ANOVA Models with a Block

Neil J. Hatfield

4/5/2022

In this tutorial, we are going to explore looking at ANOVA models involving blocks. Specifically, we'll look at Randomized Complete Block Designs (RCBDs). While we'll focus mainly on One-way ANOVA + a Block, we can expand/extend this to incorporate more than one factor. We will only look at a parametric shortcut approach. The general structure for this guide is as follows:

- Setting up `R`
  - Loading Packages, Setting Options, and Additional Tools
- Data Contexts
  - Estimation Errors
    - Background
    - Load Data
    - Check Appropriateness of ANOVA Methods
  - Farming Barley
    - Background
    - Load Data
    - Check Appropriateness of ANOVA Methods
- Fit the Models
- Assessing Assumptions
  - Gaussian Residuals
  - Homoscedasticity
  - Independence of Observations
  - Lack of Interaction Between Factor(s) and Block
- Results
  - Omnibus Results
  - Relative Efficiency
  - Point Estimates
  - Post Hoc–Pairwise
  - Post Hoc–Contrasts
- Your Turn–Results for Estimation Errors

Setting up `R`

Just as in the prior guides/tutorials, we have to first ensure that `R` is properly configured and prepared for our work. We will want to ensure that we load all of the appropriate packages, set our constraint, and load in any additional tools. As a reminder, the following code does all of these things:

```
# Demo code to set up R
## Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "hasseDiagram", "DescTools")
lapply(packages, library, character.only = TRUE)

## Set options and constraint
options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

## Load useful tools
source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")
```

Data Contexts

For this tutorial, we're going to look at two contexts: **Estimation Errors** and **Farming Barley**. I'll give a brief description of each context, show how to load the data, and discuss whether a One-way ANOVA + Block model is appropriate.

Estimation Errors

This context comes from our design with the same name in Unit 2. Here, we wanted to explore the impact of three different methods on estimating the height of the US flag pole, when accounting for the sex of the person doing the estimating.

In this context, we were using sex (dichotomous; Female, Not Female) as a block so that we can attempt to get a "cleaner"/better read on the impact of the estimation method (three fixed levels: visual, tool, tool plus training). We can get the data from the URL that I emailed out:

```
# Demo code for loading data
## Estimation Errors data
estimationData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/estimationErrors2022.csv",
  header = TRUE,
  sep = ","
)

## Set method as factor
### I'm choosing to impose a particular order
estimationData$Method <- factor(
  x = estimationData$Method,
  levels = c("Visual", "Tool", "Training")
)

## Set the block as a "factor"
estimationData$Sex <- as.factor(estimationData$Sex)
```

Notice that immediately after loading the data I told `R` what my factor is (`Method`). I also told `R` to think of `Sex` as a factor, even though this is a block and not a factor. We want to do this so that `R` will do the correct calculations for us.

Checking Appropriateness

To check the appropriateness of our model, we engage in the same kinds of reasoning as we did in Unit 3. Specifically,

- Do we have a quantitative response?
- Do we have a qualitative/categorical factor?
- Are we forming an additive model?
- Can we estimate all of the effects?
- Can we estimate the errors?

The only new piece here is to ask ourselves "Did we incorporate our block into the design and model?"

Just as in Unit 3, we can use a Hasse diagram to help us out.

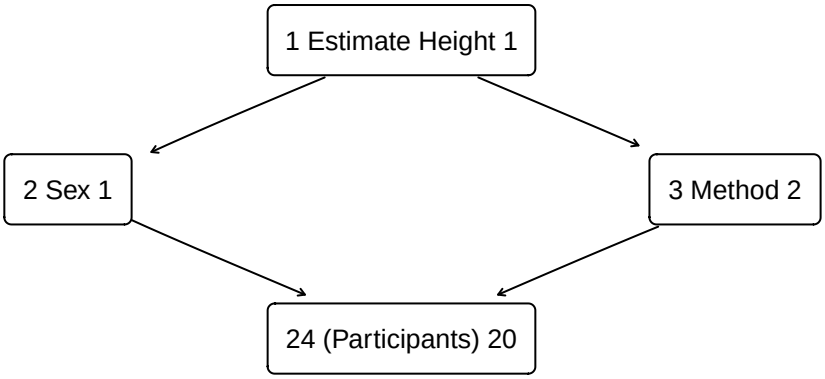


Figure 1: Hasse Diagram for Estimation Errors Study

As we look at Figure 1, we can see that we've incorporated our block, we have an additive model, and we have positive degrees of freedom everywhere indicating that we can estimate both effects and errors.

Farming Barley

A farmer wants to test out four varieties of barley and see if there is any difference in yield (bussels per acre). He has four fields in which he can plant the barley. However, the farmer is aware of differences between each field. For example,

- One field has a higher clay content in the soil than the others
- One field has rockier soil than the others
- Two fields are in wetter climates; two are in drier climates
- One field has loose soil while another field has much more compacted soil.
- Two fields are relatively flat, one has a hill in the middle, and the last has a valley.

Given that the fields will be our measurement units, there is quite a bit of variation between them. This variation could easily become confounded with the impact of barley variety on crop yield. Thus, we're in a perfect situation to make use a block of field.

We can access and clean the data through the following code:

```
# Load Barley Data
barleyData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/barley.dat",
  header = TRUE,
  sep = ","
)

# Glancing at the data frame, we can improve our column names (variables)
# as well as set our factor and block to be.
```

```r
## I'm going to change "Treatment" to "Varietal"
names(barleyData)[which(names(barleyData) == "Treatment")] <- "Varietal"

## Set Varietal as a Factor
barleyData$Varietal <- as.factor(barleyData$Varietal)

## Tell R to consider our block, Field, as a "factor"
barleyData$Field <- as.factor(barleyData$Field)

## I'm also going to simplify "Planting.Harvesting.Order" to just "Order"
names(barleyData)[which(names(barleyData) == "Planting.Harvesting.Order")] <- "Order"
```

Checking Appropriateness

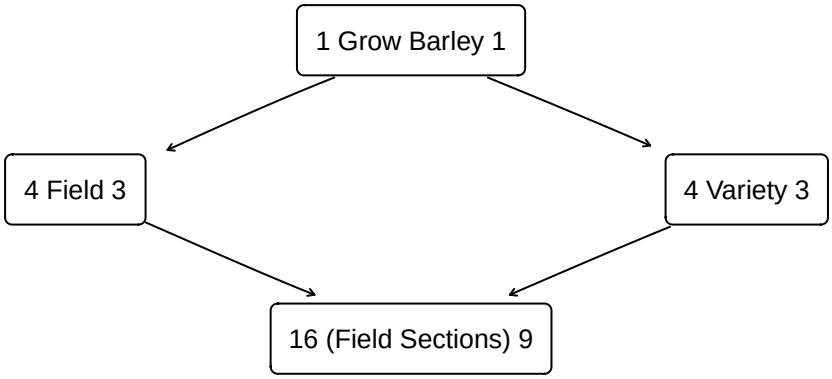


Figure 2: Hasse Diagram for Barley Crop Yield Study

From Figure 2 we can see that for our continuous response crop yield (bushels per acre) and our factor of variety of barley planted (4 types), we have an additive model with estimable effects and errors. Further, we've incorporated our four fields into the model to block any confounding between them and the type of barely.

Fit the Models

Keep in mind that all we are doing is adding new screens/terms into our model. Thus, fitting our new models is essentially just like in Unit 3. The only difference is that we place more terms after the tilde, ~, in our formula statements.

```r
# Demo code for fitting models

## Our formulas have the same basic structure:
## response ~ term1 + term2
## Notice the plus sign to reflect that we're making an *additive* model
## You can use either response ~ block + factor OR response ~ factor + block
## The order does not matter for the analysis BUT
## will play an impact on how values get calculated and listed
## For example, the order of rows in the ANOVA table or the listing of
## point estimates in dummy.coef

## Estimation Errors Study
estimationModel <- aov(
  formula = Estimate ~ Sex + Method,
  data = estimationData
```

```
)

## Farming Barley Study
barleyModel <- aov(
  formula = Yield ~ Field + Varietal,
  data = barleyData
)
```

Assessing Assumptions

In terms of assumptions, we still have our core three: Gaussian Residuals, Homoscedasticity, and Independence of Observations. With the block, we also want to check an interaction plot to make sure that there isn't anything strange/unexpected going on.

Gaussian Residuals

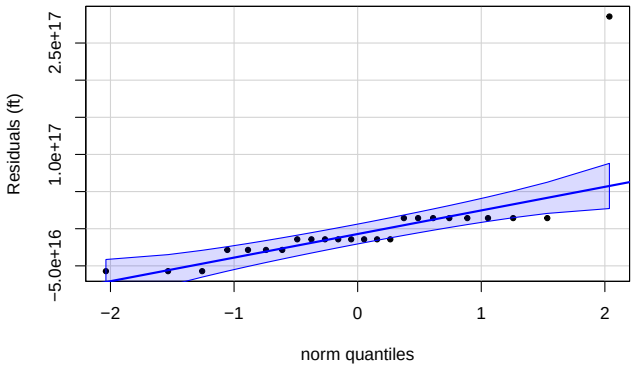
Just as before we will make use of QQ Plots.

```
# Chunk options for side-by-side
### fig.subcap=c("Estimation Errors Study", "Farming Barley Study"), fig.ncol=2,
### out.width="50%"

# QQ Plot for Estimation Model Residuals
car::qqPlot(
  x = residuals(estimationModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (ft)"
)

# QQ Plot for Barley Model Residuals
car::qqPlot(
  x = residuals(barleyModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (BPA)"
)
```

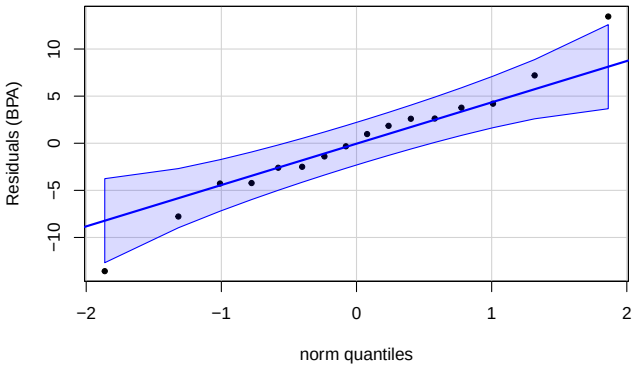
For the Farming Barley study, we have two of 16 residuals beyond the envelope or 12.5%. Given the balanced design, we will proceed with saying that this assumption has been satisfied for the Farming Barley study.

We have a bigger issue for the Estimation Errors study–we have potential outlier that may be highly influential. Participant 14 gave an estimate of 3.42816×10^{17} feet; this is several orders of magnitude (~15) beyond all other observations. Let's look at this model without this participant:

```
# Demo code for dropping single observation
## Estimation Errors Study
newEstModel <- aov(
  formula = Estimate ~ Sex + Method,
  data = estimationData[-14,] #drops row 14
```



(a) Estimation Errors Study



(b) Farming Barley Study

Figure 3: QQ Plots

```
)

par(mar = c(4, 4, 1, 1)) # Adjusting margins for aesthetics
car::qqPlot(
  x = residuals(newEstModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (ft)"
)
```

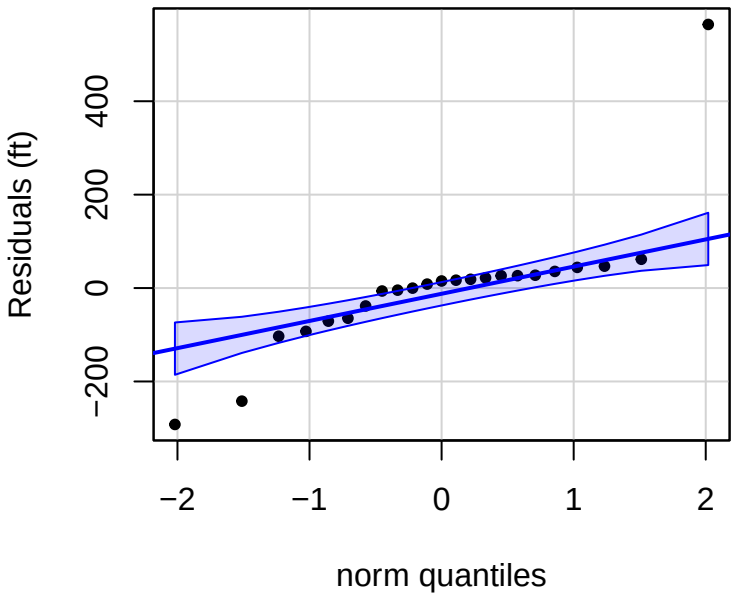


Figure 4: Updated QQ Plot for Estimation Errors Study

In Figure 4, we can see that we now have about four observations outside the envelope of 23 total. This is approxi-

mately 17%. We will want to cautiously proceed with the Gaussian assumption given our near balanced design.

Assessing Homoscedasticity

When we account for the the block in our model, we won't get the nice strips that we're used to seeing. However, we can still use the basic idea of the strip chart to assess homoscedasticity. We can still look for patterns and we'll add a smoother to our plot, transforming the strip chart to a Tukey-Anscombe Plot.

Farming Barley

We don't see any patterns to the plot (Figure 5), which is a good sign. While there does appear to be more variation on the high end of the fitted values, the smoothed line is fairly flat. The upticks at the ends might be the result of the small sample size. In all, we will take the homoscedasticity assumption to be satisfied.

```
# Demo code for making Tukey-Anscombe plot
## Farming Barley Study
ggplot(
  data = data.frame(
    residuals = residuals(barleyModel),
    fitted = fitted.values(barleyModel)
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline( ## Adds reference line at zero
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth( ## Adds the smoothed line
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (BPA)") +
  ylab("Residuals (BPA)")
```

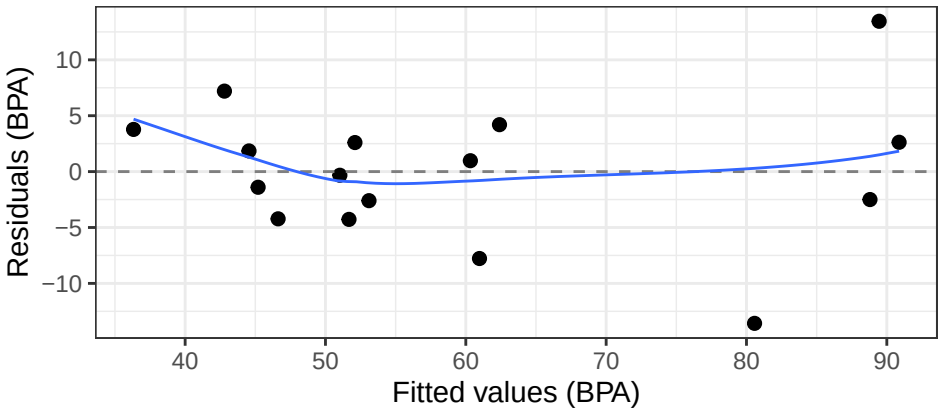
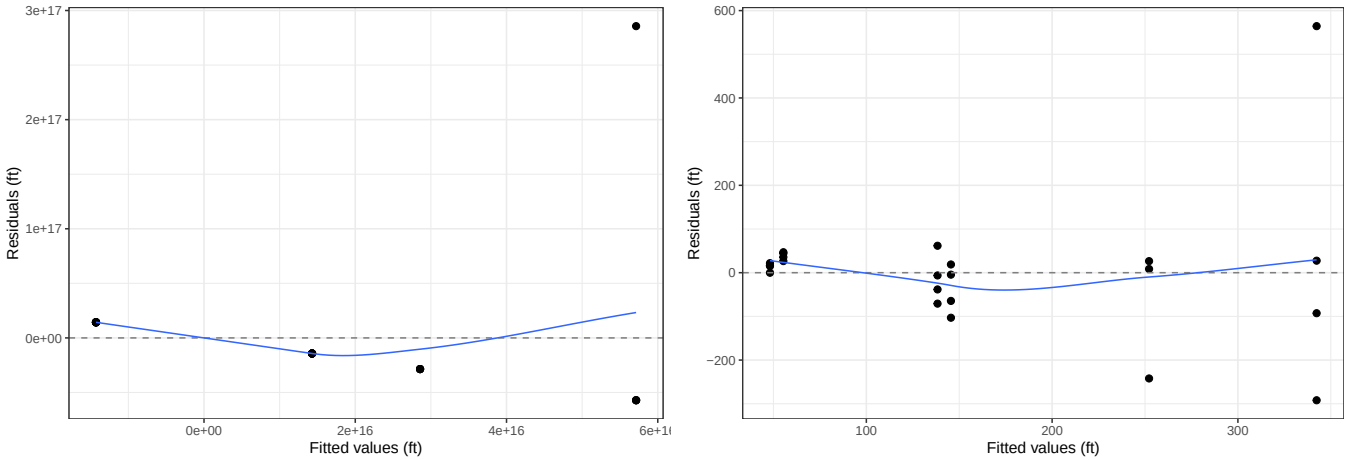


Figure 5: Tukey-Anscombe Plot for Barley Crop Yield Study

Estimation Errors

Figure 6 highlights the continued impact of the one extreme potential outlier. On the left, we can see that this one point causes all other senses of variation become obscured. While the smoothed line is fairly flat, I am concerned about the megaphone pattern that appears in the right panel. I question whether we satisfy the Homoscedasticity assumption for the Estimation Errors Study.

```
# Demo code for making Tukey-Anscombe plot
## Estimation Errors Study--All Observations
ggplot(
  data = data.frame(
    residuals = residuals(estimationModel),
    fitted = fitted.values(estimationModel)
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth(
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (ft)") +
  ylab("Residuals (ft)")

## Estimation Errors Study--Drop Extreme Potential Outlier
ggplot(
  data = data.frame(
    residuals = residuals(newEstModel),
    fitted = fitted.values(newEstModel)
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth(
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (ft)") +
  ylab("Residuals (ft)")
```



(a) Full Data Set

(b) Dropping Extreme Potential Outlier

Figure 6: Tukey-Anscombe Plots for Estimation Errors Study

Assessing Independence of Observations

Again, we can assess the Independence of Observations in the same ways as from Unit 3: making use of our knowledge of the study design, and, IF we know measurement order, index plots.

Estimation Errors

We do not have an accurate accounting of measurement order for the Estimation Errors study. Thus, we will only have to go off of our knowledge of the study design and how we carried out the study. We took every precaution to ensure that our participants did not influence each other.

Farming Barley

In the Farming Barley study, we do know measurement order. Thus, we can make use of an Index plot.

```
# Demo code for index plots
## Farming Barley Study
## Using the residuals from the model which includes the block
ggplot(
  data = data.frame(
    residuals = barleyModel$residuals,
    index = 1:length(barleyModel$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")
```

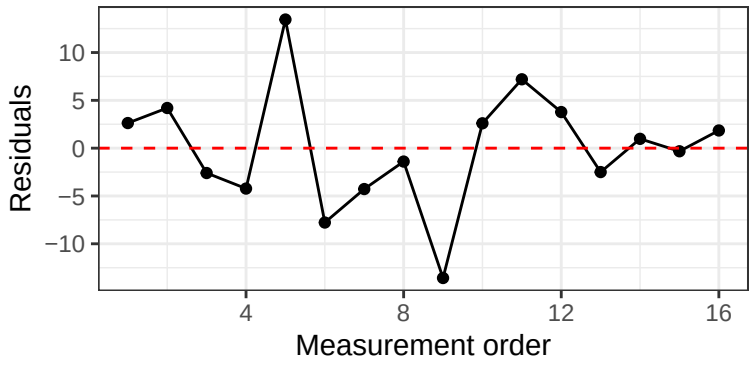


Figure 7: Index Plot for Farming Barley Residuals

From Figure 7, we don't necessarily see any patterns which would indicate a threat of the assumption of independent observations.

We can also form Index Plots in a different way: rather than looking at the residuals, we can also plot the actual response values. However, if we go this route, we will want to incorporate our block into the Index Plot.

```
# Demo code
# Alternative Index Plot for Barley Yields
ggplot(
  data = barleyData,
  mapping = aes(
    x = Order,
    y = Yield,
    color = Field,
    shape = Varietal,
    linetype = Field
  )
) +
  geom_point(size = 2) +
  geom_path(
    mapping = aes(group = Field)
  ) +
  ggplot2::theme_bw() +
  xlab("Planting/Havesting Order") +
  ylab("Yield (BPA)")
```

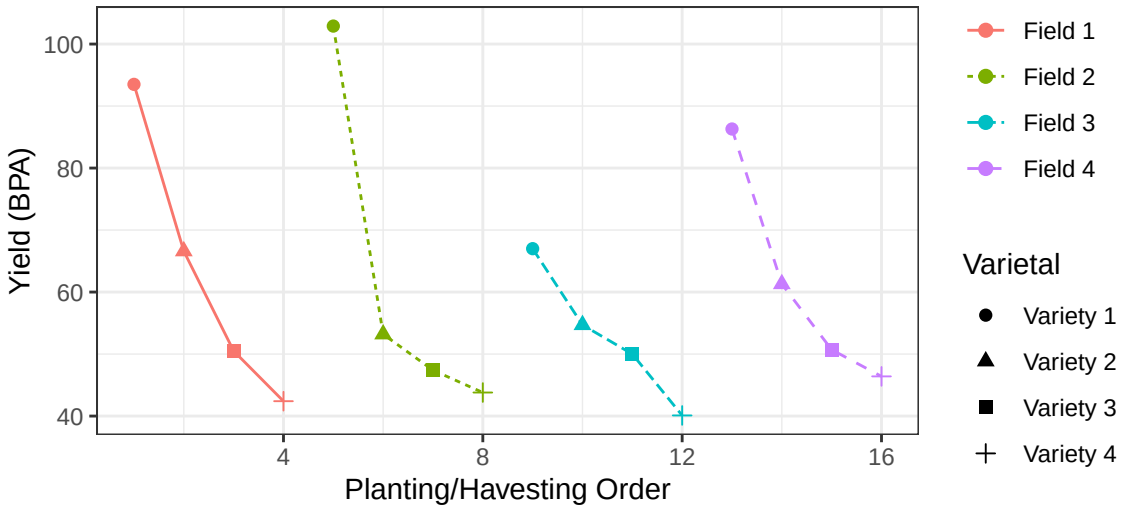


Figure 8: Index Plot for Barley Crop Yield Residuals

While at first glance, there appears to be a pattern to the response (Figure 8; a repeating downward line), we can see that this is an artifact of the block and treatments.

Assessing Interaction

The last aspect we need to check is whether there is an interaction between our block and our factor. If there is an interaction, then our model (response = factor + block) is not valid. We will make an interaction plot to assess this issue.

For an interaction plot, we will plot the values of the *Sample Arithmetic Mean* across all combinations of blocks and factors look for consistency in performance. If there is no interaction, then we should see consistent behaviors (essentially parallel lines). If there is interaction, then we should switches in behavior such as non-parallel lines moving in opposite directions.

Farming Barley

```
# Demo code for an interaction plot
# Interaction Plot for Field and Treatment
ggplot2::ggplot(
  data = barleyData,
  mapping = aes(
    x = Varietal,
    y = Yield,
    color = Field,
    shape = Field,
    linetype = Field,
    group = Field
  )
) +
  stat_summary(fun = "mean", geom = "point") +
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Variety") +
  ylab("Yield (BPA)") +
  labs(color = "Field") +
```

```
  theme(
    legend.position = "right"
  )
```

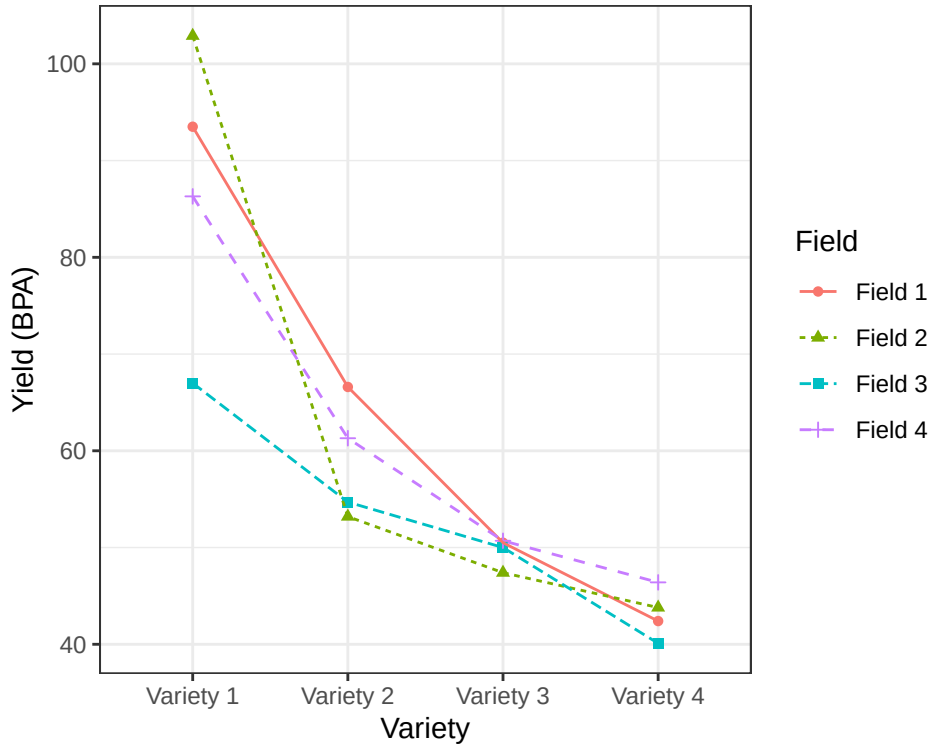


Figure 9: Interaction Plot for Barley Varietal and Field

In the interaction plot (Figure 9), we see essentially the same behavior of barley variety in each field. This indicates that there is not any type of interaction between field and barley varietal. While the lines are not perfectly parallel, they all reflect the same general behavior. An interaction to be concerned about would be if for Variety 2, Field 4 had a yield higher than Variety 1, Field 2's yield

Estimation Errors Study

For the Estimation Errors study, we're going to look at two interaction plots: one involving the full data set (Figure 10) and one dropping the extreme potential outlier (Figure 11).

```
# Demo code for an interaction plot
# Interaction Plot for Sex and Method
ggplot2::ggplot(
  data = estimationData,
  mapping = aes(
    x = Method,
    y = Estimate,
    color = Sex,
    shape = Sex,
    linetype = Sex,
    group = Sex
  )
) +
  stat_summary(fun = "mean", geom = "point") +
```

```
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Estimation Method") +
  ylab("Estimate (ft)") +
  labs(color = "Sex") +
  theme(
    legend.position = "right"
  )
```

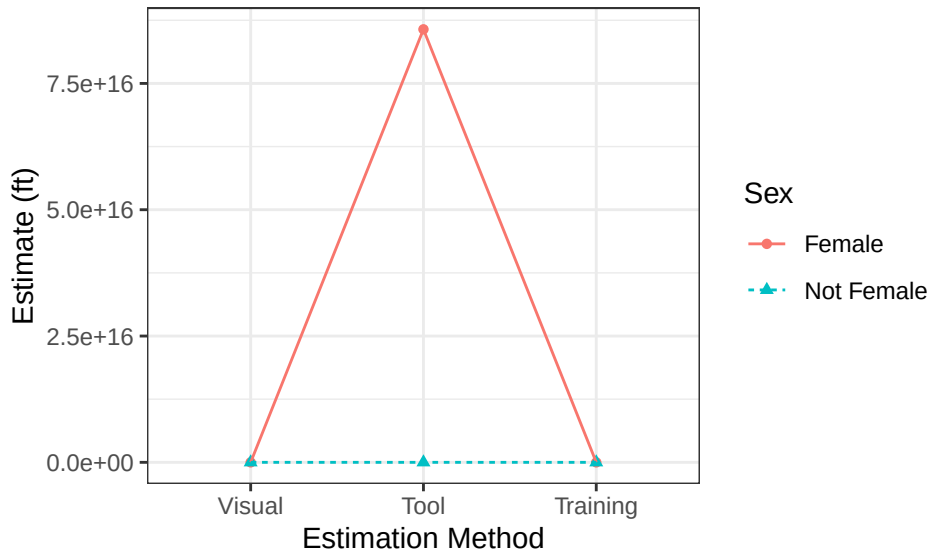


Figure 10: Interaction Plot for Full Estimation Errors Study

Again, we can see the impact of our extreme potential outlier. Visually, the outlier suggests in Figure 10 that we have an interaction between Sex and Method: for the female group we have a huge spike as we move through the methods while for the not female group we're fairly level.

If we remove this extreme potential outlier, we can see a different story altogether. Figure 11 shows that there does not appear to be any interaction between our block (sex) and our factor (estimation method).

```
# Demo code for an interaction plot
# Interaction Plot for Sex and Method
ggplot2::ggplot(
  data = estimationData[-14,],
  mapping = aes(
    x = Method,
    y = Estimate,
    color = Sex,
    shape = Sex,
    linetype = Sex,
    group = Sex
  )
) +
  stat_summary(fun = "mean", geom = "point") +
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Estimation Method") +
  ylab("Estimate (ft)") +
  labs(color = "Sex") +
```

```
theme(
  legend.position = "right"
)
```

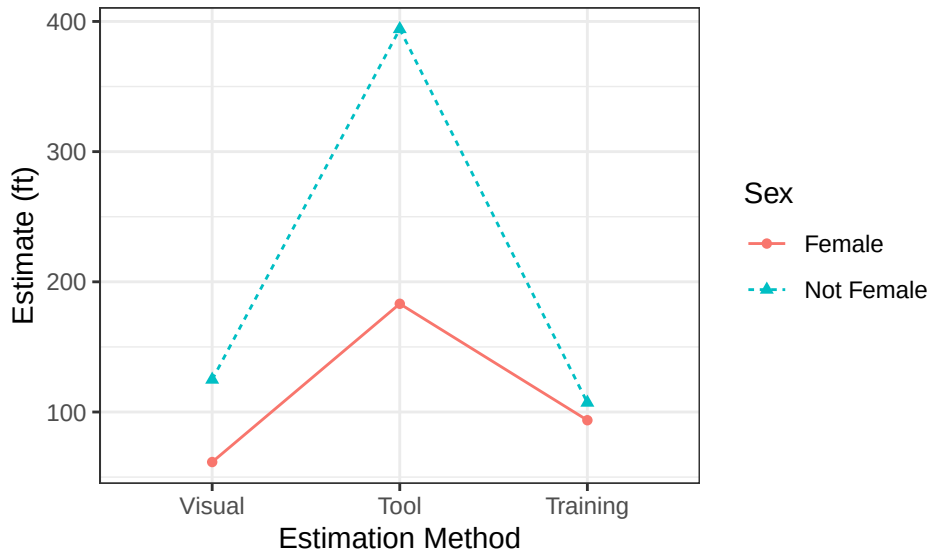


Figure 11: Interaction Plot for Reduced Estimation Errors Study

Assumption Decisions

For the Farming Barley study, I'm going to say that all four of the assumptions are satisfied.

For the Estimation Errors study, our assumptions are NOT satisfied. We have an extreme potential outlier as well as concerns to both the Gaussian and Homoscedasticity Assumptions. I'm going to propose that we fit a new model where we drop the extreme outlier and transform the data.

Your Turn–Transform the Estimation Data

Decide for yourself how best to transform the Estimation Errors data. Then re-check the assumptions.

Results

For the Results portion of this guide/tutorial, I'm going to focus on the Farming Barley study. I'll leave the transformed version of the Estimation Errors study to you as an exercise.

Remember, for any results section for a parametric shortcut, we have several parts: the omnibus tests and effect sizes, point estimates, and Post Hoc (pairwise and/or contrasts).

Omnibus Results

We generate our omnibus results *almost* exactly like we would for a one-way ANOVA model. The only change is in the Modern ANOVA table for the effect size estimates. Rather than using `"raw"`, we will use `"partial"` to reflect the fact that we now have multiple terms in our model

```
# Omnibus Test/Modern ANOVA Table
parameters::model_parameters(
    model = barleyModel,
    omega_squared = "partial",
    eta_squared = "partial",
    epsilon_squared = "partial"
) %>%
  knitr::kable(
    digits = 4,
  col.names = c("Source", "SS", "df", "MS", "F", "p-value",
                "Partial Omega Sq.", "Partial Eta Sq.", "Partial Epsilon Sq."),
  caption = "ANOVA Table for Barley Crop Yield Study",
  align = c('l',rep('c',8)),
  booktab = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  )
```

Table 1: ANOVA Table for Barley Crop Yield Study

Source	SS	df	MS	F	p-value	Partial Omega Sq.	Partial Eta Sq.	Partial Epsilon Sq.
Field	259.265	3	86.4217	1.3443	0.3203	0.0606	0.3094	0.0793
Varietal	4573.105	3	1524.3683	23.7116	0.0001	0.8098	0.8877	0.8503
Residuals	578.590	9	64.2878					

We still interpret all values just as we did in Unit 3. The biggest catch is how we treat the row of Table 1 connected to our block, Field. That is to say, we don't actually care about this row. This means that we won't interpret the F ratio, the p-value, or the effect sizes for Field. After all, we didn't want to test the impact of field on the barley yield; we just didn't want field to become confounded with the kind of barley planted (varietal).

We would still point out that the varietal planted accounted for nearly 24 times as much variation as what was left unexplained/accounted for by our model. This translates to varietal explaining around 85% of the total variation in yield. Under the null hypothesis, we would only anticipate this extreme of a result 1/100th of a percent of the time.

Relative Efficiency

One of the things that Randomized Complete Block Designs can help with is design efficiency. By this we mean, how much we can save in terms of sample size by using a block design versus a completely randomized design. To help us get this measure, I've written the `block.RelEff` function to help us.

```
# Demo code for Relative Efficiency of the Block
## Farming Barley Study
block.RelEff(
  aov.obj = barleyModel,
  blockName = "Field",
  trtName = "Varietal"
)
```

```
## [1] "The relative efficiency of the block, Field, is 1.028."
```

You can also run this code inline to have the resulting sentence appear as part of your narrative. For example, The relative efficiency of the block, Field, is 1.028..

We can interpret this relative efficiency as telling us how many times larger we would need the per group sample size to be if we didn't use the block.

Point Estimates

If we want to get point estimates for our Grand Mean, Treatment Effects, and Block Effects, we can. We use the same methods as before. I recommend that you first run `dummy.coef(anovaModel)` in your console so you can see the order of terms.

```
# Point Estimates for Farming Barley
pEst <- dummy.coef(barleyModel)
pEst <- unlist(pEst)
names(pEst) <- c(
  "Grand Mean",
  levels(barleyData$Field), # I know that this is the correct order because
  levels(barleyData$Varietal) # I ran dummy.coef(barleyModel) in my console first
  )

data.frame("Estimate" = pEst) %>%
  knitr::kable(
  digits = 3,
  caption = "Point Estimates from the Barley Crop Yield Study",
  booktabs = TRUE,
  align = "c"
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```

Table 2: Point Estimates from the Barley Crop Yield Study

	Estimate
Grand Mean	59.800
Field 1	3.450
Field 2	2.025
Field 3	-6.850
Field 4	1.375
Variety 1	27.625
Variety 2	-0.850
Variety 3	-10.150
Variety 4	-16.625

Again, we interpret these numbers as rates: "bushels per acre per test plot" or more simply, "yield per plot" (where we're using [test] plot to mean a quarter-subsection of a field).

If you don't want the estimates for the block, you can do the following:

```
pEst <- dummy.coef(barleyModel)
pEst <- unlist(pEst[which(names(pEst) != "Field")])
names(pEst) <- c(
  "Grand Mean",
  levels(barleyData$Varietal) # Using levels here will help stop the accidental
  # mislabeling of estimates
  )

data.frame("Estimate" = pEst) %>%
  knitr::kable(
  digits = 3,
  caption = "Point Estimates from the Barley Crop Yield Study",
  booktabs = TRUE,
  align = "c"
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```

Table 3: Point Estimates from the Barley Crop Yield Study

	Estimate
Grand Mean	59.800
Variety 1	27.625
Variety 2	-0.850
Variety 3	-10.150
Variety 4	-16.625

Post Hoc–Pairwise

From the slides, we noted that we were going to control the MEER at 0.1 by using Tukey's HSD. Thus, we will do so for our pairwise comparisons.

```
# Post Hoc Analysis--Pairwise comparisons
barleyPH <- TukeyHSD(
  x = barleyModel,
  which = "Varietal", #We need to specify which model term we want
  conf.level = 0.9
)

knitr::kable(
  barleyPH$Varietal,
  digits = 4,
  caption = "Post Hoc Tukey HSD Comparisons",
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'cccc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
```

```
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 4: Post Hoc Tukey HSD Comparisons

	Difference	Lower Bound	Upper Bound	Adj. p-Value
Variety 2-Variety 1	-28.475	-43.5533	-13.3967	0.0033
Variety 3-Variety 1	-37.775	-52.8533	-22.6967	0.0004
Variety 4-Variety 1	-44.250	-59.3283	-29.1717	0.0001
Variety 3-Variety 2	-9.300	-24.3783	5.7783	0.4053
Variety 4-Variety 2	-15.775	-30.8533	-0.6967	0.0833
Variety 4-Variety 3	-6.475	-21.5533	8.6033	0.6747

Notice that we added on the `which` argument to the `TukeyHSD` call. This isolates the appropriate portion of the Tukey HSD output related to our factor of interest. If you omit this, you'll get Tukey HSD reports for every term in the model. In the Farming Barley situation, one for Field and one for Varietal.

We still interpret the results in the exact same way.

You can make similar adjustments in the `DescTools::PostHocTest` if you are wanting to control a different Type I Error Rate and/or use a different method. Table 5 provides an example using Bonferroni's method for controlling SCI.

```
## DescTools Pairwise Method
dtPH <- DescTools::PostHocTest(
  x = barleyModel, # Your aov/lm object
  which = "Varietal", # Specify which factor
  method = "bonf", # Your chosen method
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)

## Kable Code for DescTools
knitr::kable(
  x = dtPH$Varietal, # Notice the use of the factor name
  digits = 3,
  caption = paste( # Creates a nice title; copy at will
    "Post Hoc",
    attr(dtPH, "method"),
    "Comparisons"
  ),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 5: Post Hoc Bonferroni Comparisons

	Difference	Lower Bound	Upper Bound	Adj. p-Value
Variety 2-Variety 1	-28.475	-45.106	-11.844	0.004
Variety 3-Variety 1	-37.775	-54.406	-21.144	0.001
Variety 4-Variety 1	-44.250	-60.881	-27.619	0.000
Variety 3-Variety 2	-9.300	-25.931	7.331	0.812
Variety 4-Variety 2	-15.775	-32.406	0.856	0.128
Variety 4-Variety 3	-6.475	-23.106	10.156	1.000

Note: Dunnett's Test does not currently allow for a block design.

Effect Sizes

To get the effect sizes for our desired pairwise comparisons, we will turn to the `anova.PostHoc` function. However, we need to take care with the arguments of this function:

- The `aov.obj` is the primary argument and is where we pass the `aov` (or `lm`) output to.
- The `response` argument takes a character string as the name of the response attribute.
- The `mainEffect` argument takes a character string that names the factor you are wanting to do pairwise comparisons on.

```
anova.PostHoc(
  aov.obj = barleyModel, # Our aov output
  response = "Yield", # Our response variable
  mainEffect = "Varietal" # Our factor variable
) %>%
  knitr::kable(
    digits = 3,
    caption = "Post Hoc Comparison Effect Sizes",
    col.names = c("Pairwise Comparison","Cohen's d", "Hedge's g",
                  "Prob. of Superiority"),
    align = 'lccc',
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 6: Post Hoc Comparison Effect Sizes

Pairwise Comparison	Cohen's d	Hedge's g	Prob. of Superiority
Variety.1 vs. Variety.2	2.451	2.131	0.958
Variety.1 vs. Variety.3	3.493	3.037	0.993
Variety.1 vs. Variety.4	4.052	3.523	0.998
Variety.2 vs. Variety.3	2.061	1.792	0.927
Variety.2 vs. Variety.4	3.313	2.881	0.990
Variety.3 vs. Variety.4	3.005	2.613	0.983

We need to make an important note here: suppose that our overall Unusualness Threshold was 0.1. From the Tukey HSD results, Variety 3 vs 2, 4 vs 2, and 4 vs 3 would NOT be statistically significant. From the effect size table, we would say that there are rather large effects as Cohen's d and Hedge's g are all quite large. Further, the probability of superiority for each pairing is far from 0.5 (no practical effect). Just as effect sizes temper statistical significance, statistical significance moderates effect sizes. In these three cases, while there appears to be a large effect, there is enough variation in those groups that the effect is not statistically large enough to escape through the noise of the group.

Post Hoc–Contrasts

Even with a block, we can still use make use of the idea of contrasts. For example, let's say that barley varieties 1 and 2 are from one company while 3 and 4 are from a second company. We can test the contrast of companies, even in this blocking design.

```
# Define the contrast
company <- c(1/2, 1/2, -1/2, -1/2)

# Bind the contrast to our factor
contrasts(barleyData$Varietal) <- company

# Refit our model so that our contrast gets tested
barleyContrast <- aov(
  formula = Yield ~ Varietal + Field,
  data = barleyData
)

# Get the updated ANOVA Table
## Remember, you could also use the DescTools package for Scheffé here
conOut <- summary.aov(
  object = barleyContrast,
  split = list(
    Varietal = list(
      "Company A vs. Company B" = 1
    )
  )
)

# Make a nice table
knitr::kable(
  x = conOut[[1]],
  digits = 4,
  col.names = c(
    "DF", "SS", "MS", "F", "p-value"),
  caption = "ANOVA Table for Barley Crop Yield Contrasts",
  booktabs = TRUE,
  align = rep("c", 5)
) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```

Table 7: ANOVA Table for Barley Crop Yield Contrasts

	DF	SS	MS	F	p-value
Varietal	3	4573.105	1524.3683	23.7116	0.0001
Varietal: Company A vs. Company B	1	2867.602	2867.6025	44.6057	0.0001
Field	3	259.265	86.4217	1.3443	0.3203
Residuals	9	578.590	64.2878		

Your Turn

Using your transformed Estimation Errors data, attempt to create the omnibus ANOVA table, get the relative efficiency for the block design, point estimates, and the pairwise comparisons.

Code Appendix

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

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

options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")
# Demo code to set up R
## Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "hasseDiagram", "DescTools")
lapply(packages, library, character.only = TRUE)

## Set options and constraint
options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

## Load useful tools
source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")

# Demo code for loading data
## Estimation Errors data
estimationData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/estimationErrors2022.csv",
  header = TRUE,
  sep = ","
)

## Set method as factor
### I'm choosing to impose a particular order
estimationData$Method <- factor(
  x = estimationData$Method,
  levels = c("Visual", "Tool", "Training")
)

## Set the block as a "factor"
estimationData$Sex <- as.factor(estimationData$Sex)

# Demo Code for Hasse diagram for Estimation Errors
modelLabels <- c("1 Estimate Height 1", "2 Sex 1", "3 Method 2", "24 (Participants) 20")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE,
           FALSE, TRUE, TRUE, TRUE, FALSE),
  nrow = 4,
  ncol = 4,
```

```
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)

# Load Barley Data
barleyData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/barley.dat",
  header = TRUE,
  sep = ","
)

# Glancing at the data frame, we can improve our column names (variables)
# as well as set our factor and block to be.

## I'm going to change "Treatment" to "Varietal"
names(barleyData)[which(names(barleyData) == "Treatment")] <- "Varietal"

## Set Varietal as a Factor
barleyData$Varietal <- as.factor(barleyData$Varietal)

## Tell R to consider our block, Field, as a "factor"
barleyData$Field <- as.factor(barleyData$Field)

## I'm also going to simplify "Planting.Harvesting.Order" to just "Order"
names(barleyData)[which(names(barleyData) == "Planting.Harvesting.Order")] <- "Order"

modelLabels <- c("1 Grow Barley 1", "4 Field 3", "4 Variety 3", "16 (Field Sections) 9")
modelMatrix <- matrix(
  data = c(FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE,
           FALSE, TRUE, TRUE, TRUE, FALSE),
  nrow = 4,
  ncol = 4,
  byrow = FALSE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)

# Demo code for fitting models

## Our formulas have the same basic structure:
## response ~ term1 + term2
## Notice the plus sign to reflect that we're making an *additive* model
## You can use either response ~ block + factor OR response ~ factor + block
## The order does not matter for the analysis BUT
## will play an impact on how values get calculated and listed
## For example, the order of rows in the ANOVA table or the listing of
## point estimates in dummy.coef

## Estimation Errors Study
estimationModel <- aov(
  formula = Estimate ~ Sex + Method,
```

```
  data = estimationData
)

## Farming Barley Study
barleyModel <- aov(
  formula = Yield ~ Field + Varietal,
  data = barleyData
)

# Chunk options for side-by-side
### fig.subcap=c("Estimation Errors Study", "Farming Barley Study"), fig.ncol=2,
### out.width="50%"

# QQ Plot for Estimation Model Residuals
car::qqPlot(
  x = residuals(estimationModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (ft)"
)

# QQ Plot for Barley Model Residuals
car::qqPlot(
  x = residuals(barleyModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (BPA)"
)

# Demo code for dropping single observation
## Estimation Errors Study
newEstModel <- aov(
  formula = Estimate ~ Sex + Method,
  data = estimationData[-14,] #drops row 14
)

par(mar = c(4, 4, 1, 1)) # Adjusting margins for aesthetics
car::qqPlot(
  x = residuals(newEstModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (ft)"
)

# Demo code for making Tukey-Anscombe plot
## Farming Barley Study
ggplot(
  data = data.frame(
    residuals = residuals(barleyModel),
    fitted = fitted.values(barleyModel)
```

```
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline( ## Adds reference line at zero
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth( ## Adds the smoothed line
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (BPA)") +
  ylab("Residuals (BPA)")

# Demo code for making Tukey-Anscombe plot
## Estimation Errors Study--All Observations
ggplot(
  data = data.frame(
    residuals = residuals(estimationModel),
    fitted = fitted.values(estimationModel)
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth(
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (ft)") +
  ylab("Residuals (ft)")

## Estimation Errors Study--Drop Extreme Potential Outlier
ggplot(
  data = data.frame(
    residuals = residuals(newEstModel),
    fitted = fitted.values(newEstModel)
  ),
  mapping = aes(x = fitted, y = residuals)
) +
  geom_point(size = 2) +
  geom_hline(
```

```
    yintercept = 0,
    linetype = "dashed",
    color = "grey50"
  ) +
  geom_smooth(
    formula = y ~ x,
    method = stats::loess,
    method.args = list(degree = 1),
    se = FALSE,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (ft)") +
  ylab("Residuals (ft)")

# Demo code for index plots
## Farming Barley Study
## Using the residuals from the model which includes the block
ggplot(
  data = data.frame(
    residuals = barleyModel$residuals,
    index = 1:length(barleyModel$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")
# Demo code
# Alternative Index Plot for Barley Yields
ggplot(
  data = barleyData,
  mapping = aes(
    x = Order,
    y = Yield,
    color = Field,
    shape = Varietal,
    linetype = Field
  )
) +
  geom_point(size = 2) +
  geom_path(
    mapping = aes(group = Field)
  ) +
  ggplot2::theme_bw() +
  xlab("Planting/Havesting Order") +
  ylab("Yield (BPA)")

# Demo code for an interaction plot
```

```r
# Interaction Plot for Field and Treatment
ggplot2::ggplot(
  data = barleyData,
  mapping = aes(
    x = Varietal,
    y = Yield,
    color = Field,
    shape = Field,
    linetype = Field,
    group = Field
  )
) +
  stat_summary(fun = "mean", geom = "point") +
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Variety") +
  ylab("Yield (BPA)") +
  labs(color = "Field") +
  theme(
    legend.position = "right"
  )

# Demo code for an interaction plot
# Interaction Plot for Sex and Method
ggplot2::ggplot(
  data = estimationData,
  mapping = aes(
    x = Method,
    y = Estimate,
    color = Sex,
    shape = Sex,
    linetype = Sex,
    group = Sex
  )
) +
  stat_summary(fun = "mean", geom = "point") +
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Estimation Method") +
  ylab("Estimate (ft)") +
  labs(color = "Sex") +
  theme(
    legend.position = "right"
  )

# Demo code for an interaction plot
# Interaction Plot for Sex and Method
ggplot2::ggplot(
  data = estimationData[-14,],
  mapping = aes(
    x = Method,
    y = Estimate,
    color = Sex,
    shape = Sex,
    linetype = Sex,
    group = Sex
```

```
  )
) +
  stat_summary(fun = "mean", geom = "point") +
  stat_summary(fun = "mean", geom = "line") +
  ggplot2::theme_bw() +
  xlab("Estimation Method") +
  ylab("Estimate (ft)") +
  labs(color = "Sex") +
  theme(
    legend.position = "right"
  )

# Omnibus Test/Modern ANOVA Table
parameters::model_parameters(
    model = barleyModel,
    omega_squared = "partial",
    eta_squared = "partial",
    epsilon_squared = "partial"
) %>%
  knitr::kable(
    digits = 4,
  col.names = c("Source", "SS", "df", "MS", "F", "p-value",
                "Partial Omega Sq.", "Partial Eta Sq.", "Partial Epsilon Sq."),
  caption = "ANOVA Table for Barley Crop Yield Study",
  align = c('l',rep('c',8)),
  booktab = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 10,
    latex_options = c("scale_down", "HOLD_position")
  )

# Demo code for Relative Efficiency of the Block
## Farming Barley Study
block.RelEff(
  aov.obj = barleyModel,
  blockName = "Field",
  trtName = "Varietal"
)

# Point Estimates for Farming Barley
pEst <- dummy.coef(barleyModel)
pEst <- unlist(pEst)
names(pEst) <- c(
  "Grand Mean",
  levels(barleyData$Field), # I know that this is the correct order because
  levels(barleyData$Varietal) # I ran dummy.coef(barleyModel) in my console first
  )

data.frame("Estimate" = pEst) %>%
  knitr::kable(
  digits = 3,
  caption = "Point Estimates from the Barley Crop Yield Study",
  booktabs = TRUE,
  align = "c"
```

```
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )

pEst <- dummy.coef(barleyModel)
pEst <- unlist(pEst[which(names(pEst) != "Field")])
names(pEst) <- c(
  "Grand Mean",
  levels(barleyData$Varietal) # Using levels here will help stop the accidental
  # mislabeling of estimates
  )

data.frame("Estimate" = pEst) %>%
  knitr::kable(
  digits = 3,
  caption = "Point Estimates from the Barley Crop Yield Study",
  booktabs = TRUE,
  align = "c"
  ) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )

# Post Hoc Analysis--Pairwise comparisons
barleyPH <- TukeyHSD(
  x = barleyModel,
  which = "Varietal", #We need to specify which model term we want
  conf.level = 0.9
)

knitr::kable(
  barleyPH$Varietal,
  digits = 4,
  caption = "Post Hoc Tukey HSD Comparisons",
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'cccc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

## DescTools Pairwise Method
dtPH <- DescTools::PostHocTest(
  x = barleyModel, # Your aov/lm object
  which = "Varietal", # Specify which factor
  method = "bonf", # Your chosen method
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)
```

```
## Kable Code for DescTools
knitr::kable(
  x = dtPH$Varietal, # Notice the use of the factor name
  digits = 3,
  caption = paste( # Creates a nice title; copy at will
    "Post Hoc",
    attr(dtPH, "method"),
    "Comparisons"
  ),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

anova.PostHoc(
  aov.obj = barleyModel, # Our aov output
  response = "Yield", # Our response variable
  mainEffect = "Varietal" # Our factor variable
) %>%
  knitr::kable(
    digits = 3,
    caption = "Post Hoc Comparison Effect Sizes",
    col.names = c("Pairwise Comparison","Cohen's d", "Hedge's g",
                  "Prob. of Superiority"),
    align = 'lccc',
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# Define the contrast
company <- c(1/2, 1/2, -1/2, -1/2)

# Bind the contrast to our factor
contrasts(barleyData$Varietal) <- company

# Refit our model so that our contrast gets tested
barleyContrast <- aov(
  formula = Yield ~ Varietal + Field,
  data = barleyData
)

# Get the updated ANOVA Table
## Remember, you could also use the DescTools package for Scheffé here
conOut <- summary.aov(
  object = barleyContrast,
  split = list(
```

```
    Varietal = list(
      "Company A vs. Company B" = 1
    )
  )
)

# Make a nice table
knitr::kable(
  x = conOut[[1]],
  digits = 4,
  col.names = c(
    "DF", "SS", "MS", "F", "p-value"),
  caption = "ANOVA Table for Barley Crop Yield Contrasts",
  booktabs = TRUE,
  align = rep("c", 5)
) %>%
  kableExtra::kable_styling(
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```