Lecture 8 Introduction to Data Structure

T. Neeman

26th August 2022

Data Structure

Data structure is a fundamental idea in biological data science, but it is not discussed much in the traditional data science literature. In biology, we measure responses on samples that are related or that share a common environment. Measurements from samples that share a common environment will be more correlated that measurements from samples from different environments. Here are some specific examples:

- Measurements from two leaves on a plant are more correlated than measurements from two leaves on two different plants.
- Measurements from two mice in the same litter are more correlated than measurements from two mice from different litters.
- Measurements from two mice who share a cage are more correlated than measurements from two mice from different cages.

It's important to capture this correlation structure in our statistical model, because it can have a substantial impact on statistical inference. There are two mistakes we can make by failing to account for the structure (relatedness, common environment). Under one scenario, we might claim there is a treatment effect when the treatment effect is confounded or partially confounded with an environmental effect. Under a second scenario, we could fail to see a treatment effect because treatment differences were masked by differing environments.

Let's explore these two scenarios with a couple of examples:

Example 1: Heights of HDR students at RSB and JCSMR

We are interested in the hypothesis that HDR students at JCSMR are taller on average than HDR students at RSB. Our experimental plan is to select 25 students at random from each school and measure their height. But COVID hits, and we can only find 1 student from each school, so we measure each student 25 times. Now we have our 50 measurements. Can we make the same inference as if we had measurements on 50 students?

Example 2: Repeating experiments

We have set up a bacterial growth assay for comparing growth rates between wild-type and several genetically modified bacteria. We have run the experiment multiple times. There is considerable variation between experiments. In fact the "experiment effect" is stronger than the genotype effect. When we combine all the measurements across the multiple experimental runs, we fail to see a difference between the mutants and the control.

A few comments on these examples

In *Example 1* , the only information we gain in this study is a measure of how reliable our measuring tool is. The total information about height differences comes from a sample of 2. So this study has much less information about height differences than a study with 50 *independent* measurements, even though both studies have 50 measurements.

In *Example 2*, information about the genotype effect is measured within each experiment. Our analysis should ideally *combine* the information from each experiment across the set of experiments. The large variation *between* experiments is not relevant to the genotype question. Recognising that the overall measurement variation can be partitioned into between-experiment variation, and within-experiment variation, statistical inference about genotype effects should be wrt within-experiment variation.

Orange-bellied parrots

Reference: Stojanovic D. et al (2020), Nestling growth and body condition of critically endangered Orange-bellied Parrots Neophema chrysogaster,Emu - Austral Ornithology 120:2, pp135-141

The Orange-bellied Parrot (OBP) is a small migratory parrot, which breeds only in coastal south-west Tasmania and winters in coastal Victoria and South Australia. The species' current breeding range is a narrow coastal strip of south-west Tasmania near Melaleuca. Orange-bellied Parrots may be the most endangered parrot in the world; in 2016 only two wild-born females bred in the last wild population (Stojanovic et al. 2018).

ANU researchers and conservationists developed an OBP-specific body condition index for nestlings in order to monitor nestling success and access the potential factors predictive of body condition. They have measurements for 106 nestlings from 35 nests across 4 years.

In this analysis, they focus on whether relative hatch order affects body condition. Because nests have different brood sizes, they classified nestlings as (1) first-hatched, (2) last-hatched, (3) middle-hatched.

Nest (eventid) is a potential source of variation for body condition. Hatch order should be compared within a nest, and order effects averaged across nests. Year is also potentially predictive of body condition. In this case, we treat it as a fixed effect because it is a factor of interest.

First we'll import our libraries, and then have a closer look at this experiment.

```
library(tidyverse)
library(lmerTest)
library(ggResidpanel)
library(emmeans)
```

Import and check data structure

```
obp <-read_csv("../Data/contemporary obp nestlings.csv")
```

```
## Rows: 106 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (9): site, eventid, ring, mother w/c, order, sex, predicted fledge date,...
## dbl (6): year, wgtdif, brood, rank, fldgday, difference days
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(obp)
```

```
## Rows: 106
## Columns: 15
## $ site                    <chr> "Melaleuca", "Melaleuca", "Melaleuca", "Melale~
## $ eventid                 <chr> "Green Shed bottom 2014_2014", "Green Shed bot~
## $ ring                    <chr> "23020137", "23020127", "23020141", "23019189"~
## $ year                    <dbl> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013~
## $ wgtdif                  <dbl> 1.57, 1.33, -0.12, -1.43, 0.33, 1.94, -1.40, 2~
## $ brood                   <dbl> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3, 3, 3, 2, 2, 3~
## $ rank                    <dbl> 4, 2, 3, 5, 1, 2, 4, 1, 3, 5, 1, 1, 2, 1, 2, 1~
## $ fldgday                 <dbl> 42, 37, 40, 45, 37, 28, 33, 25, 29, 37, 49, 49~
## $ `mother w/c`            <chr> "w", "w", "w", "w", "w", "w", "w", "w", "w", "~
## $ order                   <chr> "middle", "middle", "middle", "last", "first",~
## $ sex                     <chr> "F", "F", "M", "M", "M", "F", "F", "M", "M", "~
## $ `predicted fledge date` <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ `known hatch date`      <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ `known fledge date`     <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ `difference days`       <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
```

```
obp %>%
  group_by(year, eventid)%>%
  summarise(count = n())
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

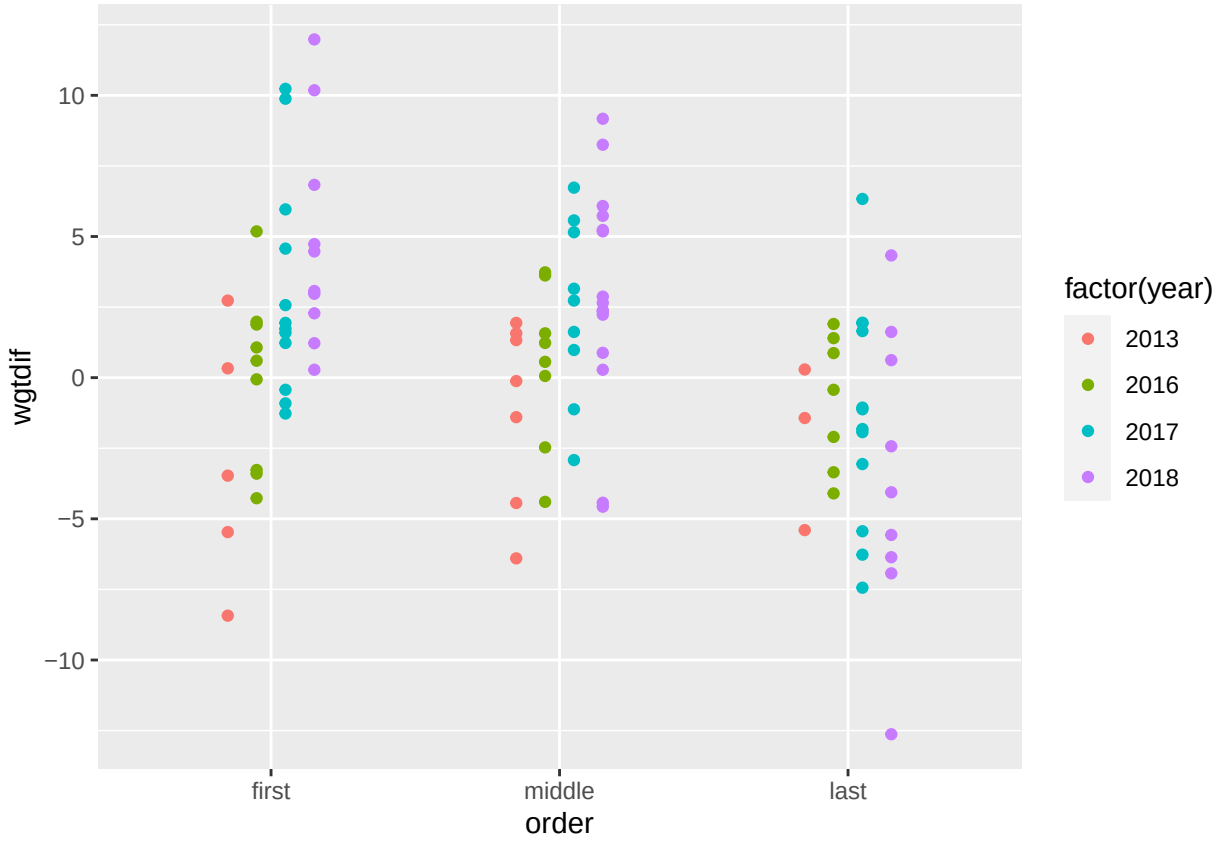
```
## # A tibble: 35 x 3
## # Groups:   year [4]
##     year eventid                     count
##    <dbl> <chr>                       <int>
##  1  2013 Green Shed bottom 2014_2014     5
##  2  2013 RS1-3T 2014_2014                5
##  3  2013 SP1/2 2014_2014                 3
##  4  2013 SP3 2014_2014                   2
##  5  2016 GS38_2016                       3
##  6  2016 H16_2016                        3
##  7  2016 H21_2016                        4
##  8  2016 MP60_2016                       3
##  9  2016 MP63_2016                       4
## 10  2016 PGS2_2016                       3
## # ... with 25 more rows
## # i Use `print(n = ...)` to see more rows
```

Data exploration

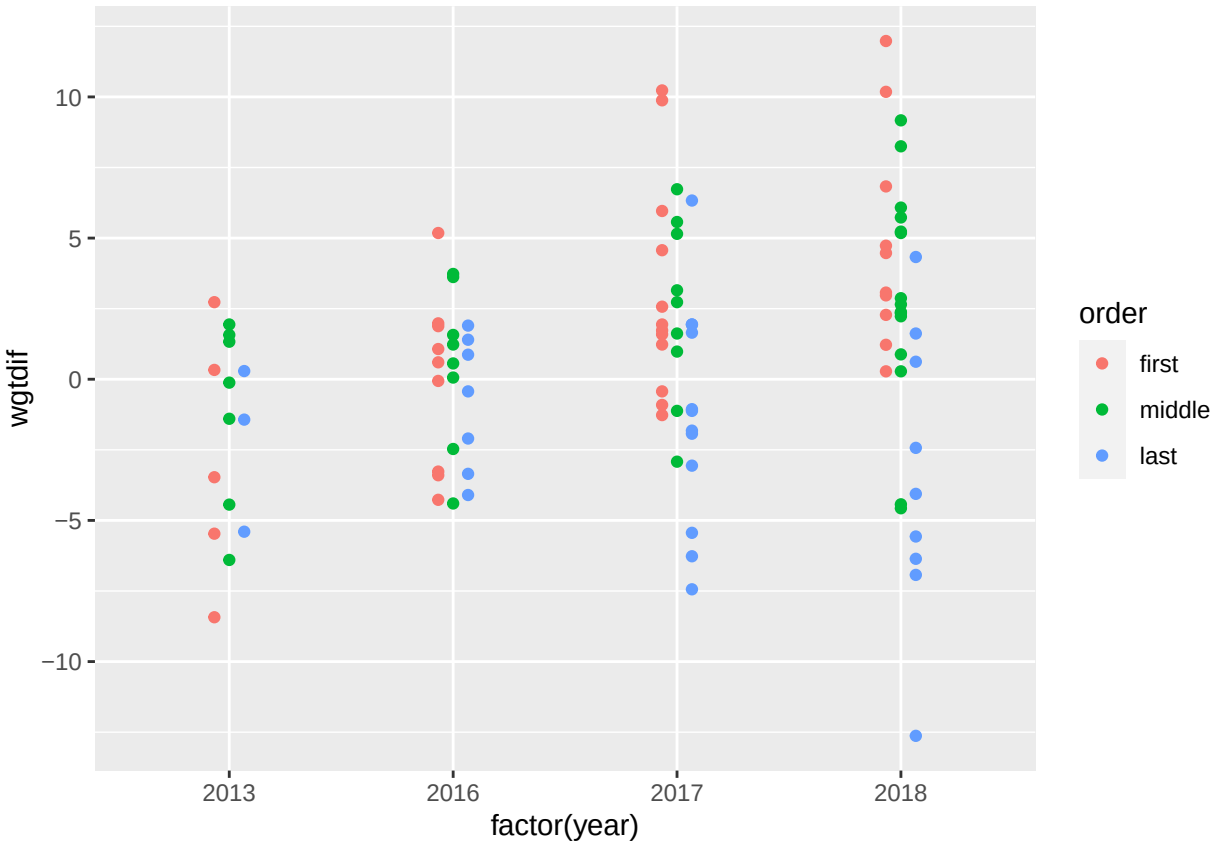
Body condition index is the variable "wgtdiff". We can explore order effects on body condition: wgtdiff on the y-axis and including year and order on the x-axis or colour. To visualise patterns in order, we put the levels of the factor order in "order": first, middle, last.

```
obp$order<- factor(obp$order, levels = c("first","middle","last"))

ggplot(obp,aes(x=order,y=wgtdif,colour=factor(year)))+
  geom_point(position = position_dodge(width = 0.4))
```



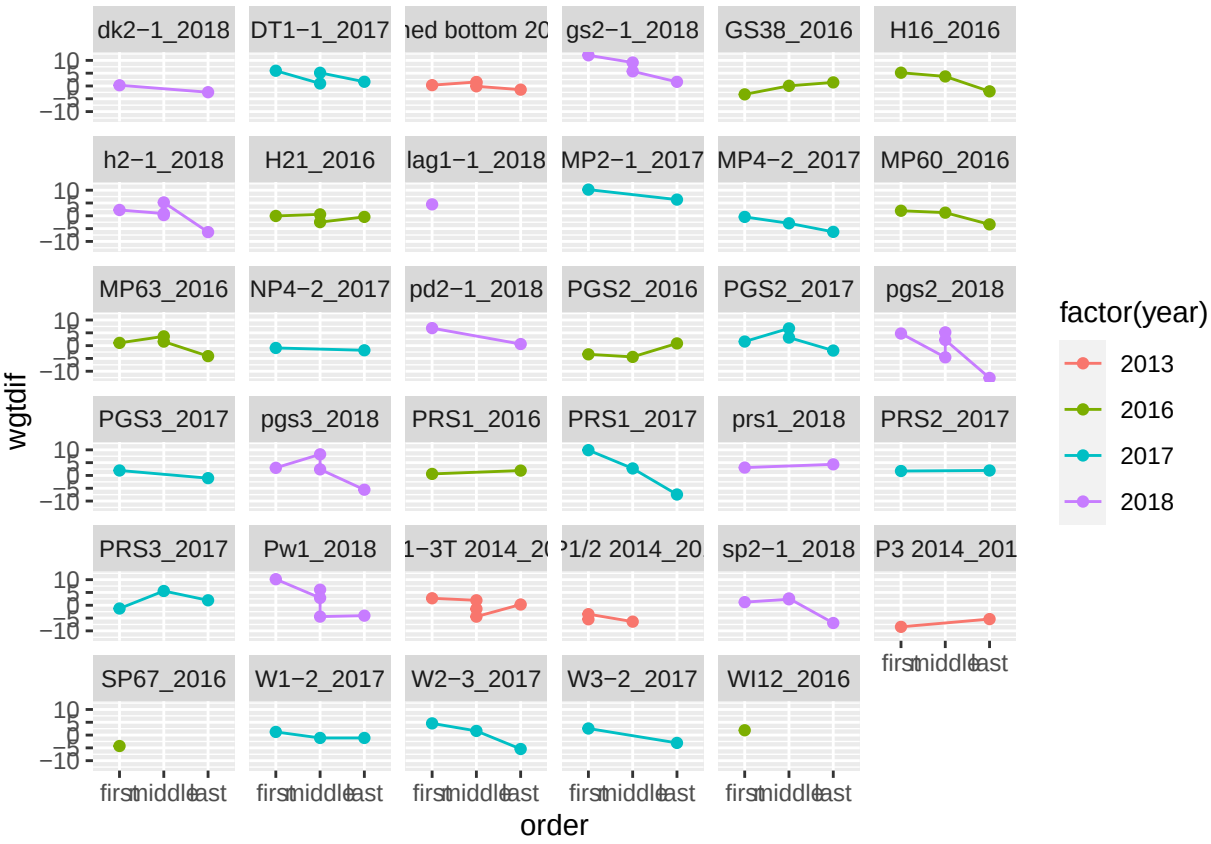

```
ggplot(obp,aes(x= factor(year),y=wgtdif,colour=order))+
  geom_point(position = position_dodge(width = 0.2))
```



We can try sub-dividing to the nest (eventid) level. This may give us some direct insight into order effects, and it may also lead to questions about the data! Which nest have no information about order effects? How will data from these nests be useful?

```
ggplot(obp,aes(x=order,y=wgtdif,colour=factor(year)))+
  geom_point() +
  geom_line(aes(group = eventid)) +
  facet_wrap(~eventid)
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



A statistical model for assessing how hatch order affects body condition

We refer to the nest (eventid) as a *blocking factor*. The effect of eventid on wgtdiff is called a *random effect*.

We introduce a new function lmer() in the package lmerTest. We need this function to fit a model that allows us to add eventid as a random effect.

```
model.obp<-lmer(wgtdif ~ order + year + (1|eventid), data = obp)
summary(model.obp)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: wgtdif ~ order + year + (1 | eventid)
##    Data: obp
##
## REML criterion at convergence: 575.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.90928 -0.44477  0.02247  0.69593  2.03981
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  eventid  (Intercept)  2.67    1.634
##  Residual             12.04    3.470
## Number of obs: 106, groups:  eventid, 35
```

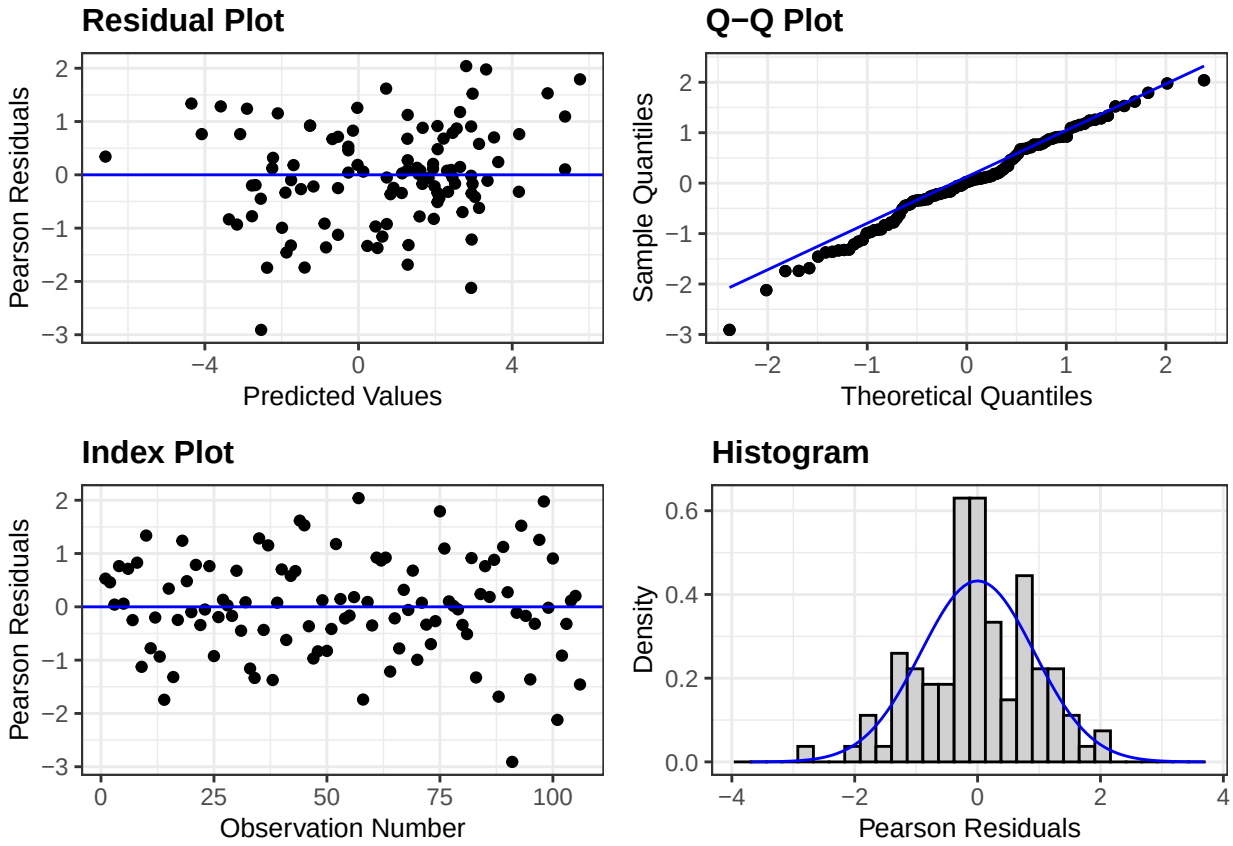
```
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept) -1914.9711   577.4870    25.2802  -3.316  0.00276 **
## ordermiddle    -0.3895     0.8305    86.5641  -0.469  0.64022
## orderlast      -4.2047     0.8575    74.6334  -4.903 5.36e-06 ***
## year            0.9507     0.2864    25.2829   3.320  0.00274 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) ordrmd ordrls
## ordermiddle -0.004
## orderlast    0.041  0.481
## year        -1.000  0.003 -0.041
```

The summary output includes "Random effects" and "Fixed effects". "Fixed effects" tells us about the effect of order and year on body condition. The intercept term is large and negative because year is a numerical variable around 2000. Centering the year variable will yield a more interpretable intercept term. Order_middle and order_last terms are contrasts from order = first. As expected, body condition depends upon hatch order. The year term estimates that body condition increases by approximately 1g per year.

"Random effects" estimates the variation attributed to blocking factors (between-nest variability) and residual variation. This variance structure is important for inferring treatment effects. For this experiment, order effects are measured within nests, so the residual variation is the relevant "noise" term. The year effect is measured between nests, so between-nest variation is the relevant "noise" term for inference around year.

Let's look at the residual plots for this model:

```
resid_panel(model.obp)
```



We would like to plot a summary of the model. We can either extract mean (SE) body condition for each order and year, or we can average the order effects across years to get *marginal order effects*. In this situation, we focus on order effects, so we opt for the marginal means. How would we change the code to get means for each year?

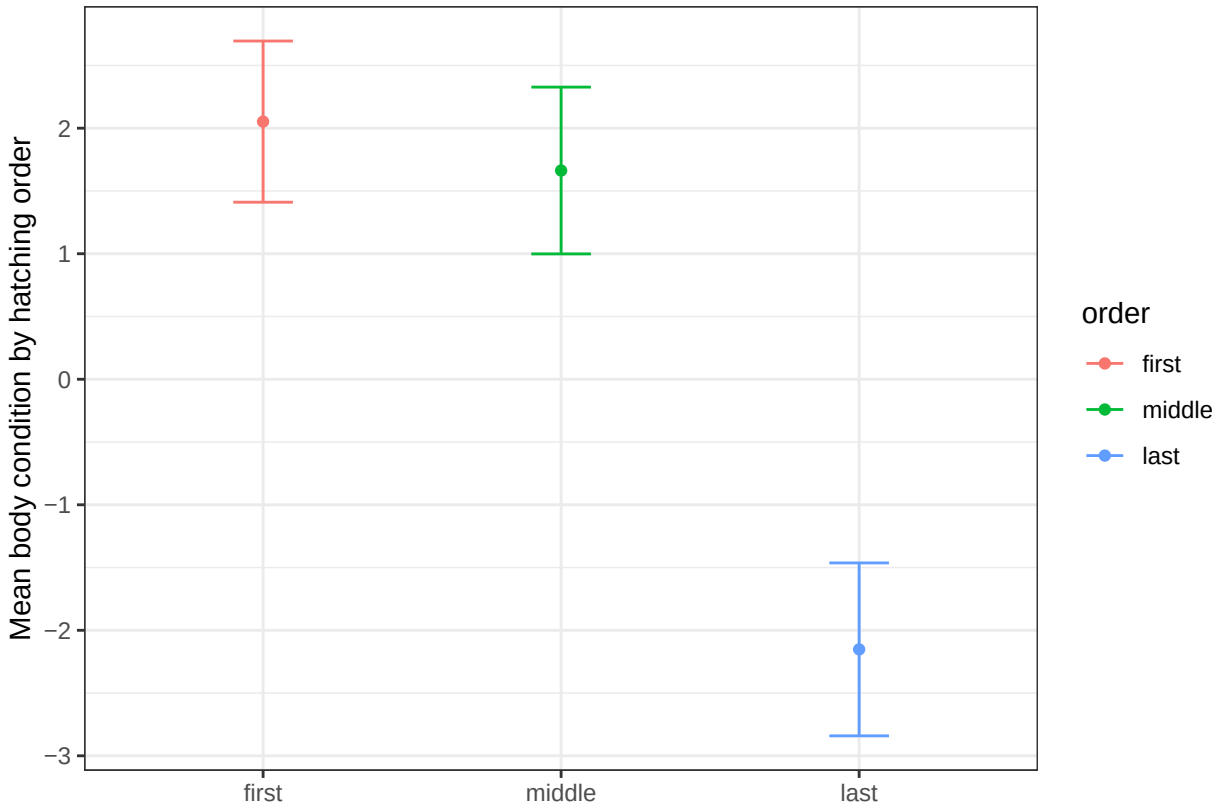
```
results1 <- emmeans(model.obp, ~order) %>%
             as_tibble()

results1
```

```
## # A tibble: 3 x 6
##   order  emmean    SE    df lower.CL upper.CL
##   <fct>   <dbl> <dbl> <dbl>    <dbl>    <dbl>
## 1 first    2.05 0.642  91.5    0.777    3.33
## 2 middle   1.66 0.664  70.9    0.338    2.99
## 3 last    -2.15 0.689  95.2   -3.52    -0.785
```

We plot the model summary using points and error bars:

```
ggplot(results1, aes(order, y = emmean, col = order))+
  geom_point()+
  geom_errorbar(aes(ymin = emmean-SE, ymax=emmean+SE), width = 0.2)+
  ylab("Mean body condition by hatching order")+ xlab("")+
  theme_bw()
```



Re-visiting the legume experiment

Let's return to the legume experiment from Lecture 6. Recall that there were 6 trays with 24 plants in each tray. The researchers arranged the treatments so that there were 4 plants with each of the 6 nitrate treatments in each tray.

Although tray is not a factor of interest, it has a large impact on our response variable, so it should be included in the statistical model. Nitrate treatments will be compared within trays, so whether tray is included as a fixed factor or as a random factor, our inference about the effect of nitrate treatments will not change. Let's check this:

Import data

```
legume <-read_csv("../Data/legume nitrate experiment.csv")
```

```
## Rows: 144 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): ID, tray, nitrate, RMR, log_RMR
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

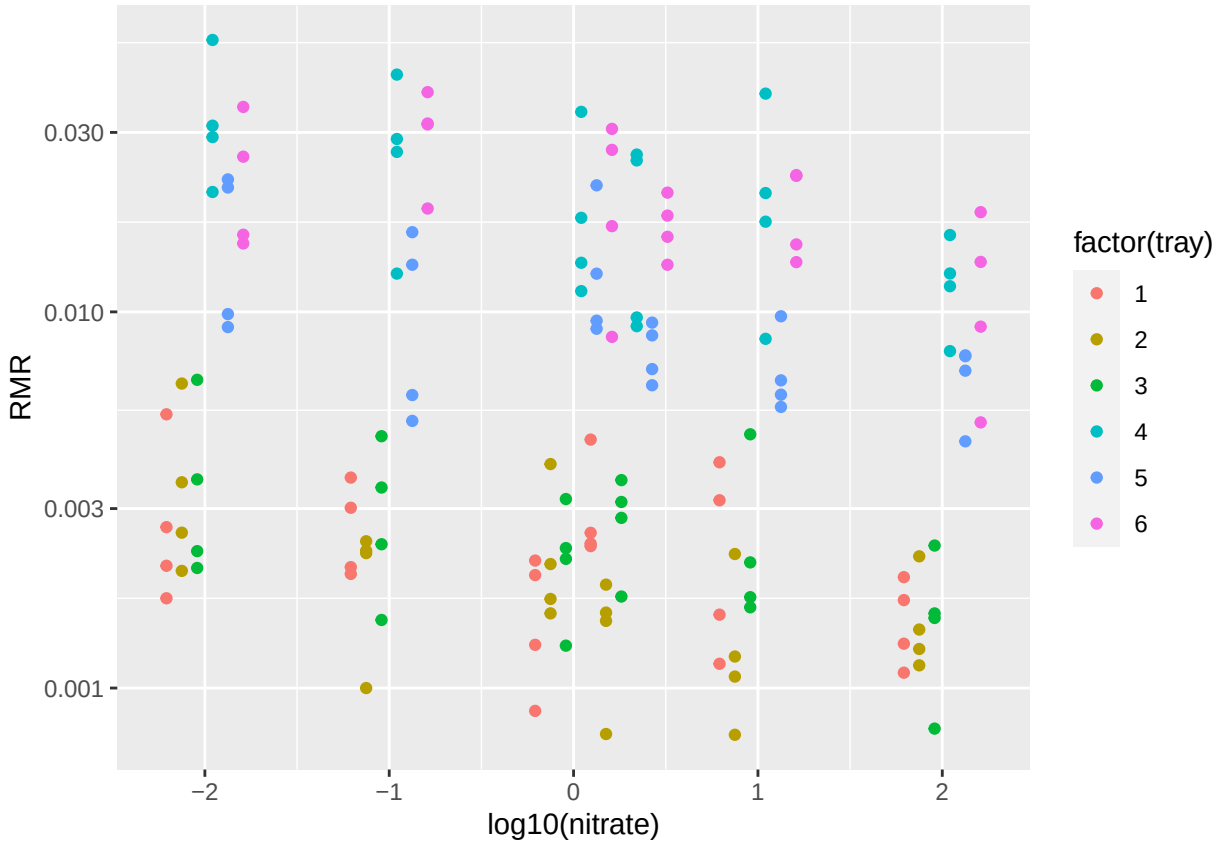
```
glimpse(legume)
```

```
## Rows: 144
## Columns: 5
## $ ID      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,~
## $ tray    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ nitrate <dbl> 1e-02, 1e-02, 1e-02, 1e-02, 1e-01, 1e-01, 1e-01, 1e-01, 1e+00,~
## $ RMR     <dbl> 0.002677648, 0.005342680, 0.002113807, 0.001732590, 0.00301487~
## $ log_RMR <dbl> -5.922817, -5.232028, -6.159265, -6.358138, -5.804198, -5.6193~
```

We visualise our data before fitting a model. What patterns do you notice?

```
ggplot(legume, aes( x=log10(nitrate), y=RMR, col=factor(tray)))+
  geom_point(position=position_dodge(width = 0.5))+
  scale_y_log10()
```

```
## Warning: position_dodge requires non-overlapping x intervals
```



Modelling legume data with tray as a fixed or random effect

When tray is treated as a fixed factor, we use the function lm(). When tray is treated as a random factor, we use the function lmer().

```
model.legume.fixed<-lm(log(RMR) ~ log10(nitrate) + factor(tray), data = legume)
anova(model.legume.fixed)
```

```
## Analysis of Variance Table
##
## Response: log(RMR)
##                 Df  Sum Sq Mean Sq F value    Pr(>F)
## log10(nitrate)   1   8.634  8.6338   49.00 1.042e-10 ***
## factor(tray)     5 145.744 29.1488  165.43 < 2.2e-16 ***
## Residuals      137  24.139  0.1762
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model.legume.random<-lmer(log(RMR) ~ log10(nitrate) + (1|tray), data = legume)
anova(model.legume.random)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
## log10(nitrate) 8.6338  8.6338     1   137      49 1.042e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The summary() functions give different output. Under the fixed effects model, the contrasts between trays is given explicitly Under the mixed effects (lmer) model, tray is a variance component. The variation in the response is partitioned into variation *between-trays* (Intercept) and *within trays* (Residual). The between-tray variation is much larger than the within-tray variation.

Model summaries (with parameter estimates)

Notice that the parameter estimates corresponding to the nitrate effect are the same for the two models. The intercept terms are different though: in model.legume.fixed, the intercept term is the intercept for tray1. In the mixed effects model, the intercept is an average across trays.

```
summary(model.legume.fixed)
```

```
##
## Call:
## lm(formula = log(RMR) ~ log10(nitrate) + factor(tray), data = legume)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.91263 -0.26403  0.00374  0.28274  0.89101
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -6.13486    0.08569 -71.590  < 2e-16 ***
## log10(nitrate) -0.18896    0.02699  -7.000 1.04e-10 ***
## factor(tray)2  -0.17782    0.12117  -1.467    0.145
## factor(tray)3   0.10174    0.12117   0.840    0.403
## factor(tray)4   2.18102    0.12117  17.999  < 2e-16 ***
## factor(tray)5   1.44273    0.12117  11.906  < 2e-16 ***
## factor(tray)6   2.14183    0.12117  17.676  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.4198 on 137 degrees of freedom
## Multiple R-squared:  0.8648, Adjusted R-squared:  0.8589
## F-statistic:   146 on 6 and 137 DF,  p-value: < 2.2e-16
```

```
summary(model.legume.random)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: log(RMR) ~ log10(nitrate) + (1 | tray)
##    Data: legume
##
## REML criterion at convergence: 192.4
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.18782 -0.62945  0.00564  0.66103  2.10645
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  tray     (Intercept) 1.2072   1.0987
##  Residual             0.1762   0.4198
## Number of obs: 144, groups:  tray, 6
##
## Fixed effects:
##                 Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)     -5.18661    0.44992   5.00009  -11.53 8.61e-05 ***
## log10(nitrate)  -0.18896    0.02699 137.00000   -7.00 1.04e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## log10(ntrt) -0.003
```

In the random effects model, we may want to understand which trays had the highest and lowest values of root:mass ratio. We can see this using the function ranef(). The random effects will average to 0. Compare the random effects with the fixed effects contrasts in the summary() of the fixed effects model.

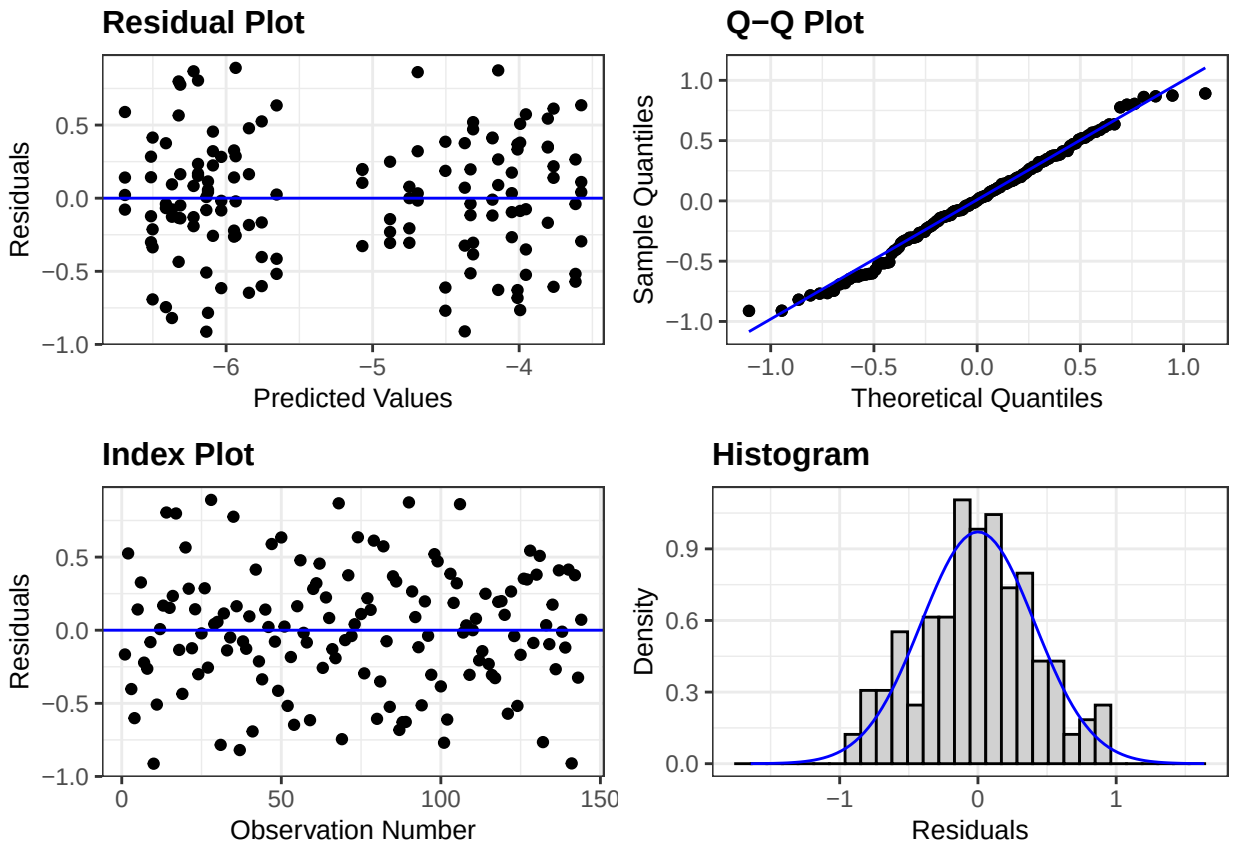
```
ranef(model.legume.random)
```

```
## $tray
##   (Intercept)
## 1  -0.9425168
## 2  -1.1192640
## 3  -0.8413938
## 4   1.2253157
## 5   0.4914910
## 6   1.1863679
##
## with conditional variances for "tray"
```

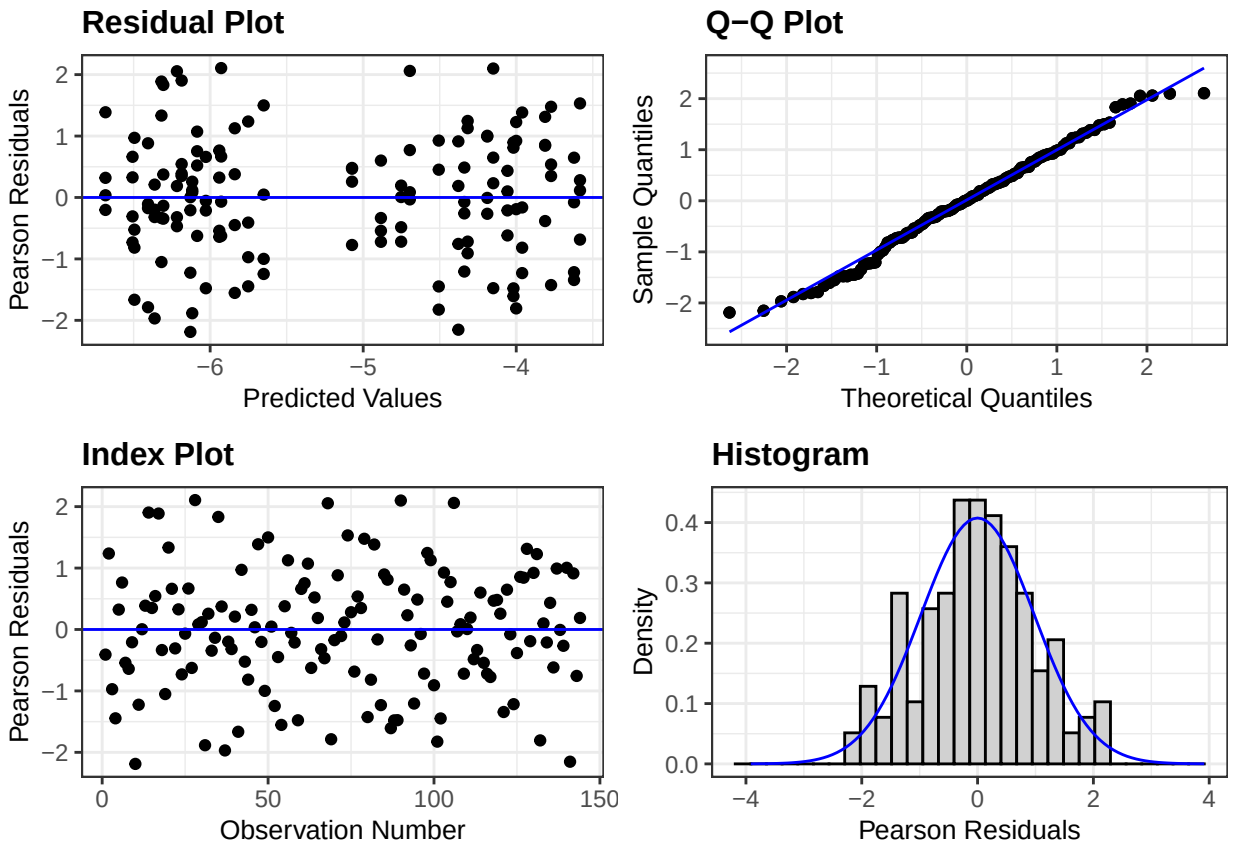
Checking model assumptions

Let's compare the residual plots between the two models. They should look the same, because both residuals correspond to the observed response after removing the effects of nitrate concentration and tray.

```
resid_panel(model.legume.fixed)
```




```
resid_panel(model.legume.random)
```



Comparing emmeans() for marginal means and their standard errors

The standard errors for the mean estimates differ between the fixed effects and the mixed effects models. The reason is evident: In the lmer() model, tray is part of the variation term. The standard error reflects the uncertainty of the mean for a random tray. In the lm() model, we have conditioned upon this particular set of trays. The standard error of the mean reflects the uncertainty of the mean, conditional on the trays that were used.

So both standard errors are "correct"; which one to use depends upon the context of the problem.

```
emmeans(model.legume.fixed, ~ log10(nitrate), at = list(nitrate = c(0.01, 0.1, 1, 2, 10, 100)), type =
```

```
##  nitrate response       SE  df lower.CL upper.CL
##    1e-02  0.00816 0.000534 137  0.00717  0.00929
##    1e-01  0.00675 0.000304 137  0.00618  0.00738
##    1e+00  0.00559 0.000196 137  0.00522  0.00599
##    2e+00  0.00528 0.000188 137  0.00492  0.00567
##    1e+01  0.00463 0.000201 137  0.00425  0.00504
##    1e+02  0.00383 0.000242 137  0.00338  0.00434
##
## Results are averaged over the levels of: tray
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

```
emmeans(model.legume.random, ~ log10(nitrate), at = list(nitrate = c(0.01, 0.1, 1, 2, 10, 100)), type =
```

```
##  nitrate response      SE   df lower.CL upper.CL
##    1e-02  0.00816 0.00370 5.15  0.00257   0.0259
##    1e-01  0.00675 0.00304 5.04  0.00213   0.0215
##    1e+00  0.00559 0.00252 5.00  0.00176   0.0178
##    2e+00  0.00528 0.00238 5.00  0.00166   0.0168
##    1e+01  0.00463 0.00209 5.03  0.00146   0.0147
##    1e+02  0.00383 0.00174 5.14  0.00121   0.0122
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```