Day 4: One-Way and Two-Way ANOVA using R

Rey R. Cuenca[1]

December 1, 2021

[1] MSU-Iligan Institute of Technology, rey.cuenca@g.msuiit.edu.ph

Contents

Topics Covered

- One-Way ANOVA
  - Data Entry and Data Manipulation
  - Hypothesis Testing
  - Checking Assumptions
- Two-Way ANOVA
  - Data Entry and Data Manipulation
  - Hypothesis Testing
  - Checking Assumptions

Chapter 1

Preliminaries

1.1 Setting Up RStudio

In order for us to be on the same page all throughout the discussion, set up RStudio as explained in the following video.

1.2 Installing the needed R packages

```r
install.packages(c("tidyverse","ggpubr","rstatix","markdown","rmarkdown","tinytex"))
```

One-Way ANOVA

Chapter 2

Data Entry

Load the necessary packages.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggpubr)
library(multcomp)
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: survival
```

```
## Loading required package: TH.data
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
##
## Attaching package: 'TH.data'
## The following object is masked from 'package:MASS':
##
##     geyser
```

Load data to R using `read_csv()` function of the `readr` package of `tidyverse` and save it with a variable name `oneway_data`.

```
# Load and save
oneway_data <- read_csv(file = "data/Tubo-USEP_One-Way Cleaned Data for R.csv")
# Preview
oneway_data
```

```
## # A tibble: 5 x 5
##   Observation Colorless  Pink Orange Green
##         <dbl>     <dbl> <dbl>  <dbl> <dbl>
## 1           1      26.5  31.2   27.9  30.8
## 2           2      28.7  28.3   25.1  29.6
## 3           3      25.1  30.8   28.5  32.4
## 4           4      29.1  27.9   24.2  31.7
## 5           5      27.2  29.6   26.5  32.8
```

Chapter 3

Data Manipulation

```
oneway_data %>%
  gather(key = "Type", value = "Sales", -Observation) %>%
  # mutate(across(c(Observation,Type), ~as_factor(.x)))
  mutate(across(Observation:Type, ~as_factor(.x))) -> clean_oneway_data
clean_oneway_data
```

```
## # A tibble: 20 x 3
##    Observation Type      Sales
##    <fct>       <fct>     <dbl>
##  1 1           Colorless  26.5
##  2 2           Colorless  28.7
##  3 3           Colorless  25.1
##  4 4           Colorless  29.1
##  5 5           Colorless  27.2
##  6 1           Pink       31.2
##  7 2           Pink       28.3
##  8 3           Pink       30.8
##  9 4           Pink       27.9
## 10 5           Pink       29.6
## 11 1           Orange     27.9
## 12 2           Orange     25.1
## 13 3           Orange     28.5
## 14 4           Orange     24.2
## 15 5           Orange     26.5
## 16 1           Green      30.8
## 17 2           Green      29.6
## 18 3           Green      32.4
## 19 4           Green      31.7
## 20 5           Green      32.8
```

```
str(clean_oneway_data)
```

```
## tibble [20 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Observation: Factor w/ 5 levels "1","2","3","4",..: 1 2 3 4 5 1 2 3 4 5 ...
##  $ Type       : Factor w/ 4 levels "Colorless","Pink",..: 1 1 1 1 1 2 2 2 2 2 ...
##  $ Sales      : num [1:20] 26.5 28.7 25.1 29.1 27.2 31.2 28.3 30.8 27.9 29.6 ...
```

```
clean_oneway_data %>%
  sample_n(10)
```

```
## # A tibble: 10 x 3
##    Observation Type      Sales
##    <fct>       <fct>     <dbl>
##  1 2           Colorless  28.7
##  2 4           Pink       27.9
##  3 1           Green      30.8
##  4 4           Green      31.7
##  5 5           Orange     26.5
##  6 5           Pink       29.6
##  7 4           Colorless  29.1
##  8 2           Green      29.6
##  9 1           Orange     27.9
## 10 3           Green      32.4
```

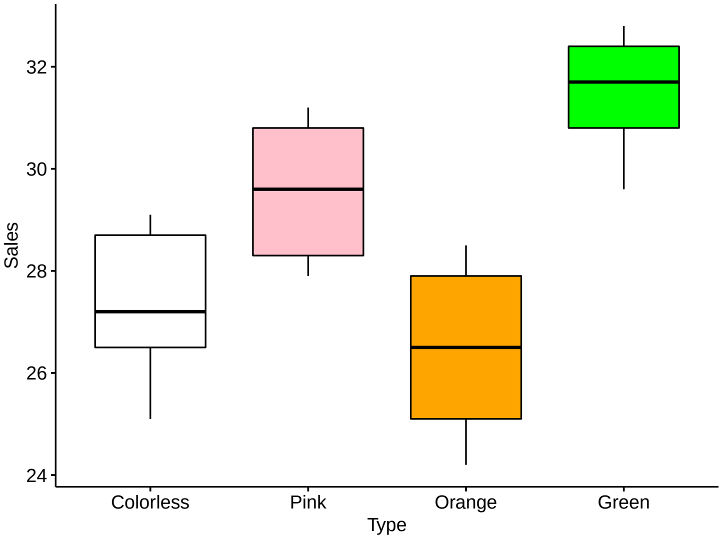
```
levels(clean_oneway_data$Type)
```

```
## [1] "Colorless" "Pink"      "Orange"    "Green"
```

Chapter 4

Data Visualization

```
clean_oneway_data %>%
  ggboxplot(x = "Type", y = 'Sales',
            fill = "Type",
            palette = c("white", "pink", "orange", "green")) +
  theme(legend.position = "none")
```



Chapter 5

Hypothesis Testing

The One-Way ANOVA Table in R

```
one_aov <- clean_oneway_data %>%
  aov(formula = Sales ~ Type, data = .)

one_aov %>%
  summary
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)    
## Type         3  76.85  25.615   10.49 0.000466 ***
## Residuals   16  39.08   2.443                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the table,

- `Df` – degress of freedom
- `Sum Sq` – sum of squares
- `Mean Sq` – mean sum of squares
- `F value` – value of F statistic
- `Pr(>F)` – p-value

Thus, from the table

$$\begin{align} SSB &= 76.85 & MSB &= 25.615 & F &= 10.49 \tag{5.1} \\ SSE &= 39.08 & MSE &= 2.443 & & \tag{5.2} \\ & & & & & \tag{5.3} \end{align}$$

Similar to when you look up at an F-table, the p-value can be computed using the following R code.

```r
pf(q = 10.49, df1 = 3, df2 = 16, lower.tail = F)
```

```
## [1] 0.0004652698
```

Chapter 6

Checking Assumptions[1]

6.1 Checking Normality Assumptions

Shapiro-Wilk Test

The Shapiro-Wilk test tests the null hypothesis that the samples come from a normal distribution against the alternative hypothesis that the samples do not come from a normal distribution.

```
oneway_data[-1,] %>%
  rstatix::shapiro_test(Colorless,Pink,Orange,Green)
```

```
## # A tibble: 4 x 3
##   variable  statistic     p
##   <chr>         <dbl> <dbl>
## 1 Colorless     0.913 0.499
## 2 Green         0.881 0.342
## 3 Orange        0.965 0.813
## 4 Pink          0.937 0.635
```

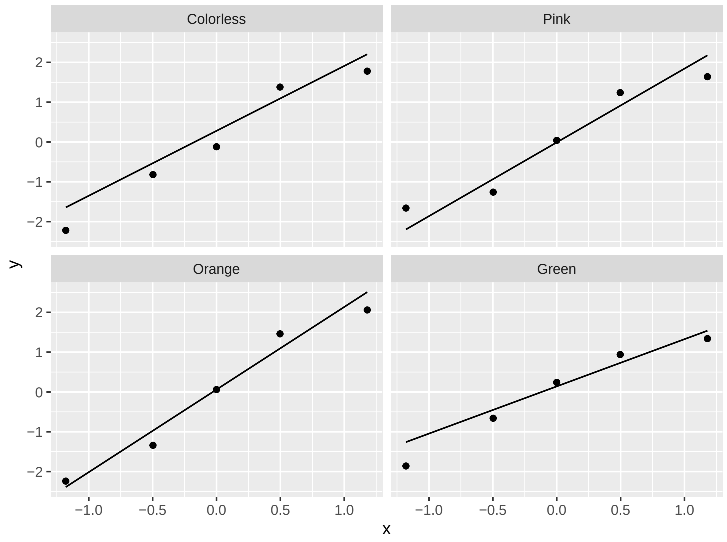
```
shapiro.test(residuals(object = one_aov))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(object = one_aov)
## W = 0.92472, p-value = 0.1222
```

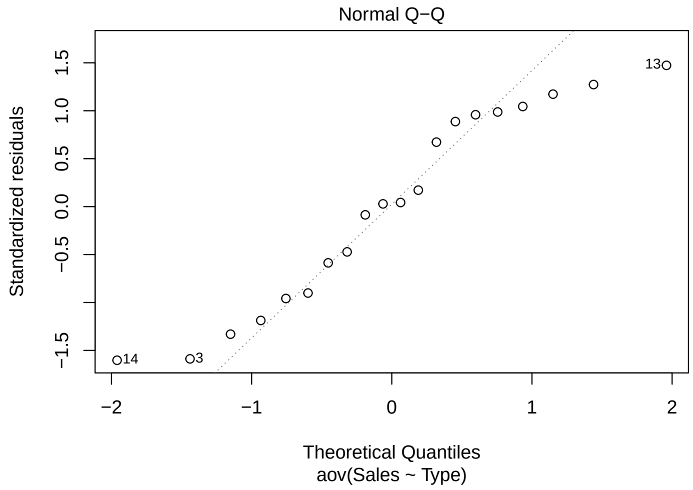
QQ Plots

[1] Except for most of the codes, the contents of this section are obtained from this link

```
clean_oneway_data %>%
  mutate(Residual = one_aov$residuals) %>%
  ggplot(aes(sample = Residual)) +
  stat_qq() +
  stat_qq_line() +
  facet_wrap(~Type)
```

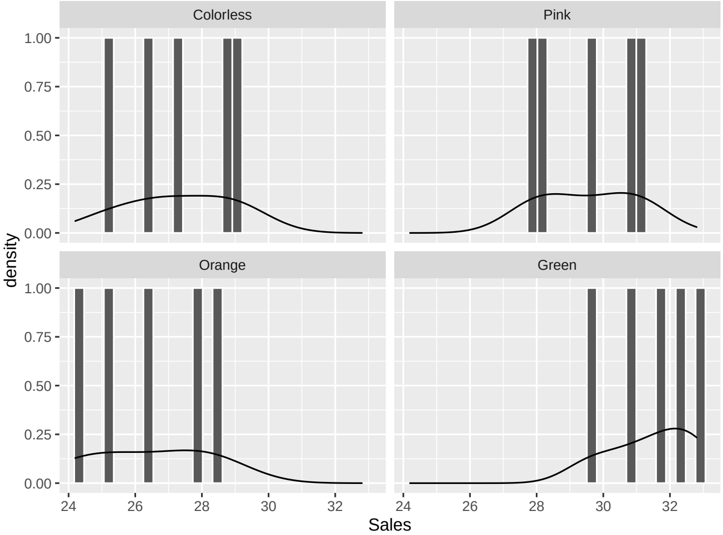


```
plot(one_aov,2)
```



Histogram

```
clean_oneway_data %>%
  ggplot(aes(x = Sales)) +
  geom_histogram(bins = 30, color = "white") +
  geom_density() +
  facet_wrap(~Type)
```



6.2 Checking Homogeneity of Variance Assumption

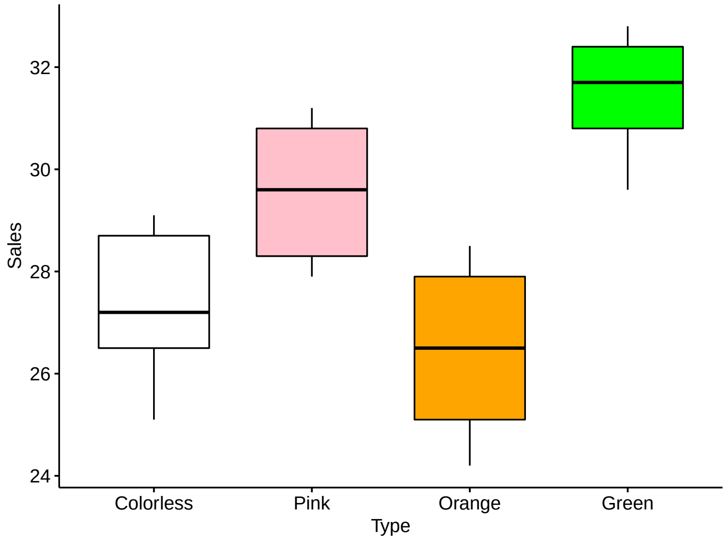
Bartlet's Test

Bartlett's test tests the null hypothesis that the group variances are equal against the alternative hypothesis that the group variances are not equal.

```
clean_oneway_data %>%
  bartlett.test(Sales ~ Type, data = .)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  Sales by Type
## Bartlett's K-squared = 0.46564, df = 3, p-value = 0.9264
```

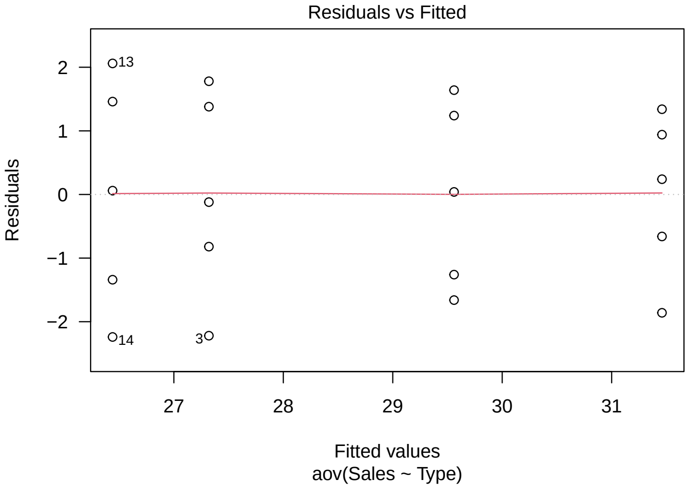
```
clean_oneway_data %>%
  ggboxplot(x = "Type", y = 'Sales',
            fill = "Type",
            palette = c("white", "pink", "orange", "green")) +
  theme(legend.position = "none")
```



The variability within each group is represented by the vertical size of each box; i.e., the interquartile range (IQR). The boxplot shows that the variability is roughly equal for each group. Let's look at some more ways to test the homogeneity of variance assumption.

Residual vs. Fitted Values Plot

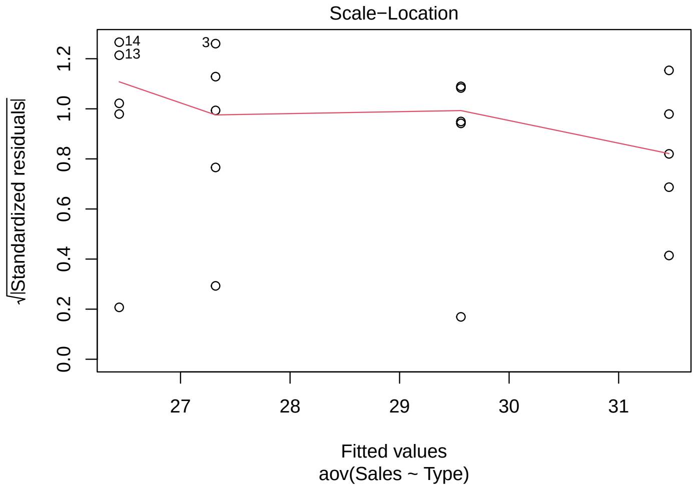
```
plot(one_aov,1, las=1)
```



This plot shows the residuals (errors) on the y-axis and the fitted values (predicted values) on the x-axis. If the variance of each group is equal, the plot should show no pattern; in other words, the points should look like a cloud of random points. The plot shows that the variances are approximately homogenous since the residuals are distributed approximately equally above and below zero.

Standardised Residuals vs Fitted values Plot

```
plot(one_aov,3)
```



The more coincident the red line plot to the horizontal line at 1, the lesser possibility the violation of the homogeneity of variance assumption.

Chapter 7

Post-hoc

7.1 TukeyHSD (Tukey's Honestly-Significant Difference) post-hoc test in R

There are at least two ways to perform a Tukey's HSD post-hoc in R. One is by using the `TukeyHSD` function of the pre-installed R package `stats`. The second is the `glht` function with `"Tukey"` option bundled along with the `multcomp` package.

7.2 Using `TukeyHSD` of `stats` package

```
TukeyHSD(one_aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Sales ~ Type, data = .)
##
## $Type
##                    diff        lwr        upr     p adj
## Pink-Colorless    2.24 -0.5880714  5.0680714 0.1479369
## Orange-Colorless -0.88 -3.7080714  1.9480714 0.8099459
## Green-Colorless   4.14  1.3119286  6.9680714 0.0034923
## Orange-Pink      -3.12 -5.9480714 -0.2919286 0.0281177
## Green-Pink        1.90 -0.9280714  4.7280714 0.2580535
## Green-Orange      5.02  2.1919286  7.8480714 0.0005837
```

Discussion of Results. Picking up from the significant ANOVA result in our soft drink data, the Tukey's HSD post-hoc analysis result above shows the following significant comparisons at 0.05:

```
cat("Avg. Sales Comparison\t P-value (adjusted)\n------------------------------------
```

```
## Avg. Sales Comparison     P-value (adjusted)
## ---------------------------------------------
## Green > Colorless     0.0034923
## Green > Orange        0.0005837
## Orange > Pink         0.0281177
## ---------------------------------------------
```

7.3 Using the multicomp package with "Tukey" option

```
summary(glht(one_aov, linfct = mcp(Type = "Tukey")))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = Sales ~ Type, data = .)
##
## Linear Hypotheses:
##                         Estimate Std. Error t value Pr(>|t|)
## Pink - Colorless == 0     2.2400     0.9885   2.266  0.14815
## Orange - Colorless == 0  -0.8800     0.9885  -0.890  0.80991
## Green - Colorless == 0    4.1400     0.9885   4.188  0.00339 **
## Orange - Pink == 0       -3.1200     0.9885  -3.156  0.02817 *
## Green - Pink == 0         1.9000     0.9885   1.922  0.25800
## Green - Orange == 0       5.0200     0.9885   5.078  < 0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Two-Way ANOVA

Chapter 8

Data Entry[1]

Load data to R using `read_csv()` function of the `readr` package of `tidyverse` and save it with a variable name `twoway_data`.

```
# Load and save
twoway_data <- read_csv(file = "data/Tubo-USEP_Two-Way Cleaned Data for R.csv")
# Preview
twoway_data
```

```
## # A tibble: 4 x 7
##   Fertilizer Manure    P1    P2    P3    P4    P5
##   <chr>      <chr>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 High       High    13.7  15.8  13.9  16.6  15.5
## 2 High       Low     16.4  12.5  14.1  14.4  12.2
## 3 Low        High    15    15.1  12    15.7  12.2
## 4 Low        Low     12.4  10.6  13.7   8.7  10.9
```

[1] The contents of the succeeding sections are obtained from this link

Chapter 9

Data Manipulation

```
twoway_data%>%
  gather(key = Plot, value = Yield, -c(Fertilizer,Manure)) %>%
  mutate(across(Fertilizer:Plot, ~ as.factor(.x))) -> clean_twoway_data
```

```
# Structure preview
str(clean_twoway_data)
```

```
## tibble [20 x 4] (S3: tbl_df/tbl/data.frame)
##  $ Fertilizer: Factor w/ 2 levels "High","Low": 1 1 2 2 1 1 2 2 1 1 ...
##  $ Manure    : Factor w/ 2 levels "High","Low": 1 2 1 2 1 2 1 2 1 2 ...
##  $ Plot      : Factor w/ 5 levels "P1","P2","P3",..: 1 1 1 1 2 2 2 2 3 3 ...
##  $ Yield     : num [1:20] 13.7 16.4 15 12.4 15.8 12.5 15.1 10.6 13.9 14.1 ...
```

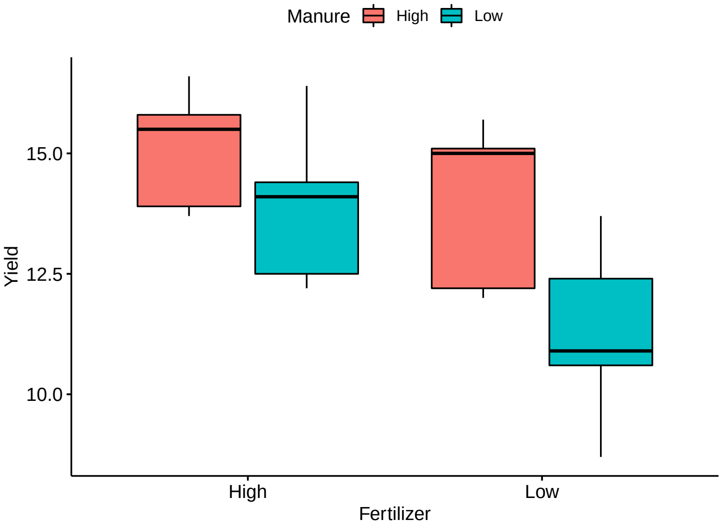
```
# Sample preview
clean_twoway_data %>%
  sample_n(10)
```

```
## # A tibble: 10 x 4
##    Fertilizer Manure Plot  Yield
##    <fct>      <fct>  <fct> <dbl>
##  1 High       High   P3     13.9
##  2 Low        Low    P5     10.9
##  3 High       High   P2     15.8
##  4 High       Low    P4     14.4
##  5 High       High   P5     15.5
##  6 High       Low    P2     12.5
##  7 Low        High   P3     12
##  8 High       High   P1     13.7
##  9 High       Low    P3     14.1
## 10 Low        Low    P3     13.7
```

Chapter 10

Data Visualization

```
ggboxplot(clean_twoway_data,
          x = "Fertilizer",
          y = "Yield",
          fill = "Manure")
```



Chapter 11

Hypothesis Testing

11.1 The Two-Way ANOVA Table with Main Effects Only

```
two_aov <- clean_twoway_data %>%
  aov(formula = Yield ~ Fertilizer + Manure, data = .)

two_aov %>%
  summary
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Fertilizer   1  17.67  17.672   6.332 0.0222 *
## Manure       1  19.21  19.208   6.883 0.0178 *
## Residuals   17  47.44   2.791
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Similar to the One-Way ANOVA table,

- `Df` – degress of freedom
- `Sum Sq` – sum of squares
- `Mean Sq` – mean sum of squares
- `F value` – value of F statistic
- `Pr(>F)` – p-value

Thus, from the table

$$
\begin{aligned}
SSR &= 17.67 & MSR &= 17.672 & F_C &= 6.332 \quad (11.1)\\
SSC &= 19.21 & MSC &= 19.208 & F_R &= 6.883 \quad (11.2)\\
SSE &= 47.44 & MSE &= 2.791 & & \quad (11.3)\\
& & & & & \quad (11.4)
\end{aligned}
$$

Similar to when you look up at an F-table, the p-values can be computed using the following R code.

```
pf(q = 6.332, df1 = 1, df2 = 17, lower.tail = F)
```

```
## [1] 0.02219209
```

```
pf(q = 6.883, df1 = 1, df2 = 17, lower.tail = F)
```

```
## [1] 0.01779112
```

11.2 The Two-Way ANOVA Table with Interactions

```
two_aov2 <- clean_twoway_data %>%
  aov(formula = Yield ~ Fertilizer*Manure, data = .)

two_aov2 %>%
  summary
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)  
## Fertilizer         1  17.67  17.672   6.368 0.0226 *
## Manure             1  19.21  19.208   6.922 0.0182 *
## Fertilizer:Manure  1   3.04   3.042   1.096 0.3107  
## Residuals         16  44.40   2.775                 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The interaction is not significant so we proceed on using the additive model (i.e., main-effects only).

Chapter 12

Checking Assumptions

12.1 Checking Normality Assumptions

Shapiro-Wilk Test

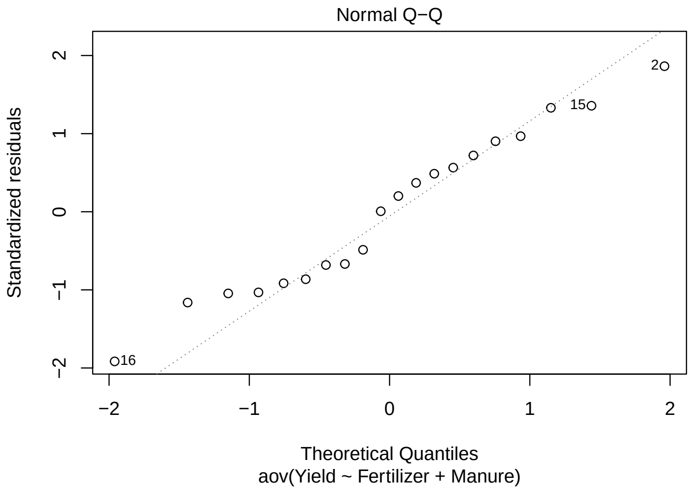
The Shapiro-Wilk test tests the null hypothesis that the samples come from a normal distribution against the alternative hypothesis that the samples do not come from a normal distribution.

```
shapiro.test(residuals(two_aov))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(two_aov)
## W = 0.9634, p-value = 0.6138
```

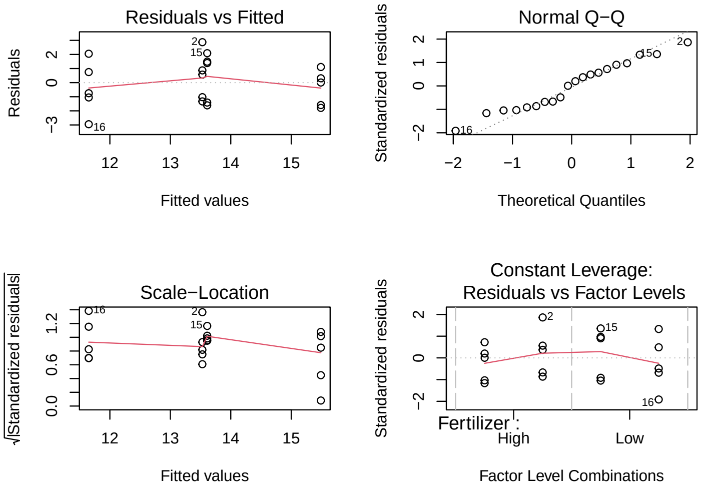
QQ Plots

```
plot(two_aov, 2)
```



12.2 Checking Homogeneity of Variance Assumption

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



```
par(mfrow=c(1,1))
```

Chapter 13

Post-hoc

We no longer need a posthoc analysis since we only have two levels in each of the significant factors. However, for the sake of illustration, we could be simply run the following code

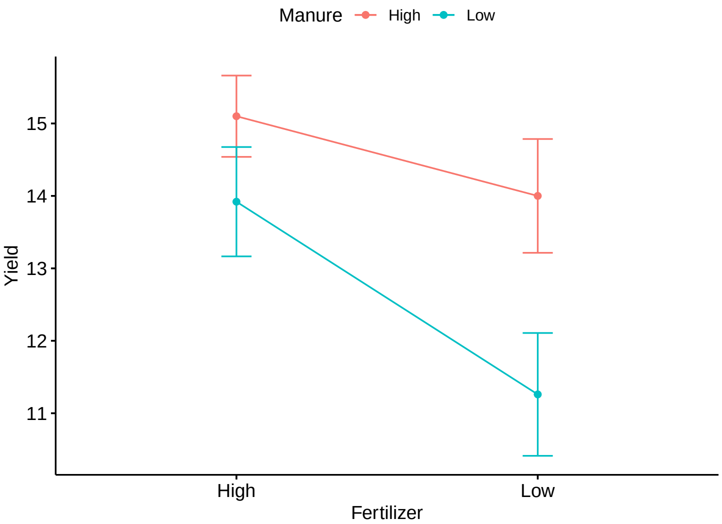
```
TukeyHSD(two_aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Yield ~ Fertilizer + Manure, data = .)
##
## $Fertilizer
##           diff       lwr        upr    p adj
## Low-High -1.88 -3.456219 -0.3037812 0.022188
##
## $Manure
##           diff       lwr        upr     p adj
## Low-High -1.96 -3.536219 -0.3837812 0.0177922
```

Chapter 14

Mean Line or Interaction Plots

```
ggline(clean_twoway_data,
       x = "Fertilizer",
       y = "Yield",
       color = "Manure",
       add = c("mean_se"))
```



The above figure supports the result before that there is no significant interaction between the two factors.