# Scatterplots

Scatterplots are the best way to present data that has a continuous response variable. When creating scatterplots, the idea is to show ALL of the data, and then show how your model is summarizing the relationships in that data.

## Setup

The code for today starts with the normal set of preliminaries, opening up the `els.RData` dataset and creating a codebook.

```
## Warning: package 'tidyverse' was built under R version 3.5.3

## -- Attaching packages --------------------------------------------------------------------------- tidyv

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## Warning: package 'ggplot2' was built under R version 3.5.3

## Warning: package 'tibble' was built under R version 3.5.3

## Warning: package 'tidyr' was built under R version 3.5.3

## Warning: package 'purrr' was built under R version 3.5.3

## Warning: package 'dplyr' was built under R version 3.5.3

## Warning: package 'stringr' was built under R version 3.5.3

## Warning: package 'forcats' was built under R version 3.5.3

## -- Conflicts ------------------------------------------------------------------------------ tidyverse_c
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## Warning: package 'modelr' was built under R version 3.5.3
```

## Bivariate Regression

We begin with a simple model of test scores as a function of socio-economic status.

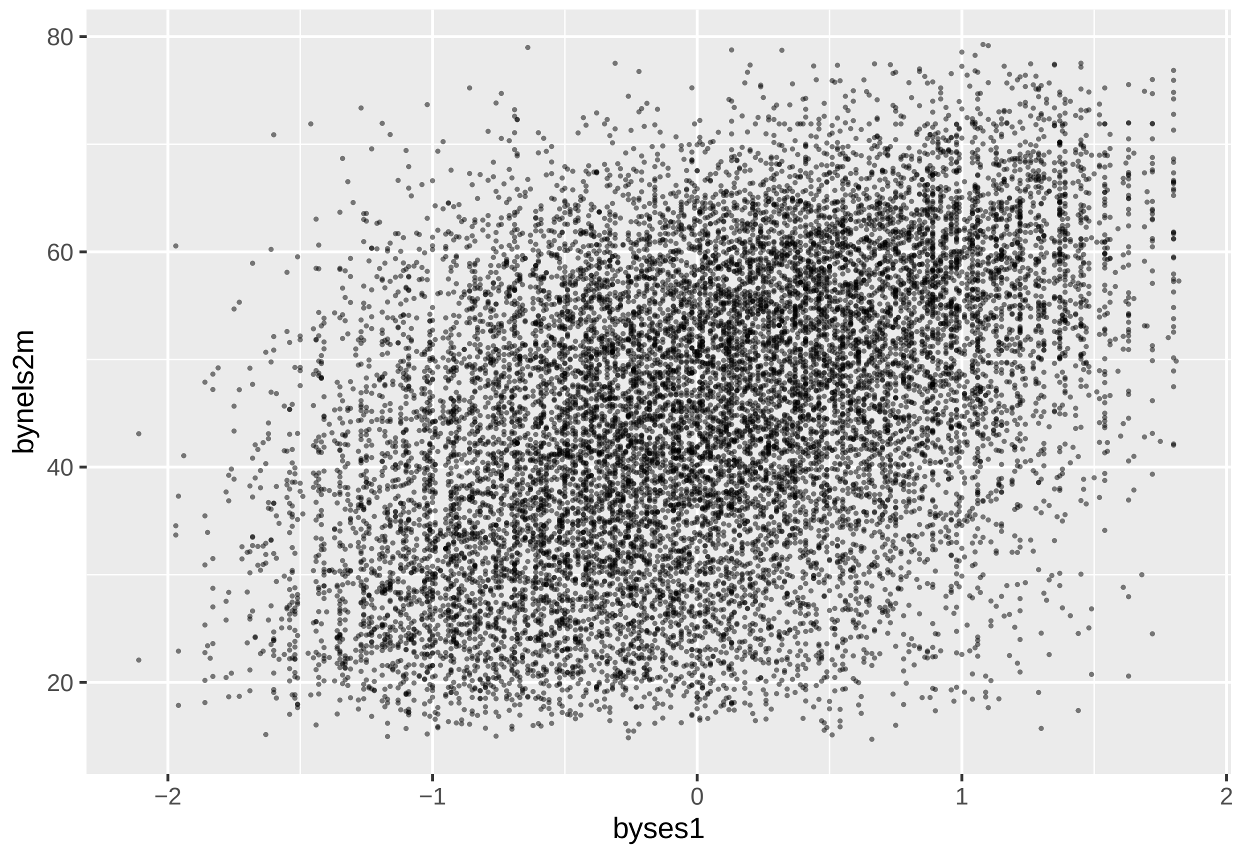
## Basics of Creating a Scatterplot

Our first step should be to plot the data. Today, we'll be using the `ggplot2` library, which is a highly functional implementation of what's known as the grammar of graphics. In a very small nutshell, the grammar of graphics refers to laying out a graphic in a series of layers. For our first scatterplot, we first specify the data that we'll be drawing on, then the "aesthetic" of the graphic, which will be based on our x and y variables from our regression. We then specify the first layer, which is a series of points defined by the intersection of the x and y variables.

```
g1<-ggplot(data=els,
           aes(x=byses1,y=bynels2m)
           )
```

```r
g1<-g1+geom_point(alpha=.5,size=.25) # Add points at x and y
g1
```

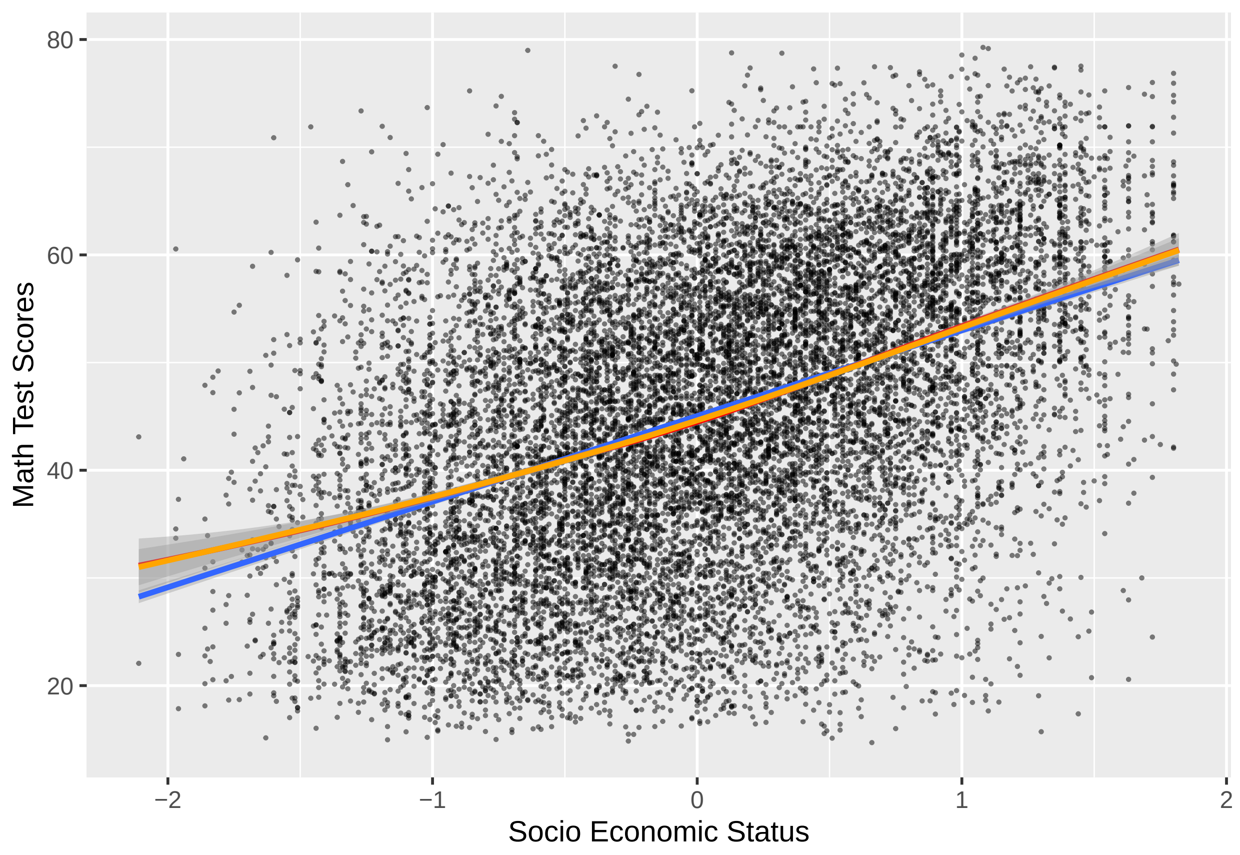
```
## Warning: Removed 964 rows containing missing values (geom_point).
```



So, this is a bit of a mess. It just looks like a blob. We need to fix it to make it more readable, but let's first get the next element we want, which is the regression line.

```r
g1<-g1+geom_smooth(method="lm")
g1<-g1+geom_smooth(method = "loess",color="red")
g1<-g1+geom_smooth(color="orange")
g1<-g1+ylab("Math Test Scores")+xlab("Socio Economic Status")
g1
```

```
## Warning: Removed 964 rows containing non-finite values (stat_smooth).

## Warning: Removed 964 rows containing non-finite values (stat_smooth).

## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

## Warning: Removed 964 rows containing non-finite values (stat_smooth).

## Warning: Removed 964 rows containing missing values (geom_point).
```

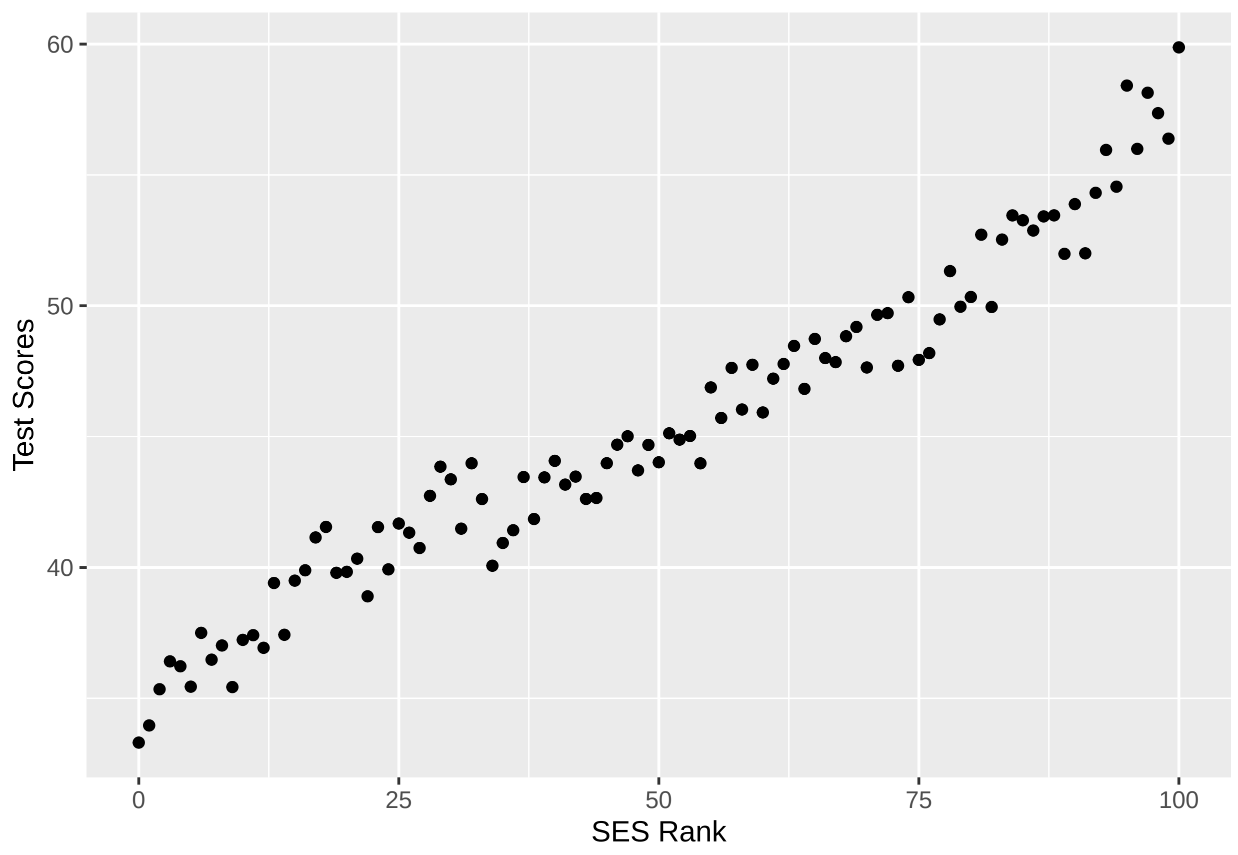


## Using Conditional Means to Create Scatterplots

It's also really hard to see. We can use conditional means to help out with that problem. Let's get the average amount of test scores at every percentile level of `byses1`. Notice the use of `round` to get income percentiles that are at two digits only.

```
els_sum<-els%>%
  mutate(ses_rank=percent_rank(byses1)*100)%>%
  mutate(ses_rank_r=round(ses_rank))%>%
  group_by(ses_rank_r)%>%
  summarize(test_mean=mean(bynels2m,na.omit=TRUE))

g1a<-ggplot(els_sum,aes(x=ses_rank_r,y=test_mean))

g1a<-g1a+geom_point()

g1a<-g1a+ylab("Test Scores")+xlab("SES Rank")

g1a
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



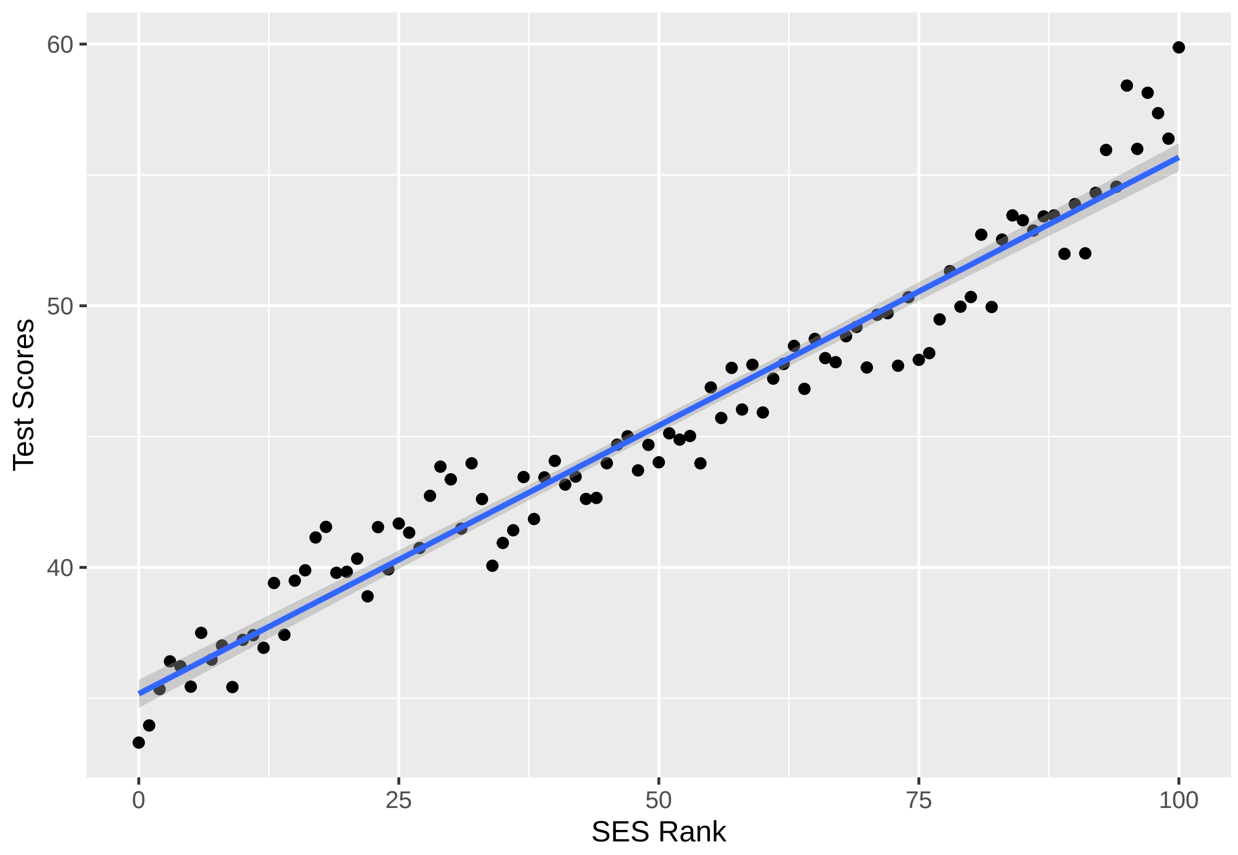
Better! Simplifying data can help.

We can add a regression line to this simpler data

```
g1b<-g1a+geom_smooth(method="lm") # Add a line
g1b
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



This summarizes the basic relationship nicely. We're ready to run the model and get results.

```
#First model

mod_1<-lm(bynels2m~byses1,data=els);summary(mod_1)
```

```
##
## Call:
## lm(formula = bynels2m ~ byses1, data = els)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.658  -8.801   0.400   9.048  39.042
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  45.0379     0.0993   453.5   <2e-16 ***
## byses1        7.9535     0.1335    59.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.27 on 15323 degrees of freedom
##   (964 observations deleted due to missingness)
## Multiple R-squared:  0.1882, Adjusted R-squared:  0.1881
## F-statistic:  3552 on 1 and 15323 DF,  p-value: < 2.2e-16
```

*Quick Exercise* Create a similar graphic, but this time use reading scores as the independent variable.

## Presenting Complex Results

The next step is to add covariates. I'll be working with the variable `bypared` which is a factor that summarizes the parental education of respondents. I'm going to set the color of the markers by the `bypared` factor.

```
g2<-ggplot(data=filter(els,is.na(bypared)==FALSE),
            aes(x=byses1,y=bynels2m,
                color=as.factor(bypared) #notice the color option
                ))
## Let's make the dots smaller for readability
g2<-g2+geom_point(size=.25)

## Changing the Legend
g2<-g2+theme(legend.position="bottom"  )#, legend.title =
              # element_blank())

g2<-g2+ylab("Test Scores")+xlab("Socio Economic Status")

g2 <- g2+ scale_color_discrete(name="parental Ed")
```

Our graphic is a bit complex, but shows the intersectionality between SES and parental education: there are very few students with low levels of parental education and/or high levels of SES or test scores.

We can see this same relationship in the model results:

## Using Scatterplots to Explain Models

```
#Model 2: with parental education

mod_2<-lm(bynels2m~
            byses1+
            as.factor(bypared),
          data=els); summary(mod_2)
```

```
##
## Call:
## lm(formula = bynels2m ~ byses1 + as.factor(bypared), data = els)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.518  -8.854   0.348   9.112  38.683
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          46.3023     0.4851  95.447  < 2e-16 ***
## byses1                8.6985     0.2357  36.903  < 2e-16 ***
## as.factor(bypared)2  -0.3953     0.4765  -0.830 0.406712
## as.factor(bypared)3  -1.9105     0.5487  -3.482 0.000499 ***
## as.factor(bypared)4  -0.8594     0.5612  -1.531 0.125676
## as.factor(bypared)5  -1.2607     0.5610  -2.247 0.024653 *
## as.factor(bypared)6  -1.5761     0.5872  -2.684 0.007285 **
## as.factor(bypared)7  -1.5823     0.6822  -2.320 0.020380 *
## as.factor(bypared)8  -3.3724     0.7722  -4.367 1.27e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.26 on 15316 degrees of freedom
##   (964 observations deleted due to missingness)
## Multiple R-squared:   0.19,  Adjusted R-squared:  0.1896
## F-statistic: 449.2 on 8 and 15316 DF,  p-value: < 2.2e-16
```

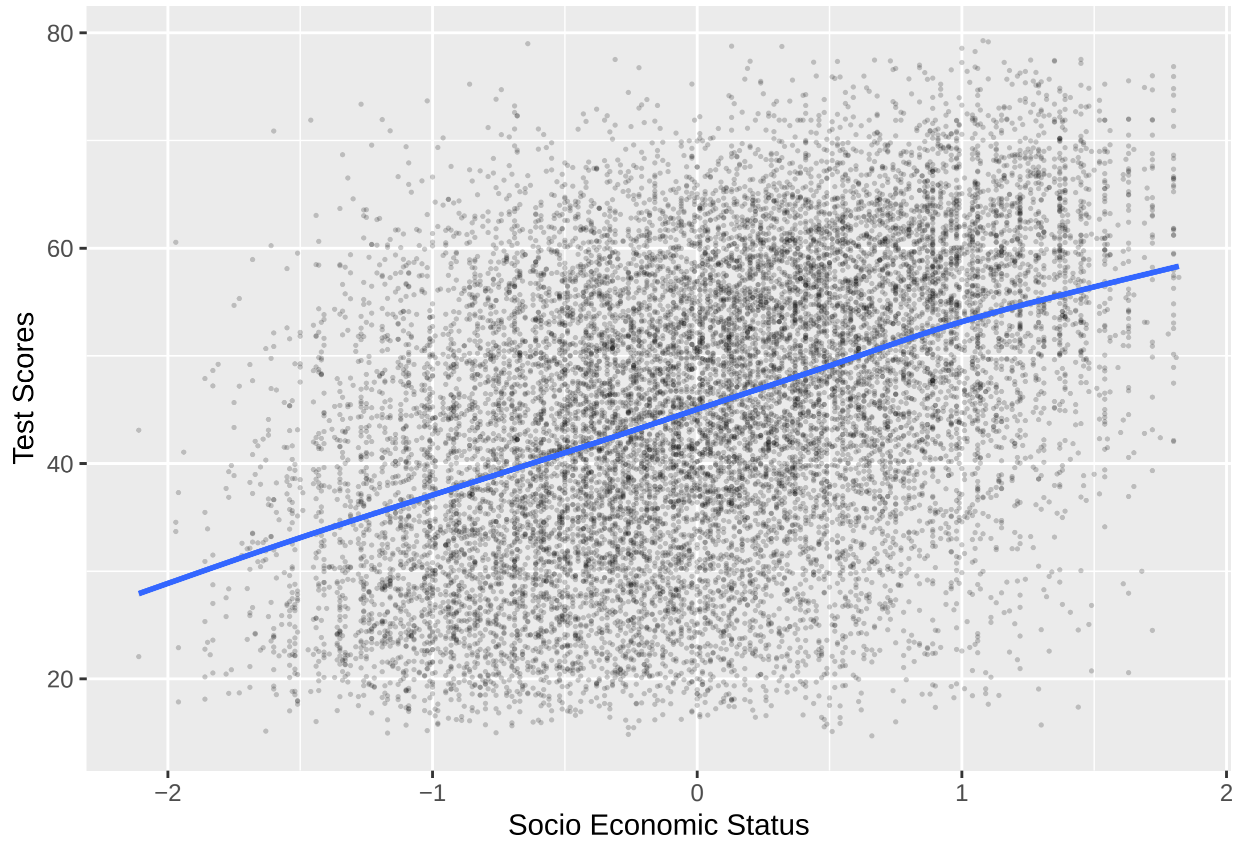
Now let's take a look at this model plotted against the actual data. I'm going to use the `alpha` setting to make the dots more transparent. I'm also going to make the dots smaller via the size specification.

```
els<-els%>%add_predictions(mod_2)%>%rename(pred_mod_2=pred)

g3<-ggplot(els,aes(x=byses1,y=bynels2m))
g3<-g3+geom_point(alpha=.2,size=.25)

g3<-g3+geom_smooth(data=els,(aes(x=byses1,y=pred_mod_2)))

g3<-g3+xlab("Socio Economic Status")+ylab("Test Scores")

g3
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
## Warning: Removed 964 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 964 rows containing missing values (geom_point).
```



As we add more variables to the model, it can get more difficult to plot relationships. One very good option is to plot lines based on a hypothetical set of data. Below, I create a hypothetical set of data that include values of SES across the range of SES, and includes values for every level of `bypared`. I then run predictions from this hypothetical data to get a prediction line for every level of parental education.

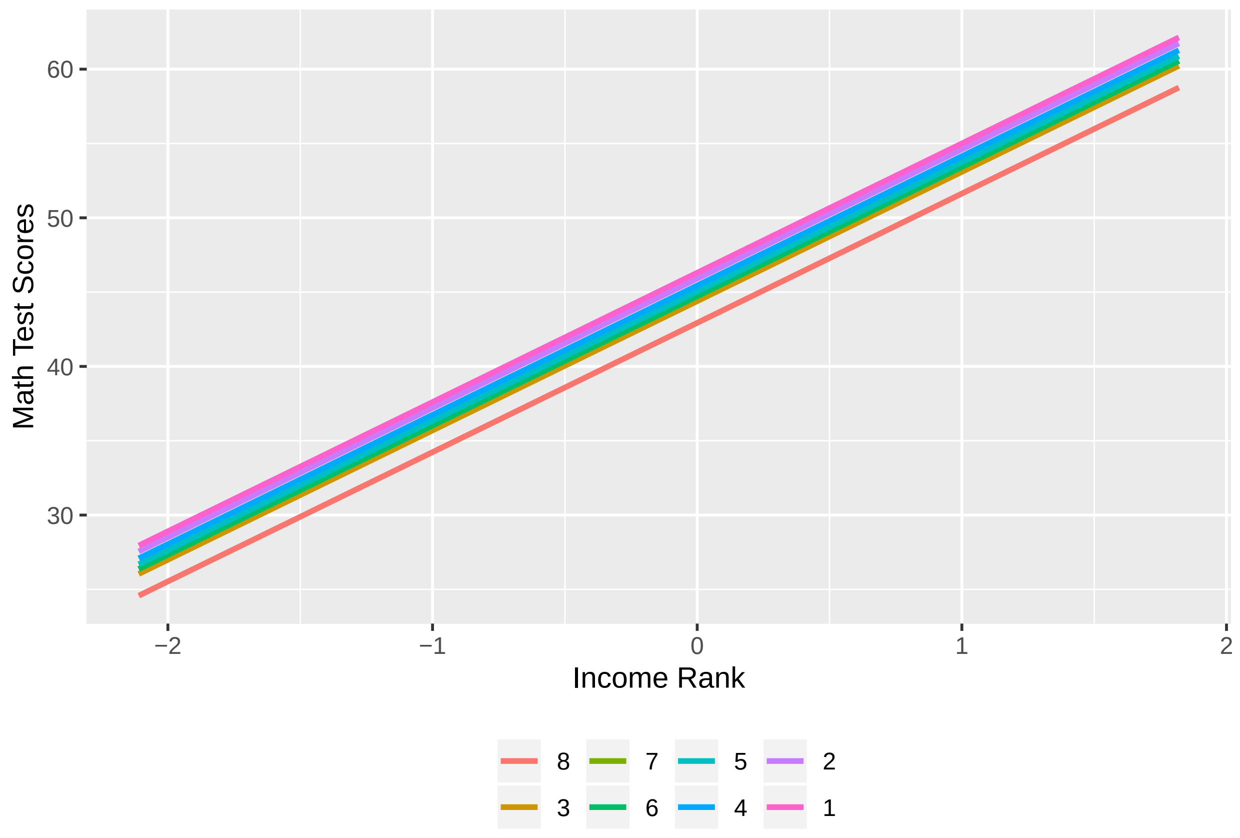
Now, using my estimates from model 2, I predict what would happen to these hypothetical individuals. Once I've got my prediction, I transform it back out of the log scale into the "response" level of dollars.

```
hypo_data<-data_grid(els, byses1 = seq_range(byses1,n=100),bypared)  %>% add_predictions(mod_2)
```

Now we can plot the result, using the `geom_smooth` layer to give us lines for every level of `childage`.

```
g4<-ggplot(data=hypo_data,
           aes(x=byses1,
               y=pred,
               color=fct_reorder(.f=as.factor(bypared),pred))) #notice color
g4<-g4+geom_smooth(method=lm,se=FALSE)
g4<-g4+theme(legend.position="bottom",legend.title = element_blank())
g4<-g4+xlab("Income Rank")+ylab("Math Test Scores")
g4
```

```
## Warning: Removed 100 rows containing non-finite values (stat_smooth).
```



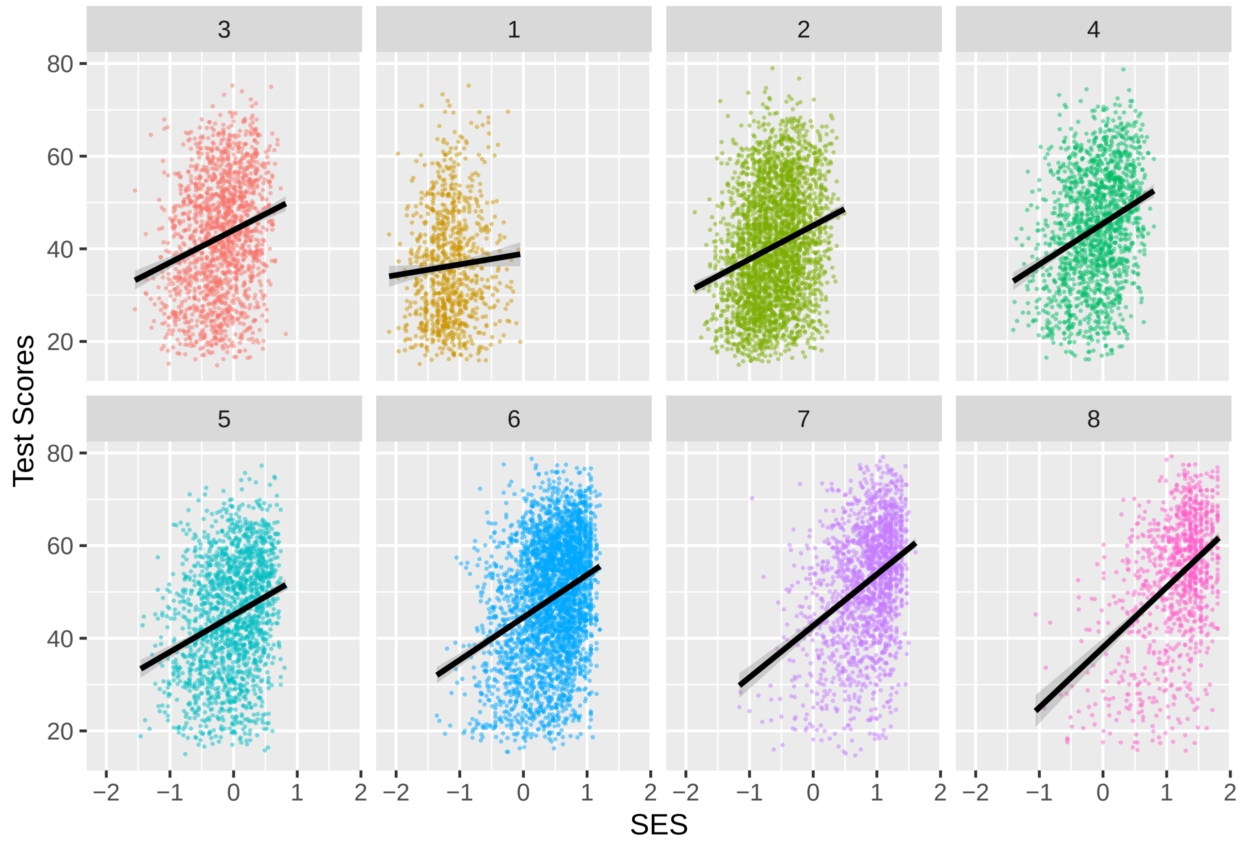
To show this in the data we can break it out for every type of parental education.

```
## Resort Parental Education for graphic
#els<-els%>%mutate(bypared = factor(bypared))
els<-els%>%mutate(bypared=reorder(x=bypared,bynels2m))
# filter nas!
g5<-ggplot(filter(els, is.na(bypared) == FALSE),aes(x=byses1,y=bynels2m, color=as.factor(bypared)))
g5<-g5+geom_point(alpha=.5,size=.1)
```

```r
g5<-g5+geom_smooth(method="lm",color="black")
g5<-g5+facet_wrap(~as.factor(bypared),nrow=2)
g5<-g5+xlab("SES")+ylab("Test Scores")
g5<-g5+theme(legend.position="none") #Suppress legend, not needed

g5
```

```
## Warning: Removed 68 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 68 rows containing missing values (geom_point).
```


Test Scores
SES