Modified Interaction Plots for GSS Data

Todd E Bodner, PhD & Joel S Steele, PhD

December 20, 2016

```r
# Things needed for Tumbleplots.
# 1) Set up the moderation regression for the outcome.
# 2) Set up the regression for the target and moderator variable
# 3) Get the mean and sd for the moderator.
# 4) Get the mean and sd for the target predictor.
# 5) Compute 1 sd above and 1 sd below mean for moderator.
# 6) Plot tumble.
# 7) Plot scatter with jitter.
# 8) Store the outcome, target, and moderator.
# 9) Store the dataframe used for the models.

library(foreign)
data <- read.spss("gssspssreduced.sav", to.data.frame=TRUE)
```

```
re-encoding from CP1252
```

```r
tumble_set = function(dat, outcome, target, moderator) {
  # full interaction model
  intrxn_formula = as.formula(paste(outcome,'~',target,"*",moderator))
  intrxn_model = lm(intrxn_formula,data=dat)
  # model between predictors, target as outcome of moderator
  inputs_formula = as.formula(paste(target,'~',moderator))
  inputs_model = lm(inputs_formula,data=dat)
  # descriptives for predictor inputs
  target_desc = c('mean'=mean(dat[,target]),
                  'sd'=sd(dat[,target]))
  moderator_desc = c('mean'=mean(dat[,moderator]),
                     'sd'=sd(dat[,moderator]))
  # computation of +/- 1 SD values for moderator
  lowmod <- moderator_desc['mean']-moderator_desc['sd']
  highmod <- moderator_desc['mean']+moderator_desc['sd']
  # target values predicted for low and high moderator
  inputs_prediction_data = data.frame(c(lowmod,highmod))
  # reuse names from the inputs model, leave out (Intercept) term
  names(inputs_prediction_data) = names(coef(inputs_model))[-1]
  predicted_targets = predict(inputs_model, inputs_prediction_data)
  # standard error of estimate for target as an outcome
  target_residual = summary(inputs_model)$sigma
  # Adjust the predicted values by adding/subtracting the se
  # low target w/ low moderator
  # high target w/ low moderator
  # low target w/ high moderator
  # high target w/ high moderator
  adjusted_predicted_targets = c(
    rep(predicted_targets,each=2)
    + rep(c(-1,1)*target_residual,times=2))
  # construct the prediction data data.frame
  intrxn_prediction_data = data.frame(
    adjusted_predicted_targets,          # target input
    c(lowmod,lowmod,highmod,highmod),    # moderator input
    c(adjusted_predicted_targets         # interaction term
      *c(lowmod,lowmod,highmod,highmod)))
  # names from the interaction model, leave out (Intercept)
  names(intrxn_prediction_data) = names(coef(intrxn_model))[-1]
  # add some factor labels to indicate the computations.
  intrxn_prediction_data = cbind(
    intrxn_prediction_data,
    'target'=rep(c('low','high'),times=2),
    'moderator'=rep(c('low','high'),each=2))
  tumble_predictions = predict(intrxn_model, intrxn_prediction_data)
  # update our data.frame
  prediction=paste(outcome,'hat',sep='_')
  intrxn_prediction_data[,prediction] = tumble_predictions
  return(list(
    'interaction_model'=intrxn_model,
    'input_model'=inputs_model,
    'tumble_data'=intrxn_prediction_data,
    'target'=target,
    'moderator'=moderator,
    'outcome'=outcome,
    'prediction'=prediction,
    'target_desc'=target_desc,
    'moderator_desc'=moderator_desc,
    'highmod'=highmod,
    'lowmod'=lowmod))
}

tumble_plot=function(tumble_dat, axis_labels=c('x','y'),legend_title=NA,plot_limits){
  # name of the column designated as the predictor of interest
  input = tumble_dat$target
  output = tumble_dat$prediction
  plot_data = tumble_dat$tumble_data
  # blank canvas set up for plotting
  if(!missing(plot_limits)){
    plot(NA,
         xlim=plot_limits[[1]],
         ylim=plot_limits[[2]],
         ann=FALSE)
  }else{
    plot(NA,
         xlim=range(plot_data[,input]),
         ylim=range(plot_data[,output]),
         ann=FALSE)
  }
  # formula for lines
  plot_formula = as.formula(paste(output,"~",input))
  # for each level of the moderator, here "high" and "low"
  lines(plot_formula,
        data=plot_data[plot_data$moderator=='high',],
        type='o')
  lines(plot_formula,
        data=plot_data[plot_data$moderator!='high',],
        type='o', lty='dashed')
  # annotations to the plot
  mtext(axis_labels[1], side=1, line=2.5, cex=1)
  mtext(axis_labels[2], side=2, line=2.5, cex=1)
  # add a legend
  legend_text=c(
    paste(round(tumble_dat$highmod,1),'(1 SD Above Mean)'),
    paste(round(tumble_dat$lowmod,1),'(1 SD Below Mean)'))
  # add a legend title if specified
  if(!is.na(legend_title)){
    legend("topleft",legend_text,cex=.8,
           lty=c("solid","dashed"),
           title=legend_title,
           bty='n',inset=.01)
  }else{legend("topleft",legend_text,cex=.8,
           lty=c("solid","dashed"),
           bty='n',inset=.01)
  }
}

# call to plot standard interaction graph
intrxn_plot = function(tumble_data,axis_labels=c('x','y'), legend_title=NA, plot_limits){
  # name of the column designated as the predictor of interest
  plot_model = tumble_data$interaction_model
  highmod = tumble_data$highmod
  lowmod = tumble_data$lowmod
  target_mean = tumble_data$target_desc['mean']
  target_sd = tumble_data$target_desc['sd']
  hightarget = target_mean + target_sd
  lowtarget = target_mean - target_sd
  # prediction setup
  prediction_data = data.frame(
    rep(c(lowtarget,hightarget), each=2),
    rep(c(lowmod,highmod), times=2),
    c(rep(c(lowtarget,hightarget), each=2)
    *rep(c(lowmod,highmod), times=2)))
  # reuse names from the model coefficients minus the (Intercept)
  names(prediction_data) = names(coef(plot_model))[-1]
  # factor labels for plotting
  prediction_data = cbind(
    prediction_data,
    'target'=rep(c('low','high'),each=2),
    'moderator'=rep(c('low','high'),times=2))
  # model predictions
  standard_predictions = predict(plot_model,prediction_data)
  prediction_data[,tumble_data$prediction] = standard_predictions
  standard_plot_data = list(
    'highmod'=tumble_data$highmod,
    'lowmod'=tumble_data$lowmod,
    'target'=tumble_data$target,
```

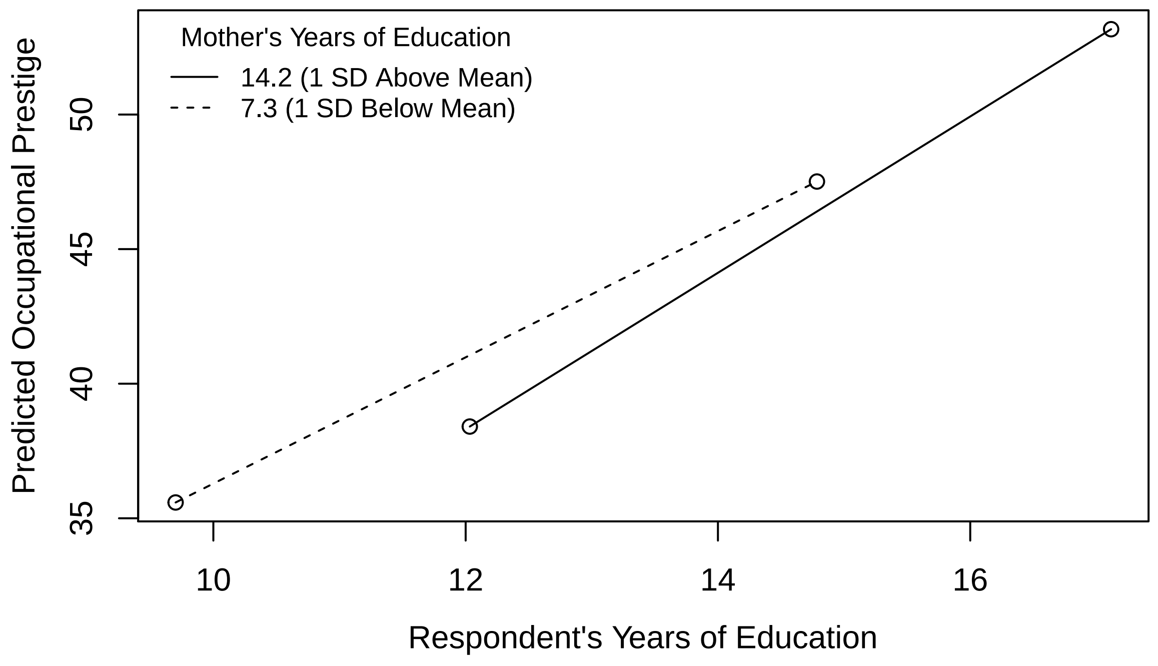
```r
    'prediction'=tumble_data$prediction,
    'tumble_data'=prediction_data)
  # call our plotting function
  tumble_plot(standard_plot_data,
            axis_labels= axis_labels,
            legend_title=legend_title,
            plot_limits = plot_limits)
}

# call to combine
compare_plot = function(tumble_data, axis_labels, legend_title, plot_limits){
  op = par(no.readonly = TRUE)
  two_plots = layout(matrix(c(1,2),ncol=1))
  par(mar=c(4,4,1,1))
  tumble_plot(tumble_data,
              axis_labels=c("",axis_labels[2]),
              legend_title = legend_title,
              plot_limits = plot_limits)
  # call standard interaction plot
  par(mar=c(4,4,1,1))
  intrxn_plot(tumble_data,axis_labels=axis_labels,
              legend_title = legend_title,
              plot_limits = plot_limits)
  show(two_plots)
  par(op)
}
```

Initialize the data

```r
# call to set up data
GSS_tumble = tumble_set(data, 'prestg80','educ','maeduc')
```

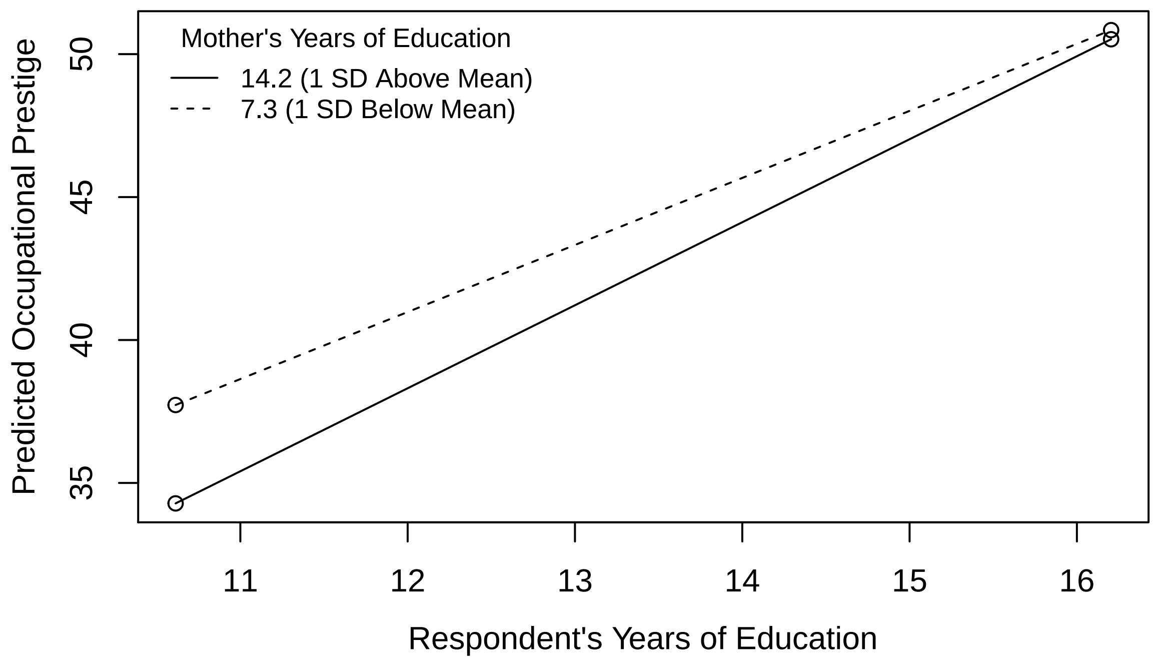
Tumble graph call

```r
# call to plot tumble plot
tumble_plot(GSS_tumble,
            axis_labels=c("Respondent's Years of Education",
                          "Predicted Occupational Prestige"),
            legend_title = "Mother's Years of Education")
```



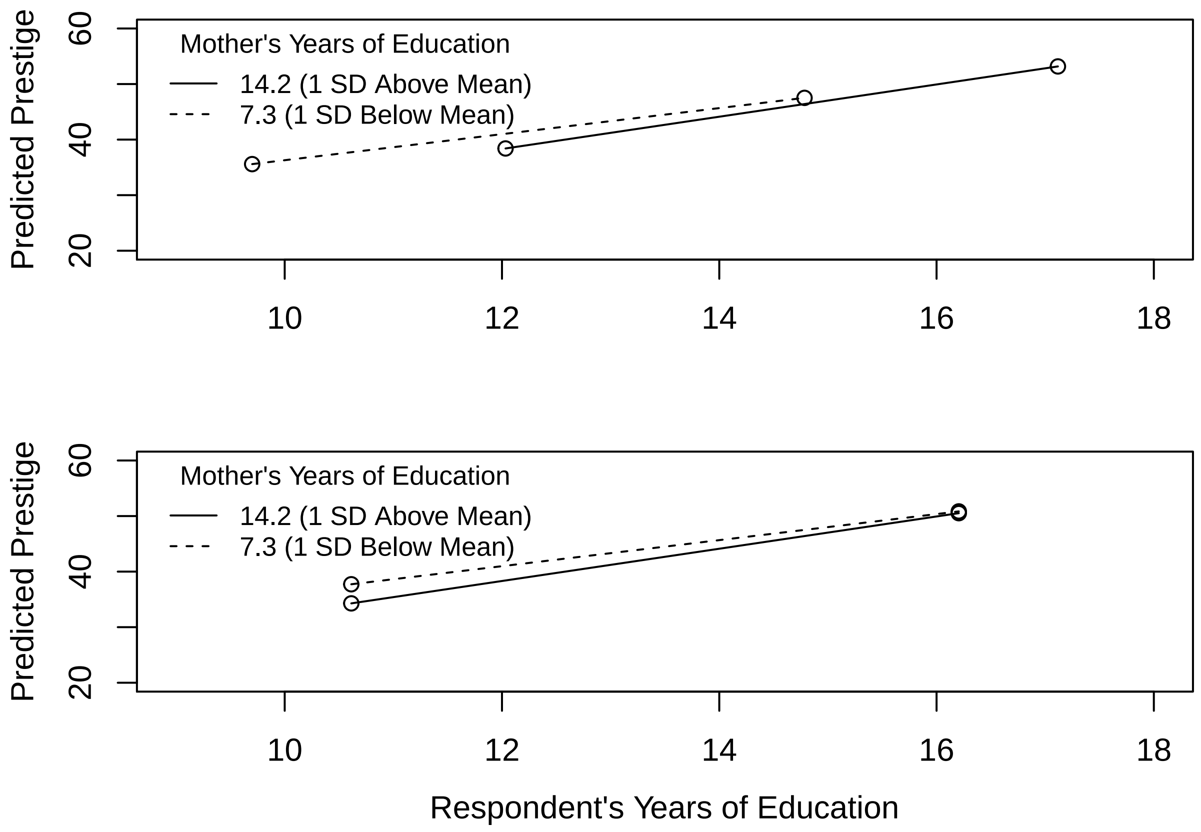
Standard interaction Graph

```r
# call standard interaction plot
intrxn_plot(GSS_tumble,axis_labels=c("Respondent's Years of Education",
                                    "Predicted Occupational Prestige"),
            legend_title = "Mother's Years of Education")
```



Both together

```
# set up axes for comparisons
plot_limits = list(c(9,18),c(20,60))
compare_plot(GSS_tumble, axis_labels=c("Respondent's Years of Education",
                           "Predicted Prestige"),
            legend_title = "Mother's Years of Education",
            plot_limits = plot_limits)
```



```
[1] 2
```