Package: mlmoderator (via r-universe)

June 2, 2026

Type Package

Title Probing, Plotting, and Interpreting Multilevel Interaction Effects

Version 0.2.1

Description Provides a unified workflow for probing, plotting, and assessing the robustness of cross-level interaction effects in two-level mixed-effects models fitted with 'lme4' (Bates et al., 2015) <doi:10.18637/jss.v067.i01>. Implements simple slopes analysis following Aiken and West (1991, ISBN:9780761907121), Johnson-Neyman intervals following Johnson and Fay (1950) <doi:10.1007/BF02288864> and Bauer and Curran (2005) <doi:10.1207/s15327906mbr4003_5>, and grand- or group-mean centering as described in Enders and Tofighi (2007) <doi:10.1037/1082-989X.12.2.121>. Includes a slope variance decomposition that separates fixed-effect uncertainty from random-slope variance (tau11), a contour surface plot of predicted outcomes over the full predictor-by-moderator space, and robustness diagnostics comprising intraclass correlation coefficient shift analysis and leave-one-cluster-out (LOCO) stability checks. Designed for researchers in education, psychology, biostatistics, epidemiology, organizational science, and other fields where outcomes are clustered within higher-level units.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports lme4, ggplot2, stats, utils, grDevices, rlang

Suggests testthat (>= 3.0.0), dplyr, tibble, knitr, rmarkdown, patchwork

Config/testthat/edition 3

VignetteBuilder knitr

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

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

Config/pak/sysreqs cmake make

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

Date/Publication 2026-04-03 22:46:26 UTC

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

RemoteRef HEAD

RemoteSha d082a04529e0b078cc29fb635e8ce5f749b9a3a4

Contents

`mlm_center` *Center variables for multilevel modeling*

Description

Performs grand-mean centering, group-mean centering, or both (within-between decomposition) on one or more variables in a data frame. Group-mean centering is the standard preparation for cross-level interaction models.

Usage

```
mlm_center(
  data,
  vars,
  cluster = NULL,
  type = c("grand", "group", "both"),
  suffix_within = "_within",
  suffix_between = "_between"
)
```

Arguments

`data`	A data frame.
`vars`	Character vector of variable names to center.
`cluster`	Character scalar: name of the clustering variable (required when `type` is `"group"` or `"both"`).
`type`	One of `"grand"`, `"group"`, or `"both"`. • `"grand"`: subtract the overall mean. • `"group"`: subtract the cluster mean (within-person / within-school centering). • `"both"`: return both the within-cluster-centered value *and* the cluster mean (between component), appended as new columns.
`suffix_within`	Suffix appended to within-centered variable names when `type = "both"`. Default is `"_within"`.
`suffix_between`	Suffix appended to between (cluster mean) variable names when `type = "both"`. Default is `"_between"`.

Value

The input data frame with new centered columns appended. Original columns are not modified.

Examples

```
data(school_data)

# Grand-mean center SES
d1 <- mlm_center(school_data, vars = "ses", type = "grand")
head(d1[, c("ses", "ses_c")])

# Group-mean center SES within schools
d2 <- mlm_center(school_data, vars = "ses", cluster = "school", type = "group")
head(d2[, c("ses", "ses_c")])

# Within-between decomposition
d3 <- mlm_center(school_data, vars = "ses", cluster = "school", type = "both")
head(d3[, c("ses", "ses_within", "ses_between")])
```

`mlm_jn` *Johnson—Neyman interval for multilevel two-way interactions*

Description

Computes the Johnson—Neyman (JN) interval: the region(s) of the moderator (`modx`) where the simple slope of `pred` transitions between statistical significance and non-significance. Useful for identifying *exactly* at which moderator values an effect becomes significant.

Usage

```
mlm_jn(model, pred, modx, alpha = 0.05, modx.range = NULL, grid = 200L)
```

Arguments

`model`	An `lmerMod` object with a two-way interaction between `pred` and `modx` in the fixed-effects structure.
`pred`	Character scalar. Focal predictor name.
`modx`	Character scalar. Moderator name.
`alpha`	Significance level. Default `0.05`.
`modx.range`	Numeric vector of length 2 giving the range over which to evaluate the slope. Defaults to the observed range of `modx`.
`grid`	Integer. Number of points at which to evaluate the slope across the moderator range. Default `200`.

Value

An object of class `mlm_jn` with components:

- `jn_bounds`: numeric vector of moderator values where the slope crosses the significance threshold. `NA` if no finite region exists.
- `slopes_df`: data frame of slope estimates and significance across the grid.
- `pred`, `modx`, `alpha`, `modx.range`.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
jn <- mlm_jn(mod, pred = "x", modx = "m")
print(jn)
```

`mlm_plot` *Publication-ready interaction plot for multilevel models*

Description

Creates a `ggplot2`-based interaction plot showing predicted values of the outcome across levels of `pred`, with separate lines for each selected value of `modx`. Confidence bands and raw data overlay are optional.

Usage

```
mlm_plot(
  model,
  pred,
  modx,
  modx.values = c("mean-sd", "quartiles", "tertiles", "custom"),
  at = NULL,
  interval = TRUE,
  conf.level = 0.95,
  points = FALSE,
  point_alpha = 0.3,
  colors = NULL,
  line_size = 1,
  x_label = NULL,
  y_label = NULL,
  legend_title = NULL
)
```

Arguments

`model`	An `lmerMod` object with a two-way interaction between `pred` and `modx` in the fixed-effects.
`pred`	Character scalar. Focal predictor (x-axis).
`modx`	Character scalar. Moderator (separate lines).
`modx.values`	Strategy for moderator values. Same options as `mlm_probe()`: `"mean-sd"`, `"quartiles"`, `"tertiles"`, `"custom"`.
`at`	Numeric vector of custom moderator values (used when `modx.values = "custom"`).
`interval`	Logical. Draw confidence bands? Default `TRUE`.
`conf.level`	Confidence level for bands. Default `0.95`.
`points`	Logical. Overlay raw data points? Default `FALSE`.
`point_alpha`	Transparency for raw data points. Default `0.3`.
`colors`	Character vector of colours for moderator lines. If `NULL`, uses a accessible default palette.
`line_size`	Line width for predicted lines. Default `1`.

`x_label`	Label for x-axis. Defaults to `pred`.
`y_label`	Label for y-axis. Defaults to response variable name.
`legend_title`	Label for the legend. Defaults to `modx`.

Value

A `ggplot` object.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200),
  x   = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
mlm_plot(mod, pred = "x", modx = "m")
mlm_plot(mod, pred = "x", modx = "m", modx.values = "quartiles")
```

`mlm_probe` *Probe simple slopes from a multilevel interaction*

Description

Computes simple slopes of a focal predictor (`pred`) at selected values of a moderator (`modx`) from a two-level mixed-effects model fitted with `lme4::lmer()`. Returns estimates, standard errors, *t*-values, *p*-values, and confidence intervals in a tidy data frame.

Usage

```
mlm_probe(
  model,
  pred,
  modx,
  modx.values = c("mean-sd", "quartiles", "tertiles", "custom"),
  at = NULL,
  conf.level = 0.95
)
```

Arguments

`model`	An `lmerMod` object containing a two-way interaction between `pred` and `modx` in the fixed-effects structure.
`pred`	Character scalar. Name of the focal predictor variable.
`modx`	Character scalar. Name of the moderator variable.
`modx.values`	Strategy for selecting moderator values. One of: • `"mean-sd"` (default): mean — 1 SD, mean, mean + 1 SD. • `"quartiles"`: 25th, 50th, 75th percentiles. • `"tertiles"`: 33rd and 67th percentiles. • `"custom"`: use values supplied via `at`.
`at`	Numeric vector of custom moderator values. Used when `modx.values = "custom"`, or to override any strategy.
`conf.level`	Confidence level for intervals. Default `0.95`.

Value

An object of class `mlm_probe` (a list) with components:

- `slopes`: a data frame with columns `modx_value`, `slope`, `se`, `t`, `df`, `p`, `ci_lower`, `ci_upper`.
- `pred`, `modx`: names of the predictor and moderator.
- `modx.values`: the strategy used.
- `conf.level`: the confidence level.
- `model`: the original model (stored for downstream use).

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
mlm_probe(mod, pred = "x", modx = "m")
mlm_probe(mod, pred = "x", modx = "m", modx.values = "quartiles")
mlm_probe(mod, pred = "x", modx = "m", modx.values = "custom", at = c(-1, 0, 1))
```

`mlm_sensitivity` *Robustness diagnostics for cross-level interaction effects*

Description

Assesses the stability of a cross-level interaction effect using two MLM-appropriate diagnostics:

Usage

```
mlm_sensitivity(
  model,
  pred,
  modx,
  alpha = 0.05,
  icc_range = c(0.01, 0.4),
  icc_grid = 50L,
  loco = TRUE,
  conf.level = 0.95,
  verbose = FALSE
)
```

Arguments

`model`	An `lmerMod` object with a two-way interaction between `pred` and `modx`.
`pred`	Character scalar. Focal predictor name.
`modx`	Character scalar. Moderator name.
`alpha`	Significance level. Default `0.05`.
`icc_range`	Numeric vector of length 2. Range of ICC values to evaluate. Default `c(0.01, 0.40)`.
`icc_grid`	Integer. Number of ICC values in the grid. Default `50`.
`loco`	Logical. Run leave-one-cluster-out analysis? Default `TRUE`. Set to `FALSE` for large datasets where refitting is slow.
`conf.level`	Confidence level. Default `0.95`.
`verbose`	Logical. Print progress during LOCO refitting? Default `FALSE`.

Details

1. **ICC-shift robustness** – how do the interaction SE and Johnson-Neyman boundary change if the intraclass correlation were different from what was observed? This is relevant because the ICC determines the effective sample size at level 2, which directly governs precision of cross-level interaction estimates.
2. **Leave-one-cluster-out (LOCO) stability** – refit the model dropping one cluster at a time and track how the interaction coefficient moves. This is nonparametric, makes no distributional assumptions, and directly answers: "Is this finding driven by a small number of influential clusters?"

Value

An object of class `mlm_sensitivity` with components:

- `icc_shift`: data frame of interaction SE, t, p, significance, and approximate JN boundary across the ICC grid.
- `loco`: data frame with one row per cluster giving the interaction coefficient, SE, t, p, and Cook's-distance-style influence measure when that cluster is omitted. `NULL` if `loco` = `FALSE`.
- `robustness_index`: proportion of the ICC range where the interaction remains significant.
- `observed`: list of observed model statistics.
- Metadata: `pred`, `modx`, `alpha`, `icc_range`, `int_term`.

Scope

These are **robustness diagnostics**, not a full causal sensitivity analysis. They do not quantify the strength of unmeasured confounding needed to explain away the interaction – that requires a level-2-aware omitted variable bound that is currently under development as a separate methodological contribution. See `vignette("robustness-diagnostics")` for interpretation guidance.

Examples

```
# Use a small dataset for fast execution
set.seed(42)
n_j <- 20; n_i <- 10
dat_small <- data.frame(
  y    = rnorm(n_j * n_i),
  x    = rnorm(n_j * n_i),
  m    = rep(rnorm(n_j), each = n_i),
  grp  = factor(rep(seq_len(n_j), each = n_i))
)
dat_small$y <- dat_small$y + 0.5 * dat_small$x * dat_small$m
mod_small <- lme4::lmer(y ~ x * m + (1 + x | grp), data = dat_small,
                        control = lme4::lmerControl(optimizer = "bobyqa"))

# ICC-shift only (fast)
sens <- mlm_sensitivity(mod_small, pred = "x", modx = "m", loco = FALSE)
print(sens)

# Full diagnostics including LOCO (20 clusters - fast)

sens_full <- mlm_sensitivity(mod_small, pred = "x", modx = "m")
plot(sens_full)
```

`mlm_summary` *Summary table for a multilevel moderation effect*

Description

Returns a consolidated summary of the moderation effect: the focal interaction coefficient, simple slopes at selected moderator values, and (optionally) the Johnson—Neyman interval. Designed for quick reporting and results sections.

Usage

```
mlm_summary(
  model,
  pred,
  modx,
  modx.values = c("mean-sd", "quartiles", "tertiles", "custom"),
  at = NULL,
  conf.level = 0.95,
  jn = TRUE,
  alpha = 0.05
)
```

Arguments

`model`	An `lmerMod` object with a two-way interaction between `pred` and `modx`.
`pred`	Character scalar. Focal predictor name.
`modx`	Character scalar. Moderator name.
`modx.values`	Moderator value strategy. See `mlm_probe()`.
`at`	Optional numeric vector of custom moderator values.
`conf.level`	Confidence level. Default `0.95`.
`jn`	Logical. Include Johnson-Neyman region? Default `TRUE`.
`alpha`	Alpha for JN interval. Default `0.05`.

Value

An object of class `mlm_summary` (a list) with components:

- `interaction`: one-row data frame for the interaction term.
- `simple_slopes`: data frame from `mlm_probe()`.
- `jn`: output of `mlm_jn()` (or `NULL` if `jn = FALSE`).
- Other metadata.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
mlm_summary(mod, pred = "x", modx = "m")
```

`mlm_surface` *Contour plot of predicted outcomes over the predictor x moderator space*

Description

Plots iso-outcome contour lines of $\hat{Y}(x, w)$ over the full joint space of `pred` (x-axis) and `modx` (y-axis). This is the most direct geometric representation of a two-way interaction:

Usage

```
mlm_surface(
  model,
  pred,
  modx,
  grid = 80L,
  n_contours = 10L,
  fill = TRUE,
  probe_lines = TRUE,
  x_label = NULL,
  y_label = NULL,
  legend_title = NULL
)
```

Arguments

`model`	An `lmerMod` object with a two-way interaction between `pred` and `modx` in the fixed effects.
`pred`	Character scalar. Focal predictor (x-axis).
`modx`	Character scalar. Moderator (y-axis).
`grid`	Integer. Grid resolution (points per axis). Default `80`.
`n_contours`	Integer. Number of contour levels to draw. Default `10`.
`fill`	Logical. Fill contour bands with colour? Default `TRUE`.

`probe_lines`	Logical. Overlay horizontal lines at mean – 1 SD of `modx`? Default `TRUE`.
`x_label`	x-axis label. Defaults to `pred`.
`y_label`	y-axis label. Defaults to `modx`.
`legend_title`	Legend title. Defaults to the outcome variable name.

Details

- **No interaction**: contour lines are perfectly straight and parallel — the effect of `pred` does not depend on `modx`.
- **Positive interaction**: contour lines fan outward (rotate clockwise) — higher `modx` steepens the `pred` slope.
- **Negative interaction**: contour lines fan inward (rotate counter-clockwise).

The degree of non-parallelism among contours is a direct visual index of interaction strength: the larger β_3, the more the lines rotate.

An optional overlay draws the three standard simple-slope evaluation lines (mean – 1 SD of `modx`) as horizontal reference lines, connecting the plot to `mlm_probe()` output.

Predicted values are computed from fixed effects only (`re.form = NA`), with all covariates held at their means or reference levels. The surface therefore represents the population-average predicted outcome, not any specific cluster.

Reading the plot: Pick any contour line. Its slope in the `pred` direction tells you how fast the outcome changes with `pred` at that `modx` value. If the contour slopes steeply up-right, `pred` has a strong positive effect there. If contours become more horizontal as `modx` increases, the `pred` effect is weakening. If they rotate from positive to flat to negative, you have a sign-changing interaction — and the `modx` value where they are perfectly horizontal is the Johnson-Neyman boundary.

Value

A `ggplot` object.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
mlm_surface(mod, pred = "x", modx = "m")
mlm_surface(mod, pred = "x", modx = "m", fill = FALSE, n_contours = 15)
```

`mlm_variance_decomp` *Decompose uncertainty in the simple slope of a multilevel interaction*

Description

In a random-slope model, the uncertainty around a simple slope has two distinct sources that standard `mlm_probe()` collapses into one SE:

Usage

```
mlm_variance_decomp(
  model,
  pred,
  modx,
  modx.values = c("mean-sd", "quartiles", "tertiles", "custom"),
  at = NULL,
  conf.level = 0.95
)
```

Arguments

`model`	An `lmerMod` object with a random slope for `pred` and a two-way interaction between `pred` and `modx`.
`pred`	Character scalar. Focal predictor name.
`modx`	Character scalar. Moderator name.
`modx.values`	Strategy for moderator values. See `mlm_probe()`.
`at`	Optional numeric vector of custom moderator values.
`conf.level`	Confidence level for fixed-effect CIs. Default `0.95`.

Details

1. **Fixed-effect uncertainty** – imprecision in the estimated average slope ($\beta_1 + \beta_3 \cdot w$), captured by the fixed-effect variance-covariance matrix.
2. **Random-slope variance** – genuine between-cluster heterogeneity in the slope of `pred` (τ_{11}), which is *not* estimation error but real variation in effects across clusters.

These answer different questions:

- Fixed-effect uncertainty: "How precisely do we know the *average* slope at this moderator value?"
- Random-slope variance: "How much does the slope *actually vary* across clusters, regardless of what the moderator does?"

The function also reports a **prediction interval** for the slope in a new (unobserved) cluster, which combines both sources and is the appropriate uncertainty interval for making cluster-level predictions.

Random-slope variance (τ_{11}) interpretation: If `tau11` is large relative to the fixed slope, the effect of `pred` varies substantially across clusters even *after* accounting for the moderator. This is important: a significant interaction does not mean the moderator fully explains between-cluster slope heterogeneity.

Prediction interval interpretation: The prediction interval answers: "For a randomly sampled new cluster at this moderator value, what range of slopes should we expect?" It will always be wider than the confidence interval because it incorporates τ_{11}.

Percentage of variance from random effects: $\%\text{random} = \frac{\tau_{11}}{\tau_{11}+\text{Var}(\hat{\beta}_{\text{simple slope}})} \times 100$

Value

An object of class `mlm_variance_decomp` (a list) with:

- `decomp`: data frame with columns `modx_value`, `slope`, `se_fixed` (SE from fixed-effect vcov only), `tau11` (random-slope SD, if estimable), `se_total` (combined SE for prediction in new cluster), `ci_lower`, `ci_upper` (fixed-effect CI), `pi_lower`, `pi_upper` (prediction interval for new cluster), `pct_random` (% of total slope variance from random effects).
- `tau11`: the random-slope variance (τ_{11}).
- `has_random_slope`: logical – does the model include a random slope for `pred`?
- Metadata: `pred`, `modx`, `conf.level`.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 + x | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
vd <- mlm_variance_decomp(mod, pred = "x", modx = "m")
print(vd)
```

`plot.mlm_jn` *Plot a Johnson-Neyman interval from a multilevel model*

Description

Creates a ggplot2 figure showing the simple slope of `pred` across the full range of `modx`, with shading indicating regions of significance. A vertical dashed line marks each Johnson-Neyman boundary.

Usage

```
## S3 method for class 'mlm_jn'
plot(
  x,
  x_label = NULL,
  y_label = NULL,
  sig_color = "#2166AC",
  nonsig_color = "#D6604D",
  ...
)
```

Arguments

`x`	An `mlm_jn` object from `mlm_jn()`.
`x_label`	Label for the x-axis. Defaults to the moderator name.
`y_label`	Label for the y-axis. Defaults to "Simple slope of pred".
`sig_color`	Fill colour for the significant region. Default `"#2166AC"`.
`nonsig_color`	Fill colour for the non-significant region. Default `"#D6604D"`.
`...`	Ignored.

Value

A `ggplot` object.

Examples

```
set.seed(1)
dat <- data.frame(
  y   = rnorm(200), x = rnorm(200),
  m   = rep(rnorm(20), each = 10),
  grp = factor(rep(1:20, each = 10))
)
dat$y <- dat$y + dat$x * dat$m
mod <- lme4::lmer(y ~ x * m + (1 | grp), data = dat,
                  control = lme4::lmerControl(optimizer = "bobyqa"))
jn <- mlm_jn(mod, pred = "x", modx = "m")
plot(jn)
```

`plot.mlm_sensitivity` *Plot robustness diagnostics for a cross-level interaction*

Description

Produces up to three panels: (1) interaction SE across the ICC range, (2) JN boundary shift across the ICC range, and (3) LOCO coefficient stability plot showing the interaction estimate when each cluster is omitted, with influential clusters flagged.

Usage

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

Arguments

`x`	An `mlm_sensitivity` object.
`...`	Ignored.

Value

A `ggplot` object.

`plot.mlm_variance_decomp` *Plot the variance decomposition of simple slopes*

Description

Shows the simple slope at each moderator value as a point with two interval layers: an inner confidence interval (fixed-effect uncertainty only) and an outer prediction interval (fixed + random-slope variance). The gap between the two intervals is the contribution of random-slope heterogeneity.

Usage

```
## S3 method for class 'mlm_variance_decomp'
plot(x, x_label = NULL, y_label = NULL, ...)
```

Arguments

`x`	An `mlm_variance_decomp` object.
`x_label`	x-axis label. Defaults to moderator name.
`y_label`	y-axis label. Defaults to "Simple slope of pred".
`...`	Ignored.

Value

A `ggplot` object.

`school_data` *Simulated school achievement dataset*

Description

A simulated two-level dataset with students nested within schools, designed to illustrate multilevel moderation analysis. The true data-generating model includes a cross-level interaction between student socioeconomic status and school climate.

Usage

```
school_data
```

Format

A data frame with 3,000 rows and 6 variables:

school A factor indicating the school identifier (1–100).

student An integer indicating the student identifier (1–3000).

math A numeric mathematics achievement score.

ses A numeric student socioeconomic status variable (standardized).

climate A numeric school climate rating (standardized level-2 variable).

gender A factor indicating student gender with levels `"female"` and `"male"`.

Details

The data were generated from the model

$$math_{ij} = 50 + 1.5\, ses_{ij} + 0.8\, climate_j + 0.5\, ses_{ij} climate_j + u_{0j} + u_{1j} ses_{ij} + e_{ij}.$$

The level-2 random effects were generated as $u_{0j} \sim N(0, 9)$ and $u_{1j} \sim N(0, 0.25)$, and the level-1 residuals were generated as $e_{ij} \sim N(0, 25)$.

Examples

```
data(school_data)
head(school_data)
str(school_data)


library(lme4)
mod <- lmer(math ~ ses * climate + gender + (1 + ses | school),
            data = school_data)
mlm_probe(mod, pred = "ses", modx = "climate")
```

Index