New Features in robustlmm 3.5.0: Robust Inference, Diagnostics and Structured Covariances

Manuel Koller

Abstract

The 3.5.0 release of **robustlmm** adds a layer of inference, diagnostic and modelling tools on top of the Robust Scoring Equations estimator introduced in the main package vignette. This companion vignette is a feature tour. It shows how to obtain a robust cluster-sandwich covariance for the fixed effects, Wald and bootstrap confidence intervals, robust Wald and bootstrap ANOVA tables, prediction intervals, and honest Satterthwaite degrees of freedom that stay valid under contamination. It then covers the influence diagnostics (observation- and cluster-level Cook's distances and robust leverages), a high-breakdown RANSAC initial estimator, Mallows-type design weights that bound the influence of high-leverage design points, and support for structured (`diag`, `cs`, `ar1`) random-effect covariances. Every example runs on a small built-in dataset so that the methods can be reproduced interactively. For the model and the estimation algorithm itself, see the main vignette, `vignette("rlmer")`.

Keywords: robust statistics, mixed-effects model, inference, influence, Cook's distance, sandwich, bootstrap, Satterthwaite, RANSAC, design weights, compound symmetry, R.

1. Introduction

The main **robustlmm** vignette (Koller 2016) introduces the Robust Scoring Equations (RSE) estimator and shows how to fit a model with `rlmer`, tune its robustness and efficiency, and read the robustness weights. This companion vignette documents the features added for the 3.5.0 release. Several of the foundations—the sandwich covariance, the Wald and bootstrap confidence intervals, the prediction intervals, the observation-level influence diagnostics and the base RANSAC start—first shipped to CRAN in releases 3.4-4/3.4-5; this vignette documents the full series. The features fall into four groups:

1. **Inference for the fixed effects** (Section 2): a robust cluster-sandwich covariance (`vcov(., type = "sandwich")`), Wald and bootstrap confidence intervals (`confint`), robust Wald and bootstrap ANOVA tables (`anova`), prediction intervals (`predict`), and honest Satterthwaite degrees of freedom and p values (`summary(., df = ...)`).

2. **Influence diagnostics** (Section 3): observation- and cluster-level Cook's distances (`cooks.distance`), the full per-observation influence matrix (`influence`) and robust leverages (`hatvalues`).

3. **A high-breakdown initial estimator** (Section 4): a RANSAC start (`init = "ransac"`, `ransac_lme4`, `rlmer_ransac`).
4. **Robustness in the design space** (Section 5): Mallows-type design weights (`design.weights`) and structured random-effect covariances (`diag`, `cs`, `ar1`, Section 5.2).

The `cs`/`ar1` covariances (group 4) are *experimental*: they are validated by simulation but rest on newer, partly heuristic machinery, and their interface or defaults may still change; their section title is marked accordingly. The RANSAC start (group 3) and the Mallows design weights (group 4) are stable: settled interfaces with simulation-backed defaults.

All examples use the small built-in datasets `sleepstudy`, `Dyestuff` and `Penicillin` from **lme4** so that they run in seconds. Throughout we fit with `method = "DASvar"`, the fast direct approximation, to keep the vignette quick to build; for final analyses the default `method = "DAStau"` is more accurate (see the main vignette). We start from a robust fit of the `sleepstudy` data, a random-slope model for reaction time as a function of days of sleep deprivation:

```
R> fm <- rlmer(Reaction ~ Days + (Days | Subject), sleepstudy,
+              method = "DASvar")
```

2. Inference for the fixed effects

2.1. Robust cluster-sandwich covariance

The fixed-effect covariance that `summary` reports by default is the linearised (model-based) covariance inherited from the **lme4** machinery. `vcov` now takes a `type` argument that selects between this default and a robust *cluster-sandwich* estimator, $\hat{V} = \hat{A}^{-1}\hat{B}\hat{A}^{-\top}$, where $\hat{A}$ is the marginal (Schur-complement) β-Jacobian and $\hat{B} = \sum_j \boldsymbol{s}_j \boldsymbol{s}_j^\top$ sums the per-cluster β-score contributions. The sandwich is heteroscedasticity- and misspecification-robust and does not assume the fitted covariance is correct.

```
R> vcov(fm, type = "default")

2 x 2 Matrix of class "dpoMatrix"
            (Intercept)   Days
(Intercept)      52.220 -1.536
Days             -1.536  2.616

R> echoWarnings(vcov(fm, type = "sandwich"))

Warning: Cluster sandwich with J = 18 < 20 clusters: with correction =
"G1" it is biased down for between-cluster coefficients, and chi-square /
```

```
normal references are anti-conservative. anova() and confint() use
correction = "CR2" with Bell-McCaffrey degrees of freedom for vcov_type =
"sandwich".
            (Intercept)   Days
(Intercept)      50.284 -1.648
Days             -1.648  2.536
attr(,"n.clusters")
[1] 18
attr(,"correction")
[1] "G1"
```

The default is preserved byte-for-byte, so existing code is unaffected. The small-sample correction `correction = "G1"` (default) applies the $J/(J-1)$ scaling. On a design with a single grouping factor the cluster variable is detected automatically; with several grouping factors pass `cluster = "factor-name"` (also to `confint` and `anova`): with the top level of a nested design the sandwich is exact, on crossed designs only approximate.

A small-J caveat. The sandwich is a large-J approximation. `sleepstudy` has only $J = 18$ subjects — fewer than the 20 clusters below which `vcov_sandwich` warns — which is why the call above prints the warning. With few clusters the G1 sandwich is biased down for between-cluster coefficients and has fewer effective degrees of freedom than $J-1$, so normal or χ^2 references on it are anti-conservative. `correction = "CR2"` gives the bias-reduced sandwich (Bell and McCaffrey 2002) adapted to the robust estimating equations, and `confint` and `anova` use it by default for `vcov_type = "sandwich"`, with Bell–McCaffrey / approximate-Hotelling degrees of freedom (Pustejovsky and Tipton 2018) (Sections 2.2 and 2.3); the simulation evidence is in the section "Small-sample references for the cluster sandwich" of `vignette("simulationStudies")`.

2.2. Confidence intervals

`confint` gains a `method` argument. The default `method = "Wald"` returns the closed-form per-coefficient interval $\hat{\beta}_k \pm q\sqrt{\hat{V}_{kk}}$ from `vcov(., type = vcov_type)`. Bare `confint(fit)` used to error on the `dpoMatrix` covariance; it now returns the Wald interval.

```
R> confint(fm, method = "Wald")
```

```
              2.5 % 97.5 %
(Intercept) 237.106 265.43
Days          7.475  13.81
attr(,"method")
[1] "Wald"
attr(,"vcov_type")
[1] "default"
```

```
R> confint(fm, method = "Wald", vcov_type = "sandwich")
```

```
              2.5 % 97.5 %
(Intercept) 236.310 266.23
Days          7.284  14.01
attr(,"method")
[1] "Wald"
attr(,"vcov_type")
[1] "sandwich"
attr(,"df")
[1] 16.99 16.99
attr(,"correction")
[1] "CR2"
```

For the default covariance the quantile q is the normal $z_{1-\alpha/2}$; for the sandwich it is a Bell–McCaffrey t quantile per coefficient with the CR2 covariance (its degrees of freedom are in `attr(., "df")`). Passing `df = "satterthwaite"` substitutes the robust Satterthwaite t-quantile (Section 2.4), which widens the interval at small J:

```
R> confint(fm, method = "Wald", df = "satterthwaite")
```

```
              2.5 % 97.5 %
(Intercept) 236.617 265.92
Days          7.258  14.03
attr(,"method")
[1] "Wald"
attr(,"vcov_type")
[1] "default"
```

When the asymptotic χ^2_p limit is suspect—typically at small J—a bootstrap is preferable. `confint` delegates `method = "boot"` and `method = "BCa"` to the **confintROB** package (Mason, Cantoni, and Ghisletta 2021, 2024), which implements the parametric and the wild bootstrap (`boot.type`, default `"wild"`), with optional bias-correction-and-acceleration (`method = "BCa"`). **confintROB** is in `Suggests`, so these paths require it to be installed; `method = "Wald"` does not. The bootstrap is costly, so it is not run here:

```
R> ## parametric BCa bootstrap (needs the 'confintROB' package)
R> confint(fm, method = "BCa",  nsim = 1000, seed = 20260601)
R> ## wild bootstrap
R> confint(fm, method = "boot", boot.type = "wild", nsim = 1000,
+          seed = 20260601)
```

Practical guidance from those papers: the χ^2_p Wald limit is adequate for $J \gtrsim 20$ groups, the wild bootstrap is robust to misspecification of the response covariance, and BCa is preferable when the bootstrap distribution is skewed.

2.3. ANOVA: robust Wald and bootstrap tests

`anova` now works on `rlmer` fits. Called on a single fit it prints a per-term robust Wald table:

```
R> anova(fm)

Robust Wald test for fixed effects (rlmer)
  vcov_type = "default"
            Df  Chisq Pr(>Chisq)
(Intercept)  1 1209.0    < 2e-16 ***
Days         1   43.3    4.6e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pass `vcov_type = "sandwich"` to base the table on the cluster-sandwich covariance instead; the table is then the small-J F-test of the CR2 sandwich with approximate-Hotelling denominator degrees of freedom (`ddf = "none"` restores the χ^2 table). Called on two nested fixed-effects models it performs a Wald restriction test:

```
R> fm0 <- rlmer(Reaction ~ 1 + (Days | Subject), sleepstudy,
+               method = "DASvar")
R> anova(fm0, fm)

Robust Wald restriction test (rlmer)
  H0: Days = 0
  vcov_type = "default"
     npar Df Chisq Pr(>Chisq)
fit0    1
fit1    2  1  43.3    4.6e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With `ddf = "satterthwaite"` the table reports an F-test with robust Satterthwaite denominator degrees of freedom:

```
R> anova(fm, ddf = "satterthwaite")

Robust Wald F-test for fixed effects (rlmer)
  denominator df: Satterthwaite (IF-based), vcov_type = "default"
```

```
            NumDF DenDF F value  Pr(>F)
(Intercept)     1  36.3  1209.0 < 2e-16 ***
Days            1  18.8    43.3 2.8e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For variance-component comparisons on the PSD-cone boundary, `test = "boot"` runs a parametric-bootstrap quasi-deviance test (using a common scale to avoid a sign-flip artefact). This refits the model many times and so is not run here:

```
R> anova(fm0, fm, test = "boot", nsim = 1000, seed = 20260601)
```

The bootstrap variance-component test is *experimental*: it is currently the only exposed variance-component path, and a simulation study shows it becomes anti-conservative when whole groups are contaminated. The test guards against this on its own: it warns automatically when the null fit heavily downweights a random-effects group (smallest random-effect robustness weight below 0.5), a direct signal that the bootstrap null built from that fit may be poisoned; `null = "robust"` (below) is the mitigation. The per-term Wald table and the fixed-effect restriction test above are not affected.

Under group contamination the bootstrap null generation itself can be poisoned by the contaminated estimates. The experimental `null = "robust"` option, `anova(fm0, fm, test = "boot", null = "robust")`, generates the bootstrap samples from a contamination-cleaned null fit: clusters whose smallest random-effect robustness weight falls below a tuned threshold are trimmed for the generation step only, while the observed statistic still uses all the data. In simulation this reduces — but does not fully remove — the contamination-induced Type-I inflation at larger J while preserving power. It is validated by simulation.

For the common special case where the alternative adds exactly one independent scalar variance component — as in the example above, `(1|Subject)` vs `(1|Subject) + (0 + Days|Subject)`, or diagonal structures adding one component — the experimental `test = "score"`, `anova(fm0, fm, test = "score")`, is the contamination-robust alternative. It computes a one-sided score statistic from psi-bounded per-cluster contributions of the robust *null* fit only and calibrates it by a score-only parametric bootstrap that refits just the null model per replicate, about four times cheaper than `test = "boot"` (default `nsim = 199`). In simulation the bounded statistic held its size where the deviance bootstrap did not: with 10% of the clusters shifted at $J = 50$, its Type-I error was 0.035 against the deviance bootstrap's 0.115, with clean-null Type-I error 0.045 (0.040) and power 0.920 (0.900), over 200 samples per setting (`inst/simulationStudy/scoreTest.R`, which reproduces these settings end-to-end through the package interface). With 18 groups, an uncentred covariate or 5% outlying observations its Type-I error stayed between 0.015 and 0.075. Contamination aligned with the tested direction is indistinguishable from the alternative for any test with power (with shifted slopes in the contaminated clusters the test rejected in 98% of the samples); the returned per-cluster contributions (`attr(., "boot")$s_j`) identify the driving clusters, to be

inspected together with the cluster diagnostics of Section 3 when a rejection is suspect. Several added independent components are tested jointly (see the section "Several added variance components" of `vignette("simulationStudies")`). Outside its scope (correlated-slope alternatives, crossed or nested grouping factors) it stops with an error; `test = "boot"` remains the default variance-component path. It is validated by simulation.

Under error-distribution contamination the robust Wald test can be substantially more powerful than the classical likelihood-ratio test while matching its Type-I error on clean data; a simulation study quantifies the power gain (see the inference study in `vignette("simulationStudies")`).

2.4. Satterthwaite degrees of freedom

By default `summary` of an `rlmer` fit reports a t-value without a reference distribution, because the small-sample distribution of the robust estimator is not pivotal. The 3.5.0 release adds robust Satterthwaite degrees of freedom (Satterthwaite 1946). Unlike **lmerTest** (Kuznetsova, Brockhoff, and Christensen 2017), the degrees of freedom derive from the robust influence-function-based covariance of the variance parameters, so they stay honest under contamination. `summary` gains a `df` argument; `df = "satterthwaite"` adds `df` and `Pr(>|t|)` columns:

```
R> summary(fm, df = "satterthwaite")$coefficients
```

```
            Estimate Std. Error    df t value  Pr(>|t|)
(Intercept)   251.27      7.226 36.27  34.771 1.884e-29
Days           10.64      1.617 18.83   6.582 2.789e-06
```

The degrees of freedom are consistent across all four inference surfaces: `summary`, `confint(df = "satterthwaite")` (Section 2.2), `anova(ddf = "satterthwaite")` (Section 2.3), and **emmeans**, which now returns finite degrees of freedom instead of `Inf`:

```
R> emmeans::emtrends(fm, ~ 1, var = "Days")
```

```
 1       Days.trend   SE   df lower.CL upper.CL
 overall       10.6 1.62 18.8     7.26       14

Confidence level used: 0.95
```

`summary` shows the degrees of freedom *by default* (`df = "auto"`) whenever it is cheap—either the underlying influence function is already cached on the fit, or a deterministic, dimension-only size workload is below the cutoff `getOption("robustlmm.summary.df.max", 5000)` (machine-independent by construction). Otherwise it falls back to the historic t-value table with a one-line note; `df = "satterthwaite"` forces the computation and `df = "none"` disables it. The (expensive) influence function is cached on the fit and shared by `summary`, `anova`, `confint`, `emmeans`, `cooks.distance` and the sandwich `vcov`, so it is computed once. Single-factor, nested (`(1 | school/class)`) and crossed (`(1 | s) + (1 | item)`) designs

are supported; nested designs use a one-way sandwich over the coarsest grouping factor and crossed designs a Cameron–Gelbach–Miller multiway cluster-robust covariance (Cameron, Gelbach, and Miller 2011). When a variance component is estimated at the boundary (zero), the degrees of freedom are computed conditional on that component being held at zero. The package options are documented on the `help("robustlmm-options")` help page.

The *robust* Satterthwaite degrees of freedom are a new construction. Their nominal coverage has been validated by the package's own Monte-Carlo simulations — for single-factor, nested, crossed and variance-component-boundary designs — rather than by an external published study (see the inference study in `vignette("simulationStudies")`). On a design they do not yet cover (e.g. crossed random slopes) `summary` falls back to the plain t-value table rather than reporting an unvalidated number.

2.5. Prediction intervals

`predict` gains an `interval` argument. The default `interval = "none"` preserves the previous numeric-vector return; `"confidence"` and `"prediction"` return a data frame with columns `fit`, `lwr`, `upr` and `se`. The fixed-effect contribution to the standard error comes from the sandwich `vcov`; the random-effect contribution from the partial influence function of $\hat{u}$.

```
R> nd <- data.frame(Days = 0:2, Subject = "308")
R> suppressWarnings(predict(fm, newdata = nd, interval = "confidence"))

    fit   lwr   upr    se
1 247.8 195.2 300.3 26.82
2 270.6 216.4 324.8 27.63
3 293.4 235.1 351.7 29.77

R> suppressWarnings(predict(fm, newdata = nd, interval = "prediction"))

    fit   lwr   upr    se
1 247.8 184.4 311.2 32.34
2 270.6 205.9 335.3 33.02
3 293.4 225.2 361.7 34.82
```

3. Influence diagnostics

3.1. Observation-level influence

`cooks.distance` measures the joint Mahalanobis influence of each observation on $(\hat{\beta}, \hat{\sigma}, \hat{\theta})$, computed from the full influence function (Cook and Weisberg 1982). We illustrate on the `Penicillin` data (a crossed design):

```
R> fpen <- rlmer(diameter ~ (1 | plate) + (1 | sample), Penicillin,
+                method = "DASvar")
R> cd <- cooks.distance(fpen)
R> sort(cd, decreasing = TRUE)[1:5]

 obs85  obs68  obs19  obs86 obs122
 2.856  2.813  2.755  2.741  2.718
```

`influence(fit)` returns the full per-observation influence matrix (one row per observation, one column per parameter) that `cooks.distance` condenses; `caseweightIF`, `vcov_sandwich` and `implicitIF_full` expose the underlying substrate for advanced use.

3.2. Cluster-level influence and robust leverage

A whole group can be an outlier even when no single observation in it is. `cooks.distance(fit, groups = TRUE)` (or `groups = "<factor>"`) gives a per-cluster Cook's distance—the Mahalanobis norm of the per-cluster influence function—a general diagnostic for such groups. It is, however, not a reliable screen for group contamination ahead of the bootstrap variance-component test of `anova` (Section 2.3): the robust fit absorbs a contaminated group into a heavily downweighted random effect, so the bounded influence saturates exactly when contamination is present. That test's guard therefore uses the random-effect robustness weights, and fires automatically. Cluster level requires a single or nested grouping structure.

```
R> cdc <- cooks.distance(fm, groups = "Subject")
R> sort(cdc, decreasing = TRUE)[1:5]

  335   337   308   309   370
3.591 3.395 3.247 3.101 2.911
```

`hatvalues(fit)` returns the robust leverage (the self-leverage A_{ii} in the random-effect-whitened convolution; it reduces to the classical **lme4** leverage at `rho = cPsi`), with a `groups` argument to sum it per cluster:

```
R> range(hatvalues(fm))

[1] 0.1193 0.3925
```

4. A high-breakdown initial estimator: RANSAC

By default the RSE estimator uses a monotone (smoothed Huber) ψ-function, which is convex and largely insensitive to the starting values. For stronger robustness against extreme outliers one can instead choose a *redescending* ψ-function—for example `rho.e = lqqPsi`—whose

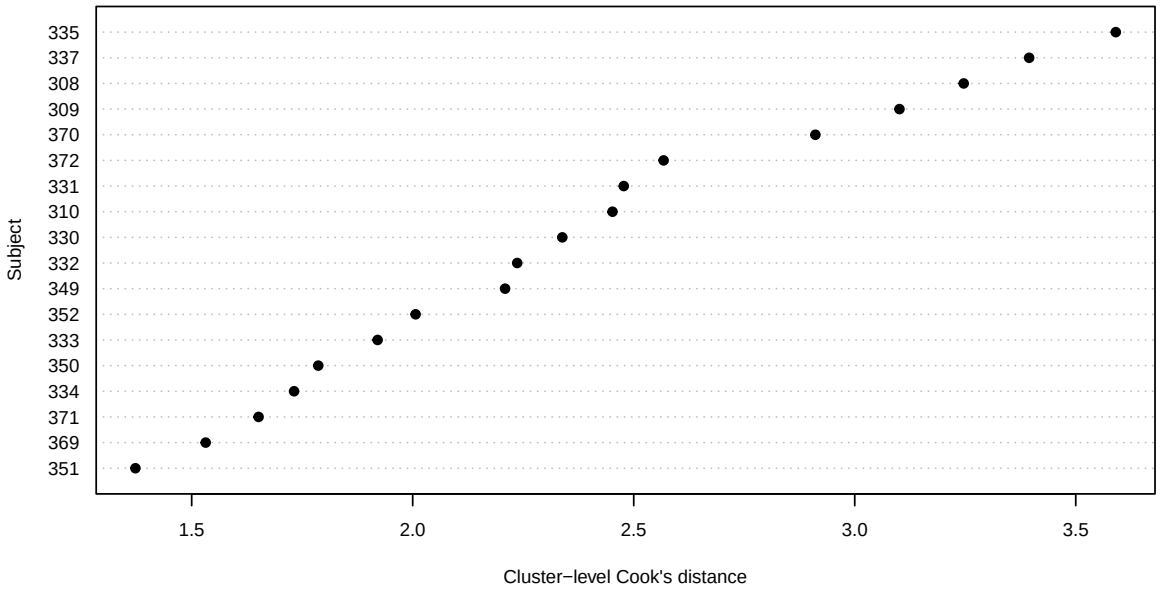


Figure 1: Per-subject (cluster-level) Cook's distances for the robust `sleepstudy` fit. Subjects at the top are the most influential on the joint parameter vector $(\hat{\beta}, \hat{\sigma}, \hat{\theta})$.

influence function returns to zero so that gross outliers are rejected outright. The bisquare (`rho.e = bisquarePsi`) is the classical redescender, but it redescends comparatively fast; the lqq (linear-quadratic-quadratic) psi of Koller and Stahel (2011), the recommended redescender used by **robustbase**'s `lmrob.control(setting = "KS2014")`, is generally preferable and is pre-built as `lqqPsi` (the general constructor `makeRobustbasePsi` additionally exposes the `optimal`, `hampel` and `ggw` families). Redescending M-estimators are more robust, but their estimating equations are non-convex and can be drawn to a bad local solution from a poor start. Release 3.4-5 added a RANSAC (RAndom SAmple Consensus, Fischler and Bolles 1981) high-breakdown start that makes such fits reliable; 3.5.0 refines it with the adaptive early stopping, stratified and nonsingular subsampling, and multi-start consensus described below. The string form `init = "ransac"` plugs it straight into `rlmer`:

```
R> fr <- rlmer(Reaction ~ Days + (Days | Subject), sleepstudy,
+              method = "DASvar", init = "ransac")
R> fixef(fr)

(Intercept)        Days
     251.27       10.64
```

`ransac_lme4` exposes the start directly and returns the fitted subsample, the consensus residual scale and diagnostics. The search is adaptive: it stops early once the best Q_n score plateaus, so `K` is a maximum budget rather than a fixed number of iterations (`adaptive = FALSE` restores the fixed loop). Subsampling is stratified by default (`stratify = TRUE`): each subsample covers every level of the finest grouping factor, keeping the per-subsample

`lmer` fittable on designs with many small clusters. Following the nonsingular subsampling of Koller and Stahel (2017) (the `setting = "KS2014"` default of **robustbase**), each draw is full-rank *by construction*: with categorical predictors a rank-deficient subsample does not always error—`lmer` would silently drop the aliased columns of a dropped factor level—so, rather than draw-then-check, an LU decomposition with column pivoting walks a random permutation of the observations and keeps only those that raise the fixed-effects rank, skipping any that are collinear with those already chosen. Whole clusters are then added until the retained data are a fully identifiable mixed-model subsample (full-rank fixed design, at least two levels per grouping factor, varying random slopes), which succeeds whenever the full design is identifiable. The collinear observations skipped by the LU are counted in `n_singular`; on a design without collinearity the draw reduces to a uniform random cluster subsample.

```
R> rs <- ransac_lme4(Reaction ~ Days + (Days | Subject), sleepstudy,
+                    K = 20L, seed = 20260601)
R> rs$K_used
```

```
[1] 20
```

`rlmer_ransac` wraps the start and the robust fit and runs three diagnostics: a fixed-effect *basin* check (warns when a redescending `rho.e` left the half-weight radius $r^* = s_{1/2}\,\sigma / \max_j \|x_j\|$ of the start, where $s_{1/2}$ is the residual mean-shift at which the expected redescender weight $m(s) = \mathrm{E}\,\psi'(Z + s)$, $Z \sim N(0,1)$, has halved — $s_{1/2} = 1.410$ for the bisquare, computed from the `Dpsi` slot for lqq and the other redescenders — so that the population Hessian stays positive definite on the ball; returned by `ransac_basin_radius`), a random-effects phony-correlation check (warns when the fitted RE covariance is near-singular, $|\hat{\rho}| \to 1$), and a refinement-singularity check (Koller and Stahel 2017, Remark 2): a redescending ψ can zero-weight observations back into a rank-deficient design during the robust fit, so `rlmer_ransac` re-seeds and re-fits from a fresh nonsingular start until the positive-weight fixed-effects design is full rank, up to `max_tries` (default 5), warning only if every attempt still collapses. The multi-start consensus `rlmer_ransac(n_starts = m)` fits from the m best distinct starts and returns the lowest-residual-scale fit whose RE covariance is interior, recovering the good solution when the single best start falls into the phony-correlation attractor (per-start summary in `attr(fit, "consensus")`):

```
R> fc <- rlmer_ransac(Reaction ~ Days + (Days | Subject), sleepstudy,
+                     K = 20L, n_starts = 3L, seed = 20260601)
R> attr(fc, "consensus")
```

```
  start max_abscor resid_scale interior chosen
1     1    0.05074        16.1     TRUE   TRUE
2     2    0.05074        16.1     TRUE  FALSE
3     3    0.05074        16.1     TRUE  FALSE
```

The per-start summary reports each start's largest absolute random-effect correlation (`max_abscor`), its residual scale, whether its random-effects covariance is interior (not a phony $|\hat{\rho}| \to 1$ solution), and which start was `chosen` (the interior fit with the lowest residual scale).

What the RANSAC start is for here is raising the breakdown point of the starting value, and thereby making redescending ψ-functions usable on general designs: their estimating equations are non-convex, so a low-breakdown start can be drawn to a bad local solution.

The defaults are settled and simulation-backed. Across the contamination scenarios of `inst/simulationStudy` the optional multi-start consensus (`n_starts > 1`) lowers the pooled phony-correlation rate from 0.183 to 0.170 at no cost in slope accuracy (slope RMSE 1.91 and 1.92); over 1200 paired fits the consensus is the interior one in all 16 replicates where the two disagree. RANSAC handles a single, crossed or nested grouping structure—subsampling is stratified by the finest grouping factor so each subsample `lmer` stays fittable on designs with many small clusters—and the start is fully reproducible from a `seed` (or, for the `init = "ransac"` string form, from an outer `set.seed`).

5. Robustness in the design space

5.1. Mallows-type design weights

The ψ-functions of the RSE bound the influence of outliers in the *response*, but a single high-leverage observation—extreme covariates with a plausible response—can still move $\hat{\beta}$ arbitrarily. This is the classical gap between Huber-type and generalised M-estimators. The `design.weights` argument adds Mallows-type weights $\eta_i \in (0, 1]$ that downweight the score contribution of high-leverage design points. The weights enter the e-side of every estimating equation $(\beta, u, \sigma, \theta)$, the model `vcov` and the influence diagnostics. Passing `"mcd"` derives η from robust Mahalanobis distances of the non-constant columns of X (via `covMcd`; note that with many binary dummy columns the MCD covariance estimate can be ill-conditioned); a numeric vector may be supplied instead.

```
R> dat <- sleepstudy
R> dat$Days[1] <- 40                  # an extreme, high-leverage design point
R> fw <- rlmer(Reaction ~ Days + (Days | Subject), dat,
+              method = "DASvar", design.weights = "mcd")
R> sum(getME(fw, "design.weights") < 1)

[1] 1
```

`summary` reports how many observations are downweighted by design. On clean data the cost is small—about a 0.8% increase in the variance of $\hat{\beta}$ relative to the plain RSE at the default tuning ($\gamma = 1$, cutoff 0.975), over 2000 replicates—and it vanishes for intercept-only designs (no non-constant columns in X). Under leverage contamination that drags ordinary and plain robust fits toward a spurious slope, the Mallows fit instead stays close to the clean-data estimate;

the efficiency–robustness trade-off is quantified in `inst/simulationStudy/mallowsEfficiency.R` and `leverageBreakdown.R`. Active design weights are supported for a single grouping factor; `rlmer` stops with an error on models with more than one.

5.2. Structured random-effect covariances (experimental)

With **lme4** $\geq$ 2.0-0, `rlmer` supports structured random-effect covariances: `diag(f | g)` (uncorrelated slopes, equivalent to `(f || g)`), `cs(f | g)` (compound symmetric: one common correlation) and `ar1(f | g)` (autoregressive, $\mathrm{Cor}(i,j) = \rho^{|i-j|}$), in both heterogeneous and homogeneous (equal-variance) forms. `diag` is fitted directly; `cs` and `ar1` are fitted by projecting each block's unstructured scoring-equation update onto the structured manifold at every iteration, so the structure is enforced exactly while the robustness machinery is unchanged. This projection is a pragmatic device rather than a fully established estimator, so `cs`/`ar1` fitting is *experimental*. It is validated only for a single grouping factor (balanced repeated measures); `rlmer` warns when a `cs`/`ar1` term is combined with more than one grouping factor, where neither the fit nor its inference is validated. Inference degrades gracefully on structured fits—the sandwich `vcov` is available, and `summary(df = "satterthwaite")` reports the robust df where the constrained covariance admits it (e.g. `cs`) and otherwise falls back to the t-table with a note.

To show the structure being enforced we simulate a balanced repeated-measures dataset with **robustlmm**'s own data generator `generateRepeatedMeasuresDatasets`: a four-level within-subject factor `visit`, replicated over 40 subjects, with an underlying AR(1) random-effect covariance ($\sigma = 1.5$, $\rho = 0.6$).

```
R> set.seed(20260623)
R> datasets <- generateRepeatedMeasuresDatasets(
+      1, numberOfSubjects = 40, numberOfVisits = 4, numberOfReplicates = 3,
+      structure = "ar1", marginalSd = 1.5, correlation = 0.6,
+      trueBeta = 10, trueSigma = 0.7)
R> d <- datasets$generateData(1)
```

We model the factor with `0 + visit` (dropping the intercept) so that *each* visit gets its own random effect: the term `(0 + visit | subject)` gives a four-dimensional random effect per subject, and that is the object on which a covariance structure is defined. An intercept-plus-slope parameterisation would instead give only a two-dimensional random effect, on which `cs` and `ar1` both reduce to a single correlation and so coincide with the unstructured fit. With four dimensions the structures differ: an unstructured fit estimates all $\binom{4}{2} = 6$ correlations, the `cs` fit collapses them to one common correlation, and the `ar1` fit to a single decay parameter ρ with $\mathrm{Cor}(i,j) = \rho^{|i-j|}$. `VarCorr` annotates the structure it enforced:

```
R> rcs  <- rlmer(y ~ 1 + cs(0 + visit | subject),  d, method = "DASvar")
R> rar1 <- rlmer(y ~ 1 + ar1(0 + visit | subject), d, method = "DASvar")
R> VarCorr(rcs)
```

```
 Groups   Name    Std.Dev. Corr
 subject  visitv1 1.265    0.45 (cs)
          visitv2 1.524
          visitv3 1.409
          visitv4 1.847
 Residual         0.653
```

```
R> VarCorr(rar1)
```

```
 Groups   Name    Std.Dev. Corr
 subject  visitv1 1.536    0.60 (ar1)
 Residual         0.651
```

Because the projection keeps the marginal variances and only constrains the correlations, a structured fit resists the over-parameterisation (and the spurious near-± 1 correlations) that an unstructured fit can fall into on short, noisy series, while remaining fully robust to outliers. A simulation study (`inst/simulationStudy/structuredCovariances.R`, 500 replicates) confirms both properties. On clean data the robust and the classical structured fits both recover the truth. But when 10% of the subjects carry an outlying random-effect vector, the classical (`lmer`) common correlation collapses toward zero—from a true 0.50 to 0.11 for `cs` and from 0.60 to 0.12 for `ar1`, with its marginal standard deviations inflating from 1.57 to 3.23—whereas the robust structured fit holds near the truth (correlation 0.42 for `cs` and 0.56 for `ar1`, marginal standard deviation 1.73).

A limitation of the default estimation method carries over to structured blocks: `method = "DAStau"` computes its consistency factors by Gauss–Hermite quadrature only for random-effect blocks of dimension ≤ 2, so for larger blocks—such as the four-dimensional `cs`/`ar1` blocks above—the fit falls back to `DASvar` with a warning (which is why the fits above request `DASvar` directly). The experimental option `options(robustlmm.dastau.mc = TRUE)` enables a Monte-Carlo calibration of the same DAS-tau fixed point for such blocks; the sample is drawn once per fit from a deterministic seed, so fits remain reproducible. In a companion simulation on clean Gaussian data this removes the small calibration residual that the `DASvar` approximation leaves in the fitted correlation: for `ar1` blocks of dimension 4 with 200 subjects the mean difference to `lmer` was at most 0.004 in absolute value with `DASvar` and 0.001 with the Monte-Carlo calibration (`inst/simulationStudy/dasmcValidation.R`). It is validated by simulation only—not by a finite-sample theorem—and for blocks of dimension ≤ 2 the quadrature path remains in place, as the Monte-Carlo path offers no improvement there. See `?"robustlmm-options"` for the option's companions and caveats.

6. A robust sensitivity analysis for an MMRM

In clinical trials with repeated measurements the primary analysis is often a mixed model for repeated measures (MMRM): treatment, visit and their interaction as fixed effects, and an

unstructured covariance of the visits within a subject. A robust fit can serve as a sensitivity analysis next to it: if a few unusual observations or subjects drive the result, the robust estimate moves away from the classical one, and the robustness weights show where. This section goes through such an analysis with the example data `fev_data` of the **mmrm** package (Sabanes Bove, Li, Dedic, Kelkhoff, Kunzmann *et al.* 2026): 537 lung-function measurements (`FEV1`) of 197 subjects at four visits, in a treatment and a placebo arm. It uses `augment()` and `contrast_test()`, which are new in version 3.5.1. The code is shown but not run here, so that the vignette does not need **mmrm**; the results were computed with **mmrm** 0.3.19 by `vignettes/precompute-mmrm.R`.

6.1. The models differ

The MMRM specifies the marginal covariance of the four visits directly and leaves it unstructured:

```
R> library("mmrm")
R> fitMmrm <- mmrm(FEV1 ~ RACE + SEX + ARMCD * AVISIT + us(AVISIT | USUBJID),
+                  data = fev_data)
```

`rlmer` cannot fit this model. Its covariance comes from random effects plus an independent residual error, and with one observation per subject and visit a visit-level random effect cannot be separated from the residual error. An unstructured block over the visits is therefore not identifiable, and the structured alternative `cs(0 + AVISIT | USUBJID)` is refused for the same reason (`rlmer` asks for more than four observations per subject). What can be fitted are random-effects models for the subject: a random intercept, which implies the same variance at every visit and the same correlation between any two, and a random intercept and slope in the visit number, which lets the variance change smoothly over the visits.

```
R> fitRob <- rlmer(FEV1 ~ RACE + SEX + ARMCD * AVISIT + (1 | USUBJID),
+                  data = fev_data)
R> fev_data$visit <- as.numeric(fev_data$AVISIT) - 2.5
R> fitRobSlope <- rlmer(FEV1 ~ RACE + SEX + ARMCD * AVISIT +
+                           (visit | USUBJID),
+                       data = fev_data, method = "DASvar")
```

The robust analysis is thus a check under a different, more restrictive covariance model, not a robust version of the MMRM.

6.2. Treatment difference at the last visit

For the MMRM and the random-intercept fit the treatment difference at the last visit comes from **emmeans**, with Satterthwaite degrees of freedom in both cases:

```
R> library("emmeans")
R> contrast(emmeans(fitMmrm, ~ ARMCD | AVISIT), "revpairwise")
R> contrast(emmeans(fitRob, ~ ARMCD | AVISIT), "revpairwise")
```

The same contrast can be written in terms of the coefficients and tested with `contrast_test()`, which also offers the cluster sandwich:

```
R> L <- c("ARMCDTRT" = 1, "ARMCDTRT:AVISITVIS4" = 1)
R> contrast_test(fitRob, L)
R> contrast_test(fitRob, L, vcov_type = "sandwich")
```

Fit	Estimate	SE	df	p
MMRM, unstructured covariance	4.40	1.68	133	0.010
`lmer`, random intercept	4.22	1.15	519	<0.001
`rlmer`, random intercept	3.44	1.11	348	0.002
with the cluster sandwich	3.44	1.91	83	0.075
`rlmer`, random intercept and slope	3.77	1.19	–	0.001
with the cluster sandwich	3.77	1.72	115	0.030

Table 1: Treatment difference in FEV1 at the last visit. The standard errors without the sandwich are model-based; "–" is the normal reference.

Table 1 shows the results. The robust estimates, 3.44 and 3.77, are somewhat smaller than the MMRM's 4.40, by less than one of its standard errors. The standard errors differ more than the estimates. The model-based standard errors of the random-effects fits, classical and robust, are about 1.1, against 1.7 for the MMRM: they rely on a common variance for all visits, while the MMRM estimates a standard deviation of 9.8 at the last visit and 3.9 at the third. The cluster sandwich does not rely on the covariance model, and with it the robust fits give standard errors of 1.9 and 1.7, close to the MMRM's; the p-value of the random-intercept fit is then 0.075. When the covariance model of the robust fit is cruder than the primary model's, as here, the sandwich is the standard error to compare.

6.3. Where the fits disagree

The robustness weights come with the data from `augment()`:

```
R> aug <- generics::augment(fitRob)
R> aug[order(aug$.wt_e)[1:6], ]
R> table(aug$AVISIT[aug$.wt_e < 0.8])
```

In the random-intercept fit, 11 observations have a weight below 0.5, and of the 58 observations with a weight below 0.8, 36 are from the last visit, which has a quarter of the observations. The downweighted observations are thus not scattered recording errors: they sit where the response is more variable than the model allows. The same shows in the variance components. The robust random-intercept fit estimates a subject standard deviation of 0.0 (`lmer`: 2.7), a boundary fit that says the common-correlation model does not describe these data. With the random slope the picture is calmer: 8 observations below 0.5, and 19 of 32 below 0.8 at the last visit. Here the weights point to a shortcoming of the model, the unequal variances of the visits, which the unstructured MMRM accommodates by construction.

The subjects that drive a fit are found with the cluster-level Cook's distance of Section 3.2:

```
R> cd <- cooks.distance(fitRobSlope, groups = TRUE)
R> head(sort(cd, decreasing = TRUE))
```

In the random-slope fit the largest value is 7.5 (subject PT138) against a median of 3.5, and the next ones follow closely: no single subject dominates. The subject with the smallest random-effect weight is PT198 (0.50); the number of subjects with a weight below one is 1. This computation needs the influence function of the fit and takes several minutes for a model of this size; `rlmer` announces it with a message.

6.4. What this analysis does and does not show

For these data the sensitivity analysis gives no sign that outliers drive the primary result: the robust estimates are within one standard error of the MMRM's, no subject dominates, and the downweighted observations are explained by the larger variance at the last visit. Three limits should be stated with any such analysis.

- The models differ. The MMRM has an unstructured marginal covariance of the visits; `rlmer` has random effects and an independent error. A difference between the two fits can come from the covariance model as well as from outliers.
- The MMRM with an unstructured covariance over the visits cannot be fitted by `rlmer`, because that block is not identifiable next to the residual error.
- Missing visits: the MMRM is valid when the dropout is missing at random. Whether the robust estimating equations are, has not been studied. With dropout that depends on the observed responses the robust fit should be read as descriptive.

7. Summary

Release 3.5.0 turns **robustlmm** from an estimator into a more complete robust-modelling toolbox: robust standard errors and confidence intervals, ANOVA tables, prediction intervals and honest degrees of freedom for inference; observation- and cluster-level influence diagnostics; a high-breakdown RANSAC start; and design weights and structured covariances for robustness in the design and covariance space. For the underlying estimator and its tuning, see the main vignette (Koller 2016); the simulation studies that validate these features live in `inst/simulationStudy`.

References

Bell RM, McCaffrey DF (2002). "Bias Reduction in Standard Errors for Linear Regression with Multi-Stage Samples." *Survey Methodology*, **28**(2), 169–181.

Cameron AC, Gelbach JB, Miller DL (2011). "Robust Inference with Multiway Clustering." *Journal of Business & Economic Statistics*, **29**(2), 238–249. doi:10.1198/jbes.2010.07136.

Cook RD, Weisberg S (1982). *Residuals and Influence in Regression*. Chapman and Hall, New York.

Fischler MA, Bolles RC (1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." *Communications of the ACM*, **24**(6), 381–395. doi:10.1145/358669.358692.

Koller M (2016). "**robustlmm**: An R Package for Robust Estimation of Linear Mixed-Effects Models." *Journal of Statistical Software*, **75**(6), 1–24. doi:10.18637/jss.v075.i06.

Koller M, Stahel WA (2011). "Sharpening Wald-Type Inference in Robust Regression for Small Samples." *Computational Statistics & Data Analysis*, **55**(8), 2504–2515. doi:10.1016/j.csda.2011.02.014.

Koller M, Stahel WA (2017). "Nonsingular Subsampling for Regression S Estimators with Categorical Predictors." *Computational Statistics*, **32**(2), 631–646. doi:10.1007/s00180-016-0679-x.

Kuznetsova A, Brockhoff PB, Christensen RHB (2017). "lmerTest Package: Tests in Linear Mixed Effects Models." *Journal of Statistical Software*, **82**(13), 1–26. doi:10.18637/jss.v082.i13.

Mason F, Cantoni E, Ghisletta P (2021). "Parametric and Semi-Parametric Bootstrap-Based Confidence Intervals for Robust Linear Mixed Models." *Methodology*, **17**(4), 271–295. doi:10.5964/meth.6607.

Mason F, Cantoni E, Ghisletta P (2024). "Linear Mixed Models and Latent Growth Curve Models for Group Comparison Studies Contaminated by Outliers." *Psychological Methods*, **30**(1), 155–173. doi:10.1037/met0000643.

Pustejovsky JE, Tipton E (2018). "Small-Sample Methods for Cluster-Robust Variance Estimation and Hypothesis Testing in Fixed Effects Models." *Journal of Business & Economic Statistics*, **36**(4), 672–683. doi:10.1080/07350015.2016.1247004.

Sabanes Bove D, Li L, Dedic J, Kelkhoff D, Kunzmann K, *et al.* (2026). *mmrm: Mixed Models for Repeated Measures*. R package version 0.3.19, URL https://CRAN.R-project.org/package=mmrm.

Satterthwaite FE (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin*, **2**(6), 110–114. doi:10.2307/3002019.

Affiliation:

Manuel Koller
E-mail: kollerma@proton.me
URL: https://github.com/kollerma/robustlmm