# LATIN SQUARES AND UNEQUAL GROUP SIZES

## CLASS 8 WORKSHOP

Gavin Simpson
Aarhus University

Wednesday, 28 October 2026

# AFTER THE BREAK

| Time | Work |
|---|---|
| 13:55–14:25 | Build, fit and question a Latin square |
| 14:25–14:55 | Analyse unequal group sizes |
| 14:55–15:10 | Break |
| 15:10–15:30 | Gaussian random effects, QQ plot and a residual check |
| 15:30–15:45 | Project design and data audit |
| 15:45–16:00 | Explain, compare and exit |

# LOAD THE PACKAGES FOR THIS WORKSHOP

Use the same modelling tools as before the break. `readr` reads the CSV files, `dplyr` prepares and counts records, and `tidyr` changes layouts and creates combinations. `tibble` creates the small simulated dataset used later.

```
library("lme4")
library("performance")
library("marginaleffects", warn.conflicts = FALSE)
library("ggplot2")
library("readr")
library("dplyr")
library("tidyr")
library("tibble")
```

# COMPARE FOUR FEEDS FOR BROILERS

Sixteen pens occupy four rows and four columns in a broiler house. Each pen holds the same number of birds of the same age and receives one of four feeds, A–D, throughout the trial.

Light levels differ between rows; airflow differs between columns. We want to compare feeds while allowing for both sources of variation.

A **Latin square** places every feed once in each row and once in each column.

# READ THE PEN LAYOUT

Each record gives one pen's position, feed and mean daily gain. We record one mean daily gain per pen, in grams per bird per day, over the same trial period.

```
l <- read_csv("broiler-latin-square.csv", show_col_types = FALSE) |>
  mutate(
    row = factor(row, levels = paste0("Row", 1:4)),
    column = factor(column, levels = paste0("Col", 1:4)),
    feed = factor(feed, levels = c("A", "B", "C", "D"))
  )
l |>
  select(row, column, feed) |>
  pivot_wider(names_from = column, values_from = feed)
```

```
# A tibble: 4 × 5
  row   Col1  Col2  Col3  Col4
  <fct> <fct> <fct> <fct> <fct>
1 Row1  B     D     A     C
2 Row2  C     A     B     D
3 Row3  A     C     D     B
4 Row4  D     B     C     A
```

Find all four pens receiving feed A. How does the layout spread them across the light and airflow differences?

# RANDOMISATION STILL MATTERS

Construct a valid square, then randomly choose the row order, column order and feed labels. Record those choices before the trial starts.

```
l |>
  count(row, feed) |>
  pivot_wider(
    names_from = feed,
    values_from = n
  )
```

```
# A tibble: 4 × 5
  row       A     B     C     D
  <fct> <int> <int> <int> <int>
1 Row1      1     1     1     1
2 Row2      1     1     1     1
3 Row3      1     1     1     1
4 Row4      1     1     1     1
```

```
l |>
  count(column, feed) |>
  pivot_wider(
    names_from = feed,
    values_from = n
  )
```

```
# A tibble: 4 × 5
  column     A     B     C     D
  <fct>  <int> <int> <int> <int>
1 Col1       1     1     1     1
2 Col2       1     1     1     1
3 Col3       1     1     1     1
4 Col4       1     1     1     1
```

The counts check the pattern. They cannot prove that feed assignment was random.

# ALLOW FOR PEN POSITION

Compare feeds while allowing these four rows and four columns to have different mean gains. Treat row and column as fixed effects.

```
latin_fit <- lm(
  gain_g_day ~ feed + row + column,
  data = l
)
summary(latin_fit)
```

There is one mean per pen, so we do not add a shared pen term. This model uses `lm()` because it has no random effects.

# ALLOW FOR PEN POSITION

```
Call:
lm(formula = gain_g_day ~ feed + row + column, data = l)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6902 -0.2478  0.0961  0.3134  0.6012

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  58.3593     0.5095  114.54    3e-11 ***
feedB         1.9042     0.4557    4.18  0.00582 **
feedC         2.9727     0.4557    6.52  0.00062 ***
feedD         1.3667     0.4557    3.00  0.02404 *
rowRow2       1.5239     0.4557    3.34  0.01553 *
rowRow3       1.6942     0.4557    3.72  0.00988 **
rowRow4       0.6428     0.4557    1.41  0.20807
columnCol2    0.4947     0.4557    1.09  0.31935
columnCol3   -0.0948     0.4557   -0.21  0.84213
columnCol4    1.3231     0.4557    2.90  0.02721 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.644 on 6 degrees of freedom
Multiple R-squared:  0.925, Adjusted R-squared:  0.813
F-statistic: 8.27 on 9 and 6 DF,  p-value: 0.00912
```

# COMPARE FEEDS C AND A

The researcher wants **C minus A** in mean daily gain, in grams per bird per day, adjusted for pen position.

```
avg_comparisons(
  latin_fit,
  variables = list(feed = c("A", "C")),
  df = df.residual(latin_fit)
)
```

```
 Estimate Std. Error    t Pr(>|t|)    S 2.5 % 97.5 % Df
     2.97      0.456 6.52   <0.001 10.7  1.86   4.09  6

Term: feed
Type: response
Comparison: C - A
```

The model assumes that feed, row and column differences add together. The t interval uses the model's six remaining error degrees of freedom.

# WHAT THE SQUARE CANNOT SEPARATE

Suppose feed C works especially well only in Row 1. We have one pen at each row-column position.

Can we separately estimate every feed-by-row change, every feed-by-column change, and ordinary pen variation?

Explain why adding more model terms cannot create missing replication.

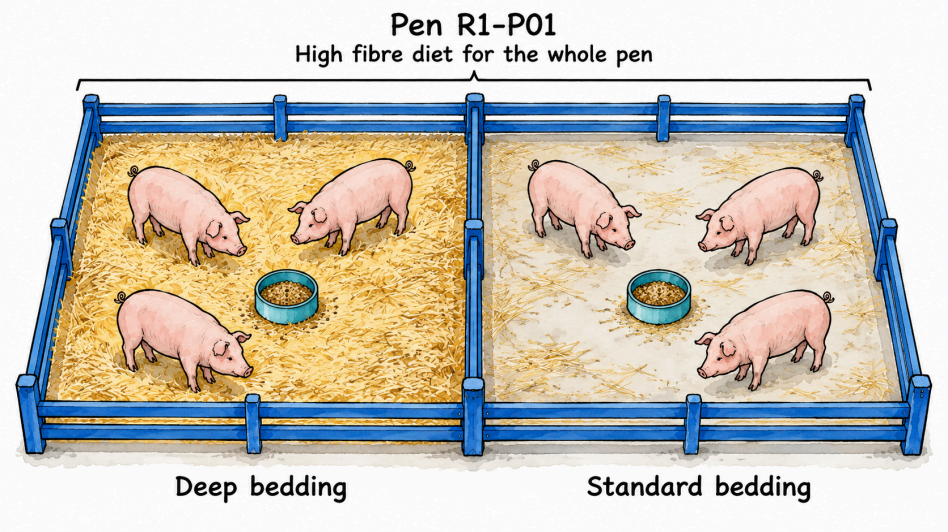
# UNEQUAL SIZES ARE A PROPERTY OF A DESIGN

A design is **unbalanced** when relevant groups do not contain equal numbers of units or measurements.

This can be planned, for example because one treatment is costly, or arise because observations are lost.

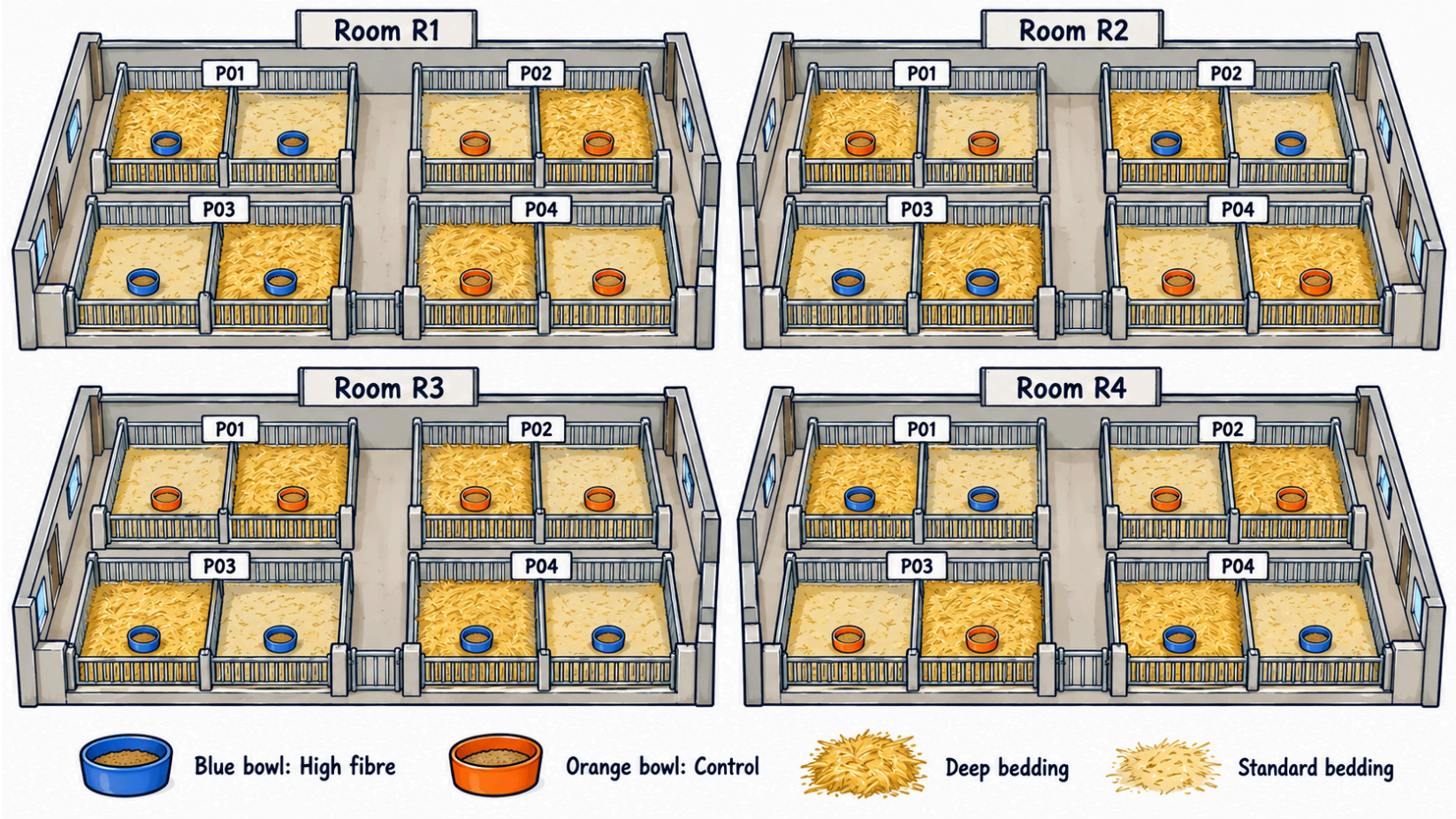
Count the pens researchers assigned diets to, as well as the measurements they recorded. Losing animal measurements within pens and losing whole pens are different changes.

# RECALL THE SPLIT-PLOT STUDY




- Researchers assign **Control or High fibre diet to whole pens**.
- Within each pen, they randomly choose which section gets **Deep bedding**; the other gets **Standard bedding**.
- Pigs and bedding stay in their assigned sections.
- Record **one mean daily gain per section**, in g/day.

# RECALL ALL FOUR ROOMS




Sixteen pens, eight per diet, **32 section means**.

# A MEASURING-EQUIPMENT FAULT

A fault leads to the loss of both section averages from pen R1-P01 and the Left section of R3-P02.

Assume this fault is unrelated to animal growth.

```
u <- read_csv("unequal-split-plot.csv", show_col_types = FALSE) |>
  mutate(
    room = factor(room, levels = paste0("R", 1:4)),
    pen = factor(pen),
    diet = factor(diet, levels = c("Control", "High fibre")),
    bedding = factor(bedding, levels = c("Standard", "Deep"))
  )
u |>
  count(diet, bedding)
```

```
# A tibble: 4 × 3
  diet       bedding      n
  <fct>      <fct>    <int>
1 Control    Standard     8
2 Control    Deep         7
3 High fibre Standard     7
4 High fibre Deep         7
```

# COUNT WHAT REMAINS

There are now fifteen pens with data. Losing a section does not remove its partner's diet assignment.

```
remaining_pens <- u |>
  distinct(room, pen, diet)
remaining_pens |>
  count(room, diet)
```

```
# A tibble: 8 × 3
  room  diet           n
  <fct> <fct>      <int>
1 R1    Control        2
2 R1    High fibre     1
3 R2    Control        2
4 R2    High fibre     2
5 R3    Control        2
6 R3    High fibre     2
7 R4    Control        2
8 R4    High fibre     2
```

If the heaviest or sickest animals were more likely to be missing, why would the same count table be insufficient?

# FIT THE RECORDED UNEQUAL DATA

Use the same design argument: diet between pens, bedding within pens, and known rooms.

```
unequal_fit <- lme4::lmer(
  gain_g_day ~ room + diet * bedding + (1 | pen),
  data = u,
  REML = TRUE
)
check_convergence(unequal_fit)
```

```
[1] TRUE
attr(,"gradient")
[1] 1.34e-08
```

```
check_singularity(unequal_fit)
```

```
[1] FALSE
```

Unequal counts alone do not require deleting good observations to make groups equal.

# RAW MEANS COUNT OBSERVED SECTIONS

These averages give each recorded section equal weight. Rooms with more surviving sections contribute more.

```
u |>
  group_by(diet, bedding) |>
  summarise(gain_g_day = mean(gain_g_day), .groups = "drop") |>
  arrange(bedding, diet)
```

```
# A tibble: 4 × 3
  diet       bedding  gain_g_day
  <fct>      <fct>         <dbl>
1 Control    Standard       747.
2 High fibre Standard       753.
3 Control    Deep           764.
4 High fibre Deep           777.
```

The farm manager instead wants each of the four rooms to count equally. We need to state that choice before comparing adjusted means.

# ESTIMATE MEANS FOR THE SAME ROOMS

Use `newdata = "balanced"` to give each room equal weight within every diet and bedding combination, even after measurements are lost.

```
avg_predictions(
  unequal_fit,
  newdata = "balanced",
  by = c("diet", "bedding"),
  re.form = NA
)
```

```
      diet  bedding Estimate Std. Error    z Pr(>|z|)     S 2.5 % 97.5 %
 Control    Standard      747       21.2 35.3   <0.001 903.5   706    789
 Control    Deep          762       21.3 35.8   <0.001 929.5   720    804
 High fibre Standard      747       23.0 32.5   <0.001 765.0   702    792
 High fibre Deep          771       23.0 33.5   <0.001 814.7   726    816

Type: response
```

Each room receives one quarter of the weight. `re.form = NA` leaves out differences fitted for particular pens; here this also gives the mean over the assumed zero-mean pen differences.

# COMPARE THE SAME TARGET AFTER LOSSES

For each bedding, compare the two diets with equal room weights.

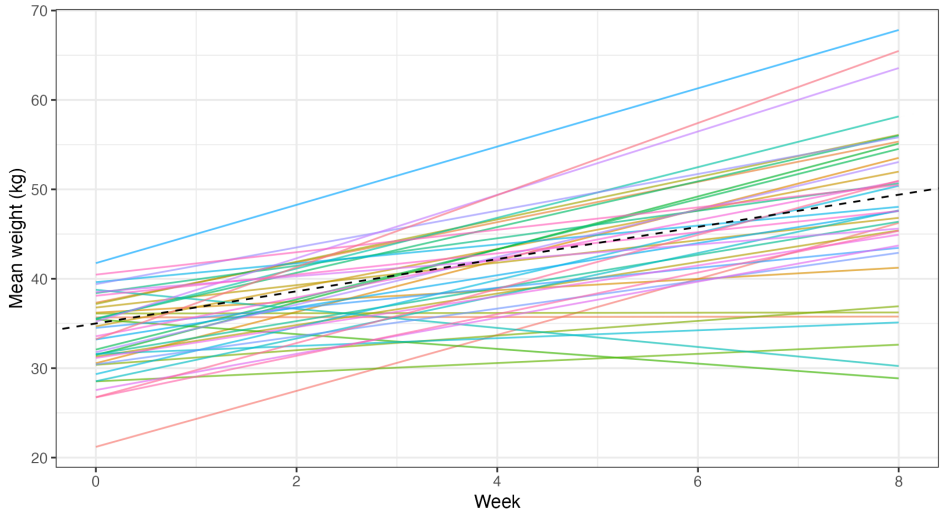
```
avg_comparisons(
  unequal_fit,
  variables = "diet",
  newdata = "balanced",
  by = "bedding",
  re.form = NA
)
```

```
  bedding Estimate Std. Error       z Pr(>|z|)   S 2.5 % 97.5 %
 Standard   -0.225       31.3 -0.0072    0.994 0.0 -61.5   61.1
 Deep        8.928       31.4  0.2848    0.776 0.4 -52.5   70.4

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

What assumption makes this comparison useful? What information about the fault would you request before using it?

# RECALL THE FORTY-CALF EXAMPLE




- We follow forty calves from **week 0 to week 8**.
- Overall starting weight is **35 kg**; average gain is **1.8 kg/week**.
- A calf's **random intercept** is its starting-weight difference from 35 kg.
- Its **random slope** is its weekly-gain difference from 1.8 kg/week.

These are simulated mean paths, not observed weights or fitted calf results.

# THE CALF RANDOM EFFECTS FOLLOW A NORMAL MODEL

For every calf, we draw a starting-weight random effect from a normal distribution with mean zero and **SD 4 kg**. We draw a weekly-gain random effect from another normal distribution with mean zero and **SD 0.9 kg/week**.

We draw each calf's pair without using another calf's draw. The pairs are **independent and identically distributed**, or **i.i.d.**, across calves.

The same pair of distributions applies to every calf. Starting-weight and weekly-gain random effects have different units and spreads. Gaussian means normal.

# KEEP THE RANDOM EFFECTS IN THEIR ORIGINAL UNITS

We want to illustrate the **shape of the random-effect distributions**. A QQ plot compares their ordered values with a normal distribution; rescaling is unnecessary.

`avg_predictions()` estimates expected outcomes and `avg_comparisons()` estimates contrasts. Those answer different questions from checking a distribution's shape.

We keep starting-weight random effects in **kg** and weekly-gain random effects in **kg/week**.

# PREPARE THE CALF RANDOM EFFECTS FOR PLOTTING

Stack the two columns from the earlier calf illustration. Each panel will show one kind of random effect, in its original units.

```
random_draws <- calf_effects |>
  transmute(
    calf,
    "Starting weight (kg)" = start_difference
    "Weekly gain (kg/week)" = gain_difference
  ) |>
  pivot_longer(
    cols = -calf,
    cols_vary = "slowest",
    names_to = "effect",
    values_to = "difference"
  )
random_draws |>
  slice(c(1, 2, 41, 42))
```

```
# A tibble: 4 × 3
  calf  effect                difference
  <fct> <chr>                      <dbl>
1 C01   Starting weight (kg)     -13.8
2 C02   Starting weight (kg)       0.680
3 C01   Weekly gain (kg/week)      1.33
4 C02   Weekly gain (kg/week)     -1.79
```

# BUILD A NORMAL QQ PLOT

A **quantile–quantile plot**, or **QQ plot**, places ordered values against values expected from a reference distribution. Here the reference is normal.

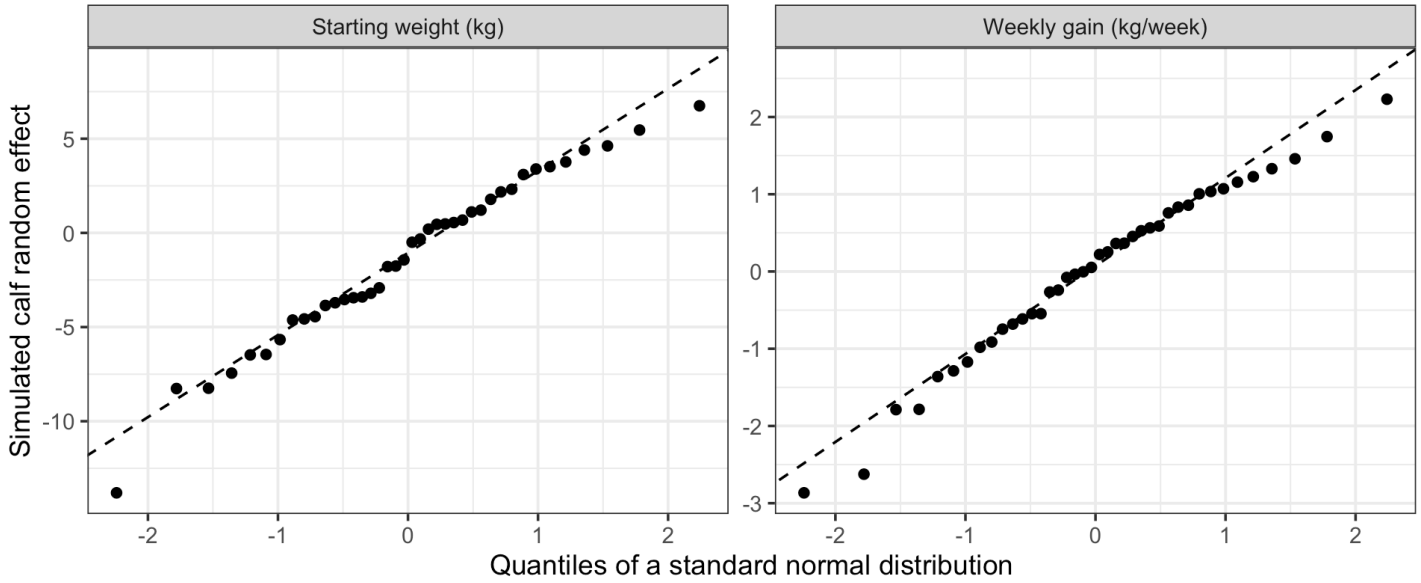
```
p_random_qq <- ggplot(random_draws, aes(sample = difference)) +
  stat_qq() +
  stat_qq_line(linetype = "dashed") +
  facet_wrap(~effect, nrow = 1, scales = "free_y") +
  labs(
    x = "Quantiles of a standard normal distribution",
    y = "Simulated calf random effect"
  ) +
  theme_bw(base_size = 14)
```

Each panel keeps its own vertical units. The dashed line provides a normal reference matched to that panel's values.

# NORMAL QQ PLOTS OF THE CALF RANDOM EFFECTS

Each panel contains forty simulated random effects. Their original units appear in the panel labels.

```
p_random_qq
```




Points near a straight line are consistent with a normal shape. We generated these values from normal distributions; this illustrates the assumption rather than checking real calves.

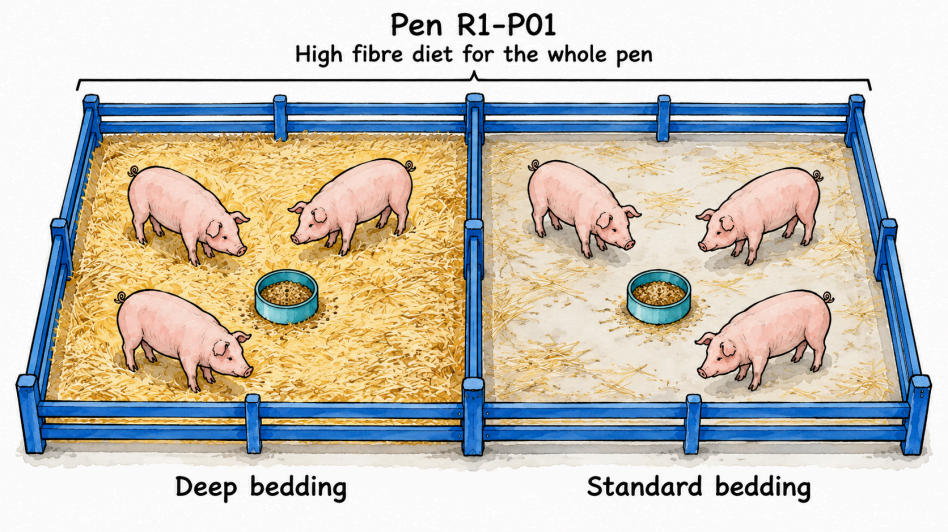
# A QQ PLOT DOES NOT ESTABLISH INDEPENDENCE

Calves sharing a feeder or a disease outbreak might have related random effects. The shape of this QQ plot cannot tell us whether they shared those conditions.

Recorded weights from one calf share its random intercept and slope. We are not assuming that all its repeated weights are separate i.i.d. draws.

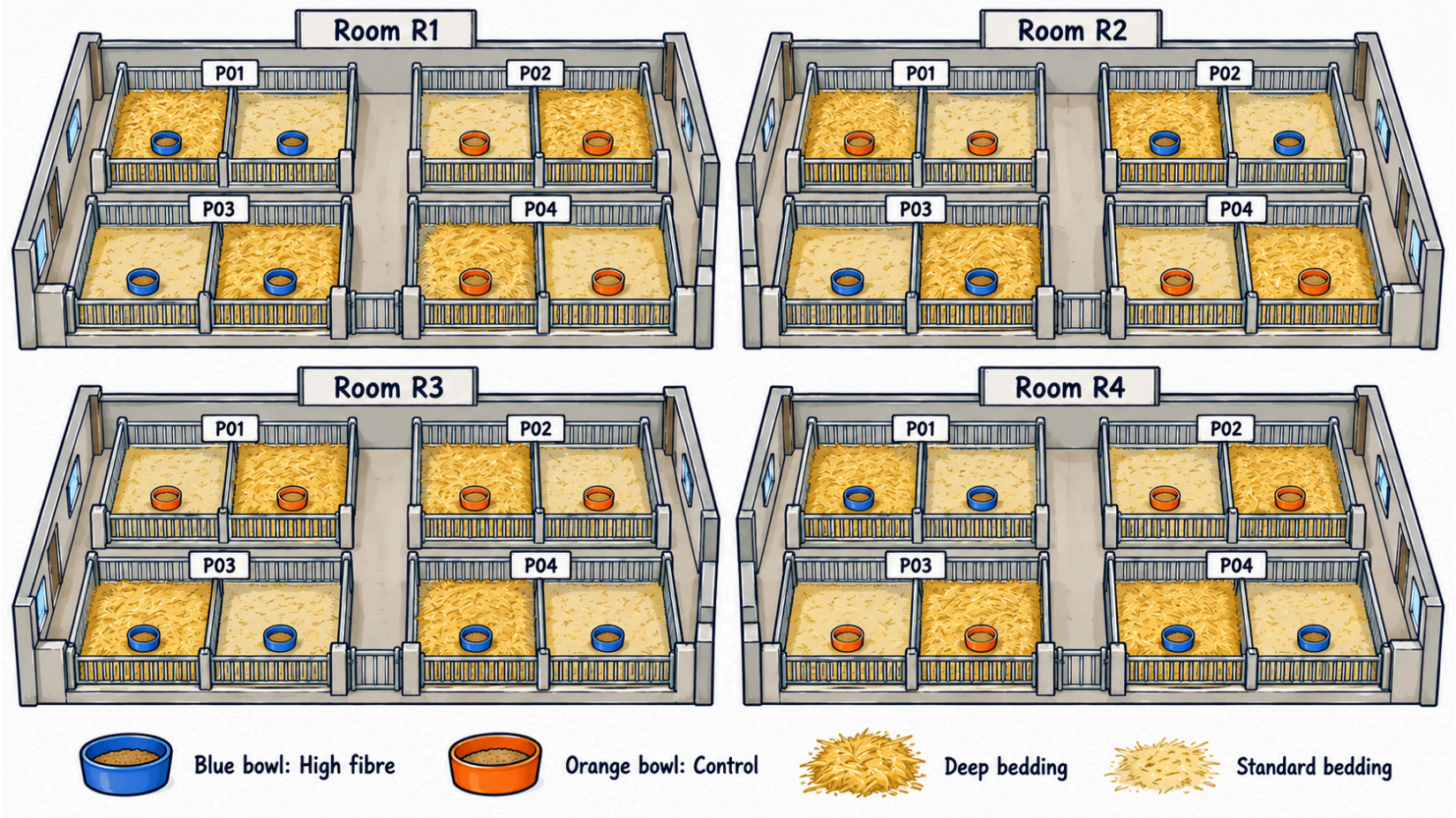
After centring time, intercept and slope random effects within a calf can be related while different calves' pairs still follow the same Gaussian model. Fitted, partly pooled random effects should not be treated as the raw simulated draws shown here.

# RETURN TO THE SPLIT-PLOT STUDY




- Researchers assign **Control or High fibre diet to whole pens**.
- Within each pen, they randomly choose which section gets **Deep bedding**; the other gets **Standard bedding**.
- Pigs and bedding stay in their assigned sections.
- Record **one mean daily gain per section**, in g/day.

# RETURN TO ALL FOUR ROOMS

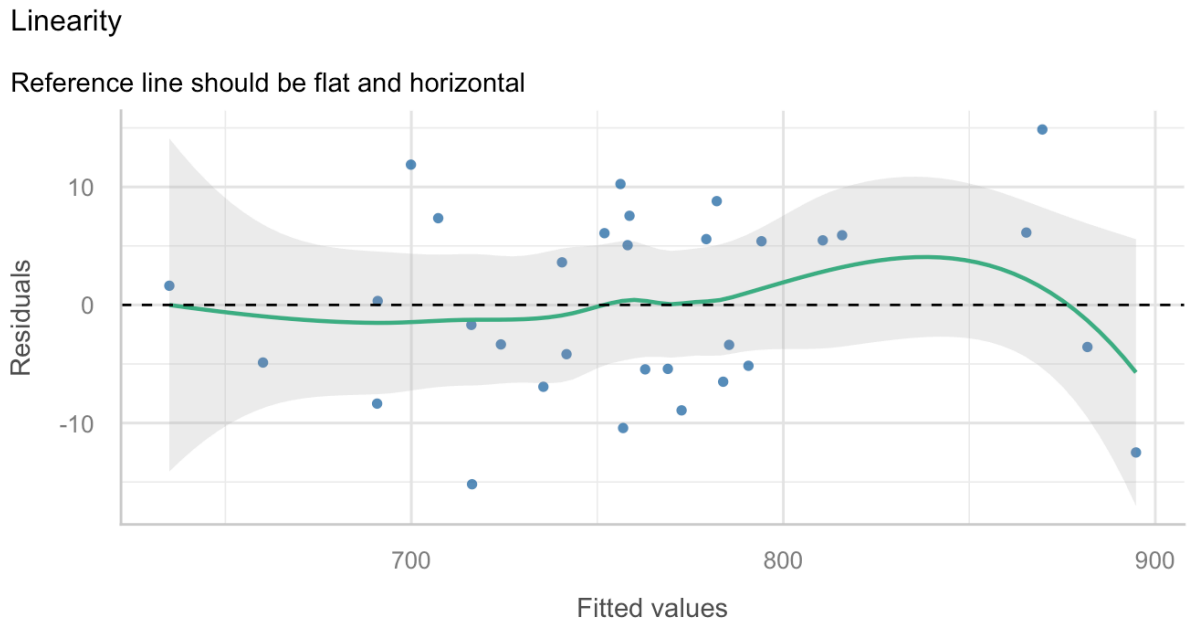



Sixteen pens, eight per diet, **32 section means**.

# CHECK THE MODEL

For the complete pig study, a **residual** is observed section gain minus fitted section gain. Both axes are in g/day.

```
plot(
  check_model(
    split_fit,
    check = "linearity"
  ),
  n_columns = 1
)
```



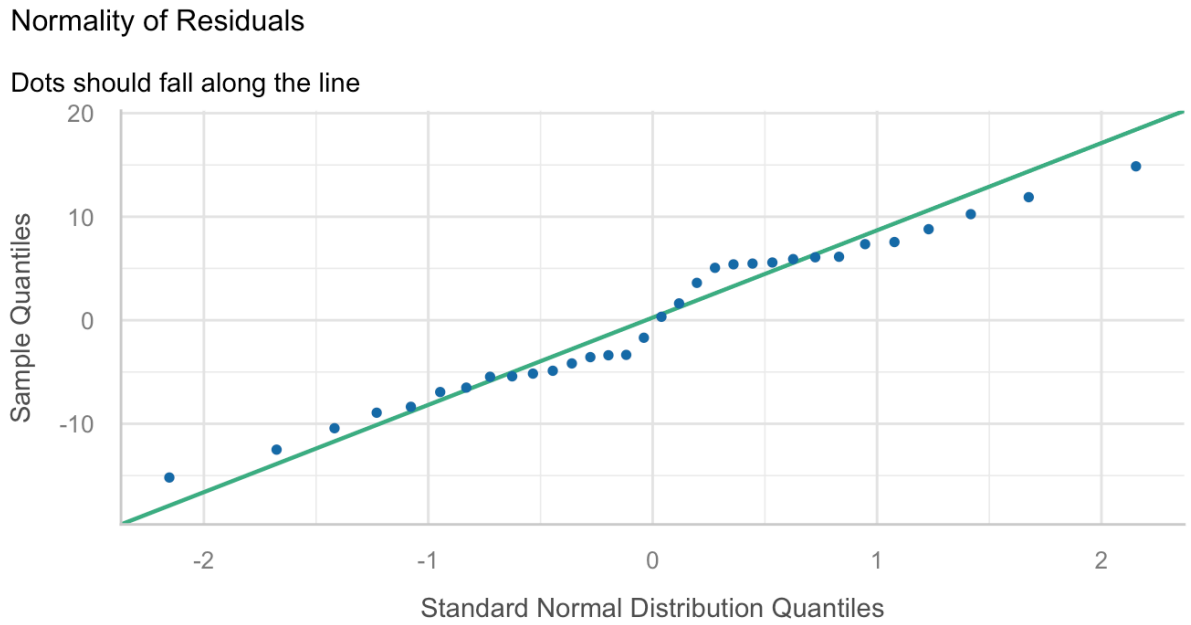

Does the green line stay roughly horizontal near zero, or show a clear curve? Describe one pattern you see and whether it would prompt you to investigate the mean model further.

# CHECK THE SHAPE OF THE SECTION RESIDUALS

This QQ plot examines the remaining **section residuals**, in g/day, after allowing for room, diet, bedding and shared pen differences.

```
plot(
  check_model(
    split_fit,
    check = "qq",
    detrend = FALSE
  ),
  n_columns = 1
)
```

```
For confidence bands, please install `qqplotr`.
```



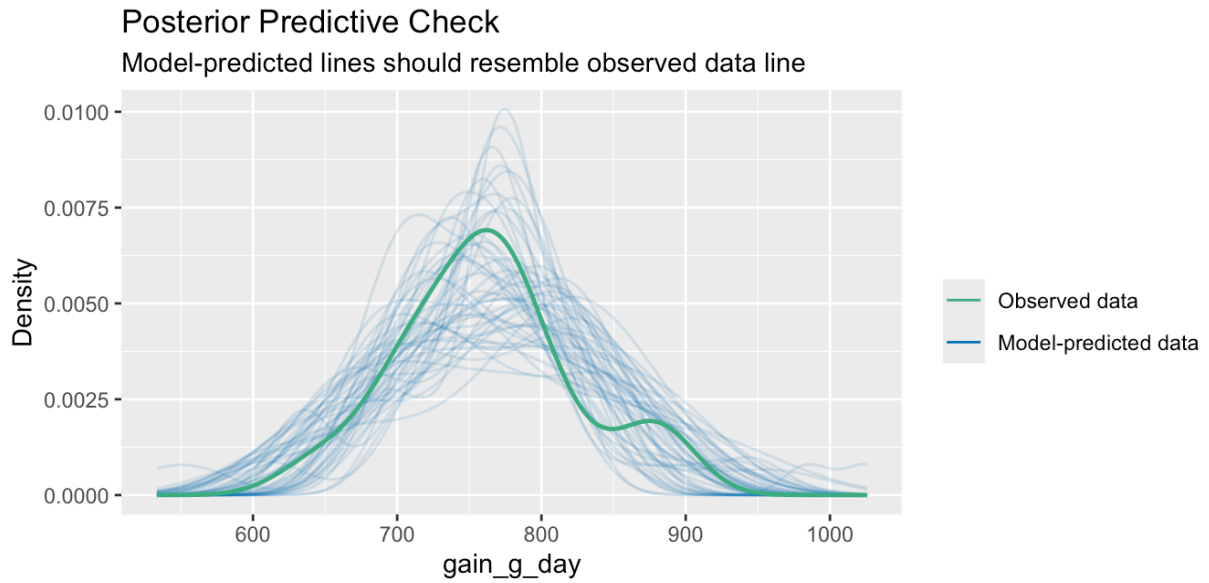

Do points follow the reference line through the middle and at both ends? Describe any systematic departure that would make you investigate the normal-error assumption.

# CAN THE MODEL REPRODUCE THESE GAINS?

Simulate **50 datasets** with the same room and treatment assignments, drawing new pen and section differences from the fitted model.

```
set.seed(82850)
plot(check_predictions(
  split_fit,
  iterations = 50,
  re_formula = NA
))
```

```
Ignoring unknown labels:
• size : ""
```




Compare the observed curve with the simulated curves. Does the model reproduce the centre, spread and tails of the observed gains? Identify one feature it captures well or poorly.

# PROJECT DESIGN AND DATA AUDIT

For your own project, draw where treatment was assigned or observations were sampled.

Mark shared groups, repeated measurements, unequal counts and missing design information.

Record one data source, one unanswered question and who will resolve it before **4 November**.

# INDIVIDUAL EXIT

A proposal gives one diet to each pen, and two beddings to sections inside each pen. One section measurement is lost.

Explain why this is both **split plot** and **unbalanced**, and why adding more animals would not add independent diet assignments.

After class: a binary example and future power planning with `simglm`.

# SOFTWARE REFERENCES

lme4 · Linear models · performance checks · marginaleffects predictions

All numerical examples are invented teaching data.

# BACKUP SLIDES

The greenhouse example is retained for future Agroecology cohorts. The workshop ends before this section.

# BACKUP: FERTILISERS IN A GREENHOUSE

Sixteen trays occupy four rows and four columns. Light changes along the rows; temperature changes along the columns.

We want to compare four fertilisers, A–D, without putting one only in a bright or warm position.

A **Latin square** places every fertiliser once in each row and once in each column.

# READ THE SQUARE

Each record is one tray's position, fertiliser and measured mass. A tray receives just one fertiliser.

```
plant_square <- read_csv("latin-square.csv", show_col_types = FALSE) |>
  mutate(
    row = factor(row, levels = paste0("Row", 1:4)),
    column = factor(column, levels = paste0("Col", 1:4)),
    fertiliser = factor(fertiliser, levels = c("A", "B", "C", "D"))
  )
plant_square |>
  select(row, column, fertiliser) |>
  pivot_wider(names_from = column, values_from = fertiliser)
```

```
# A tibble: 4 × 5
  row   Col1  Col2  Col3  Col4
  <fct> <fct> <fct> <fct> <fct>
1 Row1  D     A     B     C
2 Row2  A     D     C     B
3 Row3  C     B     D     A
4 Row4  B     C     A     D
```

Point to all four positions for fertiliser A. Which two sources of variation does this layout spread it across?

# RANDOMISATION STILL MATTERS

Construct a valid square, then randomly choose the row order, column order and fertiliser labels. Record those choices.

```
plant_square |>
  count(row, fertiliser) |>
  pivot_wider(
    names_from = fertiliser,
    values_from = n
  )
```

```
# A tibble: 4 × 5
  row       A     B     C     D
  <fct> <int> <int> <int> <int>
1 Row1      1     1     1     1
2 Row2      1     1     1     1
3 Row3      1     1     1     1
4 Row4      1     1     1     1
```

```
plant_square |>
  count(column, fertiliser) |>
  pivot_wider(
    names_from = fertiliser,
    values_from = n
  )
```

```
# A tibble: 4 × 5
  column     A     B     C     D
  <fct>  <int> <int> <int> <int>
1 Col1       1     1     1     1
2 Col2       1     1     1     1
3 Col3       1     1     1     1
4 Col4       1     1     1     1
```

The counts check the pattern. They cannot prove that treatment assignment was random.

# ALLOW FOR TRAY POSITION

Allow these four rows and columns to have different average masses. Compare fertilisers after allowing for those differences.

```
plant_fit <- lm(
  mass_g ~ fertiliser + row + column,
  data = plant_square
)
summary(plant_fit)
```

There is no shared tray term: there is only one observation per tray.

# ALLOW FOR TRAY POSITION

```
Call:
lm(formula = mass_g ~ fertiliser + row + column, data = plant_square)

Residuals:
    Min      1Q  Median      3Q     Max
-0.4929 -0.2477  0.0625  0.1374  0.5213

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   10.076      0.368   27.34  1.6e-07 ***
fertiliserB    0.862      0.330    2.62  0.03977 *
fertiliserC    1.377      0.330    4.18  0.00582 **
fertiliserD    0.550      0.330    1.67  0.14591
rowRow2        1.411      0.330    4.28  0.00520 **
rowRow3        2.031      0.330    6.16  0.00084 ***
rowRow4        1.378      0.330    4.18  0.00581 **
columnCol2     0.959      0.330    2.91  0.02695 *
columnCol3     1.349      0.330    4.09  0.00640 **
columnCol4     1.829      0.330    5.55  0.00145 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.466 on 6 degrees of freedom
Multiple R-squared:  0.939, Adjusted R-squared:  0.847
F-statistic: 10.2 on 9 and 6 DF,  p-value: 0.00519
```

# ANSWER THE GROWER'S QUESTION

The grower wants **C minus A**, in grams of plant mass per tray.

```
avg_comparisons(
  plant_fit,
  variables = list(fertiliser = c("A", "C")),
  df = df.residual(plant_fit)
)
```

```
 Estimate Std. Error    t Pr(>|t|)   S 2.5 % 97.5 % Df
     1.38       0.33 4.18  0.00582 7.4 0.571   2.18  6

Term: fertiliser
Type: response
Comparison: C - A
```

The model assumes the row, column and fertiliser changes add together. The t interval uses six remaining error degrees of freedom.

# WHAT THE SQUARE CANNOT SEPARATE

Suppose fertiliser C works especially well only in Row 1. We have one observation at each row-column position.

Can we separately estimate every fertiliser-by-row change, every fertiliser-by-column change, and ordinary tray variation?

Explain why adding more model terms cannot create missing replication.