HW3

Yuanrong Liu

2024-11-01

Packages Loaded

```
library(mlmRev)
library(ggplot2)
library(nlme)
library(lme4)
library(lmerTest)
library(glmmTMB)
library(broom.mixed)
library(pbkrtest)
library(performance)
library(DHARMa)
library(purrr)
library(dplyr)
library(dotwhisker)
```

(a) Fit a linear mixed model

- `attain` as the **response variable**, `social` (as a **factor**), `sex`, and `verbal` as **fixed effects**, `primary` as the **grouping variable**.

```
# Convert 'social' to a factor
ScotsSec$social <- as.factor(ScotsSec$social)

# Fit the linear mixed model
model_lmm <- lmer(attain ~ 1 + social + sex + verbal +
                    (1 + social + sex + verbal | primary),
                  data = ScotsSec)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
## Warning: Model failed to converge with 1 negative eigenvalue: -3.4e+00
```

(b) Find the random-effects term with the smallest estimated variance

```r
summary(model_lmm)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: attain ~ 1 + social + sex + verbal + (1 + social + sex + verbal |
##     primary)
##    Data: ScotsSec
##
## REML criterion at convergence: 14664.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.0660 -0.6790 -0.0001  0.6798  4.0138
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  primary  (Intercept) 0.367214 0.60598
##           social1     0.107446 0.32779  -0.89
##           social20    0.330082 0.57453  -0.57  0.88
##           social31    0.440613 0.66379  -0.36  0.43  0.39
##           sexF        0.318849 0.56467  -0.63  0.46  0.15 -0.19
##           verbal      0.000117 0.01082   0.77 -0.92 -0.86 -0.73 -0.22
##  Residual             3.944074 1.98597
## Number of obs: 3435, groups:  primary, 148
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 5.536e+00  8.161e-02 1.048e+02  67.832  < 2e-16 ***
## social1     1.411e+00  1.634e-01 1.047e+03   8.637  < 2e-16 ***
## social20    1.183e+00  1.072e-01 1.073e+02  11.039  < 2e-16 ***
## social31    5.470e-01  1.461e-01 8.687e+01   3.745 0.000324 ***
## sexF        1.484e-01  8.824e-02 9.721e+01   1.682 0.095864 .
## verbal      1.515e-01  2.967e-03 2.422e+02  51.075  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) socil1 socl20 socl31 sexF
## social1  -0.304
## social20 -0.463  0.246
## social31 -0.300  0.147  0.225
## sexF     -0.626  0.057  0.071 -0.024
## verbal    0.381 -0.251 -0.308 -0.223 -0.125
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

- I will remove `verbal` as a **random effect** because it has the lowest variance (`0.000117`) among all the random effects in the model.

Refit the model

```
# Simplified model without random slope for 'verbal'
model_lmm2 <- lmer(attain ~ 1 + social + sex + verbal +
                     (1 + social + sex | primary),
                   data = ScotsSec)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## unable to evaluate scaled gradient
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge: degenerate Hessian with 2 negative eigenvalues
```

```
## Warning: Model failed to converge with 2 negative eigenvalues: -3.2e-02
## -2.8e+00
```

Find the random-effects term with the smallest estimated variance

```
summary(model_lmm2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: attain ~ 1 + social + sex + verbal + (1 + social + sex | primary)
##    Data: ScotsSec
##
## REML criterion at convergence: 14672.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1239 -0.6816 -0.0067  0.6791  4.0763
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  primary  (Intercept) 0.29619  0.5442
##           social1     0.02241  0.1497   -0.91
##           social20    0.21048  0.4588   -0.43  0.76
##           social31    0.28136  0.5304   -0.20  0.13 -0.02
##           sexF        0.30301  0.5505   -0.61  0.46  0.03 -0.33
##  Residual             3.97429  1.9936
## Number of obs: 3435, groups:  primary, 148
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 5.552e+00  7.756e-02 1.126e+02  71.586  < 2e-16 ***
## social1     1.359e+00  1.612e-01 2.897e+03   8.431  < 2e-16 ***
## social20    1.163e+00  1.019e-01 1.179e+02  11.412  < 2e-16 ***
## social31    5.342e-01  1.403e-01 7.908e+01   3.809 0.000274 ***
## sexF        1.509e-01  8.751e-02 9.958e+01   1.724 0.087734 .
## verbal      1.520e-01  2.778e-03 3.383e+03  54.713  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) socil1 socl20 socl31 sexF
## social1  -0.249
## social20 -0.407  0.195
## social31 -0.252  0.121  0.146
## sexF     -0.620  0.025  0.033 -0.047
## verbal    0.209 -0.201 -0.165 -0.119 -0.083
## optimizer (nloptwrap) convergence code: 0 (OK)
## unable to evaluate scaled gradient
## Model failed to converge: degenerate  Hessian with 2 negative eigenvalues
```

- I am going to remove `social` as a **random effect** since `social1` has the lowest variance (`0.02241`) among the remaining random slopes.

Refit the model

```
# Fit the model without the random slope for 'social'
model_lmm3 <- lmer(attain ~ 1 + social + sex + verbal +
                     (1 + sex | primary),
                   data = ScotsSec)
```

```
# Check the summary to see if the convergence issue is resolved
summary(model_lmm3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: attain ~ 1 + social + sex + verbal + (1 + sex | primary)
##    Data: ScotsSec
##
## REML criterion at convergence: 14680.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2075 -0.6886 -0.0024  0.6880  3.9931
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  primary  (Intercept) 0.2405   0.4904
##           sexF        0.2868   0.5355   -0.66
##  Residual             4.0306   2.0076
## Number of obs: 3435, groups:  primary, 148
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 5.553e+00  7.436e-02 1.646e+02  74.677  < 2e-16 ***
## social1     1.331e+00  1.619e-01 3.366e+03   8.217 2.95e-16 ***
## social20    1.130e+00  9.105e-02 3.368e+03  12.410  < 2e-16 ***
## social31    5.182e-01  1.279e-01 3.405e+03   4.052 5.19e-05 ***
```

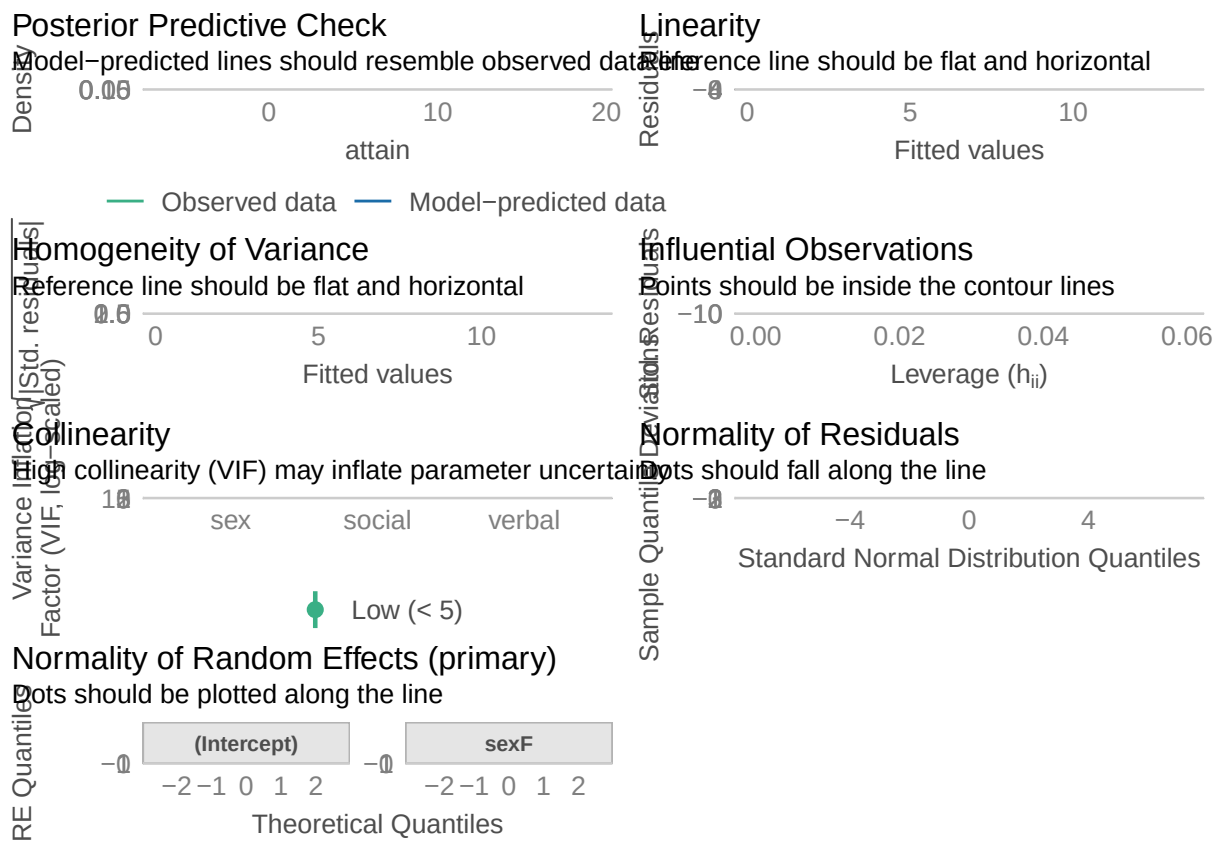
```
## sexF        1.468e-01  8.689e-02 1.002e+02   1.690   0.0941 .
## verbal      1.521e-01  2.780e-03 3.390e+03  54.721  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) socil1 socl20 socl31 sexF
## social1  -0.203
## social20 -0.335  0.191
## social31 -0.241  0.129  0.201
## sexF     -0.642  0.000  0.025  0.036
## verbal    0.219 -0.198 -0.182 -0.128 -0.083
```

- Now, only `sex` has a random slope and a random intercept across primary schools.
- The random effects summary shows that the model estimates the variability in intercepts and the effect of `sex` across schools, which appears to be a good fit based on the variance estimates.
- The fixed effects are statistically significant (except `sexF`, which is borderline at p = `0.0941`).
- The intercept of fixed effects represents the estimated average attainment (`attain`) for individuals in the reference group (`sexM` and `social0`), when `verbal` is set to 0.

(c) Model diagnostics

Run diagnostics using performance package

```
check_model(model_lmm3)
```



Posterior Predictive Check There are some notable discrepancies: one peak in the model-predicted data but two peaks in the observed data. This suggests that the model does not fully capture the observed distribution of the response variable, indicating that the fit could be improved.

Linearity The green line is generally flat for fitted values between 0 and 10, suggesting a well-captured linear relationship. However, the smoothed line shows a slight downward trend as fitted values increase beyond 10, indicating some non-linearity that is not captured by the model.

Homogeneity of Variance The reference line in green shows a general trend that is relatively flat, indicating that the model is mostly meeting the homoscedasticity assumption. There is, however, some slight curvature in the green line, especially for fitted values greater than 10, which suggests a minor deviation from homoscedasticity.

Influential Observations All points fall within the contour lines, suggesting that no observations have an undue influence on the model.

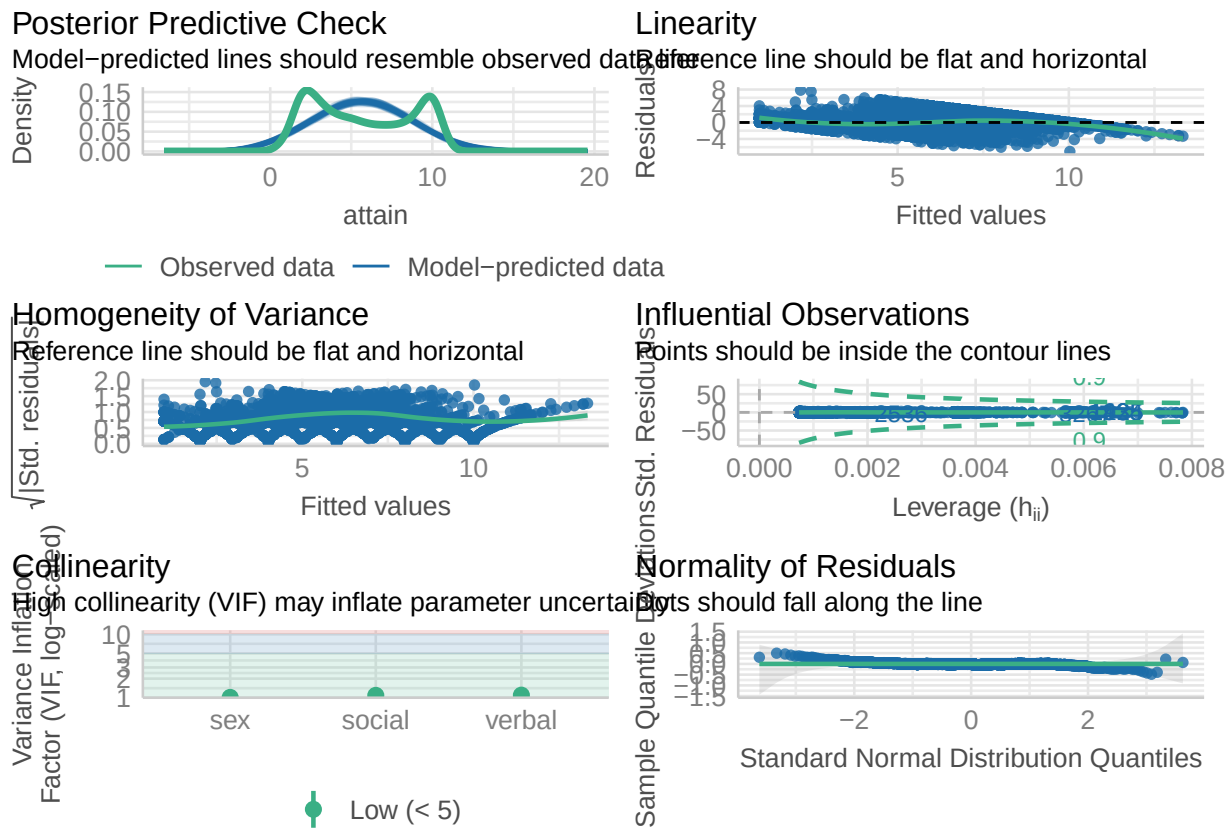
Collinearity The VIF values are close to 1, well below the threshold of 5, suggesting a minimal correlation among the predictors in the model. The predictors are sufficiently independent to provide reliable coefficient estimates, and multicollinearity is not an issue in the model.

Normality of Residuals Most of the residuals lie close to the green reference line, suggesting that the residuals are roughly normally distributed. All residuals fall within the gray confidence band, which is a good indication that the deviations from normality are not extreme or concerning.

Normality of Random Effects For both the `intercept` and `sexF`, the dots appear to fall along the reference line, suggesting that the random effects are approximately normally distributed.

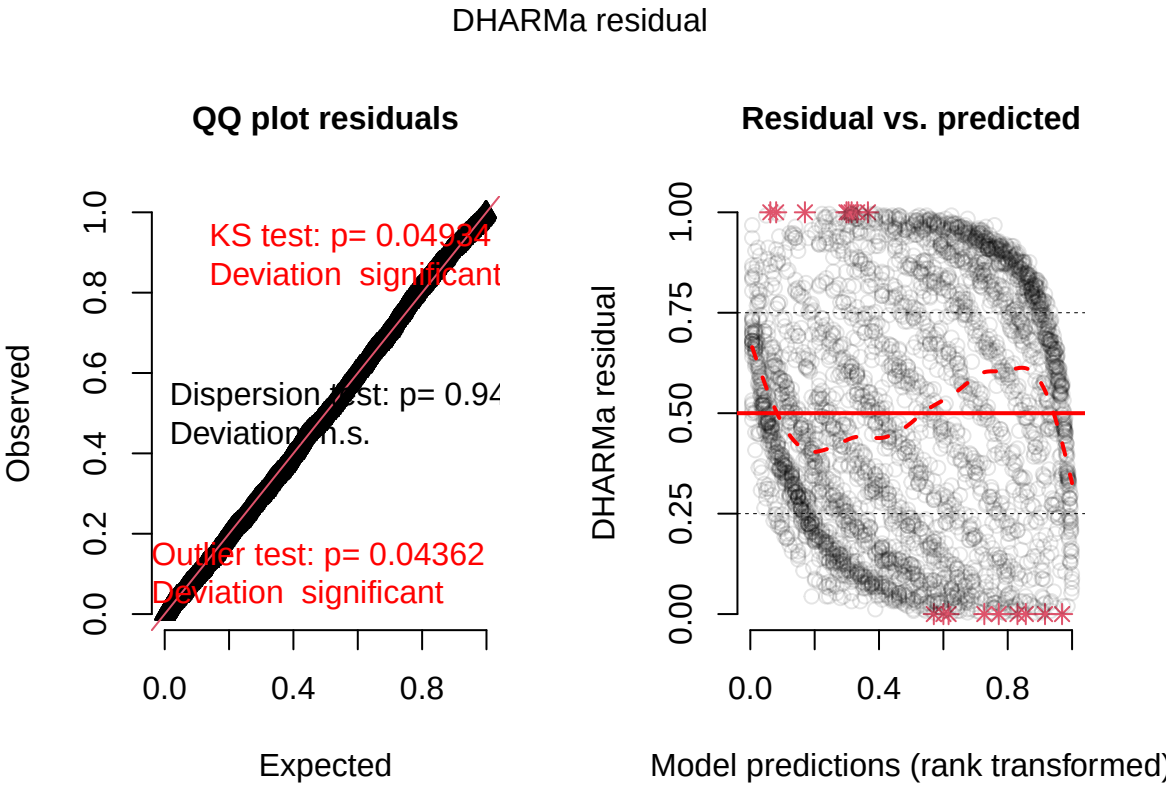
Difference from a linear model `Normality of Random Effects` has been added to linear mixed models.

```
check_model(lm(attain ~ 1 + social + sex + verbal, data = ScotsSec))
```



Run diagnostics using DHARMa

```
# Simulate residuals using DHARMa
simulation_output <- simulateResiduals(fittedModel = model_lmm3)

# Plot diagnostic plots
plot(simulation_output)
```



Q-Q Plot of Residuals The p-value of `0.04934` from the Kolmogorov-Smirnov (KS) Test indicates that there is a significant deviation from the expected uniform distribution at the 5% significance level. The model might not fully capture some aspect of the data.

Residual vs. Predicted The red dashed line deviates from the flat reference line, especially at higher predicted values, indicating some pattern in the residuals. There are red asterisks at the top and bottom, which indicate outliers or observations with standardized residuals beyond expected bounds.

(d) Fit the model with nlme::lme

```
model_lme <- lme(fixed = attain ~ 1 + social + sex + verbal,
                 random = ~ 1 + sex | primary,
                 data = ScotsSec,
                 method = "REML")
```

Fit the model with glmmTMB

```
model_glmmTMB <- glmmTMB(attain ~ 1 + social + sex + verbal +
                           (1 + sex | primary),
                         data = ScotsSec,
                         REML = TRUE)
```

(e) Create a named list of the three models

```
mod_list <- list(lme = model_lme,
                 glmmTMB = model_glmmTMB,
                 lmer = model_lmm3)

# Compare the overall model
model_comparison <- purrr::map_dfr(mod_list, glance, .id = "model")

# Print the model comparison
print(model_comparison)
```

```
## # A tibble: 3 x 9
##   model    nobs sigma logLik    AIC    BIC deviance df.residual REMLcrit
##   <chr>   <int> <dbl>  <dbl>  <dbl>  <dbl>    <dbl>       <int>    <dbl>
## 1 lme      3435  2.01 -7340. 14700. 14762.       NA          NA       NA
## 2 glmmTMB  3435  2.01 -7340. 14700. 14762.   13445.        3425       NA
## 3 lmer     3435  2.01 -7340. 14700. 14762.       NA          3425   14680.
```

(f) Extract the coefficients

```
# Extract the fixed effects coefficients using purrr::map_dfr and broom.mixed::tidy
coefficients <- purrr::map_dfr(mod_list,
                               ~tidy(., effects = "fixed"),
                               .id = "model") |> dplyr::arrange(term)
```

Qualitatively compare the coefficients

```
# Define the comparison function that prints term names explicitly
compare_models_verbose <- function(values1, values2, terms) {
  for (i in seq_along(values1)) {
    term_name <- terms[i]
    comparison <- ""

    if (all.equal(values1[i], values2[i], tolerance = 1e-4) == TRUE) {
      comparison <- "Identical or practically identical"
    } else if (all.equal(values1[i], values2[i], tolerance = 0.01) == TRUE) {
      comparison <- "Very similar"
    } else if (all.equal(values1[i], values2[i], tolerance = 0.1) == TRUE) {
      comparison <- "Slightly different"
    } else {
      comparison <- "Different"
    }

    cat(term_name, ":", comparison, "\n")
  }
}
```

```r
# Split the coefficients by model
coeff_lme <- coefficients %>% filter(model == "lme")
coeff_lmer <- coefficients %>% filter(model == "lmer")
coeff_glmmTMB <- coefficients %>% filter(model == "glmmTMB")
```

```r
# Compare Estimates
cat("Comparison of Estimates (lme vs lmer):\n")
```

```
## Comparison of Estimates (lme vs lmer):
```

```r
compare_models_verbose(coeff_lme$estimate, coeff_lmer$estimate, coeff_lme$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Identical or practically identical
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
cat("\nComparison of Estimates (lme vs glmmTMB):\n")
```

```
##
## Comparison of Estimates (lme vs glmmTMB):
```

```r
compare_models_verbose(coeff_lme$estimate, coeff_glmmTMB$estimate, coeff_lme$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Identical or practically identical
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
cat("\nComparison of Estimates (lmer vs glmmTMB):\n")
```

```
##
## Comparison of Estimates (lmer vs glmmTMB):
```

```r
compare_models_verbose(coeff_lmer$estimate, coeff_glmmTMB$estimate, coeff_lmer$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Identical or practically identical
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
# Compare Standard Errors
cat("\nComparison of Standard Errors (lme vs lmer):\n")
```

```
##
## Comparison of Standard Errors (lme vs lmer):
```

```r
compare_models_verbose(coeff_lme$std.error, coeff_lmer$std.error, coeff_lme$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Identical or practically identical
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
cat("\nComparison of Standard Errors (lme vs glmmTMB):\n")
```

```
##
## Comparison of Standard Errors (lme vs glmmTMB):
```

```r
compare_models_verbose(coeff_lme$std.error, coeff_glmmTMB$std.error, coeff_lme$term)
```

```
## (Intercept) : Very similar
## sexF : Very similar
## social1 : Slightly different
## social20 : Slightly different
## social31 : Very similar
## verbal : Very similar
```

```r
cat("\nComparison of Standard Errors (lmer vs glmmTMB):\n")
```

```
##
## Comparison of Standard Errors (lmer vs glmmTMB):
```

```r
compare_models_verbose(coeff_lmer$std.error, coeff_glmmTMB$std.error, coeff_lmer$term)
```

```
## (Intercept) : Very similar
## sexF : Very similar
## social1 : Slightly different
## social20 : Slightly different
## social31 : Very similar
## verbal : Very similar
```

```r
# Compare DFs
cat("\nComparison of df (lme vs lmer):\n")
```

```
##
## Comparison of df (lme vs lmer):
```

```r
compare_models_verbose(coeff_lme$df, coeff_lmer$df, coeff_lme$term)
```

```
## (Intercept) : Different
## sexF : Different
## social1 : Slightly different
## social20 : Slightly different
## social31 : Slightly different
## verbal : Slightly different
```

```r
# Compare P-values
## Since the df of glmmTMB does not exist, just compare lme and lmer
cat("\nComparison of P-values (lme vs lmer):\n")
```

```
##
## Comparison of P-values (lme vs lmer):
```

```r
compare_models_verbose(coeff_lme$p.value, coeff_lmer$p.value, coeff_lme$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Slightly different
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
cat("\nComparison of P-values (lme vs glmmTMB):\n")
```

```
##
## Comparison of P-values (lme vs glmmTMB):
```

```r
compare_models_verbose(coeff_lme$p.value, coeff_glmmTMB$p.value, coeff_lme$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Slightly different
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

```r
cat("\nComparison of P-values (lmer vs glmmTMB):\n")
```

```
##
## Comparison of P-values (lmer vs glmmTMB):
```

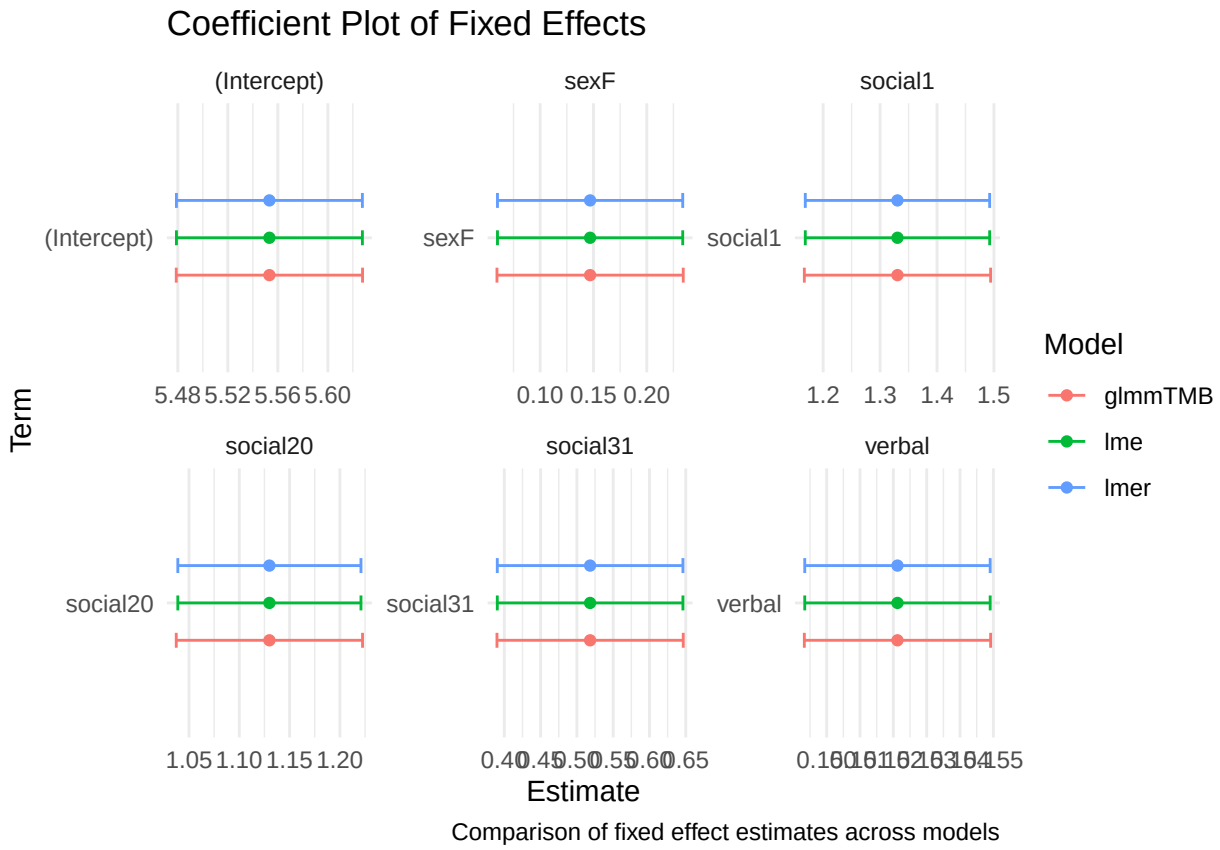
```r
compare_models_verbose(coeff_lmer$p.value, coeff_glmmTMB$p.value, coeff_lmer$term)
```

```
## (Intercept) : Identical or practically identical
## sexF : Slightly different
## social1 : Identical or practically identical
## social20 : Identical or practically identical
## social31 : Identical or practically identical
## verbal : Identical or practically identical
```

- The comparisons show that `estimates` are **"identical or practically identical"** across the models. The `standard errors` and `p-values` were mostly very similar, with slight differences in some cases.

(g) Generate a coefficient plot of the fixed effects

```
# Create a coefficient plot using ggplot2
ggplot(coefficients, aes(x = estimate, y = term, color = model)) +
  geom_point(position = position_dodge(width = 0.5)) +
  geom_errorbarh(aes(xmin = estimate - std.error, xmax = estimate + std.error),
                 height = 0.2, position = position_dodge(width = 0.5)) +
  facet_wrap(~term, scales = "free") +
  theme_minimal() +
  labs(title = "Coefficient Plot of Fixed Effects",
       x = "Estimate",
       y = "Term",
       caption = "Comparison of fixed effect estimates across models",
       color = "Model")
```



- The estimates across the models are **identical or practically identical**, as previously observed during the qualitative comparison.

(h) Compare denominator df

```
# Compute coefficients with Satterthwaite approximation
satterthwaite_summary <- coef(summary(model_lmm3, ddf = "Satterthwaite"))
satterthwaite_df <- satterthwaite_summary[, "df"]
cat("Satterthwaite degrees of freedom:\n")
```

```
## Satterthwaite degrees of freedom:
```

```
print(satterthwaite_df)
```

```
## (Intercept)     social1    social20    social31        sexF      verbal
##    164.5601   3365.7651   3367.5014   3405.3106    100.2033   3390.0592
```

```
# Compute coefficients with Kenward-Roger approximation
kenward_roger_summary <- coef(summary(model_lmm3, ddf = "Kenward-Roger"))
kenward_roger_df <- kenward_roger_summary[, "df"]
cat("Kenward-Roger degrees of freedom:\n")
```

```
## Kenward-Roger degrees of freedom:
```

```
print(kenward_roger_df)
```

```
## (Intercept)     social1    social20    social31        sexF      verbal
##    163.8634   3373.6117   3374.0254   3408.5651    111.6036   3394.5626
```

```
# Compare Satterthwaite and Kenward-Roger degrees of freedom
cat("Comparison of Degrees of Freedom (Satterthwaite vs Kenward-Roger):\n")
```

```
## Comparison of Degrees of Freedom (Satterthwaite vs Kenward-Roger):
```

```
compare_models_verbose(satterthwaite_df, kenward_roger_df, names(satterthwaite_df))
```

```
## (Intercept) : Very similar
## social1 : Very similar
## social20 : Very similar
## social31 : Very similar
## sexF : Different
## verbal : Very similar
```

- For the majority of the fixed effect terms (`(Intercept)`, `social1`, `social20`, `social31`, `verbal`), the degrees of freedom calculated by the **Satterthwaite** and **Kenward-Roger** methods are classified as "Very similar". The degrees of freedom for `sexF` are classified as **Different** between Satterthwaite and Kenward-Roger.
- This suggests that, for all terms except `sexF`, both methods produce nearly equivalent estimates for the degrees of freedom, indicating consistency in how the two methods adjust for the model's random effects.

```r
# Extract degrees of freedom from the lme model
lme_summary <- summary(model_lme)
lme_df <- lme_summary$tTable[, "DF"]
lme_df
```

```
## (Intercept)     social1    social20    social31        sexF      verbal
##        3282        3282        3282        3282        3282        3282
```

```r
# Compare lme vs Satterthwaite degrees of freedom
cat("Comparison of Degrees of Freedom (lme vs Satterthwaite):\n")
```

```
## Comparison of Degrees of Freedom (lme vs Satterthwaite):
```

```r
compare_models_verbose(lme_df, satterthwaite_df, names(lme_df))
```

```
## (Intercept) : Different
## social1 : Slightly different
## social20 : Slightly different
## social31 : Slightly different
## sexF : Different
## verbal : Slightly different
```

```r
# Compare lme vs Kenward-Roger degrees of freedom
cat("\nComparison of Degrees of Freedom (lme vs Kenward-Roger):\n")
```

```
##
## Comparison of Degrees of Freedom (lme vs Kenward-Roger):
```

```r
compare_models_verbose(lme_df, kenward_roger_df, names(lme_df))
```

```
## (Intercept) : Different
## social1 : Slightly different
## social20 : Slightly different
## social31 : Slightly different
## sexF : Different
## verbal : Slightly different
```

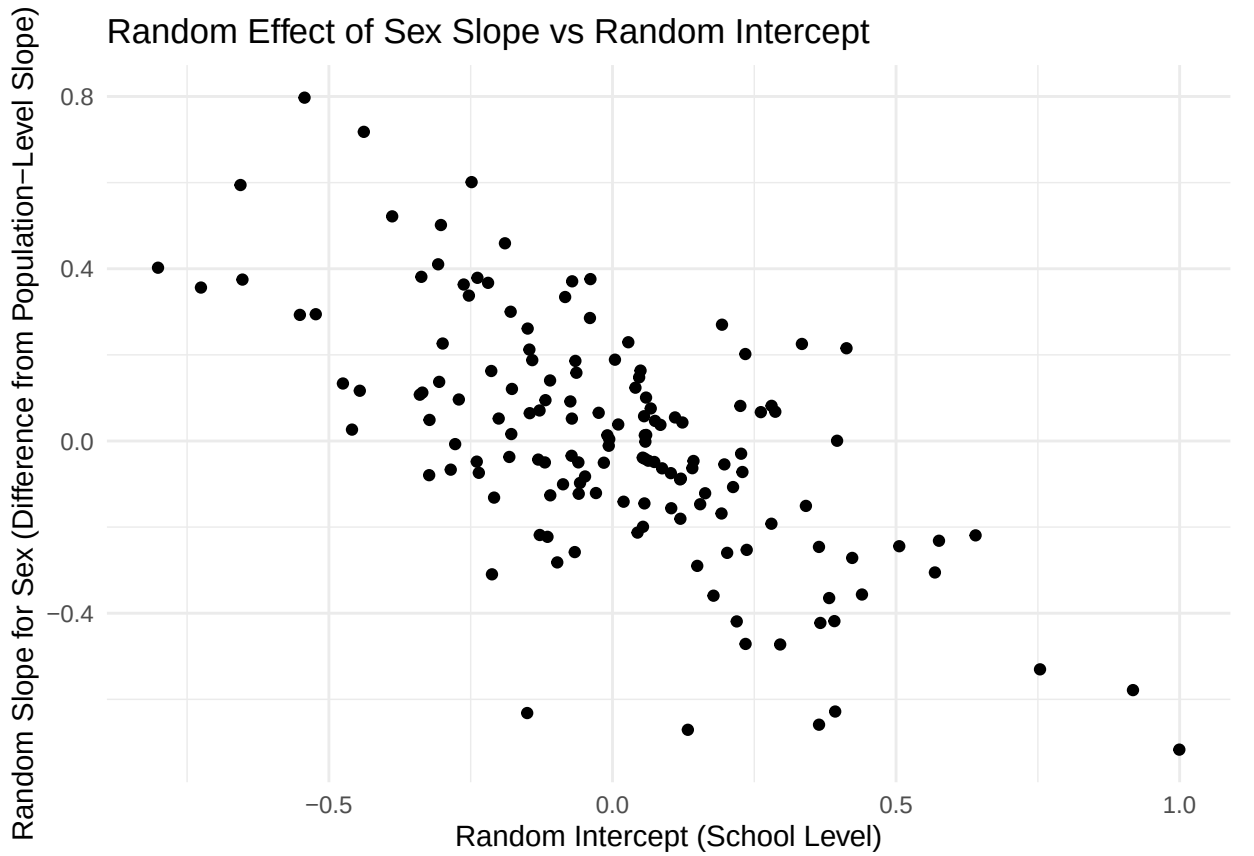
- `Intercept` and `sexF` have a **Different** df compared to both **Satterthwaite** and **Kenward-Roger**. This suggests that the **lme** model is producing degrees of freedom significantly different for the overall mean and the gender effect.

(i) Plot the random effect of sex for each school against the corresponding random intercept

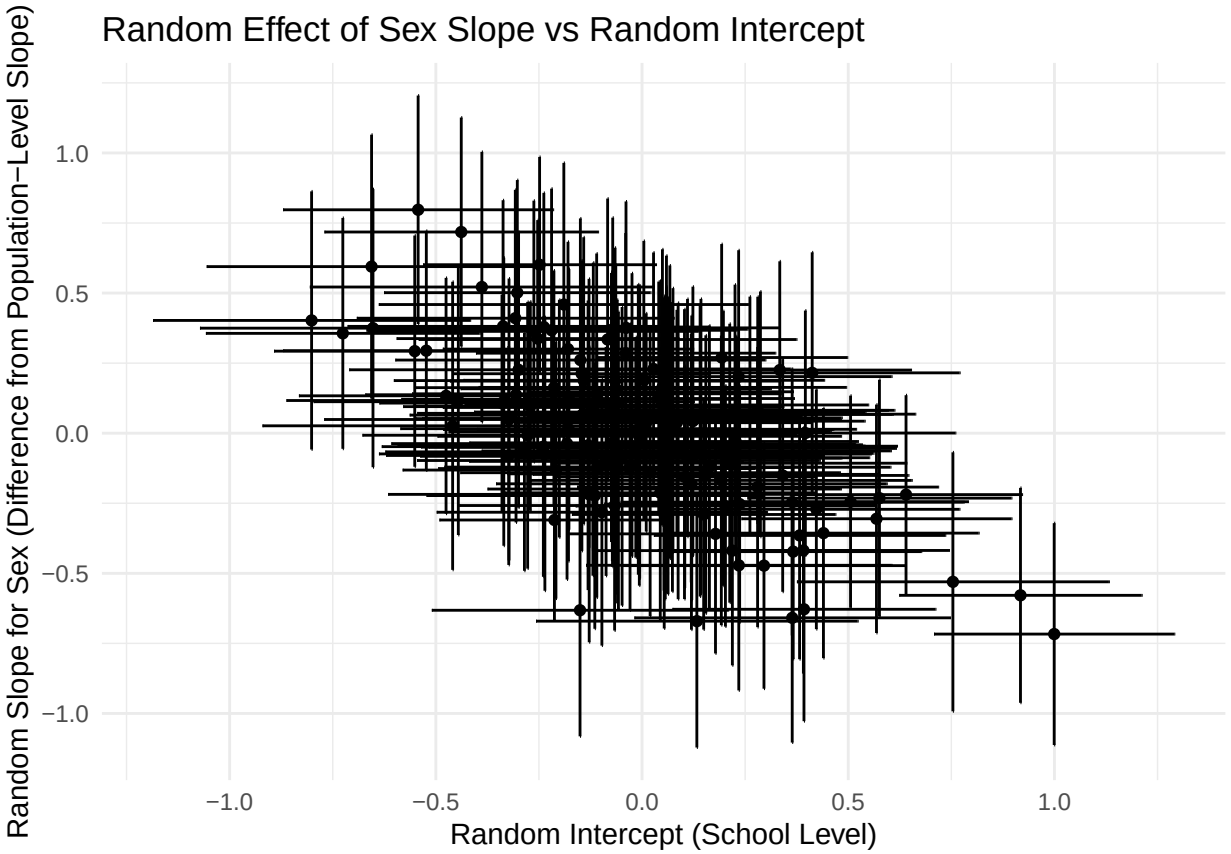
```r
# Extract random effects using broom.mixed::tidy
random_effects <- tidy(model_lmm3, effects = "ran_vals")

# Filter random effects for 'sex' slope and intercept for each school
```

```
random_intercepts <- random_effects[random_effects$term == "(Intercept)" &
                                    random_effects$group == "primary", ]
random_slopes <- random_effects[random_effects$term == "sexF" &
                                random_effects$group == "primary", ]

# Merge intercepts and slopes by school
random_effects_combined <- merge(random_intercepts,
                                 random_slopes,
                                 by = "level",
                                 suffixes = c("_intercept", "_slope"))
```

```
# Plot the random effect of sex slope against the random intercept
plot <- ggplot(random_effects_combined, aes(x = estimate_intercept, y = estimate_slope)) +
  geom_point() +
  labs(
    title = "Random Effect of Sex Slope vs Random Intercept",
    x = "Random Intercept (School Level)",
    y = "Random Slope for Sex (Difference from Population-Level Slope)"
  ) +
  theme_minimal()

# Display the plot
print(plot)
```



```r
# Plot with error bars
plot <- ggplot(random_effects_combined, aes(x = estimate_intercept, y = estimate_slope)) +
  geom_point() +
  geom_errorbar(aes(ymin = estimate_slope - std.error_slope,
                    ymax = estimate_slope + std.error_slope),
                width = 0) +
  geom_errorbarh(aes(xmin = estimate_intercept - std.error_intercept,
                     xmax = estimate_intercept + std.error_intercept),
                 height = 0) +
  labs(
    title = "Random Effect of Sex Slope vs Random Intercept",
    x = "Random Intercept (School Level)",
    y = "Random Slope for Sex (Difference from Population-Level Slope)"
  ) +
  theme_minimal()

# Display the plot
print(plot)
```



(j) Explain why not to treat social as a random-effects grouping variable

```r
summary(ScotsSec$social)
```

```
##    0    1   20   31
## 2247  185  706  297
```

- `social` is a factor with four levels: `0`, `1`, `20`, `31`. These levels are fixed, exhaustive categories, instead of randomly sampled groups. They are not drawn from a larger population.
- We are interested in the specific effects of each `social` level on the attainment, instead of the variability across random levels of 'social'.
- If we treat `social` as a random effect, the model would only estimate a variance component, which would make it difficult to make specific comparisons about the influence of each social status on attainment.
- The correct approach is to treat `social` as a fixed effect, (correct approach, `model_lmm3 <- lmer(attain ~ 1 + social + sex + verbal + (1 + sex | primary), data = ScotsSec)`), for individual `j` in primary group `i`:

$$\text{attain}_{ij} = (\beta_0 + u_{0,\text{primary}[i]} + u_{1,\text{primary}[i]} \cdot \text{sex}_{ij}) + \beta_1 \cdot \text{social}_{ij} + \beta_2 \cdot \text{sex}_{ij} + \beta_3 \cdot \text{verbal}_{ij} + \epsilon_{ij}$$

(k) Explain why not to leave the fixed effect of 'sex' out of the model

- Without the fixed effect of `sex`, there model does not provide a baseline estimate for the overall population-level effect of `sex` on `attainment`, so we cannot estimate an overall population level of `sex` on `attainment`.
- Instead, the model only estimates how the effect of 'sex' varies across schools, but not in the general population. This makes the interpretation limited because it does not tell us what the typical impact of `sex` is, only how it changes from school to school.
- Random effects are deviations from an average effect. If there is no fixed effect of sex, it will becomes difficult to interpret what the random slopes for `sex` deviates from, without a baseline effect (reference point) of sex.

(i) Fit reduced models

```
model_lmm3 <- lmer(attain ~ 1 + social + sex + verbal +
                     (1 + sex | primary),
                   data = ScotsSec,
                   REML = TRUE)

model_lmm4 <- lmer(attain ~ 1 + social + sex + verbal +
                     (1 | primary), # with random effects intercept variation only
                   data = ScotsSec,
                   REML = TRUE)
```

Compare models

```
anova(model_lmm4, model_lmm3)
```

```
## refitting model(s) with ML (instead of REML)

## Data: ScotsSec
## Models:
## model_lmm4: attain ~ 1 + social + sex + verbal + (1 | primary)
## model_lmm3: attain ~ 1 + social + sex + verbal + (1 + sex | primary)
##            npar   AIC   BIC  logLik deviance  Chisq Df Pr(>Chisq)
## model_lmm4    8 14683 14732 -7333.4    14667
## model_lmm3   10 14676 14737 -7328.0    14656 10.841  2   0.004425 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Extract values for comparison
values1 <- c(AIC(model_lmm4), logLik(model_lmm4))
values2 <- c(AIC(model_lmm3), logLik(model_lmm3))
terms <- c("AIC", "log-likelihood")

# Use the function to compare
compare_models_verbose(values1, values2, terms)
```

```
## AIC : Very similar
## log-likelihood : Very similar
```

- The AICs (`Akaike Information Criterion`) of both models are very similar: `14683` for `model_lmm4` and `14676` for `model_lmm3`. `model_lmm3` has a slightly lower AIC, suggesting that it fits the data better.
- The `log-likelihood` are also very similar: `-7333.4` for `model_lmm4` and `-7328.0` for `model_lmm3`. The log-likelihood of `model_lmm3` is slightly higher, suggesting a better fit.
- The likelihood ratio test (`LRT`) yielded a statistically significant result (`p-value = 0.004425`), indicating that including a random slope for `sex` across `primary` schools in `model_lmm3` significantly improves the model fit.
- There is significant variation in the effect of `sex` on `attainment` across different primary schools.

Use parametric bootstrapping to compare models with pbkrtest::PBmodcomp

To address convergence failures, we can scale the data `verbal` and change the optimizer (according to ChatGPT).

```
ScotsSec$verbal <- scale(ScotsSec$verbal)
# Then refit the model
model_lmm3 <- lmer(attain ~ 1 + social + sex + verbal + (1 + sex | primary),
                   data = ScotsSec,
                   REML = TRUE,
                   control = lmerControl(optCtrl = list(maxfun = 1e6), optimizer = "bobyqa"))

model_lmm4 <- lmer(attain ~ 1 + social + sex + verbal + (1 | primary),
                   data = ScotsSec,
                   REML = TRUE,
                   control = lmerControl(optCtrl = list(maxfun = 1e6), optimizer = "bobyqa"))
```

```r
# Compare models using parametric bootstrap with 1000 simulations
pb_comp <- PBmodcomp(model_lmm3, model_lmm4, nsim = 1000)

# Print results
print(pb_comp)
```

```
## Bootstrap test; time: 122.41 sec; samples: 1000; extremes: 1;
## large : attain ~ 1 + social + sex + verbal + (1 + sex | primary)
## attain ~ 1 + social + sex + verbal + (1 | primary)
##          stat df  p.value
## LRT    10.841  2 0.004425 **
## PBtest 10.841    0.001998 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Reference

- OpenAI, ChatGPT (2024). Assistance with R programming and output analysis.

Appendix

- Here is a list of prompts I used in **ChatGPT 4o with canvas**:

1. How to use `DHARMa` to run diagnostics on a model?
2. How to fit a model with `nlme::lme()` and with `glmmTMB`?
3. How to address convergence failure in a linear mixed model?