Parametric Shortcut: One-way ANOVA

Neil J. Hatfield

2/26/2021

In this tutorial, we are going to explore using R to fit a One-way ANOVA model to two data sets using the parametric shortcut known as the One-way ANOVA F Test.

Set Our Constraints

Recall that in order to ensure that we have estimable functions, we must set a side condition or constraint on our treatments:

$$\sum_{i}^{k} \alpha_i = 0$$

This is not what R does by default, but we can tell R to adopt this constraint with the following code:

```
## DEMO CODE
options(contrasts = c("contr.sum", "contr.poly"))
```

Load Data

Our first step will be to load data into our R session. We are going to use two data sets: the honey example from class and Example 3.2 Resin Lifetimes from the Oehlert textbook. For the honey data, we will create the data frame manually; for resin lifetimes, we'll import the data.

```
## DEMO CODE

# Honey Data
honey <- data.frame(
  Amount = c(150, 50, 100, 85, 90, 95, 130, 50, 80),
  Varietal = rep(c("Clover", "Orange Blossom", "Alfalfa"), each = 3)
)
## Set Varietal to factor
honey$Varietal <- as.factor(honey$Varietal)

# Resin Lifetimes Data
resin <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/resinLifetimes.dat",
  header = TRUE,
  sep = ""
)
## Set temp to factor
resin$temp <- as.factor(resin$temp)
```

Is ANOVA Even Appropriate?

Recall the base requirements for One-way ANOVA are:

- you are working with a categorized factor,
- you are working with an additive model,
- you have estimable effects, and
- you have estimable errors/residuals.

You can check these four requirements rather quickly. First, use the `str` function to ensure that R is thinking about the `Varietal` (or `temp`) as a factor. Second, examine your Hasse diagram and algebraic model–did you build upon the additivity of terms? (If you've followed the methods that we've been using, then yes.) Third, did you tell R to set the constraint? Look at the term nodes in the Hasse diagram. Do each of the terms you want to test have non-zero (and non-negative) *Degrees of Freedom*? (Yes, you're good; no, there are problems.) Fourth, look at the error term in your Hasse diagram. Do you have a positive (i.e., non-zero, non-negative) number of *Degrees of Freedom*? (Yes, you're good; no, there are problems.)

Notice the work that the Hasse diagram is doing for us. They not only help us visualize our model but they also set up our screens, and allow for us to check whether we meet the basic requirements for doing ANVOA. The Hasse Diagram App will through an error if you run into problems with your *Degrees of Freedom*.

Example 1-Honey Data

We will begin the Honey example from class. Before we look at any p-values and make any decisions, we need to ensure that the prerequisites and then assumptions are met.

Figure 1 shows the Hasse diagram for our Honey Study. We're interested in testing the type of varietal. Both that node and our error node have positive values for *Degrees of Freedom*. This taken with our study design, support our using One-way ANOVA methods.

```r
# DEMO CODE

# Hasse Diagram for the Honey Study
modelLabels <- c("1 Make Honey 1", "3 Type 2", "9 (Hives) 6")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, TRUE, FALSE),
  nrow = 3,
  ncol = 3,
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)
```

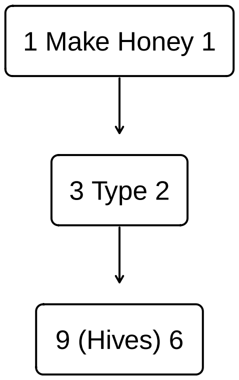


Figure 1: Hasse Diagram for Honey Study

Assessing Assumptions

To assess our assumptions, we first need to fit the model. This will provide us with the residuals that we need for checking the assumptions.

The following code demonstrates how we can fit the One-way ANOVA model:

```r
# DEMO CODE

# Fit our One-way ANOVA model to the honey data
honeyModel <- aov(
  formula = Amount ~ Varietal,
  data = honey,
  na.action = "na.omit"
)
```

We can now use `honeyModel$residuals` to access the residuals from the model. We can use them just like we would use any column of a data frame.

Assessing the Gaussian Assumption

The first assumption is that our residuals follow a Gaussian ("normal") distribution, A QQ plot is the tool of choice; one of the better versions of this plot comes from the `car` package.

```r
# DEMO CODE

# QQ plot for Honey Residuals
car::qqPlot(
  x = honeyModel$residuals,
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals"
)
```

The QQ Plot (Figure 2) for our residuals here is concerning. Nearly half of the residuals are beyond our 90% confidence envelope. The values of the *Sample Skewness* (0.11) and *Sample Excess Kurtosis* (-1.17) also suggest that a Gaussian distribution is not the greatest match for our residuals.

Use `psych::skew(honeyModel$residuals)` and `psych::kurtosi(honeyModel$residuals)` to get the values of *Sample Skewness* and *Sample Excess Kurtosis*, respectively.

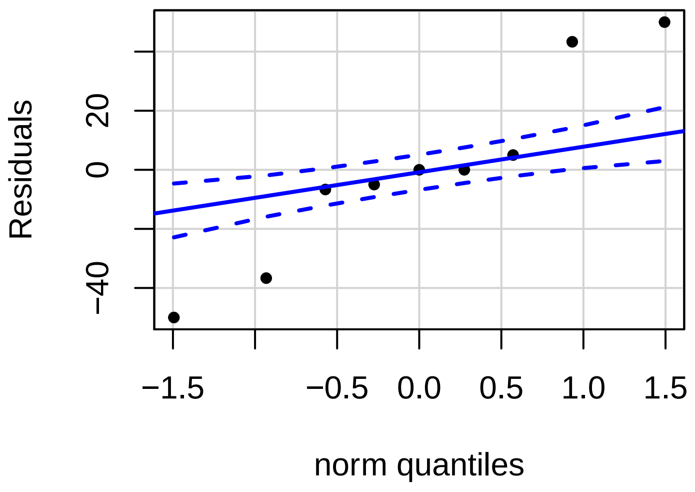


Figure 2: QQ Plot of Honey Residuals

Assessing Homoscedastcitiy

```
# DEMO CODE

# Strip chart for Honey Residuals
ggplot(
  data = data.frame(
    residuals = honeyModel$residuals,
    fitted = honeyModel$fitted.values
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 1) +
  theme_bw() +
  xlab("Fitted values (lbs)") +
  ylab("Residuals (lbs)")
```

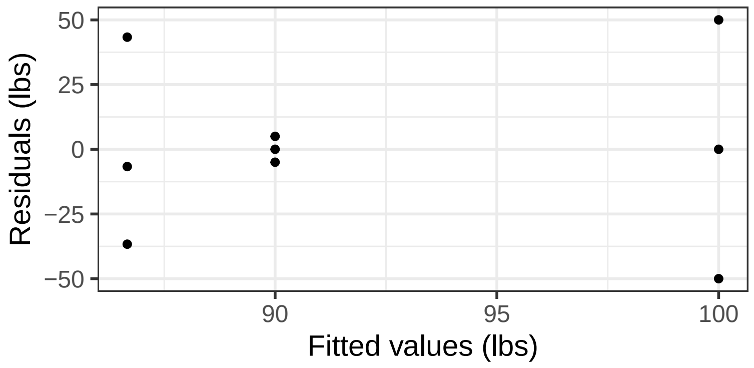


Figure 3: Stripchart for Honey Residuals

In Figure 3, we can see that one strip (located at a fitted value of 90) is rather close together. The vertical space that group uses is well under half the vertical space used by any other strip. This along with the essential bow-tie shape to the plot suggest that we do not have the same level of variation within each level of treatment.

Assessing Independence of Observations

In order to truly assess whether our observations are truly independent, we must know something about the order in which measurements were taken (or how they were arranged spatially).

For the honey data, we know that the order of the values reflects the measurement order. Thus, we can use an index plot.

```
# DEMO CODE

# Index Plot for Honey Residuals
ggplot(
  data = data.frame(
    residuals = honeyModel$residuals,
    index = 1:length(honeyModel$residuals)
  ),
  mapping = aes(x = index, y = residuals)
) +
  geom_point(size = 1.5) +
  geom_line() +
  theme_bw() +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "red"
  ) +
  xlab("Measurement order") +
  ylab("Residuals")
```

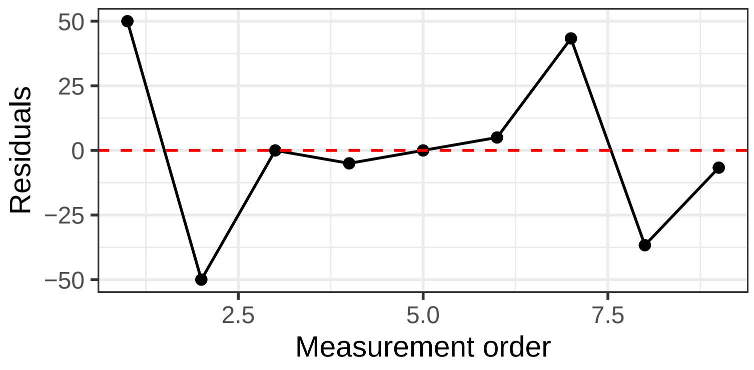


Figure 4: Index Plot for Honey Residuals

Given the order that measurements were made, there does not appear to be any strong indications of autocorrelation in Figure 4. If there was an issue, we would expect to see some patterning and/or clusters of values. Further, the Durbin-Watson statistic has a value of 2.57 which is fairly close to the 2.5 Rule of Thumb cutoff.

Use `car::durbinWatsonTest(honeyModel)$dw` to get the value of the Durbin-Watson statistic.

Conduct the Test

Since two of the assumptions are violated (and fairly badly), conducting the significance test is not recommended without first taking corrective actions. Thus, we will **NOT** use the parametric shortcut with the Honey Study data.

Example 2-Resin Lifetimes

For background on this example, please see Example 3.2 (page 32) in the Oehlert text.

The One-way ANOVA designs end up working well in this situation.

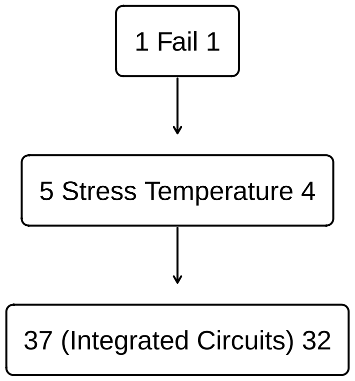


Figure 5: Hasse Diagram for Resin Study

Figure 5 shows that our factor has 5 (categorized) levels, an additive model, and that we have sufficient *Degrees of Freedom* to estimate all of our terms. Thus a One-way method is appropriate.

Assessing Assumptions

Again, we will need to first fit the One-way model to our data in order to assess the assumptions of the parametric shortcut.

We will now use `resinModel$residuals` to access the residuals from the model.

Assessing the Gaussian Assumption

We will turn to the QQ plot of the residuals to assess this assumption.

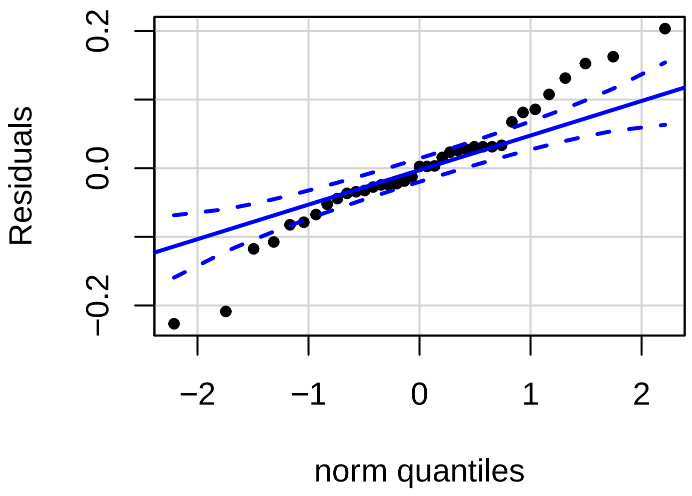


Figure 6: QQ Plot of Resin Lifetimes Residuals

We have several residuals falling outside of the 90% confidence envelope (see Figure 6); approximately 35.1% of the residuals are outside. This seems high. The values of the *Sample Skewness* (-0.15) and *Sample Excess Kurtosis* (0.45) are not too far off what we would want to see (i.e., 0). We might say that the Gaussian assumption is “Questionable” and see what the other assumptions show us.

Assessing Homoscedastcitiy

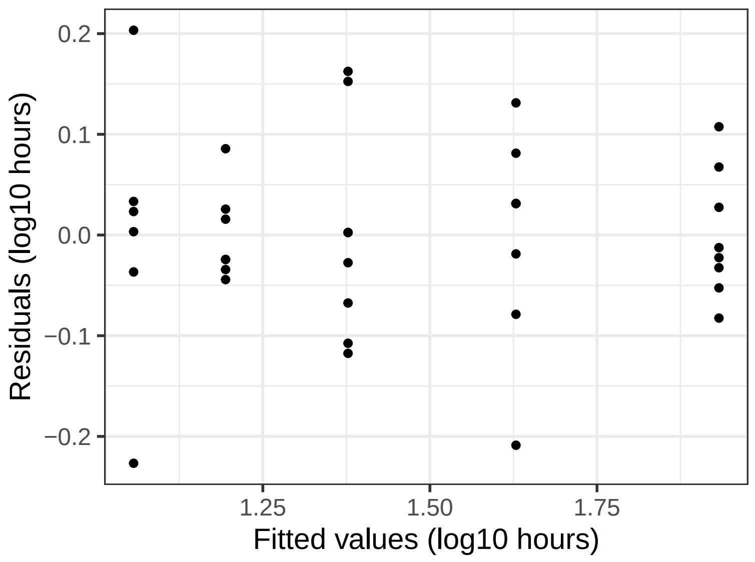


Figure 7: Stripchart for Resin Lifetimes Residuals

When examining Figure 7, the second vertical strip from the left is the strip to compare with the others as this uses the least amount of vertical space. Comparing this strip to the first and fourth strips (from the left), revels that these groups use just about twice the vertical space as our reference. This puts us just on the cusp of violation of the Homoscedasticity Assumption.

Assessing Independence of Observations

Supposing that our resin data appear in measurement order, we can look at an index plot for checking independence of observations.

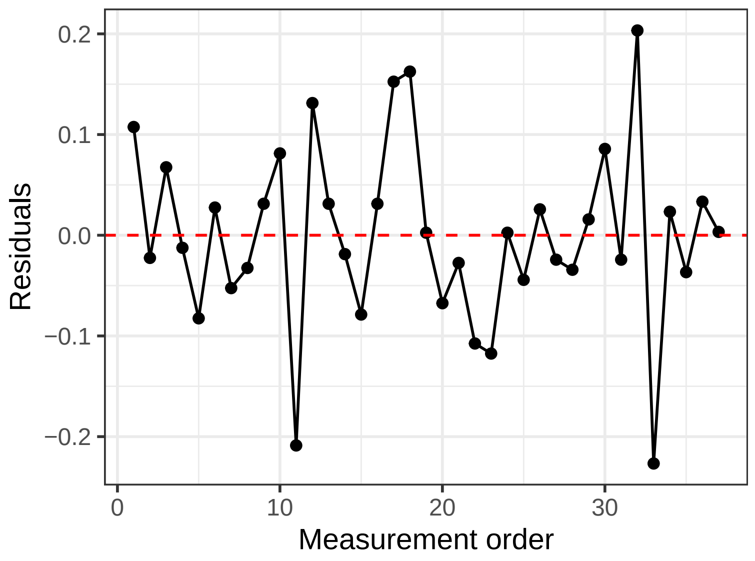


Figure 8: Index Plot for Resin Lifetimes Residuals

Figure 8 does not show any pattern that would suggest a violation of this assumption. Additionally, the Durbin-Watson statistic has a value of 2.37.

Conduct the Test

In this situation, we might decide to proceed with the parametric shortcut, albeit cautiously. Remember, the parametric shortcut is robust to moderate and minor violations of assumptions.

Before we go any further, we need to set our Unusualness Threshold (level of significance). Don't be boring and use $UT = 0.05$; think about how often something has to occur in order for you say "Hey, that's unusual." For me, the break between usual and unusual occurs around 3%; thus, I'm going to use $UT = 0.03$.

There are a couple of ways that you can see the results of the test. When we fit the model, we actually gave R everything necessary to do the test. We just now need to see the results.

Quick Look Method

The quick look method is great when you are just looking for yourself but TERRIBLE when you are going to write a report. You simply use the `summary` or `anova` functions to get the raw output as shown here:

```
# DEMO CODE
# Displaying the results of the Parametric Shortcut via summary
summary(resinModel)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)    
## temp         4  3.538  0.8844   96.36 <2e-16 ***
## Residuals   32  0.294  0.0092                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# DEMO CODE
# Displaying the results of the Parametric Shortcut via anova
anova(resinModel)
```

```
## Analysis of Variance Table
##
## Response: y
##           Df Sum Sq Mean Sq F value    Pr(>F)    
## temp       4 3.5376 0.88441  96.363 < 2.2e-16 ***
## Residuals 32 0.2937 0.00918                      
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The `anova` function is special wrapper of the `summary` function that adds just a bit more formatting to the output. However, we can do MUCH better.

Professional Looking ANOVA Tables

We are going to combine professional looking tables from the Descriptive Statistics guide along with the notion of Modern ANOVA tables so that we get both statistical and practical significance information.

To get the effect sizes, we will make use of the `parameters` package's `model_parameters` function.

```r
# DEMO CODE

# Create professional looking modern ANOVA table
parameters::model_parameters(
  model = resinModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Resin Lifetimes Study",
  format = "latex",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  ) %>%
  kableExtra::footnote(
    general = "Computer rounding has made the p-value look like zero.",
    general_title = "Note. ",
    footnote_as_chunk = TRUE
  )
```

Table 1: ANOVA Table for Resin Lifetimes Study

Source	SS	df	MS	F	p-value	Omega Sq.	Eta Sq.	Epslion Sq.
temp	3.5376	4	0.8844	96.363	0	0.9116	0.9233	0.9138
Residuals	0.2937	32	0.0092					

Note. Computer rounding has made the p-value look like zero.

Notice that Table 1 looks much more professional than the raw output we got from using `summary` and `anova`. Further, by using this approach with the `model_parameters` function, we've gotten our three estimates of effect size for the model.

There is a downside in this approach: the p-value has been made to look equal to zero. We know that while p-values may be essentially equal to zero, they are not actually zero. We added a footnote to Table 1. However, we could attempt to fix this issue using the following function:

```r
# Set up the p-value rounding function.
pvalRound <- function(x){
  if (x < 0.0001) {
    return("< 0.0001")
  } else {
    return(x)
  }
}
```

```r
# DEMO CODE

# Professional looking modern ANOVA table with fixed p-value
parameters::model_parameters(
  model = resinModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  dplyr::mutate(
    p = ifelse(
      test = is.na(p),
      yes = NA,
      no = pvalRound(p)
    )
  ) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Resin Lifetimes Study",
  format = "latex",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  )
```

Table 2: ANOVA Table for Resin Lifetimes Study

Source	SS	df	MS	F	p-value	Omega Sq.	Eta Sq.	Epslion Sq.
temp	3.5376	4	0.8844	96.363	< 0.0001	0.9116	0.9233	0.9138
Residuals	0.2937	32	0.0092					

Either table is sufficient; you only need to worry about this IF you have a p-value that is sufficiently close to zero. You can then proceed with providing interpretations of F, the p-value, the effect sizes, and make a decision.

Reporting Point Estimates

One of the last things that we're going to do here is get the point estimates for our Grand Mean and treatment effects.

Make sure that you've told R to use the Sum to Zero constraint before proceeding.

To get these point estimates, we will use the `dummy.coef` function:

```
# DEMO CODE

# Getting coefficients/point estimates
dummy.coef(resinModel)
```

```
## Full coefficients are
##
## (Intercept):        1.43794
## temp:                   175         194         213         231         250
##                  0.49455952  0.19080952 -0.06044048 -0.24365476 -0.38127381
```

A couple of things to note: these values are slightly different from what Oehlert tabulated in Example 3.5 (p. 42). However, if you add the Grand Mean value (i.e., "(Intercept)") and the estimate for each treatment effect, you will get to the values of $\widehat{\mu_i}$ in Example 3.5 as well as the listed mean values in Listings 3.1 (p. 50) and 3.2 (p. 51). (I have programmed the homework answers based on R's evaluations.)

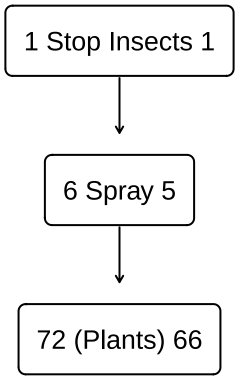
You should be prepared to interpret these estimates as well as format them in a professional layout. Note: `dummy.coef` will return a list of two elements: the grand mean and the treatment effects.

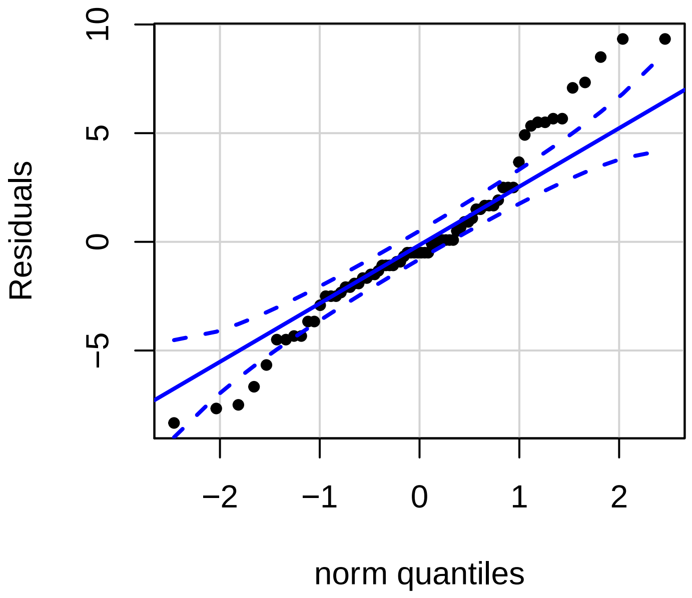
Your Turn

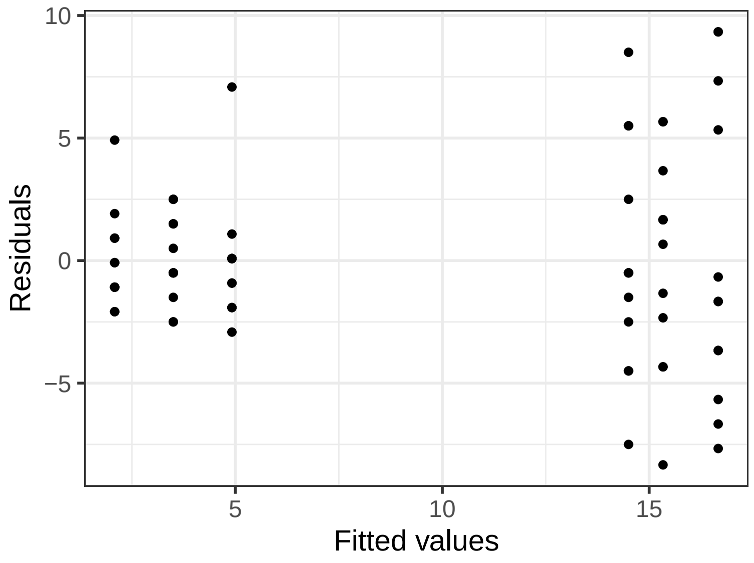
Now is an opportunity for you to get some practice. I'm going to use the data frame `InsectSprays`, which is built into R. You may load this data into your session with the command `data("InsectSprays")`.

A bit of background: the original researchers were exploring the effectiveness of various sprays on reducing the number of instances of particular type of insect for a crop. Each observation is randomly sampled plant from a field. NOTE: we do not know the measurement order.

Explore these data and build the elements that you would include a report as shown above. I'm going to provide several outputs, but no narrative. Use these as reminders for what to do. If you get stuck, check out the code appendix to see the code that I used.




Residuals
10
5
0
−5
−2
−1
0
1
2
norm quantiles


Residuals
10
5
0
−5
5
10
15
Fitted values

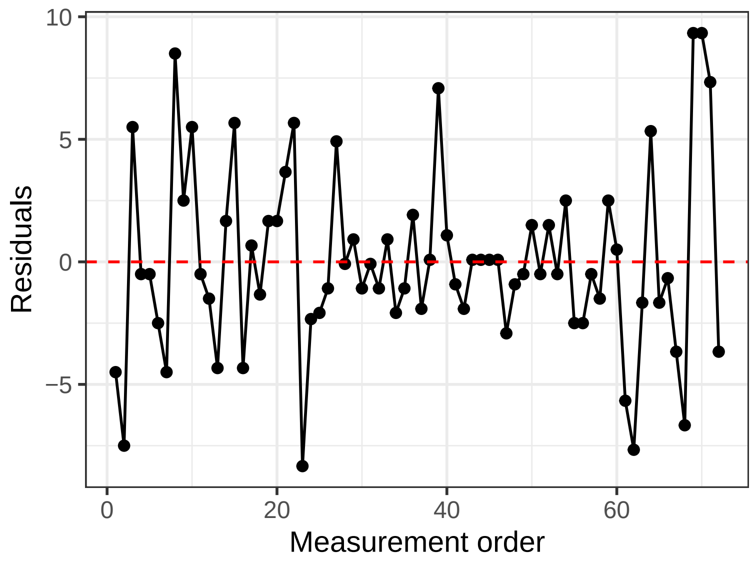


Table 3: ANOVA Table for Insect Spray Study

Source	SS	df	MS	F	p-value	Omega Sq.	Eta Sq.	Epslion Sq.
spray	2668.833	5	533.7667	34.7023	< 0.0001	0.7006	0.7244	0.7036
Residuals	1015.167	66	15.3813					

Table 4: Point Estimates from the Insect Spray Study

	Estimate
Grand Mean	9.50
Spray A	5.00
Spray B	5.83
Spray C	-7.42
Spray D	-4.58
Spray E	-6.00
Spray F	7.17

Code Appendix

```r
# Setting Document Options
knitr::opts_chunk$set(
  echo = FALSE,
  warning = FALSE,
  message = FALSE,
  fig.align = "center"
)

packages <- c("tidyverse", "hasseDiagram", "knitr",
              "kableExtra", "car", "psych",
              "parameters")
lapply(packages, library, character.only = TRUE)

# Tell Knitr to use empty space instead of NA in printed tables
options(knitr.kable.NA = "")

## DEMO CODE
options(contrasts = c("contr.sum", "contr.poly"))

## DEMO CODE

# Honey Data
honey <- data.frame(
  Amount = c(150, 50, 100, 85, 90, 95, 130, 50, 80),
  Varietal = rep(c("Clover", "Orange Blossom", "Alfalfa"), each = 3)
)
## Set Varietal to factor
honey$Varietal <- as.factor(honey$Varietal)

# Resin Lifetimes Data
resin <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/resinLifetimes.dat",
  header = TRUE,
  sep = ""
)
## Set temp to factor
resin$temp <- as.factor(resin$temp)

# DEMO CODE

# Hasse Diagram for the Honey Study
modelLabels <- c("1 Make Honey 1", "3 Type 2", "9 (Hives) 6")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, TRUE, FALSE),
  nrow = 3,
  ncol = 3,
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)
```

```r
# DEMO CODE

# Fit our One-way ANOVA model to the honey data
honeyModel <- aov(
  formula = Amount ~ Varietal,
  data = honey,
  na.action = "na.omit"
)

# DEMO CODE

# QQ plot for Honey Residuals
car::qqPlot(
  x = honeyModel$residuals,
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals"
)

# DEMO CODE

# Strip chart for Honey Residuals
ggplot(
  data = data.frame(
    residuals = honeyModel$residuals,
    fitted = honeyModel$fitted.values
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 1) +
  theme_bw() +
  xlab("Fitted values (lbs)") +
  ylab("Residuals (lbs)")

# DEMO CODE

# Index Plot for Honey Residuals
ggplot(
  data = data.frame(
    residuals = honeyModel$residuals,
    index = 1:length(honeyModel$residuals)
  ),
  mapping = aes(x = index, y = residuals)
) +
  geom_point(size = 1.5) +
  geom_line() +
  theme_bw() +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "red"
  ) +
  xlab("Measurement order") +
  ylab("Residuals")
# Hasse Diagram from Resin Study
modelLabels <- c("1 Fail 1", "5 Stress Temperature 4", "37 (Integrated Circuits) 32")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, TRUE, FALSE),
  nrow = 3,
  ncol = 3,
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)

# Fit our One-way ANOVA model to the honey data
resinModel <- aov(
  formula = y ~ temp,
  data = resin,
  na.action = "na.omit"
)

# QQ plot for Resin Lifetime Residuals
car::qqPlot(
  x = resinModel$residuals,
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals"
)

# Strip chart for Resin Lifetime Residuals
ggplot(
  data = data.frame(
    residuals = resinModel$residuals,
    fitted = resinModel$fitted.values
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 1) +
  theme_bw() +
  xlab("Fitted values (log10 hours)") +
  ylab("Residuals (log10 hours)")

# Index Plot for Resin Residuals
ggplot(
  data = data.frame(
    residuals = resinModel$residuals,
    index = 1:length(resinModel$residuals)
  ),
  mapping = aes(x = index, y = residuals)
```

```r
) +
  geom_point(size = 1.5) +
  geom_line() +
  theme_bw() +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "red"
  ) +
  xlab("Measurement order") +
  ylab("Residuals")
# DEMO CODE
# Displaying the results of the Parametric Shortcut via summary
summary(resinModel)

# DEMO CODE
# Displaying the results of the Parametric Shortcut via anova
anova(resinModel)

# DEMO CODE

# Create professional looking modern ANOVA table
parameters::model_parameters(
  model = resinModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Resin Lifetimes Study",
  format = "latex",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  ) %>%
  kableExtra::footnote(
    general = "Computer rounding has made the p-value look like zero.",
    general_title = "Note. ",
    footnote_as_chunk = TRUE
  )

# Set up the p-value rounding function.
pvalRound <- function(x){
  if (x < 0.0001) {
    return("< 0.0001")
  } else {
    return(x)
  }
}

# DEMO CODE

# Professional looking modern ANOVA table with fixed p-value
parameters::model_parameters(
  model = resinModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  dplyr::mutate(
    p = ifelse(
      test = is.na(p),
      yes = NA,
      no = pvalRound(p)
    )
  ) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Resin Lifetimes Study",
  format = "latex",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  )

# DEMO CODE

# Getting coefficients/point estimates
dummy.coef(resinModel)

# Your Turn Code --------------------------------------------------------------

# Hasse Diagram for Insect Spray Study
modelLabels <- c("1 Stop Insects 1", "6 Spray 5", "72 (Plants) 66")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, TRUE, FALSE),
  nrow = 3,
  ncol = 3,
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
```

```r
)

# Load Data
data("InsectSprays")

# Fit Model
isModel <- aov(
  formula = count ~ spray,
  data = InsectSprays
)

# Assess Gaussian Assumption with a QQ Plot
car::qqPlot(
  x = isModel$residuals,
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals"
)

# Supplement with Skewness and Kurtosis
isRSkew <- psych::skew(isModel$residuals)
isRKurt <- psych::kurtosi(isModel$residuals)

# Use a strip plot to assess homoscedasticity
ggplot(
  data = data.frame(
    residuals = isModel$residuals,
    fitted = isModel$fitted.values
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 1) +
  theme_bw() +
  xlab("Fitted values") +
  ylab("Residuals")

# RED HERRING
# We don't know measurement order so this plot does not actually
# provide any assistance to us.

# Index Plot for Resin Residuals
ggplot(
  data = data.frame(
    residuals = isModel$residuals,
    index = 1:length(isModel$residuals)
  ),
  mapping = aes(x = index, y = residuals)
) +
  geom_point(size = 1.5) +
  geom_line() +
  theme_bw() +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "red"
  ) +
  xlab("Measurement order") +
  ylab("Residuals")

dw <- car::durbinWatsonTest(resinModel)$dw

# Insect Sprays ANOVA Table
parameters::model_parameters(
  model = isModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  dplyr::mutate(
    p = ifelse(
      test = is.na(p),
      yes = NA,
      no = pvalRound(p)
    )
  ) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."
    ),
  caption = "ANOVA Table for Insect Spray Study",
  format = "latex",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("scale_down", "HOLD_position")
  )

# Point Estimates for Insect Sprays
pEst <- dummy.coef(isModel)
pEst <- unlist(pEst)
names(pEst) <- c("Grand Mean", "Spray A", "Spray B",
                 "Spray C", "Spray D", "Spray E", "Spray F")

data.frame("Estimate" = pEst) %>%
  knitr::kable(
  digits = 2,
  caption = "Point Estimates from the Insect Spray Study",
  format = "latex",
  booktabs = TRUE,
  align = "c"
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```