CH8: ANOVA for comparison of several means

1 Rice Example: One-way ANOVA

One-way ANOVA is used to compare means when there are more than two groups.

In this example, the effects of four acids on the growth of rice seedlings are compared in a completely randomized design. Seedling shoot dry weights are compared after 7 days in solution. The goal of the study is to compare mean weight for the four acid treatments.

```
library(tidyverse)
library(broom)
library(car)
library(emmeans)
library(dunn.test)
rice <- read.csv("C:/hess/STAT511_FA11/RData/CH8_Rice.csv")
str(rice)
```

```
## 'data.frame':    20 obs. of  2 variables:
##  $ trt   : Factor w/ 4 levels "acetic","butyric",..: 3 3 3 3 3 1 1 1 1 1 ...
##  $ weight: num  4.23 4.38 4.25 4.3 4.25 3.75 3.68 3.81 3.84 3.76 ...
```

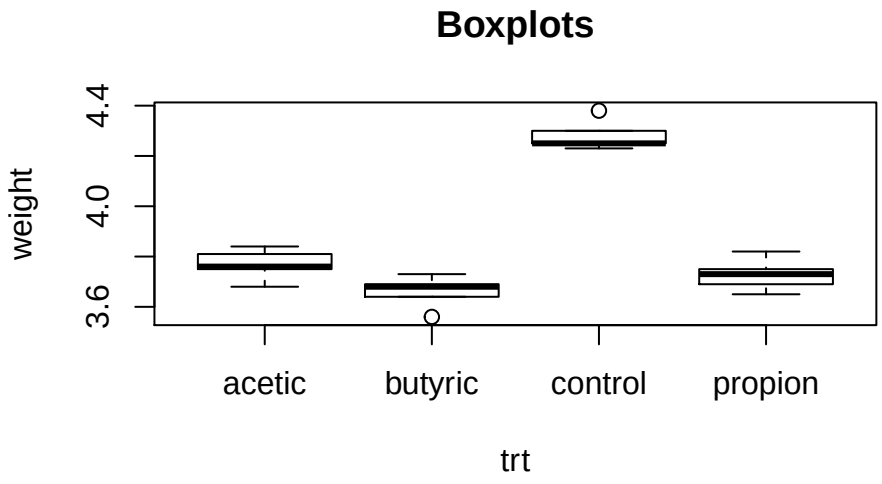
1.1 Summary Statistics and Graphics

Recall that the summarise() and group_by() functions are from the dplyr package.

```
SumStats <- summarise(group_by(rice, trt),
                      n = n(),
                      mean = mean(weight),
                      sd = sd(weight),
                      se = sd/sqrt(n))
SumStats
```

```
## # A tibble: 4 x 5
##   trt         n  mean     sd     se
##   <fct>   <int> <dbl>  <dbl>  <dbl>
## 1 acetic      5  3.77 0.0614 0.0275
## 2 butyric     5  3.66 0.0644 0.0288
## 3 control     5  4.28 0.0606 0.0271
## 4 propion     5  3.73 0.0642 0.0287
```

```
boxplot(weight ~ trt, data = rice, main = "Boxplots")
```



1.2 One-way ANOVA model and table

Note that trt should be defined as.factor. We checked this above using the str() function. We can use either lm() or aov() to fit the one-way ANOVA model. The resulting ANOVA table will be the same. The summary() output for the lm object is not directly needed to address common research questions, but shown here for illustration.

```
#Using lm
LMFit <- lm(weight ~ trt, data = rice)
anova(LMFit)
```

```
## Analysis of Variance Table
##
## Response: weight
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## trt        3 1.21985 0.40662  103.53 1.083e-10 ***
## Residuals 16 0.06284 0.00393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
LMFit
```

```
##
## Call:
## lm(formula = weight ~ trt, data = rice)
##
## Coefficients:
## (Intercept)   trtbutyric   trtcontrol   trtpropion
##       3.768       -0.108        0.514       -0.040
```

```
summary(LMFit)
```

```
##
## Call:
## lm(formula = weight ~ trt, data = rice)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.1000 -0.0335 -0.0030  0.0330  0.0980
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.76800    0.02803 134.443  < 2e-16 ***
## trtbutyric  -0.10800    0.03964  -2.725    0.015 *
## trtcontrol   0.51400    0.03964  12.968 6.63e-10 ***
## trtpropion  -0.04000    0.03964  -1.009    0.328
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06267 on 16 degrees of freedom
## Multiple R-squared:  0.951,  Adjusted R-squared:  0.9418
## F-statistic: 103.5 on 3 and 16 DF,  p-value: 1.083e-10
```

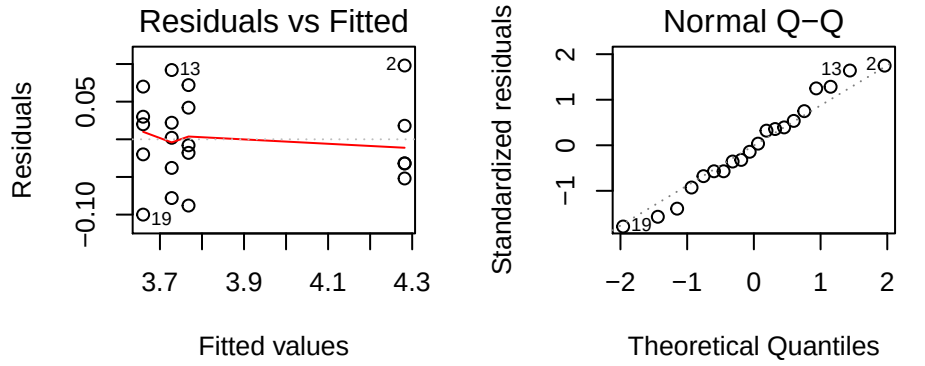
```
#Using aov
AovFit <- aov(weight ~ trt, data = rice)
summary(AovFit)
```

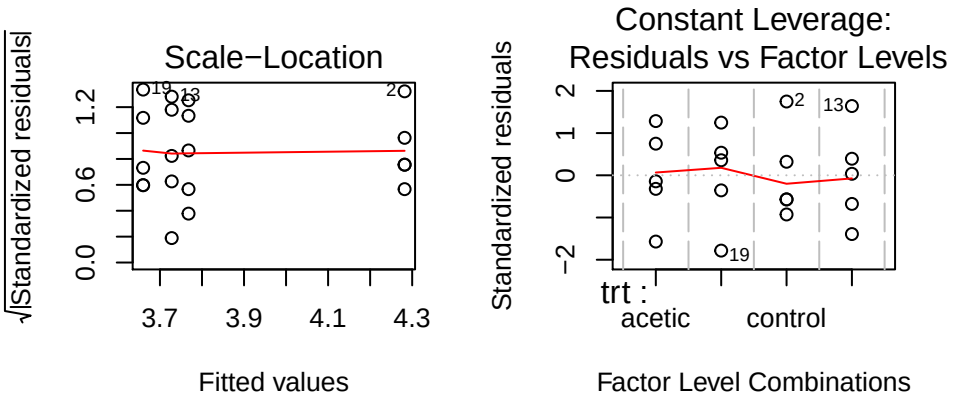
```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## trt          3 1.2199  0.4066   103.5 1.08e-10 ***
## Residuals   16 0.0628  0.0039
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.3 Diagnostics

When the plot() function is applied to an lm (or aov) object, diagnostic plots are returned. The plots of Resids vs Fitted and QQplot are of primary interest. We will not discuss the other two plots. Recall that leveneTest() is from the car package.

```
par(mfrow=c(2,2))
plot(LMFit)
```






```
shapiro.test(residuals(LMFit))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(LMFit)
## W = 0.97221, p-value = 0.8007
```

```
leveneTest(weight ~ trt, data = rice)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  3  0.0157 0.9972
##       16
```

1.4 Pairwise Comparisons

In this example, we show only unadjusted pairwise comparisons, but in practice Tukey adjustment is often used. See CH9 for a discussion of methods for multiple comparisons (includeing Tukey). The emmeans() function from the emmeans pacakge is very handy!

```
emout <- emmeans(LMFit, pairwise ~ trt, adjust = "none")
emout
```

```
## $emmeans
##  trt      emmean    SE df lower.CL upper.CL
##  acetic     3.77 0.028 16     3.71     3.83
##  butyric    3.66 0.028 16     3.60     3.72
##  control    4.28 0.028 16     4.22     4.34
##  propion    3.73 0.028 16     3.67     3.79
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast          estimate     SE df t.ratio p.value
##  acetic - butyric     0.108 0.0396 16   2.725 0.0150
##  acetic - control    -0.514 0.0396 16 -12.968 <.0001
##  acetic - propion     0.040 0.0396 16   1.009 0.3279
##  butyric - control   -0.622 0.0396 16 -15.693 <.0001
##  butyric - propion   -0.068 0.0396 16  -1.716 0.1055
##  control - propion    0.554 0.0396 16  13.977 <.0001
```

```
CLD(emout$emmeans, adjust = "none")
```

```
## Warning: 'CLD' will be deprecated. Its use is discouraged.
## See '? CLD' for an explanation. Use 'pwpp' or 'multcomp::cld' instead.
```

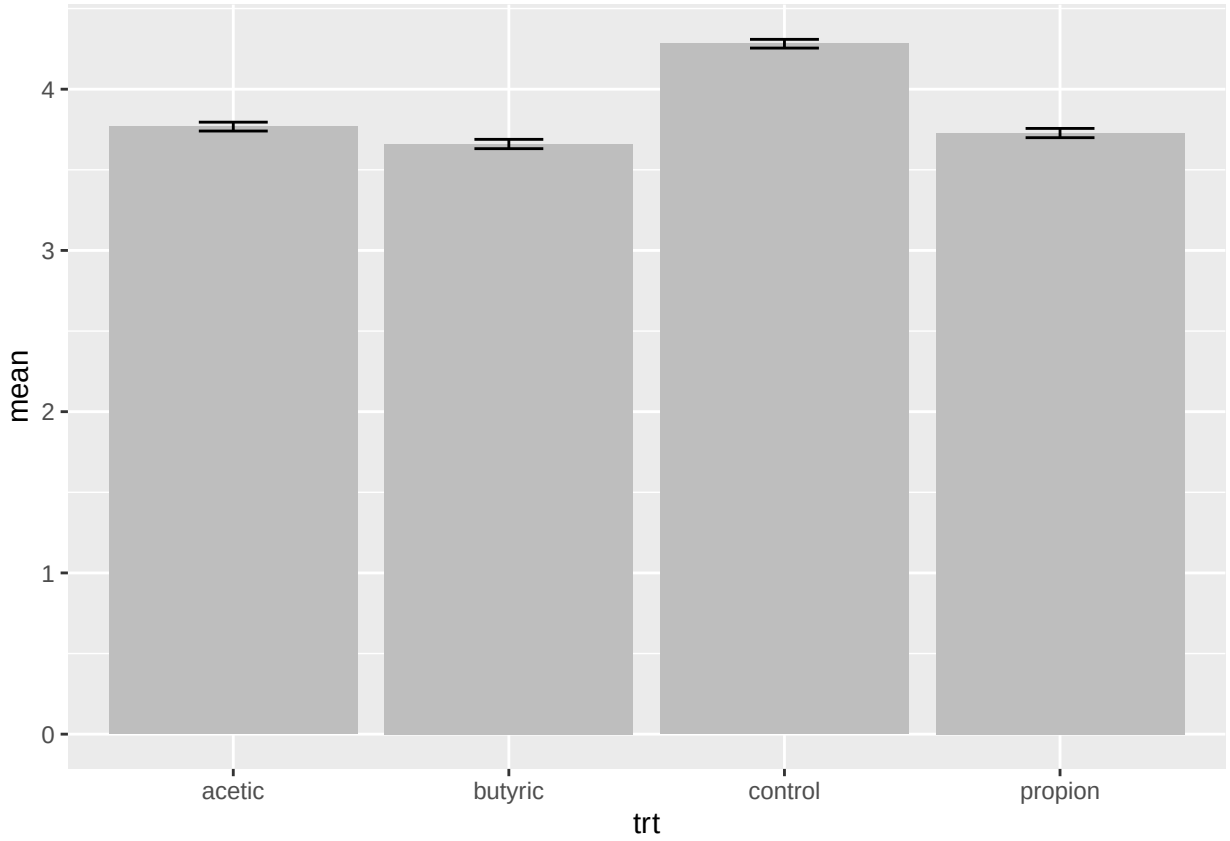
```
##  trt      emmean    SE df lower.CL upper.CL .group
##  butyric    3.66 0.028 16     3.60     3.72  1
##  propion    3.73 0.028 16     3.67     3.79  12
##  acetic     3.77 0.028 16     3.71     3.83   2
##  control    4.28 0.028 16     4.22     4.34    3
##
## Confidence level used: 0.95
## significance level used: alpha = 0.05
```

1.5 tidyverse

```
SumStats <- rice %>%
            group_by(trt) %>%
            summarise(n = n(),
                      mean = mean(weight),
                      sd = sd(weight),
                      se = sd/sqrt(n))
SumStats
```

```
## # A tibble: 4 x 5
##   trt         n  mean     sd     se
##   <fct>   <int> <dbl>  <dbl>  <dbl>
## 1 acetic      5  3.77 0.0614 0.0275
## 2 butyric     5  3.66 0.0644 0.0288
## 3 control     5  4.28 0.0606 0.0271
## 4 propion     5  3.73 0.0642 0.0287
```

```
ggplot(aes(x = trt, y = mean), data = SumStats) +
  geom_col(fill = "grey") +
  geom_errorbar(aes(ymin = mean-se, ymax = mean+se), width = .25)
```




```
tidy(anova(LMFit))
```

```
## # A tibble: 2 x 6
##   term         df  sumsq  meansq statistic   p.value
##   <chr>     <int>  <dbl>   <dbl>     <dbl>     <dbl>
## 1 trt           3 1.22   0.407        104.  1.08e-10
## 2 Residuals    16 0.0628 0.00393       NA  NA
```

```
emout <- emmeans(LMFit, ~ trt)
tidy(emout)
```

```
## # A tibble: 4 x 6
##   trt     estimate std.error    df conf.low conf.high
##   <fct>      <dbl>     <dbl> <dbl>    <dbl>     <dbl>
## 1 acetic      3.77    0.0280    16     3.71      3.83
## 2 butyric     3.66    0.0280    16     3.60      3.72
## 3 control     4.28    0.0280    16     4.22      4.34
## 4 propion     3.73    0.0280    16     3.67      3.79
```

```
tidy(pairs(emout, adjust = "none"))
```

```
## # A tibble: 6 x 7
##   level1  level2  estimate std.error    df statistic  p.value
##   <chr>   <chr>      <dbl>     <dbl> <dbl>     <dbl>    <dbl>
## 1 acetic  butyric   0.108     0.0396    16      2.72 1.50e- 2
## 2 acetic  control  -0.514     0.0396    16    -13.0  6.63e-10
## 3 acetic  propion   0.04      0.0396    16      1.01 3.28e- 1
## 4 butyric control  -0.622     0.0396    16    -15.7  3.87e-11
```

```
## 5 butyric propion  -0.0680    0.0396    16     -1.72 1.06e- 1
## 6 control propion   0.554     0.0396    16     14.0  2.19e-10
```

1.6 Kruskal-Wallis and Dunn's tests

Kruskal-Wallis is a non-parametric alternative to the one-way ANOVA that does not require the assumption of normality.

```
kruskal.test(weight ~ trt, data = rice)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  weight by trt
## Kruskal-Wallis chi-squared = 13.838, df = 3, p-value = 0.003135
```

```
dunn.test(x = rice$weight, g = rice$trt, method = "none")
```

```
##   Kruskal-Wallis rank sum test
##
## data: x and group
## Kruskal-Wallis chi-squared = 13.8377, df = 3, p-value = 0
##
##
##                            Comparison of x by group
##                                  (No adjustment)
## Col Mean-|
## Row Mean |     acetic    butyric    control
## ---------+---------------------------------
##  butyric |   1.740472
##          |     0.0409
##          |
##  control |  -1.874355  -3.614827
##          |     0.0304    0.0002*
##          |
##  propion |   0.669412  -1.071060   2.543767
##          |     0.2516     0.1421    0.0055*
##
## alpha = 0.05
## Reject Ho if p <= alpha/2
```

2 Poppies Example: Transformations for One-way ANOVA

Transforming data can be used to satisfy model assumptions (equal variance and normality).

In this example, five herbicides are compared in their ability to limit the number of poppy plants in oats. The five herbicide treatments are randomly assigned to twenty plots. The results, in number of poppy plants per 3.75 sqft of oats.

Analysis is done on the ORIGINAL scale, followed by an analysis in the SQRT and LOG scales. Residuals are plotted against predicted values to check the assumption of equality of variance.
Remember to check that Trt is defined as.factor using the str() function.

Box-Cox approach can be investigated using the boxcox() function from the MASS package.

```
library(tidyverse)
library(emmeans)
library(MASS)
poppies<-read.csv("C:/hess/STAT511_FA11/RData/CH8_Poppies.csv")
str(poppies)
```

```
## 'data.frame':    20 obs. of  2 variables:
##  $ Trt   : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ Plants: int  438 442 319 380 538 422 377 315 77 61 ...
```

```
SumStats <- summarise(group_by(poppies, Trt),
                      n = n(),
                      mean = mean(Plants),
                      sd = sd(Plants),
                      se = sd/sqrt(n))
SumStats
```

```
## # A tibble: 5 x 5
##   Trt       n  mean    sd    se
##   <fct> <int> <dbl> <dbl> <dbl>
## 1 A         4 395.   57.9  29.0
## 2 B         4 413    94.2  47.1
## 3 C         4  86.8  48.0  24.0
## 4 D         4  37.8  33.5  16.8
## 5 E         4  35.2  28.0  14.0
```

```
poppies$sqrtPlants <- sqrt(poppies$Plants)
poppies$logPlants  <- log(poppies$Plants)
str(poppies)
```

```
## 'data.frame':    20 obs. of  4 variables:
##  $ Trt       : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ Plants    : int  438 442 319 380 538 422 377 315 77 61 ...
##  $ sqrtPlants: num  20.9 21 17.9 19.5 23.2 ...
##  $ logPlants : num  6.08 6.09 5.77 5.94 6.29 ...
```

2.1 Analysis on the ORIGINAL scale

```
Fit1 <- lm(Plants ~ Trt, data = poppies)
anova(Fit1)
```

```
## Analysis of Variance Table
```

```
##
## Response: Plants
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Trt        4 597514  149378  45.451 3.271e-08 ***
## Residuals 15  49299    3287
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
emout1 <- emmeans(Fit1, pairwise ~ Trt, adjust = "none")
emout1$contrasts
```

```
##  contrast estimate   SE df t.ratio p.value
##  A - B       -18.2 40.5 15 -0.450  0.6590
##  A - C       308.0 40.5 15  7.598  <.0001
##  A - D       357.0 40.5 15  8.807  <.0001
##  A - E       359.5 40.5 15  8.868  <.0001
##  B - C       326.2 40.5 15  8.048  <.0001
##  B - D       375.2 40.5 15  9.257  <.0001
##  B - E       377.8 40.5 15  9.319  <.0001
##  C - D        49.0 40.5 15  1.209  0.2455
##  C - E        51.5 40.5 15  1.270  0.2233
##  D - E         2.5 40.5 15  0.062  0.9516
```

2.2 Analysis on the SQRT scale

```
Fit2 <- lm(sqrtPlants ~ Trt, data = poppies)
anova(Fit2)
```

```
## Analysis of Variance Table
##
## Response: sqrtPlants
##           Df Sum Sq Mean Sq F value    Pr(>F)
## Trt        4 866.96 216.739  45.527 3.233e-08 ***
## Residuals 15  71.41   4.761
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
emout2 <- emmeans(Fit2, pairwise ~ Trt, adjust = "none")
emout2$contrasts
```

```
##  contrast estimate   SE df t.ratio p.value
##  A - B      -0.399 1.54 15 -0.259  0.7995
##  A - C      10.745 1.54 15  6.964  <.0001
##  A - D      14.072 1.54 15  9.121  <.0001
##  A - E      14.179 1.54 15  9.191  <.0001
##  B - C      11.144 1.54 15  7.223  <.0001
##  B - D      14.471 1.54 15  9.380  <.0001
##  B - E      14.578 1.54 15  9.449  <.0001
##  C - D       3.327 1.54 15  2.156  0.0477
##  C - E       3.434 1.54 15  2.226  0.0418
##  D - E       0.107 1.54 15  0.070  0.9454
```

2.3 Analysis on the LOG (natural log) scale

```
Fit3 <- lm(logPlants ~ Trt, data = poppies)
anova(Fit3)
```

```
## Analysis of Variance Table
##
## Response: logPlants
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## Trt        4 27.6283  6.9071  25.173 1.669e-06 ***
## Residuals 15  4.1157  0.2744
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

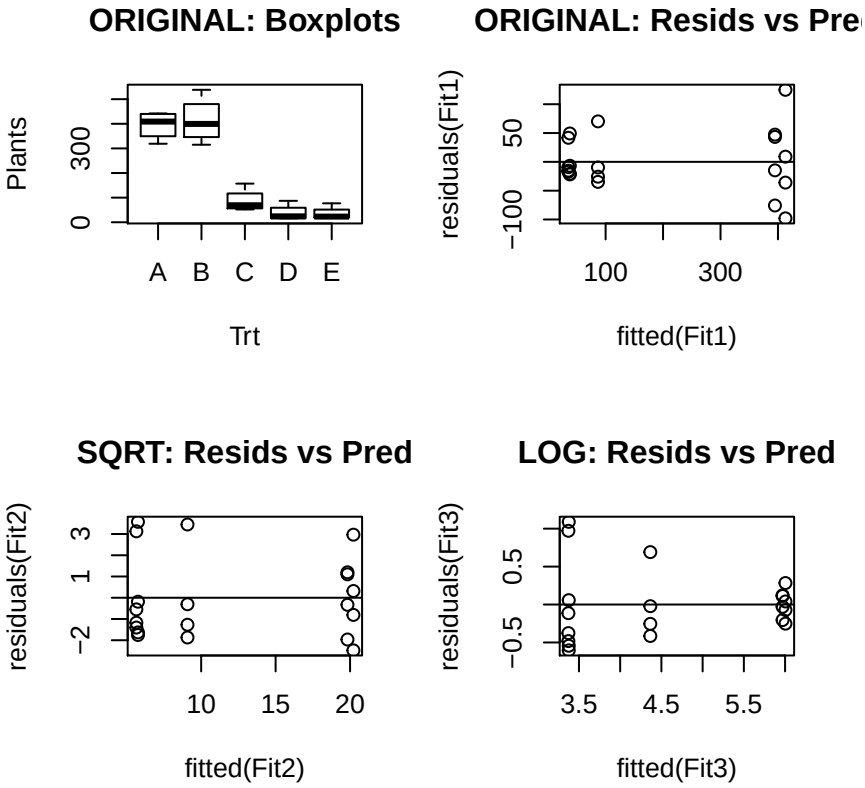
```
emout3 <- emmeans(Fit3, pairwise ~ Trt, adjust = "none")
emout3$contrasts
```

```
##  contrast estimate   SE df t.ratio p.value
##  A - B    -0.03470 0.37 15 -0.094  0.9266
##  A - C     1.60418 0.37 15  4.331  0.0006
##  A - D     2.59330 0.37 15  7.001  <.0001
##  A - E     2.59772 0.37 15  7.013  <.0001
##  B - C     1.63888 0.37 15  4.425  0.0005
##  B - D     2.62800 0.37 15  7.095  <.0001
##  B - E     2.63242 0.37 15  7.107  <.0001
##  C - D     0.98912 0.37 15  2.670  0.0175
##  C - E     0.99354 0.37 15  2.682  0.0170
##  D - E     0.00442 0.37 15  0.012  0.9906
```

2.4 Plots

The plot() function could also have been used to generate diagnostic plots for each analysis. Based on the diagnostic plots, the square root transform looks reasonable.

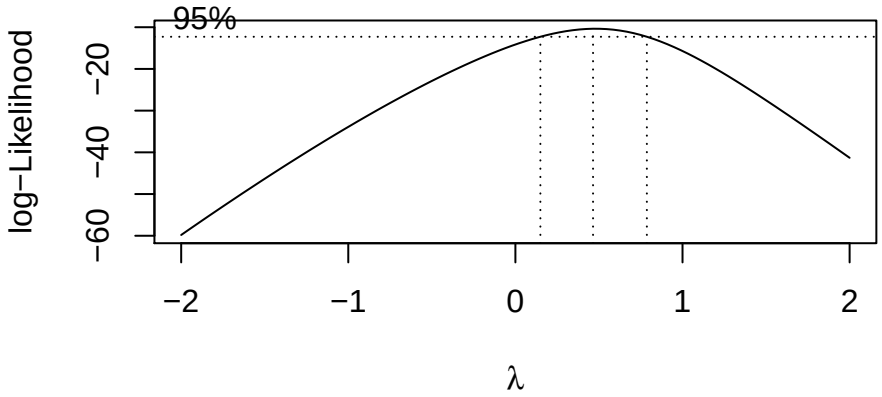
```
par(mfrow=c(2,2))
boxplot(Plants ~ Trt, data = poppies, main = "ORIGINAL: Boxplots")
plot(residuals(Fit1) ~ fitted(Fit1), main = "ORIGINAL: Resids vs Pred"); abline(h = 0)
plot(residuals(Fit2) ~ fitted(Fit2), main = "SQRT: Resids vs Pred"); abline(h = 0)
plot(residuals(Fit3) ~ fitted(Fit3),main="LOG: Resids vs Pred"); abline(h = 0)
```



2.5 Box-Cox

The boxcox() function is from the MASS package. For this example, $\lambda = 0.5$ is suggested by the Box-Cox summary graph. This supports the use of the square root transformation.

```
boxcox(Plants ~ Trt, data = poppies)
```



3 Power for One-way ANOVA

power.anova.test() works for balanced scenarios, where the sample size is the same for each group. Notice that the between and within group VARIANCES are needed for this function!

For this example with t = 4 groups, we conjecture that:
1. The true means are 1,2,3,4.
2. The true variance (within group) is 4.

3.1 Calculate the power with n=10 per group

```
var(c(1,2,3,4))
```

```
## [1] 1.666667
```

```
power.anova.test(groups = 4, n = 10,
                 between.var = 1.66667, within.var = 4,
                 sig.level = 0.05)
```

```
##
##      Balanced one-way analysis of variance power calculation
##
##          groups = 4
##               n = 10
##     between.var = 1.66667
##      within.var = 4
##       sig.level = 0.05
##           power = 0.8119587
##
## NOTE: n is number in each group
```

3.2 Calculate the sample size required to achieve 90% power

Notice this calculation returns a non-integer value - round up to an integer!

```
power.anova.test(groups = 4, power=0.90,
                 between.var = 1.66667, within.var = 4,
                 sig.level = 0.05)
```

```
##
##      Balanced one-way analysis of variance power calculation
##
##          groups = 4
##               n = 12.36347
##     between.var = 1.66667
##      within.var = 4
##       sig.level = 0.05
##           power = 0.9
##
## NOTE: n is number in each group
```