ANCOVA

Neil J. Hatfield

Last Updated: April 15, 2024

In this tutorial, we are going to explore Analysis of Covariance (ANCOVA) in R. For our purposes, we will be primarily focusing on *Parallel Lines/Separate Intercept* ANCOVA models with fixed effects (except for measurement units). Keep in mind that we can include a covariate (or several) in all of our prior models (One-way, [full] factorial, with/without a block).

At the end of this guide, I'll go over the various kinds of ANCOVA models.

1 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.

The core set of packages to load in include the following:

- `tidyverse`–for data cleaning/wrangling and the pipe, `%>%`
- `knitr` & `kableExtra`–for making professional looking tables
- `parameters`–to construct modern ANOVA tables, get omnibus effect sizes, and to switch out the type of *Sums of Squares*
- `hasseDiagram`–to construct Hasse diagrams
- `car` for nice QQ Plots
- `psych`–for easy descriptive statistics by group
- `emmeans`–for getting point estimates which attend to our models as well as doing post hoc analyses
- `rstatix`–our new package for detecting potential multivariate outliers

We are going to make use of one new package, `rstatix` for ANCOVA models. There is a function in this package, `mahalanobis_distance` that will help us with checking for any potential outliers.

As a reminder, the following code does all of these things:

```r
# Demo code to set up R ----
## Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "hasseDiagram", "car",
              "psych", "emmeans", "rstatix")
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")
```

2 Data Contexts

We will draw upon two contexts for our primary explorations: **Keyboarding Pain** and **Lego Set Prices**.

2.1 Keyboarding Pain

When people engage in repetitive motion, they can suffer from any one of a set of Repetitive Motion Disorders such as carpal tunnel syndrome, trigger finger, or tendinitis. We are wanting to understand the impact of the type of keyboard on how many hours of pain a person experiences in their hands, wrists, and forearms. We suspect that the number of hours a person spends keyboarding is related to the number of hours of pain that they feel. We have 12 volunteers who will use a specific keyboard we assign them for 2 weeks. During that time, they will record the number of hours they use the keyboard and the number of hours of repetitive motion pain during the study period.

2.1.1 Data

For this example, you'll want to import the data as shown below.

```r
# Demo Code for loading Keyboarding Data ----
keyboardingData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/keyboarding.dat",
  header = TRUE,
  sep = ""
)

# Column Notes
## hrs.pain is our response; hours experiencing pain
## kbd.type is our factor; type/style of keyboard
## hrs.kbd is our covariate; hours spent using the keyboard
### make sure that R is thinking of hrs.kbd as either as num or int

keyboardingData$kbd.type <- as.factor(keyboardingData$kbd.type)
```

2.2 Lego Set Prices

This study stems from Context 3 in the Unit 2 Study Design packet. In essence, we want to answer the question of how a Lego set's theme/collection impacts the price after accounting for the number of pieces in the set? In this study, we are looking at five particular themes/collections chosen by Neil and friends (i.e., their favorites): Architecture, Botanical, Harry Potter, Modular Buildings, and Star Wars.

One of Neil's friends scrapped two online databases to build a data file covering nearly the entire Lego catalog. Neil wrangled this data file and used the `slice_sample` function from the `dplyr` package to conduct a stratified random sample of sets based upon the theme/collection used in the study. The total sample size in our balanced study is 50.

2.2.1 Data

Here is how we can explore the Lego Data.

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

legoData$collection <- as.factor(legoData$collection)
```

3 Exploring the Data

I want to leave a quick note as a reminder that just as we've discussed all course, you should explore your data as thoroughly as possible at this junction. Look at multiple data visualizations and at various values of descriptive statistics. This will not only help you build your understanding of the data (think, EDA) but will also set the stage for assessing several of our assumptions.

4 Is ANCOVA Appropriate?

Before we write code to fit the ANCOVA model in `R`, we should check that ANCOVA is even appropriate.

4.1 Appropriateness of ANCOVA

To assess whether an ANCOVA model is even appropriate, we need to ensure that a general ANOVA model is appropriate plus the two extra conditions for ANCOVA. That is,

1) Do we have a quantitative (ideally, continuous) response?
2) Do we have at least one qualitative/categorical factor of interest?
3) Do we have enough *Degrees of Freedom* to estimate all effects of interest?
4) Do we have enough *Degrees of Freedom* to estimate residuals/errors?
5) Are we aiming for an additive model (up to interaction terms)?
6) Do we have a quantitative attribute (i.e., a covariate) that we believe is related to our response?
7) Is there a linear relationship between our response and our covariate?

Just as you might suspect, we check all of these base requirements just as we have been: run through the study design, looking for and verifying the elements. The Hasse diagram is our friend.

4.2 Hasse Diagrams

Speaking of Hasse diagrams, there is not an agreed upon convention for how to incorporate covariates into the diagram. I think of the covariate as being similar to a block and thus place a new node at the second highest level of the diagram; next to our main effects and block. Since there are many (in some cases, infinitely) levels to a covariate, we don't put a number of levels out front (to the left). Rather, we'll put the label "cov" for covariate. The number of *Degrees of Freedom* will depend upon which which type of ANCOVA model you fit. However, if you are fitting the standard ANCOVA model ("parallel lines"/"separate intercepts"), then the covariate will use 1 *Degree of Freedom*.

Figure 1 shows the Hasse diagram for the Keyboarding study. Notice that our covariate is labelled and uses one *Degree of Freedom*. We still have positive values for the rest of the nodes' *Degrees of Freedom* thus we should be able to estimate the effects and errors.

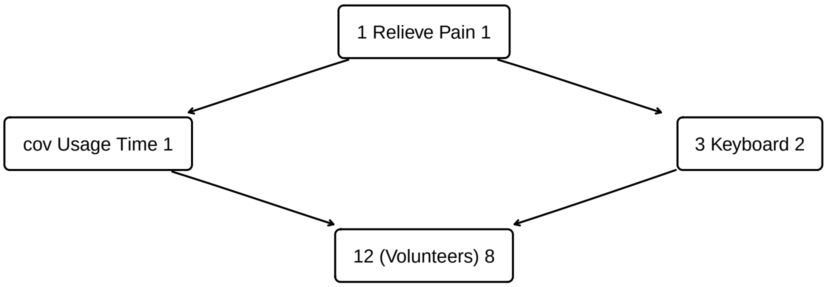


Figure 1: Hasse Diagram for Keyboarding Study

4.2.1 Your Turn

Use the information provided to create the Hasse diagram for the Lego Set Prices Study.

4.3 Linear Relationship

The last requirement for the ANCOVA is that there is a linear relationship between the response on the covariate. This requirement is also one of our assumptions. Thus, we'll look at this issue in that section.

4.4 Forming the Model

We are going to fit *two* models for ANCOVA: one will be our *actual* model that we use for checking all but two assumptions, and getting the results (both the omnibus and post hoc analyses). The *other* model we will use for checking the homogeneity of the covariate's slope parameter.

```r
# Demo Code for Fitting ANCOVA Models ----
## Keyboarding Study
### Our core model
keyboardModel <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type,
  data = keyboardingData
)

### Model for checking covariate's homogeneity
interactionCheck <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type + hrs.kbd:kbd.type,
  data = keyboardingData
)

## Lego Set Price Study
### Core model
legoModel <- aov(
  formula = price ~ numParts + collection,
  data = legoData
)

### Interacting check
legoCheck <- aov(
  formula = price ~ numParts + collection + numParts:collection,
  data = legoData
)
```

5 Assessing Assumptions

For ANCOVA models, we have six assumptions we need to check before using the parametric shortcut: our core three (Gaussian Residuals, Homoscedasticity, and Independent Observations) plus Linear Relationship between Response and Covariate, No Potential Outliers, and Homogeneity of Slopes.

For ANCOVA models, a good practice would be to start with the three ANCOVA-specific assumptions. These assumptions, if violated, are more easily fixed BUT fixing them will change the model and therefore the residuals, impacting the assumptions dealing with the residuals. Thus, by checking the ANCOVA specific assumptions first, we can save time in the long run.

5.1 Linear Relationship between Response and Covariate

The first assumption that I like to check in ANCOVA is that of the linear relationship between the covariate and the response. The best approach here is to look at a scatter plot.

```r
# Scatter plot of hours of pain and hours spent keyboarding ----
# Notice we're just using the data, not the model
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd
  )
) +
  geom_point(size = 2) +
  geom_smooth( # Adds a smoother function's graph
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm", # Fit a Linear Model
    formula = y ~ x, # Specifies the form of the "linear" model
    color = "black",
    linetype = "dashed",
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain")
```

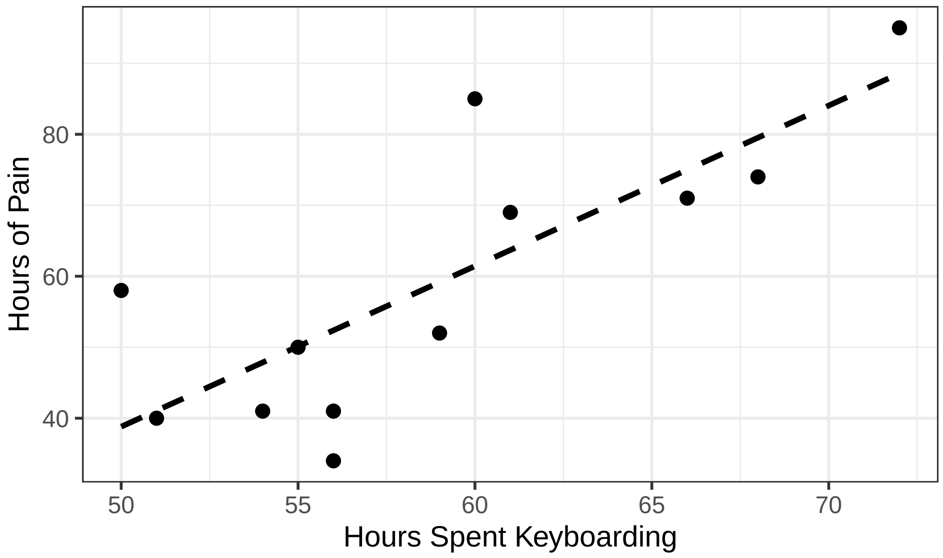
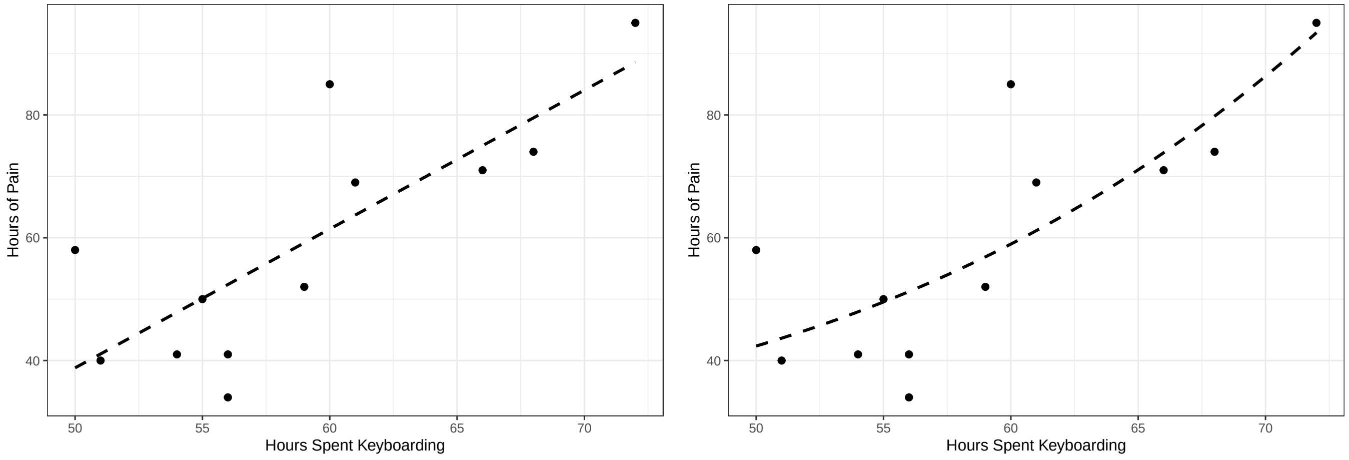


Figure 2: Hours of Pain vs Hours Spent Keyboarding

Figure 2 shows the scatter plot of hours of pain experienced by the hours spent keyboarding. Notice that I've NOT incorporated the type of keyboard each person got assigned. I want to get a clear sense of the relationship between the response and the covariate. If I add too many other attributes to this plot (e.g., the factor), I might overlook what I need to attend to.

Ultimately, we're looking to see whether there is a linear relationship between the response and the covariate. That is to say, is the a constant rate of change between them. If we are plotting in the standard Cartesian coordinate system without scale transformations (e.g., log), then we should be able to sketch a straight line. In Figure 2, I added the plot of the estimated linear regression model of the response (hours of pain) with respect to the covariate (hours spent keyboarding). This graph appears as the dashed, black line. Does this line appear to mimic what we see in the scatter plot?

From Figure 2 I would say that we have a linear relationship between the response and our covariate. While not the best of fits, the graph of the fitted linear model appears to coincide with the trend (i.e., the generalized relationship) between the response (hours of pain) and the covariate (time spent keyboarding). If the graph of the fitted model didn't, we could explore other models by tweaking the `formula` argument. For instance, we could look a quartic (fourth power) relationship (see Figure 3).



(a) Failed Quartic: formula = y ∼ x∧4

(b) Actual Quartic: formula = y ∼ I(x∧4)

Figure 3: Examples of Alternative Models

Notice that in Figure 3 we can see the impact of the "As Is" operator of R (`I`) on formula statements. Figure 3a is a carbon copy of our original linear relationship (i.e., `formula = y ~ x`). However, using the "As Is" operator in Figure 3b will force R to fit the appropriate model. If we were to see a nonlinear relationship (e.g., a quadratic relationship), then I would go back to the model step and change the formula to `y ~  I(hrs.kbd^2) + kbd.type` (don't forget to update the `interactionCheck` too) to linearize the response-covariate relationship. (You can find the code for Figure 3 in the Code Appendix; this code demonstrates the 'As Is' operator inside formula statements.)

5.1.1 Your Turn

Create a scatter plot for the Lego Set Price study between the response and the covariate. Do we have a linear relationship between these quantities?

5.2 Checking Potential Outliers

While we always want to explore our data for potential outliers, ANCOVA models are less resistant to their influence than a typical balanced ANOVA model.

To explore for potential outliers, you will want to work in a systematic way, moving through your quantitative attributes one at a time. I recommend using Box plots and the various univariate Rules of Thumb to check.

```r
# Demo Code for box plot with how to adjust the outlier detection
## Keyboarding Study
ggplot(
  data = keyboardingData,
  mapping = aes(x = hrs.pain)
) +
  geom_boxplot(
    coef = 1.5 # Use this to adjust outlier detection; coef*IQR
  ) +
  theme_void() +
  xlab("Hours of Pain") +
  theme(
    axis.line.x = element_line(),
    axis.text.x = element_text(size = 12),
    axis.title.x = element_text(size = 12)
  )
```

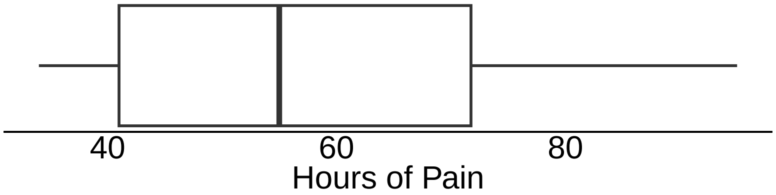


Figure 4: Box Plot of Hours of Pain

Once you complete all of the univariate checks for potential outliers, we can move on to *potential multivariate outliers*. A potential multivariate outlier is a case whose values appear to be inconsistent with the underlying structure of the rest of the collection. This is different from potential univariate outliers in that univariate are case's whose values tend to be extremes. A potential multivariate outlier may not be an outlier along any univariate check.

To help us check for potential multivariate outliers, we are going to make use of the `rstatix` package's `mahalanobis_distance` function.

```
# Demo Code for Detecting Multivariate Outliers ----
## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(keyboardingData)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
  outlierDetection,
  factor = keyboardingData$kbd.type
)

## Step 3: Make a scatter plot
ggplot(
  data = outlierDetection,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    shape = is.outlier,
    color = factor
  )
) +
  geom_point(size = 3) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard",
    shape = "Potential Outlier"
  )
```

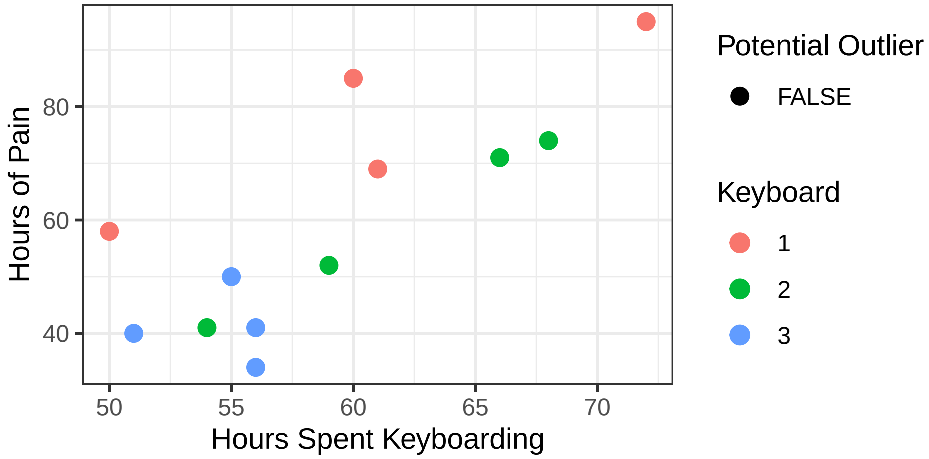


Figure 5: Potential Multivariate Outliers in Keyboarding Pain Study

In Figure 5, the shape of the points will reflect a `FALSE` or `TRUE` answer to the statement “This observation is a potential outlier.” Thus, `TRUE` would indicate that we have a potential outlier. In our case, we have all `FALSE` points, thus we do not have a potential multivariate outliers to be concerned about.

Let’s explore the issue of potential multivariate outliers in the Lego Set Prices study. Figure 6 shows the scatter plot showing the results of the calculating the Mahalanobis distances for each point. Notice that unlike Figure 5, we have two different shapes for the Potential Outlier.

```r
# Demo Code for Detecting Multivariate Outliers ----
## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(legoData)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
  outlierDetection,
  factor = legoData$collection
)

## Step 3: Make a scatter plot
ggplot(
  data = outlierDetection,
  mapping = aes(
    y = price,
    x = numParts,
    shape = is.outlier,
    color = factor
  )
) +
  geom_point(size = 1) +
  theme_bw() +
  labs(
    x = "Number of Pieces",
    y = "Price ($US)",
    shape = "Potential Outlier",
    color = "Collection"
  ) +
  scale_color_manual(values = boastUtils::psuPalette) +
  theme( # Adjust legend titles so they both appear
    legend.spacing.y = unit(-1, "points")
  ) +
  guides(
    color = guide_legend(byrow = TRUE),
    shape = guide_legend(byrow = TRUE)
  )
```

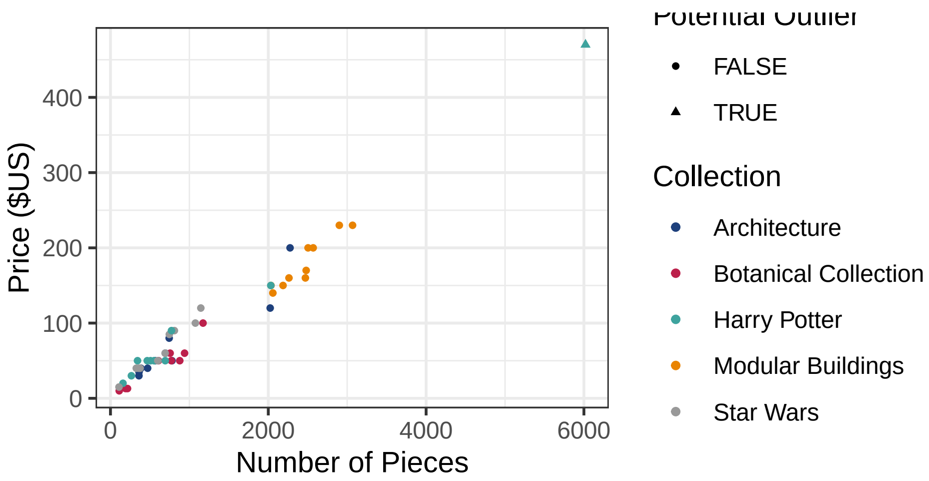


Figure 6: Potential Multivariate Outliers in the Lego Set Price Study

5.2.1 Dealing with Potential [Multivariate] Outliers

If we had potential multivariate outliers, we would want to investigate why they are potential outliers. For example, is there a data entry error or some other mistake for those cases? If we can fix those errors, we should and then re-run our analyses. If we can't fix those errors, then we will want to consider removing those cases. If we do not know, then we should run the model with and without these cases and compare the results.

In the Lego Set Price study, our potential univariate and multivariate outliers are *not* the result of data entry errors; they are legitimate values. Thus, we might want to run a second set of ANCOVA models omitting the one set that was flagged as a potential multivariate outlier to see what, if any, changes to our model occur.

5.2.2 Your Turn

Which Lego sets show up as potential univariate outliers? Which Lego set was flagged as the potential multivariate outlier?

5.3 Homogeneity of Slopes

Standard ANCOVA models assume that the model is for separate intercepts/parallel lines for the covariate. We often refer to this as homogeneity of slopes. More accurately, this is the assumption that the rate of change of the response with respect to the covariate is invariant (unchanging) regarding the factor(s) (and block). Given that we're fitting a linear model between the response and the covariate, this means that we should have identical constant rates of change for each level of our factor(s). In other words, there is no interaction between our covariate and our factor(s).

There are two ways that we can assess this assumption: looking at how well a plot for the separate intercepts model fits our data and to do an informal test of the interaction between the covariate and our factor(s).

5.3.1 Using a Plot

One route we can take is to build a plot of the response (amount of time in pain) by the covariate (time spent keyboarding) and the type of keyboard used. This is similar to what we did to check for the linear relationship between the response and the covariate. However, there are two distinct differences. First, we're going to explicitly include the factor (type of keyboard) to our plot. Second, we're going to impose three regression lines–one for each kind of keyboard–that only differ in their intercepts. If these lines appear to match up with the data reasonably well, we can be convinced that we have homogeneity of slopes. If they do not, then we might need to re-think this assumption and potentially fit a different ANCOVA model (see the final section of this guide).

```
# Demo Code for Assessing Homogeneity of Slopes in Keyboarding Pain Study ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm", # See notes below
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```

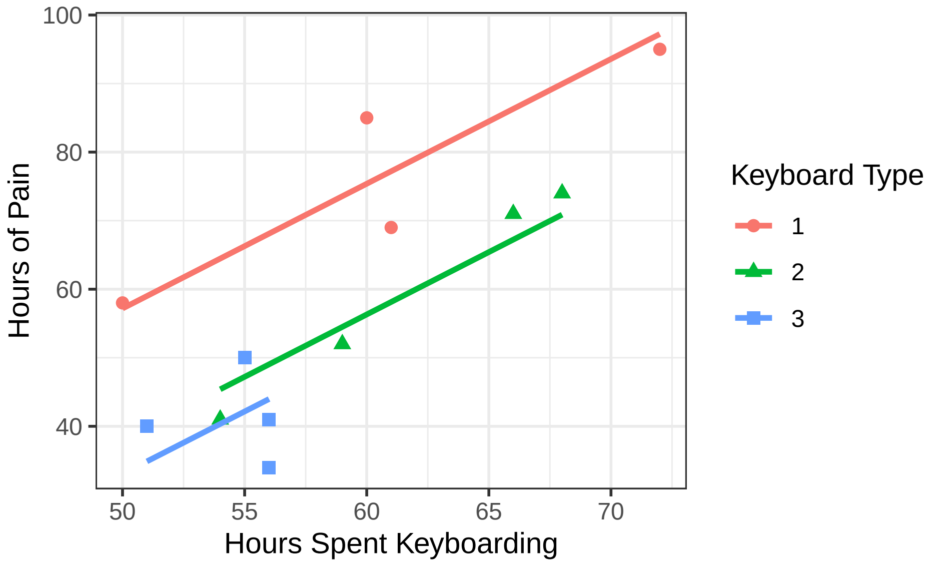


Figure 7: Homogeneity of Slopes for Keyboarding Pain Study

In the code for the plot, you'll notice that we used the `geom_smooth` geometry to add the three lines. For these lines we want to use a linear model (`method = "lm"`) and define the formula `y ~ x`. The most important aspect was that within `geom_smooth` we replaced the observed pain duration with the predicted duration from our ANCOVA model: `predict(keyboardModel)`.

We treat Figure 7 just as we would an interaction plot from our exploration of Factorial designs or Block designs. We are wanting to see that all of the regression lines have essentially the same slope as each other. Remember, we don't have to be perfectly parallel to account for the reality of dealing with real data.

I recommend checking out the section at the end of this guide to learn more about the five different ANCOVA models.

5.3.2 Informal Test of Interaction

The second approach we can take is to do an informal test of the interaction term. Recall that we formed a second ANCOVA model called `interactionCheck`. We will want to check this set of results using Type III *Sum of Squares*.

To get the information, we'll use the `car` package's `Anova` function to get the Type III *Sum of Squares*:

```
# Demo Code for INFORMAL Interaction Check ----
## Use the car package
car::Anova(
  mod = interactionCheck,
  type = 3
)
```

```
## Anova Table (Type III tests)
##
## Response: hrs.pain
##                   Sum Sq Df F value  Pr(>F)
## (Intercept)       18.807  1  0.3881 0.55622
## hrs.kbd          217.487  1  4.4880 0.07845 .
## kbd.type         147.651  2  1.5235 0.29171
## hrs.kbd:kbd.type 120.271  2  1.2410 0.35398
## Residuals        290.756  6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To do this informal check, you will want to keep your Unusualness Threshold and Type I Error Risk in mind. While we are going to look at an ANOVA table, we do not report this table. At most, you'll extract the information you need an report that in a sentence or two.

We only want to focus on the interaction term of hours spent keyboarding and the type of keyboard (i.e., `hrs.kbd:kbd.type`). The p-value for this term is approximately 0.35. This is well beyond any UT you're allowed to use in this course, thus, we will fail to reject the null hypothesis. That is to say, the interaction between the covariate and the factor is not statistically important. This tells us that we should not anticipate different rates of change of pain duration with respect to time spent keyboarding for different kinds of keyboards.

5.3.3 Your Turn

Create an interaction plot for the Lego Set Price study. What would you decide about the homogeneity of slopes there? How do your plot and your decision there coincide (or not) with an informal test of the interaction of the covariate and factor?

5.4 Gaussian Residuals

Use a QQ plot like usual:

```
# Demo Code for QQ plot in Keyboarding Pain Study ----
car::qqPlot(
  x = residuals(keyboardModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (hours)"
)
```

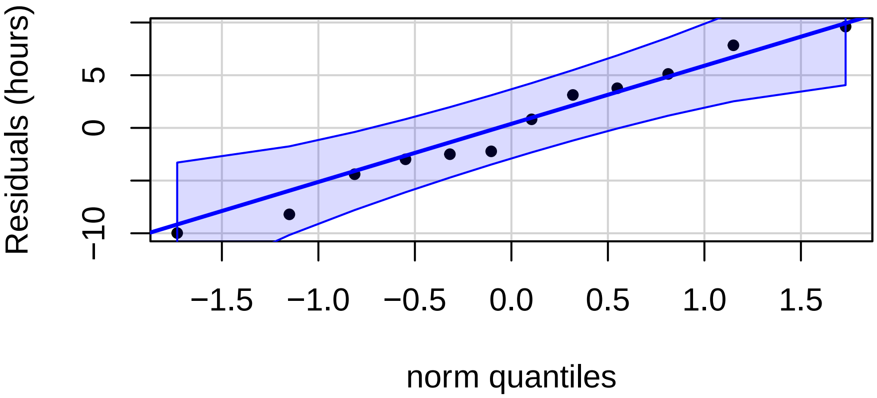


Figure 8: QQ Plot for Residuals in Keyboarding Pain Study

There is very little to be concerned about in our QQ plot (Figure 8); we will go ahead and proceed as if our residuals follow a Gaussian distribution.

5.4.1 Your Turn

Create the QQ Plot for the residuals from the Lego Set Price study.

5.5 Homoscedasticity

Given that we have more than just a single factor, we need to look at a Tukey-Anscombe plot rather than just a strip chart.

```
# Demo Code for Tukey-Anscombe Plot for the Keyboarding Pain Study ----
ggplot(
  data = data.frame(
    residuals = residuals(keyboardModel),
    fitted = fitted.values(keyboardModel)
  ),
  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,
    linewidth = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (hours)") +
  ylab("Residuals (hours)")
```

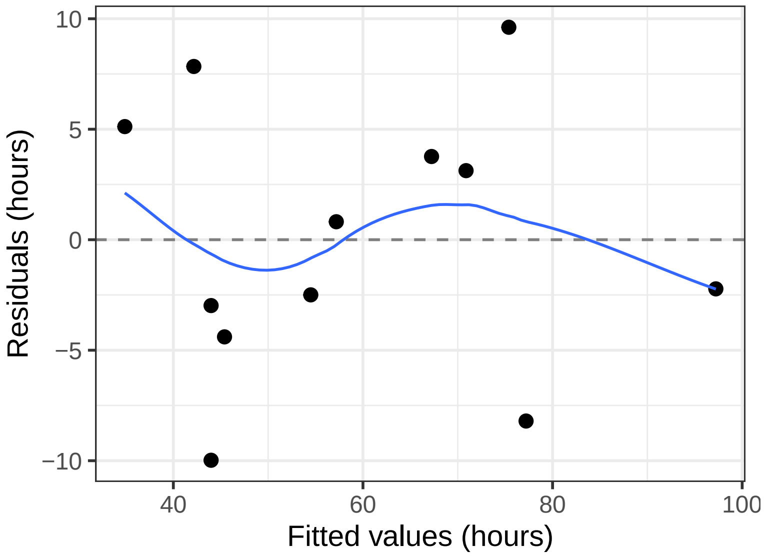


Figure 9: Tukey-Anscombe Plot for Keyboarding Pain Study

I'm a bit hesitant for the homoscedasticity assumption given the curvy nature of the line (see Figure 9). However, I don't get the sense of any pattern to the residuals. So I might say that homoscedasticity might be questionable. I would anticipate that I would go back and look at box plots and descriptive statistics by my factor to see if there might be any numeric evidence of heteroscedasticity.

5.5.1 Your Turn

Construct the Tukey-Anscombe plot for the Lego Set Price study. What is your assessment of the homoscedasticity assumption in that data?

5.6 Independence of Observations

Unfortunately, we don't know measurement order so index plots are not going to be useful here. However, we can think through the study design and reach the decision that we have independent observations.

5.6.1 Your Turn

For both the Keyboarding Pain and Lego Set Price studies, what would you be looking for to assess Independence of Observations? What assessments would you make in each study?

5.7 Additional Checks for ANCOVA

We have more assumptions to check in an ANCOVA situation than an ANOVA situation. Keep in mind that if you introduce a Block, then the assumptions associated with those models will get imported into your situation as well. That is, the block doesn't interact with factors. The approaches you used previously (as well as much of the code) remains the same.

5.8 Multiple Covariates

If you are planning on using multiple covariates in your model, then there is one extra assumption that you need to check. We assume that that all of our covariates are not highly correlated with one another; that is, we do not want to have multicollinearity amongst our covariates.

There are two tools you can use to check out the possibility of multicolinearity. The first is the Generalized Variance Inflation Factor and the second is the Squared Multiple Correlation.

For the Generalized Variance Inflation Factor (GVIF), you can use the `car::vif` function on the ANCOVA model. While, you'll get GVIF values for all terms in the model, we can focus on the covariates. Look at the column labelled `GVIF^(1/(2*Df))` and square the values there. This will give a value that you can compare to the typical VIF Rules of Thumb for 5 and 10.

For the Squared Multiple Correlation (SMC), we can make use of the function `psych::smc`. For this function, you'll use all of the covariate columns from the data frame as the input. You'll get back a listing of SMC values for each covariate. The Rule of Thumb here is that values beyond 0.5 indicate that we might have redundancy (multicollinearity).

5.8.1 Example

As a quick example, we'll draw upon the Palmer Penguins data set from the package `palmerpenguins`. Suppose that we are investigating the impact of penguin species on body mass with three covariates: bill length, bill depth, and flipper length, all measured in millimeters.

We would set up the ANCOVA model in `R` as

```r
# Demo Code for checking multicollinearity between multiple covariates ----
## Palmer Penguins
penguins <- palmerpenguins::penguins

penguinModel <- aov(
  formula = body_mass_g ~ species + bill_length_mm + bill_depth_mm + flipper_length_mm,
  data = penguins
)

## Getting Generalized Variance Inflation Factors
gvifs <- as.data.frame(car::vif(penguinModel))
gvifs$squared <- gvifs$"GVIF^(1/(2*Df))"^2
gvifs
```

```
##                        GVIF Df GVIF^(1/(2*Df))  squared
## species           34.117194  2        2.416815 5.840993
## bill_length_mm     5.226426  1        2.286138 5.226426
## bill_depth_mm      4.799098  1        2.190684 4.799098
## flipper_length_mm  6.516234  1        2.552692 6.516234
```

```r
## Getting Squared Multiple Correlations
psych::smc(penguins[,c("bill_length_mm", "bill_depth_mm", "flipper_length_mm")])
```

```
##    bill_length_mm     bill_depth_mm flipper_length_mm
##         0.4638329         0.3793799         0.6259358
```

Notice that bill length and flipper length both exceed 5 in the squared column (ignore the factor–species); this indicates that there may be some correlation between the covariates. For the SMC values, we can see that flipper length is above 0.5, indicating this covariate may be redundant given the others. I would consider fitting the model without flipper length and seeing what happens.

6 Reporting Results

Results for ANCOVA models are a bit mixed: some people will only report Omnibus Results, others will proceed to looking at Post Hoc analyses. This is to say, there's not an overriding drive to automatically conduct Post Hoc analysis like there is for a CRD/One-way layout or even Factorial designs. My suggestion is to let your research questions guide you.

6.1 Omnibus Results

In this particular situation, we have a **balanced** design, thus we do not need to worry about different types of Sums of Squares.

```
# Demo Code for Omnibus Test/Modern ANCOVA Table ----
## Note: Type I Sums of Squares is default
parameters::model_parameters(
  model = keyboardModel,
  effectsize_type = c("eta", "omega", "epsilon")
) %>%
  dplyr::mutate( #Fixing the Parameter (Source) Column's values
    Parameter = dplyr::case_when(
      Parameter == "hrs.kbd" ~ "Hours Spent Keyboarding",
      Parameter == "kbd.type" ~ "Keyboard Type",
      TRUE ~ Parameter
    )
  ) %>%
  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",
                  "Partial Eta Sq.", "Partial Omega Sq.", "Partial Epsilon Sq."),
    caption = "ANOVA Table for Keyboarding Study",
    align = c('l',rep('c',8)),
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("scale_down", "HOLD_position")
  )
```

Table 1: ANOVA Table for Keyboarding Study

Source	SS	df	MS	F	p-value	Partial Eta Sq.	Partial Omega Sq.	Partial Epsilon Sq.
Hours Spent Keyboarding	2598.8209	1	2598.8209	50.5819	0.0001	0.8634	0.8051	0.8464
Keyboard Type	1195.8180	2	597.9090	11.6373	0.0043	0.7442	0.6394	0.6803
Residuals	411.0278	8	51.3785					

While we don't necessarily want to ignore the covariate's row in the table (like we do for a block), we don't want to get caught up in that row. That is, don't forget that our focus is on our factor(s). Our interpretations remain the same as before. For example,

> The type of keyboard assigned to the patient accounts of about 11.6 times as much variation in pain duration as our residuals/what's left unexplained ($F(2, 8) = 11.64$), even when we account for how much time the patient spends keyboarding. Under the null hypothesis that keyboard has no impact on pain duration, we would only anticipate such a result less than 1% of the time ($p = 0.0043$). The type of keyboard accounts for a large poroportion of the variation in pain durations with around 64% (see Table 1).

6.1.1 Your Turn

Create two (2) omnibus results tables for the Lego Set Price study: one using the full data set and one dropping the potential multivariate outlier. What would you come up with as a paragraph describing the results for both tables? Compare the results; what was the impact of the potential multivariate outlier (if any) on your decisions?

6.2 Point Estimates

The topic of point estimates in ANCOVA is a bit complicated as there are ***two*** sets of point estimates. We can look at the straight up estimates from the model ("raw") and we can also look at covariate adjusted estimates.

6.2.1 "Raw" Estimates

To get the un-adjusted or "raw" point estimates, we will make use of the `dummy.coef` function as we've done in the past.

```r
# Demo Code of getting un-adjusted point estimates ----
# Keyboarding Model
rawPointEst <- dummy.coef(keyboardModel)
rawPointEst <- unlist(rawPointEst)
names(rawPointEst) <- c( # Be sure you know the order
  "Grand Mean",
  "Hours Spent Keyboarding",
  paste("Keyboard Type", levels(keyboardingData$kbd.type))
)

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

Table 2: Unadjusted Point Estimates from the Keyboarding Study

	Estimate
Grand Mean	-48.21
Hours Spent Keyboarding	1.82
Keyboard Type 1	14.40
Keyboard Type 2	-4.67
Keyboard Type 3	-9.73

Something that should immediately stand out to you is the point estimate for the $GSAM$ is negative. However, we know from our exploration of the data that $GSAM(\mathcal{Y}) \approx 59.17$ hrs/person. These are two very different values!

What's the deal? These "raw" or "un-adjusted" point estimates are incorporating the effect of the covariate. Thus, to account for the covariate's impact on the response, we have to make some adjustments. For the $GSAM$ this works through the following formula: $GSAM(\mathcal{Y}) = GSAM_{Raw}(\mathcal{Y}) + \widehat{\beta} \cdot SAM(\mathcal{X})$ where $\mathcal{Y}$ represents the collection pain duration (hours), $\mathcal{X}$ represents the collection of hours spent keyboarding, and with $\widehat{\beta}$ representing the estimated coefficient for the covariate. For our data, this translates to

$$59.17 \approx -48.21 + 1.82 \cdot 59$$

6.2.2 Adjusted Point Estimates

We can also get covariate adjusted point estimates for our factor's levels. Now, there is an important note I need to stress here. The point estimates for the factor's levels we previously looked at are for the *factor effects* but the adjusted point estimates are for the *factor means*. That is, we will be getting $\widehat{\mu_i} = \widehat{\mu_{\bullet\bullet}} + \widehat{\alpha_i}$ rather than just $\widehat{\alpha_i}$. The best way to get the covariate adjusted point estimates for the factor means is to use the `emmeans` package.

The `adjust` argument of `emmeans` allows for the following values `"bonferroni"`, `"tukey"`, `"scheffe"`,`"sidak"`, `"holm"`, `"hochberg"`, `"hommel"`, `"BH"` (Benjamini and Hochberg), and `"fdr"`. These will produce the appropriately adjusted confidence intervals (for the `$emmeans` objects) and adjusted p-values (for the `$contrasts` objects).

```r
# Demo code for getting adjusted factor means ----
## These means are also called "marginal means"
## Keyboarding
emmOutKey <- emmeans::emmeans(
  object = keyboardModel,
  specs = pairwise ~ kbd.type,
  adjust = "tukey",
  level = 0.9
)

## Point Estimates
as.data.frame(emmOutKey$emmeans) %>%
  knitr::kable(
    digits = 4,
    col.names = c("Keyboard Type", "Marginal Mean","SE", "DF",
                  "Lower Bound","Upper Bound"),
    caption = "Marginal Means-Tukey 90\\% Adjustment",
    align = c("l", rep("c", 5)),
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```

Table 3: Marginal Means-Tukey 90% Adjustment

Keyboard Type	Marginal Mean	SE	DF	Lower Bound	Upper Bound
1	73.5652	3.6406	8	66.7953	80.3350
2	54.4953	3.7223	8	47.5736	61.4170
3	49.4395	3.9434	8	42.1066	56.7725

6.2.3 Interpreting Point Estimates

Regardless of whether you're interpreting the "raw"/un-adjusted point estimates or the adjusted estimates; whether factor effects, factor means, or the coefficient for the covariate, all of these point estimates are *rates of change* and should be interpreted as such.

- *GSAM* of $\widehat{\mu_{\bullet\bullet}} = 59.17$ pain hours per person [covariate adjusted]
  - Our participants accumulated a total number of pain hours 59.17 times as large as the sample size.
- RoC of Pain Duration w.r.t. Hours Keyboarding, $\widehat{\beta} = 1.82$ hours per hour
  - Across all treatments and participants, the change in the amount of time a person felt pain (in hours) is approximately 1.82 times as large as the corresponding change in how long they keyboarded (in hours).
- Keyboard Type 1 Effect, $\widehat{\alpha_1} = 14.40$ pain hours per person
  - After accounting for the impact of how long a person spent keyboarding, our group of people using Keyboard Type 1 accumulated an additional 14.4 pain hours per person beyond the general baseline expectations.
- Keyboard Type 1 Mean, $\widehat{\mu_1} = 73.57$ pain hours per person
  - After accounting for the impact of how long a person spent keyboarding, our Keyboard Type 1 treatment group accumulated a total number of pain hours that was 73.57 times as large as the size of that treatment group.

6.2.4 Your Turn

Get and interpret the *adjusted* point estimates for the Lego Set Price study.

6.3 Post Hoc Analysis

For Post Hoc Analysis, we'll follow the same paths was what we've done in the past: explore which pairs of factor levels are statistically different from one another. And for those that are, what's the practical significance of those differences (i.e., what's the size of the effect)?

6.3.1 Pairwise Comparisons

Just as with Factorial models, we will use the `emmeans` package here as well. Since I've already stored the output, I don't need to call `emmeans` a second time. Remember that we ask for the `constrasts` subobject from our previously stored `emmOutKey` object to get pairwise comparisons.

```r
# Demo code for Pairwise Comparisons in ANCOVA ----
## Keyboarding Study
## Notice that I'm re-using the emmOutKey object I created earlier
as.data.frame(emmOutKey$contrasts) %>%
  knitr::kable(
    digits = 4,
    col.names = c("Comparison", "Difference","SE", "DF",
                  "t Statistic","p-value"),
    caption = "Marginal Means-Tukey 90\\% Adjustment",
    align = c("l", rep("c", 5)),
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("HOLD_position")
  )
```

Table 4: Marginal Means-Tukey 90% Adjustment

Comparison	Difference	SE	DF	t Statistic	p-value
kbd.type1 - kbd.type2	19.0699	5.0816	8	3.7527	0.0138
kbd.type1 - kbd.type3	24.1257	5.5596	8	4.3395	0.0062
kbd.type2 - kbd.type3	5.0558	5.7195	8	0.8839	0.6647

For more complicated ANCOVA models (e.g., multiple factors), we can use the techniques previously mentioned (including for contrasts). I want to note here that these pairwise comparisons are all made using the value of the *Sample Arithmetic Mean* of the covariate. In the Keyboarding Pain study, that value is 59 hrs/person. Thus, all of these pairwise comparison estimates used 59 hrs/person. If you use `specs = pairwise ~ kbd.type | hrs.kbd`, you'll get an extra line that says `hrs.kbd = 59:` at the top of your raw output; the rest of the table is identical to what we've already seen.

6.3.2 Effect Sizes

You will want to use the `emmeans` package for the effect sizes as well (the `anova.PostHoc` function won't account for the covariate).

```
# Demo code for post hoc (pairwise) effect sizes ----
## Keyboarding Study
as.data.frame(
  eff_size(
    object = emmOutKey,
    sigma = sigma(keyboardModel),
    edf = df.residual(keyboardModel)
  )
) %>%
  dplyr::mutate(
    ps = probSup(effect.size),
    .after = effect.size
  ) %>%
  dplyr::select(contrast, effect.size, ps) %>%
  knitr::kable(
    digits = 3,
    col.names = c("Keyboard Comparison", "Cohen's d", "Probability of Superiority"),
    align = "lccc",
    caption = "Effect Sizes for Keyboard Type",
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 5: Effect Sizes for Keyboard Type

Keyboard Comparison	Cohen's d	Probability of Superiority
(kbd.type1 - kbd.type2)	2.660	0.970
(kbd.type1 - kbd.type3)	3.366	0.991
(kbd.type2 - kbd.type3)	0.705	0.691

6.3.3 Your Turn

Get creative and come up with not only the pairwise comparisons but also two contrasts for the Lego Set Price study.

7 Choosing Between the Five ANCOVA Models

There are actually five (5) different ANCOVA models. Each one is a slight variation and is applicable in different situations. The choice between them comes down to what you're trying to study. The choice that you make has impacts on your designs including on the *degrees of freedom* within your model. I've provide some general notation for calculating the *Degrees of Freedom* used by each model for a One-way ANCOVA model.

7.1 Model 1: No Effects–Constant [Grand] Mean

This is the null model for ANCOVA: our factor(s) and our covariate(s) have absolutely no impact on the response. In this model, we have only used 1 *Degree of Freedom* as we only have to estimate the value of the *Grand Mean*. If this situation were true, then the blue horizontal line in Figure 10 would accurately describe the data.

```
# Demo code for the Constant Mean ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = mean(keyboardingData$hrs.pain),
    color = "blue"
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )
```

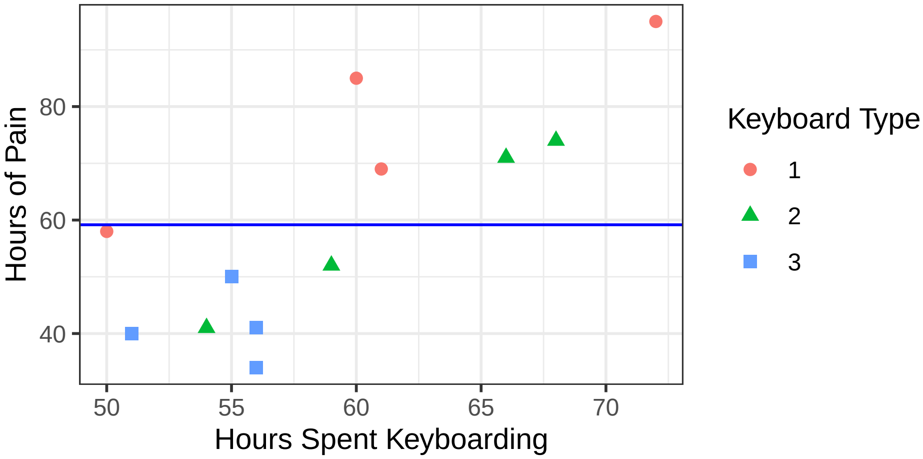


Figure 10: Graph of the Constant Mean ANCOVA Model

7.2 Model 2: Single Line

The Single Line ANCOVA model posits that while the factor(s) have no impact on the response, the covariate does. Thus, we have a single line with a non-zero rate of change of the response with respect to the covariate. Again, if this were the case, the plotted line in Figure 11 would do a great job in describing the data. This model uses 2 *Degrees of Freedom* as we are estimating the *Grand Mean* and the Rate of Change of the response with respect to the covariate.

```r
# Demo Code for the Single Line ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE, # Tells smoother to ignore the factor
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ x,
    se = FALSE
    ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )
```

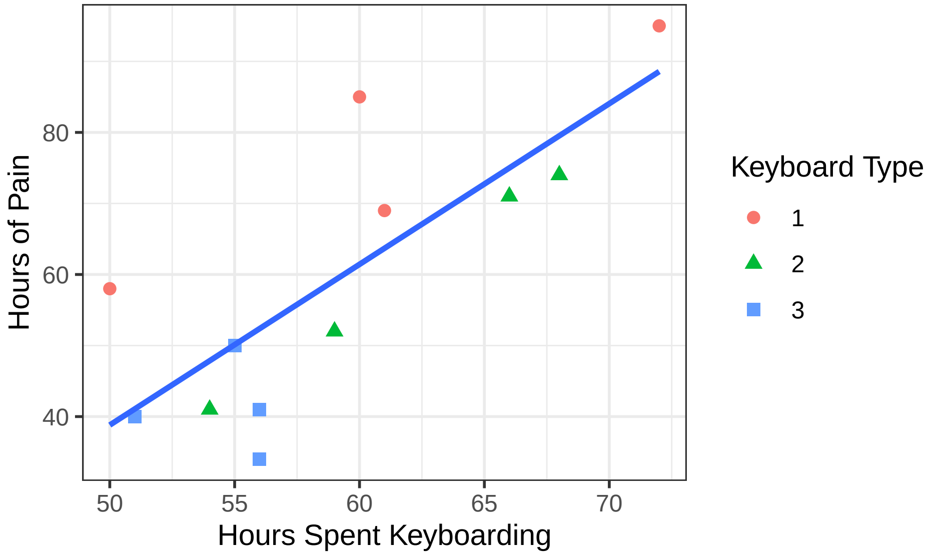


Figure 11: Graph of the Single Line ANCOVA Model

7.3 Model 3: Separate Intercepts/Parallel Lines (“Standard ANCOVA”)

This is the standard ANCOVA model and the one that we most often want to draw upon. (This is the model that we are using in this course.) Here, we are anticipating that both the covariate and the factor(s) have an impact on the response. The way we think that they do so is that the factor level effects act as vertical shifts of the linear relationship between the response and the covariate. That is, we should have a set of parallel lines whose intercepts differ thanks to the factor. In this model, we use $1 + 1 + (g - 1) = g + 1$ *Degrees of Freedom*. We are estimating the *Grand Mean*, the Rate of Change of the Response w.r.t. the Covariate, and the g-1 factor level effects which translate to the intercepts.

Again, if this is the case, the g parallel lines we plot should accurately describe the data in Figure 12.

```r
# Demo Code for the Separate Intercepts/Parallel Lines ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )
```

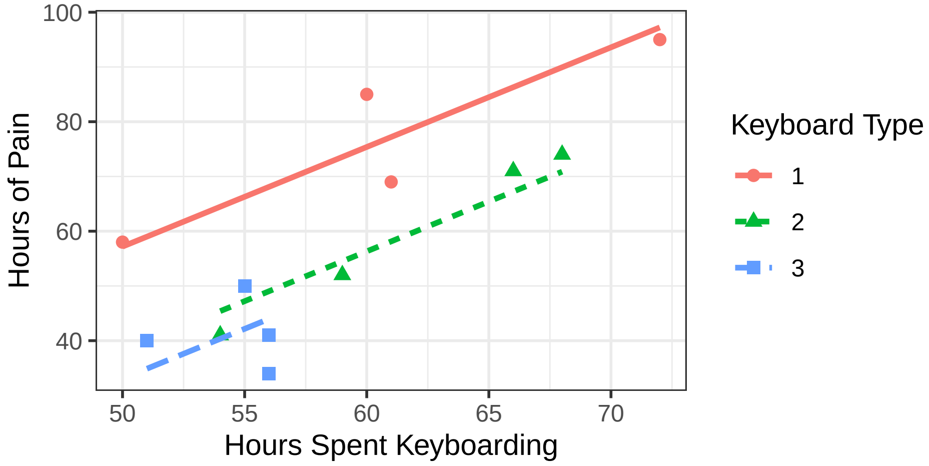


Figure 12: Graph of the Separate Intercepts/Parallel Lines ANCOVA Model

7.4 Model 4: Separate Slopes

For this version of the ANCOVA model, we opt to allow the Rate of Change of the Response w.r.t. the Covariate to change from group to group while requiring all models to share a common point (e.g., a common point of intersection). The choice of common point should be guided by your understanding of the context and be somewhat sensible. For example, we could specify that all of the models should intersect at 0 hours of keyboarding. Alternatively, we could say that all of the models should intersect at the value of the *SAM* of hours spent keyboarding (~59 hrs/person). The choice is ours but will impact the estimates for the various $\widehat{\beta}_i$ values. In this model, we use $1+1+(g-1)=g+1$ *Degrees of Freedom*. We are estimating the *Grand Mean*, the point of intersection, and the g-1 Rates of Change of the Response w.r.t. the Covariate by Factor Level.

As before, we would look at the lines in Figure 13 to see how well they describe our data.

```r
# Demo Code for the Separate Slopes ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x + 0, # Require all lines to intersect at x = 0
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )
```

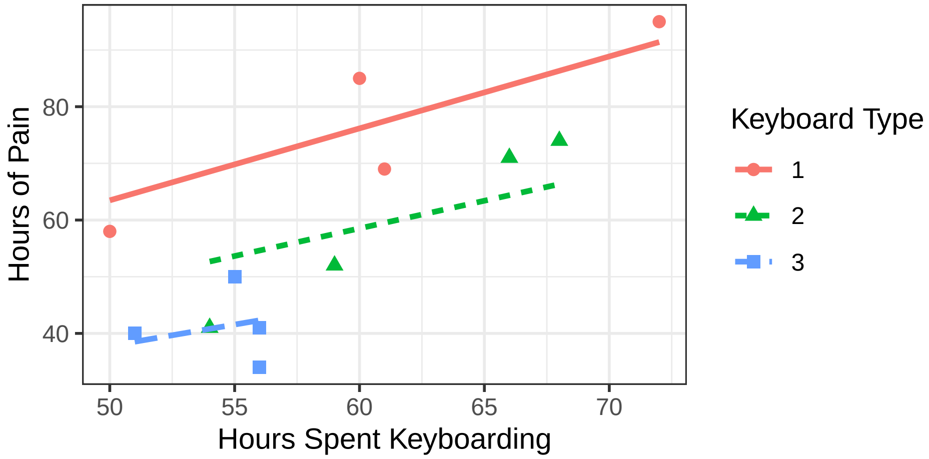


Figure 13: Graph of the Separate Slopes ANCOVA Model

7.5 Model 5: Separate Lines

The last ANCOVA model is that of separate lines; here is where we believe that there is a theoretically important reason to have the factor(s) and covariate(s) interact. We could see this with a statistically significant interaction term. In this model, we use $1 + 1 + (g - 1) + (g - 1) = 2g$ *Degrees of Freedom*. We estimate the *Grand Mean*, the various Rates of Change of the response w.r.t. the Covariate by Factor level, and the factor effects (the intercepts).

Again, we're looking to see how well the lines describe the data in Figure 14.

```
# Demo code for the Separate Lines ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
   theme(
     legend.position = "right"
   )
```

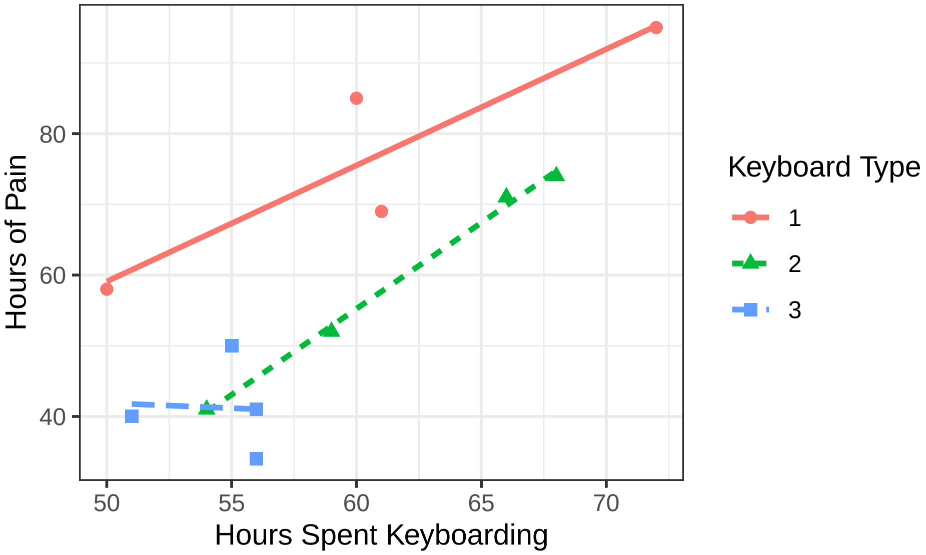


Figure 14: Graph of the Separate Lines ANCOVA Model

7.6 Making a Choice

As you look through Figures 10–14, you'll notice that the models do better and worse jobs at describing our data. Hopefully, we will all agree that the Constant Mean ANCOVA model (Figure 10), that is our Null Model, does a terrible job describing our data. While an improvement on the Constant Mean ANCOVA model, the Single Line ANCOVA model, still leaves a bit to be desired for describing our data (Figure 11).

The next three models visually do a better job in describing our data. We generally want to use the simplest model possible. We can explore whether the interaction of factor and covariate is important both theoretically and statistically. Further, we can look at whether there are any improvements to be had by looking at a more complicated model. You can learn more about these models in Section 17.3 of Oehlert.

8 Code Appendix

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

packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "hasseDiagram", "car",
              "psych", "emmeans", "rstatix")
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", "car",
              "psych", "emmeans", "rstatix")
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 Keyboarding Data ----
keyboardingData <- read.table(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/keyboarding.dat",
  header = TRUE,
  sep = ""
)

# Column Notes
## hrs.pain is our response; hours experiencing pain
## kbd.type is our factor; type/style of keyboard
## hrs.kbd is our covariate; hours spent using the keyboard
### make sure that R is thinking of hrs.kbd as either as num or int

keyboardingData$kbd.type <- as.factor(keyboardingData$kbd.type)

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

```r
legoData$collection <- as.factor(legoData$collection)

# Demo Code for Hasse diagram ----
## Keyboarding Study
modelLabels <- c("1 Relieve Pain 1", "cov Usage Time 1",
                 "3 Keyboard 2", "12 (Volunteers) 8")
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 ANCOVA Models ----
## Keyboarding Study
### Our core model
keyboardModel <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type,
  data = keyboardingData
)

### Model for checking covariate's homogeneity
interactionCheck <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type + hrs.kbd:kbd.type,
  data = keyboardingData
)

## Lego Set Price Study
### Core model
legoModel <- aov(
  formula = price ~ numParts + collection,
  data = legoData
)

### Interacting check
legoCheck <- aov(
  formula = price ~ numParts + collection + numParts:collection,
  data = legoData
)

# Scatter plot of hours of pain and hours spent keyboarding ----
# Notice we're just using the data, not the model
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd
  )
) +
  geom_point(size = 2) +
  geom_smooth( # Adds a smoother function's graph
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm", # Fit a Linear Model
    formula = y ~ x, # Specifies the form of the "linear" model
    color = "black",
    linetype = "dashed",
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain")

# Example Code Showing the As Is operator ----
# Failed Quadratic Model
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ x^4,
    color = "black",
    linetype = "dashed",
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain")

# Actual Quadratic Model
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ I(x^4),
    color = "black",
    linetype = "dashed",
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain")

# Demo Code for box plot with how to adjust the outlier detection
## Keyboarding Study
ggplot(
  data = keyboardingData,
  mapping = aes(x = hrs.pain)
) +
  geom_boxplot(
    coef = 1.5 # Use this to adjust outlier detection; coef*IQR
  ) +
  theme_void() +
  xlab("Hours of Pain") +
  theme(
    axis.line.x = element_line(),
    axis.text.x = element_text(size = 12),
    axis.title.x = element_text(size = 12)
  )

# Demo Code for Detecting Multivariate Outliers ----
## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(keyboardingData)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
  outlierDetection,
  factor = keyboardingData$kbd.type
)

## Step 3: Make a scatter plot
ggplot(
  data = outlierDetection,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    shape = is.outlier,
    color = factor
  )
) +
  geom_point(size = 3) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard",
    shape = "Potential Outlier"
  )

# Demo Code for Detecting Multivariate Outliers ----
## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(legoData)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
  outlierDetection,
  factor = legoData$collection
)

## Step 3: Make a scatter plot
ggplot(
  data = outlierDetection,
  mapping = aes(
    y = price,
    x = numParts,
    shape = is.outlier,
    color = factor
  )
) +
  geom_point(size = 1) +
  theme_bw() +
  labs(
    x = "Number of Pieces",
    y = "Price ($US)",
    shape = "Potential Outlier",
    color = "Collection"
  ) +
  scale_color_manual(values = boastUtils::psuPalette) +
  theme( # Adjust legend titles so they both appear
    legend.spacing.y = unit(-1, "points")
  ) +
  guides(
    color = guide_legend(byrow = TRUE),
    shape = guide_legend(byrow = TRUE)
  )

# Demo Code for Assessing Homogeneity of Slopes in Keyboarding Pain Study ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm", # See notes below
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )

# Demo Code for INFORMAL Interaction Check ----
## Use the car package
```

```
car::Anova(
  mod = interactionCheck,
  type = 3
)

# Demo Code for QQ plot in Keyboarding Pain Study ----
car::qqPlot(
  x = residuals(keyboardModel),
  distribution = "norm",
  envelope = 0.90,
  id = FALSE,
  pch = 20,
  ylab = "Residuals (hours)"
)

# Demo Code for Tukey-Anscombe Plot for the Keyboarding Pain Study ----
ggplot(
  data = data.frame(
    residuals = residuals(keyboardModel),
    fitted = fitted.values(keyboardModel)
  ),
  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,
    linewidth = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (hours)") +
  ylab("Residuals (hours)")

# Demo Code for checking multicollinearity between multiple covariates ----
## Palmer Penguins
penguins <- palmerpenguins::penguins

penguinModel <- aov(
  formula = body_mass_g ~ species + bill_length_mm + bill_depth_mm + flipper_length_mm,
  data = penguins
)

## Getting Generalized Variance Inflation Factors
gvifs <- as.data.frame(car::vif(penguinModel))
gvifs$squared <- gvifs$"GVIF^(1/(2*Df))"^2
gvifs


## Getting Squared Multiple Correlations
```

```
psych::smc(penguins[,c("bill_length_mm", "bill_depth_mm", "flipper_length_mm")])

# Demo Code for Omnibus Test/Modern ANCOVA Table ----
## Note: Type I Sums of Squares is default
parameters::model_parameters(
  model = keyboardModel,
  effectsize_type = c("eta", "omega", "epsilon")
) %>%
  dplyr::mutate( #Fixing the Parameter (Source) Column's values
    Parameter = dplyr::case_when(
      Parameter == "hrs.kbd" ~ "Hours Spent Keyboarding",
      Parameter == "kbd.type" ~ "Keyboard Type",
      TRUE ~ Parameter
    )
  ) %>%
  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",
                  "Partial Eta Sq.", "Partial Omega Sq.", "Partial Epsilon Sq."),
    caption = "ANOVA Table for Keyboarding Study",
    align = c('l',rep('c',8)),
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("scale_down", "HOLD_position")
  )

# Demo Code of getting un-adjusted point estimates ----
# Keyboarding Model
rawPointEst <- dummy.coef(keyboardModel)
rawPointEst <- unlist(rawPointEst)
names(rawPointEst) <- c( # Be sure you know the order
  "Grand Mean",
  "Hours Spent Keyboarding",
  paste("Keyboard Type", levels(keyboardingData$kbd.type))
)

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

```r
  )

# Demo code for getting adjusted factor means ----
## These means are also called "marginal means"
## Keyboarding
emmOutKey <- emmeans::emmeans(
  object = keyboardModel,
  specs = pairwise ~ kbd.type,
  adjust = "tukey",
  level = 0.9
)

## Point Estimates
as.data.frame(emmOutKey$emmeans) %>%
  knitr::kable(
    digits = 4,
    col.names = c("Keyboard Type", "Marginal Mean","SE", "DF",
                  "Lower Bound","Upper Bound"),
    caption = "Marginal Means-Tukey 90\\% Adjustment",
    align = c("l", rep("c", 5)),
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("HOLD_position")
  )

# Demo code for Pairwise Comparisons in ANCOVA ----
## Keyboarding Study
## Notice that I'm re-using the emmOutKey object I created earlier
as.data.frame(emmOutKey$contrasts) %>%
  knitr::kable(
    digits = 4,
    col.names = c("Comparison", "Difference","SE", "DF",
                  "t Statistic","p-value"),
    caption = "Marginal Means-Tukey 90\\% Adjustment",
    align = c("l", rep("c", 5)),
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = c("HOLD_position")
  )

# Demo code for post hoc (pairwise) effect sizes ----
## Keyboarding Study
as.data.frame(
  eff_size(
    object = emmOutKey,
    sigma = sigma(keyboardModel),
    edf = df.residual(keyboardModel)
  )
) %>%
  dplyr::mutate(
    ps = probSup(effect.size),
    .after = effect.size
  ) %>%
  dplyr::select(contrast, effect.size, ps) %>%
  knitr::kable(
    digits = 3,
    col.names = c("Keyboard Comparison", "Cohen's d", "Probability of Superiority"),
    align = "lccc",
    caption = "Effect Sizes for Keyboard Type",
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# Demo code for the Constant Mean ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = mean(keyboardingData$hrs.pain),
    color = "blue"
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )

# Demo Code for the Single Line ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE, # Tells smoother to ignore the factor
```

```r
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ x,
    se = FALSE
    ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )

# Demo Code for the Separate Intercepts/Parallel Lines ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )

# Demo Code for the Separate Slopes ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
```

```
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x + 0, # Require all lines to intersect at x = 0
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
  theme(
    legend.position = "right"
  )

# Demo code for the Separate Lines ANCOVA Model ----
ggplot(
  data = keyboardingData,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type,
    linetype = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  labs(
    x = "Hours Spent Keyboarding",
    y = "Hours of Pain",
    color = "Keyboard Type",
    shape = "Keyboard Type",
    linetype = "Keyboard Type"
  ) +
   theme(
     legend.position = "right"
   )
```