Partitioning Variance in your Data - ANOVAs and GLMMs

A Michelle Edwards, Ph.D., MLIS

September 3, 2019

Table of Contents

Setting up our R Studio

Let's first set up our working directory. Last week I suggested avoiding your Desktop as a working directory, but it did work - so please select a spot on your laptop that we will use as our working directory - the place where we will save our Excel file, and work from - for this workshop.

```
setwd("~/Workshops/R/F19")
```

Let’s also install - if required the following packages:

- readxl - most of you will have this from last week
- tidyverse - most of you will have this
- lme4 - this is a new one!
- emmeans - this is a new one!

Remember there are a few ways to install a new package:

1. Using code: install.package(“lme4”)
2. Using the menus - Tools -> Install Packages -> Enter the name of the package in the dialogue box -> ensure that the install dependencies is checked

Once you have the packages installed, we need to load them for our session. Two different ways to do this as well:

1. In R Studio - right hand bottom box - go to the Packages tab - scroll down to the package you want to load - put a check in the box next to it.
2. In the script window - using code - library()

Exercise 1

1. Install the packages you are missing
2. Load all 4 packages.

```
library(readxl)
library(tidyverse)

## -- Attaching packages --------------------------------------------------- ti
dyverse 1.2.1 --

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.1     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ---------------------------------------------------- tidyvers
e_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

library(lme4)

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:tidyr':
##
##     expand

library(emmeans)
```

Reading and cleaning the data

Download the Excel file from the OACStats blog post and save it in the directory you chose to use as your working directory for this workshop.

Let’s work together and read the default worksheet (sheet1) from the RCBD_excel Excel file and called it **rcbd**

```
rcbd <- read_excel("~/Workshops/Common_Files/RCBD_excel.xlsx", col_names=T)
rcbd

## # A tibble: 24 x 5
##    block  trmt Nitrogen  Weed Bin_weed
##    <dbl> <dbl>    <dbl> <dbl>    <dbl>
##  1     1     1     35.0    81     0.81
##  2     2     1     41.2    87     0.87
##  3     3     1     36.9    89     0.89
##  4     4     1     40.0    79     0.79
##  5     1     2     40.9    88     0.88
##  6     2     2     46.7    85     0.85
##  7     3     2     46.6    99     0.99
##  8     4     2     41.9    86     0.86
##  9     1     3     42.1    54     0.54
## 10     2     3     49.4    23     0.23
## # ... with 14 more rows
```

Review the contents of the data you just loaded. Anything unusual about it? Make sure you look at the data types in the data. For our upcoming analysis, I would like us to ensure that nitrogen, weed, and bin_weed are numeric. Then I would like us to change block and trmt to factors. Why? These variables are not variables that we will use to calculate values from - so we won’t calculate a mean for tree - but these are all “classification” or “factors”. They tell use what group our observations belong to - or they classify our data.

```
summary(rcbd)

##      block           trmt       Nitrogen          Weed      
##  Min.   :1.00   Min.   :1.0   Min.   :34.89   Min.   : 2.00  
##  1st Qu.:1.75   1st Qu.:2.0   1st Qu.:38.90   1st Qu.:10.75  
##  Median :2.50   Median :3.5   Median :41.56   Median :40.50  
##  Mean   :2.50   Mean   :3.5   Mean   :42.07   Mean   :46.38  
##  3rd Qu.:3.25   3rd Qu.:5.0   3rd Qu.:44.89   3rd Qu.:82.00  
##  Max.   :4.00   Max.   :6.0   Max.   :52.68   Max.   :99.00  
##     Bin_weed     
##  Min.   :0.0200  
##  1st Qu.:0.1075  
##  Median :0.4050  
##  Mean   :0.4637  
##  3rd Qu.:0.8200  
##  Max.   :0.9900  
```

So let’s clean our data to ensure we have numeric data and change others to factors.

```
rcbd$block <- as.factor(rcbd$block)
rcbd$trmt <- as.factor(rcbd$trmt)
rcbd$Nitrogen <- as.numeric(rcbd$Nitrogen)
rcbd$Weed <- as.numeric(rcbd$Weed)
rcbd$Bin_weed <- as.numeric(rcbd$Bin_weed)
```

Run the summary() again to make sure everything was changed. Also note the changes in your Global Environment window.

Sidenote: Case of Variable Names?

Can *YOU* remember what is capital and what is not??

Let’s use another package called **stringr** to help us change all of our variable names to lower case - then we don’t have to remember what was upper case and what was not.

First let’s review the variable names or the column names by using the colnames() function.

```
# Reviewing the column or variables names of our **rcbd** data
colnames(rcbd)

## [1] "block"    "trmt"     "Nitrogen" "Weed"     "Bin_weed"
```

Note that three of our column or variable names begin with a capital letter. Let’s use the str_to_lower() function in the **stringr** package to change them all to lowercase.

```
colnames(rcbd) <- str_to_lower(names(rcbd), locale = "en")

# Review the column or variable names to see whether they changed or not
colnames(rcbd)

## [1] "block"    "trmt"     "nitrogen" "weed"     "bin_weed"
```

Alrighty - we’re all set to begin our analysis now. If this were your data and your statistical analyses, you would probably do some intial data visualizations and run some descriptive statistics to get a feeling for your data.

Analyzing our data - Paritioning of the variation - Analysis of variation (ANOVA) - Are there treatment differences?

Many of our research projects, we are implementing “some” treatment. We do this to answer a specific research question. Maybe there are beliefs that a pariticular treatment will provide more fruit than another. Or maybe a particular treatment will reduce the severity of a disease. Just remember that we are doing this to answer a specific research question.

For many research projects, we want to determine whether there are differences between the treatments we impose. To test for these differences, we design an experiment. Ideally, we design the experiment in a way that will allow us to answer our research question.

Let’s review a few aspects of an experimental design:

- Experimental Unit: The unit to which the treatment is applied. Sometimes we lose track of what we are measuring and what we are testing. This is a key component to any analysis.
- Identify and be clear about the measurements you are taking - keep them as similar as possible - ensure we are measuring the same thing every time!
- Treatments - make sure you are applying them the same way to every experimental unit

In an ideal world and perfect world, our experimental units would be identical, our treatments would be applied identically, our measures would be perfect, leading to the only differences to be seen attributed to the treatments. Is this possible? NOPE - why?

- There is natural variation between experimental units
- There will be variability in the measurements we take
- We just cannot replicate our treatments exactly
- Some of our experimental units might react differently to the same treatement
- Other factors that may play a role - weather, lighting, etc…

All of these are what we refer to as sources of experimental error.

Our goal with an experimental design is to control the experimental error - we want to be able to explain the variation or difference we see in our measures in a way that will lead us to saying Yes there are differences between our treatments or NO there are not - while knowing we did the best we could containing that random error.

Goal of ANOVA - as we traditionally come to know it - is to partition the variation in our outcome measures. In other words, to be able to explain the variation in our outcome measures and conclude whether our treatments were similar or not.

RCBD Example

Let’s review the data collected from a small RCBD trial. There were 4 blocks, where 6 treatments were randomly assigned to each. The statistical model for this experimental design is:

$\text{Nitrogen}_{ij} = \mu + \text{block}_i + \text{trmt}_j + e_{ij}$

Where:

- Nitrogen_{ij} = nitrogen measure taken on plot_{ij}
- μ = overall mean of nitrogen
- block_i = random effect of block_i
- trmt_j = fixed effect of treatment $_j$
- e_{ij} = random experimental error

Fixed vs random effects

Fixed effects are something you want to study - you set out the levels that you are interested in. You “fix” the levels. The results from your experiment can only talk about the levels you studied.

- Example #1: I want to see whether 1st year students prefer Coke or Pepsi
- Example #2: I want to see the effect of 3 levels of fertilizer on my crop

Random effects are factors in your design that may contribute variation in your outcome measure, but you are not interested in it. You only want to account for it, before looking at your treatment effects.

- Example #1: I want to study the effect of fertilizer on my crop
- Example #2: Block effect, Weather, etc…

Let’s first try running our data using the aov() function in base R. By doing this we are going to assume that we have a CRD or a Completely Randomized Design - no block effect. I want to do this to show you the differences between a fixed effects model and the RCBD model or a mixed effects model we will run in a moment

```
#FIXED effects model - looking at trmt differences in nitrogen

model1 <- aov(nitrogen ~ trmt, data=rcbd)
```

Note that we saved the results of the ANOVA in an object called **model1** If you want to see it you have to call it up.

Try it out to see what you get

```
model1

## Call:
##    aov(formula = nitrogen ~ trmt, data = rcbd)
```

```
##
## Terms:
##                      trmt Residuals
## Sum of Squares  201.3164  305.0124
## Deg. of Freedom        5        18
##
## Residual standard error: 4.116446
## Estimated effects may be unbalanced
```

If you recall from last week, I mentioned the summary() function and that it was a very versatile function, used for a number of analyses. Let's try it out here

```
summary(model1)

##              Df Sum Sq Mean Sq F value Pr(>F)
## trmt          5  201.3   40.26   2.376 0.0802 .
## Residuals    18  305.0   16.95
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Another function that is handy with most ANOVA results is the anova() function. Remember to put object name in the ()

```
anova(model1)

## Analysis of Variance Table
##
## Response: nitrogen
##            Df Sum Sq Mean Sq F value  Pr(>F)
## trmt       5 201.32  40.263  2.3761 0.08024 .
## Residuals 18 305.01  16.945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let's review the output together.

Testing the Assumptions

Something that we learned when we were first taught ANOVAs was something about assumptions. One of those assumptions, was that the data going into the ANOVA had to have a normal distribution. I'm going to tell you now - to not worry about that. I remember being told this and also being told that ANOVA is a robust analysis for data that isn't too normal going in.

So - what assumptions should we be checking?

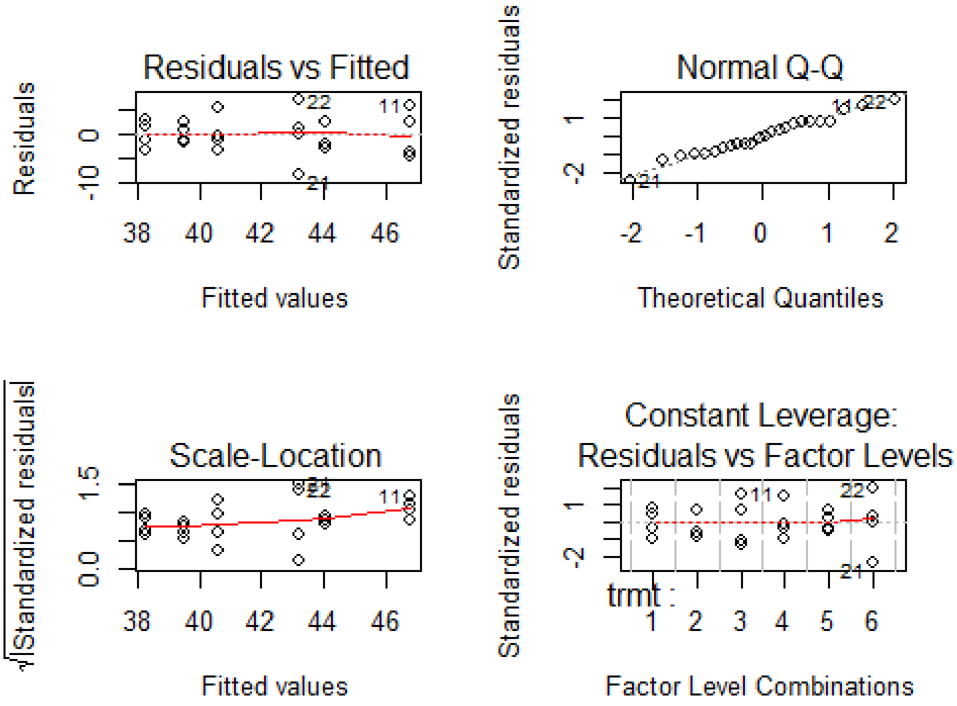
1. Residuals are random - no relationship to our treatment
2. Homogeneity of residuals across our treatment groups
3. Residuals are normally distributed
4. Residuals have a mean of 0

It's ALL about the residuals - we are testing the residuals to determine whether our model fits our data.

So how do we do this?

1. Plots
2. Shapiro Wilk test for normality

```
shapiro.test(residuals(model1))

## 
##  Shapiro-Wilk normality test
## 
## data:  residuals(model1)
## W = 0.97982, p-value = 0.8926

par(mfrow = c(2, 2))  # Split the plotting panel into a 2 x 2 grid
plot(model1)
```



Let’s review the results together.

Once we are happy with our assumptions, then we can dig further into our results to answer which treatment is different from which. I like to run the Tukeys test. What this does is adjust the p-value to protect you from comparison-wise error. It makes the p-value more conservative and protects you from identifying differences when they really may not exist.

Means Comparisons

```
TukeyHSD(model1)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = nitrogen ~ trmt, data = rcbd)
##
## $trmt
##        diff         lwr      upr     p adj
## 2-1  5.7550  -3.4955196 15.00552 0.3916933
## 3-1  8.4925  -0.7580196 17.74302 0.0827902
## 4-1  2.3375  -6.9130196 11.58802 0.9632649
## 5-1  1.2325  -8.0180196 10.48302 0.9979583
## 6-1  4.9475  -4.3030196 14.19802 0.5489169
## 3-2  2.7375  -6.5130196 11.98802 0.9304478
## 4-2 -3.4175 -12.6680196  5.83302 0.8431937
## 5-2 -4.5225 -13.7730196  4.72802 0.6365895
## 6-2 -0.8075 -10.0580196  8.44302 0.9997353
## 4-3 -6.1550 -15.4055196  3.09552 0.3233854
## 5-3 -7.2600 -16.5105196  1.99052 0.1775522
## 6-3 -3.5450 -12.7955196  5.70552 0.8226944
## 5-4 -1.1050 -10.3555196  8.14552 0.9987882
## 6-4  2.6100  -6.6405196 11.86052 0.9423789
## 6-5  3.7150  -5.5355196 12.96552 0.7936263
```

From the results we can see that all of our treatments are similar.

Mixed Model Analysis

A mixed model analysis is using a model that has fixed and random effects - mixed. How do we know if we have a mixed model? Always go back to your experimental design. Sorry - everything comes from that design. Earlier I said we were treating our data as a CRD - but I noted that this was not the case - that we were using it as an example.

Our experiment was conducted as an RCBD.

$\text{Nitrogen}_{ij} = \mu + \text{block}_i + \text{trmt}_j + e_{ij}$

trmt is a fixed effect - we are interested in differences between the treatments we set out in this trial. **block** however was our way to help explain variation we may see in the field - so it is not something I want to see if there are differences between, it is something that I want to acknowledge may exist, account for it, and then move one. So it is added to our model to explain some of the variation in our outcome variable and reduce our experimental error.

With a mixed model, we can no longer use the base R aov(). So now we will start using the lme4 package. Let's use the code below to run our Mixed Model ANOVA. Notice to identify an effect as random we need to say (1|random effect)

Let's try it out:

```
model2 <- lmer(nitrogen ~ ((1|block) + trmt), data=rcbd)
model2

## Linear mixed model fit by REML ['lmerMod']
## Formula: nitrogen ~ ((1 | block) + trmt)
##    Data: rcbd
## REML criterion at convergence: 101.5658
## Random effects:
##  Groups   Name        Std.Dev.
##  block    (Intercept) 3.122
##  Residual             2.683
## Number of obs: 24, groups:  block, 4
## Fixed Effects:
## (Intercept)        trmt2        trmt3        trmt4        trmt5
##      38.277        5.755        8.493        2.338        1.233
##       trmt6
##       4.948
```

Remember we saved the results in an object called model2. How can you view the contents?

```
anova(model2)

## Analysis of Variance Table
##      Df Sum Sq Mean Sq F value
## trmt  5 201.32  40.263  5.5917
```

```
summary(model2)

## Linear mixed model fit by REML ['lmerMod']
## Formula: nitrogen ~ ((1 | block) + trmt)
##    Data: rcbd
##
## REML criterion at convergence: 101.6
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.75517 -0.42267 -0.04919  0.61553  1.85682
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  block    (Intercept) 9.745    3.122
##  Residual             7.200    2.683
## Number of obs: 24, groups:  block, 4
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   38.277      2.058  18.597
## trmt2          5.755      1.897   3.033
## trmt3          8.493      1.897   4.476
## trmt4          2.338      1.897   1.232
## trmt5          1.233      1.897   0.650
## trmt6          4.948      1.897   2.607
##
## Correlation of Fixed Effects:
##       (Intr) trmt2  trmt3  trmt4  trmt5
## trmt2 -0.461
## trmt3 -0.461  0.500
## trmt4 -0.461  0.500  0.500
## trmt5 -0.461  0.500  0.500  0.500
## trmt6 -0.461  0.500  0.500  0.500  0.500
```

Can you see any differences between these 2 outputs? Let's also work through the output. What's missing???

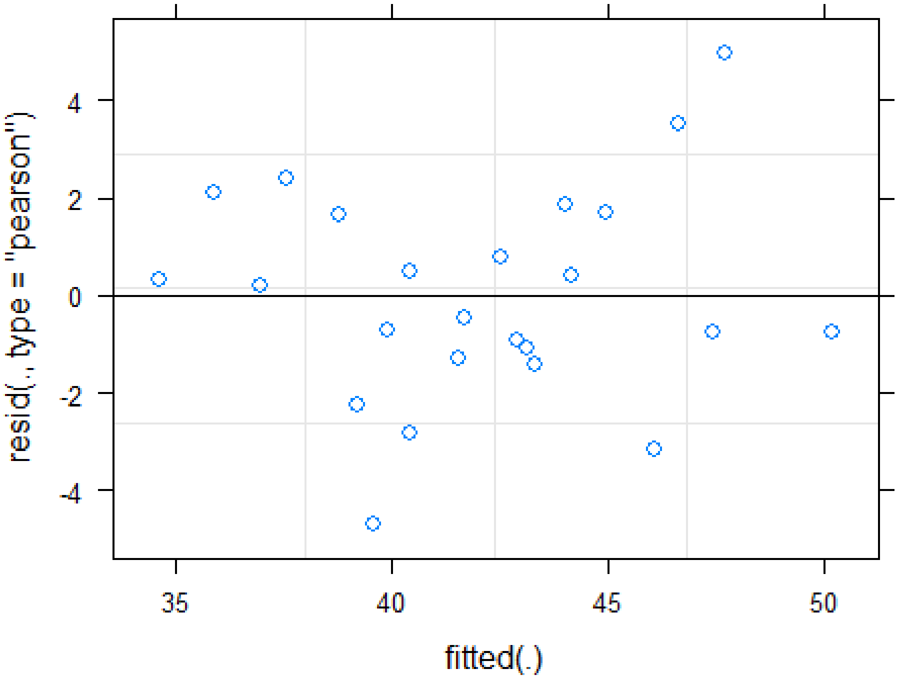
To calculate the p-value (if you feel you need it), you need the F-value, Treatment df, and Error df.

```
#  Calculating p-values
#
1-pf(355.07,2,72)

## [1] 0
```

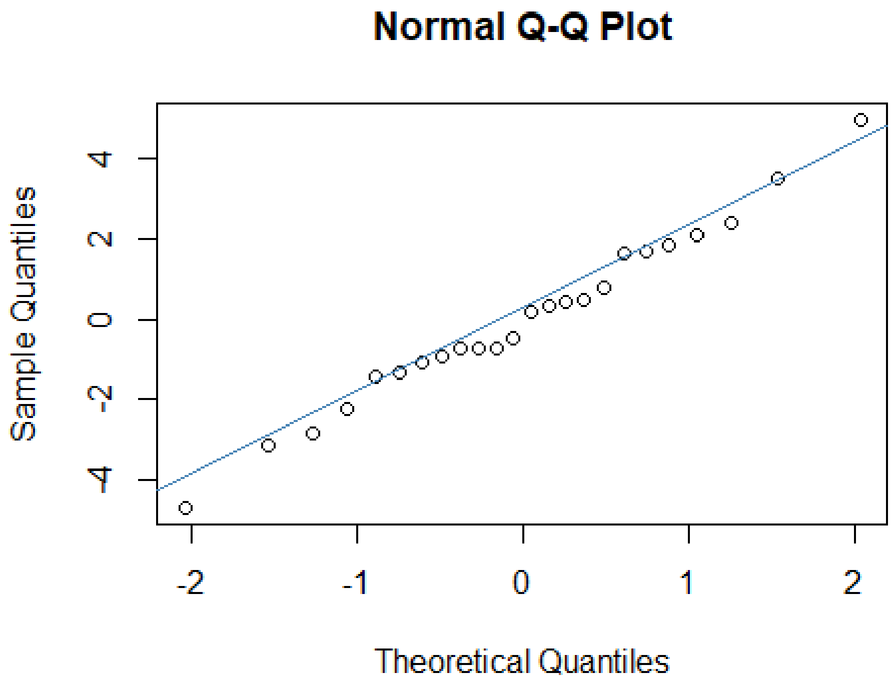
Now before we go too far, let's test our assumptions

Testing the Assumptions

```
# Shapiro-Wilk normality test
shapiro.test(residuals(model2))

##
##  Shapiro-Wilk normality test
##
## data:  residuals(model2)
## W = 0.98853, p-value = 0.9916

# Homogeneity plots
par(mfrow = c(1, 1))
plot(model2)
```



```
# Normality plots
qqnorm(residuals(model2))
qqline(residuals(model2), col = "steelblue")
```



So? What do we conclude? are we happy with this?

Let's run the means comparisons to see what's happening?

Means Comparisons

```
model2.emmean <- emmeans(model2, "trmt", adjust="tukey", type="response")
model2.emmean

##  trmt emmean   SE   df lower.CL upper.CL
##  1      38.3 2.06 6.78     33.4     43.2
##  2      44.0 2.06 6.78     39.1     48.9
##  3      46.8 2.06 6.78     41.9     51.7
##  4      40.6 2.06 6.78     35.7     45.5
##  5      39.5 2.06 6.78     34.6     44.4
##  6      43.2 2.06 6.78     38.3     48.1
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95

pairs(model2.emmean, type="response")

##  contrast estimate  SE df t.ratio p.value
##  1 - 2      -5.755 1.9 15 -3.033  0.0742
##  1 - 3      -8.492 1.9 15 -4.476  0.0049
##  1 - 4      -2.337 1.9 15 -1.232  0.8150
##  1 - 5      -1.232 1.9 15 -0.650  0.9849
##  1 - 6      -4.947 1.9 15 -2.607  0.1553
##  2 - 3      -2.737 1.9 15 -1.443  0.7025
##  2 - 4       3.417 1.9 15  1.801  0.4934
##  2 - 5       4.522 1.9 15  2.383  0.2226
##  2 - 6       0.807 1.9 15  0.426  0.9978
##  3 - 4       6.155 1.9 15  3.244  0.0505
##  3 - 5       7.260 1.9 15  3.826  0.0168
##  3 - 6       3.545 1.9 15  1.868  0.4559
##  4 - 5       1.105 1.9 15  0.582  0.9907
##  4 - 6      -2.610 1.9 15 -1.376  0.7402
##  5 - 6      -3.715 1.9 15 -1.958  0.4079
##
## Degrees-of-freedom method: kenward-roger
## P value adjustment: tukey method for comparing a family of 6 estimates
```

```
multcomp::cld(model2.emmean)
```

```
##  trmt emmean   SE   df lower.CL upper.CL .group
##  1      38.3 2.06 6.78     33.4     43.2  1
##  5      39.5 2.06 6.78     34.6     44.4  1
##  4      40.6 2.06 6.78     35.7     45.5  12
##  6      43.2 2.06 6.78     38.3     48.1  12
##  2      44.0 2.06 6.78     39.1     48.9  12
##  3      46.8 2.06 6.78     41.9     51.7   2
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 6 estimates
## significance level used: alpha = 0.05
```

Take a note of the means comparisons results - are they the same as when we ran the data using a CRD? Why or why not?

Generalized Linear Mixed Model (GLMM)

The last type of analysis I'd like to work through - GLMM. Think of this term as the umbrella term for all types of ANOVAs. Our computing power and statistical methodologies have caught up and now allow us to analyse most of the different types of data we collect. As an example - our field data, we've been working with the nitrogen variable. Have you noticed that these are continuous measures - data types we are comfortable with because we can take the mean and talk about variation around that mean. But, we don't only collect continuous measures. For example, we may have a variable that counts an object - weeds in our example.

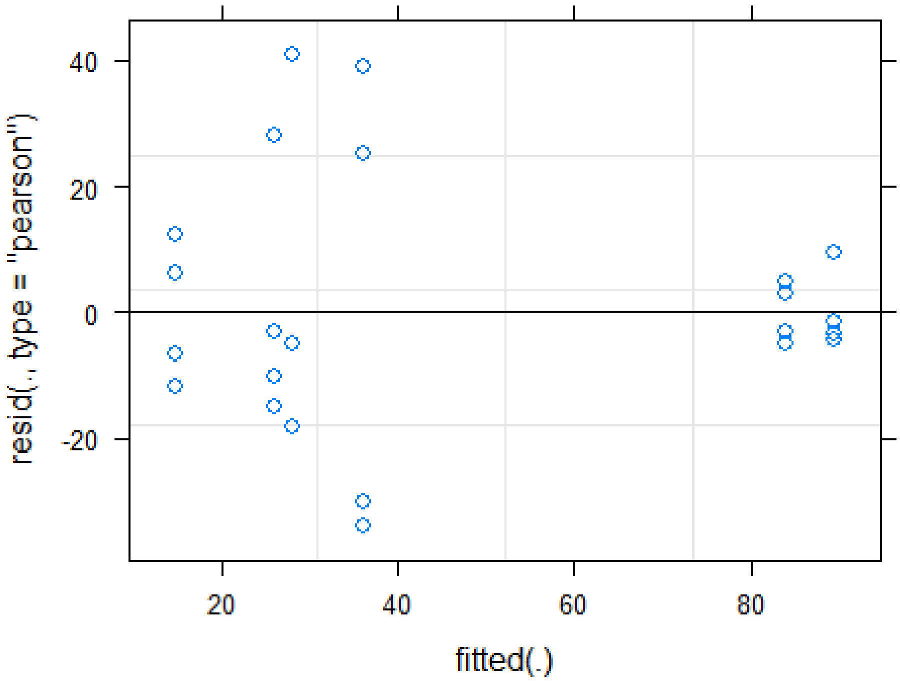
Until recently, we would force these types of data to make them normal and then run our ANOVA. But no more! We can now let our program now, what distribution we believe our data has, test the residuals to make sure our model fits the data, and we're off to the races. Sounds easy? Well… as you can imagine, finding that right distribution can be challenging. There are other aspects of our model that we can modify now, that we couldn't before, but let's wade into this new forum with changing the distribution.

Let's start with the RCBD mixed model and we will be studying the **weed** variable.

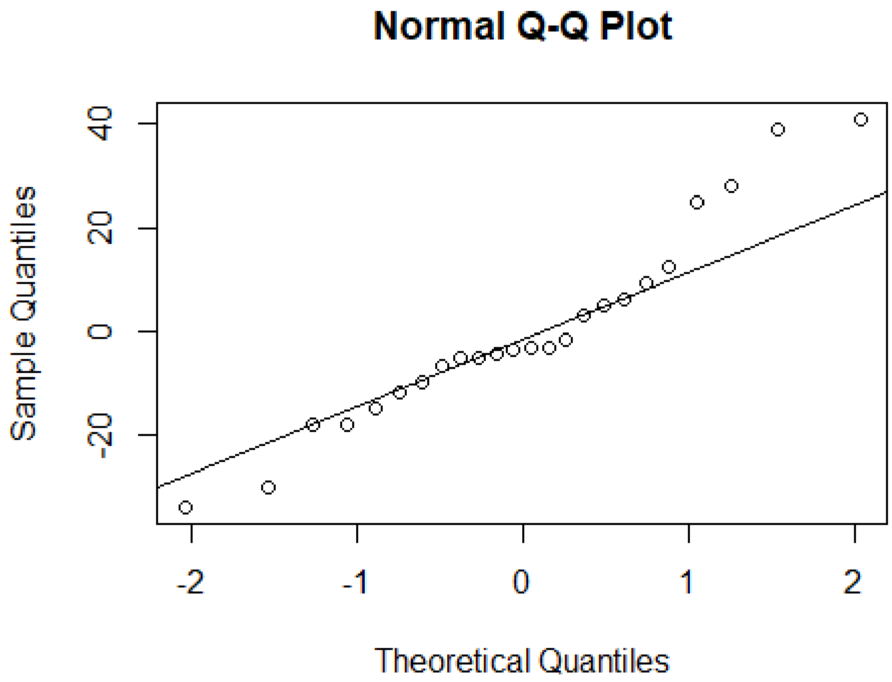
Our statistical model is the same - so try it out and see what happens with the results. Remember to check the residuals before getting too excited about the results!

```
#RCBD Mixed model analysis for weed variable

model3a <- lmer(weed ~ trmt + (1|block), data=rcbd)

## boundary (singular) fit: see ?isSingular

model3a

## Linear mixed model fit by REML ['lmerMod']
## Formula: weed ~ trmt + (1 | block)
##    Data: rcbd
## REML criterion at convergence: 169.6739
## Random effects:
##  Groups   Name        Std.Dev.
##  block    (Intercept)  0.0
##  Residual             21.4
## Number of obs: 24, groups:  block, 4
## Fixed Effects:
## (Intercept)        trmt2        trmt3        trmt4        trmt5
##       84.00         5.50       -58.00       -56.00       -48.00
##       trmt6
##      -69.25
## convergence code 0; 1 optimizer warnings; 0 lme4 warnings
```

```
anova(model3a)

## Analysis of Variance Table
##      Df Sum Sq Mean Sq F value
## trmt  5  20544  4108.8  8.9758

#Checking the residuals
# Shapiro-Wilk normality test
shapiro.test(residuals(model3a))

##
##  Shapiro-Wilk normality test
##
## data:  residuals(model3a)
## W = 0.94602, p-value = 0.2218

# Homogeneity plots
par(mfrow = c(1, 1))
plot(model3a)
```



```
# Normality plots
qqnorm(residuals(model3a))
qqline(residuals(model3a))
```



What do you think?

GLMM - Poisson Distribution

Based on the residuals we saw above, there are a few issues. So, let’s go back and think about our data. We have number of weeds counted in a plot. It’s a COUNT. We are counting how many weeds we see. Yes - someone can make the argument that the COUNT can come from a normal distribution - but - what range do we have in our data? It is a very narrow range and our residuals didn’t look the greatest so let’s try a different distribution.

Traditionally for COUNT data you would use a Poisson Distribution.

We will continue to use the lme4 package, but now we will use another function within that package. Let’s go back to our statistical model - has it changed? No! We have the same experimental design and the same research question, we just happen to have COUNT data rather than a continuous measure.

So, we will use the same model with the glmer(). Notice the code is the same as before, with the exception that we are now telling it what distribution family it belongs to: Poisson.

Let’s try it out:

```
model3b <- glmer(weed ~ trmt + (1|block), data=rcbd, family = poisson)
model3b

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: weed ~ trmt + (1 | block)
##    Data: rcbd
##       AIC       BIC    logLik  deviance  df.resid
##  407.9050  416.1514 -196.9525  393.9050        17
## Random effects:
##  Groups Name        Std.Dev.
##  block  (Intercept) 0.1456
## Number of obs: 24, groups:  block, 4
## Fixed Effects:
## (Intercept)        trmt2        trmt3        trmt4        trmt5
##     4.42026      0.06343     -1.17271     -1.09861     -0.84729
##       trmt6
##    -1.73957

anova(model3b)

## Analysis of Variance Table
##      Df Sum Sq Mean Sq F value
## trmt  5    382    76.4    76.4
```

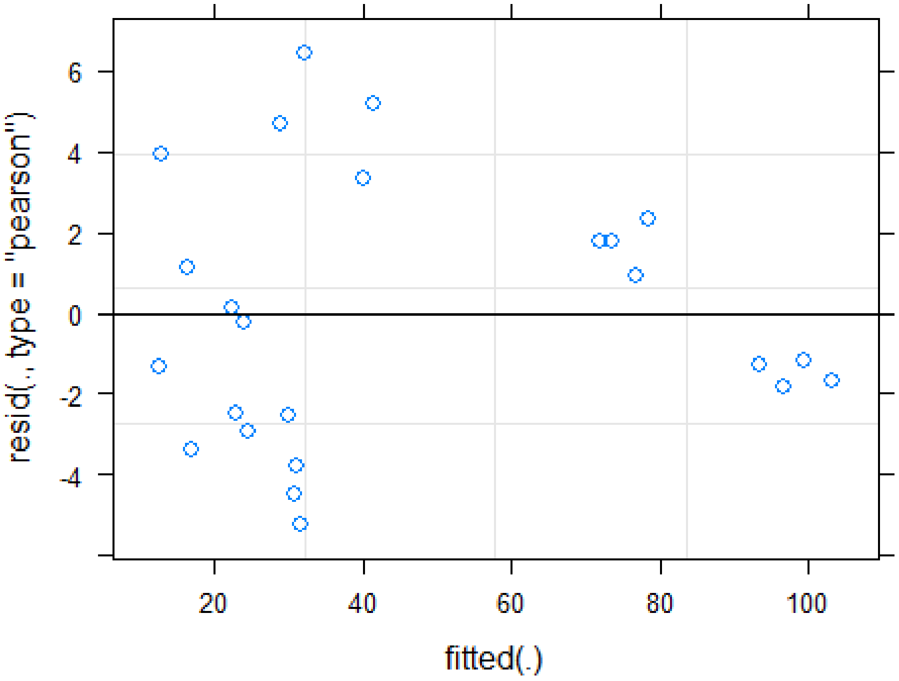
Checking Results

```
anova(model3b)

## Analysis of Variance Table
##      Df Sum Sq Mean Sq F value
## trmt  5    382    76.4    76.4

summary(model3b)

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: weed ~ trmt + (1 | block)
##    Data: rcbd
##
##      AIC      BIC   logLik deviance df.resid
##    407.9    416.2   -197.0    393.9       17
##
## Scaled residuals:
##    Min     1Q Median     3Q    Max
## -5.260 -2.489 -0.683  1.929  6.463
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  block  (Intercept) 0.0212   0.1456
## Number of obs: 24, groups:  block, 4
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.42026    0.09107  48.539   <2e-16 ***
## trmt2        0.06343    0.07591   0.836    0.403
## trmt3       -1.17271    0.11214 -10.458   <2e-16 ***
## trmt4       -1.09861    0.10904 -10.075   <2e-16 ***
## trmt5       -0.84729    0.09954  -8.512   <2e-16 ***
## trmt6       -1.73957    0.14106 -12.332   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) trmt2  trmt3  trmt4  trmt5
## trmt2 -0.430
## trmt3 -0.291  0.349
## trmt4 -0.299  0.359  0.243
## trmt5 -0.328  0.393  0.266  0.274
## trmt6 -0.231  0.278  0.188  0.193  0.212
```

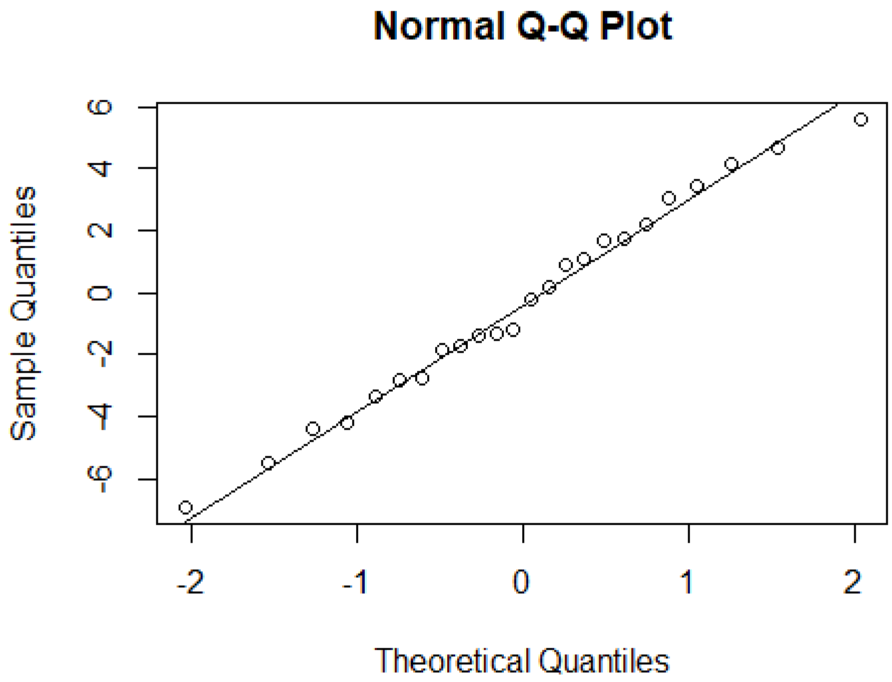
Lots of information, let’s review together. But remember, just because we changed the distribution doesn’t mean we got it right. We still need to check the residuals

Testing Assumptions

```
# Shapiro-Wilk normality test
shapiro.test(residuals(model3b))

##
##  Shapiro-Wilk normality test
##
## data:  residuals(model3b)
## W = 0.98481, p-value = 0.9656

# Homogeneity plots
par(mfrow = c(1, 1))
plot(model3b)
```



```
# Normality plots
qqnorm(residuals(model3b))
qqline(residuals(model3b))
```



What do you think? Better or worse?

Can you get a sense as to the subjectivity of this?

If we’re happy with this, lets take a look at the means comparisons.

```
model3b.emmean <- emmeans(model3b, "trmt", adjust="tukey", type="response")
model3b.emmean

##  trmt rate   SE  df asymp.LCL asymp.UCL
##  1    83.1 7.57 Inf      69.5      99.4
##  2    88.6 7.98 Inf      74.2     105.7
##  3    25.7 3.14 Inf      20.2      32.7
##  4    27.7 3.31 Inf      21.9      35.0
##  5    35.6 3.94 Inf      28.7      44.3
##  6    14.6 2.18 Inf      10.9      19.6
##
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale

pairs(model3b.emmean, type="response")

##  contrast ratio     SE  df z.ratio p.value
##  1 / 2    0.939 0.0712 Inf -0.836  0.9609
##  1 / 3    3.231 0.3623 Inf 10.458  <.0001
##  1 / 4    3.000 0.3271 Inf 10.075  <.0001
##  1 / 5    2.333 0.2323 Inf  8.512  <.0001
##  1 / 6    5.695 0.8034 Inf 12.332  <.0001
##  2 / 3    3.442 0.3832 Inf 11.104  <.0001
##  2 / 4    3.196 0.3458 Inf 10.740  <.0001
##  2 / 5    2.486 0.2452 Inf  9.235  <.0001
##  2 / 6    6.068 0.8520 Inf 12.840  <.0001
##  3 / 4    0.929 0.1264 Inf -0.545  0.9943
##  3 / 5    0.722 0.0929 Inf -2.530  0.1153
##  3 / 6    1.763 0.2871 Inf  3.480  0.0067
##  4 / 5    0.778 0.0979 Inf -1.996  0.3445
##  4 / 6    1.898 0.3052 Inf  3.987  0.0009
##  5 / 6    2.441 0.3770 Inf  5.776  <.0001
##
## P value adjustment: tukey method for comparing a family of 6 estimates
## Tests are performed on the log scale
```

```
multcomp::cld(model3b.emmean)
```

```
##  trmt rate   SE  df asymp.LCL asymp.UCL .group
##  6    14.6 2.18 Inf      10.9      19.6  1
##  3    25.7 3.14 Inf      20.2      32.7   2
##  4    27.7 3.31 Inf      21.9      35.0   2
##  5    35.6 3.94 Inf      28.7      44.3   2
##  1    83.1 7.57 Inf      69.5      99.4    3
##  2    88.6 7.98 Inf      74.2     105.7    3
##
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
## P value adjustment: tukey method for comparing a family of 6 estimates
## Tests are performed on the log scale
## significance level used: alpha = 0.05
```

Your turn!

If you happen to have some of your data with you - try this out. You should know what your exeprimental design is - from there create your statistical model - then trying running the analysis to partition your variance.

If you do not have your data with you, let me know and I can give you a practice RCBD dataset.

Workshop Review

ANOVA - History and updates

Fixed Effects ANOVA

Mixed Model ANOVA

GLMM