Example 3: General Linear Models with Multiple Slopes for One Predictor
(complete data, syntax, and output in R available online)

These example data were selected from the High School and Beyond 1980 dataset (in the `candisc` R package). The current example will use general linear models (estimated within the base R `lm` function) to examine associations of math with nominal program type (academic, general, or vocational), with a nonlinear trend for writing, and with ordinal four-category motivation. It will also describe how to obtain linear combinations of fixed effects to create predicted outcomes using the `glht` function from the `multcomp` R package.

Syntax for importing and preparing data for analysis (after loading packages `TeachingDemos`, `readxl`, `psych`, `multcomp`, and `supernova`):

```
# Set working directory (to import and export files to)
# Paste in the folder address where your data file is saved in quotes
# Note the slashes are backwards relative to Windows file paths
setwd("C:/Dropbox/26_EDF9770/Example3/")

# Import "HSB_Example.xlsx" from sheet "Sheet1" with first row as variable names
Example3 = read_excel(path="HSB_Example.xlsx", sheet="Sheet1", col_names=TRUE)
# Convert to data frame to use for analysis
Example3 = as.data.frame(Example3)

# Load R functions for this class from R file in working directory
source("EDF9770_Functions.R")
```

Exploring math differences by nominal program type

```
print("Frequencies for Categorical prog")
table(x=Example3$prog) # 1=general, 2=academic, 3=vocational

  1   2   3
145 308 147

# Work-around to add value labels for categorical variable prog
# Make a concatenated list of labels in order of values to be labeled
progLabels = c("1.General","2.Academic","3.Vocational")
# Make new text-format string variable with the labels instead of values
Example3$progLabeled = progLabels[Example3$prog]

print("Frequencies for Categorical progLabeled")
table(x=Example3$progLabeled)

  1.General   2.Academic 3.Vocational
        145          308          147

# Get and print means for math by program type
Example3_Prog = aggregate(x=Example3$math, by=list(Example3$progLabeled), FUN=mean)
colnames(Example3_Prog) = c("prog", "mathMean")  # Rename columns for clarity
Example3_Prog # Show result

          prog mathMean
1    1.General    49.09
2   2.Academic    55.81
3 3.Vocational    46.27
```

```
# Create indicator-coded binary predictors for program type
Example3$AcavGen=NA; Example3$AcavVoc=NA  # 2 new empty variables
Example3$AcavGen[which(Example3$prog==1)]=1  # Replace for general
Example3$AcavVoc[which(Example3$prog==1)]=0
Example3$AcavGen[which(Example3$prog==2)]=0  # Replace for academic
Example3$AcavVoc[which(Example3$prog==2)]=0
Example3$AcavGen[which(Example3$prog==3)]=0  # Replace for vocational
Example3$AcavVoc[which(Example3$prog==3)]=1
# AcavGen: Academic=0 vs General=1, AcavVoc: Academic=0 vs Vocational=1
```

program	AcavGen	AcavVoc
general	1	0
academic	0	0
vocational	0	1

Do math outcomes differ across the three types of programs?

Because program type is a nominal variable (stored as 1, 2, 3), we first need to recode it into two new binary variables so that 0 will be a meaningful value for the β_0 fixed intercept as one of the programs. In syntax, I tend to name binary variables with the letters in order of the 0 and 1 groups, as shown below. Alternatively, it is more common to name them after the group coded 1 (i.e., as demonstrated in the model equation).

$$math_i = \beta_0 + \beta_1(General_i) + \beta_2(Vocational_i) + e_i$$

```
print("GLM Predicting Math from 2 New Binary Variables for prog")
ModelProg = lm(data=Example3, formula=math~1+AcavGen+AcavVoc)
obj=LMsummary(ModelProg, explain=TRUE)  # Custom output
```

```
Sums of Squares Table
             SS  DF       MS      F      p    R2
Model 10515.135   2 5257.567 73.717 <0.001 0.198
Error 42578.583 597   71.321
Total 53093.718 599   88.637

Explanation:
SS = Sum of Squares, MS = Mean Square, DF = Degrees of Freedom,
F = F test-statistic, p = two-sided p-value, R2 = R-square
```

How much math variance is leftover after considering program type (i.e., what is σ_e^2)?

```
Fixed Effects Table
             Est    SE       t      p     LCI    UCI
Intercept 55.811 0.481 115.980 <0.001  54.866 56.756
AcavGen   -6.719 0.851  -7.900 <0.001  -8.389 -5.049
AcavVoc   -9.543 0.847 -11.272 <0.001 -11.205 -7.880

Explanation:
Est = Estimate, SE = Standard Error, t = t test-statistic,
p = p-value, LCI = Lower Confidence Interval,
UCI = Upper Confidence Interval
```

Interpret β_0 = intercept:

Interpret β_1 = slope of AcavGen:

Interpret β_2 = slope of AcavVoc:

Now let's get other quantities of interest—predicted means by category and mean differences:

$$\widehat{math}_i = \beta_0(1) + \beta_1(General_i) + \beta_2(Vocational_i)$$

Pred math for academic: $\widehat{math}_a = 55.811(1) - 6.719(0) - 9.543(0) = 55.811$
Pred math for general: $\widehat{math}_g = 55.811(1) - 6.719(1) - 9.543(0) = 49.092$
Pred math for vocational: $\widehat{math}_v = 55.811(1) - 6.719(0) - 9.543(1) = 46.268$

```
print("Get predicted math per category and category differences")
print("Values below are multipliers for each fixed effect IN ORDER")
PredProg = multcomp::glht(model=ModelProg, linfct=rbind(
  "Pred Math: Academic"    = c(1, 0, 0),  # Already in model
  "Pred Math: General"     = c(1, 1, 0),
  "Pred Math: Vocational"  = c(1, 0, 1),
  "Aca vs Gen Diff"        = c(0, 1, 0),  # Already in model
  "Aca vs Voc Diff"        = c(0, 0, 1),  # Already in model
  "Gen vs Voc Diff"        = c(0,-1, 1),

  "Pred Math: Gen&Voc Mean"   = c(1, 1/2, 1/2),  # Combined mean
  "Pred Math: Aca (repeat)"   = c(1,   0,   0),  # Repeated for comparison
  "Aca vs Gen&Voc Diff"       = c(0, 1/2, 1/2))) # Combined difference
obj=glhtSummary(glhtObject=PredProg, effectsizes=TRUE, explain=TRUE) # Custom output
```

```
Linear Combinations Table
                           Est    SE       t      p     LCI    UCI
Pred Math: Academic     55.811 0.481 115.980 <0.001  54.866 56.756
Pred Math: General      49.092 0.701  69.998 <0.001  47.714 50.469
Pred Math: Vocational   46.268 0.697  66.425 <0.001  44.900 47.636

Aca vs Gen Diff         -6.719 0.851  -7.900 <0.001  -8.389 -5.049
Aca vs Voc Diff         -9.543 0.847 -11.272 <0.001 -11.205 -7.880
Gen vs Voc Diff         -2.824 0.988  -2.857  0.004  -4.765 -0.882

Pred Math: Gen&Voc Mean 47.680 0.494  96.473 <0.001  46.709 48.651
Pred Math: Aca (repeat) 55.811 0.481 115.980 <0.001  54.866 56.756
Aca vs Gen&Voc Diff     -8.131 0.690 -11.787 <0.001  -9.486 -6.776


Effect Sizes for Linear Combinations Table
                           Est      p      d     pr    sR2
Pred Math: Academic     55.811 <0.001  9.494  0.979 18.069
Pred Math: General      49.092 <0.001  5.730  0.944  6.582
Pred Math: Vocational   46.268 <0.001  5.437  0.939  5.927

Aca vs Gen Diff         -6.719 <0.001 -0.647 -0.308  0.084
Aca vs Voc Diff         -9.543 <0.001 -0.923 -0.419  0.171
Gen vs Voc Diff         -2.824  0.004 -0.234 -0.116  0.011

Pred Math: Gen&Voc Mean 47.680 <0.001  7.897  0.969 12.502
Pred Math: Aca (repeat) 55.811 <0.001  9.494  0.979 18.069
Aca vs Gen&Voc Diff     -8.131 <0.001 -0.965 -0.435  0.187

Explanation:
Est = Estimate, p = p-value, d = Cohen's d,
pr = Partial r, sR2 = Semi-Partial R-square
```

Now let's see if a linear slope for writing is sufficient to predict math…

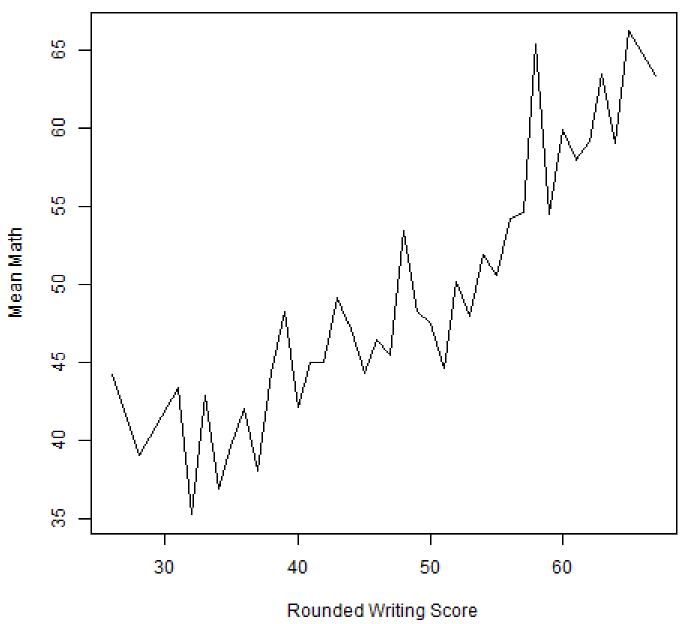
Previously in Example 2 we saw that writing had a significant linear relation with math. However, we should also examine a potential nonlinear relationship (i.e., if we need a bendy line). Let's start by exploring the data.

```
print("Descriptive Statistics for Quantitative Variables")
print(psych::describe(x=Example3[ , c("math","write")], fast=TRUE), digits=3)
```

```
      vars   n  mean    sd median  min  max range   skew kurtosis    se
math     1 600 51.85 9.415   51.3 31.8 75.5  43.7  0.263   -0.653 0.384
write    2 600 52.38 9.726   54.1 25.5 67.1  41.6 -0.470   -0.714 0.397
```

```
# Compute rounded version of write (by which to see mean math trend)
Example3$writeRound = round(Example3$write, digits=0)  # writeRound: Rounded Writing Score
# Get means for math by rounded writeRound
Example3_Round = aggregate(x=Example3$math, by=list(Example3$writeRound), FUN=mean)
colnames(Example3_Round) = c("writeRound", "mathMean")  # Rename columns for clarity

png(file = "Mean Math by Rounded Writing Plot.png")  # open file
plot(y=Example3_Round$mathMean, x=Example3_Round$writeRound,
     lty=1, type="l", ylab="Mean Math", xlab="Rounded Writing Score")
dev.off()  # close file
```



	writeRound	mathMean
1	26	44.30
2	28	39.04
3	31	43.44
4	32	35.30
5	33	42.94

…

Although there is definitely some linear relation between math and writing, it appears the relation may be flatter towards the lower end of writing (perhaps due to floor effects in the measures).

Previously we had centered write at 50 (near the mean). Here we center it near the minimum value instead to facilitate interpretation of the linear trend as the starting point within the quadratic model to follow.

```
# Center quantitative write variable to be used as a predictor
Example3$write26 = Example3$write-26  # write26: Writing Score (0=26 as min)
```

Let us first revisit the results using a linear slope for writing predicting math:

$$math_i = \beta_0 + \beta_1(write_i - 26) + e_i$$

```
print("GLM Predicting Math from Linear Centered Write 0=26 -- save as ModelLinWrite")
ModelLinWrite = lm(data=Example3, formula=math~1+write26)
obj=LMsummary(ModelLinWrite, explain=TRUE)  # Custom output with explanations
```

```
Sums of Squares Table
             SS  DF         MS       F      p    R2
Model 21251.652   1 21251.652 399.110 <0.001 0.400
Error 31842.066 598    53.248
Total 53093.718 599    88.637

Fixed Effects Table
             Est    SE      t      p    LCI    UCI
Intercept 35.691 0.862 41.409 <0.001 33.998 37.384
write26    0.612 0.031 19.978 <0.001  0.552  0.673
```

Now let's see the results using linear+quadratic slopes for writing predicting math:

$$math_i = \beta_0 + \beta_1(write_i - 26) + \beta_2(write_i - 26)^2 + e_i$$

```
print("GLM Predicting Math from Quadratic Centered Write 0=26 -- save as ModelQuadWrite")
print("I(x^2) squares predictor to create quadratic term")
ModelQuadWrite = lm(data=Example3, formula=math~1+write26+I(write26^2))
obj=LMsummary(ModelQuadWrite, explain=TRUE)  # Custom output with explanations
```

```
Sums of Squares Table
             SS  DF         MS       F      p    R2
Model 21898.241   2 10949.120 209.538 <0.001 0.412
Error 31195.477 597    52.254
Total 53093.718 599    88.637
```

How much math variance is leftover after considering writing (i.e., what is σ_e^2)?

```
Fixed Effects Table
                Est    SE      t      p    LCI    UCI
Intercept    40.798 1.684 24.224 <0.001 37.490 44.105
write26       0.101 0.149  0.677  0.499 -0.191  0.393
I(write26^2)  0.011 0.003  3.518 <0.001  0.005  0.017
```

Interpret β_0 = intercept:

Interpret β_1 = linear slope of write26:

Interpret β_2 = quadratic slope of write26:

Why did the results for β_1 change so much relative to the model with only a linear slope ?

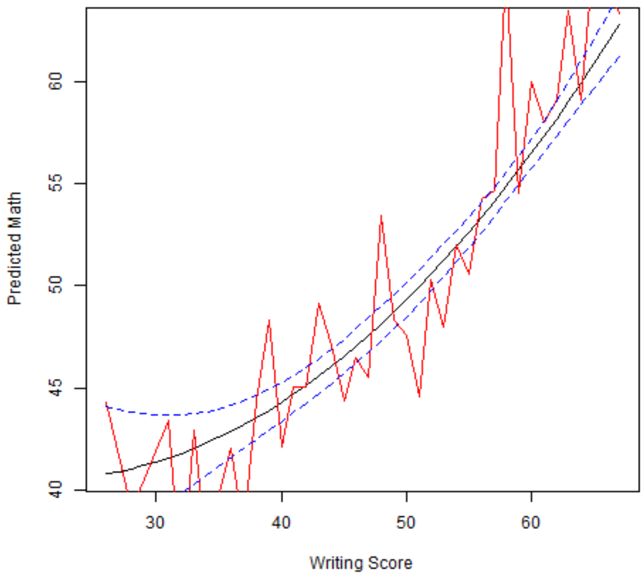
```
print("Compute aperture point from quadratic model results: [-1*linear/2*quad] + center")
(-1*ModelQuadWrite[["coefficients"]][["write26"]]) /
( 2*ModelQuadWrite[["coefficients"]][["I(write26^2)"]])+26
```

```
21.26 → writing score where slope=0 (not observed in this sample)
```

Generating predicted outcomes to show model-predicted quadratic trend:

```
# Generate predicted values using predict more efficiently and plot them
PredQuadWrite = data.frame(write26=seq(from=0, to=41, by=1))  # Write 26 to 67 (rounded min and max)
PredQuadWrite = predict(object=ModelQuadWrite, newdata=PredQuadWrite, se.fit=TRUE,
interval="confidence")
PredQuadWrite = as.data.frame(PredQuadWrite)  # Need to put x variables back in
PredQuadWrite = cbind(PredQuadWrite, data.frame(write26=seq(from=0, to=41, by=1)))
PredQuadWrite$write = PredQuadWrite$write26 + 26  # Add 26 back to get real write scores
```

```
png(file = "Predicted Math by Quadratic Writing Plot.png")  # open file
plot(y=PredQuadWrite$fit.fit, x=PredQuadWrite$write,
     lty=1, type="l", ylab="Predicted Math", xlab="Writing Score")
lines(y=Example3_Round$mathMean, x=Example3_Round$writeRound, lty=1, col="red1") # Add mean by rounded
lines(y=PredQuadWrite$fit.upr, x=PredQuadWrite$write, lty=2, col="blue1") # Upper CI
lines(y=PredQuadWrite$fit.lwr, x=PredQuadWrite$write, lty=2, col="blue1") # Lower CI
legend(x=20, y=5, legend=c("Predicted Math", "95% CI"), lty=1:2)
dev.off()  # close file
```



The red line is the mean per rounded writing as a point of comparison. The solid blue line is predicted math from quantitative writing (with dashed lines for confidence intervals).

Exploring math differences by ordinal motivation (four categories, labels unknown)

```
print("Descriptive Statistics for Categorical motiv")
table(x=Example3$motiv) # 1-4 ordinal scale

  1   2   3   4 
 67 120 170 243 

# Get and print means for math by motivation
Example3_Motiv = aggregate(x=Example3$math, by=list(Example3$motiv), FUN=mean)
colnames(Example3_Motiv) = c("motiv", "mathMean")  # Rename columns for clarity
Example3_Motiv # Show result

  motiv mathMean
1     1    48.89
2     2    49.53 → diff = 0.64
3     3    51.99 → diff = 2.46
4     4    53.71 → diff = 1.72 (average of differences = 1.61)

# Center ordinal motiv variable to be used as a predictor
Example3$motiv1 = Example3$motiv-1  # motiv1: motivation (0=1 as min)

print("GLM Predicting Math from linear slope of motiv")
ModelMotiv1 = lm(data=Example3, formula=math~1+motiv1)
obj=LMsummary(ModelMotiv1, explain=TRUE)  # Custom output

Sums of Squares Table
             SS  DF       MS      F      p    R2
Model  2016.168   1 2016.168 23.605 <0.001 0.038
Error 51077.550 598   85.414
Total 53093.718 599   88.637
```

```
Fixed Effects Table
             Est    SE      t      p    LCI    UCI
Intercept 48.306 0.821 58.836 <0.001 46.694 49.919
motiv1     1.788 0.368  4.858 <0.001  1.065  2.510
```

Now let's test whether a linear slope is sufficient to predict math from ordinal motivation!

Motivation was stored as an ordinal, four-category variable. Let us test empirically whether it makes sense to treat it as interval instead of giving it a single linear slope in predicting math (which assumes equal successive differences). To do so, we will build binary predictors that capture sequential differences among the ordinal scale.

```
# Create indicator-binary-coded predictors for motiv
Example3$m1v2=NA; Example3$m2v3=NA; Example3$m3v4=NA  # Make 3 new empty variables
# Replace each with 0 values
Example3$m1v2[which(Example3$motiv<2)]=0
Example3$m2v3[which(Example3$motiv<3)]=0
Example3$m3v4[which(Example3$motiv<4)]=0
# Replace each with 1 values
Example3$m1v2[which(Example3$motiv>=2)]=1
Example3$m2v3[which(Example3$motiv>=3)]=1
Example3$m3v4[which(Example3$motiv>=4)]=1
# m1v2: Slope from motiv 1 to 2
# m2v3: Slope from motiv 2 to 3
# m3v4: Slope from motiv 3 to 4
```

motiv	m1v2	m2v3	m3v4
1	0	0	0
2	1	0	0
3	1	1	0
4	1	1	1

$$math_i = \beta_0 + \beta_1(m1v2_i) + \beta_2(m2v3_i) + \beta_3(m3v4_i) + e_i$$

```
print("GLM Predicting Math from 3 New Binary Variables for motiv")
ModelMotiv = lm(data=Example3, formula=math~1+m1v2+m2v3+m3v4)
obj=LMsummary(ModelMotiv, explain=TRUE)  # Custom output

Sums of Squares Table
             SS  DF      MS     F      p    R2
Model  2080.030   3 693.343 8.100 <0.001 0.039
Error 51013.688 596  85.593
Total 53093.718 599  88.637

Fixed Effects Table
             Est    SE      t      p    LCI    UCI
Intercept 48.891 1.130 43.256 <0.001 46.671 51.111
m1v2       0.636 1.411  0.451  0.652 -2.135  3.408
m2v3       2.463 1.103  2.232  0.026  0.296  4.629
m3v4       1.722 0.925  1.862  0.063 -0.094  3.539
```

What do the slopes represent in this sequential coding scheme?

Generating predicted outcomes and slope comparisons:

$$\widehat{math}_i = \beta_0(1) + \beta_1(m1v2_i) + \beta_2(m2v3_i) + \beta_3(m3v4_i)$$

Pred math for motiv=1: $\widehat{math} = 48.891(1) + 0.636(0) + 2.463(0) + 1.722(0) = 48.891$

Pred math for motiv=2: $\widehat{math} = 48.891(1) + 0.636(1) + 2.463(0) + 1.722(0) = 49.527$

Pred math for motiv=3: $\widehat{math} = 48.891(1) + 0.636(1) + 2.463(1) + 1.722(0) = 51.990$

Pred math for motiv=4: $\widehat{math} = 48.891(1) + 0.636(1) + 2.463(1) + 1.722(1) = 53.712$

```
print("Get predicted math per category and category differences")
print("Values below are multipliers for each fixed effect IN ORDER")
PredMotiv = multcomp::glht(model=ModelMotiv, linfct=rbind(
  "Pred Math: motiv=1"  = c(1, 0, 0, 0),  # Already in model
  "Pred Math: motiv=2"  = c(1, 1, 0, 0),
  "Pred Math: motiv=3"  = c(1, 1, 1, 0),
  "Pred Math: motiv=4"  = c(1, 1, 1, 1),

  "m1v2 vs m2v3 Diff"   = c(0,-1, 1, 0),
  "m2v3 vs m3v4 Diff"   = c(0, 0,-1, 1)))
obj=glhtSummary(glhtObject=PredMotiv, effectsizes=TRUE) # Custom output
```

motiv	m1v2	m2v3	m3v4
1	0	0	0
2	1	0	0
3	1	1	0
4	1	1	1

```
Linear Combinations Table
                      Est    SE      t      p    LCI    UCI
Pred Math: motiv=1 48.891 1.130 43.256 <0.001 46.671 51.111
Pred Math: motiv=2 49.527 0.845 58.643 <0.001 47.869 51.186
Pred Math: motiv=3 51.990 0.710 73.270 <0.001 50.596 53.384
Pred Math: motiv=4 53.712 0.593 90.502 <0.001 52.547 54.878

m1v2 vs m2v3 Diff   1.826 2.153  0.848  0.397 -2.402  6.054
m2v3 vs m3v4 Diff  -0.740 1.755 -0.422  0.673 -4.187  2.706

Effect Sizes for Linear Combinations Table
                      Est      p      d     pr    sR2
Pred Math: motiv=1 48.891 <0.001  3.544  0.871  3.016
Pred Math: motiv=2 49.527 <0.001  4.804  0.923  5.544
Pred Math: motiv=3 51.990 <0.001  6.002  0.949  8.655
Pred Math: motiv=4 53.712 <0.001  7.414  0.965 13.204

m1v2 vs m2v3 Diff   1.826  0.397  0.069  0.035  0.001
m2v3 vs m3v4 Diff  -0.740  0.673 -0.035 -0.017  0.000
```

Example Results Section
(although it's more verbose than would be typical for the sake of completeness here):

The extent to which math could be predicted from nominal program type, quantitative writing, and ordinal motivation was examined in separate general linear models. All analyses were conducted using the lm function in R v. 4.5.2. Predicted outcomes and linear combinations of the fixed effects were generated using the glht function within the multcomp package v. 1.4-29.

We used a general linear model (i.e., analysis of variance) to examine the extent to which math outcomes could be predicted from nominal program type. To do so, we created two binary contrasts to distinguish the three categories, in which academic program served as the reference to be compared to general and vocational programs. Cohen's *d* standardized mean differences were then computed from the *t* test-statistics to index effect size per slope. Program type significantly predicted math, $F(2, 597) = 73.72$, MSE = 71.32, $p < .001$, $R^2 = .198$. Relative to students in an academic program (mean = 55.81, SE = 0.48), math was significantly lower on average for students in either a general program (mean = 49.09, SE = 0.70; difference = −6.72, SE = 0.85, *d* = −0.65) or a vocational program (mean = 46.27, SE = 0.70; difference = −9.54, SE = 0.85, *d* = −0.92). In addition, math was significantly lower for students in a vocational program than for students in a general program (difference = −2.82, SE = 0.99, *d* = −0.23).

We used a general linear model (i.e., linear regression) to examine the extent to which math outcomes could be predicted from writing scores. We first examined the writing-specific means for the math outcome to identify plausible types of nonlinear associations. Given the apparent curvilinear trend (in which writing appeared less associated with math at lower levels of writing), we fit a model including fixed linear and quadratic slopes for writing (centered such that 0 = 26, near the sample minimum). The quadratic writing model captured a significant amount of variance in math, $F(2, 597) = 209.54$, MSE = 52.25, $p < .001$, $R^2 = .412$. The quadratic writing model was also a significant improvement over a linear writing model ($R^2 = .400$), as indicated by the significant quadratic fixed slope of writing. The model fixed effects can be interpreted as follows. The fixed intercept indicated that at writing = 26, math was predicted to be 40.80 (SE = 1.584) and to become nonsignificantly greater by 0.10 per unit writing (i.e., the instantaneous linear slope for writing at writing = 26; SE = 0.15, $p = .499$). However, the linear writing slope was predicted to become significantly more positive per unit writing by twice the quadratic coefficient of −0.011 (SE = 0.003, $p < .001$). As given by the quantity (−1*linear slope) / (2*quadratic slope) + 26, the age of minimum predicted math was 21.26 (i.e., the age at which the linear writing slope is predicted to be 0), which was outside the range of writing scores observed in the current sample.

We then used a general linear model (i.e., linear regression) to examine the extent to which math outcomes could be predicted from four-category ordinal motivation (scaled 1–4). In first examining a linear effect of motivation (centered at 1), the model fixed effects indicated that math was predicted to be 48.31 (SE = 1.54) for motivation = 1 (i.e., as given by the fixed intercept), and that math was predicted to be significantly higher by 1.79 (SE = 0.37, $p < .001$, $R^2 = .038$) per additional ordinal level of motivation. However, given that a linear slope for motivation assumes interval differences with respect to predicted math, we tested this assumption by specifying a model by which to estimate all sequential differences in predicted math by ordinal level of motivation. The revised model—predicting three sequential differences across the four levels of motivation—also captured a significant amount of variance in math, $F(3, 596) = 8.10$, MSE = 85.59, $p < .001$, $R^2 = .039$. The model fixed effects indicated that math was predicted to be 48.89 (SE = 1.13) for motivation = 1 (i.e., as given by the fixed intercept). Math was nonsignificantly higher by 0.64 (SE = 1.41, $p = .652$) for motivation = 2 than 1, significantly higher by 2.46 (SE = 1.10, $p = .026$) for motivation = 3 than 2, and nonsignificantly higher by 1.72 (SE = 0.93, $p = .063$) for motivation = 4 than 3. However, neither of the differences between these adjacent differences were significant (as given by linear combinations of the model fixed effects, requested separately). Further, the model R^2 increased by only .001 after treating motivation as ordinal (via three separate slopes), indicating that a linear slope appears to be sufficient.