavelength, t, p, p_adj, significant; otherwise NULL.
call	the matched call.

Methodology

Hotelling T-squared (two-group case):

$$T^2 = \frac{n_1 n_2}{n_1 + n_2}(\bar{y}_1 - \bar{y}_2)' S_p^{-1}(\bar{y}_1 - \bar{y}_2),$$

where S_p is the pooled within-group covariance. When $p < n_1 + n_2 - 1$ and S_p is invertible, the F-transform

$$F = \frac{n_1 + n_2 - p - 1}{p(n_1 + n_2 - 2)} T^2$$

is referenced to $F_{p,\, n_1+n_2-p-1}$ (Hotelling, 1931).

When S_p is singular (typical when $p \geq n$), the Moore-Penrose inverse (MASS::ginv) is used and the chi-square approximation $T^2 \sim \chi^2_p$ is reported; this is liberal in small samples and should be supplemented with PERMANOVA.

PERMANOVA pseudo-F (Anderson, 2001). Let d_{ij} be the Euclidean distance between rows `i` and `j` of `spectra`. The total sum of squares is

$$\mathrm{SS}_T = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^2,$$

the within-group sum of squares is

$$\mathrm{SS}_W = \sum_{g} \frac{1}{n_g} \sum_{i,j \in g} d_{ij}^2,$$

and the between-group sum of squares is $\mathrm{SS}_A = \mathrm{SS}_T - \mathrm{SS}_W$. The pseudo-F is

$$F = \frac{\mathrm{SS}_A / (g-1)}{\mathrm{SS}_W / (n-g)}.$$

Permutation of group labels gives the empirical p-value

$$p = \frac{1 + \#\{F_b \geq F_{\mathrm{obs}}\}}{B + 1}.$$

Pointwise two-sample t-tests are run at each wavelength, with p-values adjusted using `stats::p.adjust` (default `"fdr"`, Benjamini and Hochberg, 1995). The reported `test_statistic` is the maximum absolute t-statistic; the reported `p_value` is the minimum adjusted p-value, which provides a global test with strong control under independence.

Limitations

- The Hotelling F-transform requires $p < n_1 + n_2 - 1$. For wider spectra the method falls back to a chi-square approximation, which is anti-conservative; PERMANOVA is the safer choice in this regime.
- PERMANOVA assumes exchangeability of observations under the null. When groups have very different dispersions (not just means) the test can be significant even without location differences; see Anderson (2006) and Warton et al. (2012).
- The pointwise tests ignore the multivariate dependence across wavelengths. The minimum adjusted p-value should be read as exploratory rather than as a definitive global test.
- Rows with missing values are silently dropped before testing; this may bias the analysis if missingness is informative.

References

- Anderson, M. J. (2001). A new method for non-parametric multivariate analysis of variance. Austral Ecology, 26, 32-46. doi:10.1111/j.14429993.2001.01070.pp.x.
- Anderson, M. J. (2006). Distance-Based Tests for Homogeneity of Multivariate Dispersions. Biometrics, 62, 245-253. doi:10.1111/j.15410420.2005.00440.x.
- Hotelling, H. (1931). The Generalization of Student's Ratio. The Annals of Mathematical Statistics, 2, 360-378. doi:10.1214/aoms/1177732979.
- Benjamini, Y. and Hochberg, Y. (1995). Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society Series B: Statistical Methodology, 57, 289-300. doi:10.1111/j.25176161.1995.tb02031.x.
- Warton, D. I., Wright, S. T. and Wang, Y. (2012). Distance-based multivariate analyses confound location and dispersion effects. Methods in Ecology and Evolution, 3, 89-101. doi:10.1111/j.2041210X.2011.00127.x.

Examples

```
set.seed(1)
# 12 spectra x 50 wavelengths, 2 groups differing in 5 channels
wl <- seq(400, 798, length.out = 50)
Y <- matrix(rnorm(12 * 50, sd = 0.3), nrow = 12, ncol = 50)
Y[1:6, 21:25] <- Y[1:6, 21:25] + 0.8
g <- factor(rep(c("ctrl", "trt"), each = 6))

# PERMANOVA on the full spectra
spectral_change_test(Y, g, wavelengths = wl,
                     method = "permanova", n_perm = 199)

# Hotelling T^2
spectral_change_test(Y, g, method = "t2")

# Pointwise t-tests with BH adjustment
r <- spectral_change_test(Y, g, wavelengths = wl, method = "pointwise_t")
head(r$pointwise)
```

`summary.crossstatr_result`

Generic: summary of crossStatR result

Description

Generic: summary of crossStatR result

Usage

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

Arguments

`object`	result object
`...`	additional arguments

`tidy.crossstatr_result` *Generic: tidy a crossStatR result*

Description

Generic: tidy a crossStatR result

Usage

```
tidy.crossstatr_result(x, ...)
```

Arguments

`x`	result object
`...`	additional arguments

`timepoint_optimize` *Optimise sampling timepoints for an experiment*

Description

Given a kinetic model `model_func(params, t) -> y`, chooses additional sampling timepoints that maximise the expected information about the parameters, given a set of fixed (already scheduled) times. Several optimal-design criteria are supported: D-optimal (maximise the determinant of the information matrix), A-optimal (minimise the trace of its inverse), C-optimal (minimise the variance of a specific linear combination of parameters), and a model-discrimination criterion that maximises the detectable difference between two competing models.

Usage

```
timepoint_optimize(
  model_func,
  params,
  t_range = c(0, 100),
  n_candidates = 10L,
  criterion = c("d_optimal", "a_optimal", "c_optimal", "model_discrimination"),
  fixed_times = NULL,
  n_extra = 3L,
  conf_level = 0.95,
```

```
  competitor_model = NULL,
  competitor_params = NULL,
  c_vector = NULL
)
```

Arguments

model_func	function f(params, t) -> y returning predicted responses.
params	numeric vector of (assumed true) parameter values.
t_range	numeric length-2 vector giving the time interval.
n_candidates	integer; number of points in the candidate grid.
criterion	"d_optimal", "a_optimal", "c_optimal", or "model_discrimination".
fixed_times	numeric vector of already-fixed sampling times.
n_extra	integer; number of additional points to choose.
conf_level	confidence level (stored on the result).
competitor_model	required for criterion = "model_discrimination": a function f(params, t) -> y for the competitor model.
competitor_params	parameter values for the competitor model.
c_vector	required for criterion = "c_optimal": a numeric vector of length length(params) defining the linear combination $c'\theta$ whose variance is to be minimised.

Details

The search is greedy: starting from fixed_times, the function picks the candidate that most improves the criterion, repeats for n_extra points, and returns the augmented design. Greedy is fast but not guaranteed to find the global optimum; it is a good practical approximation when the number of points is small.

Value

A crossstatr_timepoint object with elements:

optimal_times	numeric vector of chosen sampling times (fixed_times + extra).
criterion_value	value of the criterion at the chosen design (for D-optimal, det(M); for A-optimal, -trace(M^-1); for C-optimal, -c'M^-1 c; for model_discrimination, the squared prediction difference scaled by total sensitivity).
sensitivity_matrix	the matrix G of dy/dparam at the chosen times.
information_matrix	M = G'G (returned for D/A/C criteria).
criterion	criterion label.
model_label	label identifying the model(s) used.
call	the matched call.

Methodology

Let $G_{ij} = \partial f_i / \partial \theta_j$ be the sensitivity matrix evaluated at the candidate times. The Fisher information matrix under a Gaussian noise model with unit variance is

$$M = G'G.$$

D-optimal design maximises $\det(M)$, which corresponds to minimising the volume of the joint parameter confidence ellipsoid (Fedorov, 1972).

A-optimal design minimises $\mathrm{tr}(M^{-1})$, the sum of the parameter variances (Kiefer, 1959).

C-optimal design minimises $c'M^{-1}c$, the variance of a specific linear combination $c'\theta$ (Silvey, 1980).

Model discrimination: The package maximises

$$\frac{\sum_i (f_1(\theta_1, t_i) - f_2(\theta_2, t_i))^2}{\sum_i (G_{1i}^2 + G_{2i}^2) + \epsilon}$$

where the numerator captures how far apart the models are at the chosen times and the denominator approximates the squared standard error of the difference (Lopez-Fidalgo and Tommasi, 2007). Higher values are better.

Sensitivities are computed by central finite differences via `.num_jacobian`.

Limitations

- Local optimality: the criteria are evaluated at the supplied `params` only. Bayesian or minimax designs that integrate over a prior on `params` are more robust when the assumed parameter values are uncertain; this function does not implement them.
- The greedy search may not find the global optimum. For `n_extra <= 4` and small candidate grids, an exhaustive search would be feasible but is not currently implemented.
- The information matrix assumes unit noise variance; for non-unit variance the criteria scale by `1 / sigma^2` and the relative ordering of designs is unchanged, so the same `optimal_times` would be selected.
- For `a_optimal` and `c_optimal`, when `M` is singular (which happens early in the greedy search when fewer timepoints are chosen than there are parameters), the method falls back to the Moore-Penrose pseudo-inverse `MASS::ginv(M)`. The criterion is still well-defined and guides the search toward full-rank designs, but the literal interpretation as `trace(M^{-1})` requires a non-singular `M`.

References

- Fedorov, V. V. (1972). Theory of Optimal Experiments. Academic Press. https://shop.elsevier.com/books/theory-of-optimal-experiments/fedorov/978-0-12-250750-2.
- Atkinson, A. C., Donev, A. N. and Tobias, R. D. (2007). Optimum Experimental Designs, with SAS. Oxford University Press. doi:10.1093/oso/9780199296590.001.0001.

Examples

```
# First-order decay model f(p, t) = A * exp(-k * t)
model <- function(p, t) p[1] * exp(-p[2] * t)
params <- c(A = 5, k = 0.3)

# D-optimal: 3 extra timepoints in [0, 20]
r <- timepoint_optimize(model, params, t_range = c(0, 20),
                        n_candidates = 20L, criterion = "d_optimal",
                        fixed_times = c(0, 20), n_extra = 3L)
print(r)

# C-optimal: minimise variance of the parameter 'k' (second parameter)
timepoint_optimize(model, params, t_range = c(0, 20),
                   criterion = "c_optimal",
                   c_vector = c(0, 1), n_extra = 3L)

# Model discrimination between first-order and saturation
sat <- function(p, t) p[1] * t / (p[2] + t)
timepoint_optimize(model, params, t_range = c(0, 20),
                   criterion = "model_discrimination",
                   competitor_model = sat,
                   competitor_params = c(Vmax = 5, Km = 3),
                   n_extra = 3L)
```

---

uncertainty_origin *Stage-wise uncertainty attribution framework*

Description

Provides a structured way to attribute the final measurement uncertainty to the various experimental stages: sampling, instrument noise, calibration, preprocessing, replicate variation, model selection, parameter estimation, and drift. The framework is explicit about which attributions are mathematically additive (variance decomposition under independence) and which require a more careful treatment (correlated sources, multiplicative effects).

Usage

```
uncertainty_origin(
  stages,
  correlations = NULL,
  covariances = NULL,
  model_selection_uncertainty = NULL,
  n_mc = 5000L,
  conf_level = 0.95,
  seed = NULL
)
```

Arguments

`stages`	named list of variance estimates. Names correspond to stages (e.g. `sampling`, `instrument_noise`, `calibration`, `replicate`, `drift`). Each entry is a numeric scalar giving the variance contributed by that stage. Required.
`correlations`	optional named list of pairwise correlations between stages. Each entry is a numeric scalar with name like `"sampling__calibration"` giving the correlation between those two stages. Defaults to no correlation (full additivity).
`covariances`	optional covariance matrix, superseding `correlations` if supplied. Must have `length(stages)` rows/cols.
`model_selection_uncertainty`	optional numeric scalar giving the additional variance attributable to model-selection uncertainty (this is *not* variance decomposition in the strict sense; see Details).
`n_mc`	number of Monte Carlo replicates for the joint propagation under non-zero correlations (default 5000).
`conf_level`	confidence level for the final CI.
`seed`	optional integer seed.

Value

A `crossstatr_uncertainty_origin` object.

Methodology

Additive case (all sources independent): the total variance is

$$\sigma^2_{\text{total}} = \sum_k \sigma^2_k,$$

and the contribution of stage k is

$$c_k = \sigma^2_k / \sigma^2_{\text{total}}.$$

This is the simplest case and is the only one in which the attribution is rigorously additive.

Correlated sources: if stages k and j are correlated with ρ_{kj}, the total variance is

$$\sigma^2_{\text{total}} = \sum_k \sigma^2_k + 2 \sum_{k<j} \rho_{kj}\, \sigma_k\, \sigma_j.$$

The implemented allocation uses the cooperative game $v(S) = \text{Var}(\sum_{k \in S} X_k)$. Its Shapley allocation splits each covariance term equally between its stages:

$$\phi_k = \sigma^2_k + \sum_{j \neq k} \text{Cov}(X_k, X_j).$$

This sums to the variance of the additive total. Contributions can be negative when covariances are negative. It is a different game from the conditional-expectation Shapley effects studied by Owen and Prieur (2017) and Song et al. (2016); those papers are related reading, not a derivation of the implemented allocation.

Monte Carlo draws use a zero-mean multivariate normal distribution with the supplied covariance matrix and are summed additively. `total_ci` contains quantiles of that simulated total, not a confidence interval for its variance or its stage allocations. Nonlinear propagation is not implemented. The optional `model_selection_uncertainty` value is returned separately without being added to the simulated total. Draper (1995) discusses model uncertainty generally; this function does not implement Bayesian model averaging.

Limitations

- The Shapley attribution is well-defined only when the variance contributions and correlations are themselves estimable. With noisy estimates of ρ_{kj}, the attribution is unstable.
- Model-selection uncertainty is treated as a separate addendum, not folded into the variance decomposition. This is intentional; the alternative (BMA-style folding) requires priors over models.
- For multiplicative uncertainty (e.g. relative standard uncertainties), the additive decomposition does not apply; use a separately specified transformation and propagation model.
- The function assumes the stage variances have already been estimated (e.g. via `batch_effect_quantify()`, `drift_uncertainty()`, `noise_characterize()`); it does not estimate them.

References

- Owen, A. B. and Prieur, C. (2017). On Shapley Value for Measuring Importance of Dependent Inputs. SIAM/ASA Journal on Uncertainty Quantification, 5, 986-1002. doi:10.1137/16M1097717.
- Song, E., Nelson, B. L. and Staum, J. (2016). Shapley effects for global sensitivity analysis: Theory and computation. SIAM/ASA Journal on Uncertainty Quantification, 4, 1060-1083. doi:10.1137/15M1048070.
- Draper, D. (1995). Assessment and Propagation of Model Uncertainty. Journal of the Royal Statistical Society Series B: Statistical Methodology, 57, 45-70. doi:10.1111/j.25176161.1995.tb02015.x.
- Joint Committee for Guides in Metrology (2008). Evaluation of measurement data - Guide to the expression of uncertainty in measurement. JCGM 100:2008. BIPM. doi:10.59161/JCGM1002008E.

Examples

```
stages <- c(sampling = 0.05, instrument = 0.10,
            calibration = 0.02, drift = 0.03)
res <- uncertainty_origin(stages)
print(res)

# With correlations
res2 <- uncertainty_origin(stages,
  correlations = list(sampling__instrument = 0.3,
                      calibration__drift = 0.4))
print(res2)
```

`weighted_peak_fit` *Weighted multi-peak fitting*

Description

Fits one or more peaks of a common shape to a signal by nonlinear least squares. Peaks may be located automatically by a local-maxima plus prominence rule, or supplied manually via `start`. The three supported shapes are Gaussian, Lorentzian, and pseudo-Voigt (a linear mixture of the two). The fit uses `stats::nls` with `algorithm = "port"` and lower bounds on heights and widths, which prevents degenerate solutions where a peak collapses to a delta spike or vanishes to negative height.

Usage

```
weighted_peak_fit(
  x,
  y,
  peaks = c("auto", "manual"),
  n_peaks = NULL,
  peak_shape = c("gaussian", "lorentzian", "voigt"),
  start = NULL,
  weights = NULL,
  max_iter = 500L,
  conf_level = 0.95
)
```

Arguments

`x`	numeric vector of abscissae.
`y`	numeric vector of signal intensities, same length as `x`.
`peaks`	character; `"auto"` (default) for automatic peak detection, or `"manual"` to use `start`.
`n_peaks`	integer; if provided with `peaks = "auto"`, take the top `n_peaks` peaks by prominence. If NULL, all peaks above the prominence threshold are kept.
`peak_shape`	character; one of `"gaussian"`, `"lorentzian"`, or `"voigt"` (pseudo-Voigt approximation).
`start`	optional list or named numeric vector of starting values for the parameters when `peaks = "manual"`. Names should follow the pattern `h1, c1, w1, [eta1], h2, c2, w2, [eta2], ...`.
`weights`	optional numeric weights (e.g. `1/sigma^2`). Length must match `x` and `y`. Recycled if length 1.
`max_iter`	integer; passed to `nls` as `control$maxiter`.
`conf_level`	confidence level for parameter CIs.

Details

The package does not automatically pick the number of peaks by AIC. The consequences of an extra spurious peak in downstream analysis are usually larger than the consequences of a missed peak, so the package leaves the judgement about peak count to the analyst. Inspect `aic` and `bic` alongside the residual plot before committing to a fit.

Value

A `crossstatr_peak_fit` object with elements:

`coefficients`	data frame with columns `peak`, `param`, `est`, `se`, `ci_low`, `ci_high`.
`fitted`	fitted signal at each `x`.
`residual`	`y - fitted`.
`n_peaks`	number of peaks fitted.
`peak_shape`	shape used.
`aic`, `bic`	AIC and BIC of the fit.
`converged`	logical; whether `nls` converged.
`call`	the matched call.

Methodology

Each peak is modelled by a parametric line shape. For a Gaussian,

$$G(x; h, c, w) = h \, \exp\!\left(-\frac{(x-c)^2}{2w^2} \right),$$

and for a Lorentzian,

$$L(x; h, c, w) = h \, \frac{w^2}{(x-c)^2 + w^2},$$

where h is the peak height, c the centre, and w a width parameter (equal to the standard deviation for the Gaussian and to the half-width at half-maximum for the Lorentzian).

The implemented pseudo-Voigt-style mixture is

$$pV(x) = \eta \, L(x; h, c, w) + (1 - \eta) \, G(x; h, c, w),$$

with mixing parameter $\eta \in [0, 1]$. This shares the same height, centre, and width between the Gaussian and Lorentzian components, which is a simplification of the true Voigt profile (the convolution of Gaussian and Lorentzian). The mixing weight is fitted from a starting value of 0.5; this is not the width-dependent approximation of Thompson, Cox and Hastings (1987), which is related background.

The total model is the sum of K peaks,

$$\hat{y}(x_i) = \sum_{k=1}^{K} s_k(x_i; h_k, c_k, w_k, [\eta_k]),$$

and the package minimises the (weighted) sum of squared residuals

$$S(\theta) = \sum_i w_i \, (y_i - \hat{y}(x_i))^2.$$

Parameter standard errors come from the `nls` asymptotic covariance $\widehat{\mathrm{Cov}}(\hat{\theta}) = \hat{\sigma}^2 (J^\top W J)^{-1}$, and CIs are normal-approximation using the residual df.

Automatic peak detection finds local maxima (points where the signal exceeds both neighbours) whose prominence exceeds a robust threshold $2 \cdot 1.4826 \cdot \mathrm{MAD}(y)$. The starting width for each peak is set to one-tenth of the x-range divided by the number of peaks, which is a safe default for typical chromatographic resolution.

Limitations

- `nls` with `algorithm = "port"` will fail to converge if the peaks are too close together relative to their widths, or if `start` is far from the optimum. Always inspect the residuals.
- Pseudo-Voigt with shared width between Gaussian and Lorentzian components is an approximation. For high-resolution spectroscopy where the line shape itself is the quantity of interest, fit a true Voigt profile (e.g. via the `voigt` package) instead.
- Parameter CIs use the t-distribution with residual df and assume the asymptotic normal approximation holds. For low signal-to-noise or near-boundary parameters this can be misleading.
- The prominence threshold of 2*MAD is permissive. On noisy data it will detect spurious peaks; supply `n_peaks` or use `peaks = "manual"` when the peak count is known.

References

- Bates, D. M. and Watts, D. G. (1988). Nonlinear Regression Analysis and Its Applications. Wiley. doi:10.1002/9780470316757.
- Thompson, P., Cox, D. E. and Hastings, J. B. (1987). Rietveld refinement of Debye-Scherrer synchrotron X-ray data from Al2O3. Journal of Applied Crystallography, 20, 79-83. doi:10.1107/S0021889887087090.

Examples

```
set.seed(1)
x <- seq(0, 10, length.out = 300)
y <- 5 * exp(-(x - 3)^2 / (2 * 0.2^2)) +
     3 * exp(-(x - 7)^2 / (2 * 0.3^2)) +
     rnorm(length(x), sd = 0.05)
fit <- weighted_peak_fit(x, y, peaks = "auto", n_peaks = 2,
                         peak_shape = "gaussian")
print(fit)

# Manual start for two known peaks
fit2 <- weighted_peak_fit(x, y, peaks = "manual", peak_shape = "gaussian",
                          start = list(h1 = 5, c1 = 3, w1 = 0.2,
                                       h2 = 3, c2 = 7, w2 = 0.3))
print(fit2)
```

Index

adaptive_baseline, 3
adaptive_smoothing, 5

batch_effect_quantify, 8
batch_effect_quantify(), *19, 20*, *79*

calibration_compare, 11
calibration_transfer_risk, 13
curve_similarity, 15

detection_reliability, 19
drift_correct_stat, 21
drift_uncertainty, 24
drift_uncertainty(), *15*, *19, 20*, *79*

experiment_fragility, 26

kinetic_discrimination, 28
kinetic_model_compare, 30
kinetic_model_compare(), *30*

lme4::lmer(), *9*

noise_characterize, 33
noise_characterize(), *79*

parameter_uncertainty, 35
peak_evidence, 37

replicate_decompose, 40
replicate_decompose(), *55*
robust_lod, 42

sci_anova, 45
sci_bootstrap, 46
sci_bootstrap(), *58*
sci_compare, 49
sci_correlation, 51
sci_correlation(), *58*
sci_describe, 54
sci_effect_size, 56
sci_effect_size(), *50*
sci_normality, 59
sci_outliers, 61
sci_power, 63
sci_regression, 66
sci_regression(), *62*
signal_difference_regions, 68
spectral_change_test, 70
stats::anova(), *10*
stats::lm(), *9*
stats::p.adjust(), *52*, *71*
stats::power.t.test, *64*
summary.crossstatr_result, 73

tidy.crossstatr_result, 74
timepoint_optimize, 74

uncertainty_origin, 77

weighted_peak_fit, 80
weighted_peak_fit(), *39*