MIXED MODELS: REVIEW AND PRACTICE

CLASS 8 FOLLOW-UP: ANALYSE YES/NO OUTCOMES AND PLAN A PEN TRIAL

Gavin Simpson
Aarhus University

Wednesday, 28 October 2026

YOUR 75-MINUTE ROUTE

Work	Minutes
Review group differences, notation and pen-model checks	15
Fit an individual yes/no outcome	15
Plan a future pen trial with simglm	30
Try larger pen counts and recommend a plan	15

Use `after-class-student.R` and `power-planning.R` from the supplied project. Run `setup.R` if packages are missing.

SET UP THE FOLLOW-UP R SESSION

Open the same Class 8 project. Load the tools before reading the pig records.

```
library("readr")
library("dplyr", warn.conflicts = FALSE)
library("tidyr")
library("tibble")
library("lme4", warn.conflicts = FALSE)
library("performance")
library("marginaleffects", warn.conflicts = FALSE)
```

Use `lmer()` for daily gains and `glmer()` for yes/no outcomes. Both come from `lme4`. `performance` checks the fits; `marginaleffects` estimates comparisons.

SAME PENS, NEW OUTCOME

An individual welfare screen is 0 or 1 for each animal. These invented outcomes include shared pen differences. This differs from the earlier pen-level "any animal treated" response.

```
d <- readr::read_csv("pen-animals.csv", show_col_types = FALSE) |>
  mutate(
    room = factor(room),
    pen = factor(pen),
    diet = factor(diet, levels = c("Control", "High fibre"))
  )
d |> count(diet, screened_positive)
```

```
# A tibble: 4 × 3
  diet       screened_positive     n
  <fct>                  <dbl> <int>
1 Control                    0    29
2 Control                    1    19
3 High fibre                 0    37
4 High fibre                 1    11
```

WHAT CHANGES IF WE IGNORE PENS?

Return to daily gain: researchers assigned diets to whole pens. Both models estimate the High fibre minus Control mean difference, allowing for room differences.

- `lmer()` adds a shared pen difference, `(1 | pen)`, because pigs in the same pen share housing and care.
- `lm()` omits that term and assumes pigs' remaining differences are unrelated, even within a pen.

We fit both to see how ignoring shared pen conditions changes uncertainty in the diet comparison on the next slide.

```
shared_pen <- lme4::lmer(
  gain_g_day ~ diet + room + (1 | pen),
  data = d,
  REML = TRUE
)
independent <- lm(
  gain_g_day ~ diet + room,
  data = d
)
```

COMPARE THE DIET ESTIMATES

Compare High fibre minus Control in g/day: first `lm()`, then `lmer()`. The 95% confidence intervals describe uncertainty in the **mean diet difference**.

```
avg_comparisons(
  independent,
  variables = "diet"
)
```

```
 Estimate Std. Error   z Pr(>|z|)   S 2.5 % 97.5 %
     29.9       11.5 2.6  0.00922 6.8  7.38   52.3

Term: diet
Type: response
Comparison: High fibre - Control
```

```
avg_comparisons(
  shared_pen,
  variables = "diet",
  re.form = NA
)
```

```
 Estimate Std. Error    z Pr(>|z|)   S 2.5 % 97.5 %
     29.9       25.9 1.15    0.249 2.0 -20.9   80.6

Term: diet
Type: response
Comparison: High fibre - Control
```

Ignoring pens gives p = 0.009 and an interval excluding zero; accounting for pens gives p = 0.249 and an interval including zero, changing the test conclusion at the 5% level.

USE THE CHECKS AND READ THEM

Convergence asks whether the numerical fitting succeeded. A near-zero estimated shared pen variation is called a **singular fit**.

```
check_convergence(shared_pen)
```

```
[1] TRUE
attr(,"gradient")
[1] 2.51e-09
```

```
check_singularity(shared_pen)
```

```
[1] FALSE
```

```
VarCorr(shared_pen)
```

```
 Groups   Name        Std.Dev.
 pen      (Intercept) 49.5
 Residual             37.1
```

A successful calculation does not establish that all study assumptions are correct.

FIT THE YES/NO MODEL DIRECTLY

The binomial family is for yes/no outcomes. Pen differences are shared by animals in the same pen.

```
binary <- lme4::glmer(
  screened_positive ~ diet + room + (1 | pen),
  data = d,
  family = binomial()
)
```

```
summary(binary)
```

FIT THE YES/NO MODEL DIRECTLY

```
Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: screened_positive ~ diet + room + (1 | pen)
   Data: d

      AIC       BIC    logLik -2*log(L)  df.resid
    123.8     139.2     -55.9     111.8        90

Scaled residuals:
   Min     1Q Median     3Q    Max
-1.163 -0.686 -0.450  0.904  2.321

Random effects:
 Groups Name        Variance Std.Dev.
 pen    (Intercept) 0.117    0.342
Number of obs: 96, groups:  pen, 16

Fixed effects:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    -0.763      0.562   -1.36    0.175
dietHigh fibre -0.834      0.501   -1.66    0.096 .
roomR2          1.000      0.698    1.43    0.152
roomR3         -0.211      0.751   -0.28    0.779
roomR4          0.432      0.707    0.61    0.541
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
            (Intr) dtHghf roomR2 roomR3
dietHighfbr -0.344
roomR2      -0.682 -0.070
roomR3      -0.644 -0.024  0.525
roomR4      -0.669 -0.029  0.548  0.507
```

CHECK THE YES/NO MODEL

The full summary shows the estimated pen spread and diet comparison. Now check whether fitting succeeded and whether the estimated pen spread is near zero.

```
check_convergence(binary)
```

```
[1] TRUE
attr(,"gradient")
[1] 3.3e-05
```

```
check_singularity(binary)
```

```
[1] FALSE
```

Here the estimated pen spread is above zero, and `check_singularity()` returns `FALSE`. Sixteen pens still give limited information about that spread.

ASK FOR A PROBABILITY DIFFERENCE

Compare High fibre minus Control, giving each room equal weight and setting fitted pen differences to zero. `newdata = "balanced"` supplies the category combinations.

```
avg_comparisons(
  binary,
  variables = "diet",
  newdata = "balanced",
  re.form = NA
)
```

```
 Estimate Std. Error     z Pr(>|z|)   S  2.5 % 97.5 %
   -0.167     0.0977 -1.71   0.0882 3.5 -0.358 0.0249

Term: diet
Type: response
Comparison: High fibre - Control
```

These are differences in probabilities, not odds ratios. For a yes/no model, setting pen differences to zero generally differs from averaging probabilities across varying pens.

PLAN A FUTURE EXPERIMENT

Suppose 35 g/day is a worthwhile diet improvement. We plan six animals per pen and half the pens on each diet.

Compare **16 versus 32 independently assigned pens**, and shared pen spreads of **40 versus 60 g/day**.

This simpler future study has no room blocks or section treatments. It is not yet a power calculation for the complete split-plot experiment.

DESCRIBE THE PLANNED ASSIGNMENTS

`simglm` uses `level1` for animals within pens and `level2` for pens. Diet belongs to level 2 and is balanced across pens.

```
library("simglm")
library("purrr")
settings <- list(
  formula = gain ~ diet + (1 | pen),
  fixed = list(
    diet = list(
      var_type = "factor",
      levels = c("Control", "Supplement"),
      var_level = 2,
      force_equal = TRUE
    )
  ),
  sample_size = list(level1 = 6, level2 = 16)
)
settings$sample_size
```

```
$level1
[1] 6

$level2
[1] 16
```

`simglm` can read coefficients directly from the `lmer()` fits used below.

SUPPLY THE ASSUMED GAINS AND SPREAD

Control mean: 700 g/day; diet change: 35 g/day. Shared pen SD: 40; individual SD: 35. An SD describes typical spread; variance is SD squared.

```
settings$reg_weights <- c(700, 35)
settings$randomeffect <- list(
  int_pen = list(variance = 40^2, var_level = 2)
)
settings$error <- list(variance = 35^2)
settings$model_fit <- list(
  model_function = lme4::lmer,
  REML = TRUE
)
settings$replications <- 100
settings$power <- list(alpha = 0.05, dist = "qnorm", direction = "two-tailed")
settings$reg_weights
```

```
[1] 700  35
```

- `model_function` stores `lme4::lmer` without calling it yet.
- When `replicate_simulation()` runs, it uses this function, the formula above and `REML = TRUE` to fit each simulated trial and estimate the diet change and pen spread.

INSPECT ONE SIMULATED TRIAL FIRST

One simulated record is one animal. Animals in the same pen receive the same diet.

```
set.seed(821029)
one_trial <- simglm(settings)
one_trial |>
  select(pen, diet, gain) |>
  slice_head(n = 3)
```

```
  pen    diet gain
1   1 Control  692
2   1 Control  710
3   1 Control  696
```

```
one_trial |>
  distinct(pen, diet) |>
  count(diet)
```

```
        diet n
1    Control 8
2 Supplement 8
```

Before repeating the simulation, check that this matches the study you intended.

REPEAT THE SIMULATION AND FIT

`simglm::replicate_simulation()` creates a new dataset and fits the specified `lmer()` model each time.

```
trials <- replicate_simulation(settings)
length(trials)
```

```
[1] 100
```

Keep the fitted models so failed final checks remain visible. This short batch call stops on a fitting error; the full planning script records errors one trial at a time.

CHECK THE FINAL FITS

The stored fitting checks say whether the final numerical solution is usable. Missing coefficient uncertainty also makes a fit unusable here. Remove stored p-values so simglm applies the stated two-sided threshold to the test statistic.

```
estimates <- list()
for (j in seq_along(trials)) {
  fit <- trials[[j]]
  coefficients <- extract_coefficients(fit)
  coefficients <- coefficients |>
    select(-any_of("p.value"))
  usable <- isTRUE(check_convergence(fit)) &&
    all(is.finite(coefficients$std.error) & coefficients$std.error > 0) &&
    all(is.finite(coefficients$statistic))
  if (usable) estimates[[length(estimates) + 1L]] <- coefficients
}
c(
  attempted = length(trials),
  usable = length(estimates),
  failed = length(trials) - length(estimates)
)
```

```
attempted    usable    failed
      100       100         0
```

CALCULATE POWER WITH SIMGLM

Power is the fraction of usable trials detecting the **assumed** diet change with the specified two-sided approximate test at 0.05.

```
power <- compute_statistics(
  estimates,
  settings,
  power = TRUE,
  type_1_error = FALSE,
  precision = FALSE
)
power |> filter(term == "dietSupplement")
```

```
# A tibble: 1 × 6
  term          avg_estimate power avg_test_stat crit_value_power replications
  <chr>                <dbl> <dbl>         <dbl>            <dbl>        <dbl>
1 dietSuppleme…         33.0  0.31          1.62             1.96          100
```

This is conditional on the simulated study and fitted test. It does not say how likely a hypothesis is to be true.

COMPARE SIZES AND UNCERTAIN ASSUMPTIONS

`power-planning.R` uses `map2()` to pair each pen count with its SD, then `map()` to repeat the trials. Its comments explain each step. This table reads its executed results.

```
plans <- read_csv("reference/power-plans.csv", show_col_types = FALSE)
plans |>
  select(pens, pen_sd, power_among_usable, failed, simulation_se)
```

```
# A tibble: 4 × 5
   pens pen_sd power_among_usable failed simulation_se
  <dbl>  <dbl>              <dbl>  <dbl>         <dbl>
1    16     40               0.43      0        0.0495
2    32     40               0.61      0        0.0488
3    16     60               0.19      0        0.0392
4    32     60               0.42      0        0.0494
```

More pens add assignments. Greater shared pen spread makes diet differences harder to detect. One hundred trials provide only a rough comparison.

DO NOT HIDE SIMULATION UNCERTAINTY

Near power 0.5, one hundred trials give simulation SE about **0.05**. More trials reduce this simulation noise; they do not fix unrealistic assumptions.

If some fits fail, investigate and report power bounds counting failures first as missed detections, then as detections.

Before making a real planning decision, justify the target, spread, analysis, number of trials, cost and feasibility.

HOW MANY PENS WOULD YOU RECOMMEND?

The manager wants at least **0.8 power** to detect a diet improvement of 35 g/day, with six pigs per pen.

Try to answer the manager's question: **"How many pens should we use, and does the answer change if the between-pen SD is 40 or 60 g/day?"**

- Check the power estimates for the existing plans against the target of 0.8.
- In section 4 of `power-planning.R`, add pen counts above 32 to `pens = c(16, 32)`. Keep both SD values and six pigs per pen.
- Rerun the script from the start. Use your new results to recommend a pen count, allowing for the uncertainty from only 100 simulated trials per plan.

SOURCES AND SCOPE

simglm simulation · simglm power · lme4 models · performance · marginaleffects

These are teaching simulations. Small numbers of pens limit approximate model uncertainty; the chosen test must also be checked for the proposed real design.