ANCOVA

Neil J. Hatfield

4/16/2021

In this tutorial, we are going to explore Analysis of Covariance (ANCOVA) in R.

New Package

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.

ANCOVA Context

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.

Examine the Hasse Diagram

There are **TWO** ways that you could make the Hasse diagram in this situation; the choice is yours on which to use.

Hasse Diagram App

The first route is what I programmed into the Hasse diagram app: this will treat the covariate(s) as being the same level as your main effect(s).

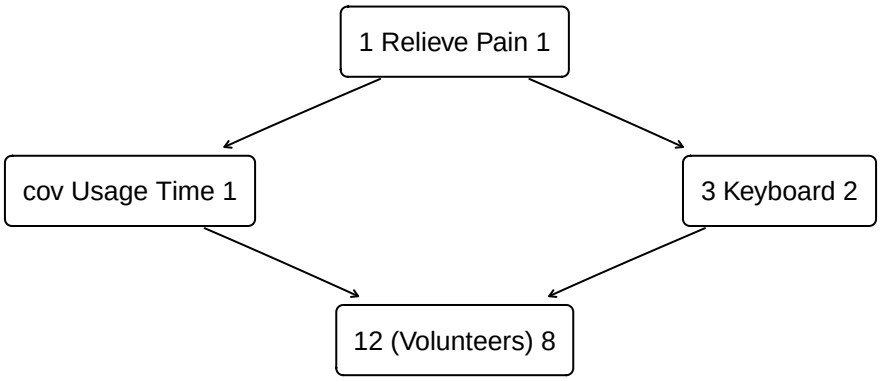


Figure 1: Hasse Diagram for Keyboarding Study-Version 1

Nested–Customized

The second route you can take is to nest the covariates inside another term (for example, the main effect or the highest order interaction term). This can't be done using the Hasse Diagram App, but you can manually do this by altering the code the app generates.

```
modelLabels <- c("1 Relieve Pain 1", "cov Usage Time 1",
                 "3 Keyboard 2", "12 (Volunteers) 8")
modelMatrix <- matrix(
  data = c(FALSE, TRUE, TRUE, TRUE, #Row order matches label order
           FALSE, FALSE, FALSE, TRUE,
           FALSE, TRUE, FALSE, TRUE,
           FALSE, FALSE, FALSE, FALSE),
  nrow = 4,
  ncol = 4,
  byrow = TRUE # Set this to TRUE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)
```

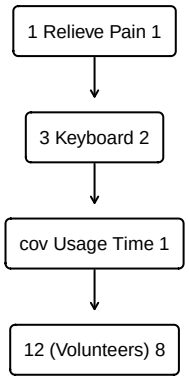


Figure 2: Hasse Diagram for Keyboarding Study, Version 2

The choice of which version is entirely up to you.

Data

For this example, you'll want to import the data as shown below. You'll notice that I'm using the `recode_factor` function from the `dplyr` package to translate the integers for both temperature and plate into more meaningful values (plus this tells R to treat those as factors).

```
# Load Keyboarding data
keyboarding <- 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
```

```r
## hrs.kbd is our covariate; hours spent using the keyboard
### make sure that R is NOT thinking of hrs.kbd as factor but
### either as num or int

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

Explore the data

When exploring the data, you'll want to make data visualizations and look at values of descriptive statistics. For ANCOVA situations, you'll want to be sure that you look at at least a scatter plot of the response by the covariate.

Fit the Model

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

```r
# Our main model
keyboardModel <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type,
  data = keyboarding
)

# Model for Checking covariate's homogeneity
interactionCheck <- aov(
  formula = hrs.pain ~ hrs.kbd * kbd.type,
  data = keyboarding
)
```

Check Assumptions

We do have more assumptions to check in an ANCOVA situation than an ANOVA situation. Keep in mind that if you introduce a Block or Random effects, then those assumptions will get imported into your situation as well. That is, block doesn't interact with factors, and each random effect factor has treatment effects which follow a Gaussian distribution. The approaches you used previously (as well as much of the code) remains the same.

Linear Relationship

The first assumption that I like to check in ANCOVA is that of the linear relationship between the covariate and the response. Again, 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 = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    shape = kbd.type, # Using shape and color is a good idea
    color = kbd.type
  )
) +
  geom_point(size = 2) +
```

```
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard Type", # Using identical values will merge to one legend
    shape = "Keyboard Type"
  )
```

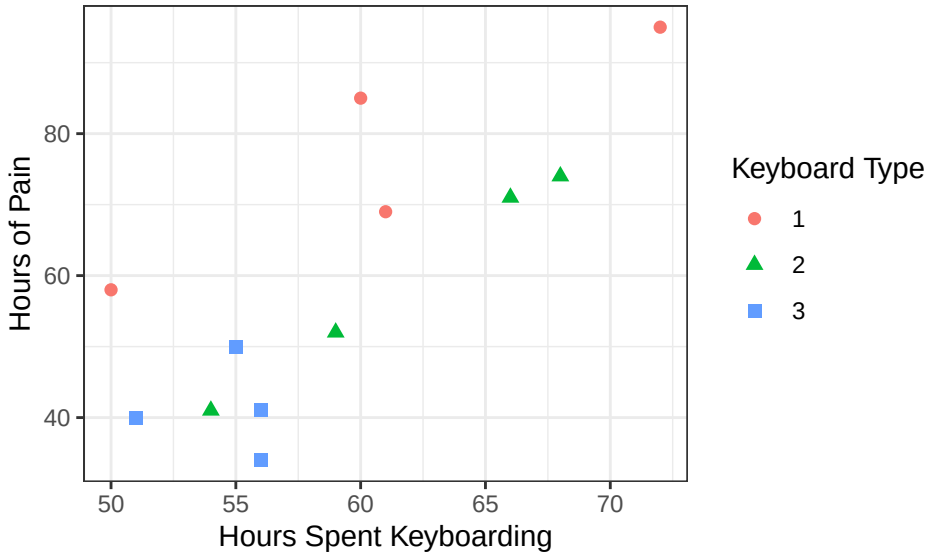


Figure 3: Hours of Pain vs Hours Spent Keyboarding

We're looking to see whether there is a linear relationship between the covariate and the response. If there isn't, we can use the plot to help us see what, if any, kind of relationship is present.

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 ~ hrs.kbd^2 + kbd.type` (don't forget to update the `interactionCheck` too) to linearize the response-covariate relationship.

Checking Potential Outliers

To help us check for potential outliers, we are going to make use of the `rstatix` package's `mahalanobis_distance` function. This is particularly useful whenever you have more than one numeric/continuous attribute (like a response and a covariate). Box plots and the various univariate Rules of Thumb only apply to one numeric/continuous attribute at a time.

```
# Detecting Multivariate Outliers

## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(keyboarding)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
  outlierDetection,
  factor = keyboarding$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"
  )
```

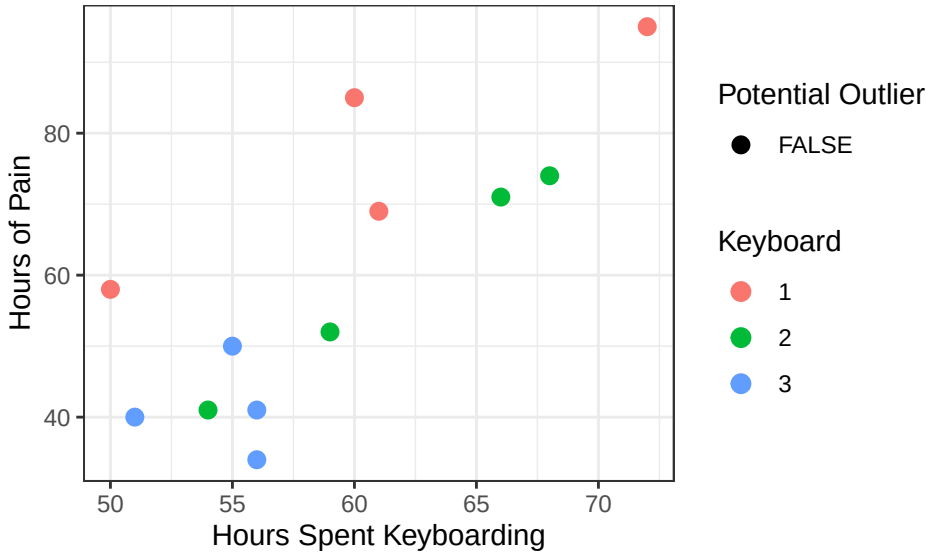


Figure 4: Potential Multivariate Outliers

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 an potential multivariate outliers to be concerned about.

Gaussian Residuals

Use a QQ plot like usual:

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

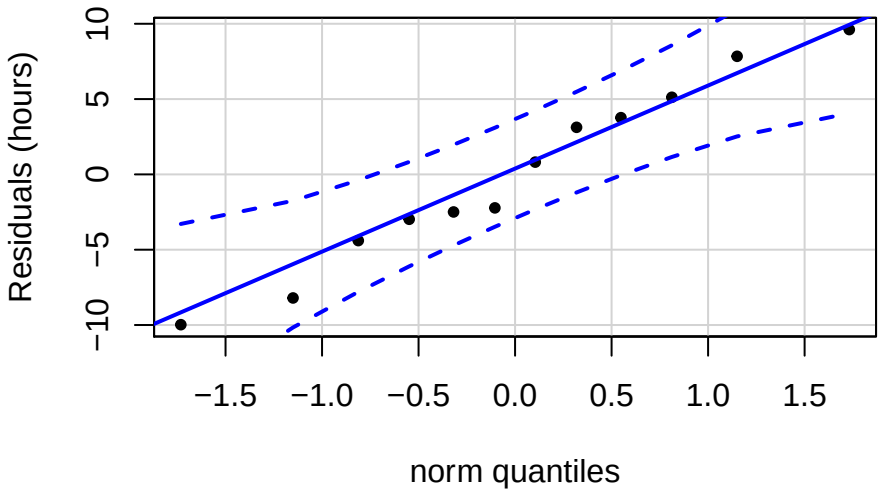


Figure 5: QQ Plot for Residuals

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

Homoscedasticity

Just as in the One-way ANOVA with a Block, we will want to look at a Tukey-Anscombe plot rather than a strip chart for our factorial designs.

```
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,
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (hours)") +
  ylab("Residuals (hours)")
```

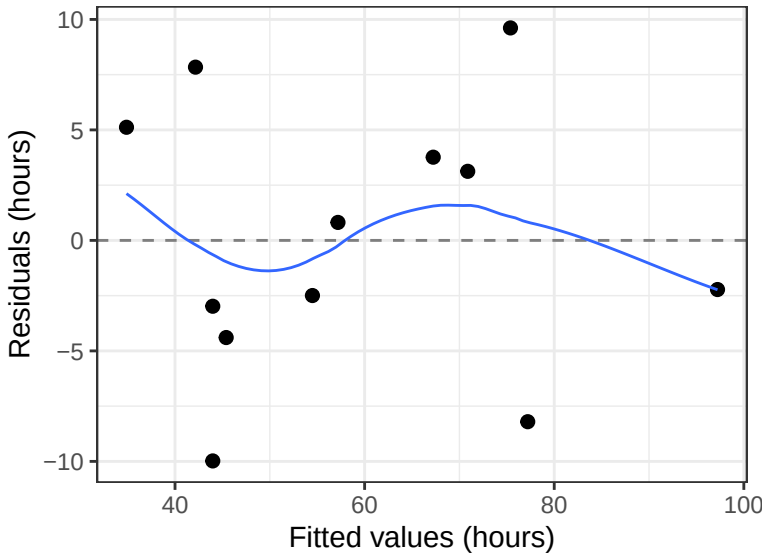


Figure 6: Tukey-Anscombe Plot for Keyboarding Study

I'm a bit hesitant for the homoscedasticity assumption given the curvy nature of the line. 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.

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.

(I'm leaving this to each of you to practice and come up with a justification for why we can say that we have independence of observations.)

Homogenity of the Covariate's Slope

The last assumption for ANCOVA is the homogeneity of the covariate's slope parameter. Remember, this is really just asking us to make sure that there is not a statistically significant interaction between the factor(s) and the covariate(s). We assess this assumption in two ways: check the Type III Sums of Squares ANOVA table of the interaction term (as an ***informal*** test) and look at a plot.

Informal Test Our first method is to look at the interaction term from a Type III Sums of Squares ANOVA table. Typically, we do not include this in any report, so we can use the raw output here.

```
# Remember to 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
```

We want to look at the interaction term `kbd.type:hrs.kbd` and see if this term would be statistically significant under our overall Type I Error Risk. In this situation, the interaction term is NOT statistically significant, thus we have some evidence for the homogeneity of the covariate's slope parameter.

Plot The second method 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 (i.e. Figure 3). However, this time we're going to add in some lines.

```
# Homogeneity of Slopes
ggplot(
  data = keyboarding,
  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"
  )
```

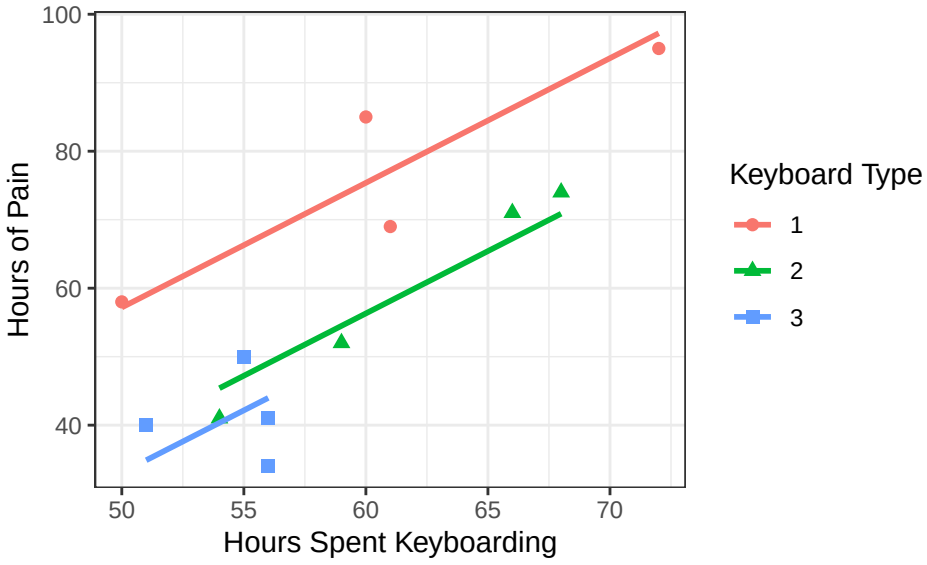


Figure 7: Homogeneity of Slopes

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 durations with the predicted durations from our ANCOVA model: `predict(keyboardModel)`.

What we want to do is examine this plot to see how well the three parallel lines match up with our data. That is to say, do the lines reasonably match up with the data?

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

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. My suggestion is to let the research questions posed guide you.

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.

```
# Omnibus Test/Modern ANOVA Table
parameters::model_parameters(
    model = keyboardModel,
    omega_squared = "partial",
    eta_squared = "partial",
    epsilon_squared = "partial"
) %>%
  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
```

```
    )
  ) %>%
  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 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 Omega Sq.	Partial Eta Sq.	Partial Epsilon Sq.
Hours Spent Keyboarding	2598.8209	1	2598.8209	50.5819	0.0001	0.8051	0.8634	0.8464
Keyboard Type	1195.8180	2	597.9090	11.6373	0.0043	0.6394	0.7442	0.6803
Residuals	411.0278	8	51.3785					

While we don't necessarily want to ignore the covariate's row in the table, 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.

Post Hoc Analysis

Point Estimates I want to quickly remind you that you can get point estimates for your main effects and treatment effects using the `dummy.coef` function. If you need confidence intervals for these, you can use the `confint` function (don't forget to provide an *adjusted* confidence level).

```
# Point Estimates for Keyboarding Model
## Don't use raw output in your reports, make a nice table
dummy.coef(keyboardModel)
```

```
## Full coefficients are
##
## (Intercept):     -48.2072
## hrs.kbd:          1.819896
## kbd.type:                1         2         3
##                  14.398515 -4.671381 -9.727135
```

Notice that our intercept is negative! For the ANCOVA models, the `(Intercept)` is NO LONGER the *Grand Mean*. To get the estimate of the *Grand Mean* you will need to take the intercept value and add 1.819896 * the *SAM* of the covariate (which is 59): $-48.2072 + 1.819896 * 59 = 59.17$ hours per person.

To get the the covariate-adjusted point estimates (i.e., the marginal cell means), we will turn to the `emmeans` package.

```
## Use emmeans to get the appropriate margins AND do Pairwise
emmOut <- emmeans::emmeans(
  object = keyboardModel,
  specs = pairwise ~ kbd.type,
  adjust = "tukey",
```

```
  level = 0.9
)

## Point Estimates
as.data.frame(emmOut$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 2: 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

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. I had forgotten to mention that the using `$contrasts` on the output of `emmeans` will give you when I made the Factorial Designs document.

```
# Pairwise Comparisons
as.data.frame(emmOut$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 3: Marginal Means-Tukey 90% Adjustment

Comparison	Difference	SE	DF	t Statistic	p-value
1 - 2	19.0699	5.0816	8	3.7527	0.0138
1 - 3	24.1257	5.5596	8	4.3395	0.0062
2 - 3	5.0558	5.7195	8	0.8839	0.6647

The `adjust` argument of `emmeans` allows for the following values for confidence intervals: `"bonferroni"`, `"tukey"`, `"scheffe"`, and `"sidak"`. If you do not want confidence intervals you may use values of `"holm"`, `"hochberg"`, `"hommel"`, `"BH"` (Benjamini and Hochberg), and `"fdr"`.

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).

```
# Pass the stored marginals into the effect size function
cohenType <- emmeans::eff_size(
  object = emmOut$emmeans,
  sigma = sigma(keyboardModel),
  edf = df.residual(keyboardModel)
)

# Create a data frame, add on the probability of superiority
# Send that data frame into a nice table
as.data.frame(cohenType) %>%
  dplyr::mutate(
    ps = probSup(effect.size),
    .after = effect.size
  ) %>%
  dplyr::select(contrast, effect.size, ps) %>%
  knitr::kable(
    digits = 3,
    col.names = c("Comparison", "Cohen's d", "Probability of Superiority"),
    align = "lcc",
    caption = "Effect Sizes for Keyboard Type",
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 4: Effect Sizes for Keyboard Type

Comparison	Cohen's d	Probability of Superiority
1 - 2	2.660	0.970
1 - 3	3.366	0.991
2 - 3	0.705	0.691

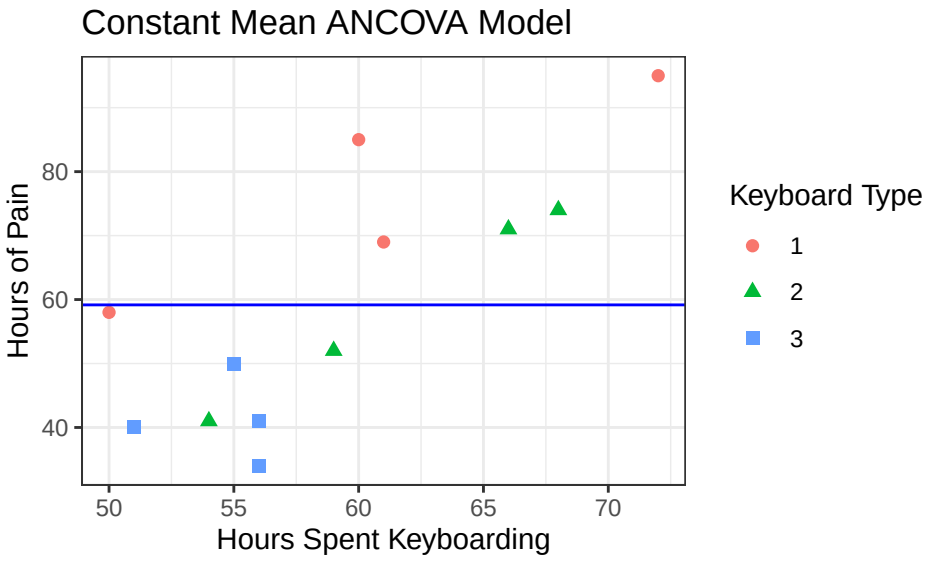
Five ANCOVA Models

There are actually five (5) different ANCOVA models. Each one is a slight variation and is applicable in different situations. They all have impacts on the *degrees of freedom* within the model, so you'll want to be cautious.

No Effects–Constant [Grand] Mean

This is the null model: that our factor(s) and our covariate(s) have absolutely no impact on the response. If this were true, we would end up with the following plot:

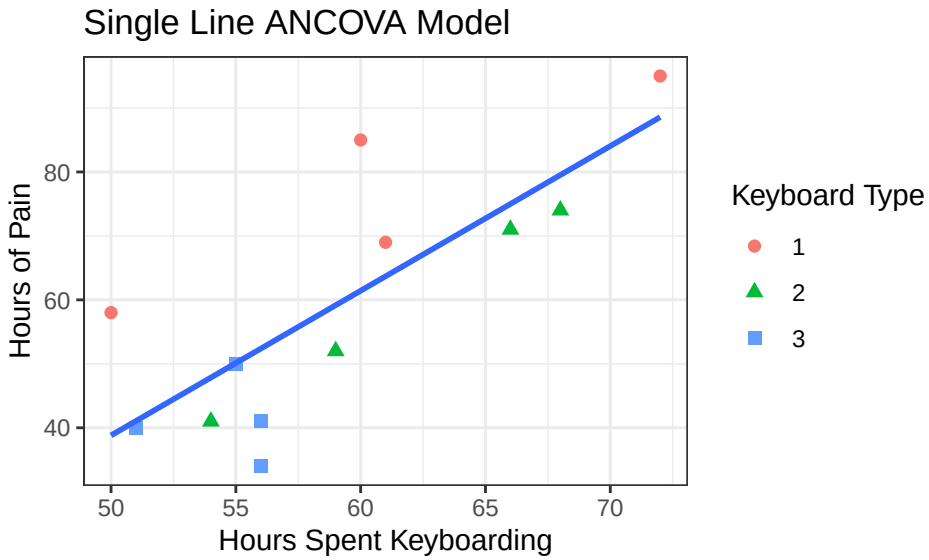
```
# Constant Mean
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = mean(keyboarding$hrs.pain),
    color = "blue"
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Constant Mean ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```



Single Line

This ANCOVA model assumes that the covariate(s) affect the response but the factor(s) does not. Thus, there is a single line for the relationship.

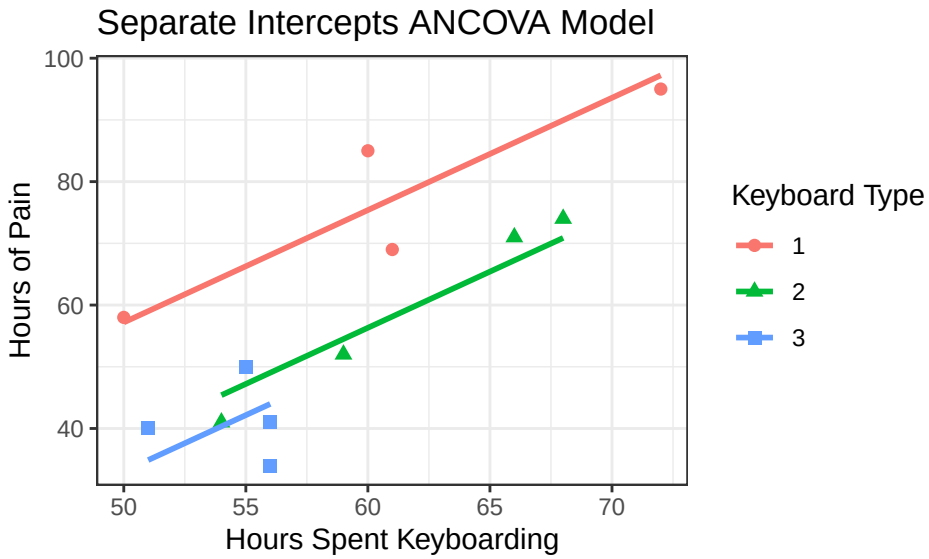
```
# Single Line
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ x,
    se = FALSE
    ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Single Line ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```



Separate Intercept/Parallel Lines

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.)

```
# Separate Intercepts
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Intercepts ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```

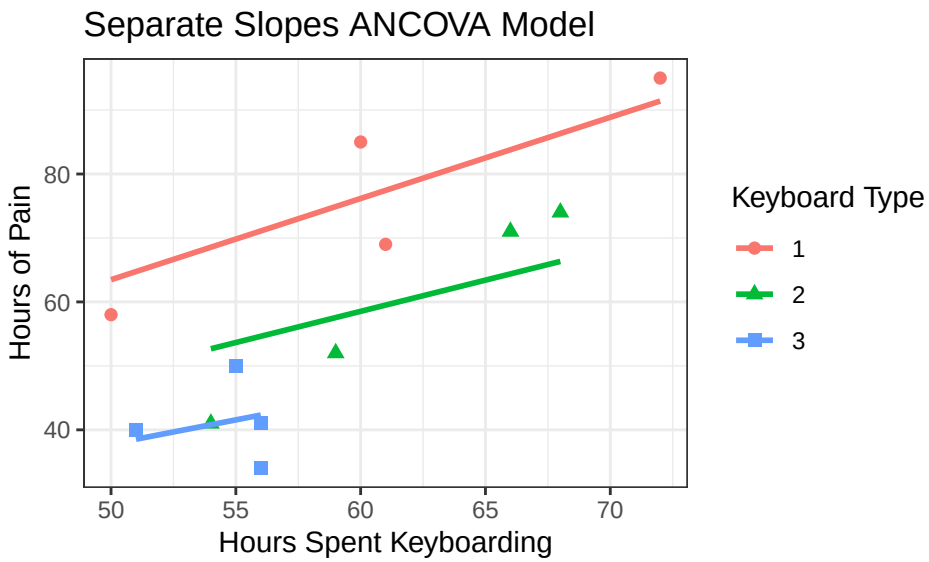


Separate Slopes

This version of the ANCOVA model fixes the y-intercept to the same value for all groups but allows each group to have their own constant rate of change (slopes) for the covariate term.

```
# Separate Slopes
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
```

```
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x + 0,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Slopes ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```

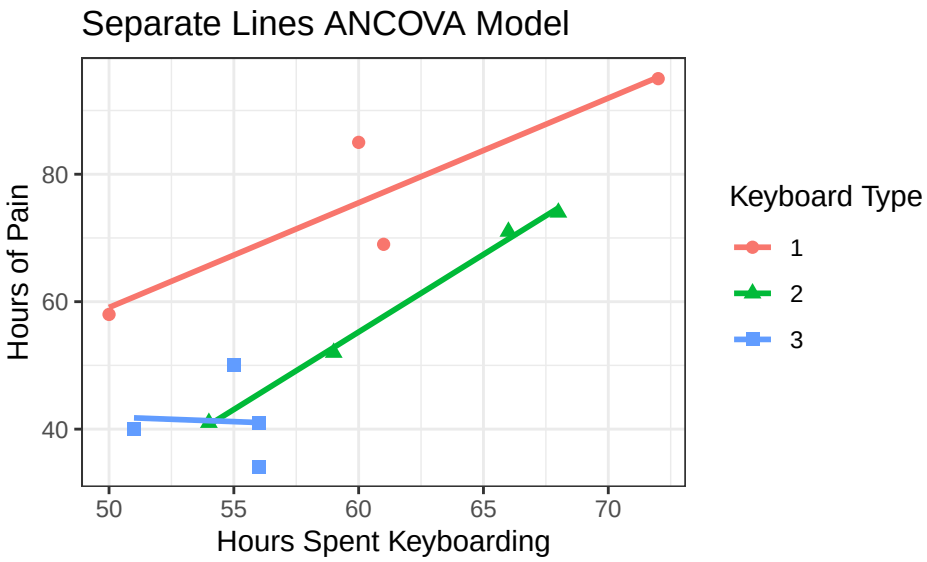


Separate Lines

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. For our keyboarding data, we would see the following.

```
# Separate Lines
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
```

```
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Lines ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```



Notice that each of these models looks different within our Keyboarding context. We can also see that some models appear to be inconsistent with our data (e.g., the Constant Mean ANCOVA model). You can learn more about these models in Section 17.3 of Oehlert.

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", "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")

# Hasse Diagram
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
)

modelLabels <- c("1 Relieve Pain 1", "cov Usage Time 1",
                 "3 Keyboard 2", "12 (Volunteers) 8")
modelMatrix <- matrix(
  data = c(FALSE, TRUE, TRUE, TRUE, #Row order matches label order
           FALSE, FALSE, FALSE, TRUE,
           FALSE, TRUE, FALSE, TRUE,
           FALSE, FALSE, FALSE, FALSE),
  nrow = 4,
  ncol = 4,
  byrow = TRUE # Set this to TRUE
)
hasseDiagram::hasse(
 data = modelMatrix,
 labels = modelLabels
)

# Load Keyboarding data
keyboarding <- 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 NOT thinking of hrs.kbd as factor but
### either as num or int

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

# Our main model
keyboardModel <- aov(
  formula = hrs.pain ~ hrs.kbd + kbd.type,
  data = keyboarding
)

# Model for Checking covariate's homogeneity
interactionCheck <- aov(
  formula = hrs.pain ~ hrs.kbd * kbd.type,
  data = keyboarding
)

# Scatter plot of hours of pain and hours spent keyboarding
# Notice we're just using the data, not the model
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    shape = kbd.type, # Using shape and color is a good idea
    color = kbd.type
  )
) +
  geom_point(size = 2) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    color = "Keyboard Type", # Using identical values will merge to one legend
    shape = "Keyboard Type"
  )

# Detecting Multivariate Outliers

## Step 1: send the data through the Mahalanobis function
outlierDetection <- rstatix::mahalanobis_distance(keyboarding)

## Step 2: OPTIONAL--reattach the factor
outlierDetection <- cbind(
```

```
  outlierDetection,
  factor = keyboarding$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"
  )


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


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,
```

```
    size = 0.5
  ) +
  theme_bw() +
  xlab("Fitted values (hours)") +
  ylab("Residuals (hours)")

# Remember to use the car package
car::Anova(
  mod = interactionCheck,
  type = 3
)

# Homogeneity of Slopes
ggplot(
  data = keyboarding,
  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"
  )

# Omnibus Test/Modern ANOVA Table
parameters::model_parameters(
    model = keyboardModel,
    omega_squared = "partial",
    eta_squared = "partial",
    epsilon_squared = "partial"
) %>%
  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
    )
  ) %>%
  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 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")
  )

# Point Estimates for Keyboarding Model
## Don't use raw output in your reports, make a nice table
dummy.coef(keyboardModel)

## Use emmeans to get the appropriate margins AND do Pairwise
emmOut <- emmeans::emmeans(
  object = keyboardModel,
  specs = pairwise ~ kbd.type,
  adjust = "tukey",
  level = 0.9
)

## Point Estimates
as.data.frame(emmOut$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")
  )

# Pairwise Comparisons
as.data.frame(emmOut$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")
  )

# Pass the stored marginals into the effect size function
cohenType <- emmeans::eff_size(
  object = emmOut$emmeans,
  sigma = sigma(keyboardModel),
  edf = df.residual(keyboardModel)
)

# Create a data frame, add on the probability of superiority
# Send that data frame into a nice table
as.data.frame(cohenType) %>%
  dplyr::mutate(
    ps = probSup(effect.size),
    .after = effect.size
  ) %>%
  dplyr::select(contrast, effect.size, ps) %>%
  knitr::kable(
    digits = 3,
    col.names = c("Comparison", "Cohen's d", "Probability of Superiority"),
    align = "lcc",
    caption = "Effect Sizes for Keyboard Type",
    booktab = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("striped", "condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# Constant Mean
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_hline(
    yintercept = mean(keyboarding$hrs.pain),
    color = "blue"
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Constant Mean ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
```

```
  )

# Single Line
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    inherit.aes = FALSE,
    mapping = aes(x = hrs.kbd, y = hrs.pain),
    method = "lm",
    formula = y ~ x,
    se = FALSE
    ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Single Line ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )

# Separate Intercepts
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    mapping = aes(y = predict(keyboardModel)),
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Intercepts ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
```

```
  )

# Separate Slopes
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x + 0,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Slopes ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )

# Separate Lines
ggplot(
  data = keyboarding,
  mapping = aes(
    y = hrs.pain,
    x = hrs.kbd,
    color = kbd.type,
    shape = kbd.type
  )
) +
  geom_point(size = 2) +
  geom_smooth(
    method = "lm",
    formula = y ~ x,
    se = FALSE
  ) +
  theme_bw() +
  xlab("Hours Spent Keyboarding") +
  ylab("Hours of Pain") +
  labs(
    title = "Separate Lines ANCOVA Model",
    color = "Keyboard Type",
    shape = "Keyboard Type"
  )
```