Package: MoPlot (via r-universe)

November 25, 2026

Title Visualize and Interpret Contrasts in Regression Models

Version 0.1.0

Description Visualizes and interprets the contrasts used to code categorical predictors in fitted regression models. Each coefficient is translated into the comparison between the levels, or groups of levels, that it estimates, derived from the hypothesis matrix of the contrast coding, and is displayed together with the coefficient, its confidence interval, and an approximate effect size. Interactions among categorical predictors are decomposed into differences of differences. Supports models fitted with 'stats', 'lme4', 'lmerTest', 'nlme', and 'glmmTMB'.

License MIT + file LICENSE

Encoding UTF-8

Imports cli, effectsize, ggplot2 (>= 4.0.0), glmmTMB, grid, lme4, lmerTest, MASS, nlme, parameters, patchwork, reformulas, tibble

Suggests hypr, knitr, rmarkdown, testthat (>= 3.0.0), vdiffr

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Umberto Granziol [aut, cre] (ORCID: <https://orcid.org/0000-0002-6286-6569>, affiliation: University of Padova), Stefano Dalla Bona [aut] (ORCID: <https://orcid.org/0009-0006-7749-3642>, affiliation: University of Padova), Lorenzo Atzeni [aut] (ORCID: <https://orcid.org/0009-0006-5307-9201>, affiliation: University of Padova), Marianna Musa [aut] (ORCID: <https://orcid.org/0009-0000-8791-1218>, affiliation: University of Padova)

Maintainer Umberto Granziol <umberto.granziol@unipd.it>

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

Date/Publication 2026-11-25 17:26:24 UTC

RemoteUrl https://github.com/cran/MoPlot

RemoteRef HEAD

RemoteSha 387a7ba92fbda0a1f28fa3d2a2ed6780b9f3d7ae

Contents

contrasts_multpred 2
contrasts_onepred 14
extract_param 25
model_contrast_table 29
moplot_export 30
moplot_multpred 32
moplot_onepred 38
reverse_helmert 41

Index **43**

`contrasts_multpred` *Decompose Interaction Contrasts into Interpretable Configurations*

Description

`contrasts_multpred()` decomposes model coefficients associated with interactions among categorical predictors into interpretable configurations of factor levels.

For each selected interaction coefficient, the function identifies the positive and negative level groups implied by the contrast matrices used to fit the model. It then returns observed means or model-based predictions for the resulting configurations.

One variable involved in each interaction is selected as the *split variable*. The contrasts of the remaining target variable or variables are then displayed within the levels or contrast-defined groups of the split variable.

The function supports two-way and higher-order interactions.

Usage

```
contrasts_multpred(
  model,
  data = NULL,
  interaction = NULL,
  coefficients = NULL,
  outcome = NULL,
  split_rule = c("last_variable", "first_variable", "lowest_complexity",
    "highest_complexity"),
  top_split = NULL,
  sig_level = 0.05,
```

```
  conf_level = 0.95,
  configuration = c("none", "all", "explicit", "semantic"),
  estimate_type = c("auto", "observed", "predicted"),
  prediction_scale = c("response", "link"),
  methodological_note = TRUE,
  bounds = NULL,
  hypotheses = NULL
)
```

Arguments

`model`	A fitted model object. Supported model classes include objects returned by `stats::lm()`, `stats::glm()`, `lme4::lmer()`, `lme4::glmer()`, `nlme::lme()`, and `glmmTMB::glmmTMB()`.
`data`	Optional data frame containing the variables used to fit `model`. If `NULL`, the data supplied in the model call are recovered, restricted to the rows used for fitting; the model frame is used only when they are unavailable. The original variables are needed for model-based estimates when the model contains transformed terms such as `log(x)`. Rows of a supplied data frame that were excluded from the fit, through `subset` or missing values, are dropped, matching rows by name.
`interaction`	Optional character string identifying the interaction term to decompose, for example `"A:B"` or `"A:B:C"`. When supplied, the interaction is resolved using the fitted model matrix and the coefficients associated with that interaction are selected automatically. If `coefficients` is also supplied, those coefficients are validated against the requested interaction. When both `interaction` and `coefficients` are `NULL`, all non-intercept coefficients containing `":"` are analyzed, preserving the default behavior of this function.
`coefficients`	Optional character vector containing the interaction coefficients to decompose. If `interaction` is `NULL` and `coefficients` is `NULL`, all non-intercept coefficients containing `":"` are analyzed. When `interaction` is supplied, `coefficients` may be used to select a subset of the coefficients belonging to that interaction.
`outcome`	Optional character string indicating the name of the dependent variable; by default, the dependent variable of `model`. It is used only for observed estimates, which summarize the response of the model evaluated in `data`: a transformed response such as `log(y)` is summarized on the scale of the model, and a logical response or a factor response of a binomial model is converted to 0/1 as in `stats::glm()`.
`split_rule`	Character string indicating how the split variable is selected for each interaction coefficient. Available options are: `"last_variable"` Selects the last variable appearing in the interaction coefficient. `"first_variable"` Selects the first variable appearing in the interaction coefficient. `"lowest_complexity"` Selects the variable with the lowest contrast-complexity score.

"highest_complexity" Selects the variable with the highest contrast-complexity score.

top_split Optional character string identifying a variable that must be used as the split variable. When supplied, top_split overrides split_rule.

sig_level Numeric value indicating the significance threshold used to classify the original interaction coefficients as significant or non-significant. The default is 0.05.

conf_level Numeric value indicating the confidence level used for confidence intervals. The default is 0.95.

configuration Character string controlling which configuration labels are included in the output. Available options are:

"none" Does not return configuration-label columns.

"all" Returns both explicit and semantic configuration labels.

"explicit" Returns the complete mathematical composition of each configuration.

"semantic" Returns a compact description of the levels or groups represented by each configuration.

estimate_type Character string indicating which estimates are returned. Available options are:

"auto" Uses observed means when every analyzed interaction involves all the predictors of the fixed part, the model has no random effects and no unequal prior weights, and, for generalized models, prediction_scale = "response". Model-based predictions are used otherwise, as in contrasts_onepred().

"observed" Returns observed aggregated means computed directly from data.

"predicted" Returns model-based predictions computed from the fixed-effect coefficients and their variance-covariance matrix.

prediction_scale

Character string indicating the scale on which model-based predictions are returned. Available options are:

"response" Returns predictions on the response scale. For generalized models, the inverse link function is applied.

"link" Returns predictions on the linear-predictor scale.

This argument is ignored when estimate_type = "observed".

methodological_note

Logical. Should a methodological note describing the settings relevant to the current analysis be printed? The default is TRUE. The note is stored as an attribute of the returned object regardless of this setting.

bounds Optional numeric vector c(lower, upper) giving the natural range of a response analyzed with a Gaussian (identity link) model, such as c(1, 7) for a Likert score or c(0, Inf) for a reaction time. When supplied, confidence intervals of observed and predicted estimates are computed on a transformed scale (logit or logarithm) and transformed back, so that they remain within the bounds. Ignored, with a warning, for generalized models. See contrasts_onepred().

hypotheses Optional named list declaring, for one or more categorical predictors, the comparisons that their contrast coefficients are intended to estimate, for example list(treatment = H_treatment, context = H_context). Each element is a

numeric matrix with one row per contrast column of that predictor and one column per level, as the rows of a hypothesis matrix (see `contrasts_onepred()`). Each declared predictor is checked against the comparisons implied by its coding, and a message reports the result for each contrast column. Because the hypothesis of an interaction coefficient is the Kronecker product of the hypotheses of its components, each interaction coefficient estimates the product of the declared hypotheses when all its declared components do, multiplied by the product of their factors. Predictors not listed are not checked. The model is not modified. The result is stored in the `hypothesis_check` attribute, with one row per analyzed coefficient.

Details

Each interaction coefficient is separated into its contrast-coded components. The function then identifies:

- the categorical predictor represented by each component;
- the corresponding contrast column;
- the factor levels receiving positive, negative, or zero weights.

One predictor is selected as the split variable. The remaining predictor or predictors are treated as target variables whose contrasts are interpreted within the levels or contrast-defined groups of the split variable.

For example, suppose that `FactorA` has three levels and uses Helmert contrasts, whereas `FactorB` has two levels and uses sliding-difference contrasts. If `FactorB` is selected as the split variable, the function displays the Helmert contrasts of `FactorA` separately within the levels or contrast-defined groups of `FactorB`.

Therefore, the split variable is not necessarily the variable of primary interpretive interest. It defines the conditions under which the contrasts of the remaining target variable or variables are displayed.

The split variable is selected separately for each interaction coefficient. Consequently, when all model interactions are analyzed simultaneously, more than one value may appear in the `split_var` column. For example, in a model containing `A:B`, `A:C`, `B:C`, and `A:B:C`, the variable selected by `"highest_complexity"` may differ across coefficients because only the variables involved in each coefficient are considered.

Random-effect grouping variables are excluded when identifying the categorical predictors involved in the fixed-effect interactions.

Value

A data frame of class `"plot_ready_contrasts_df"` with one row per decomposed configuration.

The returned columns include:

`coefficient` Name of the original interaction coefficient being decomposed.

`contrast_id` Sequential identifier assigned to the selected coefficient.

`significance` Significance classification of the original coefficient: `"Sig"`, `"NotSig"`, or `"NoPValue"`.

`p_value` P-value associated with the original interaction coefficient, when available.

`scale_factor` Scale factor such that the coefficient equals `scale_factor` multiplied by the displayed difference of differences on the linear-predictor scale.

`split_var` Split variable selected for the coefficient.

factor columns One column for each categorical predictor involved in the analyzed interactions. Values identify specific levels or aggregated groups.

`n` Number of observed rows underlying the displayed configuration.

`estimate` Observed aggregate or model-based aggregate.

`se` Standard error of the displayed estimate.

`ci_low` Lower confidence-interval boundary.

`ci_high` Upper confidence-interval boundary.

Depending on `configuration`, the output may also include:

`configuration_explicit` Mathematical description of the cells entering the aggregate.

`configuration_semantic` Compact interpretation of the specific and averaged levels.

The returned object contains the following attributes:

`note` Methodological description of the estimation and aggregation procedures.

`estimate_type` Whether the output contains `"observed"` or `"predicted"` estimates.

`prediction_scale` Prediction scale used for model-based estimates. This is `NA` for observed estimates.

`estimate_info` Human-readable description of the `estimate` column.

`pvalue_source` Source of the p-values, such as `"model"`, `"lmerTest"`, or `"not_available"`.

`split_rule` Rule used to select the split variable.

`top_split` Split variable explicitly requested through `top_split`, when supplied.

`scale_factors` Named vector of scale factors, one per coefficient.

`identity_holds` Named logical vector indicating, for each coefficient, whether the displayed comparisons reconstruct the coefficient on the linear-predictor scale.

Contrast complexity and split selection

The options `"lowest_complexity"` and `"highest_complexity"` select the split variable according to the contrast coding assigned to each predictor.

Contrast systems are ranked using the following approximate scores:

- Treatment contrasts: `0`.
- Sum contrasts: `1`.
- Sliding-difference contrasts: `1`.
- Helmert contrasts: `2`.
- Reverse Helmert contrasts: `2`.
- Polynomial contrasts: `3`.
- Custom or unidentified contrast matrices: `4`.

These scores affect only the choice of split variable. They do not determine the order in which variables are displayed in the configuration labels.

If several variables have the same complexity score, the rightmost tied variable in the interaction coefficient is selected. A warning is issued to document the choice.

The argument `top_split` bypasses the automatic rule and forces the specified variable to be used as the split variable.

Ordering of configuration labels

Both explicit and semantic configuration labels use the following order:

```
target variable(s) -> split variable
```

The target variables are displayed first, following their stable order among the categorical predictors in the fixed-effects model. The split variable selected by `split_rule` or `top_split` is always displayed last.

For a two-way interaction between `A` and `B`, if `A` is the split variable, labels follow the order:

```
B_A
```

If `B` is the split variable, labels follow the order:

```
A_B
```

For a three-way interaction among `A`, `B`, and `C`, if `C` is the split variable, labels follow the order:

```
A_B_C
```

If `A` is the split variable, labels follow the order:

```
B_C_A
```

This ordering rule affects only the presentation and interpretation of the labels. It does not modify the contrast-complexity scores, the selected split variable, or the numerical estimates.

Explicit and semantic configurations

`configuration_explicit` reports the individual cells entering an aggregate. It is intended to show the exact mathematical composition of the displayed estimate.

For example:

```
(B2_A1 + B2_A2)/2
```

indicates that the estimate is the arithmetic mean of cells `B2_A1` and `B2_A2`.

`configuration_semantic` reports the corresponding interpretation:

```
B2_Mean(A1,A2)
```

Both labels use the same variable order. In this example, `B` is the target variable and `A` is the split variable.

With `B` as the split variable, the corresponding labels could be:

```
(A1_B2 + A2_B2)/2
Mean(A1,A2)_B2
```

The explicit label answers the question, "Which cells were combined?", whereas the semantic label answers the question, "What does the resulting estimate represent?"

Raw estimates

For `estimate_type = "observed"`, estimates are computed directly from the observed response values.

At the individual-cell level, the function returns the observed mean, standard error, and confidence interval.

When several disjoint cells are combined, their means are aggregated using normalized contrast weights. Standard errors are combined through classical variance propagation.

Confidence intervals for aggregated observed estimates use a Satterthwaite-like approximation to the degrees of freedom when sufficient information is available.

Confidence intervals of observed means respect the natural range of the response. For proportions (binary outcomes or binomial models), the Wilson score interval is used; for averages of several cells it is computed with the effective sample size $\hat{p}(1-\hat{p})/\mathrm{SE}^2$, which reduces to the classical Wilson interval for a single cell. For positive responses analyzed with count or positive-continuous families (for example Poisson, negative binomial, or Gamma), the interval is computed on the log scale, $\exp(\log m \pm z\,\mathrm{SE}/m)$; a mean of zero receives the exact Poisson upper bound $-\log(\alpha/2)/n$. Other responses use symmetric intervals.

Model-based predictions

For `estimate_type = "predicted"`, predictions are computed from the fixed-effect model:

$$\hat{\eta} = X\hat{\beta}$$

where X is the fixed-effect model matrix and $\hat{\beta}$ is the vector of fixed-effect coefficients.

The covariance matrix of the linear predictors is obtained from:

$$\mathrm{Var}(\hat{\eta}) = XV_{\hat{\beta}}X^{\mathsf{T}}$$

where $V_{\hat{\beta}}$ is the complete fixed-effect variance-covariance matrix.

In mixed-effects models, random effects are excluded. The returned predictions are therefore population-level or fixed-effect predictions.

Predictions are computed on a reference grid built for each coefficient. The grid crosses the levels of the factors involved in the coefficient with the levels of the other factors; numeric predictors

are fixed at their observed means. Each displayed cell averages over the other factors with equal weights (a balanced average, independent of the observed cell sizes). A factor that interacts with the variables of the coefficient, such as `C` for a coefficient of `A:B` in a model containing `A:B:C`, is instead held at the combination of its levels at which the coefficient is defined, given by the first row of its hypothesis matrix: the balanced average for sum, Helmert, and sliding-difference coding, and the reference level for treatment coding.

Averaging the predictions of the observed rows of each cell would instead mix in the effects of covariates whose distribution differs between cells, so that the displayed interaction contrast would no longer correspond to the coefficient.

For `glmmTMB` models, only the conditional component is analyzed. Zero-inflation and dispersion components are not currently decomposed.

Response-scale predictions

For generalized models with `prediction_scale = "response"`, each linear-predictor value is first transformed through the inverse link function. The transformed predictions are then averaged.

Therefore, the function computes:

$$\frac{1}{n}\sum_{i=1}^{n} g^{-1}(\hat{\eta}_i)$$

rather than:

$$g^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\hat{\eta}_i\right)$$

These two quantities are generally different when the inverse link is nonlinear.

Aggregation of predicted estimates

Model-based predictions for different configurations are generally correlated because they depend on the same fixed-effect coefficient vector.

`contrasts_multpred()` therefore does not combine predicted standard errors under an assumption of independence.

Instead, each configuration is represented by its weights over the rows of the reference grid, which are retained throughout the recursive decomposition. When configurations are aggregated, their weights are combined and the aggregate is calculated directly from the complete fixed-effect variance-covariance matrix.

On the link scale, an aggregated prediction with normalized weights a_i is:

$$\hat{\theta} = \sum_i a_i X_i \hat{\beta}$$

Defining:

$$L = \sum_i a_i X_i$$

its variance is:

$$\mathrm{Var}(\hat{\theta}) = L V_{\hat{\beta}} L^{\mathsf{T}}$$

On the response scale, the aggregated prediction is:

$$\hat{\theta} = \sum_i a_i g^{-1}(X_i \hat{\beta})$$

Its variance is approximated using the multivariate delta method. The gradient is:

$$\nabla\hat{\theta} = \sum_i a_i g^{-1\prime}(\hat{\eta}_i) X_i$$

and:

$$\mathrm{Var}(\hat{\theta}) \approx \nabla\hat{\theta} V_{\hat{\beta}} \nabla\hat{\theta}^{\mathsf{T}}$$

This procedure preserves the covariance among the predictions entering each aggregate.

On the response scale, the confidence interval of a prediction or of an average of predictions $\hat{\theta}$ is computed on the link scale and transformed back,

$$g^{-1}\left(g(\hat{\theta}) \pm z \frac{\mathrm{SE}(\hat{\theta})}{\left| g^{-1\prime}(g(\hat{\theta})) \right|} \right),$$

so that its bounds remain within the range of the response (for example between 0 and 1 for probabilities). For a single prediction this is the interval computed on the linear predictor and transformed with the inverse link. The interval is therefore generally asymmetric around the estimate, whereas the reported standard error refers to the response scale.

Supported contrast types

The function supports treatment contrasts (with any reference level), sum contrasts (with any omitted level), Helmert, reverse Helmert, and sliding-difference contrasts, and custom contrast matrices.

For every factor, the levels compared by each contrast column and their weights are taken from the hypothesis matrix $H = [1, C]^{-1}$ of its coding matrix C, as described in `contrasts_onepred()`. Reading them from the signs of C is correct only for orthogonal codings. Groups of levels with unequal weights, which arise with non-orthogonal custom codings, are labelled `W(a1 .20, a3 .80)`, and a message describes the comparison actually estimated when it differs from the one suggested by the signs of the coding column.

Sum contrasts are interpreted as a focal level relative to the unweighted mean across all levels of the corresponding factor.

Scale factors

The hypothesis of an interaction coefficient is the Kronecker product of the hypothesis rows of its components. Each interaction coefficient therefore equals the product of the scale factors of its components, returned in `scale_factor`, multiplied by the displayed difference of differences on the linear-predictor scale. For example, the interaction between the second column of a Helmert-coded factor ($s = 1/3$) and a sum-coded factor ($s = 1$) has $s = 1/3$.

The function verifies this identity for each coefficient. For predicted estimates it holds exactly unless a variable of the coefficient interacts with a numeric predictor, in which case a warning is issued. For observed estimates it holds only when the model contains the interacting factors alone.

Polynomial contrasts are not currently supported for interaction decomposition because all levels generally contribute to each polynomial component through nonzero weights and require a different interpretive strategy.

P-values

P-values refer to the original interaction coefficients being decomposed. They are not direct tests of the individual configuration-specific estimates displayed in the output.

The function recognizes coefficient-table columns named:

- `"Pr(>|t|)"`;
- `"Pr(>|z|)"`;
- `"p-value"`.

Models fitted with `lme4::lmer()` do not ordinarily provide p-values. When `lmerTest` is installed, the model is converted internally using `lmerTest::as_lmerModLmerTest()`, and approximate p-values are extracted from the resulting coefficient table.

If no p-values are available, `p_value` is set to `NA` and `significance` is set to `"NoPValue"`.

Examples

```
################################################################
## Two-way interaction
################################################################


library(lme4)
library(MASS)

set.seed(123)

dat <- expand.grid(
  ID = factor(paste0("S", 1:40)),
  A = factor(c("A1", "A2", "A3")),
  B = factor(c("B1", "B2", "B3", "B4")),
  Rep = 1:2
)

contrasts(dat$A) <- stats::contr.helmert(3)
```

```
contrasts(dat$B) <- MASS::contr.sdif(4)

random_intercept <- rnorm(
  nlevels(dat$ID),
  mean = 0,
  sd = 1
)

effect_A <- c(
  A1 = 0,
  A2 = 1,
  A3 = 2
)

effect_B <- c(
  B1 = 0,
  B2 = 0.5,
  B3 = 1.5,
  B4 = 2.5
)

dat$Score <- 10 +
  effect_A[dat$A] +
  effect_B[dat$B] +
  0.7 * effect_A[dat$A] * effect_B[dat$B] +
  random_intercept[dat$ID] +
  rnorm(nrow(dat), mean = 0, sd = 1.5)

mod <- lme4::lmer(
  Score ~ A * B + (1 | ID),
  data = dat
)

# A is selected as the conditioning variable.
# Labels follow the order B -> A.
out_high <- contrasts_multpred(
  model = mod,
  outcome = "Score",
  estimate_type = "predicted",
  split_rule = "highest_complexity",
  configuration = "all"
)

# B is selected as the conditioning variable.
# Labels follow the order A -> B.
out_low <- contrasts_multpred(
  model = mod,
  outcome = "Score",
  estimate_type = "predicted",
  split_rule = "lowest_complexity",
  configuration = "all"
)
```

```
#################################################################
## Three-way interaction
#################################################################

dat3 <- expand.grid(
  ID = factor(paste0("S", 1:30)),
  A = factor(c("A1", "A2", "A3")),
  B = factor(c("B1", "B2", "B3", "B4")),
  C = factor(c("C1", "C2")),
  Rep = 1:2
)

contrasts(dat3$A) <- stats::contr.helmert(3)
contrasts(dat3$B) <- MASS::contr.sdif(4)
contrasts(dat3$C) <- stats::contr.treatment(2)

dat3$Score <- 10 +
  effect_A[dat3$A] +
  effect_B[dat3$B] +
  1.2 * (dat3$C == "C2") +
  0.4 * effect_A[dat3$A] * effect_B[dat3$B] +
  rnorm(nrow(dat3), mean = 0, sd = 1.5)

mod3 <- stats::lm(
  Score ~ A * B * C,
  data = dat3
)

out3 <- contrasts_multpred(
  model = mod3,
  outcome = "Score",
  estimate_type = "predicted",
  split_rule = "highest_complexity",
  configuration = "all"
)


#################################################################
## glmmTMB model
#################################################################

eta_count <- 1.2 +
  0.2 * effect_A[dat$A] +
  0.15 * effect_B[dat$B]

dat$count <- rpois(
  nrow(dat),
  lambda = exp(eta_count)
)

mod_tmb <- glmmTMB::glmmTMB(
  count ~ A * B + (1 | ID),
```

```
  family = poisson,
  data = dat
)

out_tmb <- contrasts_multpred(
  model = mod_tmb,
  outcome = "count",
  estimate_type = "predicted",
  prediction_scale = "response",
  split_rule = "highest_complexity",
  configuration = "semantic"
)
```

`contrasts_onepred` *Summarize and Interpret Contrasts for One Categorical Predictor*

Description

contrasts_onepred() returns observed or model-based estimates associated with the contrasts used to code a single categorical predictor in a fitted regression model.

The function identifies the predictor's contrast coding and returns the estimates for the levels or groups of levels involved in each contrast.

The function supports standard linear and generalized linear models, mixed-effects models fitted with lme4, mixed-effects models fitted with nlme, and conditional models fitted with glmmTMB.

Usage

```
contrasts_onepred(
  model,
  predictor,
  data = NULL,
  outcome = NULL,
  sig_level = 0.05,
  conf_level = 0.95,
  estimate_type = c("auto", "observed", "predicted"),
  prediction_scale = c("response", "link"),
  bounds = NULL,
  hypotheses = NULL
)
```

Arguments

`model`	A fitted model object. Supported classes include objects returned by `stats::lm()`, `stats::glm()`, `lme4::lmer()`, `lme4::glmer()`, `nlme::lme()`, and `glmmTMB::glmmTMB()`.
`predictor`	Character string indicating the name of the categorical predictor whose contrasts should be summarized. The corresponding variable in data must be a factor.

`data` Optional data frame containing the variables used to fit `model`. If `NULL`, the data supplied in the model call are recovered. Rows excluded from the fit, through `subset` or missing values, are dropped, matching rows by name. Factors are given the contrasts used for fitting, including those supplied through the `contrasts` argument of the model call.

`outcome` Optional character string indicating the name of the dependent variable; by default, the dependent variable of `model`. It is used only for observed estimates, which summarize the response of the model evaluated in `data`: a transformed response such as `log(y)` is summarized on the scale of the model, and a logical response or a factor response of a binomial model is converted to 0/1 as in `stats::glm()` (the first level is 0). Observed estimates are not available for responses with several columns, such as `cbind(successes, failures)`.

`sig_level` Numeric value indicating the significance threshold used to classify contrasts as "sig" or "notsig". The default is 0.05.

`conf_level` Numeric value indicating the confidence level used for confidence intervals. The default is 0.95.

`estimate_type` Character string specifying how the level-specific estimates should be computed. Available options are:

`"auto"` Uses observed means when the fixed part of the model contains only the focal predictor, the model has no random effects and no unequal prior weights, and, for generalized models, `prediction_scale = "response"`. Model-based predictions are used otherwise: with other predictors or interactions, with random effects (observed means would ignore the clustering of the observations), with `weights` (observed means are unweighted), and for estimates on the link scale.

`"observed"` Computes observed means directly from the supplied data.

`"predicted"` Computes model-based predictions from the fixed effects and their variance-covariance matrix.

The default is `"auto"`.

`prediction_scale`

Character string indicating the scale used for model-based predictions. Available options are:

"response" Predictions are reported on the response scale. For generalized models, the inverse link function is applied.

"link" Predictions are reported on the linear-predictor scale.

This argument is relevant only when estimate_type = "predicted".

`bounds` Optional numeric vector `c(lower, upper)` giving the natural range of the response, for responses analyzed with a Gaussian (identity link) model, such as a Likert score (`c(1, 7)`), a percentage (`c(0, 100)`), or a reaction time (`c(0, Inf)`). The bounds cannot be inferred from the data, because the observed range need not coincide with the possible range. When supplied, confidence intervals of observed and predicted estimates are computed on a transformed scale and transformed back, so that they remain within the bounds: the logit of $(m-a)/(b-a)$ for two finite bounds, and the logarithm of the distance from the finite bound otherwise. Estimates are unchanged. The argument is ignored, with

a warning, for generalized models, whose intervals already respect the range implied by their family. Bounds correct the representation of uncertainty, not the model: when many observations lie at a bound, a model suited to the scale of the response (for example an ordinal model) is more appropriate.

`hypotheses` Optional numeric matrix declaring the comparisons that the coefficients of `predictor` are intended to estimate, with one row per coefficient, in the order of the contrast columns, and one column per factor level, as the rows of a hypothesis matrix (for example `rbind("Treat-Control" = c(-1, 0.5, 0.5), "T2-Control" = c(-1, 0, 1))`). Each row must sum to zero. Column names, when present, must be the factor levels; row names label the hypotheses. Each declared row is compared with the comparison that the coefficient actually estimates, and a message reports whether the coefficient estimates the declared hypothesis, exactly, up to a positive factor, or with reversed sign, or which comparison it estimates instead. The model is not modified: the function reports what the fitted coding estimates. The result is stored in the `hypothesis_check` attribute.

Details

The function retrieves the contrast (coding) matrix used for predictor and derives from it the hypothesis matrix, which states the combination of level-specific means estimated by each coefficient. The levels contributing positively and negatively to each contrast, and their weights, are taken from the hypothesis matrix (see the section "Coding matrix and hypothesis matrix").

Model-based predictions are computed manually from the fixed-effects model:

$$\hat{\eta} = X\hat{\beta}$$

Their variance on the linear-predictor scale is computed as:

$$\mathrm{Var}(\hat{\eta}) = X V_{\hat{\beta}} X^{\mathsf{T}}$$

where X is the fixed-effects model matrix and $V_{\hat{\beta}}$ is the variance-covariance matrix of the fixed-effect coefficients.

For estimate_type = "observed", estimates are observed means computed directly from the data.

Confidence intervals of observed means respect the natural range of the response. For proportions (binary outcomes or binomial models), the Wilson score interval is used; for averages of several levels or cells it is computed with the effective sample size $\hat{p}(1-\hat{p})/\mathrm{SE}^2$, which reduces to the classical Wilson interval for a single level. For positive responses analyzed with count or positive-continuous families (for example Poisson, negative binomial, or Gamma), the interval is computed on the log scale, $\exp(\log m \pm z\,\mathrm{SE}/m)$; a mean of zero receives the exact Poisson upper bound $-\log(\alpha/2)/n$. Other responses use symmetric intervals.

For estimate_type = "predicted", estimates are computed from the fixed-effect coefficients and their variance-covariance matrix.

In mixed-effects models, random effects are excluded from model-based predictions. Predicted estimates are therefore population-level or fixed-effects predictions.

Model-based estimates are computed on a reference grid. Other numeric predictors are fixed at their observed means. Each level-specific estimate averages over the levels of the other categorical predictors with equal weights (a balanced average, independent of the observed cell sizes). A

categorical predictor that interacts with `predictor` is instead held at the combination of its levels at which the coefficients of `predictor` are defined, given by the first row of its hypothesis matrix: this is again the balanced average for sum, Helmert, and sliding-difference coding, and the reference level for treatment coding. On the linear-predictor scale, the displayed comparisons therefore reconstruct the coefficients exactly; if `predictor` interacts with a numeric predictor, its coefficients refer to the value zero of that predictor and the function issues a warning.

On the response scale, level estimates that average over other factors are means of the response-scale predictions, and their confidence intervals are obtained with the multivariate delta method.

For generalized models with prediction_scale = "response", estimates are transformed using the inverse link function. Standard errors on the response scale are approximated using the delta method.

On the response scale, the confidence interval of a prediction or of an average of predictions $\hat{\theta}$ is computed on the link scale and transformed back,

$$g^{-1}\left(g(\hat{\theta}) \pm z\,\frac{\mathrm{SE}(\hat{\theta})}{\left|g^{-1\prime}(g(\hat{\theta}))\right|}\right),$$

so that its bounds remain within the range of the response (for example between 0 and 1 for probabilities). For a single prediction this is the interval computed on the linear predictor and transformed with the inverse link. The interval is therefore generally asymmetric around the estimate, whereas the reported standard error refers to the response scale.

For non-polynomial contrasts involving more than one level on the positive or negative side, the reported group estimate is the weighted mean of the corresponding level-specific estimates, with weights taken from the hypothesis matrix. For all standard codings these weights are equal, so the group estimate is the unweighted arithmetic mean. Groups with unequal weights, which arise with non-orthogonal custom codings, are labelled `W(A .20, C .80)`.

Standard errors of groups of observed means treat the level means as independent. Standard errors of groups of model-based predictions use the complete variance-covariance matrix of the fixed effects; on the response scale they are obtained with the multivariate delta method, and the confidence intervals are computed on the link scale as described above.

The Significance label refers to the p-value of the model coefficient associated with the corresponding contrast. It does not represent a direct test of the individual level-specific estimates or their confidence intervals.

For glmmTMB objects, only the conditional model component, cond, is analyzed. Zero-inflation and dispersion components are not currently summarized.

Value

A data frame with one or more rows per contrast.

For non-polynomial contrasts, the output contains two rows per contrast, the positive and the negative side, and includes:

Contrast Name or sequential identifier of the contrast.

Levels Factor level or group of levels represented by the contrast: a level name, `"Mean(A, B)"` for equally weighted groups, `"W(A .20, C .80)"` for groups with unequal weights, or `"Grand Mean"` for sum contrasts.

Estimate Observed mean or model-based estimate.

SE Standard error of the reported estimate.

CI_low Lower confidence-interval boundary.

CI_high Upper confidence-interval boundary.

Significance Significance label associated with the corresponding model coefficient: "sig", "not-sig", or NA.

Scale_factor Scale factor s_j such that the model coefficient equals s_j multiplied by the difference between the two displayed quantities on the linear-predictor scale. It equals 1 for treatment, sum, and sliding-difference contrasts.

Display_Levels Labels used for display; currently identical to `Levels`.

For polynomial contrasts, the output contains one row per factor level for each polynomial component, has no `Scale_factor` column, and additionally includes:

Weight Polynomial contrast weight associated with each factor level.

Polynomial_fit Curve implied by the single polynomial component: the mean of the level estimates on the linear-predictor scale plus the component coefficient multiplied by the polynomial weight, transformed to the scale of `Estimate` (for example with the inverse logit for a binomial model on the response scale). The baseline is computed on the same reference grid as the estimates, so that the curve passes through the region of the displayed points. On the linear-predictor scale, the baseline plus the sum of all components reproduces every level estimate.

The returned object also contains the following attributes:

disclaimer Description of how estimates, standard errors, and confidence intervals were computed, including any note on non-orthogonal custom codings. The same information is printed as a message when the function is executed.

contrast_type The identified contrast type.

hypothesis_matrix The hypothesis matrix H without the intercept row, with one row per contrast and one column per factor level.

family, link, estimate_type, prediction_scale Model family and link (NA when unavailable) and the settings used.

Sum contrasts and the Grand Mean

For sum-to-zero contrasts, each contrast is displayed as a comparison between one factor level and a row labelled "Grand Mean".

The Grand Mean is defined as the unweighted arithmetic mean of the level-specific estimates. Each factor level therefore contributes equally, regardless of its sample size.

When estimate_type = "observed", the Grand Mean is:

$$\frac{\bar{Y}_1 + \bar{Y}_2 + \cdots + \bar{Y}_k}{k}$$

where $\bar{Y}_j$ is the observed mean for level j.

When estimate_type = "predicted" and prediction_scale = "link", the Grand Mean is the arithmetic mean of the level-specific linear predictors:

$$\frac{\hat{\eta}_1 + \hat{\eta}_2 + \cdots + \hat{\eta}_k}{k}$$

With sum contrasts, this quantity generally corresponds to the model intercept when the remaining predictors are fixed at the values used to construct the prediction data.

When estimate_type = "predicted" and prediction_scale = "response", the Grand Mean is the arithmetic mean of the response-scale predictions:

$$\frac{g^{-1}(\hat{\eta}_1) + g^{-1}(\hat{\eta}_2) + \cdots + g^{-1}(\hat{\eta}_k)}{k}$$

This is not generally equal to applying the inverse link function to the average linear predictor:

$$g^{-1}\left(\frac{1}{k}\sum_j \hat{\eta}_j\right) \neq \frac{1}{k}\sum_j g^{-1}(\hat{\eta}_j)$$

For a binomial model with a logit link, the response-scale Grand Mean is the arithmetic mean of the predicted probabilities.

For a Poisson or negative-binomial model with a log link, the response-scale Grand Mean is the arithmetic mean of the predicted expected counts.

The standard error of an observed Grand Mean is computed by treating the level-specific observed means as independent estimates.

The standard error of a predicted Grand Mean is computed using the complete variance-covariance matrix of the fixed-effect coefficients. On the response scale, the required gradient is obtained using the multivariate delta method.

Coding matrix and hypothesis matrix

The contrast matrix C describes how the factor is coded in the model. It does not, in general, describe what the coefficients compare. For a factor with k levels and $k - 1$ contrasts, the level-specific means satisfy

$$\mu = X_C\beta, \qquad X_C = [1, C],$$

so that

$$\beta = H\mu, \qquad H = X_C^{-1}.$$

Row h_j of H states the combination of level means estimated by coefficient β_j. The function always derives the compared levels from H. Reading them from the signs of C is correct only for orthogonal codings; for non-orthogonal custom codings it can display a comparison different from the one estimated by the coefficient. If X_C is not full rank, the function stops.

Every non-intercept row of H sums to zero and is written as

$$h_j = s_j(w_j^+ - w_j^-),$$

where w_j^+ and w_j^- are non-negative weights summing to one (the two displayed groups) and $s_j > 0$ is the scale factor returned in `Scale_factor`. Consequently,

$$\beta_j = s_j \left(\sum_i w_{ji}^+ \mu_i - \sum_i w_{ji}^- \mu_i \right).$$

On the linear-predictor scale this identity holds exactly for model-based predictions of an additive model, whatever values the other predictors are fixed at, because each row of H sums to zero. The function verifies it and issues a warning when it fails, which usually indicates that the predictor is involved in an interaction. For observed means, the identity holds only when the model contains the focal predictor alone; the disclaimer reports when it does not.

For non-orthogonal custom codings, the function also compares the levels suggested by the signs of each coding column with the levels involved in the corresponding hypothesis. When they differ, a message describes the comparison actually estimated by the coefficient.

Contrast types

The contrast type is identified from the coding matrix and, for matrices that do not coincide with a standard generator, from the structure of the hypothesis matrix. It determines only the presentation of the comparisons; their content always comes from the hypothesis matrix. The function recognizes:

- Treatment contrasts, with any reference level (for example `stats::contr.treatment()` with any `base`, or `stats::contr.SAS()`): each level versus the reference level, $s = 1$.
- Sum contrasts, with any omitted level (for example `stats::contr.sum()`): each level versus the Grand Mean, $s = 1$.
- Helmert contrasts generated by `stats::contr.helmert()`: each level versus the mean of the preceding levels, $s_j = 1/(j+1)$.
- Reverse Helmert contrasts generated by `reverse_helmert()`: each level versus the mean of the subsequent levels, $s_j = 1/(k-j+1)$.
- Sliding-difference contrasts generated by `MASS::contr.sdif()`: each level versus the preceding level, $s = 1$.
- Polynomial contrasts generated by `stats::contr.poly()`, with equally spaced or user-supplied `scores`.
- Other user-defined (custom) contrast matrices, displayed as the two weighted groups of levels implied by the hypothesis matrix, with an arbitrary scale factor.

Rescaled versions of these codings are recognized as well; rescaling a coding column changes only the scale factor.

Sum contrasts are displayed as one level versus the Grand Mean of all levels, which includes the focal level. This presentation is equivalent to the split of the hypothesis row into disjoint groups and is used because it matches the usual interpretation of sum coding.

Polynomial contrasts are handled separately because all factor levels contribute to each polynomial component through their polynomial weights, so that a split into positive and negative groups is not

meaningful. For each component, the function returns one row per factor level and reports both the polynomial weight and the fitted curve implied by that component.

P-values

P-values are extracted from the coefficient table associated with the fixed-effects or conditional component of the model.

The function recognizes coefficient-table columns named:

- "Pr(>|t|)";
- "Pr(>|z|)";
- "p-value".

Only the main-effect coefficients associated with predictor are used. Interaction coefficients involving predictor are excluded.

For model classes that do not provide frequentist p-values, the Significance column is set to NA.

Examples

```
################################################################
## Linear model with treatment contrasts
################################################################

set.seed(123)

dat <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 40))
)

dat$y <- c(
  rnorm(40, mean = 10, sd = 2),
  rnorm(40, mean = 13, sd = 2),
  rnorm(40, mean = 17, sd = 2)
)

contrasts(dat$group) <- contr.treatment(3)

mod_lm <- lm(
  y ~ group,
  data = dat
)

contrasts_onepred(
  model = mod_lm,
  predictor = "group",
  outcome = "y",
  data = dat,
  estimate_type = "observed"
)

contrasts_onepred(
```

```
  model = mod_lm,
  predictor = "group",
  outcome = "y",
  data = dat,
  estimate_type = "predicted"
)


#################################################################
## Generalized linear model with sum contrasts
#################################################################

contrasts(dat$group) <- contr.sum(3)

dat$binary_y <- rbinom(
  nrow(dat),
  size = 1,
  prob = plogis(
    -1 +
      0.5 * (dat$group == "B") +
      1.2 * (dat$group == "C")
  )
)

mod_glm <- glm(
  binary_y ~ group,
  family = binomial,
  data = dat
)

contrasts_onepred(
  model = mod_glm,
  predictor = "group",
  outcome = "binary_y",
  data = dat,
  estimate_type = "predicted",
  prediction_scale = "response"
)


#################################################################
## Linear model with polynomial contrasts
#################################################################

dat$ordered_group <- ordered(
  dat$group,
  levels = c("A", "B", "C")
)

contrasts(dat$ordered_group) <- contr.poly(3)

mod_poly <- lm(
  y ~ ordered_group,
```

```
  data = dat
)

contrasts_onepred(
  model = mod_poly,
  predictor = "ordered_group",
  outcome = "y",
  data = dat,
  estimate_type = "predicted"
)


######################################################################
## Mixed-effects models
######################################################################


library(lme4)
library(nlme)
library(glmmTMB)
library(MASS)

set.seed(123)

dat_mixed <- expand.grid(
  id = factor(1:60),
  time = 1:5
)

dat_mixed$group <- factor(
  sample(
    c("A", "B", "C"),
    nrow(dat_mixed),
    replace = TRUE
  )
)

random_intercept <- rnorm(
  nlevels(dat_mixed$id),
  mean = 0,
  sd = 1
)

eta <- 10 +
  0.3 * dat_mixed$time +
  2 * (dat_mixed$group == "B") +
  5 * (dat_mixed$group == "C") +
  random_intercept[dat_mixed$id]

dat_mixed$y <- eta +
  rnorm(
    nrow(dat_mixed),
    mean = 0,
```

```
    sd = 2
  )


## Linear mixed model with Helmert contrasts

contrasts(dat_mixed$group) <- contr.helmert(3)

mod_lmer <- lmer(
  y ~ group + time + (1 | id),
  data = dat_mixed
)

contrasts_onepred(
  model = mod_lmer,
  predictor = "group",
  outcome = "y",
  data = dat_mixed,
  estimate_type = "predicted"
)


## Linear mixed model fitted with nlme and sliding differences

contrasts(dat_mixed$group) <- MASS::contr.sdif(3)

mod_lme <- nlme::lme(
  y ~ group + time,
  random = ~1 | id,
  data = dat_mixed
)

contrasts_onepred(
  model = mod_lme,
  predictor = "group",
  outcome = "y",
  data = dat_mixed,
  estimate_type = "predicted"
)


## Generalized mixed model with sum contrasts

contrasts(dat_mixed$group) <- contr.sum(3)

eta_count <- 1.5 +
  0.2 * dat_mixed$time +
  0.3 * (dat_mixed$group == "B") +
  0.7 * (dat_mixed$group == "C") +
  random_intercept[dat_mixed$id]

dat_mixed$count <- rpois(
  nrow(dat_mixed),
```

```
  lambda = exp(eta_count)
)

mod_tmb <- glmmTMB(
  count ~ group + time + (1 | id),
  family = poisson,
  data = dat_mixed
)

contrasts_onepred(
  model = mod_tmb,
  predictor = "group",
  outcome = "count",
  data = dat_mixed,
  estimate_type = "predicted",
  prediction_scale = "response"
)
```

`extract_param` *Extract and interpret model parameters*

Description

`extract_param()` extracts model coefficients and inferential statistics using `parameters::model_parameters()`. It returns a compact, non-redundant parameter table and, when appropriate, transforms coefficients and confidence intervals into a more directly interpretable metric.

Depending on the model family and link function, the interpreted estimate may represent an odds ratio, risk ratio, incidence rate ratio, hazard ratio, or another multiplicative ratio. For models for which no meaningful automatic transformation is available, the original coefficient is retained.

The function also computes an approximate Cohen's d for each coefficient: from the log odds ratio for logistic models, and from the coefficient-level t or z statistic otherwise.

Usage

```
extract_param(
  model,
  conf_level = 0.95,
  include_intercept = FALSE,
  effects = "fixed",
  component = "conditional",
  transform = c("auto", "none", "exponentiate")
)
```

Arguments

model	A fitted statistical model supported by parameters::model_parameters().
conf_level	A single numeric value between 0 and 1 specifying the confidence level. The default is 0.95.
include_intercept	Logical. Should the intercept be included in the returned table? The default is FALSE.
effects	Character string specifying the model effects to extract for models that distinguish between fixed and random effects. The default is "fixed".
component	Character string specifying the model component to extract for models with multiple components, such as zero-inflated or hurdle models. The default is "conditional".
transform	Character string specifying how model coefficients should be transformed for interpretation. Possible values are: "auto" Automatically exponentiate coefficients from models with log or logit links and from Cox proportional-hazards models. "none" Retain the original coefficient and confidence interval as the interpreted estimate. "exponentiate" Exponentiate the coefficient and confidence interval regardless of the detected model family or link. The default is "auto".

Details

The Estimate column always contains the coefficient on the scale on which the model was fitted. For generalized models, this is generally the scale of the linear predictor.

When transform = "auto", the following transformations are applied:

Identity link No transformation. Interpreted_Estimate is equal to Estimate.

Binomial model with logit link The coefficient is exponentiated and interpreted as an odds ratio.

Binomial model with log link The coefficient is exponentiated and interpreted as a risk ratio.

Poisson or negative-binomial model with log link The coefficient is exponentiated and interpreted as an incidence rate ratio.

Other models with a log link The coefficient is exponentiated and interpreted as a multiplicative ratio.

Cox proportional-hazards model The coefficient is exponentiated and interpreted as a hazard ratio.

No automatic exponentiation is performed for links such as probit, cauchit, or cloglog, because exponentiating an individual coefficient does not generally produce a standard effect measure with a direct interpretation.

The interpreted estimates are not necessarily predictions on the response scale. For example, an odds ratio is a ratio of odds rather than a predicted probability. Predicted probabilities or expected responses require the intercept and values of the other model predictors.

Confidence intervals and p-values use the same inferential method as the significance labels of the plotting functions, so that an interval excluding zero always corresponds to a significant coefficient:

- linear mixed models fitted with lme4 or lmerTest use Satterthwaite degrees of freedom when lmerTest is installed, as in their p-values;
- generalized linear (mixed) models use Wald intervals, which correspond to the Wald z tests reported by `summary()`; profile intervals can disagree with these tests near the significance threshold;
- other models use the default method of `parameters::model_parameters()`.

Approximate Cohen's d values are computed as follows, as reported in `d_method`:

- for models with a logit link, the log odds ratio is converted as $d = \beta\sqrt{3}/\pi$ (Chinn, 2000), which treats the logistic distribution of the latent response as the reference standard deviation; the same conversion is applied to the confidence limits;
- t statistics are converted using `effectsize::t_to_d()` with the coefficient-specific degrees of freedom;
- z statistics are converted using `effectsize::z_to_d()` with the number of observations returned by `stats::nobs()`.

The conversions based on t or z do not depend on how the contrast is scaled. The conversion from the log odds ratio refers to the coefficient as coded: for a contrast whose coefficient equals a scale factor s multiplied by the compared difference (for example Helmert contrasts; see `contrasts_onepred()`), the d of the compared difference is d/s. The plotting functions apply this rescaling.

These conversions are approximations and assume conditions implicit in the conversion formulas. In particular, the conversion to Cohen's d assumes two groups of approximately equal size, and the conversion from the log odds ratio assumes a logistic latent response. Directly estimated effect sizes should be preferred when the necessary data and an appropriate estimator are available.

Value

A data frame with one row for each extracted model parameter and the following columns:

Parameter Name of the model parameter.

Estimate Untransformed model coefficient on the model scale.

SE Standard error of the untransformed coefficient.

CI Confidence level used for the interval.

CI_low Lower confidence limit of `Estimate`.

CI_high Upper confidence limit of `Estimate`.

Statistic Coefficient-level test statistic.

Statistic_type Type of test statistic, usually `"t"` or `"z"`.

df Coefficient-level degrees of freedom, when available.

p Coefficient-level p-value, when available.

Interpreted_Estimate Coefficient transformed into the interpretation indicated by `Estimate_type`.

Interpreted_CI_low Lower confidence limit of the interpreted estimate.

Interpreted_CI_high Upper confidence limit of the interpreted estimate.

Estimate_type Interpretation of `Interpreted_Estimate`, such as `"Coefficient"`, `"Odds ratio"`, `"Risk ratio"`, `"Incidence rate ratio"`, `"Hazard ratio"`, or `"Multiplicative ratio"`.

d Approximate Cohen's d.

d_CI_low Lower confidence limit of approximate Cohen's d.

d_CI_high Upper confidence limit of approximate Cohen's d.

d_method Conversion used for `d`: `"log odds ratio"`, `"t"`, or `"z"`; `NA` when no conversion was possible.

Note

The coefficient transformation is based on the model family and link detected through `stats::family()` whenever available. Some model classes may use non-standard structures or parameterizations and may therefore require `transform = "none"` or `transform = "exponentiate"` to override automatic behaviour.

In models with multiple components, the interpretation may depend on the component selected through `component`.

References

Chinn, S. (2000). A simple method for converting an odds ratio to effect size for use in meta-analysis. *Statistics in Medicine, 19*(22), 3127-3131.

Examples

```
set.seed(123)

# Linear regression
dat_lm <- data.frame(
  y = rnorm(120),
  x = rnorm(120),
  group = factor(rep(c("A", "B", "C"), each = 40))
)

contrasts(dat_lm$group) <- contr.sum(3)

mod_lm <- lm(y ~ x + group, data = dat_lm)

extract_param(mod_lm)

# Binomial logistic regression
dat_glm <- data.frame(
  y = rbinom(200, size = 1, prob = 0.40),
  x = rnorm(200),
  group = factor(sample(c("A", "B", "C"), 200, replace = TRUE))
)

contrasts(dat_glm$group) <- contr.sum(3)

mod_glm <- glm(
  y ~ x + group,
```

```
  data = dat_glm,
  family = binomial(link = "logit")
)

extract_param(mod_glm)

# Disable automatic coefficient transformation
extract_param(mod_glm, transform = "none")
```

`model_contrast_table` *Summarize Model Variables, Families, and Contrast Matrices*

Description

`model_contrast_table()` extracts model-level and variable-level information from a fitted model object.

It reports the model class, family and link function when available, number of variables, dependent variable, predictors, interactions, random-effect grouping variables, variable types, number of factor levels, and contrast matrix types for categorical variables.

The function is designed to work with fitted models supported by the package's internal model-extraction infrastructure, including models fitted with `stats::lm()`, `stats::glm()`, `lme4::lmer()`, `lme4::glmer()`, `nlme::lme()`, and `glmmTMB::glmmTMB()`.

Usage

```
model_contrast_table(model, type = c("all", "summary", "x&y", "x", "y"))
```

Arguments

`model`	A fitted model object.
`type`	Character string indicating which output should be returned. Available options are `"all"`, `"summary"`, `"x&y"`, `"x"`, and `"y"`.

Details

Variables found in the model frame are reported in the variable-level summary. In mixed-effects models, grouping variables used in the random structure may therefore also appear among the predictors.

To avoid interpreting these variables as fixed effects only, the model-level summary includes a separate `"Random Effects"` row listing the grouping variables used in the random-effects structure.

The variable-level summary also contains a `Random_Effect` column that indicates whether each variable is used as a random-effect grouping variable.

The function identifies the following contrast matrix types:

- `"Sum"`: sum-to-zero contrasts generated by `stats::contr.sum()`.

- "Treatment": treatment contrasts generated by stats::contr.treatment().
- "Polynomial": polynomial contrasts generated by stats::contr.poly(), with equally spaced or user-supplied scores.
- "Helmert": Helmert contrasts generated by stats::contr.helmert().
- "Sliding Differences": sliding-difference contrasts generated by MASS::contr.sdif().
- "Reverse Helmert": reverse Helmert contrasts generated by reverse_helmert().
- "Custom": any other user-defined contrast matrix.

Matrices that do not coincide with one of these generators are classified from the structure of their hypothesis matrix, the inverse of the coding matrix with an intercept column. Treatment contrasts are therefore recognized with any reference level, sum contrasts with any omitted level, and all types after rescaling of their columns. The same classification is used by contrasts_onepred() and moplot_onepred().

Value

Depending on type, either a list of two data frames or a single data frame.

Examples

```
set.seed(1234)
dat <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 20)),
  y = rnorm(60)
)
contrasts(dat$group) <- stats::contr.helmert(3)
mod <- stats::lm(y ~ group, data = dat)
model_contrast_table(mod)
```

---

moplot_export *Export a MoPlot figure using publication-ready settings*

Description

moplot_export() exports a plot created by moplot_onepred() or moplot_multpred() using dimensions recommended from the graphical complexity of the plot: the number of displayed quantities (or of polynomial components and levels), the depth of the interaction hierarchy, and the presence of the coefficient panel.

PNG and TIFF files are exported at high resolution. TIFF files use lossless LZW compression. PDF files are exported through the Cairo device to improve rendering of Unicode symbols used in MoPlot figures.

Usage

```
moplot_export(
  plot,
  filename,
  width = NULL,
  height = NULL,
  dpi = 600,
  bg = "white"
)
```

Arguments

plot	A plot object created by moplot_onepred() or moplot_multpred(). Other plots can be exported by supplying both width and height.
filename	Character string specifying the output filename. Supported extensions are .png, .tif, .tiff, and .pdf.
width	Optional export width in inches. If NULL, the recommended width stored in plot is used.
height	Optional export height in inches. If NULL, the recommended height stored in plot is used.
dpi	Resolution in dots per inch for raster formats. The default is 600. This argument is ignored for PDF output.
bg	Background colour. The default is "white".

Value

Invisibly returns the normalized path of the exported file.

Examples

```
set.seed(123)

dat <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 20)),
  y = rnorm(60, mean = rep(c(0, 0.4, 0.9), each = 20))
)

contrasts(dat$group) <- contr.helmert(3)

plot <- moplot_onepred(
  model = lm(y ~ group, data = dat),
  predictor = "group",
  caption = FALSE
)

# Recommended dimensions, stored in the plot.
attr(plot, "moplot_export_size")

path <- moplot_export(
```

```
  plot,
  filename = file.path(tempdir(), "moplot.png"),
  dpi = 150
)

file.exists(path)
unlink(path)
```

`moplot_multpred` *Plot contrasts for interacting categorical predictors*

Description

`moplot_multpred()` creates a graphical representation of model interactions involving two or more categorical predictors.

The function decomposes each selected interaction coefficient into the elementary and higher-order contrasts implied by the contrast matrices used to fit the model. The upper panel displays the observed or model-predicted quantities involved in this decomposition. The optional lower panel displays the corresponding model coefficient, its confidence interval, and an approximate Cohen's d.

By default, the function identifies the interaction of highest order in the fitted model and displays all coefficients associated with that interaction. If more than one interaction of the highest order is present, the function asks the user to select one explicitly through `interaction`.

Usage

```
moplot_multpred(
  model,
  data = NULL,
  interaction = NULL,
  coefficients = NULL,
  interaction_info = TRUE,
  split_rule = c("last_variable", "first_variable", "lowest_complexity",
    "highest_complexity"),
  top_split = NULL,
  sig_level = 0.05,
  conf_level = 0.95,
  estimate_type = c("auto", "observed", "predicted"),
  prediction_scale = c("response", "link"),
  coefficient_scale = c("model", "interpreted"),
  per_difference = FALSE,
  bounds = NULL,
  hypotheses = NULL,
  show_coef = TRUE,
  style = c("minimal", "tree"),
  caption = TRUE,
```

```
  digits = 2L,
  base_size = 13,
  panel_heights = c(2, 1)
)
```

Arguments

`model`	A fitted model supported by the package. Supported model classes include objects returned by `stats::lm()`, `stats::glm()`, `lme4::lmer()`, `lme4::glmer()`, `nlme::lme()`, and `glmmTMB::glmmTMB()`.
`data`	Optional data frame containing the variables used to fit `model`. If `NULL`, the data supplied in the model call are recovered, restricted to the rows used for fitting; the model frame is used only when they are unavailable. The original variables are needed for model-based estimates when the model contains transformed terms such as `log(x)`. Rows of a supplied data frame that were excluded from the fit, through `subset` or missing values, are dropped, matching rows by name.
`interaction`	Optional character string identifying the interaction term to display, for example `"A:B:C"`. If `NULL`, the function automatically selects the unique interaction of highest order in the fitted model. If several interactions share the highest order, the function stops and asks the user to specify `interaction`.
`coefficients`	Optional character vector identifying specific model coefficients to display. If `NULL`, all coefficients associated with the selected `interaction` are displayed. If `interaction = NULL` and `coefficients` is supplied, the corresponding interaction is inferred automatically, provided that all selected coefficients belong to the same interaction.
`interaction_info`	Logical. Should the function print the available interaction terms and their corresponding coefficient selections? The default is `TRUE`. The printed table shows syntax that can be copied directly into `interaction` and `coefficients` in subsequent calls.
`split_rule`	Character string specifying how the top-level split variable is selected during the hierarchical decomposition. Available options are: `"last_variable"` Selects the last variable appearing in the interaction coefficient. `"first_variable"` Selects the first variable appearing in the interaction coefficient. `"lowest_complexity"` Selects the variable with the lowest contrast-complexity score. `"highest_complexity"` Selects the variable with the highest contrast-complexity score.
`top_split`	Optional character string identifying the predictor to force as the top-level split variable. When supplied, `top_split` overrides `split_rule`.
`sig_level`	Numeric value between 0 and 1 defining the significance threshold used for coefficient labels. The default is `0.05`.

`conf_level` Numeric value between 0 and 1 specifying the confidence level used for displayed confidence intervals. The default is `0.95`.

`estimate_type` Character string specifying how quantities in the upper panel are computed. Available options are:

`"auto"` Uses observed means when the selected interaction involves all the predictors of the fixed part, the model has no random effects and no unequal prior weights, and, for generalized models, `prediction_scale = "response"`. Model-based predictions are used otherwise.

`"observed"` Displays observed aggregated means computed directly from `data`.

`"predicted"` Displays model-based predictions computed from the fixed effects and their variance-covariance matrix.

The default is `"auto"`.

`prediction_scale`

Character string specifying the scale used for model-based predictions when `estimate_type = "predicted"`. Available options are:

`"response"` Displays predictions on the response scale.

`"link"` Displays predictions on the linear-predictor/link scale.

The default is `"response"`. This argument is ignored when `estimate_type = "observed"`.

`coefficient_scale`

Character string specifying the quantity displayed in the lower panel. Available options are:

`"model"` Displays the coefficient on its original model scale.

`"interpreted"` Displays the interpreted coefficient when an appropriate transformation is available, such as an odds ratio or incidence rate ratio.

The default is `"model"`.

`per_difference` Logical. Should the lower-panel coefficients be divided by their scale factor? The default is `FALSE`.

Each interaction coefficient equals the product of the scale factors of its components multiplied by the displayed difference of differences (for example `s = 1/3` when a Helmert contrast with `s = 1/3` interacts with a sum contrast). When `per_difference = TRUE`, the lower panel reports `beta / s` and its confidence interval; with `coefficient_scale = "interpreted"`, ratios are computed as `exp(beta / s)`. Test statistics, p-values, and Cohen's d are unaffected.

`bounds` Optional numeric vector `c(lower, upper)` giving the natural range of a response analyzed with a Gaussian (identity link) model, such as `c(1, 7)` for a Likert score or `c(0, Inf)` for a reaction time, so that confidence intervals remain within it. Ignored, with a warning, for generalized models. See `contrasts_multpred()`.

`hypotheses` Optional named list declaring, for one or more categorical predictors, the comparisons that their contrast coefficients are intended to estimate, one matrix per predictor. Each interaction coefficient is checked through its components; the result is reported in a message and in the caption. The figure always shows the comparisons that the coefficients estimate. See `contrasts_multpred()`.

`show_coef` Logical. Should the lower coefficient panel be displayed? The default is `TRUE`.

`style` Character string specifying the graphical representation of the interaction decomposition. Available options are:

`"minimal"` Displays the observed or predicted quantities and the dashed segments representing the elementary component contrasts, without displaying intermediate delta formulas.

`"tree"` Displays the decomposition as a hierarchical tree showing how elementary contrasts are recursively combined into higher-order contrasts and ultimately into the final interaction contrast.

The default is `"minimal"`.

`caption` Controls whether a descriptive caption is printed in the console. It can be:

- `TRUE`, to generate and print an automatic caption;
- `FALSE` or `NULL`, to suppress the caption;
- a single character string to print a custom caption.

The automatic caption describes the selected interaction, the graphical hierarchy, the contrast coding assigned to each interacting variable, the displayed estimate type, the selected graphical style, and the lower coefficient panel when present.

The caption is printed in the console and is not added to the graphical object.

`digits` Non-negative integer specifying the number of decimal places used in lower-panel estimate and effect-size labels. The default is `2`.

`base_size` Positive numeric value defining the base font size used in both panels. The default is `13`.

`panel_heights` Numeric vector of length two specifying the relative heights of the upper and lower panels. The default is `c(2, 1)`. This argument is used only when `show_coef = TRUE`.

Details

`moplot_multpred()` is a graphical interface to the hierarchical decomposition performed by `contrasts_multpred()`.

Each selected interaction coefficient is decomposed according to the contrast matrices assigned to the categorical predictors involved in the interaction. This allows treatment, sum, Helmert, reverse Helmert, sliding-difference, and interpretable custom contrasts to be represented directly in terms of the factor levels or level aggregates implied by the fitted model.

The graphical hierarchy is determined by `split_rule` or `top_split`. The top-level split variable defines the outermost grouping structure, whereas successive predictors define progressively lower levels of the decomposition. The final predictor in the resulting hierarchy defines the quantities represented on the x-axis.

When an interaction contains more than two predictors, the decomposition is recursive. Elementary contrasts are first constructed from pairs of displayed quantities and are then combined into progressively higher-order contrasts until the complete model interaction contrast is obtained.

For `estimate_type = "predicted"`, predictions are population-level predictions based on the fixed effects. For generalized models on the response scale, uncertainty is propagated using the fixed-effect variance-covariance matrix and the multivariate delta method where appropriate. A methodological note describing only the settings relevant to the current call is printed when the function is run.

Value

When `show_coef = FALSE`, a `ggplot` object containing the upper interaction decomposition panel.

When `show_coef = TRUE`, a `patchwork` object containing the aligned upper and lower panels.

Interaction and coefficient selection

The arguments `interaction` and `coefficients` provide two complementary levels of control.

To display all coefficients associated with one interaction, specify only `interaction`, for example:

```
interaction = "A:B:C"
```

To display only selected coefficients from that interaction, specify both:

```
interaction = "A:B:C"
coefficients = c("A1:B2:C2", "A2:B2:C2")
```

If neither argument is supplied, the unique highest-order interaction is selected automatically.

When `interaction_info = TRUE`, the function prints the interactions found in the model together with valid coefficient selections, providing copy-ready syntax for subsequent calls.

Graphical styles

The three graphical styles differ in how much of the hierarchical decomposition is shown.

`"minimal"` emphasizes the observed or predicted quantities and elementary contrasts.

`"tree"` represents the same hierarchy through branching segments, emphasizing the recursive structure of the decomposition.

The lower coefficient panel, when requested, remains horizontally aligned with the corresponding contrast block in the upper panel for all three styles.

Examples

```
set.seed(123)

dat <- expand.grid(
  A = factor(c("A1", "A2", "A3")),
  B = factor(c("B1", "B2")),
  C = factor(c("C1", "C2")),
  rep = seq_len(20)
)

contrasts(dat$A) <- reverse_helmert(3)
contrasts(dat$B) <- MASS::contr.sdif(2)
contrasts(dat$C) <- stats::contr.treatment(2)

dat$y <- with(
  dat,
  10 +
    2 * (A == "A2") +
```

```
    4 * (A == "A3") +
    2 * (B == "B2") +
    2 * (C == "C2") +
    2 * (A == "A2" & B == "B2" & C == "C2") +
    rnorm(nrow(dat))
)

mod <- stats::lm(
  y ~ A * B * C,
  data = dat
)

# Automatically display the unique highest-order interaction.
plot_auto <- moplot_multpred(
  model = mod,
  data = dat
)

# Display all coefficients from a specific interaction.
plot_tree <- moplot_multpred(
  model = mod,
  interaction = "A:B:C",
  style = "tree"
)

# Display one specific coefficient using the minimal representation.
plot_minimal <- moplot_multpred(
  model = mod,
  interaction = "A:B:C",
  coefficients = "A1:B2-1:C2",
  style = "minimal"
)

# Use model-based predictions instead of observed means.
plot_predicted <- moplot_multpred(
  model = mod,
  interaction = "A:B:C",
  estimate_type = "predicted",
  prediction_scale = "response"
)

# The recommended size is stored in the plot and used by moplot_export().
attr(plot_tree, "moplot_export_size")

path <- moplot_export(
  plot_tree,
  filename = file.path(tempdir(), "moplot_multpred.png"),
  dpi = 72
)

unlink(path)
```

`moplot_onepred` *Plot contrasts for one categorical predictor*

Description

`moplot_onepred()` creates a graphical representation of the contrasts associated with one categorical predictor in a fitted model.

The upper panel displays the quantities compared by each contrast, together with their confidence intervals. The optional lower panel displays the corresponding model parameter, its confidence interval, and an approximate Cohen's d.

The horizontal position of each parameter in the lower panel is inherited from the midpoint of the corresponding contrast in the upper panel.

Usage

```
moplot_onepred(
  model,
  predictor,
  data = NULL,
  sig_level = 0.05,
  conf_level = 0.95,
  estimate_type = c("auto", "observed", "predicted"),
  prediction_scale = c("response", "link"),
  coefficient_scale = c("model", "interpreted"),
  per_difference = FALSE,
  bounds = NULL,
  hypotheses = NULL,
  show_coef = TRUE,
  show_data = FALSE,
  style = c("minimal", "publication"),
  caption = TRUE,
  digits = 2L,
  base_size = 13,
  panel_heights = c(2, 1)
)
```

Arguments

`model`	A fitted model supported by the package.
`predictor`	A single character string identifying the categorical predictor to display.
`data`	Optional data frame containing the variables used to fit `model`. If `NULL`, the data supplied in the model call are recovered; the model frame is used only when they are unavailable. The original variables are needed for model-based estimates when the model contains transformed terms such as `log(x)`. Rows of a supplied data frame that were excluded from the fit, through `subset` or missing values, are dropped, matching rows by name.

`sig_level` A numeric value between 0 and 1 defining the significance threshold used for the contrast labels. The default is `0.05`.

`conf_level` A numeric value between 0 and 1 specifying the confidence level. The default is `0.95`.

`estimate_type` Character string specifying how the quantities in the upper panel should be computed. Available options are:

`"auto"` Uses observed means when the fixed part of the model contains only the focal predictor, the model has no random effects and no unequal prior weights, and, for generalized models, `prediction_scale = "response"`. Model-based predictions are used otherwise: with other predictors or interactions, with random effects (observed means would ignore the clustering of the observations), with `weights` (observed means are unweighted), and for estimates on the link scale.

`"observed"` Uses observed means computed directly from the model data.

`"predicted"` Uses model-based fixed-effect predictions.

`prediction_scale`

Character string specifying the scale used for model-based predictions. Available options are `"response"` and `"link"`. The default is `"response"`.

`coefficient_scale`

Character string specifying the estimate displayed in the lower panel. Available options are:

`"model"` Displays the model coefficient on its original model scale.

`"interpreted"` Displays the interpreted estimate returned by `extract_param()`, such as an odds ratio or incidence rate ratio when appropriate.

`per_difference` Logical. Should the lower-panel coefficients be divided by their scale factor? The default is `FALSE`.

For some codings, such as Helmert and reverse Helmert contrasts or custom matrices, each model coefficient equals a scale factor `s` multiplied by the difference displayed in the upper panel (for example `s = 1/3` for the second Helmert contrast). When `per_difference = TRUE`, the lower panel reports `beta / s` and its confidence interval, so that both panels refer to the same difference; with `coefficient_scale = "interpreted"`, ratios are computed as `exp(beta / s)`, the ratio for the displayed comparison. Test statistics, p-values, and Cohen's d are unaffected. For generalized models, `beta / s` is a difference on the link scale. Codings with `s = 1` for every contrast (treatment, sum, sliding differences) are unaffected, and the argument is ignored with a warning for polynomial contrasts.

`bounds` Optional numeric vector `c(lower, upper)` giving the natural range of a response analyzed with a Gaussian (identity link) model, such as `c(1, 7)` for a Likert score or `c(0, Inf)` for a reaction time, so that confidence intervals remain within it. Ignored, with a warning, for generalized models. See `contrasts_onepred()`.

`hypotheses` Optional numeric matrix declaring the comparisons that the coefficients of `predictor` are intended to estimate, with one row per coefficient and one column per factor level. Each declared hypothesis is checked against the comparison that the coefficient estimates; the result is reported in a message and in the caption.

	The figure always shows the comparisons that the coefficients estimate. See `contrasts_onepred()`.
`show_coef`	Logical. Should the lower coefficient panel be displayed? The default is `TRUE`.
`show_data`	Logical. Should raw observations be displayed in the upper panel using horizontal jittering? The default is `FALSE`.
`style`	Character string specifying the graphical style. Available options are: `"minimal"` Uses a restrained grayscale palette. `"publication"` Uses a colorblind-safe palette and stronger visual emphasis.
`caption`	Controls whether a descriptive caption is printed in the console. It can be: • `TRUE`, to generate and print an automatic caption; • `FALSE` or `NULL`, to suppress the caption; • a single character string to print a custom caption. The caption is not added to the graphical object. Each sentence is printed on a separate console line so that the text can be copied and used independently of the plot.
`digits`	Number of decimal places used in the lower-panel labels. The default is `2`.
`base_size`	Base font size used in both panels. The default is `13`.
`panel_heights`	A numeric vector of length two specifying the relative heights of the upper and lower panels. The default is `c(2, 1)`. This argument is used only when `show_coef = TRUE`.

Details

The function displays one focal categorical predictor at a time, but the fitted model may contain additional predictors.

Models in which the focal predictor is involved in an interaction are not currently supported.

Treatment and Sum contrasts are displayed as independent pairs to improve readability. Other supported contrast types reuse repeated displayed quantities where appropriate.

Polynomial contrasts, generated by `stats::contr.poly()` with equally spaced or user-supplied `scores`, use a dedicated layout, because each component involves every level rather than a comparison between two quantities. The upper panel contains one facet per component (linear, quadratic, and so forth): each shows the level estimates with their confidence intervals and the curve implied by that component alone, drawn around the mean of the estimates on the linear-predictor scale and transformed to the scale of the estimates. Crosses mark the value of the curve at each level, which is `Polynomial_fit` in the output of `contrasts_onepred()`. Levels are spaced according to the scores of the coding. The lower panel shows each coefficient beneath the facet of its component. `per_difference` is ignored, with a warning, because polynomial coefficients have no scale factor.

No model-intercept line is displayed.

Value

When `show_coef = FALSE`, a `ggplot` object containing the upper panel.

When `show_coef = TRUE`, a `patchwork` object containing the aligned upper and lower panels.

In both cases the object carries recommended export dimensions, used by `moplot_export()` when `width` or `height` is not supplied.

Examples

```
set.seed(123)

dat <- data.frame(
  group = factor(
    rep(c("A", "B", "C"), each = 40),
    levels = c("A", "B", "C")
  ),
  y = c(
    rnorm(40, 0.0, 0.9),
    rnorm(40, 0.4, 0.9),
    rnorm(40, 0.9, 0.9)
  )
)

contrasts(dat$group) <- contr.helmert(3)

mod <- lm(
  y ~ group,
  data = dat
)

moplot_onepred(
  model = mod,
  predictor = "group"
)

moplot_onepred(
  model = mod,
  predictor = "group",
  show_data = TRUE,
  style = "publication"
)

moplot_onepred(
  model = mod,
  predictor = "group",
  show_coef = FALSE
)
```

---

reverse_helmert *Reverse Helmert Contrasts*

Description

Creates a Reverse Helmert contrast matrix by reversing both the rows and columns of a standard Helmert contrast matrix.

Usage

```
reverse_helmert(n, contrasts = TRUE)
```

Arguments

n	A factor, whose levels are used; a vector of level names; or a whole number specifying the number of factor levels (at least 2).
contrasts	Logical indicating whether to return contrasts (default TRUE).

Details

Reverse Helmert contrasts compare each factor level, except the last, with the mean of the subsequent levels.

The contrast matrix is constructed from `stats::contr.helmert()` by reversing the row order and subsequently the column order. This preserves the Helmert coding structure while reversing the direction in which the sequential comparisons are defined.

Value

A numeric matrix with n rows and n - 1 columns containing Reverse Helmert contrasts.

See Also

`stats::contr.helmert()`

Examples

```
reverse_helmert(3)
reverse_helmert(c("Control", "Trt1", "Trt2"))
reverse_helmert(factor(c("b", "a", "b", "c")))

dat <- data.frame(
  group = factor(rep(c("A", "B", "C"), each = 10)),
  y = rnorm(30)
)

contrasts(dat$group) <- reverse_helmert(3)
model <- lm(y ~ group, data = dat)
```

Index

contrasts_multpred, 2
contrasts_multpred(), *34, 35*
contrasts_onepred, 14
contrasts_onepred(), *4, 5*, *10*, *27*, *30*, *39, 40*

extract_param, 25

glmmTMB::glmmTMB(), *3*, *14*, *29*, *33*

lme4::glmer(), *3*, *14*, *29*, *33*
lme4::lmer(), *3*, *11*, *14*, *29*, *33*
lmerTest::as_lmerModLmerTest(), *11*

MASS::contr.sdif(), *20*, *30*
model_contrast_table, 29
moplot_export, 30
moplot_export(), *40*
moplot_multpred, 32
moplot_onepred, 38
moplot_onepred(), *30*

nlme::lme(), *3*, *14*, *29*, *33*

reverse_helmert, 41
reverse_helmert(), *20*, *30*

stats::contr.helmert(), *20*, *30*, *42*
stats::contr.poly(), *20*, *30*, *40*
stats::contr.SAS(), *20*
stats::contr.sum(), *20*, *29*
stats::contr.treatment(), *20*, *30*
stats::glm(), *3*, *14, 15*, *29*, *33*
stats::lm(), *3*, *14*, *29*, *33*