Recently had the need to run an ANCOVA, not a task I perform all that  
often and my first time using R to do so (I’ve done it in SPSS and  
SAS before). Having a decent theoretical idea of what I had to do I  
set off in search of decent documentation on how to accomplish it in R. I  
was quite disappointed with what I found after a decent amount of time  
scouring the web (or at least I thought so). I found “answers” in places  
like “Stack Overflow” and “Cross Validated” as well as various free and  
open notes from academic courses. Many were dated, a few off topic, a  
few outright incorrect, if you ask me, but nothing I could just pick up  
and use.

So I wrote my own top to bottom example that I’ll publish on this blog. Not  
necessarily because others will find it, but more to ensure I document my  
own learning. I may also cross post a shortened version in a couple of  
places and point back to this longish posting.

**Before you read any farther**

Some constraints I’m placing on this post that may impact your desire to  
read it.

1. Yes ANOVA is a subset of the general linear model and specifically  
   in R aov is just a wrapper around lm. I can’t tell you the  
   number of times I read that and it’s true. **But**, in many  
   disciplines and for me it is an important subset worthy of it’s own  
   time. You won’t find any info here about how to do things with lm  
   although you certainly could.
2. Yes contrasts can play an important role in understanding your  
   results. This is especially true if you happen to have an unbalanced  
   design. I’m only going to glance on the topic here since I’m going  
   to run a balanced design and therefore contrasts are tangential. For  
   a very good academic discussion of contrasts especially using R and  
   especially at the introductory level
3. The text is free for the  
   download and is invaluable although it doesn’t cover ANCOVA per se.
4. I’m going to use packages above and beyond base and stats as  
   needed. Everything could be done with the basics but this is a  
   practical approach not so much a theoretical approach. I’ll even put  
   in a very subtle plug for a function I wrote and maintain on CRAN  
   although it’s by no means required.

*N.B.* – I don’t expect this will reach the New York Times best-seller  
list but questions or comments if it is useful or if I have missed  
something, are most certainly welcome.

**Background and problem statement**

The [Wikipedia definition of  
ANCOVA](https://en.wikipedia.org/wiki/Analysis_of_covariance) is  
actually quite good and I won’t bother to repeat it. Some other keys  
phrases you’ll hear are that ANCOVA allows you to *“control for”* or  
*“partial out”* the covariate which gives you the opportunity to  
estimate partial means or marginal means which at the end of the day  
is why one uses ANOVA/ANCOVA versus regression. They use the same math  
but ANOVA/ANCOVA typically reports in means and mean differences while  
regressions reports on the slopes of the regression terms. Yes you can  
move back and forth but many disciplines have a preference and it can be  
one less step.

I wanted to make sure I used a dataset that was easily accessible. I  
mean that both in terms of being available in a package that most users  
will all ready have as well on a topic that many will find intuitively  
understandable with no specialized knowledge. I selected the diamonds  
dataset from ggplot2. I’ll cut it down some and balance it but those  
are trivial steps that will hopefully make things clearer.

So imagine that you’re shopping for a diamond. You’d like to get the  
best possible value for the money you spend but you have very little  
knowledge about what influences the price you pay versus the value you  
get. The sales people speak about various things that influence the  
price such as “cut” and “clarity” and “color” and “carats”. You don’t  
have a lot to spend so you’re going to limit yourself to something  
modest but you’d like to know you got good value for what you paid.  
Enter the diamonds dataset from ggplot2 if you have library available  
you can get a terse description with ?diamonds.

Let’s say for the sake of argument you’d like to know more about how  
these factors of cut and color impact the price you’ll pay. Let’s go  
ahead and get things set up in R so we can proceed, load the right  
libraries etc.. You’ll see in the code I recommend grabbing the latest  
version of a package I maintain but it is totally optional and there’s  
nothing there you can’t do for yourself if you prefer. I just wrote it  
so I didn’t have to remember to repeat a bunch of steps in R to run a  
2Way ANOVA. I’m going to assume you’re comfortable with a basic ANOVA although  
you’re welcome to review the vignette if that’s helpful. So let’s load  
the libraries (I suppressed all the messages here), and check out the  
structure of the dataset.

require(car) # get the right sums of squares calculations

require(dplyr) # for manipulating our data

require(ggplot2) # for plotting and for our dataset

require(sjstats) # save us time computing key ANOVA stats beyond car

require(broom) # nice for making our results in neat tibbles

require(emmeans) # for marginal means calculations

# a shameless plug for a function I wrote called Plot2WayANOVA

# optional for you

library(CGPfunctions)

theme\_set(theme\_bw()) # set theme

str(diamonds)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 53940 obs. of 10 variables:

## $ carat : num 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...

## $ cut : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...

## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...

## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...

## $ depth : num 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...

## $ table : num 55 61 65 58 58 57 57 55 61 61 ...

## $ price : int 326 326 327 334 335 336 336 337 337 338 ...

## $ x : num 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...

## $ y : num 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...

## $ z : num 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...

Okay just crossing cut and color would give us 35 cells in a table  
of means ( 5 levels times 7 levels). I’ve also admitted I’m a cheapskate  
and don’t want a huge diamond so let’s pare our data down a bit to  
something more manageable. So let’s use dplyr to create a subset of  
the data where we focus on “fair” and “good” cuts, and colors “E”, “F”,  
and “G” with a carat weight of less than 1.75. This is also a good time  
to admit I cheated and peeked and saw that the data were likely to be  
very unbalanced table(diamonds$color, diamonds$cut), so at this point  
I’m also going to force our data into a balanced design by ensuring that  
we randomly sample the same number of data points into each cell. I’ve  
used set.seed so you should be able to reproduce the same dataset if you  
choose.

set.seed(1234)

diamonds2 <- filter(diamonds,

cut %in% c("Fair", "Good") &

color %in% c("E", "F", "G") &

carat < 1.75)

diamonds2 <- droplevels(diamonds2)

one <- diamonds2 %>% filter(cut == "Fair" & color == "E") %>% sample\_n(218)

two <- diamonds2 %>% filter(cut == "Fair" & color == "F") %>% sample\_n(218)

three <- diamonds2 %>% filter(cut == "Fair" & color == "G") %>% sample\_n(218)

four <- diamonds2 %>% filter(cut == "Good" & color == "E") %>% sample\_n(218)

five <- diamonds2 %>% filter(cut == "Good" & color == "F") %>% sample\_n(218)

six <- diamonds2 %>% filter(cut == "Good" & color == "G") %>% sample\_n(218)

diamonds2 <- bind\_rows(one, two, three, four, five, six)

str(diamonds2)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 1308 obs. of 10 variables:

## $ carat : num 1.01 0.47 0.4 0.47 0.5 0.5 1.01 1 0.3 1.51 ...

## $ cut : Ord.factor w/ 2 levels "Fair"<"Good": 1 1 1 1 1 1 1 1 1 1 ...

## $ color : Ord.factor w/ 3 levels "E"<"F"<"G": 1 1 1 1 1 1 1 1 1 1 ...

## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 4 3 5 2 1 2 6 3 ...

## $ depth : num 64.6 67.9 64.7 65.2 56.1 58 64.5 64.6 51 67.2 ...

## $ table : num 59 56 58 59 64 67 58 60 67 53 ...

## $ price : int 3294 829 813 828 1950 851 2788 4077 945 7468 ...

## $ x : num 6.22 4.84 4.63 4.75 5.31 5.26 6.29 6.23 4.67 7.15 ...

## $ y : num 6.18 4.65 4.67 4.82 5.28 5.17 6.21 6.18 4.62 7.1 ...

## $ z : num 4.01 3.24 3.01 3.12 2.97 3.02 4.03 4.01 2.37 4.79 ...

**A note on balanced designs**

As I noted earlier I’m really not interested in digressing to talk about  
why the concept of a balanced design is important to your work. At this point  
I’m simply going to encourage you to always use “Type II” sums of  
squares from the car package if there is any chance your design is  
unbalanced. If your design is balanced they give the same answer. The  
contrast you choose is also if you are unbalanced and are using Type  
III.

Are unbalanced designs completely wrong and to be avoided at all costs?  
Not exactly… here are three things they impact in order of likelihood.

1. They always impact your power, you ability to detect significant  
   differences. Your power is limited by the size of your smallest  
   cell.
2. They usually impact your ability to divide the sums of squares  
   cleanly to 100%. You may wind up with unexplained variance that is  
   due to an effect but you won’t know which effect. This is different than  
   unexplained (residual variance).
3. The least likely but most worrisome is that it will mask an  
   important relationship in your data.

**Back to the diamonds**

Before we look at ANCOVA lets run an ANOVA. We have two ordinal factors  
for independent (predictor) variables cut and color and one  
dependent (outcome) variable the price. A classic two-way ANOVA. We  
could simply run `aov(price ~ cut \* color, diamonds2) and then a  
bunch of other commands to get the information we need. I found that a  
bit tedious and annoying plus I wanted to be able to plot the results to  
look at any possible interactions. So I wrote a function. Everything in  
it you can do by hand but I think it does a pretty good job wrapping the  
process in one function. So …

Plot2WayANOVA(price ~ color \* cut,

diamonds2,

mean.label = TRUE)

##

## You have a balanced design.

## term sumsq meansq df statistic p.value etasq

## 1 color 1.827925e+07 9139623.81 2 1.246 0.288 0.002

## 2 cut 2.030077e+04 20300.77 1 0.003 0.958 0.000

## 3 color:cut 3.744738e+07 18723690.62 2 2.552 0.078 0.004

## 4 Residuals 9.553303e+09 7337406.33 1302 NA NA NA

## partial.etasq omegasq partial.omegasq cohens.f power

## 1 0.002 0.000 0.000 0.044 0.273

## 2 0.000 -0.001 -0.001 0.001 0.050

## 3 0.004 0.002 0.002 0.063 0.512

## 4 NA NA NA NA NA

##

## Measures of overall model fit

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC

##

## 1 0.00580 0.00198 2709. 1.52 0.181 6 -12192. 24397.

## # … with 3 more variables: BIC , deviance , df.residual

##

## Table of group means

## # A tibble: 6 x 9

## # Groups: color [3]

## color cut TheMean TheSD TheSEM CIMuliplier LowerBound UpperBound N

##

## 1 E Fair 3406. 2485. 168. 1.97 3075. 3738. 218

## 2 E Good 3454. 3418. 232. 1.97 2998. 3911. 218

## 3 F Fair 3429. 2571. 174. 1.97 3086. 3773. 218

## 4 F Good 3004. 2290. 155. 1.97 2699. 3310. 218

## 5 G Fair 3292. 2263. 153. 1.97 2990. 3594. 218

## 6 G Good 3693. 3031. 205. 1.97 3289. 4098. 218

##

## Post hoc tests for all effects that were significant

## [1] "No signfiicant effects"

##

## Testing Homogeneity of Variance with Brown-Forsythe

## \*\*\* Possible violation of the assumption \*\*\*

## Levene's Test for Homogeneity of Variance (center = median)

## Df F value Pr(>F)

## group 5 4.2215 0.0008235 \*\*\*

## 1302

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Testing Normality Assumption with Shapiro-Wilk

## \*\*\* Possible violation of the assumption. You may

## want to plot the residuals to see how they vary from normal \*\*\*

##

## Shapiro-Wilk normality test

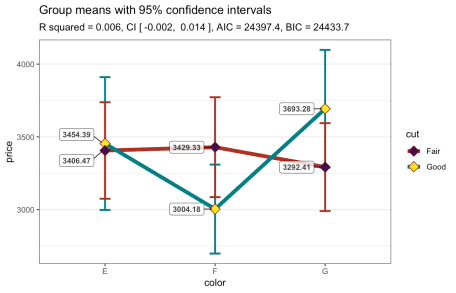
##

## data: MyAOV\_residuals

## W = 0.85121, p-value < 2.2e-16

##

## Interaction graph plotted...



It does it’s job of plotting the results and providing you with nice  
summaries of not just the ANOVA table but the table of means, post hoc  
tests if needed and even testing classic assumptions. it also tests for  
whether your design is balanced and always uses type II sums of squares.

Well this is unfortunate! Looks like I won’t be publishing the results  
in *JODR – The Journal of Obscure Diamond Results* since not a single  
one of my ANOVA terms is significant at the p<.05 level. The model  
seems to be a terrible fit whether I look at R Squared or AIC or BIC,  
neither cut nor color seem to matter although the interaction term  
is “marginally significant”.

Should we conclude cut and color don’t matter? Are they just  
immaterial with no discernible impact on pricing?

**ANCOVA helps our understanding**

As you have probably already guessed that’s not where we’re heading.  
Remember that Wikipedia article? A telling quote from Tabachnick, B. G.  
and Fidell, L. S. (2007) is in there…

ANCOVA can be used to increase statistical power (the probability a  
significant difference is found between groups when one exists) by  
reducing the within-group error variance.

So with ANCOVA we’re going to add one or more continuous variables known  
as covariates which are “hiding” the relationship between our factors of  
interest cut and color, if we can control for or partial the  
covariate out then we’ll hopefully be likely to “see” the impact of  
cut and color on price. It should be something we know is linearly  
related to price but distinct from cut and color.

So looking at our data above it hopefully is becoming obvious to you. On  
surface to meet the criteria of being a continuous numeric variable we  
have "carat", "depth", "table", "x", "y" and "z". Since  
clarity is ordered we could force it to an integer and use it but  
let’s not. Speaking for me, I’m pretty sure size as measured in weight  
carat is going to be related to price of the diamond. I don’t know  
much about diamonds but I’ve heard size adds to the cost…

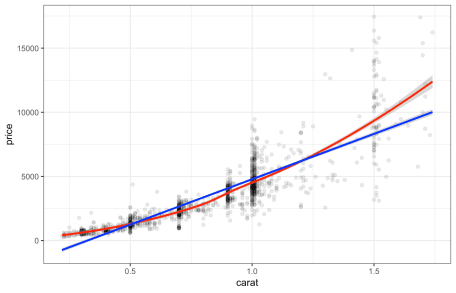
So what we want to do now is confirm our hunch it is linearly related as  
well as hope that it is not strongly correlated with our current  
predictors cut and color we’re looking for a variable that reduces  
the variance but is not entangled with our current IVs. So first a  
scatter plot with the regression and loess lines followed by a glance  
at the linear models price ~ carat, carat ~ cut, and carat ~  
color.

ggplot(diamonds2, aes(x = carat, y= price)) +

geom\_point(alpha = 1/10) +

geom\_smooth(method = "loess", color = "red") +

geom\_smooth(method = "lm", color = "blue")



broom::glance(lm(price ~ carat, diamonds2))

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC

##

## 1 0.743 0.743 1374. 3781. 0 2 -11306. 22618.

## # … with 3 more variables: BIC , deviance , df.residual

broom::glance(lm(carat ~ cut, diamonds2))

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC

##

## 1 0.0163 0.0156 0.328 21.7 3.59e-6 2 -397. 799. 815.

## # … with 2 more variables: deviance , df.residual

broom::glance(lm(carat ~ color, diamonds2))

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC

##

## 1 0.00730 0.00578 0.330 4.80 0.00837 3 -402. 813. 834.

## # … with 2 more variables: deviance , df.residual

caratonly <- lm(price ~ carat, diamonds2)

**Excellent news! Our potential covariate carat is highly correlated  
with price (r = 0.86) while having near zero correlations with cut  
(r = 0.13) and color (r = 0.09).**

**Comparing models**

We’re going to compare two different models in a step by step fashion  
using the same tools in R for each step, aov to create the model,  
car::Anova to display the ANOVA table and ensure we’re using type 2  
sums of squares. broom::glance to get information about overall fir  
like R squared and AIC, and finally sjstats::anova\_stats to give us a  
nice clean display including effect sizes for the terms.

So our original model without a covariate looked like this (you can  
scroll back and look but I assure you this is it).

noCOVmodel <- aov(price ~ cut \* color, diamonds2)

car::Anova(noCOVmodel, type = 2)

## Anova Table (Type II tests)

##

## Response: price

## Sum Sq Df F value Pr(>F)

## cut 20301 1 0.0028 0.95806

## color 18279248 2 1.2456 0.28810

## cut:color 37447381 2 2.5518 0.07833 .

## Residuals 9553303035 1302

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

broom::glance(noCOVmodel)

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC

##

## 1 0.00580 0.00198 2709. 1.52 0.181 6 -12192. 24397.

## # … with 3 more variables: BIC , deviance , df.residual

sjstats::anova\_stats(car::Anova(noCOVmodel, type = 2)) %>% select(1:7)

## term sumsq meansq df statistic p.value etasq

## 1 cut 2.030077e+04 20300.77 1 0.003 0.958 0.000

## 2 color 1.827925e+07 9139623.81 2 1.246 0.288 0.002

## 3 cut:color 3.744738e+07 18723690.62 2 2.552 0.078 0.004

## 4 Residuals 9.553303e+09 7337406.33 1302 NA NA NA

**Adding carat to the model**

Let’s add another term to the model to include carat, we’re not going  
to let it interact with the other factors so we’ll use a plus sign.  
There’s nothing especially tricky about this, we’re just adding another  
predictor to our model, it’s not exactly traditional ANOVA because it’s  
a continuous numeric variable rather than a factor, but it’s simple to  
imagine.

COVmodel <- aov(price ~ cut \* color + carat, diamonds2)

car::Anova(COVmodel, type = 2)

## Anova Table (Type II tests)

##

## Response: price

## Sum Sq Df F value Pr(>F)

## cut 122611531 1 69.0892 2.342e-16 \*\*\*

## color 26713642 2 7.5263 0.0005625 \*\*\*

## carat 7244439236 1 4082.1011 < 2.2e-16 \*\*\*

## cut:color 9593270 2 2.7028 0.0673932 .

## Residuals 2308863799 1301

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

broomExtra::glance(COVmodel)

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC

##

## 1 0.760 0.759 1332. 686. 0 7 -11263. 22542.

## # … with 3 more variables: BIC , deviance , df.residual

sjstats::anova\_stats(car::Anova(COVmodel, type = 2)) %>% select(1:7)

## term sumsq meansq df statistic p.value etasq

## 1 cut 122611531 122611531 1 69.089 0.000 0.013

## 2 color 26713642 13356821 2 7.526 0.001 0.003

## 3 carat 7244439236 7244439236 1 4082.101 0.000 0.746

## 4 cut:color 9593270 4796635 2 2.703 0.067 0.001

## 5 Residuals 2308863799 1774684 1301 NA NA NA

Wow that sure changed our results didn’t it? Suddenly cut and color  
matter! Make no mistake they don’t have nearly the impact that carat  
does but at least we can reliably see their impact on price by  
measures such as eta squared (etasq).

That’s because we originally had 9,553,303,035 in residual sums of  
squares but by adding carat we’ve reduced that number to 2,308,863,799  
which makes our factors (cut and color) (the numerators) much more  
potent.

A quick peek into what’s changing may help. Let’s build a little tibble  
that shows us what’s going on. The first three columns are straight from  
the diamonds2 dataset, our price, cut, and color. The next  
column shows the “prediction” made using our initial model. Every row  
with the same condition e.g. “Fair” & “E” gets the same entry “3406” the  
mean for the cell. We can see that when we use just carat as the  
predictor we get very different predictions (although of course the same  
size gets the same prediction). Our COVmodel predictions yield a third  
set of answers in the final column that makes use of all the information  
available.

diamonds3 <- diamonds2 %>%

mutate(OriginalPred = predict(noCOVmodel),

WithCaratPred = predict(COVmodel),

CaratOnlyPred = predict(caratonly)) %>%

select(price, cut, color, OriginalPred, carat, CaratOnlyPred, WithCaratPred)

diamonds3

## # A tibble: 1,308 x 7

## price cut color OriginalPred carat CaratOnlyPred WithCaratPred

##

## 1 3294 Fair E 3406. 1.01 4850. 4743.

## 2 829 Fair E 3406. 0.47 1030. 848.

## 3 813 Fair E 3406. 0.4 535. 343.

## 4 828 Fair E 3406. 0.47 1030. 848.

## 5 1950 Fair E 3406. 0.5 1242. 1064.

## 6 851 Fair E 3406. 0.5 1242. 1064.

## 7 2788 Fair E 3406. 1.01 4850. 4743.

## 8 4077 Fair E 3406. 1 4779. 4671.

## 9 945 Fair E 3406. 0.3 -172. -379.

## 10 7468 Fair E 3406. 1.51 8387. 8350.

## # … with 1,298 more rows

**More progress with emmeans**

Okay, we’re making progress here but this isn’t all we can or should do.  
We have a good sense that adding carat as a covariate makes for a much  
more accurate model. But, we’re not interested in carat per se, it’s  
not that it’s unimportant (clearly it matters) we’re just interested in  
what happens to cut and color when we control for carat. That’s  
where the emmeans package can help.

The emmeans package allows us to take our model(s) and compute the  
*estimated marginal means* a.k.a. *predicted model means* or *least  
squares means*. The package includes functions to not only compute them  
but also plot them as well as make comparisons. We’ve already done that  
above for our model with no covariate noCOVmodel but let’s see what  
that looks like just as a baseline.

# first the means

emmeans::pmmeans(noCOVmodel, "cut", by = "color")

## color = E:

## cut pmmean SE df lower.CL upper.CL

## Fair 3406 183 1302 3047 3766

## Good 3454 183 1302 3094 3814

##

## color = F:

## cut pmmean SE df lower.CL upper.CL

## Fair 3429 183 1302 3069 3789

## Good 3004 183 1302 2644 3364

##

## color = G:

## cut pmmean SE df lower.CL upper.CL

## Fair 3292 183 1302 2932 3652

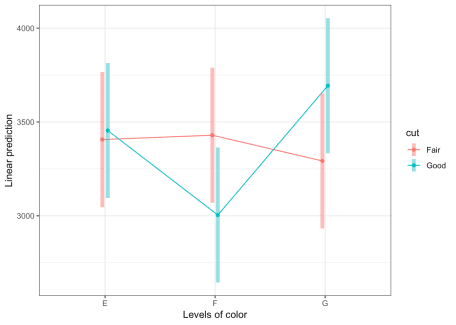
## Good 3693 183 1302 3333 4053

##

## Confidence level used: 0.95

# then plot them

emmeans::emmip(noCOVmodel, cut ~ color, CIs = TRUE)



# pairwise comparisons

pairs(emmeans::pmmeans(noCOVmodel, "color", by = "cut"), adjust = "scheffe")

## cut = Fair:

## contrast estimate SE df t.ratio p.value

## E - F -22.9 259 1302 -0.088 0.9961

## E - G 114.1 259 1302 0.440 0.9079

## F - G 136.9 259 1302 0.528 0.8700

##

## cut = Good:

## contrast estimate SE df t.ratio p.value

## E - F 450.2 259 1302 1.735 0.2223

## E - G -238.9 259 1302 -0.921 0.6546

## F - G -689.1 259 1302 -2.656 0.0297

##

## P value adjustment: scheffe method with dimensionality 2

# not done above you can easily specify just one factor

emmeans::pmmeans(noCOVmodel, "cut")

## NOTE: Results may be misleading due to involvement in interactions

## cut pmmean SE df lower.CL upper.CL

## Fair 3376 106 1302 3168 3584

## Good 3384 106 1302 3176 3592

##

## Results are averaged over the levels of: color

## Confidence level used: 0.95

# or the other factor

emmeans::pmmeans(noCOVmodel, "color")

## NOTE: Results may be misleading due to involvement in interactions

## color pmmean SE df lower.CL upper.CL

## E 3430 130 1302 3176 3685

## F 3217 130 1302 2962 3471

## G 3493 130 1302 3238 3747

##

## Results are averaged over the levels of: cut

## Confidence level used: 0.95

**Controlling for carat**

None of that information is what we’re after however. We have the other  
model COVmodel with carat added and what we need are the estimated  
means with carat controlled for, or partialled out. We want to know  
the predicted or estimated means for our 6 conditions as if size  
(carat) were controlled for.

emmeans::pmmeans(COVmodel, "carat")

## carat pmmean SE df lower.CL upper.CL

## 0.802 3380 36.8 1301 3308 3452

##

## Results are averaged over the levels of: cut, color

## Confidence level used: 0.95

emmeans::pmmeans(COVmodel, "cut", by = "color")

## color = E:

## cut pmmean SE df lower.CL upper.CL

## Fair 3244 90.3 1301 3067 3421

## Good 3868 90.5 1301 3690 4045

##

## color = F:

## cut pmmean SE df lower.CL upper.CL

## Fair 3177 90.3 1301 3000 3354

## Good 3580 90.7 1301 3402 3758

##

## color = G:

## cut pmmean SE df lower.CL upper.CL

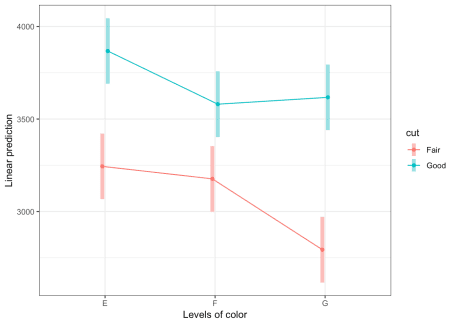
## Fair 2794 90.6 1301 2616 2972

## Good 3617 90.2 1301 3440 3794

##

## Confidence level used: 0.95

emmip(COVmodel, cut ~ color, CIs = TRUE)



For me comparing the two plots tells a striking story about what role  
cut and color play if we separate out the effect of size. If you  
look t the tables of means you can see they are different but looking at  
the plot gives you a much better idea of just how much the pattern has  
changed.

We can also run the appropriate significance tests using the very  
conservative scheffe option.

pairs(emmeans::pmmeans(COVmodel, "cut", by = "color"), adjust = "scheffe")

## color = E:

## contrast estimate SE df t.ratio p.value

## Fair - Good -623 128 1301 -4.873 <.0001

##

## color = F:

## contrast estimate SE df t.ratio p.value

## Fair - Good -403 128 1301 -3.146 0.0017

##

## color = G:

## contrast estimate SE df t.ratio p.value

## Fair - Good -823 128 1301 -6.442 <.0001

pairs(emmeans::pmmeans(COVmodel, "color", by = "cut"), adjust = "scheffe")

## cut = Fair:

## contrast estimate SE df t.ratio p.value

## E - F 67.5 128 1301 0.529 0.8695

## E - G 450.3 128 1301 3.526 0.0021

## F - G 382.8 128 1301 2.999 0.0113

##

## cut = Good:

## contrast estimate SE df t.ratio p.value

## E - F 287.4 128 1301 2.252 0.0796

## E - G 250.5 128 1301 1.960 0.1469

## F - G -36.9 128 1301 -0.288 0.9594

##

## P value adjustment: scheffe method with dimensionality 2

pairs(emmeans::pmmeans(COVmodel, "color"), adjust = "scheffe")

## NOTE: Results may be misleading due to involvement in interactions

## contrast estimate SE df t.ratio p.value

## E - F 177 90.2 1301 1.967 0.1450

## E - G 350 90.5 1301 3.874 0.0006

## F - G 173 90.5 1301 1.911 0.1614

##

## Results are averaged over the levels of: cut

## P value adjustment: scheffe method with dimensionality 2

pairs(emmeans::pmmeans(COVmodel, "cut"), adjust = "scheffe")

## NOTE: Results may be misleading due to involvement in interactions

## contrast estimate SE df t.ratio p.value

## Fair - Good -617 74.3 1301 -8.301 <.0001

##

## Results are averaged over the levels of: color

Let’s use ggplot to combine the two manually into one plot. We’ll plot  
the original model with dashed lines and the new model with covariate in  
dark bold lines.

withCOV <- broom::tidy(emmeans::pmmeans(COVmodel, "cut", by = "color"))

noCOV <- broom::tidy(emmeans::pmmeans(noCOVmodel, "cut", by = "color"))

ggplot(data = withCOV,

aes(x = color,

y = estimate,

group = cut,

color = cut)) +

geom\_point(shape = 18,

size = 4) +

geom\_line(size = 2) +

ggrepel::geom\_label\_repel(aes(label = round(estimate, 2)),

nudge\_x = -.35,

color = "black") +

geom\_point(data = noCOV,

aes(x = color,

y = estimate,

group = cut,

color = cut)) +

geom\_line(data = noCOV,

aes(x =color,

y = estimate,

group = cut,

color = cut),

linetype = 2) +

ggrepel::geom\_label\_repel(data = noCOV,

aes(label = round(estimate, 2)),

nudge\_x = .35,

color = "black") +

labs(title = "Estimated Mean Diamond Price",

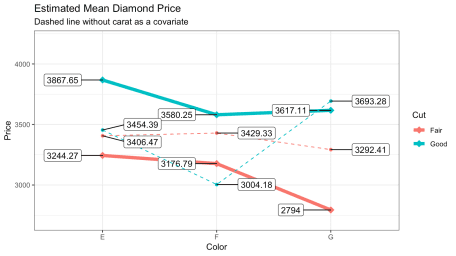
subtitle = "Dashed line without carat as a covariate",

color = "Cut") +

ylab("Price") +

xlab("Color") +

expand\_limits(y = c(2700,4200))



**Measuring the effects**

The final thing we’d like to do is to better our understanding of the  
“effect size” of color and cut when we control for carat.  
Earlier we ran sjstats::anova\_stats(car::Anova(COVmodel, type = 2)) %>%  
select(1:7) but that isn’t quite what we want, since the information  
about carat is still influencing our computations. We want to remove  
carat from the calculations. The trick to doing that in R with aov  
is to make carat an Error term. It sounds strange in some ways if  
you ask me, but it is effective in getting what we want.

Under the hood aov will create a model with two strata, Stratum 1 is  
all about our covariate carat. It pulls out or controls for its  
influence so that Stratum 2 which is labelled *“Within”* now contains  
the ANOVA for the other variables controlling for carat. The Within  
label always puts me off a bit since it makes me want to think of a  
within subjects design (which this clearly isn’t). But it’s having just  
the impact we like when you inspect the output. cut and color do  
matter! There’s hope we’ll get published in the *JODR* yet.

COVmodelError <- aov(price ~ cut \* color + Error(carat), diamonds2)

summary(COVmodelError)

##

## Error: carat

## Df Sum Sq Mean Sq

## cut 1 7.142e+09 7.142e+09

##

## Error: Within

## Df Sum Sq Mean Sq F value Pr(>F)

## cut 1 1.216e+08 121588173 68.513 3.09e-16 \*\*\*

## color 2 2.671e+07 13356821 7.526 0.000563 \*\*\*

## cut:color 2 9.593e+06 4796635 2.703 0.067393 .

## Residuals 1301 2.309e+09 1774684

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

car::Anova(COVmodelError$Within, type = 2)

## Anova Table (Type II tests)

##

## Response: price

## Sum Sq Df F value Pr(>F)

## cut 122611531 1 69.0892 2.342e-16 \*\*\*

## color 26713642 2 7.5263 0.0005625 \*\*\*

## cut:color 9593270 2 2.7028 0.0673932 .

## Residuals 2308863799 1301

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

broomExtra::glance(COVmodelError$Within)

## # A tibble: 1 x 11

## r.squared adj.r.squared sigma statistic p.value df logLik AIC

##

## 1 0.0640 0.0611 1332. 22.2 8.78e-18 5 -11247. 22505.

## # … with 3 more variables: BIC , deviance , df.residual

sjstats::anova\_stats(car::Anova(COVmodelError$Within, type = 2)) %>% select(1:7)

## term sumsq meansq df statistic p.value etasq

## 1 cut 122611531 122611531 1 69.089 0.000 0.050

## 2 color 26713642 13356821 2 7.526 0.001 0.011

## 3 cut:color 9593270 4796635 2 2.703 0.067 0.004

## 4 Residuals 2308863799 1774684 1301 NA NA NA

**What don’t we know?**

1. **That this example would apply to across the rest of the dataset.**  
   Remember that one of the steps we took early on was to select only  
   certain levels of our factors color and cut and even after that  
   to choose balance over completeness.
2. **How badly we violated key assumptions about our data.** We know we  
   have some evidence of heteroskedasticity and non-normality. I also,  
   for the sake of brevity, did not discuss interactions at the second  
   order between the covariate and our factors.