R-5Ar, Simple slope analysis of interactions

Prof Colleen F Moore
Montana State University
February 2014

Overview: This handout describes 'simple slope' tests as a follow-up to a significant interaction in multiple regression. For the academic background the classic source is Cohen & Cohen, Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences. See Handout R4 for simple effect tests as an interaction follow-up in ANOVA. Simple slope tests are analogous to simple effects.

This handout uses the data set called 'hmwk1' which is in an excel sheet listed with this (same data as for handout R5r or R2Graphs).

Outline:

Quick look at code:

```
-- Download and install the 'pequod' package.
> library(pequod)
> mod5=lmres(dv~Dist*Train, centered=c("Dist", "Train"), data=hwk1)
> slopedist=simpleSlope(mod5, pred="Dist", mod1="Train") #mod1 refers to
moderator variable, mod5 is what I called my model fit by 'lmres'. You can
reverse the roles of the predictor and moderator, of course
> summary(slopedist) # gives a summary of the simple slope test
> PlotSlope(slopedist) # plots the simple slopes for +/- 1 s.d. for whatever
variable you ran the 'simpleSlope' function on.
```

I. Bring in the data, check it over.

```
> hwk1=read.table(pipe("pbpaste"),header=T) # I pasted from the clipboard
> attach(hwk1) # I like to attach, there are dangers
> summary(hwk1)  # summary, tells us if there is missing data. There isn't.
       dv              Dist          Train
 Min.   : 1.000   Min.   :1.00   Min.   :2
 1st Qu.: 3.750   1st Qu.:1.75   1st Qu.:2
 Median : 5.000   Median :2.50   Median :4
 Mean   : 5.861   Mean   :2.50   Mean   :4
 3rd Qu.: 7.000   3rd Qu.:3.25   3rd Qu.:6
 Max.   :16.000   Max.   :4.00   Max.   :6
```

```
> nrow(hwk1)  # how many observations do we have?
[1] 36

> cor(hwk1) # correlation matrix
# because the data are from a factorial design that is balanced (equal n), Dist
and Train are  uncorrelated. Dist is distance run in a test, and Train is
number of weeks of training prior to the test.

              dv      Dist      Train
dv     1.0000000 0.7319437 -0.4618687
Dist   0.7319437 1.0000000  0.0000000
Train -0.4618687 0.0000000  1.0000000
```

II. Do some statistical analyses

Handout R-5r shows this example analyzed as a factorial ANOVA with contrast codes constructed by hand, and R2Graphs shows graphs. Here we first construct an additive regression model of Dist and Train, then we add the interaction.

An important question is how to interpret the regression coefficient of an interaction term. This relates to the question of whether to center the variables, etc.

II. A. Additive regression with original numerical values of predictors.

Remember that order can matter in regression, unless your predictors are orthogonal. Here Dist and Train are orthogonal, so we don't worry about order.

```
> mod1=lm(dv~Dist+Train) # additive regression model
> summary(mod1);anova(mod1)  # get summary and analysis of regression table

Call:
lm(formula = dv ~ Dist + Train)

Residuals:
    Min      1Q  Median      3Q     Max
-4.1500 -0.9833  0.0500  0.9653  4.4389

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.0000     1.1125   3.595  0.00104 **
Dist          2.4111     0.2873   8.394 1.07e-09 ***
Train        -1.0417     0.1967  -5.297 7.69e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.927 on 33 degrees of freedom
Multiple R-squared:  0.7491,  Adjusted R-squared:  0.7339
F-statistic: 49.25 on 2 and 33 DF,  p-value: 1.238e-10

Analysis of Variance Table

Response: dv
          Df Sum Sq Mean Sq F value   Pr(>F)
Dist       1 261.61 261.606  70.454 1.07e-09 ***
```

```
Train       1 104.17 104.167  28.054 7.69e-06 ***
Residuals 33 122.53   3.713
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

II.B. Additive regression with centered predictors.
The results should be the same as above, but the coefficient for the grand mean will change.

```
> traincent=Train-4; distcent=Dist-2.5  # subtract the means
> summary(distcent) # check that mean is now zero
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  -1.50   -0.75    0.00    0.00    0.75    1.50
> summary(traincent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     -2      -2       0       0       2       2

# construct new model. The regression coeffs will differ
> mod2=lm(dv~traincent+distcent)
> summary(mod2); anova(mod2)

Call:
lm(formula = dv ~ traincent + distcent)

Residuals:
    Min      1Q  Median      3Q     Max
-4.1500 -0.9833  0.0500  0.9653  4.4389

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.8611     0.3212  18.250  < 2e-16 ***
traincent    -1.0417     0.1967  -5.297 7.69e-06 ***
distcent      2.4111     0.2873   8.394 1.07e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.927 on 33 degrees of freedom
Multiple R-squared:  0.7491,  Adjusted R-squared:  0.7339
F-statistic: 49.25 on 2 and 33 DF, p-value: 1.238e-10

Analysis of Variance Table

Response: dv
          Df Sum Sq Mean Sq F value   Pr(>F)
traincent  1 104.17 104.167  28.054 7.69e-06 ***
distcent   1 261.61 261.606  70.454 1.07e-09 ***
Residuals 33 122.53   3.713
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## Now that the predictors are centered the coefficient of the intercept is
the grand mean. The coefficients of traincent and distcent are the same as
the coeffs of Train and Dist in the first run. But remember, the
coefficients will be multiplied by different predictor values now, so the
```

predicted values from mod1 and mod2 should match. Calculate a couple of predicted values by hand to show this.

II.C. Model with interaction using raw (un-centered) values as predictors

```
# construct product of Train * Dist
> TxD=Train*Dist
> summary(TxD) # notice that the product term is skewed
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0     5.5     8.0    10.0    13.0    24.0

> mod3a=lm(dv~Train+Dist+TxD)  # include interaction last
> summary(mod3a);anova(mod3a)

Call:
lm(formula = dv ~ Train + Dist + TxD)

Residuals:
    Min      1Q  Median      3Q     Max
-4.3500 -0.7833  0.2167  1.2722  3.8389

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.0000     2.0718   0.965 0.341612
Train        -0.5417     0.4795  -1.130 0.267049
Dist          3.2111     0.7565   4.245 0.000175 ***
TxD          -0.2000     0.1751  -1.142 0.261842
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.918 on 32 degrees of freedom
Multiple R-squared:  0.7589,  Adjusted R-squared:  0.7363
F-statistic: 33.57 on 3 and 32 DF,  p-value: 5.295e-10

Analysis of Variance Table

Response: dv
          Df Sum Sq Mean Sq F value    Pr(>F)
Train      1 104.17 104.167 28.3126 7.809e-06 ***
Dist       1 261.61 261.606 71.1046 1.233e-09 ***
TxD        1   4.80   4.800  1.3046    0.2618
Residuals 32 117.73   3.679
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# the interaction isn't significant. Compare to Handout R-5r for ANOVA
results. If we treat this as a 4x3 design, then the error term has a much
lower SS. This shows the statistical power advantage of a designed
experiment over an observational one (usually).
# Notice also that the coeffs for the Intercept, Train and Dist changed
relative to mod1.
```

check the correlation matrix. To do this, it is easiest to make a new dataframe

```
> new1=data.frame(dv,Train,Dist,TxD) # puts the values into 'new1'
```

```
> cor(new1)
              dv      Train       Dist       TxD
dv     1.0000000 -0.4618687 0.7319437 0.1908064
Train -0.4618687  1.0000000 0.0000000 0.6454972
Dist   0.7319437  0.0000000 1.0000000 0.7071068
TxD    0.1908064  0.6454972 0.7071068 1.0000000

# notice that the interaction term is correlated with both Train and Dist
# because it is a factorial design, it should be orthogonal to them.

# Note: if you have R calculate the interaction for you starting from un-
centered predictor variables, > modx=lm(dv~Train*Dist) the results will be
identical to those here
```

II.D. Interaction model with centered variables

How do you center an interaction variable? Is it best to multiply centered predictors together, or should you first multiply raw predictors and then center the product? Having a factorial design helps us here because we know that the interaction term should be uncorrelated with the two main effect predictors. **A:** first center the predictors, then multiply the centered variables to create the interaction variable.

```
# we created 'traincent' and 'distcent' earlier by subtracting the mean from
the original variables. Now we create the interaction term.


> cTxD=traincent*distcent
> summary(cTxD)  # check that the mean is zero. Notice it is not skewed like
the product of the raw variables was
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     -3      -1       0       0       1       3

# check the correlation matrix

> new2=data.frame(dv,traincent,distcent,cTxD) # put values in new2
> cor(new2)
                  dv  traincent  distcent        cTxD
dv         1.00000000 -0.4618687 0.7319437 -0.09914591
traincent -0.46186872  1.0000000 0.0000000  0.00000000
distcent   0.73194365  0.0000000 1.0000000  0.00000000
cTxD      -0.09914591  0.0000000 0.0000000  1.00000000

# Now the interaction term is orthogonal to the two 'main effect' terms


> mod4=lm(dv~traincent+distcent+cTxD)
> summary(mod4);anova(mod4)

Call:
lm(formula = dv ~ traincent + distcent + cTxD)

Residuals:
    Min      1Q  Median      3Q     Max
-4.3500 -0.7833  0.2167  1.2722  3.8389
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.8611     0.3197  18.334  < 2e-16 ***
traincent    -1.0417     0.1958  -5.321 7.81e-06 ***
distcent      2.4111     0.2859   8.432 1.23e-09 ***
cTxD         -0.2000     0.1751  -1.142    0.262
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.918 on 32 degrees of freedom
Multiple R-squared:  0.7589,  Adjusted R-squared:  0.7363
F-statistic: 33.57 on 3 and 32 DF,  p-value: 5.295e-10

Analysis of Variance Table

Response: dv
          Df Sum Sq Mean Sq F value    Pr(>F)
traincent  1 104.17 104.167 28.3126 7.809e-06 ***
distcent   1 261.61 261.606 71.1046 1.233e-09 ***
cTxD       1   4.80   4.800  1.3046    0.2618
Residuals 32 117.73   3.679
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


# Notice that the coefficient of the intercept is now the grand mean. The
coefficients of ‘traincent’ and ‘distcent’ are now the same as what they
were in the analyses in II.A. and II.B. The significance tests of Train and
Dist are consistent across analyses because it is an orthogonal design.
```

III. Simple slopes.

I am not going to let the non-significance of the Train x Distance interaction stop me from doing the simple slope test because the means of the data plot as an interaction when analyzed by ANOVA, and the interaction reaches significance. (See Handout R-5r).

III.A. Use package ‘pequod’.

```
# activate package ‘pequod’ in memory (you previously installed it)

> library(pequod) # notice ‘dependencies’ on ggplot2 and car
Loading required package: ggplot2
Loading required package: car
```

Step 1. Construct a model using the ‘lmres’ function in ‘pequod’. The pequod package will center the variables for you, or you can center them yourself. Let’s make sure ‘pequod’ gets the same results when we tell it to center the variables.

```
> mod5=lmres(dv~Dist*Train, centered=c("Dist", "Train"), data=hwk1)
> summary(mod5)
Formula:
dv ~ Dist + Train + Dist.XX.Train
<environment: 0x11da98cd0>
```

```
Models
         R     R^2   Adj. R^2     F     df1  df2  p.value
Model  0.871  0.759     0.736 33.574  3.000   32 5.3e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residuals
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-4.3500 -0.7833  0.2167  0.0000  1.2720  3.8390

Coefficients
              Estimate   StdErr  t.value    beta p.value
(Intercept)    5.86111  0.31969 18.33397          <2e-16 ***
Dist           2.41111  0.28594  8.43235  0.7319  <2e-16 ***
Train         -1.04167  0.19577 -5.32096 -0.4619   1e-05 ***
Dist.XX.Train -0.20000  0.17510 -1.14221 -0.0992  0.2618
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Collinearity
              VIF Tolerance
Dist            1         1
Train           1         1
Dist.XX.Train   1         1
```

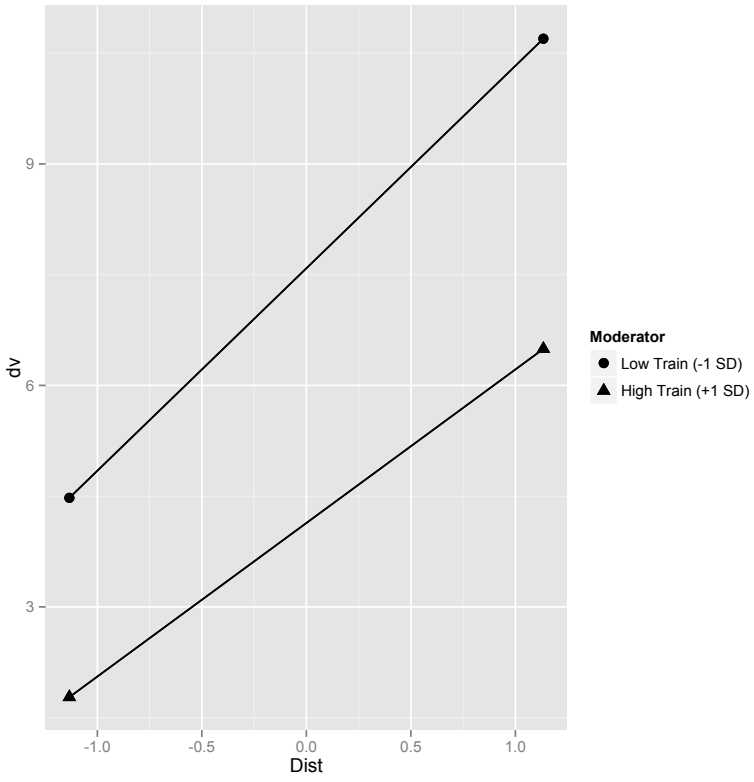
Step 2. Ask for a simple slope test on the model constructed in Step 1.

```
> slopedist=simpleSlope(mod5, pred="Dist", mod1="Train")
> summary(slopedist)  # get the summary of the 'simpleSlope' function

** Estimated points of dv  **

                   Low Dist (-1 SD) High Dist (+1 SD)
Low Train (-1 SD)             4.4768            10.6958
High Train (+1 SD)            1.7776             6.4943



** Simple Slopes analysis ( df= 32 ) **

                   simple slope standard error t-value p.value
Low Train (-1 SD)         2.742          0.407    6.73  <2e-16 ***
High Train (+1 SD)        2.080          0.407    5.11  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1



** Bauer & Curran 95% CI **

      lower CI upper CI
Train  -16.646   4.2473
```

```
## The simple slope tests the significance of the Dist variable at +/- 1
s.d. of the moderator variable (in this case Train. Both slopes are
significant. The simple slope test does not test the difference between the
slopes.
```

Step 3. Plot the simple slopes.

```
> PlotSlope(slopedist) # name the result of the simpleSlope from Step 2.
# The summary above gives the estimated data points that are used in
plotting this graph.
```



III.B. Calculate slopes by hand for +/- 1 s.d. of each predictor.

```
The regression equation is:
Eq 1: Y = 5.8611 + 2.4111*Dist + (-1.0467)*Train + (-.2000)*(Dist*Train),
where the variables are centered.
      Cohen & Cohen show that with a little algebra this is equivalent to:
Eq 2: Y = 5.8611 + [2.4111 + (-.2000)*Train]*Dist + (-1.0467)*Train
      Calculate first for Dist 1sd above its mean, and Train 1sd above its
mean. Use the centered variables we created earlier, 'traincent' and
'distcent'.
      First, find the sd of 'traincent' and 'distcent'.
> sd(traincent)
[1] 1.656157
```

```
> sd(distcent)
[1] 1.133893
```

To find the first value use 1.6562 for Train, and 1.1339 for Dist, and plug in to Eq 2.

```
> y11=5.8611+(2.4111 + (-.2*1.6562))*1.133893 + (-1.0467*1.656157)
> y11
[1] 6.485939
# this matches (within rounding) the High Dist High Train value above

# repeat flipping the signs sequentially for Dist and Train to calculate with – 1 s.d.

> y12=5.8611+(2.4111 + (-.2*1.6562))*(-1.133893) + (-1.0467*1.656157)
> y12
[1] 1.769262  # this is High Train, Low Dist

>  y21=5.8611+(2.4111 + (-.2*(-1.6562)))*(1.133893) + (-1.0467*(-1.656157))
> y21
[1] 10.70412  # this is Low Train, High Dist

> y22=5.8611+(2.4111 + (-.2*(-1.6562)))*(-1.133893) + (-1.0467*(-1.656157))
> y22
[1] 4.485079   # this is Low Train, Low Dist
```

III.C. Calculate slopes by hand for arbitrary values of the centered predictor variables.

There is nothing sacred about +/- 1 s.d., especially where the levels of the predictor variables have been chosen deliberately.

Step 1. Write the regression equation based on centered values. We already did this above; use the simplified version in Eq. 2.

Eq 2: Y = 5.8611 + [2.4111 + (-.2000)*Train]*Dist + (-1.0467)*Train

Step 2. Plug in pairs of high and low values of Train and Dist to get 4 values. Why not choose +/- 1 quartile?

```
> summary(traincent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     -2      -2       0       0       2       2
> summary(distcent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  -1.50   -0.75    0.00    0.00    0.75    1.50

# plug in values and calculate from Eq 2 above
# I named these with a 'q' for quartile
> y11q=5.8611+(2.4111 + (-.2*2))*.75 + (-1.0467*2)
> y12q=5.8611+(2.4111 + (-.2*2))*(-.75) + (-1.0467*2)
> y21q=5.8611+(2.4111 + (-.2*(-2)))*(.75) + (-1.0467*(-2))
> y22q=5.8611+(2.4111 + (-.2*(-2)))*(-.75) + (-1.0467*(-2))
> y11q;y12q;y21q;y22q   # show results
[1] 5.276025  # High Train, High Dist
[1] 2.259375   # High Train, Low Dist
```

```
[1] 10.06283     # Low Train, High Dist
[1] 5.846175     # Low Train, Low Dist
```

Step 3. Use these points and draw the graph 'by hand' in R.

```
> x=c(.75,-.75,.75,-.75) # create x-axis values as Dist
> y=c(5.28,2.26,10.06,5.85)  # these are the calculated y predicted values

> plot(x,y,main=" +/- 1 quartile of Training, High training is bottom line",
xlab="Dist", ylab="predicted dv", ylim=c(1,10),xlim=c(-1,1))
# this makes a plot of just the points with the labels
> segments(.75,5.28,-.75,2.26); segments(.75,10.06, -.75, 5.85)
# connect pairs of points with lines. I could use different types of lines
to be fancy. I could embed a legend, which would be nice.
```