In this post, I want to demonstrate the different options to calculate and visualize marginal effects from mixed models.

**Marginal effects for mixed effects models**

Basically, the type of predictions, i.e. whether to account for the uncertainty of random effects or not, can be set with the type-argument.

**Marginal effects conditioned on fixed effects**

The default, type = "fe", means that predictions are on the *population-level* and do not account for the random effect variances.

library(ggeffects)

library(lme4)

data(sleepstudy)

m <- lmer(Reaction ~ Days + (1 + Days | Subject), data = sleepstudy)

pr <- ggpredict(m, "Days")

pr

#>

#> # Predicted values of Reaction

#> # x = Days

#>

#> x predicted std.error conf.low conf.high

#> 0 251.405 6.825 238.029 264.781

#> 1 261.872 6.787 248.570 275.174

#> 2 272.340 7.094 258.435 286.244

#> 3 282.807 7.705 267.705 297.909

#> 5 303.742 9.581 284.963 322.520

#> 6 314.209 10.732 293.174 335.244

#> 7 324.676 11.973 301.210 348.142

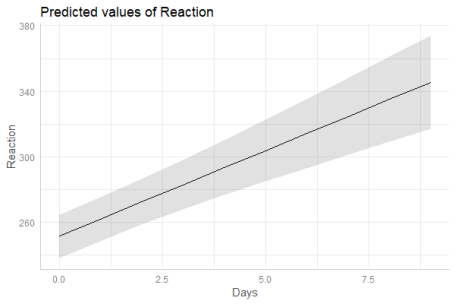
#> 9 345.611 14.629 316.939 374.283

#>

#> Adjusted for:

#> \* Subject = 0 (population-level)

plot(pr)



**Marginal effects conditioned on fixed effects with random effects uncertainty**

When type = "re", the predicted values *are still on the population-level*. However, the random effect variances are taken into account, meaning that the prediction interval becomes larger. More technically speaking, type = "re" accounts for the uncertainty of the fixed effects *conditional on the estimates* of the random-effect variances and conditional modes (BLUPs).

The random-effect variance is the mean random-effect variance. Calculation is based on the proposal from *Johnson et al. 2014*, which is applicable for mixed models with more complex random effects structures.

As can be seen, compared to the previous example with type = "fe", predicted values are identical (both on the population-level). However, standard errors, and thus the resulting confidence (or prediction) intervals are much larger .

pr <- ggpredict(m, "Days", type = "re")

pr

#>

#> # Predicted values of Reaction

#> # x = Days

#>

#> x predicted std.error conf.low conf.high

#> 0 251.405 41.769 169.539 333.271

#> 1 261.872 41.763 180.019 343.726

#> 2 272.340 41.814 190.386 354.293

#> 3 282.807 41.922 200.642 364.972

#> 5 303.742 42.307 220.822 386.661

#> 6 314.209 42.582 230.749 397.669

#> 7 324.676 42.912 240.571 408.781

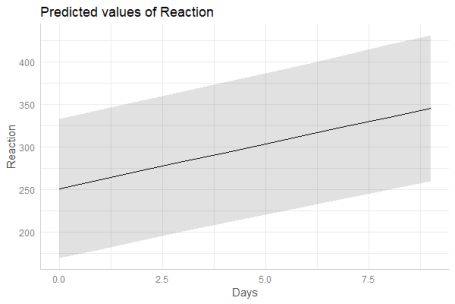
#> 9 345.611 43.727 259.907 431.315

#>

#> Adjusted for:

#> \* Subject = 0 (population-level)

plot(pr)



The reason why both type = "fe" and type = "re" return predictions at population-level is because ggpredict() returns predicted values of the response *at specific levels* of given model predictors, which are defined in the data frame that is passed to the newdata-argument (of predict()). The data frame requires data from all model terms, including random effect terms. This again requires to choose certain levels or values also for each random effect term, or to set those terms to zero or NA (for population-level). Since there is no general rule, which level(s) of random effect terms to choose in order to represent the random effects structure in the data, using the population-level seems the most clear and consistent approach.

**Marginal effects conditioned on fixed effects and specific group levels**

To get predicted values for a specific level of the random effect term, simply define this level in the condition-argument.

ggpredict(m, "Days", type = "re", condition = c(Subject = 330))

#>

#> # Predicted values of Reaction

#> # x = Days

#>

#> x predicted std.error conf.low conf.high

#> 0 275.096 41.769 193.230 356.961

#> 1 280.749 41.763 198.895 362.602

#> 2 286.402 41.814 204.448 368.355

#> 3 292.054 41.922 209.889 374.220

#> 5 303.360 42.307 220.440 386.280

#> 6 309.013 42.582 225.554 392.473

#> 7 314.666 42.912 230.561 398.772

#> 9 325.972 43.727 240.268 411.676

**Marginal effects based on simulations**

Finally, it is possible to obtain predicted values by simulating from the model, where predictions are based on simulate().

pr <- ggpredict(m, "Days", type = "sim")

pr

#>

#> # Predicted values of Reaction

#> # x = Days

#>

#> x predicted conf.low conf.high

#> 0 251.440 200.838 301.996

#> 1 261.860 212.637 311.678

#> 2 272.157 221.595 321.667

#> 3 282.800 233.416 332.738

#> 5 303.770 252.720 353.472

#> 6 314.146 264.651 363.752

#> 7 324.606 273.460 374.462

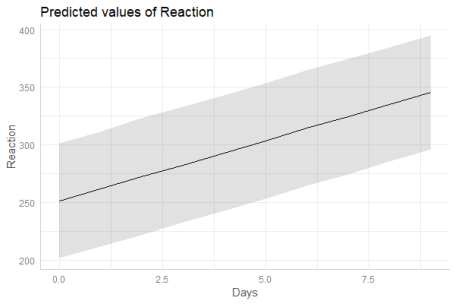
#> 9 345.319 295.069 394.735

#>

#> Adjusted for:

#> \* Subject = 0 (population-level)

plot(pr)



**Marginal effects for zero-inflated mixed models**

For zero-inflated mixed effects models, typically fitted with the **glmmTMB**-package, predicted values can be conditioned on

* the fixed effects of the conditional model only (type = "fe")
* the fixed effects and zero-inflation component (type = "fe.zi")
* the fixed effects of the conditional model only (population-level), taking the random-effect variances into account (type = "re")
* the fixed effects and zero-inflation component (population-level), taking the random-effect variances into account (type = "re.zi")
* all model parameters (type = "sim")

library(glmmTMB)

data(Salamanders)

m <- glmmTMB(

count ~ spp + mined + (1 | site),

ziformula = ~ spp + mined,

family = truncated\_poisson,

data = Salamanders

)

**Marginal effects conditioned on the count model**

Similar to mixed models without zero-inflation component, type = "fe" and type = "re" for glmmTMB-models (with zero-inflation) both return predictions on the population-level, where the latter option accounts for the uncertainty of the random effects. In short, predict(..., type = "link") is called (however, predictions are finally back-transformed to the response scale).

pr <- ggpredict(m, "spp")

pr

#>

#> # Predicted counts of count

#> # x = spp

#>

#> x predicted std.error conf.low conf.high

#> 1 0.935 0.206 0.624 1.400

#> 2 0.555 0.308 0.304 1.015

#> 3 1.171 0.192 0.804 1.704

#> 4 0.769 0.241 0.480 1.233

#> 5 1.786 0.182 1.250 2.550

#> 6 1.713 0.182 1.200 2.445

#> 7 0.979 0.196 0.667 1.437

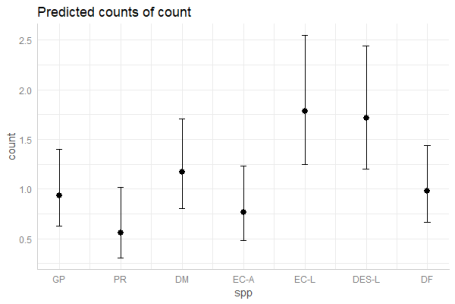
#>

#> Adjusted for:

#> \* mined = yes

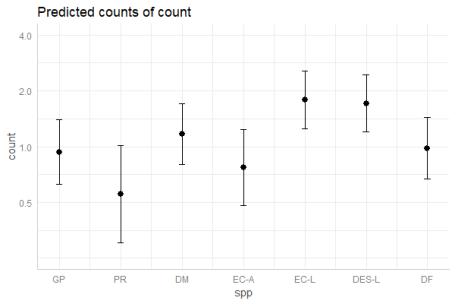
#> \* site = NA (population-level)

plot(pr)



For models with log-link, it make sense to use a log-transformed y-axis as well, to get proportional confidence intervals for the plot. You can do this by using the log.y-argument:

plot(pr, log.y = TRUE)



**Marginal effects conditioned on the count model with random effects uncertainty**

ggpredict(m, "spp", type = "re")

#>

#> # Predicted counts of count

#> # x = spp

#>

#> x predicted std.error conf.low conf.high

#> 1 0.935 0.309 0.510 1.714

#> 2 0.555 0.384 0.261 1.180

#> 3 1.171 0.300 0.650 2.107

#> 4 0.769 0.333 0.400 1.478

#> 5 1.786 0.294 1.004 3.175

#> 6 1.713 0.294 0.964 3.045

#> 7 0.979 0.303 0.541 1.772

#>

#> Adjusted for:

#> \* mined = yes

#> \* site = NA (population-level)

**Marginal effects conditioned on the count and zero-inflation model**

For type = "fe.zi", the predicted response value is the expected value mu\*(1-p) *without conditioning* on random effects. Since the zero inflation and the conditional model are working in “opposite directions”, a higher expected value for the zero inflation means a lower response, but a higher value for the conditional model means a higher response. While it is possible to calculate predicted values with predict(..., type = "response"), standard errors and confidence intervals can not be derived directly from the predict()-function. Thus, confidence intervals for type = "fe.zi" are based on quantiles of simulated draws from a multivariate normal distribution (see also *Brooks et al. 2017, pp.391-392* for details).

ggpredict(m, "spp", type = "fe.zi")

#>

#> # Predicted counts of count

#> # x = spp

#>

#> x predicted std.error conf.low conf.high

#> 1 0.138 0.045 0.052 0.224

#> 2 0.017 0.009 0.000 0.035

#> 3 0.245 0.072 0.109 0.381

#> 4 0.042 0.018 0.007 0.076

#> 5 0.374 0.108 0.166 0.582

#> 6 0.433 0.117 0.208 0.657

#> 7 0.205 0.063 0.082 0.328

#>

#> Adjusted for:

#> \* mined = yes

#> \* site = NA (population-level)

**Marginal effects conditioned on the count and zero-inflation model with random effects uncertainty**

For type = "re.zi", the predicted response value is the expected value mu\*(1-p), accounting for the random-effect variances. Prediction intervals are calculated in the same way as for type = "fe.zi", except that the mean random effect variance is considered for the confidence intervals.

ggpredict(m, "spp", type = "re.zi")

#>

#> # Predicted counts of count

#> # x = spp

#>

#> x predicted std.error conf.low conf.high

#> 1 0.138 0.235 0.032 0.354

#> 2 0.017 0.231 0.000 0.054

#> 3 0.245 0.243 0.065 0.609

#> 4 0.042 0.231 0.002 0.126

#> 5 0.374 0.257 0.098 0.932

#> 6 0.433 0.263 0.122 1.060

#> 7 0.205 0.239 0.054 0.510

#>

#> Adjusted for:

#> \* mined = yes

#> \* site = NA (population-level)

**Marginal effects simulated from zero-inflated models**

Finally, it is possible to obtain predicted values by simulating from the model, where predictions are based on simulate() (see *Brooks et al. 2017, pp.392-393* for details). To achieve this, use type = "sim".

ggpredict(m, "spp", type = "sim")

#>

#> # Predicted counts of count

#> # x = spp

#>

#> x predicted std.error conf.low conf.high

#> 1 1.089 1.288 0 4.131

#> 2 0.292 0.667 0 2.306

#> 3 1.520 1.550 0 5.241

#> 4 0.536 0.946 0 3.087

#> 5 2.212 2.125 0 7.153

#> 6 2.289 2.065 0 7.121

#> 7 1.314 1.367 0 4.697

#>

#> Adjusted for:

#> \* mined = yes

#> \* site = NA (population-level)

**References**

* Brooks ME, Kristensen K, Benthem KJ van, Magnusson A, Berg CW, Nielsen A, et al. glmmTMB Balances Speed and Flexibility Among Packages for Zero-inflated Generalized Linear Mixed Modeling. The R Journal. 2017;9: 378–400.
* Johnson PC, O’Hara RB. 2014. Extension of Nakagawa & Schielzeth’s R2GLMM to random slopes models. Methods Ecol Evol, 5: 944-946. (doi: 10.1111/2041-210X.12225)