Package: metaLong (via r-universe)

June 1, 2026

Title Longitudinal Meta-Analysis with Robust Variance Estimation and Sensitivity Analysis

Version 0.1.0

Description Tools for longitudinal meta-analysis where studies contribute effect sizes at multiple follow-up time points. Implements robust variance estimation (RVE) with Tipton small-sample corrections following Hedges, Tipton, and Johnson (2010) <doi:10.1002/jrsm.5> and Tipton (2015) <doi:10.1037/met0000011>, time-varying sensitivity analysis via the Impact Threshold for a Confounding Variable (ITCV) following Frank (2000) <doi:10.1177/0049124100029002003>, benchmark calibration of the ITCV threshold against observed study-level covariates, spline-based nonlinear time-trend modeling with a nonlinearity test, and leave-k-out fragility analysis across the follow-up trajectory. Designed for researchers synthesising evidence from studies with repeated outcome measurement in education, psychology, health, and the social sciences.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports metafor (>= 3.8-1), splines, stats, utils

Suggests clubSandwich (>= 0.5.10), testthat (>= 3.0.0), knitr, rmarkdown, ggplot2, dplyr

VignetteBuilder knitr

Config/testthat/edition 3

URL `https://github.com/causalfragility-lab/metaLong`

BugReports `https://github.com/causalfragility-lab/metaLong/issues`

Repository https://causalfragility-lab.r-universe.dev

Date/Publication 2026-03-25 19:47:03 UTC

RemoteUrl https://github.com/causalfragility-lab/metalong

RemoteRef HEAD

RemoteSha 24cd4bd26ef0860c4a59c52b0fe443865c22ed4e

Contents

`fits` *Extract stored fitted model objects*

Description

Extract stored fitted model objects

Usage

```
fits(x)
```

Arguments

`x`	An `ml_meta` object.

Value

Named list of fitted model objects, one per estimable time point.

`ml_benchmark` *Benchmark Calibration of Longitudinal ITCV Against Observed Covariates*

Description

For each follow-up time point, regresses each observed study-level covariate on the effect sizes using RVE meta-regression, extracts the covariate's partial correlation with the outcome, and compares it to the significance-adjusted ITCV threshold from `ml_sens()`. A covariate that *beats* the threshold demonstrates that real-world confounding of at least that magnitude exists, which is direct evidence of effect fragility.

Usage

```
ml_benchmark(
  data,
  meta_obj,
  sens_obj,
  yi,
  vi,
  study,
  time,
  covariates,
  alpha = NULL,
  rho = 0.8,
  small_sample = TRUE,
  min_k = 3L
)
```

Arguments

`data`	Long-format `data.frame`.
`meta_obj`	Output from `ml_meta()`.
`sens_obj`	Output from `ml_sens()`.
`yi, vi, study, time`	Column names.
`covariates`	Character vector of observed moderator column names to benchmark.
`alpha`	Significance level (inherits from `meta_obj` if `NULL`).
`rho`	Working within-study correlation for V matrix.
`small_sample`	Logical; use CR2 + Satterthwaite?
`min_k`	Minimum studies required at a time point. Default `3L` (one extra relative to `ml_meta()` because regression needs more d.f.).

Value

Object of class `ml_benchmark` (a `data.frame`) with columns:

- `time` Follow-up time.
- `covariate` Covariate name.
- `k` Number of studies.
- `r_partial` Partial correlation of covariate with effect size.
- `t_stat`, `df`, `p_val` RVE inference for the covariate slope.
- `itcv_alpha` ITCV_alpha threshold at this time point.
- `beats_threshold` Logical: does `|r_partial| >= itcv_alpha`?
- `skip_reason` Character; reason a cell was skipped, else `NA`.

The `"fragile_summary"` attribute contains one row per time with counts.

Interpretation

If an observed covariate (e.g., publication year, sample quality, attrition rate) has `|r_partial| >= ITCV_alpha(t)`, then an *unobserved* confounder with the same relationship to exposure and outcome would be sufficient to nullify the pooled effect at time t. This does not prove confounding–it calibrates the plausibility threshold.

See Also

`ml_sens()`, `ml_meta()`

Examples

```
dat   <- sim_longitudinal_meta(k = 15, times = c(0, 6, 12), seed = 2)
meta  <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
sens  <- ml_sens(dat, meta, yi = "yi", vi = "vi", study = "study", time = "time")
bench <- ml_benchmark(dat, meta, sens,
                       yi = "yi", vi = "vi", study = "study", time = "time",
                       covariates = c("pub_year", "quality"))
print(bench)
plot(bench)
```

`ml_fragility` *Leave-One-Out and Leave-k-Out Fragility Analysis*

Description

Computes fragility indices for each time point by systematically removing studies and re-estimating the pooled effect. The fragility index at time t is the minimum number of studies whose removal changes the statistical conclusion (significant -> non-significant or vice versa).

Usage

```
ml_fragility(
  data,
  meta_obj,
  yi,
  vi,
  study,
  time,
  max_k = 5L,
  max_combinations = 500L,
  alpha = NULL,
  rho = 0.8,
  small_sample = TRUE,
  seed = NULL
)
```

Arguments

`data` Long-format `data.frame`.

`meta_obj` Output from `ml_meta()`.

`yi`, `vi`, `study`, `time`
Column names.

`max_k` Maximum number of studies to remove. Default `5`.

`max_combinations`
Maximum number of combinations to test per k. Default `500`. Larger values are more exhaustive but slower.

`alpha` Significance level.

`rho` Working correlation.

`small_sample` Use CR2 + Satterthwaite?

`seed` Random seed for sampling combinations. Default `NULL`.

Details

At each time point, studies are removed one at a time (or in combinations for the leave-k-out version) and the model is re-fit. The fragility index is the smallest k such that removing any set of k studies flips the significance of the pooled estimate. A fragility index of 1 means a single study's removal changes the conclusion.

For the leave-k-out version, a random sample of combinations is used when the number of combinations is large (controlled by `max_combinations`).

Value

Object of class `ml_fragility` (a `data.frame`) with columns:

`time` Follow-up time.

`k_studies` Number of studies at this time point.

p_original Original p-value.

sig_original Was the original result significant?

fragility_index Min number of removals to flip significance. NA if not found within max_k.

fragility_quotient fragility_index / k_studies (proportion).

study_removed Study ID whose removal achieved the flip (leave-one-out only).

Examples

```
dat  <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12), seed = 5)
meta <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
frag <- ml_fragility(dat, meta, yi = "yi", vi = "vi",
                      study = "study", time = "time",
                      max_k = 1L, seed = 1)
print(frag)
```

---

ml_meta *Longitudinal Meta-Analysis with Robust Variance Estimation*

Description

Fits a random-effects meta-analytic model at each unique time point in a long-format dataset of multi-wave effect sizes. Inference uses robust variance estimation (RVE) with optional Tipton (2015) small-sample corrections via the clubSandwich package.

Usage

```
ml_meta(
  data,
  yi,
  vi,
  study,
  time,
  alpha = 0.05,
  rho = 0.8,
  small_sample = TRUE,
  min_k = 2L,
  method = "REML",
  engine = c("rma.uni", "rma.mv")
)
```

Arguments

`data`	A `data.frame` in **long format**: one row per study x time point.
`yi`	Character. Name of the effect-size column.
`vi`	Character. Name of the sampling-variance column.
`study`	Character. Name of the study-ID column (cluster variable).
`time`	Character. Name of the follow-up time column (numeric).
`alpha`	Significance level for confidence intervals and p-values. Default `0.05`.
`rho`	Assumed within-study correlation between effect sizes (used only when `engine = "rma.mv"`). Default `0.8`.
`small_sample`	Logical. If `TRUE` (default), applies CR2 sandwich variance estimation with Satterthwaite degrees of freedom (Tipton, 2015). If `FALSE`, uses uncorrected z-based inference.
`min_k`	Integer. Minimum number of studies required to fit a model at a given time point. Default `2`.
`method`	Character. Variance estimator passed to metafor. Default `"REML"`.
`engine`	Character. Fitting engine: `"rma.uni"` (default) or `"rma.mv"`. See section *Engine choice*.

Value

An object of class `ml_meta` (a `data.frame`) with one row per time point and columns: `time`, `k`, `theta`, `se`, `df`, `t_stat`, `p_val`, `ci_lb`, `ci_ub`, `tau2`, `note`.

Attributes:

`"fits"` Named list of fitted model objects (one per time point).

`"weights_by_time"` Named list of weight vectors for downstream use by `ml_sens()` and `ml_benchmark()`.

`"engine"`, `"alpha"`, `"rho"`, `"small_sample"` Call metadata.

Engine choice

Two fitting engines are supported:

`"rma.uni"` **(default)** `metafor::rma.uni()` – appropriate when each study contributes exactly one effect size per time point. Simpler, faster, and stores `tau2` directly from the REML estimate.

`"rma.mv"` `metafor::rma.mv()` with a prebuilt working covariance matrix – appropriate when studies contribute *multiple* effect sizes at the same time point (dependent effects within cluster). Requires the `rho` argument.

References

Hedges, L. V., Tipton, E., & Johnson, M. C. (2010). Robust variance estimation in meta-regression with dependent effect size estimates. *Research Synthesis Methods*, 1(1), 39-65.

Tipton, E. (2015). Small sample adjustments for robust variance estimation with meta-regression. *Psychological Methods*, 20(3), 375-393.

See Also

ml_sens(), ml_benchmark(), ml_spline()

Examples

```
dat <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12), seed = 1)
result <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
print(result)
plot(result)

# rma.mv engine for dependent effects

result_mv <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time",
                     engine = "rma.mv", rho = 0.8)
```

---

ml_plot *Combined Publication-Ready Trajectory Figure*

Description

Produces a multi-panel figure combining the pooled trajectory, confidence band, spline fit (if supplied), and ITCV sensitivity profile. Designed for direct inclusion in manuscripts.

Usage

```
ml_plot(
  meta_obj,
  sens_obj = NULL,
  bench_obj = NULL,
  spline_obj = NULL,
  frag_obj = NULL,
  ncol = NULL,
  main = NULL,
  col_effect = "#2166ac",
  col_sens = "#d73027",
  col_spline = "#1a9641",
  delta = NULL
)
```

Arguments

meta_obj	Output from ml_meta() (required).
sens_obj	Output from ml_sens() (optional; adds ITCV panel).
bench_obj	Output from ml_benchmark() (optional; adds benchmark marks).
spline_obj	Output from ml_spline() (optional; overlays spline).

frag_obj	Output from ml_fragility() (optional; adds fragility panel).
ncol	Number of columns in the panel layout. Default auto.
main	Overall figure title.
col_effect	Colour for the pooled effect trajectory.
col_sens	Colour for the ITCV line.
col_spline	Colour for the spline curve.
delta	Fragility benchmark line on the ITCV panel. Inherits from sens_obj if available.

Value

Invisibly returns a list of the objects passed in.

Examples

```
dat  <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12), seed = 1)
meta <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
sens <- ml_sens(dat, meta, yi = "yi", vi = "vi", study = "study", time = "time")
ml_plot(meta, sens_obj = sens)


spl  <- ml_spline(meta, df = 2)
ml_plot(meta, sens_obj = sens, spline_obj = spl,
        main = "Longitudinal Meta-Analysis Profile")
```

---

ml_sens *Time-Varying Sensitivity Analysis via Longitudinal ITCV*

Description

Computes the Impact Threshold for a Confounding Variable (ITCV) at each follow-up time point using the pooled estimates and robust inference from ml_meta(). Two versions are returned: the raw ITCV (threshold to nullify the pooled effect) and the significance-adjusted ITCV_alpha (threshold to render the result non-significant under small-sample-corrected inference).

Usage

```
ml_sens(data, meta_obj, yi, vi, study, time, alpha = NULL, delta = 0.15)
```

Arguments

`data`	A `data.frame` in long format (same as passed to `ml_meta()`).
`meta_obj`	Output from `ml_meta()`.
`yi, vi, study, time`	Column names (same meaning as in `ml_meta()`).
`alpha`	Significance level. Defaults to the value stored in `meta_obj` (or `0.05` if absent).
`delta`	Numeric. User-defined practical fragility benchmark: time points with `ITCV_alpha(t)` < `delta` are flagged as "practically fragile". Default `0.15`.

Value

An object of class `ml_sens` (a `data.frame`) with columns:

`time` Follow-up time.

`theta`**,** `se`**,** `df` Copied from `meta_obj`.

`sy` Weighted SD of observed effect sizes.

`r_effect` Pooled effect on correlation scale.

`itcv` Raw ITCV: confounding needed to nullify the estimate.

`itcv_alpha` Significance-adjusted ITCV: confounding needed to make the result non-significant.

`fragile` Logical; `TRUE` when `itcv_alpha` < `delta`.

Attributes include trajectory summaries (`itcv_min`, `itcv_mean`, `fragile_prop`) and a `"fragile_times"` character vector.

Mathematical background

At each time t, let $\hat{\theta}_t$ be the pooled effect, $s^2_{y,t}$ the weighted variance of observed effect sizes, and $c_t = t_{1-\alpha/2,\nu_t} \cdot \widehat{SE}(\hat{\theta}_t)$ the minimum effect still deemed significant. The correlation-scale pooled effect is

$$r_t = \hat{\theta}_t / \sqrt{\hat{\theta}_t^2 + s^2_{y,t}}$$

and the raw ITCV is $\sqrt{|r_t|}$. The significance-adjusted version replaces $\hat{\theta}_t$ with $|\hat{\theta}_t| - c_t$.

References

Frank, K. A. (2000). Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29(2), 147-194.

See Also

`ml_meta()`, `ml_benchmark()`, `ml_plot()`

Examples

```
dat  <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12), seed = 1)
meta <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
sens <- ml_sens(dat, meta, yi = "yi", vi = "vi", study = "study", time = "time")
print(sens)
plot(sens)
```

---

ml_spline — *Spline-Based Nonlinear Time Trend in Longitudinal Meta-Analysis*

Description

Fits a natural cubic spline meta-regression over follow-up time using the pooled time-point estimates from `ml_meta()`. Produces a smooth pooled trajectory with simultaneous pointwise confidence bands and tests for nonlinearity.

Usage

```
ml_spline(meta_obj, df = 3L, n_pred = 200L, alpha = NULL, test_linear = TRUE)
```

Arguments

`meta_obj`	Output from `ml_meta()`.
`df`	Degrees of freedom for the natural cubic spline. Default `3`. A value of `1` recovers a linear fit.
`n_pred`	Number of prediction points for the smooth curve. Default `200`.
`alpha`	Confidence level (inherits from `meta_obj` if `NULL`).
`test_linear`	Logical. If `TRUE`, performs an F-test of nonlinearity (spline df > 1 vs linear fit). Default `TRUE`.

Details

The spline is fit by weighted least squares on the `ml_meta()` estimates, using `1 / se^2` as weights (i.e., inverse squared SE weighting to reflect the precision of each time-point estimate). This is a second-stage model.

For a fully joint spline model at the individual-effect level, users should call `metafor::rma.mv()` directly with `mods = ~ ns(time, df)`. This function is primarily intended for visualisation and trajectory testing.

Value

Object of class ml_spline with elements:

pred data.frame with time, fit, ci_lb, ci_ub for the smooth prediction grid.

coef Spline coefficient estimates.

vcov Coefficient covariance matrix.

r_squared Weighted R-squared of the spline fit.

p_nonlinear p-value for nonlinearity test (if requested).

df Spline degrees of freedom used.

meta_obj The original ml_meta object (for plotting).

See Also

ml_meta(), ml_plot()

Examples

```
dat  <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12, 24), seed = 3)
meta <- ml_meta(dat, yi = "yi", vi = "vi", study = "study", time = "time")
spl  <- ml_spline(meta, df = 2)
print(spl)
plot(spl)
```

---

sim_longitudinal_meta *Simulate a Longitudinal Meta-Analytic Dataset*

Description

Generates a synthetic long-format dataset suitable for testing and illustrating all metaLong functions. Studies contribute effect sizes at multiple follow-up time points with within-study correlation.

Usage

```
sim_longitudinal_meta(
  k = 20L,
  times = c(0, 6, 12, 24),
  mu = 0.4,
  tau = 0.2,
  v_range = c(0.02, 0.12),
  missing_prop = 0,
  add_covariates = TRUE,
  seed = NULL
)
```

Arguments

`k`	Number of studies. Default `20`.
`times`	Numeric vector of follow-up time points. Default `c(0, 6, 12, 24)`.
`mu`	Named numeric vector of true effects at each time point, or a single value (recycled). Default `0.4`.
`tau`	Between-study SD. Default `0.2`.
`v_range`	Two-element vector for the uniform sampling variance range. Default `c(0.02, 0.12)`.
`missing_prop`	Proportion of study x time combinations to set missing (simulates unbalanced follow-up). Default `0.0`.
`add_covariates`	Logical. If `TRUE`, adds study-level covariates `pub_year`, `quality`, and `n` for use with `ml_benchmark()`. Default `TRUE`.
`seed`	Random seed. Default `NULL`.

Details

The true effect at time t for study i is

$$\theta_{it} = \mu_t + u_i + \epsilon_{it}$$

where μ_t is a time-varying mean effect (optionally nonlinear), $u_i \sim N(0, \tau^2)$ is a study-level random effect, and $\epsilon_{it} \sim N(0, v_{it})$ is sampling error. Within-study correlation between time points is introduced through u_i.

Value

A `data.frame` in long format with columns:

`study` Study identifier (character).

`time` Follow-up time.

`yi` Observed effect size.

`vi` Sampling variance.

`pub_year`, `quality`, `n` Study-level covariates (if `add_covariates` = `TRUE`).

Examples

```
dat <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12), seed = 42)
head(dat)

# Nonlinear true trajectory

mu_t <- c("0" = 0.2, "6" = 0.5, "12" = 0.4, "24" = 0.1)
dat2 <- sim_longitudinal_meta(k = 10, times = c(0, 6, 12, 24), mu = mu_t,
                               missing_prop = 0.1, seed = 99)
```

tidy *Tidy an metaLong object into a clean data frame*

Description

Tidy an metaLong object into a clean data frame

Usage

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

Arguments

x	A `ml_sens` or `ml_benchmark` object.
...	Additional arguments (unused).

Value

A tidy `data.frame`.

Index